Universidad Rey Juan Carlos
Departamento de Sistemas Telemáticos y Computación (GSYC)

Tesis Doctoral

Quid Pro Quo: Mecanismos para la asignación de tareas en entornos distribuidos

D. Agustín Santos Méndez
agustinsantosmendez@gmail.com

Directores de Tesis:

Dr. D. Antonio Fernández Anta
anto@gsyc.es
Dr. D. Luis López Fernández
llopez@gsyc.es

Fuenlabrada 9 de enero de 2013

Autorizo la defensa de la tesis doctoral *Quid Pro Quo: Mecanismos para la asignación de tareas en entornos distribuidos* cuyo autor es D. Agustín Santos Méndez

El director de la tesis

Dr. D. Antonio Fernández Anta

Madrid, 9 de enero de 2013

TESIS DOCTORAL:	*Quid Pro Quo: Mecanismos para la asignación de tareas en entornos distribuidos*
AUTOR:	D. Agustín Santos Méndez
DIRECTORES DE TESIS:	Dr. D. Antonio Fernández Anta
	Dr. D. Luis López Fernández

El tribunal nombrado para juzgar la Tesis arriba indicada, compuesto por los siguientes doctores::

PRESIDENTE:

VOCAL 1:

VOCAL 2:

VOCAL 3:

SECRETARIO:

acuerda otorgarle la calificación de:

Fuenlabrada (Madrid, España), de de 2013

El secretario del comité.

D. Agustín Santos Méndez

Departamento de Sistemas Telemáticos y Computación (GSYC)

Universidad Rey Juan Carlos,

Fuenlabrada, Madrid

© 2013 D. Agustín Santos Méndez
This work is licensed under a *Creative Commons Attribution-NonCommercial 3.0 Unported License* or any later version published by the Creative Commons. You are free:

1. **to share** - to copy, distribute and transmit the work
2. **to remix** - to adapt the work

Under the following conditions:

1. **attribution** - You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
2. **share alike** - If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

Dedico este trabajo a todos los que luchan contra el egoísmo y la injusticia en el mundo.

Agradecimientos

"The great difference between voyages rests not with the ships, but with the people you meet on them."

Amelia E. Barr

Esta es una tesis que habla del egoísmo, de la mentira y del engaño. Es una tesis que trata de la injusticia y del abuso. Pero es una suerte y una gran satisfacción poder iniciar mi trabajo hablando de amistad, sinceridad, colaboración y amor. Esta tesis se ha elaborado gracias a que muchas personas de mi entorno han demostrado a diario que la ayuda, la amistad y el amor no necesitan matemáticas, sólo del corazón.

El proceso de elaboración de este trabajo ha sido largo. Pasando muchas veces por periodos aparentemente improductivos. Tengo que reconocer que no siempre se veía la luz al final del túnel. Pero he tendido la suerte de tener a mi lado a un director de tesis excepcional, D. Antonio Fernández Anta. Con paciencia infinita y dando ánimos continuamente, ha demostrado ser no sólo un gran profesional, sino también un excelente compañero y un magnifico amigo. Estoy sinceramente agradecido por todo el apoyo y cariño que me ha prestado en estos largos años.

También quiero agradecer a D. Luis López por sus consejos y su fe en este trabajo. Como codirector de la tesis, ha sufrido de los largos periodos sin resultados y de las ideas descabelladas que proponía. Siempre me animó a explorar todos esos caminos, aunque no compartiera mi punto de vista. Es imposible imaginar como hubiera resultado esta tesis sin su ayuda y consejos.

Debo agradecer a Juan Antonio Almendral la ayuda prestada en la recta final de la tesis y sobre todo con la formulación matemática de una parte de este trabajo. A él le debo una demostración que supo simplificar de forma magistral. Sin su ayuda no hubiera sabido abordarla de una forma tan intuitiva. Espero tener la ocasión de poder

seguir trabajando en todos los frentes que nos quedan por delante, sería un inmenso placer.

La tesis abre nuevos frentes de estudio que deberán ser analizados mediante simulaciones meticulosas y complejas. Es una gran satisfacción personal poder comprobar que esa línea de trabajo tiene continuación y más saber que es gracias a una persona con una enorme capacidad de trabajo y valía. Esa persona se llama Miguel Ángel López. Estoy convencido que de su trabajo y esfuerzo surgirán muchos aspectos relevantes y novedosos. Es todo un honor saber que puedo colaborar con él.

Varias personas de la Universidad Rey Juan Carlos han compartido mis inquietudes y participado de las discusiones y planteamientos iniciales. Recuerdo cientos de conversaciones con Gregorio Robles, Jesús M. González Barahona y Sergio Arévalo en las que su pragmatismo y críticas me hizo replantearme el modelo que debía construir. Además, han demostrado ser unos compañeros de trabajo insuperables desde cualquier punto de vista que se mire. Les admiro profundamente.

Pero no fueron los únicos. A lo largo de estos años he convivido con compañeros de la universidad a los que les debo mucho. Es imposible nombrar a todos, pero tengo una deuda en especial con Luis Rodero, Andrés Leonardo Martínez, Álvaro Polo y Pedro de la Heras.

Es difícil imaginar cómo hubiera podido realizar esta tesis sin la posibilidad de utilizar tiempo de mi vida profesional y personal. Tiempo que han tenido que suplir mis compañeros de la empresa en la que he trabajado todo este tiempo. Por ello quisiera agradecer a Paloma Neila y a mi familia política por todo el apoyo recibido. Y sobre todo por su comprensión en estos tiempos de crisis que nos ha tocado vivir. Quiero agradecer a mi madre todo el amor que me ha demostrado.

Pero es mi familia a la que más tiempo le he robado y a la que más le debo.

A mi mujer M. José y a mis cuatro hijos (Silvia, Jorge, Belén y Jaime) sólo puedo decirles que les quiero. Son la razón única de mi vida. No quiero despedirme sin antes dedicarles una pequeña broma. Espero que algún día, cuando sean mayores, sepan interpretarla. También las matemáticas saben hablar del corazón:

$$(2x^2 + y^2 + z^2 - 1)^3 - \frac{1}{10}x^2z^3 - y^2z^3 = 0$$

Agustín Santos Méndez

Resumen

En este trabajo proponemos una solución para la asignación de tareas en un entorno distribuido complejo y auto-organizado (sería el caso de las redes entre iguales ó P2P). Estamos interesados en las tareas que son comunes a todos los participantes o nodos del sistema. Cada uno de los nodos puede ejecutar estas tareas y, además, está interesado en que éstas se ejecuten. Cada nodo dispone de capacidad para la ejecución de cada una de las tareas. El coste para cada nodo es una información que no puede ser auditada y que es únicamente conocido por el nodo en cuestión. Suponemos que los nodos pueden mentir sobre su coste si eso les supone un beneficio; por ejemplo, por el ahorro que implicaría verse libre de ejecutar las tareas.

La presente tesis se enfrenta al problema de diseñar un sistema que permita la asignación de tareas entre nodos de forma que podamos garantizar un mínimo en el coste de ejecución, con la dificultad añadida de que los nodos son considerados egoístas y que, por lo tanto, tienen incentivos para mentir sobre el valor declarado. A estas dificultades, podemos añadir una serie de requisitos que consideramos oportunos si queremos que el sistema funcione en redes descentralizadas y anárquicas. Nos referimos a: la ausencia de nodos centrales que actúen de árbitros o controladores, la no utilización de sistemas de pago monetarios, nodos limitados total o parcialmente en su racionalidad, reparto justo de trabajo, limitada degradación del rendimiento ante agentes egoístas, el aprendizaje de los nodos sobre el comportamiento de los demás no debe aportar ventajas estratégicas, los nodos deben buscar la máxima agregación y no tener interés es crear grupos separados de trabajo y, finalmente, que la complejidad del algoritmo sea abordable desde un punto práctico.

En nuestro trabajo proponemos un modelo matemático y un conjunto de soluciones que resuelven el problema de la asignación de tareas de forma óptima sin necesidad de pagos entre nodos y que responde a los requisitos expuestos. Se proponen varios algoritmos que son analizados con las herramientas matemáticas de la “teoría de juegos” y, en concreto, del “diseño de mecanismos”. A estas soluciones las hemos denominados mecanismos “Quid Pro Quo” (QPQ). Esta expresión se suele utilizar cuando alguien realiza un trabajo o favor y espera en compensación un trabajo o fa-

vor equivalente. Presentamos varios mecanismos, correspondiendo cada una de ellos a un determinado nivel de correlación entre las valoraciones de los jugadores o a una diferente noción de justicia.

El primer mecanismo, denominado *QPQ Básico*, aporta una solución al caso en el que los jugadores tienen valoraciones de coste independientes. En la presente tesis se demuestra que este primer mecanismo tiene todas las propiedades buscadas. Además, se han realizado simulaciones que comprueban la aplicación de estas propiedades en diferentes escenarios.

La segunda familia de mecanismos se denomina *QPQ Correlados*, ya que el modelo previsto supone que los jugadores pueden tener cierta correlación en las distribuciones de los valores declarados. De nuevo se demuestra que la mayor parte de las propiedades se siguen manteniendo vigentes, aunque algunos conceptos han tenido que ser redefinidos (tales como justicia o eficiencia).

Finalmente, se aporta un esquema para la construcción de mecanismos QPQ basados en mecanismos de Groves. Nuestra propuesta incluye el estudio de las relaciones que pueden existir entre los mecanismos tradicionales de pagos y los mecanismos QPQ. Hemos denominado *QPQ VCG o de Groves* a este tipo de mecanismos. Aunque en realidad es un caso particular de los modelos anteriores, creemos que tiene especial interés por su relación con los mecanismos de pago.

Índice general

1. Introducción. **1**

1.1. Antecedentes de la investigación 1
1.2. Planteamiento General del Problema 6
1.3. Objetivos e hipótesis de trabajo 12
1.4. Metodología 13
1.5. Estado del arte 14
1.6. Sobre la notación utilizada 16
1.7. Estructura de la tesis 18

2. Teoría de Juegos y Diseño de Mecanismos **21**

2.1. Teoría de Juegos 21
2.1.1. Introducción 22
2.1.2. Jugadores, preferencias y función de utilidad 24
2.1.3. Estrategia 30
2.1.4. Clasificación de los juegos y tipos de solución 33
2.2. Diseño de Mecanismos 38
2.2.1. Introducción 39
2.2.2. Conceptos básicos 41
2.2.3. Desiderata. 50
2.2.4. Otros conceptos relacionados. 53
2.2.5. Otra notación relevante. 55
2.2.6. Cuestiones. 56

3. Formalización del modelo y soluciones. **59**
3.1. Introducción 59
3.2. Desarrollo del modelo 66
3.2.1. Algoritmos que no son strategy-proof 68
3.3. Mecanismos con pagos 73
3.3.1. Dos Jugadores a una ronda. 73
3.3.2. N jugadores-una ronda. 77
3.3.3. Mecanismos VCG con reparto de pagos. 80
3.4. Conclusiones del capítulo 91

4. Algoritmo Quid Pro Quo básico. **95**
4.1. Introducción 95
4.2. Mecanismos de Jackson y Sonnenschein 96
4.3. Aplicación de los mecanismos de enlace 100
4.3.1. Transformación de la función de utilidad 101
4.3.2. Aceptación de valores 109
4.4. Algoritmo Quid Pro Quo (QPQ) Básico 112
4.4.1. Aspectos formales del algoritmo 114
4.5. Aspectos prácticos de QPQ 127
4.5.1. Generación del valor alternativo 128
4.5.2. Aplicación práctica del test de bondad 128
4.5.3. Formulación práctica de QPQ 134
4.6. Resultados experimentales 135

5. QPQ - Jugadores Correlados **151**
5.1. Introducción 151
5.2. Sobre la noción de justicia 158
5.3. Sobre la función de decisión 161
5.4. Primeros aspectos formales 165
5.5. Función de corte 173
5.6. Caso 1: Justicia en el número de tareas 175

5.6.1. QPQ Básico como caso particular . . . 186
5.6.2. Sobre el cálculo de λ . . . 187
5.6.3. Algoritmo QPQ con justicia en volumen de tareas . . . 190
5.6.4. Propiedades . . . 190
5.6.5. Comentarios finales . . . 195
5.7. Caso 2: Justicia en el esfuerzo . . . 197
5.7.1. Sobre el nuevo cálculo de lambda . . . 199
5.7.2. Algoritmo QPQ con justicia en costes . . . 201
5.7.3. Propiedades . . . 201

6. Otros aspectos de QPQ y los mecanismos VCG 205
6.1. Introducción . . . 205
6.2. Modificación de la correlación . . . 206
6.2.1. Modificación correlación en QPQ Básico . . . 208
6.2.2. QPQ con justicia de tareas . . . 209
6.2.3. QPQ con justicia en coste . . . 214
6.2.4. Relación entre trabajo y esfuerzo . . . 217
6.2.5. Fijando la utilidad del contrario . . . 219
6.3. Un nuevo concepto de justicia en QPQ . . . 222
6.3.1. Análisis y exploración de la solución . . . 223
6.3.2. Sobre el cálculo de la función de corte . . . 232
6.4. Número de jugadores . . . 233
6.4.1. Número de jugadores variable en el tiempo . . . 234
6.4.2. Elevado número de jugadores . . . 235
6.4.3. Número de jugadores virtuales . . . 236

7. Conclusiones y trabajos futuros 249
7.1. Conclusiones . . . 249
7.2. Validación o refutación de las hipótesis . . . 253
7.3. Investigación futura y problemas abiertos . . . 255
7.3.1. Implementación de QPQ . . . 256

7.3.2. Elementos teóricos 258
7.3.3. Nuevos modelos 260
7.3.4. Comportamiento dinámico de QPQ 261

Índice de figuras

2.1. Equilibrio de Nash 37
2.2. Mecanismo: Diagrama de Stanley Reiter 47
2.3. Mecanismo 49

4.1. Ejemplo 1 de distribución de coste. 102
4.2. Ejemplo 2 de distribución de coste. 102
4.3. Ejemplo de distribución. 103
4.4. Ejemplo de distribución escalada. 103
4.5. **Ejemplo transformada PIT (I)**. El jugador declara un valor de $0{,}8$ pero el sistema trabajrá con un valor normalizado cercano a $0{,}6$. . . . 107
4.6. **Ejemplo transformada PIT (II)**. El jugador declara un valor de 2 pero el sistema trabajrá con un valor normalizado cercano a $0{,}5$. . . . 108
4.7. Umbral de aceptación a varias rondas. Según se incrementa el número de rondas, el sistema dispone de más información y por tanto tiende a creer más a los jugadores. 133
4.8. Dos jugadores honestos sin control (100 rondas). 138
4.9. Dos jugadores honestos sin control (100 rondas). 139
4.10. Dos jugadores honestos con control (100 rondas). 140
4.11. Dos jugadores honestos con control (100 rondas). 141
4.12. Dos jugadores honestos sin control (1.000 rondas). 142
4.13. Dos jugadores honestos sin control (1.000 rondas). 143
4.14. Dos jugadores honestos con control (1.000 rondas). 144
4.15. Dos jug. honestos con control (1.000 rondas). 145

4.16. Comparativa de tres distribuciones similares (100 muestras). 146
4.17. Dos jugadores (sincero vs mentiroso Beta-9) sin control I. 147
4.18. Dos jugadores (sincero vs mentiroso Beta-9) sin control II. 148
4.19. Dos jugadores (sincero vs mentiroso Beta-9) con control (100 rondas) I. 149
4.20. Dos jugadores (sincero vs mentiroso Beta-9) con control (100 rondas) II. 150

5.1. **Visión de QPQ Básico**. La versión básica de QPQ no distingue entre los 3 jugadores. Todos los participantes tienen distribuciones uniformes. 152
5.2. Los dos primeros jugadores están correlados. 153
5.3. Esfuerzo y trabajo de los tres jugadores. 154
5.4. Los dos primeros jugadores están correlados (II). 155
5.5. Esfuerzo y trabajo de los tres jugadores (II). 156
5.6. Varios ejemplos de correlación entre jugadores. 157
5.7. Función de decisión en QPQ básico. 166
5.8. Función de decisión con áreas aleatorias. 169
5.9. Función de decisión sin áreas aleatorias. 170
5.10. Función de decisión con áreas aisladas. 171
5.11. Desplazamiento de las áreas. 172
5.12. Función de decisión con áreas separadas. 173
5.13. Ejemplo no real de función de corte. 174
5.14. Ejemplo real de función de corte. 179
5.15. Reparto de áreas de decisión. 183
5.16. Reparto de áreas de decisión (II). 186
5.17. Volumen trabajo en función de λ. 188

6.1. Estrategia de desplazamiento de la probabilidad. 207
6.2. 2 Jugadores correlados. 209
6.3. 2 Jugadores correlados, Vista en 3D. 210
6.4. Estrategia de desplazamiento de la probabilidad (II). 211
6.5. Contraejemplo 1, vista 3D. 212

6.6. Contraejemplo 1, vista de contorno. 213
6.7. Jugador 1 mintiendo, vista 3D. 214
6.8. Jugador 1 mintiendo, vista de contorno. 215
6.9. Jugador 2 mintiendo, vista 3D. 216
6.10. Jugador 2 mintiendo, vista de contorno. 217
6.11. Imposibilidad de fijar la probabilidad, vista 3D. 221
6.12. Jugadores correlados. 239

Capítulo 1

Introducción.

"Coming together is a beginning. Keeping together is progress. Working together is success."

Henry Ford

1.1. Antecedentes de la investigación

Internet ha evolucionado dramáticamente en los últimos años; no sólo debido a las aplicaciones Web más habituales, sino por todas las tecnologías que emergen día a día y que permiten construir plataformas descentralizadas cada vez más interesantes. Ejemplos de tales aplicaciones son los juegos multijugador en 3D y los mundos virtuales online. Todos ellos permiten una interacción social inimaginable apenas hace unos años y que en cambio hoy en día no sorprende a nadie. Internet aporta oportunidades únicas para la creación de estos "espacios virtuales", entornos abstractos que funcionan y se simulan en una red distribuida de ordenadores, algunas veces sin interacción humana.

Imagine un sistema similar a "Second Life" [Lab11] pero que funcione bajo una arquitectura sin organización centralizada y entre iguales (P2P). Aplicaciones similares podrían ser viables en breve: juegos persistentes no centralizados, sistemas de

computación o de backup compartido, sistemas de entrenamiento aerospacial o militar, simulaciones complejas de aplicación en meteorología, química, médica, mantenimiento de repositorios de software distribuido (junto con las tareas de compilación o pruebas asociadas), etc.

El problema planteado en la presente tesis emerge como punto clave cuando se estudian los requisitos y algoritmos que podrían surgir en la construcción de tales entornos virtuales.

Independientemente de la aplicación, en dichos sistemas existirán tareas que deben ser realizadas y que benefician a la totalidad de los participantes, tales como la simulación del entorno (lluvia, agua, viento, etc.) o de organización interna (validación de usuarios, encaminamiento de los mensajes o eventos, gestión de las búsquedas, etc). En realidad no es difícil imaginar sistemas distribuidos en los que todos o algunos de los participantes en ese sistema estarán interesados en que las tareas se realicen. No debe sorprender que el mismo problema pueda aparecer también fuera del contexto informático. Situaciones similares a las estudiadas en la presente tesis suceden a diario en grupos humanos de trabajo: convivencia familiar o de amigos, reparto de trabajo en empresas, juegos infantiles, etc.

En nuestro trabajo proponemos una solución para la asignación y ejecución de tareas en un entorno distribuido complejo y sin una organización centralizada. Cada uno de los nodos o partícipes de ese sistema distribuido dispone de una capacidad para la ejecución de cada una de las tareas. El hecho de realizar una tarea hace que el nodo consuma parte de esa capacidad y por lo tanto le suponga un cierto esfuerzo o coste. Se desea que la asignación de tareas tenga en cuenta dicho esfuerzo de forma que el coste final sea el mínimo posible. Se ha considerado que el coste para cada nodo es una información que no puede ser auditada y que es únicamente conocida por el nodo en cuestión. Suponemos también que los nodos pueden mentir sobre su coste si eso les supone un ahorro al evitar ejecutar las tareas asignadas. Como más adelante se analiza, nuestro algoritmo permite la asignación de tareas a los nodos de forma óptima, sin permitir que los nodos mentirosos o egoístas puedan degradar el sistema.

Es cierto que son múltiples los trabajos relacionados con la asignación de tareas, pero estamos interesados en el problema de asignación de tareas bajo un aspecto clave: **el egoísmo de los participantes**.

Richard Dawkins, en su libro "El gen egoísta" expresa la siguiente opinión: "Siempre que se desarrolla un sistema de comunicación existe el peligro constante de que alguien explote el sistema para sus propios fines."[Daw76]. Aunque la frase se refiere al comportamiento animal, no deja de ser inquietante que siga siendo cierta para tantos aspectos de nuestra vida cotidiana, y en particular para los sistemas informáticos distribuidos.

Recientemente, las redes entre iguales ("peer-to-peer" o "P2P") han puesto de manifiesto un tipo de comportamiento egoísta ("selfish") que hasta la fecha no se había contemplado en el diseño de sistemas o aplicaciones distribuidas complejas. Tradicionalmente, los diseñadores de tales aplicaciones o protocolos reflejaban únicamente los comportamientos normales y, en ciertos casos, aquellos que derivan de ciertos tipos de fallos (aleatorios, bizantinos, etc.). Aunque la aplicación de nuestro trabajo no se limita a este tipo de redes, es cierto que éstas tienen muchas de las características que hemos considerado interesantes de estudio: no suelen tener agentes centrales, no existen acuerdos monetarios, son propensas al egoísmo, etc.

En los últimos años, la comunidad informática ha propuesto la utilización de técnicas matemáticas basadas en la *Teoría de Juegos* (en inglés "Game Theory") para el modelado de comportamientos egoístas en el diseño de sistemas distribuidos.

El *Diseño de Mecanismos* ("Mechanism Design") es una rama de las matemáticas relacionada con la teoría de juegos, y aporta la base matemática necesaria para el estudio y diseño de sistemas distribuidos bajo el comportamiento de nodos o participantes egoístas. Estas matemáticas nacen en el contexto de la economía e intentan dar respuesta a problemas relacionados con subastas de bienes, reparto de propiedades en un divorcio, adjudicación de contratos, y otros problemas similares muy frecuentes en dicho ámbito. En informática ya es frecuente encontrar trabajos y estudios basados en estas matemáticas. Sería inviable citar apenas una mínima parte de todos estos trabajos y, sin ánimo de ser exhaustivo, una pequeña muestra sería la siguiente:

[DJP03, FPSS02, NGNP09, AE03, Pap01, NR99, FPS01, FS02, FPSS02]).

Muchos de los trabajos publicados en informática hasta la fecha, al menos de los conocidos por el autor, han utilizando premisas heredadas de la teoría económica. Nos referimos en concreto a tres premisas:

1. la existencia de un nodo central o planificador (llamado en economía "principal"),

2. a la existencia de una moneda común que permite realizar pagos entre nodos,

3. que la relación entre los nodos se reduce a una única intervención.

Estas premisas pueden ser consideradas evidentes en el contexto económico pero, según nuestra opinión y la de algunos autores [BFB06], son difícilmente aplicables en el contexto que nos ocupa. Veamos en detalle cada una de estas premisas.

En primer lugar, la mayoría de los mecanismos tradicionales proponen la existencia de un nodo que actúa de juez o planificador central. Esta figura tiene sentido en problemas de subastas (un agente realiza y coordina la subasta mientras que otros participan en la misma ofertando precios), en los divorcios (un juez reparte los bienes de la pareja), o en los mecanismos de contratación (el estado contrata y varias empresas ofrecen sus servicios). En el ámbito informático, este rol también ha sido muy utilizado. La mayoría de los mecanismos suponen un nodo central que recibe toda la información, calcula la decisión y la comunica a los participantes. En algunos trabajos recientes ya se contempla la posibilidad de que no exista un agente con un papel o rol diferente a los demás. Aunque en las redes entre iguales existen nodos que realizan trabajos especiales (por ejemplo, mantenimiento de las tablas *hash distribuidas* y las búsquedas), en general todos los nodos se consideran iguales entre ellos. En nuestro modelo, deseamos que la decisión de asignación de la tarea sea completamente distribuida.

Quizás uno de los aspectos más controvertidos de las soluciones propuestas hasta la fecha es la existencia de un pago entre nodos. Casi la totalidad de mecanismos propuestos suponen, lo que en terminología económica se denomina, *transferencia*

de utilidades. Esto lo que significa es que al aplicar el mecanismo es posible transferir parte del beneficio (utilidad) de un agente a otro mediante un pago. Para que ello sea posible, tenemos que presuponer principalmente dos premisas. En primer lugar que todos los nodos miden su coste y/o beneficio en la misma unidad (monetaria o relativa a conceptos como capacidad de disco, velocidad de la red, tiempos, etc.). En segundo lugar, se supone que los nodos tienen una utilidad resultante de la suma de su beneficio más la transferencia (positiva o negativa) que el mecanismo propone. Esta característica se denomina *utilidad casi-lineal* ("quasi-linear utility"). Creemos que estos conceptos son difícilmente aplicables en un contexto de aplicaciones distribuidas en redes descentralizadas. Uno de los objetivos más importantes de nuestro algoritmo es la de encontrar una solución en la que se relajen ambas premisas. Es decir, no presuponemos que las unidades de coste sean comunes entre los nodos y, por consiguiente, ya que no existe una moneda común, no existe un mecanismo de pago entre nodos.

Otro aspecto que hemos considerado en el diseño de nuestro modelo es que los nodos interaccionan entre ellos de forma repetida. Desde hace tiempo se sabe que la solución de una interacción única (un juego simple o no repetido) no tiene que ser la misma que pudiera existir si la interacción se repite en el tiempo. Algunos juegos simples como el juego del prisionero han demostrado este hecho. También se sabe que, en una interacción repetida en el tiempo, si los jugadores son suficientemente pacientes, es posible encontrar o forzar otras soluciones. Esta observación está relacionada con lo que, en inglés, se conoce como el "folk theorem". [1]

El concepto de paciencia está relacionado con la intensidad que el jugador le da a los beneficios actuales frente a los beneficios futuros. Cuando un jugador es paciente, se entiende que puede sacrificar beneficios presentes por otros futuros. Esa intensidad se suele modelar con el denominado "factor de descuento" y afecta a la estrategia del jugador.

La razón de que una solución para un juego no repetido no sea válida para juegos

[1] El nombre de este concepto procede del hecho de que era conocido antes de que se formulara por escrito. Es un concepto interesante y el lector puede consultar, entre otros, a [AS92, BK98, MMS08].

repetidos reside en el hecho de que los jugadores pueden seguir estrategias de premio o castigo que no son aplicables a una sola ronda. Además, los nodos pueden aprender de los demás, por ejemplo adquiriendo cierto conocimiento sobre las preferencias de los contrincantes (se establece un conocimiento probabilista o bayesiano). Mailath y Samuelson cubren muchos de estos conceptos en su libro [MS06]. En nuestro trabajo hemos intentado encontrar una solución que sea aplicable en un modelo de interacciones repetidas en el tiempo.

Según nuestro conocimiento, el problema que proponemos a estudio es el primero que se aborda bajo estos condicionantes. A continuación se describe en más detalle el contexto y las particularidades del problema objeto de la presente tesis.

1.2. Planteamiento General del Problema

La mayoría de los sistemas distribuidos en redes anárquicas (tipo P2P) se basan en la ejecución de una serie de tareas básicas, que van desde el mantenimiento de las tablas de búsqueda hasta el almacenamiento y gestión de información común (por ejemplo, la información de la reputación de cada nodo). Muchas de estas tareas son necesarias para el correcto mantenimiento y funcionamiento del sistema. Aunque todos los nodos pueden entender que estas tareas son vitales para ellos, es habitual delegar su ejecución en los demás, llegando al caso de que un porcentaje bajo de los nodos en una red asuman el mayor peso en la ejecución de dichas tareas. Este funcionamiento puede ocasionar problemas de rendimiento y fiabilidad, si algunos de estos nodos se saturan o fallan. Sería pues conveniente que cada nodo pudiera asumir parte de la responsabilidad en la ejecución de estas tareas.

Para ello, proponemos utilizar un concepto de capacidad u oportunidad en la ejecución de las tareas. Suponemos que cada nodo puede tener una cierta capacidad puntual para la ejecución de una tarea básica. Esta capacidad puede variar con el tiempo y con las tareas a realizar. Por ejemplo, un nodo puede tener mucho disco libre en un momento dado y por el contrario tener ocupada la CPU al máximo. En ese instante, el nodo puede tener mayor capacidad para ejecutar tareas que impliquen

almacenamiento de información, mientras que un instante más tarde cambiar su situación y optar a tareas en las que el cálculo sea relevante. Si deseamos repartir la carga de estas tareas básicas entre los nodos, sería deseable tener este aspecto en cuenta. Esta oportunidad o capacidad puntual de ejecución también se puede observar en su forma negativa, a la que denominaremos "coste". Es decir, suponemos que cada nodo tiene un coste para cada tarea. Este coste puede variar de una tarea a otra (incluso cuando la tarea es la misma, pero en un instante posterior).

Con tal planteamiento del problema, surge una serie de factores determinantes para alcanzar una solución adecuada. Por una parte, el coste de cada nodo para cada tarea es algo que únicamente conoce el propio nodo. Estamos suponiendo que es difícil auditar o comprobar si un nodo tiene más o menos capacidad para la ejecución de una tarea concreta. Este concepto se denomina en la teoría de juegos como *información privada*. Aunque podamos pensar en alternativas para conocer esta información, una forma de conocer los costes de cada nodo es preguntar a los nodos directamente, y esperar que declaren correctamente dicho valor. Y aquí es donde surge el siguiente aspecto que debemos tener en cuenta. Cada participante puede elegir entre seguir o no el protocolo establecido, si con ello puede evitarse realizar ciertos trabajos. En definitiva, cada nodo es un ordenador que pertenece a una persona que puede alterar el protocolo o el código para su propio beneficio, es decir se aprovechan de los beneficios del sistema sin necesidad de colaborar en el mantenimiento del mismo. Este comportamiento, ciertamente egoísta, es uno de los factores que más pueden alterar un protocolo o aplicación distribuida. Como veremos más adelante, la pérdida de rendimiento por culpa de nodos egoístas es un parámetro muy a tener en cuenta y que se denomina *precio de la anarquía* ("price of anarchy" [2]).

Nos enfrentamos pues, al problema de diseñar un sistema que permita la asignación de tareas entre nodos de forma que el sistema sea eficiente en la ejecución del conjunto de tareas, con la dificultad añadida de que el coste que declara cada nodo no

[2]Este concepto fue definido inicialmente por Koutsoupias y Papadimitriou [KP09, Pap11]. También se le conoce como el *ratio de coordinación* o "coordination ratio". El libro de Roughgarden [Rou05] es un ejemplo de cómo aplicar estos conceptos al enrutamiento de redes.

puede ser auditado, y además los nodos tienen incentivos para mentir sobre el valor declarado. A estas dificultades, podemos añadir una serie de requisitos que consideramos oportunos si queremos que el sistema funcione en situaciones similares a las descritas (redes descentralizadas y anárquicas tipo "P2P").

La mayoría de las soluciones propuestas para algoritmos con agentes egoístas presuponen una serie de principios que, dependiendo del contexto, pueden ser considerados muy simplistas. Bauer, Faratin y Berly en su artículo [BFB06] reflexionan sobre la aplicación de estos principios en el contexto de los sistemas telemáticos. En su análisis estudian las hipótesis de algunos trabajos bastante conocidos como [AE03], [FPSS02] y [FS02]. Aunque no son los únicos, también en [Ami] parten de estas hipótesis y aplican mecanismos clásicos.

En concreto, Bauer, Faratin y Berly abordan dos puntos que consideran relevantes:

1. El diseñador de los algoritmos suele tener cierto conocimiento sobre las preferencias de los nodos. En concreto, se suele suponer un determinado modelo de beneficio/coste que se acompaña con un sistema de pagos.

2. El diseñador de los algoritmos supone que la interacción de los jugadores se limita a una única ronda. Es conocido que una solución para una única ronda no tiene que funcionar cuando el juego es repetido.

En nuestro trabajo, hemos querido profundizar en dicha reflexión y abordar el problema de la ejecución de tareas con ella en mente. Veamos cada uno de estos requisitos en detalle.

Si entendemos el coste de una tarea para cada nodo, no tanto como una cantidad monetaria, sino como algo más general (p.e la utilización de recursos internos, la pérdida de una oportunidad o la reducción de una capacidad que dicho nodo tiene para la ejecución de la misma), es razonable pensar que cada uno medirá dicho parámetro según sus propios criterios. En cierta forma, el coste mide el interés y esfuerzo de colaboración que tiene un nodo. Por ejemplo, un nodo puede decidir asumir con mayor o menor facilidad sus tareas según la ocupación de su propia CPU. Por el contrario,

otro nodo puede pensar que su grado de colaboración depende del espacio disponible en el disco. En realidad, el número de factores que pueden influir en el grado de colaboración es tan amplio como queramos (desde parámetros físicos medibles por el nodo tales como ancho de banda, grado de ocupación de la CPU, disco, etc. hasta parámetros menos informáticos como impacto de publicidad hipotética, etc.). Incluso pueden ser combinaciones de todos ellos. Nuestro objetivo es tener en cuenta este hecho. Lo más importante, es que estos razonamientos nos llevan irremediablemente a la hipótesis de que no es posible transformar un coste en otro simplemente suponiendo una "moneda de cambio" entre costes. Por ello queremos decir que no existe una transformación evidente entre, por ejemplo, el "grado en el impacto de la publicidad" que declara un nodo y la "ocupación del disco modificada por la ocupación de la CPU" que puede declarar otro. Podemos pensar que permitir distintas unidades de coste es un requisito muy fuerte, y de hecho una gran parte de los trabajos hasta ahora publicados suponen que el coste puede ser medido mediante una cierta moneda.

La siguiente consecuencia de nuestro planteamiento es que es difícil plantear pagos entre nodos cuando no existen marcos contractuales externos al sistema. La utilización de pagos entre nodos es un mecanismo habitual para incentivar a los jugadores a declarar el coste real. Para entender las razones de tales incentivos, pensemos en un posible ejemplo hipotético. Imaginemos que una comunidad de vecinos desea saber si es o no conveniente la instalación de una antena parabólica. Para ello, los responsables preguntan a todos los vecinos si desean instalar la antena, sabiendo que el coste de la instalación es C. Si al final la comunidad de vecinos opta por instalar dicha antena, cada vecino deberá asumir su parte en el coste (por ejemplo C/n, donde n es el número de vecinos). En este ejemplo, si un vecino "valora" la instalación de la antena por debajo de C/n declarará que no desea instalar la antena, mientras que si un vecino cree que C/n es un coste razonable declarará que desea la antena. Este problema, tal y como está planteado, puede llevar a conclusiones erróneas. Por ejemplo, si hay un único vecino que desea la antena, pero que estaría dispuesto a correr con la mayor parte del coste, es posible que simplemente por la opinión de la mayoría que está dispuesto a pagar menos de C/n, se quedará sin la antena. Una

alternativa es preguntar cuánto estaría dispuesto a pagar cada uno de los vecinos por tener la antena. Eso nos llevaría también a una solución equivocada. En este caso, los vecinos que valoren el proyecto por debajo de C/n declararan un valor de 0, mientras que los que valoran mucho el proyecto declararán un valor de C. Esto es así ya que todos deberán pagar lo mismo si la antena se instala. Por lo tanto, la solución pasa por obligar a pagar más cantidad a aquellos que deseen la antena y a subvencionar a aquellos que valoran menos el proyecto. Con esta solución lo que hemos provocado es el establecimiento de un pago entre vecinos.

Pero justo éste es el mecanismo de incentivos que no queremos aplicar en nuestro algoritmo. Es razonable usar este tipo de mecanismos en el ejemplo anterior, ya que todos los nodos manejan la misma moneda (euro, dólar, etc.) y además, podemos entender que un vecino pueda pagar o cobrar una cantidad ya que existe un contexto legal que lo permite y controla (comunidad de vecinos, ayuntamiento, etc.). En nuestro caso es dudoso que podamos aplicar estas premisas. Si un nodo mide su grado de colaboración en una unidad que él entiende relacionada con la publicidad, difícilmente le puede "pagar" a otro nodo que mide sus costes en unidades de CPU. Estos pagos no son posibles principalmente debido a que, por un lado no existe una "moneda de cambio", pero por otro lado, no siempre podremos expresar el beneficio de un nodo simplemente como la suma del coste de ejecutar la tarea más un pago. Incluso si suponemos que existe una moneda de cambio, este beneficio podría ser resultado de aplicar cualquier función al coste y al pago *-no necesariamente la suma lineal-*. Evidentemente, en el contexto económico es habitual entender esta función como una adición. Veremos como formalizar este concepto matemáticamente, y como afecta a la construcción de los mecanismos tradicionales en el siguiente apartado.

Sigamos con nuestro análisis sobre qué hipótesis son razonables en los modelos actuales. El siguiente paso es pensar que todos los nodos son racionales y que conocen perfectamente su coste. Quizás es uno de los puntos más criticados de la teoría de juegos, y quizás de los menos estudiados cuando abordamos el diseño de tales sistemas. Este aspecto tiene diversas interpretaciones, según lo que podamos entender por "inteligente" o "racional". En la teoría de juegos se suelen distinguir varios grados

o niveles. Por una parte, se reconoce que un jugador es capaz de valorar las alternativas y establecer un orden de preferencia sobre éstas (se dice entonces que existe un orden de preferencia racional). Además, el jugador debe tener una capacidad de conocimiento tanto de las reglas del juego como de lo que otros jugadores pueden conocer ("conocimiento completo"). Finalmente, a un jugador se le reconoce cierta capacidad para razonar sobre la mejor estrategia a seguir y actuar en consecuencia (el jugador es "racional").

En general, se supone que los nodos tienen un conocimiento completo de su coste. En muchos de los trabajos sobre los incentivos al intercambio de ficheros, es razonable suponer que los nodos conocen perfectamente el coste, ya que es una función del tamaño del fichero, o del ancho de banda. En nuestro contexto, el coste puede ser algo difícil de estimar (por ejemplo, el impacto de una cierta publicidad). Por otra parte, se suele pensar que los nodos actúan de forma completamente racional. Es decir, se supone que son capaces de calcular matemáticamente la mejor solución o estrategia y optar por ella. Son numerosos los trabajos que contemplan la dificultad que ello conlleva, por ejemplo debido a que ciertos problemas de optimización tienen cierta complejidad intratable o demasiado costosa en tiempo de computación [3]. Estos agentes se consideran racionalmente limitados. Aunque este punto es difícil de abordar de forma general, en nuestro trabajo hemos intentado reflejar el hecho de que los nodos pueden no ser capaces de conocer correctamente su coste, o no son completamente racionales. y por lo tanto pueden optar por soluciones no óptimas. Es decir, en nuestro trabajo se ha intentado reducir al mínimo los requisitos de inteligencia y racionalidad.

Esta reflexión nos lleva a dos nuevos requisitos de trabajo. Por una parte, la solución que apliquemos deberá permitir que nodos no racionales o racionalmente limitados no perjudiquen el rendimiento de los demás, o al menos no perjudiquen a los jugadores o nodos que sean completamente racionales. Un segundo requisito es que

[3] La literatura sobre la violación de racionalidad es muy amplia y algunos trabajos ya incorporan modelos de racionalidad limitada. Sin ser exhaustivos, el lector puede consultar [Sim56, AR88, BT01, OR98, PY94, CI05, CR02, CH01, RLEM94].

todos los nodos deberán sacar beneficio del sistema, incluso si no son completamente racionales. Posiblemente no extraigan todo el beneficio posible, pero al menos no deberíamos penalizar a dichos agentes. La combinación de ambos requisitos nos lleva a que deberíamos garantizar un mínimo beneficio a todos los partícipes en el sistema, tanto si son racionales como si no.

En nuestra opinión, todos estos requisitos deberían ser estudiados bajo un cierto criterio de justicia. Nuestro trabajo debe permitirnos encontrar un óptimo en la asignación de las tareas, pero siempre dentro del contexto de un reparto ”justo”. La teoría de juegos distingue muchos y muy variados conceptos de justicia. No es difícil ver que nos enfrentamos a una noción muy esquiva. Por justo podemos entender que el reparto se realice igualando el número de tareas. También se puede razonar que la justicia se alcanza cuando los jugadores realizan esfuerzos similares. El presente trabajo debe explorar la búsqueda de una solución que nos permita alcanzar una o varias de estas nociones de justicia y, de ser posible, demostrar las condiciones en las que son viables o no.

Resumiendo, lo que nuestro trabajo propone es el diseño de un algoritmo o protocolo que nos permita asignar tareas dentro de un conjunto de nodos. Cada nodo tiene una capacidad o interés en ejecutar cada una de las tareas, pero esta información es privada y no auditable. Deseamos que la ejecución de tareas se realice de forma óptima sin necesidad de pagos entre nodos, sin la participación de un agente mediador y que cumpla con un mínimo de requisitos tales como alcanzar cierta justicia o garantizar un máximo de trabajo a cada nodo.

1.3. Objetivos e hipótesis de trabajo

Se puede formular lo dicho hasta aquí diciendo que tenemos dos objetivos en esta tesis. El primer objetivo es construir algoritmos que permitan asignar tareas en un sistema distribuido y sin organización centralizada, teniendo en consideración las preferencias de los participantes y que sea resistente a las manipulaciones o mentiras de los nodos. Estos algoritmos deberán permitir, bajo las restricciones oportunas, una

eficiencia en la ejecución de las tareas. Adicionalmente, se desea que dicho algoritmo se pueda aplicar en situaciones de interacción repetida y en las que no sea posible el uso de moneda.

Pero detrás de este primer objetivo (explícito) subyace otro objetivo que consiste en formalizar un modelo de tal sistema, y estudiar matemáticamente las propiedades y limitaciones de tales algoritmos. En base a tales conclusiones, un objetivo colateral es estudiar si existe alguna limitación en el modelo y plantear alternativas para estudios posteriores.

Las hipótesis de trabajo que se proponen en esta tesis son las siguientes:

Hipótesis 1: Es posible formalizar matemáticamente un sistema distribuido en el que los nodos tienen información privada, que pueden revelar de forma correcta o incorrecta con el objetivo de manipular la asignación de tareas en su propio beneficio.

Hipótesis 2: Existen algoritmos capaces de asignar tareas en un sistema distribuido que incentivan a los participantes a revelar correctamente su información privada.

Hipótesis 3: Tales algoritmos existen en condiciones que puedan hacer viable su implementación en entornos reales, tales como sistemas de intercambio de ficheros en redes P2P o computación distribuida (Grid Computing).

Hipótesis 4: Que estos algoritmos tienen las características adecuadas para que los nodos participantes no deseen cambiar las reglas y propongan sistemas alternativos. En particular, realizarán una asignación razonablemente justa de las tareas a los participantes.

1.4. Metodología

La metodología que hemos utilizado a lo largo de la investigación ha sido la habitual en los trabajos teóricos de computación así como en las matemáticas aplicadas. Esta metodología consiste en:

1. Estudio riguroso de los conceptos, resultados y métodos relativos a cada problema.

2. Revisión profunda de los libros y artículos más recientes sobre el tema.

3. Formalización de todos los conceptos, propiedades y algoritmos. Demostración formal de dichas propiedades.

4. Validación de las conclusiones mediante simulación de escenarios. Para aquellas propiedades que sean difícilmente demostrables o que cuyo comportamiento sea poco intuitivo, se realizarán simulaciones.

1.5. Estado del arte

El problema de la planificación y asignación de tareas en sistemas paralelos o distribuidos es un ámbito muy estudiado en informática [Sin07]. La asignación de tareas y los problemas de planificación han sido estudiados ampliamente en informática, por ejemplo en [Bak74], [Chu80], [CL87], [Fre82], [WS74], [SM94], [AER93], [BCS92],[Bok93]. Estos trabajos sugieren planteamientos en las que se incluyen soluciones basadas en la teoría de grafos [Sto77], [SB78] y de programación entera [ea82] entre otros [ST85], [Sin87], [Lo88]. Existen múltiples propuestas de taxonomías en las que se intenta ordenar y clasificar el problema de la asignación y ejecución de tareas, por ejemplo [CJK88]. Según nuestro conocimiento, ninguno de estos trabajos han abordado el problema que nos ocupa desde el enfoque del egoísmo y utilizando las herramientas de la teoría de juegos para su análisis

Otra línea de estudio, habitual en informática, es la que analiza la asignación de tareas bajo el enfoque de tolerancia a fallos. Por ejemplo, en [AGLS11], [FLSG06] y [AGM10] se proponen soluciones para la asignación de tareas en entornos distribuidos similares a SETI y que intentan paliar el impacto del comportamiento de los nodos bizantinos. De nuevo, consideramos que los modelos de fallos (generales o de tipo bizantino) estudiados en esos trabajos no nos sirven como herramienta para representar el comportamiento de los jugadores egoístas.

El problema de la asignación de tareas ha sido estudiado bajo perspectivas más cercanas a nuestros objetivos, siendo quizás el realizado por Rosenschein y Zlotkin

[RZ94] uno de los más extensos. Ellos definen un "Dominio Orientado a Tareas" o en inglés "Task Oriented Domain". Este concepto formaliza una situación en la que un grupo de agentes deben negociar sobre cómo cooperar para la ejecución de una serie de tareas, reduciendo hasta donde sea posible el coste de la ejecución.

Nuestro trabajo se distingue del propuesto por Rosenschein y Zlotkin en varios aspectos. Por una parte, deseamos eliminar por completo el proceso de negociación ya que, a nuestro modo de ver, esto provocaría un alto número de mensajes, lo que podría dar lugar a una pérdida de rendimiento. Además, el tipo de solución que proponen podría verse afectada si el juego se repitiera en el tiempo. Los agentes podrían aprender de los valores negociados con antelación y aprovechar ese conocimiento. Por otra parte, los autores suponen un modelo de coste diferente al que proponemos. Por ejemplo, en su trabajo el coste es una función que se aplica a un conjunto de tareas, siendo esta función monotónica (es decir, el coste de ejecutar un conjunto reducido de tareas es menor o igual que el coste de ejecutar un subconjunto más amplio).

Otro ámbito en el que se han estudiado problema similares es el correspondiente a la asignación de recursos en redes. En el libro de Narahari y otros [NGNP09] y en la tesis de Hastagiri [Pra06] se tratan varios algoritmos que resuelven dicho problema en Grid-Computing. Aunque muy cercanos a nuestro objetivo, los mecanismos propuestos ("G-DISC", "G-BIC" y "G-OPT") utilizan sistemas de pago, por lo que consideramos que no son aplicables a nuestros fines. Algo similar ocurre con el trabajo de Czumaj y Ronen [Ami] en el que proponen un mecanismo para la asignación de tareas.

Pero las matemáticas de la teoría de juegos impregnan cada día más el mundo científico y es frecuente ver su aplicación en muy diversas ciencias: biología, ciencias sociales, informática e incluso en física. Es por ello que, aunque no hemos podido encontrar trabajos aplicables al propuesto en esta tesis dentro del contexto informático, sí existan múltiples referencias en el ámbito económico. En dicho contexto existen numerosas propuestas cercanas, todas ellas relacionadas con la contratación, venta o subasta de bienes. Como veremos, la ejecución de tareas se podría ver como una

subasta entre varios jugadores. La tarea se asigna (”subasta”) al mejor postor (“el que menor coste declara”). Otra posibilidad sería una variación del problema conocido como *asignación de múltiples unidades bajo demanda única* (“multi-unit single-demand”). Estos modelos se denominan así ya que suponen la venta de varios bienes (“multi-unit”) pero cada agente únicamente consume un bien (“single demand”). Este tema se aborda ampliamente en el capítulo tercero, remitiendo al lector que desee ampliar su exposición a dicho punto.

Finalmente, queremos hacer especial hincapié en un trabajo que proviene del mundo económico pero que consideramos vital para la elaboración de la tesis. La solución propuesta por nosotros se basa en una propuesta publicada por Jackson y Sonnenschein [JS03, JS04, JS07]. En lugar de ofrecer incentivos monetarios (o pagos) a los jugadores, estos mecanismos limitan el espectro de respuestas de los jugadores. En el capítulo cuarto se hace un análisis de su trabajo comparándolo en detalle con los resultados de nuestro trabajo.

1.6. Sobre la notación utilizada

Como se ha comentado anteriormente, la tesis plantea el estudio de un problema pensado inicialmente en el contexto de los sistemas distribuidos. Pronto fue evidente que muchas de las herramientas, ideas y literatura a utilizar procedían de campos tan dispares como la informática (redes, inteligencia artificial, etc.) o la economía (mercados, subastas, etc.), pasando por algunos campos utilizados ampliamente en otras ciencias (p.e. biología) o ingenierías (como puede ser la teoría de control). Todos con un trasfondo matemático que utiliza herramientas procedentes de la teoría de la decisión, teoría de juegos, aproximaciones estocásticas, sistemas de control, probabilidades y estadística, etc.

En este punto surgen dudas sobre cómo abordar y nombrar muchos de los conceptos a utilizar. Un ejemplo claro fue el correspondiente al de nodo. En un contexto informático, el ordenador situado en la red se suele denominar “nodo”. Pero las herramientas estudiadas hacen referencias a ideas similares aunque con nombres muy

dispares. En el ámbito de la inteligencia artificial es probable que se le denomine "agente". En economía, el lector puede encontrar otras formas tales como "principal" y "agente". En matemáticas, especialmente en la teoría de juegos, se le denomina "jugador".

Algo más complejo resulta cuando trabajamos con notaciones matemáticas. Cuando hablamos del diseño de mecanismos nos encontramos con un área de estudio muy dinámica e interesante. En el año 2007 le fue concedido el premio Nobel a algunos de los precursores [Com07, Cih08]. Como hemos comentado, nos encontramos con la circunstancia añadida de que estas herramientas matemáticas se están utilizando en diferentes ámbitos (economía, ciencia de la computación, biología, etc.) por lo que cada ámbito tiende a utilizar una notación y nomenclatura propia. Probablemente sea debido a todo esto que no existe una notación matemática unificada.

Por lo tanto, surge una duda razonable sobre qué nomenclatura y notación debemos utilizar a lo largo del presente trabajo. Cuando se trata de nomenclatura se ha procurado utilizar el mayor rango posible de nombres (nodo, agente, jugador, etc.) aunque siempre intentando definir claramente el sentido del mismo cuando el contexto pudiera ser ambiguo. Cierto es que esta decisión puede generar confusión en el experto de un determinado ámbito, pero se ha tomado esta decisión con la esperanza de aportar una visión más amplia.

Existe una excepción a este criterio; siendo quizás el concepto más utilizado en la tesis y que forma parte del propio título. Me refiero al concepto de *mecanismo*. En el ámbito de la ingeniería, un mecanismo se puede entender como algo físico, como un conjunto de sólidos resistentes, móviles unos respecto de otros. Algo cercano al concepto de máquina. En informática quizás se hubiera hablado de algoritmo o de protocolo. Pero la noción de mecanismo tiene un significado matemático muy concreto en la teoría de juegos (que veremos en el siguiente capítulo), por lo que se ha preferido conservar tal término a lo largo de todo el trabajo.

Por otra parte, la notación matemática intenta seguir una guía de estilo más concreta y aplica el mismo esquema que se suele utilizar en los trabajos de economía o matemáticos.

Aunque la tesis intenta ofrecer un gran formalismo y rigor matemático, en general, se han supuesto las condiciones habituales de las matemáticas aplicadas. Por ejemplo se han supuesto condiciones habituales de continuidad o derivabilidad, se utilizan las integrales de Riemman en lugar de integrales de Lebesgue, se habla de probabilidad cuando deberíamos hablar de medidas en espacios de probabilidad. Muchas veces no se utiliza el concepto en su forma más concreta, principalmente cuando el contexto utilizado es claro. Por ejemplo, se suele decir que los jugadores no están correlados, aunque deberíamos decir que son o no mutuamente independientes. Lamentamos si estas licencias han supuesto un problema a algún lector.

1.7. Estructura de la tesis

La estructura del presente documento refleja la evolución de la tesis. Aunque inicialmente el problema planteado se analizó bajo un modelo considerado amplio y ambicioso, pronto fueron evidentes nuevas alternativas que enriquecían dicho modelo y que abrían nuevas expectativas de investigación. Por ello, la tesis fue evolucionando conforme se intentaba generalizar al máximo posible todos los resultados obtenidos. En realidad, la presente tesis aporta una colección de algoritmos, cada uno de ellos más o menos independiente de los demás, pero que constituyen soluciones al mismo problema aunque con diferencias en los modelos.

Aún siendo una tesis con una orientación informática, no es posible evitar un cierto contenido formal y matemático. Es por ello que hemos considerado oportuno introducir en el siguiente capítulo los principales conceptos que se utilizarán a lo largo de la misma, especialmente aquellos que proceden de la teoría de juegos y el diseño de mecanismos. Nuestra experiencia es que la notación utilizada suele variar mucho en la literatura explorada, y hemos considerado oportuno dedicar un capítulo a formalizar y a unificar dicha notación. Aunque el lector esté familiarizado con dichos conceptos, este capítulo puede ser utilizado como punto de referencia y de aclaración cuando la notación resulte confusa.

El tercer capítulo esta dedicado a la definición del modelo y a algunos de los

algoritmos que consideran pagos para evitar la manipulación de los agentes. En él se introduce más a fondo el problema y se argumenta los criterios utilizados para el planteamiento del modelo. Al final de este capítulo se ha incorporado una revisión del estado del arte de los mecanismos con reparto justo de pagos.

El cuarto capítulo presenta el primer algoritmo que resuelve el problema sin la necesidad de utilizar pagos. Como veremos en los capítulos siguientes, la principal restricción del modelo utilizado en el capítulo cuarto es que se considera que los nodos (jugadores) son entes que tienen costes independientes entre ellos.

En el quinto capítulo se exploran las consecuencias de relajar el modelo anterior, permitiendo que los jugadores puedan tener correlación en sus costes. Como veremos en dicho capítulo, esto provocará un replanteamiento de conceptos tales como la justicia en el reparto del trabajo. De esta forma, surgen dos nuevos algoritmos que abordan el problema según estas variantes.

Una vez completada esta etapa, han surgido nuevas ideas y planteamientos al modelo. El capítulo sexto está dedicado a las limitaciones que pueden surgir al relajar los modelos anteriores. Se abre la puerta a todo un amplio espectro de cuestiones muy interesantes desde nuestro personal punto de vista. El interés de este capítulo es doble, por una parte hemos intentado describir todas las limitaciones del modelo más avanzado de los propuestos. En segundo lugar, creemos que este capítulo añade nuevas hipótesis de trabajo, herramientas y conceptos que esperamos sean útiles en un futura investigación.

El capítulo séptimo finaliza con las conclusiones del trabajo, aporta algunas reflexiones sobre el mismo, problemas abiertos y resume las líneas de trabajo futuras.

Capítulo 2

Teoría de Juegos y Diseño de Mecanismos

> "The selfishness must be discovered and understood before it can be removed. It is powerless to remove itself, neither will it pass away of itself. Darkness ceases only when light is introduced; so ignorance can only be dispersed by Knowledge; selfishness by Love."
>
> James Allen

2.1. Teoría de Juegos

En este capítulo se introducirán los principales conceptos matemáticos relacionados con la teoría de juegos y el diseño de mecanismos. Nuestra intención al incluir este contenido ha sido establecer un único punto de referencia en el que el lector pueda consultar la notación utilizada. Además, acompañando a la definición de cada concepto, se han incluido reflexiones y preguntas sobre las posibles implicaciones que dicho concepto pueda tener en nuestro futuro modelo.

2.1.1. Introducción

Ejemplos similares a los descritos en el capítulo anterior constituyen el tipo de interacción que estudia la teoría de juegos. Esta rama de la matemáticas iniciada en los primeros años del siglo veinte, estudia las interacciones entre jugadores con conflictos de intereses. Su nombre proviene de que los primeros ejemplos estudiados eran variantes de juegos tradicionales (juegos de cartas o de tablero tipo ajedrez). Es un ámbito de estudio fascinante. Existen numerosos libros que pueden servir de introducción al lector, por ejemplo [Osb04, R.91, Ras07, OR94].

En esencia, la teoría de juegos estudia las situaciones que suponen:

1. Por un lado, una interacción (juego) entre jugadores definida por una serie de reglas conocidas por dichos jugadores.

2. Los jugadores pueden optar a lo largo de la interacción por una serie de acciones que definen su comportamiento o estrategia.

3. Este comportamiento influirá en el desarrollo de la interacción y por lo tanto influirá en el resultado final del juego.

4. La interacción concluye en una serie de estados finales que son más o menos deseables por cada uno de los jugadores.

5. Estos resultados ("outcomes") son conocidos por los jugadores y son capaces de valorar si prefieren un estado a otro.

Probablemente el ejemplo más conocido de la teoría de juegos es el denominado "juego de los prisioneros". Quizás sea uno de los ejemplos más utilizados y más malinterpretados de la literatura [1]. En su lugar, nos gustaría proponer un juego

[1] El "juego de los prisioneros" se conoce también como el "dilema del prisionero". Este concepto fue desarrollado inicialmente por Merrill Flood y Melvin Dresher trabajando para la RAND Corporation. Posteriormente el matemático Albert W. Tucker le dió forma matemática. La bibliografía que trata este concepto es inmensa.

	C	NC
NM	$c1_c - c2_c$	$c1_c$
M	$-c2_c$	$-C_c$

Cuadro 2.1: Beneficio para el jugador consumidor

que, aunque pudiera recordar al dilema de los prisioneros, refleja el contexto de la computación distribuida.

Supongamos que un ordenador (productor) debe ejecutar una tarea y comunicar el resultado a un segundo ordenador (consumidor). Consideremos que el productor puede realizar la tarea según le corresponde, o por el contrario, puede inventarse el resultado y mentir en el mismo. Si realiza efectivamente la tarea, el productor gasta ciertos recursos que puede ahorrarse si decide mentir. El consumidor puede realizar también dos acciones: comprobar o no el resultado de la tarea. El beneficio (o pérdida) que obtienen ambos jugadores se suele representar mediante tablas en las que cada casilla representa el beneficio que obtiene un jugador dadas las estrategias de cada jugador. En nuestro caso, esto se ha representado por las tablas 2.2 y 2.1. En ambas tablas se muestran las cuatro posibles combinaciones resultado de combinar las dos posibles acciones del productor (representadas como M para mentir y NM para no mentir) con las dos acciones del consumidor (representadas como C para comprobar y NC para no comprobar).

Según se observa en estas tablas, el consumidor puede ganar $c1_c$ unidades monetarias si el productor no miente y puede perder hasta C_c si el productor miente y el consumidor no comprueba. El coste de la comprobación se ha representado por $c2_c$. Para que el juego tenga sentido se supone que C_c es mayor que el coste de comprobar $C_c \geq c2_c$. Si el consumidor comprueba la tarea y el productor miente, entonces el consumidor únicamente pierde el valor de la comprobación.

El productor pierde $-c_p$ cuando ejecuta la tarea, independientemente de que el consumidor compruebe o no la tarea. En cambio puede ahorrarse dicha cantidad si

	C	NC
NM	$-c_p$	$-c_p$
M	$-C_p$	0

Cuadro 2.2: Beneficio para el jugador productor

decide mentir y el nodo consumidor no lo comprueba. El riesgo es que si decide mentir y el consumidor lo comprueba, entonces el productor pierde una cantidad alta (por ejemplo debido a una multa) $C_p > c_p$.

Este juego es un claro exponente de los primeros juegos que estudiaba la teoría de juegos. En él vemos las principales características de un juego. Los jugadores influyen en los beneficios de los demás según sus propias acciones. La pregunta que trata de responder la teoría de juegos es ¿qué deben hacer cada uno de los jugadores para maximizar su beneficio (o minimizar sus pérdidas)?.

En las siguientes secciones se concretan algunas de las definiciones más utilizadas en la especificación formal de juego, utilizando el ejemplo anterior como guía aclaratoria. El orden de la exposición será el siguiente:

1. Definición de jugador racional, preferencias y función de utilidad

2. Definición de estrategia

3. Tipos de soluciones o puntos de equilibrio.

2.1.2. Jugadores, preferencias y función de utilidad

Definición 2.1.1 (Jugadores)**.** El conjunto de jugadores se denota por $N = \{1, 2, \ldots, i, \ldots, n\}$.

La cardinalidad del conjunto es finita, siendo el número de jugadores igual a $|N| = n$.

Se representa a los jugadores individuales por letras (por ejemplo i, j o k).

En nuestro ejemplo anterior, el conjunto de jugadores es $N = \{productor, consumidor\}$, siendo $n = 2$.

Es evidente que N debe ser un conjunto no vacío y finito. Para que el juego sea una interacción en conflicto, se considera que al menos existen dos jugadores. Cuando únicamente existe un jugador, entonces se suelen utilizar las matemáticas relacionadas con la teoría de la decisión. Algunos autores consideran la teoría de juegos como una extensión natural de la teoría de decisión.

Definición 2.1.2 (Estados finales)**.** El juego pasa por una serie de estados que concluye en un estado final o conclusión del juego (en inglés "outcomes"). El conjunto de posibles estados finales o resultados del juego se representa por D, siendo un elemento concreto d o d'. El conjunto de posibles estados finales puede ser finito o infinito dependiendo del juego.

En la notación anterior se ha representado el conjunto de resultados por la letra D ya que, como veremos, en la teoría de mecanismos estos estado finales se denominan *Decisiones Sociales* (en inglés "Social Decisions").

Uno de los aspectos más criticados de la teoría de juegos es la suposición de que un jugador es *racional*. Aunque realmente lo que esto significa es simplemente que los jugadores pueden decidir qué alternativas o estados finales prefieren. Este concepto de racionalidad de un jugador se basa en un planteamiento axiomático que genera, en última instancia, una función de utilidad. Este punto ha sido ampliamente estudiado en la denominada teoría de la decisión y en concreto la teoría de la racionalidad y de la utilidad esperada.

Durante los últimos años, la Teoría de la Utilidad Esperada (TUE) de von Neumann y Morgenstern ha sido el estándar de la decisión individual en régimen de riesgo. Como es perfectamente conocido, el primer antecedente histórico de la TUE es debido a Bernouilli (1738). Posteriormente, von Neumann y Morgenstern [VNM44] dan forma axiomática al comportamiento del individuo ante la decisión en riesgo. Éste es, desde entonces, el paradigma de la decisión individual. En esta teoría, la de-

finición de racionalidad se sustenta sobre la estructura axiomática de la relación de preferencias.

No obstante, desde principios de los años cincuenta, está teoría clásica ha sido fuertemente criticada y discutida, generando alternativas y generalizaciones en un intento de modelar el comportamiento observado de los individuos. Como consecuencia, surgieron grandes familias de teorías alternativas, tales como las "teorías binarias", teorías de la utilidad Hemisimétrica de Fishburn ("Skew-Symmetric Bilinear"), las teorías del arrepentimiento ("Regret Theory") y otras.

Muchos de esos trabajos buscan generalizar la estructura y topología de los conjuntos de elección. A modo de ejemplo, véase Herstein y Milnor [HM53] en la que proponen conjuntos mezclados ("mixture sets") en lugar de espacios vectoriales. Otros, buscan la simplificación o una elección diferente de axiomas. Existe una amplia bibliografía al respecto, desde la propuesta original de Von Neumann y Morgenstern [VNM44] hasta estudios sobre las limitaciones del modelo [Sch82]. Nuestro objetivo en estas líneas es simplemente mostrar que el concepto de utilidad tiene una base matemática (axiomática) y, en cualquier caso, no deseamos hacer un recorrido exhaustivo de todas esas teorías. Así pues, nos limitaremos a exponer un breve resumen de un modelo concreto, siendo conscientes de que apenas hemos arañado una mínima parte de toda esta extensa y amplia teoría.

En el centro de la teoría de la elección tradicional se halla un individuo que se enfrenta a un conjunto de alternativas, o *conjunto de elección*, denotado por cualquier letra mayúscula, sea ésta A, B, S, o X. La teoría de la decisión estudia estos conjuntos desde un punto de vista muy general, ya que contempla una amplia variedad de posibilidades (conjuntos finitos e infinitos, con diversas topologías, etc.). A modo de ejemplo, una forma sencilla y muy habitual de representar el conjunto de elección es como subconjunto convexo del espacio euclídeo n-dimensional: $A \subseteq \mathbb{R}^n$. Por convexo, recuérdese que se entiende que si $x, y \in A$, entonces $\alpha x + (1 - \alpha)y \in A$ para todo $\alpha \in [0, 1]$.

La elección de A como nombre del conjunto de elección es meramente puntual y es únicamente a título del presente ejemplo. Como veremos, las preferencias del

jugador se denotará de forma diferente.

El problema de decisión del individuo es determinar qué elemento del conjunto de elección prefiere. Para determinar el elemento del conjunto de elección que se prefiere, el individuo considerará dos elementos arbitrarios, $x, y \in A$. De estos elementos elige el elemento preferido, x, y descarta el otro, y.

Para ordenar las preferencias, el jugador establece un orden con las combinaciones específicas de elementos del conjunto de elección en función de sus preferencias y se supone que cada individuo tiene una relación de preferencias, que denotamos por el símbolo $\succeq$. La relación binaria permite la comparación de pares de alternativas $x, y \in A$, donde $x \succeq y$ significa que x es preferido a y. A partir de este concepto se suelen definir otros símbolos ($\succ, \preceq, \prec$ o $\sim$) con el significado habitual. La preferencia estricta ($\succ$) significa que el agente prefiere una alternativa pero no la contraria ($x \succeq y$ pero no $y \succeq x$). La indiferencia $x \sim y$ es la ausencia de preferencia estricta; el agente es indiferente entre x e y ($x \succeq y$ y además $y \succeq x$). Toda la teoría presupone que las preferencias del individuo son racionales y que las hipótesis de racionalidad se representan por la estructura axiomática siguiente:

1. *Axioma 1. Reflexibilidad.*

   $$\forall x \in A, x \sim x.$$

   Este axioma dice que si x se compara con x, el jugador es indiferente entre las dos elecciones.

2. *Axioma 2. Comparabilidad (completitud).*

   $$\forall x, y \in A, x \succeq y \text{ o bien } y \succeq x.$$

   Cada par de elementos de un conjunto de elección A pueden compararse.

3. *Axioma 3. Transitividad.*

   $$\forall x, y, z \in A, \text{si } x \succeq y, y \succeq z \Rightarrow x \succeq z.$$

   Este axioma asegura que las elecciones son consistentes. Si x es preferido a y y y es preferido a z entonces x es preferido a z.

Los axiomas 1 a 3 definen una relación de preferencia, también denominada una relación de preferencia racional o relación de orden débil. Aunque parecen axiomas muy razonables, en realidad son puntos de partida que han sido objeto estudios muy profundos como se ha comentado anteriormente.

Cuando el conjunto A es finito (o infinito pero contable) se puede asegurar la existencia de una función de utilidad por medio del siguiente teorema:

Teorema 2.1.1. *Dada una relación de preferencias sobre un conjunto de elección A finito (o contable), que satisface los axiomas 1 a 3, existe una función de utilidad $U : A \to \mathbb{R}$ tal que:*

1. $x \sim y$ *sí y sólo sí* $U(x) = U(y)$ *y*

2. $x \succeq y$ *sí y sólo sí* $U(x) \geq U(y)$

Cuando el conjunto A es no es finito (ni contable), se necesitan condiciones adicionales. Por ejemplo, si el conjunto A es compacto y es un subconjunto convexo de un espacio métrico separable, se puede afirmar que existe la función de utilidad sí y sólo sí la relación $\succeq$ es continua (la preferencia débil se conserva en el límite). Otras veces, se definen axiomas adicionales que ayudan a concretar la topología de A, por ejemplo con:

1. *Axioma 4.*

$$\forall x, y \in A \text{ donde } x \succeq y, \text{ y } \forall \alpha, \beta \in [0, 1]$$

$$[\alpha x + (1 - \alpha)y] \succeq [\beta x + (1 - \beta)y] \text{ si y sólo si } \alpha \geq \beta.$$

2. *Axioma 5.*

$$\forall x, y, z \in A, \text{ si } x \succeq y \succeq z \Rightarrow \exists \alpha \in (0, 1) \text{ tal que } \alpha x + (1 - \alpha)z \sim y.$$

3. *Axioma 6.*

$$\exists x, z \in A, \text{ tal que } x \succeq y \succeq z \forall y \in A.$$

El conjunto de axiomas anterior no es único y existen múltiples propuestas, ya que intentan extender estos resultados a estructuras y topologías más generales. La existencia de la función de utilidad para casos más generales es un tema ampliamente estudiado. Se remite al lector a Fishburn [Fis70, Fis10], Hertein, Milnor (1953) [HM53], Debreu [Deb59, Deb72] o al monográfico que Barbera y otros publicaron sobre el tema [BHS04a, BHS04b].

La función de utilidad nos permite representar el comportamiento de las preferencias del jugador de una manera más fácil. Solamente se necesita conocer la función de utilidad $U(\cdot)$ para poder trabajar con las preferencias de un individuo, lo que ofrece una gran simplificación en términos del análisis matemático.

La importancia de lo expuesto anteriormente es que se supone que un nodo puede conocer si prefiere una alternativa a otra y que es posible crear funciones de utilidad. En nuestro ejemplo de los ordenadores productores/consumidores, cada uno de ellos se enfrenta a cuatro posibles resultados (representados por las celdas de las tablas anteriores). Se supone que cada uno de los jugadores puede medir "la intensidad" de esta preferencia ya que existe una valoración económica asociada al estado. Y de hecho, esa valoración económica puede ser utilizada como el valor definido por la función de utilidad.

En muchos de los ejemplos estudiados por la teoría de juegos, el valor de la función de utilidad suele coincidir con un valor monetario. Es decir, los jugadores deciden entre uno o varios estados en función de los beneficios o pérdidas que le produce dicho estado. Además, en esos juegos se suele aceptar pagos o cobros entre jugadores. En ese caso, se dice que un jugador transfiere utilidad a otro jugador. Cuando existe un pago, lo que realmente está pasando es que un jugador verá reducida su función de utilidad en una cantidad igual al pago. El estado asociado se hará por tanto menos preferible. El jugador que recibe el pago, tendrá un valor más alto de utilidad y por lo tanto, el estado asociado se hará más preferible.

Aunque esta transferencia de utilidad suele ser muy habitual en los modelos económicos, e incluso en los trabajos existentes en redes de ordenadores, no siempre se puede aceptar la existencia de utilidades compatibles entre jugadores.

Volveremos a este tema cuando analicemos el problema que aborda esta tesis y tendremos que reflexionar sobre qué suposiciones serán las apropiadas para la elaboración del modelo. Principalmente, tendremos que hacer frente a las siguientes cuestiones:

1. ¿Podemos suponer que los nodos en la red tienen capacidad para determinar sus preferencias en la ejecución de tareas?

2. ¿Qué sucede si éstos no son completamente racionales?

3. Teniendo en cuenta que los medios de pagos son en realidad una transferencia de utilidad, ¿deberíamos suponer que las funciones de utilidad son compatibles entre jugadores?

2.1.3. Estrategia

En la teoría de juegos se supone que los jugadores conocen los posibles estados finales del juego y que tienen unas preferencias sobre dicho conjunto. Además, para que podamos hablar propiamente de juego, suponemos que los jugadores tienen un conflicto de intereses y que diferentes jugadores prefieren diferentes estados. Además, los jugadores disponen de acciones que pueden alterar el curso del juego y, por lo tanto, influir en el estado final del mismo. Sin este conflicto, la teoría de juegos se vuelve a convertir en una teoría de la decisión, limitándonos a problemas de optimización tradicional.

Un jugador intenta obtener un estado final que le sea favorable, y para ello realiza las acciones que le puedan conducir a ello. Es decir, decide y realiza una estrategia de juego.

Definición 2.1.3 (Estrategia pura)**.** Una estrategia pura es un curso de acción, un plan o una regla de decisión que define las acciones de un jugador en cada estado del juego. Sea $s_i \in \Sigma_i$ una estrategia de un jugador i en la que Σ_i es el conjunto de todas las posibles estrategias para dicho jugador.

En la anterior definición hemos tratado al jugador i como una entidad de características conocidas y únicas. Se dice que el jugador tiene un tipo único. Si alguno de los parámetros que definen al jugador no es conocido se introduce el concepto de "tipo de jugador" y de "información privada" que se verán más adelante.

Es importante introducir en este punto la notación $s = (s_1, \ldots, s_n)$ para la estrategia conjunta de todos los jugadores ("strategy profile"), y

$$s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$$

es la estrategia conjunta de todos los jugadores excepto el jugador i.

En la teoría de juegos se supone que cada jugador puede elegir una o varias estrategias (o planes) de forma aleatoria en cuyo caso se dice que opta por una estrategia mixta.

Definición 2.1.4 (Estrategia mixta)**.** Una estrategia mixta $\sigma_i \in \Delta(\Sigma_i)$ define una distribución de probabilidad sobre el conjunto de estrategias puras, donde $\Delta(\Sigma_i)$ representa el conjunto de todas las posibles distribuciones de probabilidad definidas sobre Σ_i.

Es evidente que el seguimiento de todo $s = (s_1, \ldots, s_n)$ conduce a un estado final d.

En nuestro ejemplo anterior, el conjunto de estrategias puras para el consumidor es

$$\Sigma_c = \{Comprueba, NoComprueba\}$$

y el del productor es

$$\Sigma_p = \{Miente, NoMiente\}$$

. Una estrategia mixta vendría determinada por dos probabilidades p y q; siendo p la probabilidad con la que el consumidor decide "comprobar" y q la probabilidad con la que el productor decide "mentir". Dado que el conjunto de estrategias puras tiene únicamente dos alternativas, $1 - p$ y $1 - q$ son las probabilidades de realizar las acciones opuestas ("no-comprobar" y "no-mentir" respectivamente).

Obsérvese que en juegos no repetidos, el conjunto de estrategias suele coincidir con el conjunto de acciones posibles. En juegos repetidos, el conjunto de estrategias suele ser el producto cartesiano de las acciones o un subconjunto del mismo.

Siguiendo nuestro ejemplo del consumidor-productor, el conjunto de estados finales es simplemente el producto cartesiano de Σ_c y Σ_p. Es decir $D = \{Miente - Comprueba, Miente - NoComprueba, NoMiente - Comprueba, NoMiente - NoComprueba\}$.

Como se ha visto en la sección de preferencias y utilidad esperada, es posible definir una función de utilidad para cada jugador.

Definición 2.1.5 (Preferencias)**.** Cada jugador i tiene una preferencia definida sobre los estados del juego D. Como se ha visto anteriormente, estas preferencias pueden trasladarse a una función de utilidad. Se dice pues que $v_i : D \to \mathbb{R}$.

Esta valoración numérica se suele corresponder con el beneficio positivo o negativo que el jugador obtiene en ese estado. Por lo tanto, $v_i(d)$ indica el beneficio que obtiene i si el juego concluye en el estado $d \in D$. Igualmente, $v_i(d) > v_i(d')$ indica que el jugador i prefiere el estado d al estado d'.

En algunos casos, y abusando de la notación, se denotará la función de utilidad como $v_i : S \to \mathbb{R}$, siendo S el conjunto producto cartesiano de todas las posibles estrategias de todos los jugadores. Por lo tanto, $v_i(s)$ o $v_i(s_1, \ldots, s_n)$ representa el mismo valor que $v_i(d)$ siendo d un estado final.

A lo largo de este trabajo se verá como existen dos posibles hipótesis con respecto a dicha función de utilidad. Cuando se supone que la utilidad de un jugador se corresponde con un sistema monetario, es razonable pensar que la utilidad o beneficio puede transferirse entre los jugadores (por ejemplo mediante pagos). Otra alternativa es suponer que no es posible el traspaso en cuyo caso se dice que la "utilidad es no transferible".

De nuevo, tenemos que reflexionar sobre qué suposiciones podrían ser las apropiadas para la elaboración del modelo que deseamos estudiar. Principalmente, tendremos que hacer frente a las siguientes cuestiones:

1. ¿Qué podemos suponer sobre el conjunto de acciones de los nodos? Este punto es una de las claves del modelo como veremos más adelante. Las principales diferencias entre los mecanismos propuestos son consecuencia de ampliar el conjunto de acciones.
2. ¿Permitimos que el juego sea repetido?
3. ¿Podemos suponer que los jugadores sean capaces de crear coaliciones entre ellos?
4. Debemos suponer que existe información privada, ¿Debemos poner alguna limitación a este supuesto?

2.1.4. Clasificación de los juegos y tipos de solución

La definición formal de juego se suele realizar utilizando tres formas diferentes: la forma normal o estratégica ("strategic form"), la forma extensiva ("extensive form") y la forma para coaliciones ("coalitional form"). Para una explicación detallada de estos conceptos, se remite al lector a cualquier libro de introducción a la teoría de juegos [Osb04].

Definición 2.1.6 (Juego en forma estratégica)**.** Un juego en su forma estratégica es cualquier Γ con la estructura $\Gamma = \langle N, (\Sigma_i)_{i\in N}, (v_i)_{i\in N}\rangle$ donde N, Σ_i y v_i son los conjuntos no vacíos descritos en los párrafos anteriores.

El fin de la teoría de juegos es estudiar el desarrollo más probable del juego, y para ello suponen que todos los jugadores son racionales y que todos disponen de la misma información (es decir tienen un conocimiento común). En realidad un juego puede tener varios desarrollos o estados finales a los que se puede llegar si todos los jugadores son racionales. Esos estados se denominan *soluciones del juego* o *puntos de equilibrio*.

El primero en trabajar y definir una solución de un juego fue el premio Nobel John Nash. Esta solución se denomina equilibrio de Nash. Se dice que un conjunto

de estrategias constituye un equilibrio de Nash cuando ningún jugador puede mejorar su beneficio alterando su estrategia, considerando que las estrategias del resto de los jugadores han sido fijadas.

Formalmente, el equilibrio de Nash se define como:

Definición 2.1.7 (Equilibrio de Nash)**.** Una estrategia conjunta $s = (s_1, \ldots, s_n)$ se dice que constituye un equilibrio de Nash si todos los jugadores maximizan su utilidad esperada. Es decir, para todo i se cumple que:

$$v_i(s_i, s_{-i}) \geq v_i(s'_i, s_{-i}), \forall s'_i \neq s_i.$$

Una solución basada en el equilibrio de Nash es uno de los posibles conceptos de solución que se puede aplicar a un juego. Aunque se considera una de las más importantes, existen otros conceptos de soluciones que se aplican cuando se relajan las hipótesis de trabajo, como la de información completa. Por ejemplo, cuando los jugadores no tienen la certeza de la función de beneficio de los otros jugadores (aunque tienen un conocimiento basado en cierta distribución de probabilidad) se considera el concepto de equilibrio bayesiano de Nash. Este tipo de solución se definirá más adelante, cuando se vean los conceptos de información incompleta.

Existe un tercer concepto de solución con mayor relevancia para el problema que nos ocupa. Este tipo de solución se basa en el concepto de que la mejor estrategia de un jugador no depende de las estrategias de los demás. En este caso se dice que la estrategia es dominante.

Definición 2.1.8 (Estrategia Dominante)**.** Una estrategia s_i se dice que es una *estrategia dominante* si con ella el jugador maximiza su beneficio independientemente de las estrategias del resto de los jugadores. Es decir, se cumple que:

$$v_i(s_i, s_{-i}) \geq v_i(s'_i, s_{-i}), \forall s'_i \neq s_i, \forall s_{-i} \in \Sigma_{-i}.$$

En un principio, un equilibrio basado únicamente en estrategias dominantes puede parecer un concepto poco interesante para ser estudiado en la teoría de juegos. Sin

embargo, este concepto es uno de los más relevantes cuando se trabajan con el diseño de mecanismos. De hecho es el concepto de solución deseado, ya que supone muy pocos conocimientos por parte de los jugadores. Según nuestra opinión, el concepto tradicional de solución basada en el equilibrio de Nash tiene poco sentido, salvo que todos los jugadores tengan conocimiento completo de las funciones de utilidad de los demás jugadores. Se verá más en profundidad este concepto en la sección siguiente.

En este punto, podemos intentar resolver nuestro juego sencillo del consumidor y del productor. Dado que tenemos información completa de los jugadores (conocemos la tabla de beneficios para todos los estados finales y de todos los jugadores), parece razonable buscar el equilibrio de Nash para dicho juego. Es decir, debemos buscar un par de estrategias (puras o mixtas) que cumplan con la definición anterior (2.1.7); una para el productor y otra para el consumidor.

Simplemente explorando la tabla de beneficios, vemos que los jugadores no tienen estrategias puras que sean de interés. Pero, por el famoso teorema de Nash sabemos que existe al menos un par de estrategias mixtas que cumplen dicha condición. Por lo tanto, debemos buscar dos valores de probabilidad p y q (ver definición en 2.1.3).

El valor que obtiene el consumidor si decide comprobar es:

$$v_c(Comprueba) = (1-q)(c1_c - c2_c) + q(-c2_c)$$

Nótese que la utilidad del consumidor al comprobar viene dada por la combinación de los dos estados posibles: Comprueba-Miente y Comprueba-NoMiente. Dado que el productor está siguiendo una estrategia mixta, cada uno de esos estados vienen dados por la probabilidad de que el productor mienta (q). La expresión anterior refleja este hecho.

Por otra parte, si el consumidor decide no comprobar:

$$v_c(NoComprueba) = (1-q)(c1_c) + q(-C_c)$$

Según la frecuencia de la mentira del productor, el consumidor le interesará comprobar o no. Por ejemplo, si el consumidor supiera de forma cierta que el productor no

miente ($q = 0$), su estrategia óptima sería no comprobar ya que $v_c(NoComprueba) > v_c(Comprueba)$. Las ecuaciones anteriores nos definen el comportamiento del consumidor en función de lo que podría hacer el productor. Este concepto se le suele llamar la "mejor respuesta". Una estrategia mixta para el consumidor únicamente tiene sentido cuando los dos valores anteriores se igualan (al consumidor le es indiferente comprobar o no). En ese caso, tenemos que:

$$(1-q)(c1_c - c2_c) + q(-c2_c) = (1-q)(c1_c) + q(-C_c)$$

$$c1_c - c2_c - q(c1_c - c2_c) - qc2_c = c1_c - qc1_c - qC_c$$

$$qC_c = c2_c$$

$$q = \frac{c2_c}{C_c}$$

De igual forma podemos hacer con el productor:

$$v_c(Miente) = p(-C_p)$$

$$v_c(NoMiente) = (1-p)(-c_p) + p(-c_p) = -c_p$$

Igualando:

$$p(-C_p) = -c_p$$

$$p = \frac{c_p}{C_p}$$

Los resultados anteriores se pueden representar por la gráfica 2.1. En esta gráfica se representan las respuestas del consumidor (línea azul) y del productor (línea roja) según las probabilidades del contrario. Si observamos el comportamiento del consumidor podemos ver que su estrategia sería no comprobar si la probabilidad de mentir q es suficientemente baja. Por el contrario, su mejor estrategia sería comprobar a partir de cierto umbral de mentiras. Únicamente en ese umbral le resultaría indiferente comprobar o no. Esa es la razón de que la línea azul tenga esa forma quebrada. El mismo razonamiento se aplicaría al productor. El punto de equilibrio se produce en el cruce de ambas líneas.

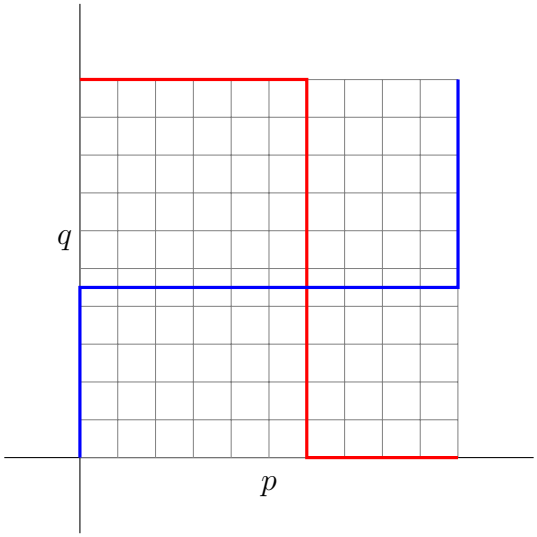

Figura 2.1: Equilibrio de Nash

En este punto es relevante resaltar un aspecto muy interesante del ejemplo. Las probabilidades de cada jugador *no dependen de sus propios valores*. En realidad dependen *únicamente* de los valores del otro jugador. Recordemos que p es la probabilidad de comprobar o no, y ¡esta probabilidad está en función de los valores del productor! Quizás al principio pueda parecer extraño, pero se si piensa detenidamente, es lógico que el consumidor establezca la probabilidad de comprobar en función de cuanto puede perder o ganar el productor. Igualmente sucede con la probabilidad q.

Finalmente, se puede determinar la probabilidad de que el nodo productor mienta y el nodo consumidor no compruebe:

$$Prob\{fallo\} = (1-p)q = (1 - \frac{c_p}{C_p})\frac{c2_c}{C_c} =$$

$$\frac{(C_p - c_p)c2_c}{C_p C_c}$$

Aunque la representación del juego sea relevante para el análisis del mismo,

lo más interesante para nuestros propósitos es determinar el tipo de solución que buscamos para nuestro modelo. Y tenemos que volver a enfrentarnos a cuestiones claves:

1. ¿Qué tipo de solución deberíamos exigir al planteamiento del modelo? ¿Dominante o bayesiana de Nash?

2. ¿Es posible buscar una solución de estrategia dominante?

3. ¿Existen soluciones de estrategia dominante para modelos generales de nuestro problema?

4. De nuevo, consideramos importante suponer que existe información privada, ¿En qué medida nos afecta? ¿Podríamos crear soluciones que no sean dominantes?

En este punto, el lector habrá observado que estamos utilizando la teoría de juegos pero no tenemos un juego concreto al que podamos aplicar todas estas nociones. Aún no tenemos un modelo de juego creado. La teoría de juegos aporta las herramientas para analizar un juego cuando éste está ya definido, pero por sí misma no nos ayuda en el diseño o planteamiento del mismo.

Para ello existe una rama de la teoría de juegos denominada diseño de mecanismos que nos servirá en nuestros propósitos. Esta teoría, muy reciente en su desarrollo, nos ayudará a conocer qué tipo de soluciones son posibles y a cómo formalizar el problema que nos ocupa. En la siguiente sección veremos un breve resumen del diseño de mecanismos, estado del arte, los conceptos matemáticos básicos y su aplicación al diseño de sistemas distribuidos complejos.

2.2. Diseño de Mecanismos

En todos los ejemplos anteriores hemos visto cómo la teoría de juegos estudia la interacción, una vez que se han definido claramente todos los componentes del

mismo, entre ellas las reglas del juego. Cuando se trata de realizar el proceso inverso, es decir, diseñar el juego para que cumpla con ciertas propiedades, se utiliza una sub-rama de la teoría de juegos denominada "Diseño de Mecanismos".

En este apartado trataremos brevemente el concepto de diseño de mecanismos como herramienta matemática para crear juegos que cumplan con las propiedades deseadas. Aunque es un campo de estudio bastante reciente, existe una amplia bibliografía al respecto, incluyendo algunos resúmenes o revisiones del estado del arte. Véase por ejemplo dos revisiones de Jackson ([Jac03, Jac99]) y una revisión reciente de Corchon [Cor07]. También es muy abundante la literatura al respecto en algunos de libros muy utilizados en la economía, tales como [Ber05] y [MCWG95]; o incluso en informática [NRTV07], y [NGNP09]. Otras referencias que podrían ser útiles al lector relativas a la teoría de la implementación y diseño de mecanismo son [Hur73], [Var95], el capítulo 10 de Osborne y Rubinstein [OR94], capítulos dos y tres de [ASB05], [FS06], [BS08].

2.2.1. Introducción

Eric S. Maskin define la teoría de mecanismos [Mas08] como

> *The theory of mechanism design can be thought of as the "engineering" side of economic theory.*

El diseño de mecanismos nació, y ha estado, muy ligado al mundo económico. Nace fundamentalmente en el ámbito económico como necesidad para la adjudicación o subastas de contratos públicos. Roger B. Myerson, en [Mye08], resume la historia del diseño de mecanismos. El lector interesado, también puede encontrar una breve introducción a la historia del diseño de mecanismos en el trabajo de Eric S. Maskin [Mas08].

Los primeros estudios analizaban ejemplos relacionados con subastas, sistemas de votaciones, división de bienes -por ejemplo en la disolución de un matrimonio o de una empresa-, etc. La principal pregunta es si se puede o no diseñar un mecanismo, protocolo o sistema que permita llegar a una conclusión o resultado deseado

independientemente de las características de los jugadores. Su nombre proviene de los primeros mecanismos o protocolos para la realización de subastas.

En todos los ejemplos mencionados (subastas, contratación, etc.), vemos los elementos comunes del diseño de mecanismos. Por un lado, el diseñador del juego debe decidir qué opciones o resultados finales son preferibles. De todos los posibles resultados de la interacción, el diseñador del mecanismo elige el subconjunto que desea obtener (implementar). Es importante destacar esta diferencia con respecto a la teoría de juegos. En la teoría de juegos, las reglas están definidas y la solución del juego implica calcular cuál o cuáles de los resultados se pueden obtener. Es decir, se obtienen las soluciones al problema determinando los posibles resultados a la interacción. En el diseño de mecanismos, primero se decide qué subconjunto de soluciones se desea y posteriormente se definen las reglas del juego para que se produzcan dichas soluciones.

Por otra parte, los agentes o jugadores mantienen una información que se considera privada o confidencial. Esta información no es conocida por los demás jugadores ni por el propio diseñador del juego. Por supuesto, pueden ser valores obtenidos de ciertas distribuciones probabilísticas que son conocidas, al menos por el diseñador del juego. Una vez más, esto marca una gran diferencia con la teoría de juegos. En la teoría de juegos, los jugadores tienen un conocimiento de lo que los demás pueden preferir (aunque estas preferencia procedan de distribuciones de probabilidad), y además los otros jugadores saben que el otro lo sabe. Este conocimiento, aplicado hasta el infinito, se denomina “conocimiento común”. El diseñador de un mecanismo no suele suponer este tipo de premisas, e intenta diseñar el juego para que sea posible obtener este conocimiento mediante los incentivos correspondientes.

El diseño de un mecanismo no suele ser un problema trivial, principalmente debido a la información privada de los jugadores. Esta información afecta, en la mayoría de los casos, al grado de preferencias sobre los resultados o decisiones finales que tiene cada jugador. El diseñador del mecanismo debe establecer unas reglas sin conocer las preferencias de los jugadores.

	C	NC
NM	$-c_{p2}$	$-c_{p2}$
M	$-C_p$	0

Cuadro 2.3: Beneficio para el jugador productor de tipo 2

2.2.2. Conceptos básicos

Para el diseño de mecanismos es fundamental introducir el concepto de información privada de un jugador. Cada jugador tiene unas preferencias sobre el resultado final, que depende de un estado propio.

Definición 2.2.1 (Información Privada)**.** El conjunto de elementos que influyen en sus preferencias, y que no son conocidos por los demás participantes, se denomina "Información Privada". La información privada de un jugador i se representa por θ_i que pertenece al conjunto de todos los tipos posibles para el jugador i Θ_i.

Cuando hay n jugadores, el vector formado por todos los tipos se representa por $\theta = (\theta_1, \cdots, \theta_n)$, siendo $\Theta = \times_i \Theta_i$. Resulta conveniente representar por θ_{-i} el vector de tipos de todos los jugadores excepto i.

Intuitivamente, el concepto de información privada refleja la idea de que los demás jugadores no conocen exactamente al jugador en cuestión. El único conocimiento que hay es que ese jugador puede ser de diversos tipos (incluso infinitos). Un ejemplo sencillo de esto se puede encontrar con nuestros jugadores consumidor/productor. Supongamos que tenemos dos posibles tipos de productores. Uno de ellos es el que hemos descrito anteriormente, mientras que el segundo tiene como utilidad la descrita en 2.3.

En este caso, decimos que el jugador productor tiene dos posibles tipos. El tipo 1 viene dado por el vector de utilidad $\theta_{p1} = (-c_p, -c_p, -C_p, 0)$ y el productor de tipo 2 viene dado por el vector de utilidad $\theta_{p2} = (-c_{p2}, -c_{p2}, -C_p, 0)$. El conjunto de tipos

del jugador productor sería por tanto: $\Theta_p = \{\theta_{p1}, \theta_{p2}\}$.

Nuestro problema ahora es que el consumidor debe decidir una estrategia, pero desconoce si su contrincante es del tipo 1 o del tipo 2. Sin más información, el consumidor no puede determinar una estrategia. En estos casos se suele suponer que hay un conocimiento previo sobre las probabilidades de que el productor sea de un tipo o de otro. En esos casos, el juego entra en el terreno de los juegos bayesianos.

Cuando los jugadores pueden tener información privada, esto es, pueden ser de un tipo desconocido para el resto de los jugadores, entonces es necesario reformular de nuevo todas las definiciones que se han visto en la introducción a la teoría de juegos.

Por ejemplo, las preferencias sobre los resultados finales se ven ahora como una función que toma como parámetro el tipo del jugador. Matemáticamente, la función de utilidad se define ahora como:

Definición 2.2.2 (Función de utilidad con información privada)**.** Cada jugador i tiene una preferencia definida sobre los estados del juego D, representada por una función de utilidad de la forma $v_i : D \times \Theta_i \to \mathbb{R}$.

Siguiendo esta notación, entendemos por $v_i(d, \theta_i)$ la utilidad que obtiene el jugador i del tipo $\theta_i \in \Theta_i$ cuando se concluye una interacción en el estado $d \in D$ (en el diseño de mecanismos se diría que se ha optado por una decisión d).

Según lo visto anteriormente de las funciones de utilidad, se entiende que $v_i(d, \theta_i) > v_i(d', \theta_i)$ indica que el jugador i del tipo θ_i prefiere la decisión d a la decisión d'.

Siguiendo con nuestro ejemplo anterior de consumidores/productores se podría decir que, $v_p(NoMiente - Comprueba, \theta_p) = -c_p$, mientras que $v_p(NoMiente - Comprueba, \theta_{p2}) = -c_{p2}$.

En un primer momento, se podría pensar que no es realmente necesario representar la función de utilidad como una función del tipo del jugador, ya que éste debe conocer su tipo, y por lo tanto no le aporta nada a la hora de elegir su estrategia. Este planteamiento es equivocado por varias razones. En un primer lugar, suponer que el jugador conoce su tipo es una hipótesis invalida en muchos casos. El jugador no

conoce su tipo hasta el último momento (para entender este punto debemos entrar a detallar el funcionamiento temporal de la interacción). Pero la razón más importante para construir la función de utilidad de esta forma es que los demás jugadores, y el propio diseñador del juego, deben disponer de una herramienta matemática que les permita contemplar *todas* las posibles alternativas del jugador.

En este punto, estamos suponiendo que la función de utilidad existe, y que representa las preferencias del jugador sobre los resultados finales. Más adelante veremos un tratamiento más completo de estos conceptos.

En este punto tenemos los elementos que forman la estructura básica con la que debe trabajar el diseñador del juego. A esta estructura básica se le denomina "entorno":

Definición 2.2.3 (Entorno)**.** Nos referiremos a la estructura $E = \langle N, D, (v_i)_{i \in N} \rangle$ como un entorno. Denotamos con $\mathcal{E}$ la clase de todos los posibles entornos.

El objetivo del diseñador del mecanismo es encontrar un juego tal que, dado un entorno cualquiera, se opte por el conjunto de decisiones o resultados finales que se ha considerado oportuno. El diseñador debe intentar diseñar un juego que "implemente" una regla de decisión social. Formalmente:

Definición 2.2.4 (Regla de Decisión Social)**.** Una regla de decisión social es una aplicación $\mathcal{E} \to D$.

La definición anterior de decisión social es bastante rigurosa, y, para el caso que nos ocupa puede simplificarse (suponiendo que los conjuntos N y D permanecen fijos). Obsérvese también que esta aplicación puede tomar la forma de una función, en cuyo caso se denomina *función de decisión social*.

Definición 2.2.5 (Función de Decisión Social)**.** Una función de decisión social es $d : \Theta \to D$, indicando la decisión (resultado final) $d(\theta) \in D$ como una función de θ.

A veces, el diseñador del juego se enfrenta a situaciones en las que es necesario incentivar a los jugadores para que opten por determinadas puntos de equilibrio. Una

	C	NC
NM	$-c_p + t_p$	$-c_p + t_p$
M	$-C_p$	0

Cuadro 2.4: Beneficio para el jugador productor con transferencias

forma de realizar esto es mediante un sistema de pagos/cobros. Entonces se dice que existe una función de transferencia.

Siguiendo con nuestro ejemplo de productores/consumidores, supongamos que el estado deseado sea que el productor no mienta y que el consumidor no tenga que comprobar. Podemos pensar en incentivar al productor con pago si no miente, o con un cobro si se detecta una mentira. La idea es que la acción de no-mentir sea preferible a la de mentir. Esto se ve reflejado en la tabla 2.4. En esta tabla se refleja mediante un pago t_p que le hace el sistema por no mentir. Se podría haber puesto que t_p fuera negativo (el sistema cobra al jugador) si se hubiera sumado en las valoraciones correspondientes a la mentira.

A este pago/cobro se le suele denominar *transferencia de utilidad*, ya que en realidad esa cantidad monetaria procede de las aportaciones de los otros jugadores. Sin embargo, esto no siempre es cierto. A veces, el sistema debe inyectar (subvencionar) o retirar el exceso de dinero.

Otro aspecto interesante de este ejemplo es que el diseñador de mecanismos ha conseguido que la balanza se incline por la opción de no-mentir (y por lo tanto por la de no comprobar). Es decir, ha inclinado el equilibrio de Nash alterando las probabilidades de la estrategia mixta. Dependiendo del valor de la transferencia, los valores de p y q que vimos anteriormente serán mas o menos altos. Lo que queremos resaltar es que el diseñador del juego puede influir en el resultado del mismo alterando las utilidades de los jugadores mediante sistemas de pagos (por ejemplo).

En realidad, el diseñador del juego debe enfrentarse a varios problemas simultáneos cuando desea que el juego resulte en un estado final determinado. Por una parte des-

conoce el tipo concreto de un jugador y por otra debe definir un sistema de pagos que, por normal general, suele estar relacionado con las valoraciones de los jugadores. Es decir, el pago suele ser la cantidad justa para que un jugador se incline por una estrategia dada, pero no suele tener sentido hacer que dicha cantidad sea arbitrariamente alta. Para determinar esa cantidad es conveniente conocer las valoraciones concretas y reales de los jugadores.

Para entender esto último, sigamos con nuestro ejemplo. Supongamos que el diseñador de mecanismos desea una determinada probabilidad en la estrategia mixta de los jugadores, pero desconoce si el productor es del tipo 1 o del tipo 2 (recordemos que es información privada). Si fuera del tipo 1, supongamos que el diseñador del mecanismo determina que el pago debería ser t_{p1}. Por otra parte, si el jugador fuera del tipo 2, el pago debería ser t_{p2}. El diseñador del mecanismo se encuentra con el hecho de que la función de transferencia es una función del tipo del jugador. Una función de transferencia es, por lo tanto una aplicación de $t : \Theta \rightarrow \mathbb{R}^n$ y donde $t_i(\theta)$ representa el pago o cobro que debe realizar el jugador i cuando los tipos de los jugadores son θ.

Cuando el mecanismo contempla pagos, la utilidad del jugador es la suma de su utilidad real más la transferencia (pago). Es decir:

$$u_i(d, \theta_i) = v_i(d, \theta_i) + t_i(\theta)$$

Como veremos, la teoría del diseño de mecanismos contempla todos estos conceptos y los relaciona: jugadores, preferencias, estados finales deseados, estrategias, tipo de equilibrio y transferencias.

Equipados con estas nociones y las definiciones anteriores, ya podemos introducir formalmente el concepto de mecanismo. Se entiende por tal a:

Definición 2.2.6 (Mecanismo)**.** Un mecanismo es el par (M, g), donde $M = M_1 \times ... \times M_n$ es el producto de todos posibles mensajes (o el espacio de posibles estrategias) y donde $g : M \rightarrow D \times \mathbb{R}^n$ es una función de resolución. Esto es, para posible elemento de mensajes o estrategias $m = (m_1, \cdots, m_n)$, se representa por

$g(m) = (g_d(m), g_{t,1}(m), ..., g_{t,n}(m))$ la función de resolución (que incluye posibles transferencias o pagos/cobros).

Veamos el significado de esta definición en detalle. Lo que estamos diciendo es que un mecanismo es un par de elementos: los mensajes (acciones, estrategias) de los jugadores y una función de resolución. Es decir, un mecanismo queda definido por el conjunto de posibles mensajes o declaraciones que realizan los jugadores, conjuntamente con una función que asigna para cada posible ocurrencia de estos mensajes a un resultado o estado final del juego.

La definición de mecanismo contempla un juego en el que los jugadores pueden optar por un sistema complejo de mensajes. Quizás un ejemplo concreto nos ayude a aportar algo de luz a esta definición. Imaginemos una subasta tradicional. El juego se podría construir de muchas formas: preguntando al jugador su valoración del bien, pidiendo que digan valores máximos y mínimos de adquisición, solicitando un orden de preferencia de los bienes a subastar, etc. Es decir, un mecanismo se podría diseñar de una forma muy amplia y contemplar muchas y diversas formas de mensajes. Como veremos, podemos limitar la búsqueda de mecanismos a aquellos en los que el espacio de mensajes se restringe a declarar el tipo del jugador.

En este ejemplo, la función de decisión es clara: se adjudica el bien al último postor que levante la mano, o al que mayor valoración haya declarado. Los estados finales del juego vienen definidos por dos elementos: el precio final de adjudicación y el postor. Este mecanismo se puede formalizar matemáticamente utilizando la definición anterior y definiendo claramente los conjuntos de mensajes M y la función de resolución g.

Nótese que existe una diferencia clara entre mecanismo y juego. Un mecanismo define una función de decisión y una función de transferencias, mientras que el juego viene definido por el conjunto de preferencias y su estudio produce un conjunto de equilibrios posible. Cuando en un mecanismo se fijan la función de decisión y la función de transferencia, entonces se dice que el mecanismo induce un juego.

El diseñador del mecanismo, en su proceso de diseño, debe preguntarse sobre la reacción de los jugadores ante una determinada función de preferencia (de utilidad).

Es decir, debe estudiar cómo serían las estrategias que realizarán los jugadores ante tal juego y cuál sería el resultado esperable. Al hacer este análisis, el diseñador del juego estará estudiando los *conceptos de solución* ("Solution Concepts").

Un concepto de solución estudia la(s) estrategia(s) que cada nodo realizará cuando se le enfrenta a un mecanismo (M, g) dadas unas preferencias concretas (Θ). Matemáticamente, un concepto de solución es una correspondencia S que identifica un subconjunto de M para cada mecanismo (M, g, Θ). Algunos autores prefieren hablar de concepto de solución en lugar de punto de equilibrio o solución de equilibrio. El diagrama de Stanley Reiter 2.2 representa esta idea.

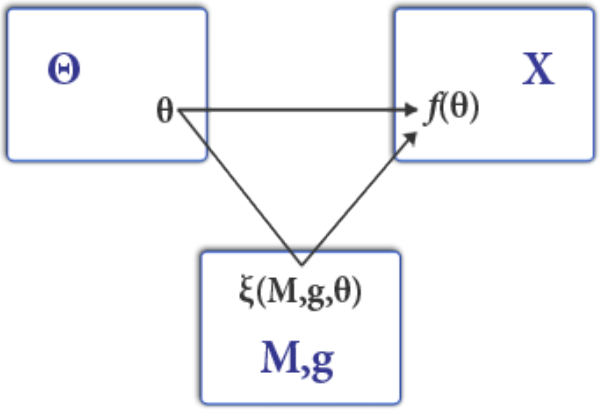

Figura 2.2: Mecanismo: Diagrama de Stanley Reiter

Un diseño de mecanismos debe establecer los estados finales que se desean obtener. En el ejemplo de la subasta, un estado deseable sería aquel en el que se adjudica el bien al que más lo valora y además se obtiene el máximo valor por el bien subastado. Recuérdese que los estados finales pueden ser conjuntos finitos o infinitos. El diseñador de mecanismos elige un subconjunto de tales estados como elementos preferidos o deseados. El concepto de función de decisión social está estrechamente relacionado con esta idea. La función de decisión social nos indica para cada θ el conjunto de estados deseados.

Finalmente, el diseñador de mecanismos debe estudiar si un mecanismo dado puede concluir en los estados finales deseados para cualquier posible conjunto de

preferencia. En ese caso, se dice que un mecanismo "implementa" la función de decisión social deseada y utilizando un concepto de solución. También se dice que la función social es "implementable" por un mecanismo bajo un determinado concepto de solución. Es importante notar que la noción de "implementabilidad" incluye el *concepto de solución*.

Uno de los principios básicos del diseño de mecanismos nos dice que dado un mecanismo cualquiera podemos obtener el mismo resultado si definimos el espacio de mensajes como el conjunto de tipos de jugadores. Es decir, podemos preguntar directamente a los jugadores sobre su tipo real (su información privada). Por ejemplo, un mecanismo de subasta muy utilizado es el que solicita a los participantes que escriban su valoración en un sobre cerrado. La subasta se adjudica al mejor postor (aunque el precio a pagar será el segundo mejor como ya se verá). Este mecanismo sería similar otro mecanismo basado en mensajes más o menos complejos (manos levantadas, etc.).

Este principio tiene el nombre de *principio de revelación directa* y es fundamental, ya que nos simplifica la búsqueda de posibles mecanismos implementables ([Jac99]). Formalmente es:

Lema 2.2.1 (Principio de Revelación)**.** *Si existe un mecanismo que implementa una función de decisión social f en estrategia dominante, entonces existe un mecanismo de consulta directa que implementa f en estrategia dominante. Es decir, f es implementable en estrategia dominante si y solo si f es implementable.*

Cuando se pregunta directamente por las preferencias o tipo de jugador, el espacio de mensajes M coincide con el de tipos Θ. En ese caso, el mecanismo trata de encontrar la función de decisión social f que incorpora una regla de decisión d y una función de transferencia t. Alteramos ligeramente la notación para poder distinguirla de la función de resolución g anterior. Normalmente la función de decisión social f trata de optar por una decisión óptima $d^* = f(\theta)$ según los tipos declarados por los jugadores $\theta = (\theta_1, \cdots, \theta_n)$. Para distinguir el tipo declarado del tipo real, se suele

emplear la notación $\hat{\theta}_i$ para indicar el valor declarado por el jugador; frente a θ_i que es su tipo real.

En la gráfica 2.3 se representa la idea de mecanismo directo. Los jugadores declaran cada uno un posible tipo $\theta_i \in \Theta_i$, y el mecanismo aplica a la regla de decisión $d(\theta)$ y de transferencia $t_i(\theta)$ para cada jugador i.

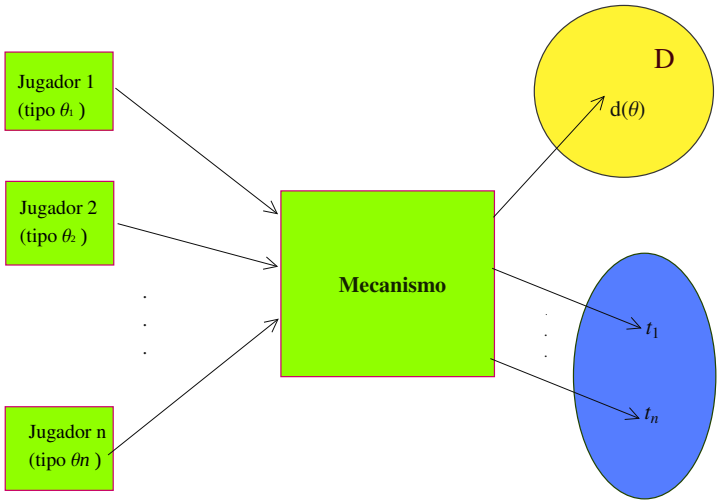

Figura 2.3: Mecanismo

Habitualmente se escoge como función de decisión social aquella que maximiza la suma de las utilidades de todos los agentes: $f(\theta) = argmax_{d \in D} \sum_i v_i(d, \theta_i)$, para todos los $\theta \in \Theta$.

A modo de resumen, lo que debe realizar un diseñador de mecanismos es:

1. Determinar los elementos que definen el juego: jugadores, preferencias, tipos de jugadores, información privada, etc..

2. Definir los estados finales posibles, y dentro de este conjunto, definir los estados deseables (función de decisión social)

3. Establecer el concepto de solución sobre el que se desea trabajar

4. Determinar el mecanismo que implemente la función social bajo el concepto de solución, utilizando para ello el principio de revelación directa.

Opcionalmente, se podrán establecer propiedades adicionales que deba cumplir el mecanismo. En la siguiente sección se analizan algunas de las propiedades más habituales en el diseño de mecanismos, siendo muchas de éstas parte de los objetivos de nuestro trabajo.

2.2.3. Desiderata.

En este apartado estudiamos algunas de las propiedades que se consideraran para los mecanismos propuestos en el presente trabajo. En general se buscarán que se cumplan algunas de las propiedades descritas en esta sección.

Uno de las características más deseadas para nuestro trabajo es que la asignación de los trabajos sea óptima, entendiendo como tal cuando el sistema adjudica las tareas a los nodos que tengan menor coste. En realidad veremos que la noción de óptimo es el resultado de minimizar una serie de ecuaciones sujetas a unas determinadas ligaduras o restricciones.

En su forma más simple, entendemos por óptima el concepto de *adjudicación eficiente* (en inglés "Allocative efficiency"), cuya definición es la siguiente:

Definición 2.2.7 (Allocative efficiency)**.** Se dice que una función de decisión social $f(\theta) = (d(\theta), t(\theta))$ es eficiente o adjudica eficientemente si para todas las preferencias $\theta = (\theta_1, \ldots, \theta_n)$:

$$\sum_i v_i(d(\theta), \theta_i) \geq \sum_i v_i(d', \theta_i) \tag{2.2.1}$$

para todo $d' \in D$.

Lo que nos dice la definición anterior es que el sistema intenta maximizar la suma de todas las valoraciones de los agentes optando por la alternativa más deseada. Esta definición nos será muy útil, ya que como veremos en el desarrollo de nuestro trabajo, buscaremos la adjudicación de trabajos a aquellos jugadores que declaren menor coste de ejecución. Aunque tendremos que utilizar alternativas a esta definición cuando queramos minimizar el coste sujeto a restricciones de justicia.

Otra definición muy útil refleja el hecho de que los jugadores desean participar en el juego. Es decir, si participan en el mismo, su utilidad será positiva. Esta noción se refleja por la siguiente definición de racionalidad individual (en inglés "Individual-rationality"):

Definición 2.2.8 (Individual-rationality)**.** La utilidad esperada de un jugador que participa en el juego es positiva siempre que siga una estrategia racional, e independientemente de las estrategias de los demás jugadores. Formalmente:

$$u_i(d(\theta), \theta_i) = v_i(d(\theta), \theta_i) + t_i(\theta) \geq 0, \forall \theta \in \Theta \tag{2.2.2}$$

Esta propiedad se suele denominar también como restricciones en la *participación voluntaria* (en inglés *"voluntary participation"*). Este concepto permite reflejar la idea de que los jugadores desean participar del juego ya que pueden obtener un beneficio positivo. Los jugadores pueden optar o no por jugar al mecanismo propuesto, y con esta definición se fijan las condiciones para que los agentes deseen participar.

Cuando los mecanismos tienen transferencias de utilidad, es interesante comprobar si la suma de los pagos cubre los cobros a realizar. Es decir, si el sistema no debe inyectar o retirar dinero de las transferencias. Esto se entiende como balance equilibrado (en inglés "budget-balance"), cuya definición se ofrece a continuación.

Definición 2.2.9 (budget-balance)**.** Una función de decisión social $f(\theta) = (d(\theta), t(\theta))$ se dice que tiene un balance equilibrado si para todas las preferencias $\theta = (\theta_1, \ldots, \theta_n)$ se cumple que:

$$\sum_i t_i(\theta) = 0$$

Es decir, no hay transferencias desde o hacia el exterior del sistema.

Finalmente, una de las propiedades más relevantes es la que nos dice que el concepto de solución se basa en decir la verdad independientemente de lo que digan los demás jugadores. Lo que esto quiere decir es que si un jugador decide alterar su estrategia unilateralmente, éste no obtendrá ningún beneficio, con independencia de lo que estén declarando los otros jugadores. A esta propiedad se le denomina en inglés "strategy-proofness".

Definición 2.2.10 (Strategy-proofness)**.** Se dice que un mecanismo es *strategy-proof* si el espacio de mensajes o estrategias (M) es idéntico al de tipos (Θ) y si el mecanismo implementa una solución bajo estrategia dominante.

$$u_i(d(\theta_i, \theta_{-i}), \theta_i) \geq u_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i), \tag{2.2.3}$$

$\forall \hat{\theta}_i \neq \theta_i, \forall \theta_{-i} \in \Theta_{-i}$

La propiedad anterior nos indica que un agente aislado no puede obtener mejor beneficio si modifica su declaración. Pero a veces puede interesarnos estudiar qué sucedería si se crean colaciones o agrupaciones de agentes que, de forma coordinada, manipulan sus declaraciones. En ese caso, se habla de otro concepto muy habitual : "group-strategyproofness". Este concepto nos dice que un agente no puede mejorar su utilidad sin empeorar a algún otro jugador de su grupo o coalición. Evidentemente, este concepto es más fuerte que el anterior.

Veamos la definición formal de lo anterior. Sea $S \subseteq 1, 2, \cdots, n$ un subconjunto cualquiera de jugadores, y sea $\theta_S = \theta_i | i \in S$ un vector de sus tipos declarados. Decimos que S dispone de una estrategia exitosa si existe algún vector $\hat{\theta_S}$ que, al ser declarado, puede mejorar a alguno de los miembros del grupo sin que los demás vean reducida su utilidad. Como de costumbre, dada una declaración θ, se representa por θ_{-S} el vector de estrategias de los jugadores que no están en S: $\theta_{-S} = \{\theta_i | i \notin S\}$.

Definición 2.2.11 (Group-Strategyproof)**.** Se dice que un mecanismo es *group-strategyproof* si el espacio de mensajes o estrategias (M) es idéntico al de tipos (Θ) y ningún grupo

dispone de una estrategia exitosa, i.e.,

$$\text{si } \exists i \in S \text{ tal que } \; u_i(d(\hat{\theta_S}, \theta_{-S}), \theta_i) \geq u_i(d(\theta_S, \theta_{-S}), \theta_i), \tag{2.2.4}$$

entonces

$$\exists j \in S \text{ tal que } \; u_j(d(\hat{\theta_S}, \theta_{-S}), \theta_j) \leq u_j(d(\theta_S, \theta_{-S}), \theta_j), \tag{2.2.5}$$

$\forall S \subseteq N, \forall \hat{\theta_S} \neq \theta_S, \forall \theta_{-S} \in \Theta_{-S}$

2.2.4. Otros conceptos relacionados.

Hemos visto la forma y notación de los mecanismos clásicos; lo que nos ha permitido introducir los conceptos básicos. Todos estos conceptos se verán modificados correspondientemente cuando se añadan nuevos aspectos, tales como conceptos de probabilidad, conocimientos previos, juegos repetidos, etc..

Cuando se trabajen con probabilidades, los juegos y los mecanismos se llaman bayesianos. En estos casos se entiende que alguno de los parámetros del juego proviene como realizaciones de una variable aleatoria que sigue una determinada distribución de probabilidad. Por ejemplo, se suele suponer que el tipo de un jugador θ_i proviene de una distribución de probabilidad $\sigma_i \in \Delta(\Theta)$. Según esta notación $\Delta(\Theta)$ representa el conjunto de todas las posibles distribuciones de probabilidad (con soporte en Θ), y σ_i representa una distribución concreta y particular.

Muchos de los conceptos anteriores se verán modificados para incluir nociones de esperanza. Por ejemplo, la propiedad de "individual-rationality" se modifica para indicar que un jugador desea participar en el juego si la esperanza de obtener una utilidad es positiva. En éste y en otros casos similares se suelen presentar alternativas a la formulación para distinguir diferentes conocimientos del jugador. No es lo mismo hablar de esperanza que hablar de la esperanza condicionada a que el jugador conoce algún parámetro del juego. Por ejemplo, antes de que el jugador realmente conozca su tipo, se puede calcular una esperanza de beneficio. Igualmente se podría calcular para cuando el jugador ya sabe su tipo, pero desconoce los tipos de los demás jugadores.

Finalmente, se podría hablar de la esperanza sabiendo los tipos de todos los jugadores (incluyendo el suyo propio). Para distinguir estas situaciones se suele prefijar los conceptos vistos en los apartados anteriores por "ex-ante", "interim" o "ex-post".

Por ejemplo, el concepto de "interim individual racionality", entendido como que la esperanza de obtener un beneficio positivo habiendo conocido su propio tipo sería:

$$\mathbb{E}[u_i(d(\theta), \theta_i)|\theta_i] = \mathbb{E}[v_i(d(\theta), \theta_i) + t_i(\theta)|\theta_i] \geq 0, \forall \theta \in \Theta \tag{2.2.6}$$

Mientras que el concepto de "ex-ante individual racionality" sería la esperanza de obtener un beneficio positivo (sin conocimiento previo):

$$\mathbb{E}[u_i(d(\theta), \theta_i)] = \mathbb{E}[v_i(d(\theta), \theta_i) + t_i(\theta)] \geq 0, \forall \theta \in \Theta \tag{2.2.7}$$

Salvo que se indique lo contrario, en el presente trabajo se hablará de los conceptos sin información previa ("ex-ante"). Aunque es el concepto más débil de los mencionados, en realidad es el que tiene más sentido en nuestro contexto (dado a la realización de múltiples de tareas).

Otro concepto importante en el diseño de mecanismos es el de "implementación virtual". Esta noción aparece cuando no es posible implementar o conseguir un estado final concreto, pero se puede diseñar un juego que nos permita acercarnos tanto como deseemos a dicho estado. Esa noción de "cercanía" implica añadir nuevas estructuras al conjunto de alternativas añadiendo alguna noción de probabilidad y haciendo que el juego termine en un estado dado con una determinada probabilidad suficientemente alta. Se suele utilizar nociones de distancia euclidea para determinar como de cerca está la solución probable de la deseada. Se dice entonces que la solución es "$\epsilon - cercana$" ($\epsilon - close$).

Finalmente, un tipo de mecanismo que tendrá utilidad en nuestro trabajo es el de "mecanismo de enlace", en el que se eliminan las transferencias de utilidades, aunque a cambio se incorporan varias realizaciones del mismo. El mecanismo de enlace se verá mucho más en detalle en el capítulo siguiente.

2.2.5. Otra notación relevante.

Para las demostraciones a veces se utilizará una notación más próxima al cálculo tradicional. Así por ejemplo se utilizará la variable x en lugar del coste del jugador c_i o de su tipo θ_i. De igual formas, utilizaremos la variable y para denotar el coste de un jugador contrario $c_j = c_{-i}$ que se corresponde con el tipo (información privada) de éste $\theta_j = \theta_{-i}$. Además, nuestro algoritmo aplica una transformada a los valores declarados. En ese caso, y dada una tarea, usaremos $x = PIT(\theta_i)$ para denotar el coste normalizado real $\bar{c}_i$ del jugador i para esa tarea, mientras que X o X_i se referirán a las variables aleatorias de dicho valor. Cuando los jugadores participen en QPQ podrán optar por publicar sus valores reales (normalizados) $x = \bar{c}_i$ o mentir y declarar valores falsos. En ese caso, utilizaremos z para referirnos a ese valor falso $\hat{c}_i$. Suponemos que estos valores z son realizaciones de alguna variable aleatoria Z. Dada una tarea con coste c_i, el jugador obtiene una utilidad normalizada igual a $\bar{u}_i = \bar{c}_i$ si no ejecuta la tarea (idependientemente del valor declarado) y realiza un esfuerzo normalizado igual a $\bar{w}_i = \bar{c}_i$ si debe ejecutar la tarea (donde $\bar{W}_i$ se refiere a la variable aleatoria asociada). Además, utilizaremos $y = \theta_{-i}$ para referirnos a menor valor publicado por el jugador agregado $y = \min \bar{c}_{-i}$. Utilizando la notación del diseño de mecanismos, decimos que la función de decisión social d de QPQ es:

$$d = \underset{i \in N}{\textit{argmin}}\ \bar{c}_i.$$

Recuerde el lector que la función *argmin* retorna la posición que ocupa el menor valor de sus argumentos. En caso de empate, decide de forma aleatoria.

Definimos $Pr[d = i]$ como la probabilidad de que el jugador i declare el menor valor (y que se le asigne la tarea). Cuando nos referimos al jugador agregado, Y es un vector de variables aleatorias y utilizaremos $Pr[Y \leq y]$ para referirnos a la probabilidad de que al menos uno de los elementos de Y, digamos j, asegure $Y_j \leq y$.

El algoritmo será el mismo para cada tarea que aparezca en el tiempo o en orden. En general se supondrá que el instante o la tarea está fijada y se denotará por k ese instante en el tiempo o el índice que ocupa la tarea.

2.2.6. Cuestiones.

Como nuevo paso en el progreso hacia el modelo de nuestro problema, ahora estamos equipados con conceptos y herramientas muy útiles, que podemos incorporar a nuestro estudio. Es por ello que tenemos que enfrentarnos nuevamente a preguntas de gran importancia:

1. ¿Existen diferentes tipos de jugadores en las redes distribuidas?
2. ¿Cómo podemos o debemos modelar esa tipología? ¿Existe alguna restricción razonable?
3. ¿Es posible auditar los costes declarados por los nodos?
4. ¿Debemos considerar algún sistema de negociación previa a la ejecución de tareas? o por el contrario ¿Existe alguna forma de asignar tareas que evite una larga y compleja negociación?
5. ¿Podemos imponer sistemas de incentivos que sean compatibles con cualquier tipología de jugador, sin siquiera conocerlos?
6. De todas las propiedades mencionadas, ¿Qué podemos exigir a nuestra solución? ¿Son alcanzables?

N	Conjunto de jugadores
i, j	Un jugador concreto
D	Conjunto de estados finales (o de decisiones)
d	Un estado final concreto (o una decisión concreta)
v_i	Utilidad del jugador i
u_i	Utilidad del jugador i (habitualmente incluye pagos)
Σ_i, S_i o M_i	Conjunto de acciones, de estrategias puras o mensajes del jugador i
s_i o m_i	Una estrategia pura o mensaje concreto del jugador i
θ_i	Tipo de un jugador, información privada de un jugador i
Θ_i	Conjunto de posibles tipos del jugador i
t_i	Transferencia de utilidad para el jugador i
$d(\theta)$	Regla de decisión
f	Función de decisión social (incluye la regla de decisión y la transferencia)
p, q	Probabilidades
f, F	Funciones de densidad y acumulada
$\Delta(\Theta_i)$	Conjunto de distribuciones de probabilidad con soporte Θ_i
σ_i	Una distribución de probabilidad concreta para el jugador i
c_i	Coste de una tarea concreta, se corresponde con el tipo de un jugador i
$\bar{c}_i$	Coste normalizado del jugador i
x, y	Variables de cálculo. Se corresponden con los valores normalizados.
$\hat{c}_i$	Coste falso declarado por el jugador i
z	Variable de cálculo que se corresponde con la mentira del jugador i
k	Ronda o índice de una tarea en el tiempo

Cuadro 2.5: Resumen de la notación utilizada

Capítulo 3

Formalización del modelo y soluciones.

> "The truth is incontrovertible, malice may attack it, ignorance may deride it, but in the end; there it is."
>
> Winston Churchill

3.1. Introducción

En este capítulo se formaliza el problema de ejecutar una serie de tareas en un entorno distribuido en el que los nodos tienen preferencias sobre las mismas. A lo largo del mismo se discutirán varios modelos de este problema, así como las soluciones propuestas utilizando mecanismos basados en pagos.

El interés del presente capítulo es múltiple. Por una parte sirve de base introductoria a los conceptos comentados anteriormente, aunque aplicados al problema que nos ocupa. Por otra parte, aborda una familia de soluciones o algoritmos que, aunque utilizan mecanismos basados en pagos, nos ofrecen un primer acercamiento a las características que estamos buscando en nuestra solución; tales como posibilidad de ejecución distribuida del algoritmo, poca complejidad del mismo, etc. Finalmente, este capítulo nos aportará las herramientas para la aplicación de la solución final

de nuestro trabajo en la que ligamos los mecanismos con pagos con los mecanismos ligados.

Como se ha visto anteriormente, lo que hace este problema diferente es que se desea asignar las tareas de forma eficiente teniendo en cuenta la declaración de coste que realizan todos los participantes, pero suponiendo que pueden mentir sobre estos valores. A continuación se expone y formaliza el problema, junto con todas las características que lo define.

Definición 3.1.1 (Ejecución de Tareas)**.** Un problema de la ejecución de tareas es una tupla $\langle T, N, (C)_{i \in N} \rangle$ donde:

1. $T = \{t_1, t_2, \ldots\}$ es un conjunto (no necesariamente finito) de tareas ordenadas (por ejempo por aparecer en el tiempo). Suponemos que las tareas son atómicas, independientes y de duración fija. Las tareas son conocidas y gestionadas en un instante de tiempo discreto. Se representa por t_k la tarea con posición k, siendo k el índice de dicha tarea.

2. $N = \{1, 2, \ldots, n\}$ es una lista ordenada de individuos o jugadores, donde N es finito,

3. $(C(t))_{i \in N}$ es un vector de funciones coste o de utilidad, en la que $C_i(t)$ representa el coste de ejecutar la tarea $t \in T$ por el nodo $i \in N$. Dicho coste únicamente lo conoce el jugador. El sistema percibe este valor como una variable aleatoria.

En cada uno de los elementos de esta definición se han aceptado una serie de características que requieren cierta puntualización. Entremos a analizar cada uno de estos supuestos.

Una de las primeras reflexiones sobre el modelo a desarrollar implica definir correctamente el conjunto de jugadores. En el modelo propuesto se presupone que

dicho conjunto es conocido de antemano. De hecho, esta suposición es razonable en muchos de los escenarios planteados. Si estamos pensando en un mundo virtual donde los jugadores son realmente personas u ordenadores con avatares virtuales, es evidente que el conjunto que hemos denominado N es conocido, ya que estaría formado por todos los nodos con avatar. En ejemplos que pudieran ser de aplicación en el mundo real esta suposición es aún más clara (partícipes de una empresa o de un hogar). Parece pues que, al menos en un primer momento, resulta factible desarrollar un modelo en el que el conjunto de jugadores es conocido de antemano. Sin embargo, no siempre podría ser cierto. Es habitual que en las redes P2P los nodos entren y salgan de la red de forma aleatoria. Incluso, si no hay un mecanismo de identidad claro, los jugadores pueden ocultarse detrás de un representante. Este problema se aborda al final de la tesis en el apartado 6.4.

Por otra parte, lo que realmente supone nuestro modelo es lo siguiente:

1. El coste de las tareas es la información privada del jugador. Es decir, es información que inicialmente está disponible únicamente para el jugador en cuestión. El jugador podrá decidir hacer publica o no esta información (puede mentir).

2. Las tareas son independientes entre ellas y la ejecución de una tarea no implica una variación en el coste de las siguientes.

3. Las tareas tienen un orden único y bien establecido. Este orden es conocido por todos los jugadores. En principio, este orden puede proceder del hecho de que las tareas se dan a conocer y se procesan en instantes de tiempo discreto. Este requisito es fundamental para que las herramientas estadísticas posteriores generen resultados idénticos con independencia del jugador que lo calcule.

4. Todos los jugadores están interesados en la ejecución de todas las tareas, pudiendo delegar la ejecución de la misma en otro jugador. Cuando una tarea se realiza, todos los jugadores pueden aprovecharse del resultado de la misma. Es decir, obtienen un beneficio por la ejecución de la misma.

5. Las tareas, una vez asignadas a un jugador, son ejecutadas correctamente. La

ejecución de las tareas es siempre correcta y los jugadores no pueden mentir o alterar el resultado.

Por otra parte, nuestro modelo admite ciertas variaciones que no alteran significativamente el resultado de nuestro trabajo:

1. En el tiempo aparecerán una o varias tareas. El hecho de que las tareas aparezcan agrupadas en el tiempo no influye en el resultado, siempre que sea posible asignar un orden a las mismas. Todos los jugadores deberán observar el mismo conjunto de tareas y el mismo orden.

2. Estas tareas pueden ser repetición de tareas anteriores o ser tareas nuevas. Si son tareas similares a las anteriores, suponemos que el contexto en el que aparecen es diferente. Lo importante en nuestro caso es que se desconoce el coste de las tareas en cada instante.

3. Los jugadores pueden conocer el coste de sus tareas cuando éstas son publicadas. Podemos suponer que el jugador tiene información previa de la distribución de costes, pero no sabe los costes concretos que aparecerán en el futuro. Es decir, los jugadores pueden conocer las distribuciones de costes, pero no las realizaciones concretas.

4. El modelo admite que un jugador sea no racional o limitado racionalmente, por lo podría desconocer el coste de una tarea. Puede estar limitado en su capacidad de cómputo y/o no tener conocimiento sobre cómo calcular los posibles equilibrios (Nash, bayesiano, etc).

5. Los jugadores podrían entrar y salir del juego. En la solución presentada en la presente tesis, hemos supuesto un número fijo de jugadores, pero es posible adaptar los resultados de nuestro trabajo para admitir número variable de jugadores.

Cuando reflexionamos sobre la construcción de un mundo virtual distribuido o de computación P2P, observamos que estas suposiciones parecen razonables. Por

ejemplo, podemos conocer de antemano que es necesario calcular o simular el efecto del viento en un determinado mundo virtual, pero realmente desconocemos cuanto puede costarle a un jugador ejecutar la tarea asociada a la simulación del viento. Incluso, si a un participante le cuesta una cantidad dada en un instante, este coste podría variar en el futuro si su contexto ha cambiado (dispone de menor ancho de banda o su CPU se encuentra saturada). También es fácil imaginar situaciones en las que pudieran aparecer tareas no previstas inicialmente. Por ejemplo, si es necesario realizar una búsqueda compleja, o la red se encuentra saturada y si es necesario una reorganización de la misma. Esto implicaría un proceso no previsto de antemano.

En cualquier caso, lo que sí parece una suposición razonable es que todas las tareas implican mejorar o mantener el sistema, y que todos los nodos están interesados en que dichas labores se realicen.

Una alternativa al modelo propuesto es que los nodos estén interesados parcialmente en la ejecución de las tareas. Por ejemplo, un nodo podría no estar interesado en el proceso de reorganización de la red en una zona que no le afecte directamente. Pero incluso en este caso, el modelo se podría plantear de igual forma, ya que el nodo en cuestión no formaría parte del conjunto de jugadores N en esa ronda. De igual forma, se podría plantear como alternativa que no todos los nodos sean capaces de realizar las tareas. En resumen, parece razonable pensar que el modelo se podría aplicar tanto al caso en el que los nodos no quieran o no puedan participar en una determinada ronda.

Finalmente, la definición anterior se basa en la hipótesis de que cada nodo tiene un coste para la ejecución de una tarea concreta. El coste de la ejecución de las tareas se considera información privada del nodo. Esto puede parecer razonable, ya que podríamos suponer que este coste viene definido por cierto porcentaje de consumo de CPU, memoria, etc. cuya medida únicamente conoce el nodo en cuestión. Como se vio en el capítulo anterior, el coste es una función de utilidad que refleja unas preferencias para la ejecución de tareas. Este coste no tiene que estar basado en una unidad monetaria común, y de hecho esta característica nos permitirá trabajar con dos tipos de modelos a desarrollar y estudiar. A saber, el que nos permita hacer transferencia de

utilidad (pagos y cobros) y aquel en el que no sea posible realizar esta transferencia.

Otro de los factores clave del modelo es que las tareas son independientes entre ellas y la ejecución de una tarea no implica una variación en el coste de las siguientes. Podemos entender que los valores de los costes es una variable aleatoria y que siguen una distribución de probabilidad concreta (aunque posiblemente desconocida). Por el momento no consideramos relevante que las tareas aparezcan de una en una o en conjuntos más o menos amplios, ya que esto afecta principalmente a la capacidad que tienen de modificar su creencia sobre los demás jugadores. Si a un jugador se le pide que declare los costes de un conjunto de tareas antes de conocer los costes de los demás jugadores, éste tendrá que decidir una estrategia con una mínima información. Si, por el contrario, las tareas aparecen en el tiempo de una en una, el jugador podrá crear una creencia ("belief") sobre los valores de los contrarios, predecir los futuros valores y adaptar de esta forma su propia estrategia. Como se verá posteriormente, ambos modelos serán tratados a lo largo de la presente tesis.

Para cada nodo, se genera una secuencia de costes $(C_i(t_1), C_i(t_2), \ldots, C_i(t_k), \ldots)$ donde $C_i(t_k) \in C_i$, que pueden entenderse como muestras independientes de una distribución de probabilidad $\sigma_i \in \Delta(C_i)$ propia de cada nodo. En esta definición C_i es el soporte de la distribución (rango de valores para el que la probabilidad de ese valor es diferente de cero); y $\Delta(C_i)$ es el conjunto de todas las distribuciones de probabilidad sobre C_i. A veces, se utilizar $c_{ik} = C_i(t_k)$ para indicar el coste de un jugador, siendo t_k una tarea (o un instante de tiempo) cualquiera. Cuando el tiempo no es relevante o se pueda inferir del contexto, se omitirá el subíndice k, y se escribirá c_i para expresar el coste declarado por el jugador i en un instante dado.

Pueden existir alternativas a la suposición de que un nodo disponga de una distribución de costes según una distribución de probabilidad desconocida por los demás (e incluso por sí mismo). Como alternativa, podríamos suponer que el jugador tiene varias distribuciones de probabilidad en el tiempo, conforme evoluciona su contexto o el propio juego. Como veremos más adelante, la modificación para hacer posible tal variación es relativamente simple ya que implica adaptar la velocidad de convergencia de los algoritmos propuestos.

Visto todo lo anterior, parece razonable que la función de utilidad de un jugador i sea,

$$v_i(d, \theta_i) = \begin{cases} 0 & \text{si ejecuta la tarea,} \\ c_i & \text{si otro jugador ejecuta la tarea.} \end{cases}$$

Es decir, un nodo obtiene una utilidad igual a su coste cuando se encuentra la tarea hecha por otro jugador y 0 cuando tiene que ejecutar él mismo la tarea. Se supone que el jugador valora el beneficio de obtener una tarea hecha igual al coste que le supondría realizarla él mismo. Este modelo es bastante general, pero no contempla otros beneficios indirectos. Por ejemplo, se podría pensar que el jugador podría obtener un beneficio adicional si carga de trabajo a un determinado jugador al que considera competidor; o que obtiene un beneficio por dañar el rendimiento del sistema. También existirían modelos alternativos que contemplaran comportamientos bizantinos (maliciosos). Este tipo de modelos no han sido estudiados en la presente tesis y creemos que serían de gran interés para trabajos de investigación futura.

También se ha supuesto que el jugador es neutral al riesgo y que el beneficio en el tiempo es simplemente una suma de las utilidades individuales de cada tarea. Es decir, no existe *factor de descuento*. En la teoría de la decisión o en la teoría de juegos se suele aplicar una factor de descuento que, multiplicado por los valores de coste, hacen que éstos pierdan (o ganen) valor en el tiempo. Es decir, no se considera igual un valor a día de hoy que dentro de un año. Su significado es similar al que habitualmente podemos entender por el interés financiero. Que nuestro modelo no refleje dicho factor de descuento puede entenderse como consecuencia lógica del modelo de generación de tareas mencionado anteriormente. Si suponemos que hay un número determinado de tareas que son conocidas de antemano, no tiene sentido introducir el factor de descuento. De igual forma, si las tareas aparecen en instantes de tiempo muy cercanos, podemos considerar despreciable el efecto del descuento. Sin embargo, cuando hablamos de juegos repetidos en el tiempo, es habitual introducir tal factor. Claramente, un modelo con un desarrollo temporal necesitaría que los jugadores tengan factores de descuento. Pero esto abre la puerta a nuevos modelos más complejos, ya que los jugadores podrían tener diferentes factores de descuentos, y además no nos parece evidente si éstos deberían ser o no considerados como in-

formación privada. En nuestro caso, y pensando en los ejemplos anteriores (mundos virtuales, etc.), se presupone que la utilidad del jugador no se degrada en el tiempo. Recordemos que en teoría de juegos a este tipo de jugador se le llama *jugador infinitamente paciente* ya que una alteración en el beneficio puede ser compensada en el futuro si existe un marco temporal suficientemente amplio. Un jugador podría ver mermada su utilidad en el momento actual pero recuperar la pérdida en el futuro si juega durante suficiente tiempo.

3.2. Desarrollo del modelo

En el contexto expuesto, y según los conceptos de diseño de mecanismos vistos anteriormente, el problema del diseño queda definido por los conjuntos:

1. **Conjunto de jugadores** Se corresponde con el conjunto N de la definición anterior 3.1.1.

2. **Información Privada** En este caso, Θ_i es $\sigma_i \in \Delta(C_i)$ y $\theta_i(t_k) = C_i(t_k) \in C_i$. Cuando no sea relevante el instante del tiempo k, escribiremos $\theta_i(t) = C_i(t) \in C_i$ y $\theta_i = c_i$.

3. **Conjunto de Decisiones** Es el conjunto de posibles asignaciones de la tarea. En general $D = 2^N$. En la versión más simple, suponemos que la tarea se asigna a un solo nodo o jugador. En ese caso, $D = \{1, 2, \ldots, n\}$ y d representa el nodo al que ha sido asignado la tarea.

4. **Función de utilidad** La función de utilidad del jugador i será,

$$v_i(d, \theta_i) = \begin{cases} 0 & \text{si } i = d, \\ c_i & \text{si } i \neq d. \end{cases}$$

5. **Función de decisión** La función de decisión deseada es la que garantice una decisión d tal que

$$\sum_{i \in N} v_i(d(\theta), \theta_i) \geq \sum_{i \in N} v_i(d', \theta_i)$$

para todo θ y $d' \in D$.

Según la definición anterior, el mecanismo queda perfectamente definido por su función de decisión ya que estamos buscando mecanismos que no necesiten de transferencias de utilidad. Nótese que la función de decisión busca maximizar la suma de las utilidades de todos los jugadores. Es decir, se desea un sistema óptimo; entendiendo como tal el sistema que minimiza el coste del sistema en su totalidad. En su forma más básica, deseamos un mecanismo que implemente una solución "allocative-efficiency".

Lo que se busca es implementar bajo estrategia dominante (2.1.8) un mecanismo que satisfaga las propiedades mencionadas en el apartado 2.2.3. A saber:

1. La función social optimice el coste del sistema (allocative-efficiency).
2. No necesite de una función de pago.
3. Que todos los jugadores obtengan un beneficio positivo (Individual-rationality).
4. Que sea resistente a jugadores poco o nada racionales.

Se requiere que el mecanismo implemente el sistema bajo estrategia dominante, y no bajo un equilibrio bayesiano de Nash. Recordemos que esto implica que el mecanismo garantiza un juego en el que la estrategia de un jugador no depende de las estrategias de los demás. Este punto es fundamental ya que un equilibrio bajo estrategia dominante nos garantiza muchas propiedades muy interesantes para nuestro modelo. Por ejemplo, este tipo de solución implica que el jugador puede calcular su curso de acción independientemente de si los otros jugadores son o no racionales. Incluso, nos permite trabajar con un modelo donde el aprendizaje no sería necesario. Una conclusión clara de utilizar este concepto de solución, es que se simplifica el análisis del juego repetido en el tiempo, pudiendo compararlo a juegos no repetidos. Como veremos más adelante, es posible construir el mecanismo deseado con muy pocas restricciones.

Todo lo anterior significa que estamos buscando un mecanismo que asigne de forma eficiente la tarea. Pero esto sólo es posible si conocemos los costes reales de

los nodos. Para ello, el mecanismo debe preguntar directamente a cada nodo su coste real ("revelación directa"). Dado que los nodos tienen incentivos para mentir, lo que deseamos es que el mecanismo sea strategy-proof y que, por lo tanto, los jugadores "deseen" declarar sus valores reales ya que eso irá en su propio beneficio. Además, los agentes desearán hacerlo con independencia de lo que declaren los contrarios.

Con el fin de aclarar todo lo visto hasta el momento, y antes de continuar con la solución propuesta, nos gustaría presentar algunas soluciones intuitivas, pero que es fácil demostrar que no cumplen con las propiedades buscadas. En particular, veremos que *no* cumplen con el objetivo de incentivar a los jugadores a declarar su tipo real. A continuación se propone una solución basada en pagos, más tradicional y siguiendo los mecanismos clásicos de subastas.

3.2.1. Algoritmos que no son strategy-proof

Para entender el problema veamos algunas de las soluciones en modelos más simples. Cuando el juego no es repetido, una posible solución sería la representada por el algoritmo 1:

Algorithm 1 Simple mechanism (code for node i)

1: Estimate the cost $\theta_i = c_i$ of the task
2: Publish the cost c_i
3: Wait to receive the costs c_j from other players $j \in N$
4: Let $d = \underset{j \in N}{\textit{argmin}}\ c_j$
5: **if** $d = i$ **then**
6: execute the task
7: **else**
8: do nothing (node d will execute the task)
9: **end if**

Este sería el primer algoritmo que pudiera pensarse si el contexto del sistema

distribuido fuera el habitual. Hasta hace poco, era normal pensar que los nodos pudieran fallar, modelando posibles tipos de fallos (desde fallos aleatorios hasta fallos bizantinos). En general, no era habitual pensar que los nodos en un sistema distribuido pudieran *engañar* al protocolo si con ello se consigue un beneficio. Este tipo de comportamiento egoísta es el que nos interesa contemplar y resolver. Es fácil ver como un nodo egoísta puede aprovecharse del algoritmo anterior. Dado que la tarea debe ser ejecutada por el nodo con menor valor declarado, el nodo puede ahorrarse la ejecución de la tarea diciendo valores muy grandes. En este caso la estrategia del nodo que maximiza su beneficio (via la reducción del coste) es mentir y declarar un coste superior.

Evidentemente, esta estrategia es también válida para los demás nodos si suponemos que todos son nodos racionales (que será lo habitual en nuestro trabajo). Si existe un valor máximo en el coste (p.e. $max(C)$), los nodos tenderán a declarar como coste dicho valor. Entonces decimos que tenemos un punto de equilibrio.

Ese punto de equilibrio no es un estado deseable por el diseñador del sistema distribuido ya que no nos garantiza la ejecución de la tarea por el nodo con menor coste. Si todos los nodos declaran ese coste máximo, el algoritmo tendrá que decidir la asignación mediante algún mecanismo aleatorio.

Lo que ha sucedido con este algoritmo es que hemos dejado libertad a los jugadores para declarar su tipo real, pero no existen los incentivos apropiados para que declaren la verdad. Si los jugadores son racionales y actúan de forma egoísta, lo más razonable será declarar valores que nada tienen que ver con la realidad. Eso ocasiona problemas colaterales como, por ejemplo, que el sistema no sea eficiente en su asignación de tareas. En el contexto de nuestro trabajo, decimos que estos algoritmos son *erróneos* ya que fallan en su intento de eliminar los efectos dañinos del comportamiento egoísta.

La solución clásica pasa por ofrecer algún tipo de incentivo a los nodos, de tal forma que mentir no sea rentable. Ese tipo de incentivo suele ser un pago (o cobro) que debe hacer el nodo en función de la decisión tomada.

Si no tenemos cuidado en definir correctamente el pago, nuestro nuevo algoritmo

podría resultar propenso a las mentiras. Veamos una versión nueva de nuestro algoritmo 2 que, aunque aparentemente constituye una versión mejorada, sigue teniendo problemas, como veremos.

Algorithm 2 Simple mechanism with Payments(code for node i)

1: Estimate the cost $\theta_i = c_i$ of the task
2: Publish the cost c_i
3: Wait to receive the costs c_j from other players $j \in N$
4: Let $d = \underset{j \in N}{\textit{argmin}}\ c_j$
5: **if** $d = i$ **then**
6: execute the task
7: recibe payment $t_i = t(c_i)(n-1)$
8: **else**
9: do nothing (node d will execute the task)
10: pay $t_i = -t(c_i)$ to node d
11: **end if**

La función de decisión d asigna la tarea al jugador con un menor valor de coste declarado. Recuérdese que $argmin$ está definida como la función que devuelve el índice que tiene el menor valor (y al azar en caso de empate).

En este algoritmo hemos introducido una función de transferencia que hemos denominado $t(\cdot)$. Antes de continuar, quisiéramos puntualizar que la notación t_k se utilizará para representar dos conceptos muy diferentes entre si. Nos referimos al hecho de que t_k podrá representar a una tarea en el instante k o ser una forma abreviada de $t_i = t(c_i)$, siendo éste el importe a transferir (pago/cobro) por un jugador. Confiamos que no sea fuente de confusión para el lector, ya que esperamos que el contexto sea el suficiente para determinar el sentido correcto en cada caso.

La versión más simple de dicha función podría ser $t(c_i) = \pm c_i$ (es negativo o positiva según el sentido del pago/cobro). Para los nodos que ejecutan la tarea será una

cantidad positiva que se suma a su utilidad, mientras que los nodos que se aprovechan de la ejecución de la tarea verán una cantidad negativa que se resta de su utilidad.

Pero ésta no es la única forma que podría tener la función de pago. Otra alternativa podría ser $t(c_d) = \pm\frac{c_d}{n}$. Es decir, la transferencia es una parte proporcional del valor declarado por el nodo ejecutor. Evidentemente, existe una amplia gama de funciones de pago (de hecho infinitas). La pregunta sería cuales de esas funciones consiguen eliminar los comportamientos egoístas de los nodos y obligar mediante estrategias dominantes a que digan la verdad. Veamos en concreto cuál sería el comportamiento de los nodos bajo este último sistema de transferencias.

El ejecutor de la tarea tiene una utilidad de 0 antes de recibir el pago, y tiene una utilidad de $v_d + t_d = 0 + c_d\frac{(n-1)}{n} = c_d\frac{(n-1)}{n}$ cuando recibe la transferencia.

Por su parte, los consumidores tiene un un beneficio igual al coste de la tarea que se han ahorrado menos el pago correspondiente: $v_i - t_i = c_i - c_d/n, \forall i \neq d$.

En el ejemplo anterior, un nodo no tiene interés en declarar valores inferiores al suyo real ya que podría suceder que se le asigne la ejecución de la tarea, y no ser compensado suficientemente por el trabajo. Es decir si el nodo declara $\hat{c}_i$, siendo $\hat{c}_i < c_i$, puede encontrarse con un beneficio de $0 + \hat{c}_i\frac{(n-1)}{n} < c_i\frac{(n-1)}{n}$.

El problema para nuestros objetivos arranca cuando el nodo analiza sus posibilidades de declarar $\hat{c}_i > c_i$. Y es que pueden darse varios casos según el nodo tenga o no información de los contrarios. Su estrategia dependerá de los valores contrarios y eso implicará que no es una estrategia dominante (algo que no deseamos). Veamos el razonamiento de tal afirmación. Si el nodo declara $\hat{c}_i > c_i$ puede ser que se le asigne o no la tarea. Cuando se le asigna la tarea, el jugador gana ya que recibe una transferencia proporcional a lo declarado y por lo tanto mayor. Si la tarea no se le asigna, únicamente tiene que pagar una parte de lo que ha solicitado el nodo productor (c_d),en cuyo caso su utilidad no se ve mermada por la mentira. Un tercer caso se da cuando se libra de ejecutar la tarea al decir $\hat{c}_i > c_i$ pero le hubiera correspondido si hubiera dicho su verdadero coste c_i. Esto es, existe otro jugador que ha declarado un valor c_d de tal forma que $\hat{c}_i > c_d > c_i$. La utilidad al decir su verdadero coste hubiera sido $0 + c_i(n-1)/n$ y al mentir será $c_i - c_d/n$. Pero dado que $c_d > c_i$ implica que

$c_i - \frac{c_d}{n} < c_i - \frac{c_i}{n} < c_i \frac{(n-1)}{n}$ y por lo tanto pierde mintiendo.

Es decir, cuando el jugador decide seguir una estrategia de mentir y declarar $\hat{c_i} > c_i$, puede encontrarse con diferentes resultados según el valor del contrario. Lo que nos dice el ejemplo anterior es que, dependiendo de los valores de los contrarios, el jugador puede optar por una estrategia u otra. Pero eso implica que decir la verdad no es una estrategia dominante y por lo tanto no es un algoritmo strategy-proof. Si el jugador tuviera cierto conocimiento a priori de los valores de los contrarios, éste podría crear una estrategia que no tiene que ser necesariamente la verdad. Este conocimiento lo podrá adquirir si juega de forma repetida al mismo, haciendo inviable su aplicación en nuestro contexto.

Este último algoritmo, en el que la transferencia es una función del valor declarado, se conoce como subasta al primer precio ("first-price auction"). Este, y otros modelos similares, han sido ampliamente estudiados en la teoría de las subastas, y se conocen sus principales propiedades así como los puntos de equilibrio. Se remite al lector al trabajo de Krishna [Kri02].

Dependiendo de las funciones de pago se puede o no conseguir eliminar el comportamiento egoísta y hacer que los nodos declaren sus tipos bajo estrategias dominantes. Existe una familia de mecanismos clásicos denominados mecanismos de VCG (en honor a sus autores Vickrey, Clarke y Groves) que, en esencia, aplican la idea de que el pago no debe depender del valor declarado por el jugador. Dichos mecanismos siguen el esquema de las subastas estudiadas por Vickrey [Vic61] y desarrolladas posteriormente por Clarke [Cla71] y Groves [Gro73]. Para una discusión sobre ese tipo de mecanismo, el lector puede consultar Tideman y Tullock [TT76].

Este tipo de mecanismo es el utilizado en las subastas a mano alzada. Se puede constatar que en estas subastas la persona que se adjudica el bien deberá pagar una valoración equivalente al valor que le asigna el participante más próximo. De hecho el adjudicatario no paga su propia valoración del bien. Todos estos sistemas se basan en el siguiente teorema de Groves.

Teorema 3.2.1 (Teorema de Groves)**.** *Dada la función de decisión social* $f(\theta) = (d^*(\theta), t_1(\theta), ..., t_n(\theta))$ *y siendo ésta eficiente, entonces* $f(\cdot)$ *es implementable bajo*

estrategia dominante si satisface el siguiente esquema de transferencias:

$$t_i(\theta) = (\sum_{i \neq j} v_j(d^*(\theta), \theta_j)) + h_i(\theta_{-i}) \ \ \forall i = 1, ..., n$$

donde $h_i : \Theta_{-i} \rightarrow \mathbb{R}$ *es cualquier función arbitraria que respeta la condición* $\sum_i t_i(\theta_{-i}) \leq 0 \ \ \forall \theta \in \Theta$.

Si observamos la forma anterior de la expresión de $t_i(\theta)$ podemos comprobar que esta función no depende del valor declarado por el jugador. El hecho de que se suela desglosar en dos sumandos es, otras razones, una conveniencia para facilitar su demostración posterior.

Las construcciones basadas en los mecanismos VCG parecen ofrecernos todo los que podríamos necesitar: los jugadores declaran su tipo real ("strategyproofness") y eficiencia. Desgraciadamente, los mecanismos VCG no tienen un balance de pagos equilibrado o nulo ("budget-balance"). Siempre generan un excedente o un déficit en el total de pagos/cobros. Esta es una de las principales conclusiones del estudio de Green y Laffont [GL79], en el que se demuestra que, en general, no existe un mecanismo strategyproof que sea a la vez budget-balanced y efficiente. Por otra parte, se sabe que los mecanismos VCG no suelen ser group-strategyproof.

En la siguiente sección haremos un rápido repaso a estos conceptos y plantearemos una solución basada en mecanismos VCG.

3.3. Mecanismos con pagos

3.3.1. Dos Jugadores a una ronda.

El mecanismo directo para dos jugadores que realizan un juego de una única ronda viene definido por $M_1 = (d, t)$ donde:

$$d = \underset{i \in D}{\textit{argmin}} \ \{c_i\} \tag{3.3.1}$$

y

$$t_i(\theta) = \begin{cases} +c_j & \text{if } i = d, \\ -c_j & \text{if } i \neq d, \end{cases} \tag{3.3.2}$$

donde j representa al jugador contrario, $j \neq i$. El algoritmo 3 sería el correspondiente al mecanismo anterior.

Algorithm 3 M_1: VCG mechanism with 2 players(code for arbitrary node i)

1: Estimate the cost $\theta_i = c_i$ of the task
2: Publish the cost c_i
3: Wait to receive the cost c_j from the other player $j \neq i$
4: Let $d = \underset{j \in N}{\textit{argmin}}\ c_j$
5: **if** $d = i$ **then**
6: execute the task
7: recibe payment $t_i = t(c_j)$
8: **else**
9: do nothing (node $d = j$ will execute the task)
10: pay $t_i = -t(c_j)$ to node j
11: **end if**

Por otra parte, la función de transferencia o de pago nos dice que el pago (o el cobro) es el valor declarado por el jugador contrario. Este mecanismo es un reflejo claro y directo de los mecanismos VCG. Al igual que todos los mecanismos de Grove, el mecanismo tiene las siguientes propiedades:

1. M_1 es **Strategy-proof** (Declarar los valores reales es una estrategia óptima para todos los jugadores),

2. M_1 presenta **Individual-rationality** Todos los jugadores tienen una utilidad positiva. Es decir, todos los jugadores están interesados en participar en el juego ya que siempre ganan,

3. M_1 es **Allocative-efficient (2.2.1)** y **no es budget-balanced**.

Lema 3.3.1 (Eficiencia)**.** *M_1 es eficiente. Ver definición de eficiencia en (2.2.1).*

El lema es fácilmente demostrable.

Demostración. Queremos demostrar que, por definición de eficiencia,

$$\sum_i u_i(d(\theta), \theta_i) \geq \sum_i u_i(d', \theta_i),$$

$\forall d' \in D$ y $\forall \theta \in \Theta$.

Dado que el conjunto D tiene sólo dos posibles alternativas, la demostración es sencilla. Llamando l y h a los nodos con el coste más bajo y más alto respectivamente. Denotemos con $l = d_l = d(\theta)$ la decisión de elegir l, mientras que $h = d_h = d'(\theta)$ denota la decisión de elegir h.

Supongamos, por reducción al absurdo, que es cierto lo contrario. Es decir, que el sistema no sea eficiente. Por lo tanto:

$$\sum_i u_i(d(\theta), \theta_i) = u_l(d_l, \theta_l) + u_h(d_l, \theta_h) < \sum_i u_i(d', \theta_i) = u_l(d_h, \theta_l) + u_h(d_h, \theta_h)$$

$$v_l(d_l, \theta_l) + t_l(\theta) + v_h(d_l, \theta_h) + t_h(\theta) < v_l(d_h, \theta_l) + t_l(\theta) + v_h(d_h, \theta_h) + t_h(\theta)$$

$$v_l(l, \theta_l) + t_l(\theta) + v_h(l, \theta_h) + t_h(\theta) < v_l(h, \theta_l) + t_l(\theta) + v_h(h, \theta_h) + t_h(\theta)$$

$$0 + c_h + c_h - c_l < c_l - c_h + 0 + c_l \qquad \text{por (3.3.2)}$$

$$2c_h - c_l < 2c_l - c_h$$

$$c_h < c_l$$

Contradiciendo, por tanto, que c_h sea el valor más alto. □

Lema 3.3.2 (Individual-rationality)**.** *M_1 satisface Individual-rationality, según la definición de (2.2.2).*

Demostración. De nuevo, dado que D sólo contiene dos posibles decisiones, la demostración vuelve a ser sencilla. Utilizando la notación de la demostración anterior, $c_h \geq c_l$. Entonces $l = d(\theta)$ y por la definición de $t_i(\theta)$ (3.3.2):

$$u_l(d(\theta), \theta_l) = v_l(l, \theta_l) + t_l(\theta) = 0 + c_h \geq 0$$

Por otra parte, si $h \neq d(\theta)$ entonces:

$$u_h(d(\theta), \theta_h) = v_h(l, \theta_h) + t_h(\theta) = c_h - c_l \geq 0$$

□

Lema 3.3.3 (Strategy-proof)**.** *M_1 es Strategy-proof (según lo definido en 2.2.3).*

La demostración vuelve a resultar muy sencilla e ilustrativa.

Demostración. Supongamos por contradicción que (d, t) no tiene una estrategia dominante. Esto significa que para algún jugador i, existe una declaración $\hat{\theta}_i$ diferente de su valor real θ_i tal que:

$$u_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i) > u_i(d(\theta_i, \theta_{-i}), \theta_i),$$

Pero eso es lo mismo que:

$$v_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i) + t_i(\hat{\theta}_i, \theta_{-i}) > v_i(d(\theta_i, \theta_{-i}), \theta_i) + t_i(\theta_i, \theta_{-i}), \tag{3.3.3}$$

Por la definición de t_i (3.3.2) es fácil comprobar que:

$$t_i(\hat{\theta}_i, \theta_{-i}) = v_j(d(\hat{\theta}_i, \theta_{-i}), \theta_j)$$

y

$$t_i(\theta_i, \theta_{-i}) = v_j(d(\theta_i, \theta_{-i}), \theta_j)$$

Por lo tanto, la ecuación anterior (3.3.3) es equivalente a:

$$v_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i) + v_j(d(\hat{\theta}_i, \theta_{-i}), \theta_j) > v_i(d(\theta_i, \theta_{-i}), \theta_i) + v_j(d(\theta_i, \theta_{-i}), \theta_j)$$

Pero esto contradice que M_1 sea eficiente y por lo tanto nuestra suposición era incorrecta. □

Uno de los problemas de esta solución es que el mecanismo no es equilibrado (la suma de los pagos y de los cobros no es cero). Es decir, un jugador paga menos de lo que el otro recibe. Es fácil de comprobar esta situación: el jugador que ejecuta la tarea recibe como pago el coste más alto, mientras que el que paga lo hace en función del coste del contrario, que es más bajo. En estos casos, es habitual pensar que el sistema subvenciona el juego (por ejemplo, sería el caso de un estado que subvencione a las empresas participantes en una subasta pública).

Continuemos con nuestra exploración de soluciones más complejas. Nuestro siguiente paso será analizar las soluciones con pagos para varios jugadores y a una única ronda. Aunque podremos encontrar muchas variantes interesantes dentro de la familia de mecanismos VCG, veremos que todas ellas siguen teniendo las características y limitaciones muy similares.

3.3.2. N jugadores-una ronda.

El mecanismo definido para varios jugadores y a una sola ronda vendría definido por $M_2 = (d, t)$ donde:

$$d = \underset{i \in D}{\textit{argmin}} \{c_i\}$$

y

$$t_i(\theta) = \begin{cases} +CS_i(min_{j \neq i}\{c_j\}, \theta_{-i}) & \text{if } i = d, \\ -CS_i(c_d, \theta_{-i}) & \text{if } i \neq d. \end{cases}$$

en la que la función $CS_i : \mathbb{R}^{n-1} \rightarrow \mathbb{R}$ nos indica el pago del jugador $i \neq d$. Esta función no depende del valor declarado por el propio jugador i y será estudiada a continuación. Para simplificar la redacción, y en este contexto se denominarán a los nodos que no ejecutan la tarea $i \neq d$ como "consumidores" y al nodo que ejecuta la tarea $i = d$ como "productor". No debemos confundir con la nomenclatura utilizada en el ejemplo del capítulo anterior.

Nótese que el mecanismo M_2 sigue el esquema del modelo VCG. Esto es, la función de decisión es óptima y la función de transferencias no depende del valor declarado por el jugador. En el caso de que el jugador sea el nodo productor, se le abonará el menor coste declarado por los contrarios. Si el jugador es un nodo consumidor, el pago lo repartirá entre todos los demás jugadores consumidores según lo que determine la función CS. Nótese que hemos forzado a que la función CS no dependa del valor del jugador.

Una primera forma de definir el reparto de coste es definir CS de tal forma que todos los consumidores aporten la misma cantidad hasta sumar la cantidad de c_d:

$$\forall i \neq d, CS_i(c_d, \theta_{-i}) = \frac{c_d}{n}$$

y el productor reciba una parte proporcional:

$$CS_d(min_{j \neq d} c_j, \theta_{-d}) = \frac{(n-1)}{n} min_{j \neq d} c_j$$

Realmente, este mecanismo es una de las posibles variantes que se podrían haber construido siguiendo el modelo establecido por VCG. Otra alternativa hubiera sido que cada nodo consumidor tuviera una transferencia $t_i = c_d$. Como se verá más adelante, existen propuestas algo más complejas y que garantizan propiedades interesantes.

Con esta definición del algoritmo, el jugador decidirá seguir una estrategia en la que decir la verdad es lo más conveniente para él. El mecanismo es strategy-proof, lo que es fácil de comprobar ya que es una aplicación directa de los mecanismos VCG.

De nuevo, parece inmediato la definición del algoritmo asociado a este mecanismo. Siendo éste el representado en el algoritmo de la figura 4.

Algorithm 4 M_2: VCG mechanism with N players(code for arbitrary node i)

1: Estimate the cost $\theta_i = c_i$ of the task
2: Publish the cost c_i
3: Wait to receive the costs c_j from other players $j \in N$
4: Let $d = \underset{j \in N}{\textit{argmin}}\ c_j$
5: **if** $d = i$ **then**
6: execute the task
7: recibe payment $t_i = \min_{j \neq i}(c_j)$
8: **else**
9: do nothing (node $d = j$ will execute the task)
10: pay $t_i = -c_d/(n-1)$ to node d
11: **end if**

Aunque no se incluyen demostraciones formales, el mecanismo M_2 cumple con siguientes propiedades:

1. M_2 es **Strategy-proof**: La declaración de los valores reales es una estrategia óptima para todos los jugadores,

2. M_2 es **Individual-rationality**: Todos los jugadores obtienen una utilidad mayor que cero,

3. M_2 es **Allocative-efficient**.

Es fácil comprobar esta afirmación ya que por ser un mecanimso VCG, es strategy-proof. Además, el sistema siempre opta por el jugador con el valor más bajo, y por lo tanto es eficiente. Finalmente, el sistema de pagos hace que los jugadores en M_2 siempre tengan una utilidad positiva. Cuando son productores, reciben una compensación positiva. Cuando son consumidores, siempre pagarán una cantidad inferior al trabajo ahorrado.

La ventaja del enfoque anterior es que es fácil de calcular, incluso de forma distribuida, y sobre todo, que trata por igual a todos los nodos al repartir el pago de forma equitativa. De todas formas, pensamos que sería deseable que la función de reparto de pagos tuviera en cuenta dos aspectos interesantes. Por una parte, el mecanismo propuesto trata por igual a los jugadores pero podrían ser deseables otras nociones de justicia, por ejemplo haciendo que los jugadores más relevantes paguen proporcionalmente más que el resto. Por otra parte, se sabe que los mecanismos VCG no tienen un sistema de pagos equilibrados. De hecho, es imposible encontrar un mecanismo con pagos balanceados y que, al mismo tiempo, sea strategy-proof y eficiente. Como consecuencia de la imposibilidad de conseguir pagos balanceados, una posible solución es redistribuir los pagos reduciendo al mínimo el desequilibrio producido por los mismos. Siendo este aspecto un tema ampliamente estudiado en la teoría de juegos, conocidos bajo el epígrafe de algoritmos VCG de redistribución de pagos o mecanismos "justos" de Groves ("Fair Groves Mechanisms").

En la siguiente sección se realizará una revisión del estado del arte de estos mecanismos, que pueden ser aplicados tanto a nuestro problema como a una generalización del mismo.

3.3.3. Mecanismos VCG con reparto de pagos.

Si recordamos la definición de mecanismo de Groves 3.2.1 la función h_i era cualquier función arbitraria que no dependía del valor declarado por el jugador i. El diseñador del mecanismo se encuentra con una amplia libertad para el diseño de esta función y, por tanto, del mecanismo en general. Ante tal abanico de posibilidades, es interesante utilizar nuevos criterios que nos puedan guiar en el diseño del mismo. Del contenido de esta sección queremos resaltar dos características que han resultado ser interesantes para el desarrollo de nuestro trabajo posterior:

1. Por una parte, ver cómo los mecanismos VCG se pueden aplicar a una forma más general del problema que nos ocupa, y en los que varios nodos pueden ser los responsables de la ejecución de la tarea.

2. Por la otra, profundizar en la idea de que no existe un criterio único para determinar la bondad del algoritmo. En concreto, la noción de justicia se verá afectada por esta libertad.

Veamos estos dos puntos en más detalle.

En nuestro trabajo central a la presente tesis hemos centrado nuestra atención en el problema de que varios nodos asignan la tarea a un único agente, confiando en que la ejecución de la tarea se realizará de forma cierta, sin problemas y pudiendo aprovecharse de los resultados producidos. Sin embargo, es probable que bajo determinados contextos, el conjunto de nodos deseen que la ejecución de la tarea se realice de forma conjunta o independientemente por varios nodos. Por ejemplo, si pensamos que existe una cierta probabilidad de fallo y deseamos disponer de sistemas de tolerancia a fallos. En estos escenarios, el sistema desea asignar la tarea a varios nodos, intentando mantener la eficiencia de la ejecución. Otro escenario posible sería cuando la ejecución de la tarea es compleja o necesita de cierto tiempo y que sea necesaria su ejecución en paralelo. En esta variante del modelo, seguimos suponiendo que existe un coste para la ejecución de la tarea, que es información privada y que es independiente del factor que determina la redundancia en la ejecución de la tarea (por ejemplo, es independiente de la probabilidad de fallo o del tiempo de ejecución).

En esta variante del modelo, el sistema determina cuántos nodos deben ejecutar la tarea. A partir del coste publicado, el mecanismo asigna la ejecución de la tarea a varios productores. La utilidad de los nodos es idéntica a la que hemos expuesto hasta el momento. Es decir, los nodos tienen una utilidad de 0 si ejecutan la tarea y de $\theta_i = c_i$ si pueden utilizar el resultado de la tarea sin trabajar.

Todos los artículos revisados para el presente trabajo consideran escenarios más o menos generales, principalmente adaptados a sistemas de subastas o distribución de bienes. Aunque algunos de ellos presentan modelos muy cercanos a nuestro problema no se corresponden exactamente con nuestro planteamiento. Nuestro modelo de ejecución de tareas se puede ver como una variación del problema conocido como *asignación de múltiples unidades bajo demanda única* (“multi-unit single-demand”). Estos modelos se denominan así ya que suponen la venta de varios bienes (“multi-

unit") pero cada agente únicamente consume un bien ("single demand"). Este modelo contempla la situación en la que se desean vender o adjudicar m bienes entre n agentes que ven mejorada su utilidad por la posesión del mismo. En nuestro caso, lo que deseamos es identificar a m nodos que se verán beneficiados por la ejecución de la tarea sin tener que trabajar por ello. Por otra parte, también se puede ver como el problema de asignar la tarea a $n - m$ nodos que debido al coste de ejecución tienen un beneficio igual a cero, de tal forma que es necesario compensarlos mediante un sistema de pago. Utilizando la notación utilizada en los modelos económicos se pueden entender que los nodos productores son los "vendedores" y los nodos consumidores son los "compradores". En general, los artículos revisados no contemplan un modelo en el que los jugadores puedan ser a la vez compradores y vendedores, ya que suponen que un centro vende los bienes a un posible conjunto de compradores. Este centro decide a quién asignar la "venta" y cobrar en consecuencia. Realmente la principal implicación que esto puede tener es que la mayoría de los trabajos entienden que la redistribución de los pagos nunca podrá originar una pérdida para el vendedor (suma de pagos negativa). Este hecho no es relevante para nuestro modelo ya que no existen elementos externos y parece plausible que el sistema de pagos sea deficitario. Nuestro modelo es, pues, ligeramente diferente y por lo tanto requerirá aclarar y adaptar, en su caso, algunos de los planteamientos originales presentados por los trabajos revisados.

Según hemos podido comprobar, existen un número reducido de trabajos relacionados y que podemos agrupar principalmente en cuatro grandes familias de mecanismos, considerando que constituyen el grueso de los trabajos publicados en este contexto. A continuación, se revisaran estos trabajos. Aunque nuestra revisión puede que no sea exhaustiva, a nuestro entender creemos que cubren el estado del arte y reflejan la mayor parte de los conceptos utilizados. La principal diferencia entre estos trabajos radica en el objetivo que tratan de alcanzar. Dada la gran libertad que nos ofrecen los mecanismos VCG, cada uno de los trabajos expuestos intentan conseguir una redistribución de los pagos intentando conseguir algún objetivo secundario que consideran relevante. En concreto veremos que los objetivos tratados se pueden resumir en:

1. Un mecanismos que trate a los jugadores bajo cierto criterio de justicia de tal forma que la redistribución de los pagos se haga teniendo en cuenta ese factor. La principal dificultad en este punto es el hecho conocido de que no existe una única definición de justicia ("fair").

2. Un mecanismo que se acerque al equilibrio lo máximo posible en escenarios del peor caso. Esta familia de algoritmos de redistribución estudian la peor situación posible e intentan minimizar el desequilibrio. (son "worst-case optimal").

3. Por último, una familia de mecanismos que intentan reducir el desequilibrio en esperanza. Suponiendo un cierto conocimiento sobre la distribución de valores, el mecanismo reduce la esperanza del desequilibrio.

A continuación se presentan los principales grupos de mecanismos considerados en nuestra revisión. En todos ellos, el mecanismo pide las valoraciones de los nodos (θ) y suponemos que la numeración de los nodos se corresponden con el orden de valoración $\theta_1 \geq \theta_2 \geq ... \geq \theta_n$. Dado que los mecanismos VCG son strategy-proof podemos suponer que los valores declarados se corresponden con su valoración real ($\hat{\theta}_i = \theta_i$). El mecanismo adjudica la tarea a los $n - m$ nodos con valores más bajos (siendo estos los productores) y el resto de los agentes m se benefician de los resultados de la tarea sin haber dedicado un esfuerzo a su realización (consumidores). Una vez realizada la asignación de la tareas entre los nodos, éstos se cruzan unos pagos compensando el esfuerzo de la realización.

Desde un punto conceptual se pueden entender estos mecanismos como un mecanismo VCG en dos fases:

1. Se ejecuta un mecanismo VCG clásico en el que los nodos $m+1, .., n$ ejecutan la tarea y en el que los nodos $1, 2, ..., m$ reciben los resultados de la tarea pagando θ_{m+1} cada uno. En total el sistema mueve un volumen de transferencias igual a $m \cdot \theta_{m+1}$.

2. Se procede al reparto del desequilibrio anterior, haciendo una transferencia z_i a cada nodo. Para determinar este valor, no se tiene en cuenta el valor del jugador en cuestión θ_i.

Evidentemente, estas dos fases son meramente ilustrativas ya que el mecanismo decide los pagos de forma unificada. Sin más comentarios, pasemos pues a revisar los mecanismos de redistribución de pagos.

Mecanismo de Moulin

Este mecanismo se basa en la propuesta publicada recientemente por H. Moulin en [Mou10]. Lo hemos incluido en primer lugar por ser uno de los más recientes y por ser uno de los más cercanos a nuestro problema (considera un único productor o ejecutor de la tarea). En concreto se remite al lector al apartado 4 del artículo, en el que se estudia la asignación de una tarea no deseada.

En el artículo de Moulin se utilizan conceptos típicos de las subastas al segundo precio (o subastas Vickrey). Estas subastas no dejan de ser mecanismos VCG y son muy habituales en las adjudicaciones con sobre cerrado. Se trata de que el vendedor adjudica la venta al mejor postor, pero recibe como pago lo que ha dicho el segundo. En este contexto, el excedente ("surplus") producido por la venta es la suma de todo lo que obtienen los participantes. En el caso de las subastas Vickrey, el excedente coincide con la valoración del comprador (c_1). Esto es debido a que el vendedor tiene como beneficio $0 + c_2$, mientras que el comprador tiene $c_1 - c_2$. El resto de los participantes no tienen beneficio. Por lo tanto el sistema ha tenido un excedente de c_1. La idea de Moulin es poder repartir esa cantidad de forma proporcional entre todos los participantes de la subasta. La pregunta a la que responde el trabajo de Moulin es si es posible, para un $\mu \in [0, 1]$ dado, que todos los compradores obtengan μc_1 mientras que el vendedor reciba $(1 - \mu)c_1$.

Se trata por tanto de buscar el mecanismo con el menor desequilibrio en los pagos posible. Para medir la bondad del mecanismo, cuando se intenta reducir al mínimo el desequilibrio producido por la suma de los pagos, parece conveniente definir algún tipo de coeficiente indicador de dicha bondad.

En su trabajo, Moulin define un coeficiente de (in)eficiencia que permita acotar el nivel de desequilibrio. Es decir, el mecanismo de Moulin desea acotar esta cantidad según un parámetro ρ tal que:

$$0 \leq \sum_i V_i(c) - c_1 \leq \rho c_1 \; \forall c \in \mathbb{R}_+^N$$

Claramente $0 < \rho \leq \infty$.

Con estas definiciones, Moulin propone el siguiente índice de eficiencia:

$$\rho = \underset{c \in \mathbb{R}_+^N}{sup} \frac{\sum_N V_j(c) - c_1}{c_n - c_1}$$

La razón de ser de esta definición se basa en la consideración de que la cantidad $c_n - c_1$ se considera como la diferencia entre el peor desequilibrio posible (c_n) y el óptimo (c_1). Aunque esto es cierto en el modelo de Moulin, más adelante veremos que tendremos que hacer pequeños cambios a esta definición para adaptarlo a nuestro modelo, principalmente por la diferencia de utilidad asignada a los agentes.

En su trabajo, también se utiliza una versión del concepto "q-justicia", muy similar a la que se verá en la siguiente sección, aunque adaptada a este modelo.

Definición 3.3.1 (q-Justicia)**.** Se dice que un mecanismo alcanza "q-justicia", para un entero q, con $1 \leq q \leq n$, cuando la utilidad neta de un nodo (incluyendo las transferencias):

$$V_i(\theta) \leq \frac{c_q}{n}, \forall i \in N, \theta \in \Theta$$

Lo que nos dice esta noción de justicia es que la utilidad neta de un nodo es un porcentaje del valor que ha dicho otro jugador. El ideal sería que el sistema fuera 1-justo, ya que esto indicaría que el jugador tiene una utilidad proporcionalmente a su propio valor. Desgraciadamente, Moulin demuestra que no es posible construir un sistema 1-justo ni 2-justo que sea strategy-proof.

Además demuestra que el mejor coeficiente que se puede obtener para un mecanismo q-justo con $q = 3, ..., n$ es igual a $\rho = \frac{n-1}{2^{n-1}-1}$ si n es impar y $\rho = \frac{n-1}{2^{n-1}-2}$ si n par.

Definiendo $\bar{q} = q - 1$ si q es impar y $\bar{q} = q - 2$ si q es par, y con las consideraciones anteriores, el mecanismo queda definido por:

$$V_i^q(c) = c_1 - \sum_{k=1} \bar{q} \frac{(-1)^{k-1} B_{n-1}^{k,\bar{q}}}{\binom{n-2}{k-1} B_{n-1}^{1,\bar{q}}} c_{-i}^k$$

Donde $B_m^{k,k'}$ se define como:

$$B_m^{k,k'} = \sum_{l=k}^{k'} \binom{m}{l}, \;\; 0 \leq k \leq k' \leq m$$

Este mecanismo tiene la ventaja de que se acerca rápidamente al ideal de desequilibrio cero conforme aumenta el número de jugadores, ya que el coeficiente de desequilibrio es una función de $\frac{1}{2^n}$.

Mecanismos de Bailey y Cavallo

A continuación veremos una familia de mecanismos propuestos por Bailey y Cavallo en [Bai97] y [Cav06]. En su formulación original este tipo de mecanismos redistribuye parte del pago realizado por los m agentes que reciben el bien (y que realizan el pago). En su propuesta, el mecanismo devuelve a los m agentes una cantidad igual a $\frac{m\theta_{m+2}}{n}$ y a los $n - m$ nodos que no obtienen el bien una cantidad igual a $\frac{m\theta_{m+1}}{n}$. Una forma de verlo es que a cada agente se le devuelve una cantidad proporcional al valor del agente que ocupa la posición $(m + 1)$ sin contar con el propio jugador. La intuición del mecanismo es la siguiente. El mecanismo calcula cual es el ingreso mínimo que podría obtener el sistema si un jugador i pudiera mentir y el resto de los jugadores mantienen su valor real θ_{-i}. Los mecanismos de Bailey y Cavallo reparten una parte proporcional $1/n$ de esta cantidad mínima. Formalmente, el mecanismo reparte a cada agente i una parte proporcional igual a $\frac{\mathcal{G}_i(\Theta_i, \theta_i)}{n}$ donde :

$$\mathcal{G}_i(\Theta_i, \theta_i) = \min_{\theta_i \in \Theta_i} \sum_{j=1}^{n} (v_{-j}(\theta_{-j}, d^*(\theta)) - v_{-j}(\theta_{-j}, d^*(\theta_{-j})))$$

A continuación veamos un ejemplo que aporte luz sobre lo anterior y que nos servirá como referencia para comparar los trabajos estudiados. Supongamos cinco

jugadores y que queremos que dos de ellos ejecuten la tarea. Esto es similar al un proceso de subasta en el que se quiere asignar tres bienes. Las valoraciones de los jugadores, así como los pagos, utilidades, etc del ejemplo se reflejan en la tabla 3.1. En el ejemplo, la asignación del bien le corresponden a los tres nodos que más valoran el bien (en el ejemplo los agentes $3, 4$ y 5), pagando cada uno de ellos el siguiente valor más alto 30. Los nodos que reciben el bien, tiene una utilidad igual a su valoración menos el pago realizado. El sistema ha recibido en total una cantidad de $3 \cdot 30$, redistribuyendo una parte de nuevo hacia los jugadores. La suma de lo distribuido es igual a $2\frac{3 \cdot 30}{5} + 3\frac{3 \cdot 20}{5} = 2 \cdot 18 + 3 \cdot 12 = 72 < 90$.

Jugador	Valoración	Pago	Utilidad	Redistribución	Utilidad final
1^*	90	-30	90-30	$\frac{3 \cdot 20}{5} = 12$	90-30+12
2^*	70	-30	70-30	$\frac{3 \cdot 20}{5} = 12$	70-30+12
3^*	60	-30	60-30	$\frac{3 \cdot 20}{5} = 12$	60-30+12
4	30	0	0	$\frac{3 \cdot 30}{5} = 18$	0+18
5	20	0	0	$\frac{3 \cdot 30}{5} = 18$	0+18

Cuadro 3.1: Ejemplo del mecanismo de Bailey-Cavallo

En el caso que nos ocupa, podemos entender que la tarea ha sido ejecutada por los nodos 4 y 5, mientras que el resto del sistema se beneficia del resultado de la ejecución.

Además de las propiedades habituales de los mecanismos de Groves, sus autores demuestran otras características adicionales. Por una parte, son claramente strategy-proof y eficientes (dado que son VCG). Además garantizan utilidades positivas para todos los jugadores (son "ex-post individual rational"). Pero una de las propiedades más interesantes es que la cantidad no redistribuida siempre es positiva y tiende a cero conforme el número de jugadores crece. Es decir, para un número elevado de jugadores, el mecanismo se acerca al concepto ideal de budget-balanced según m/n. Como

veremos, existen otros mecanismos que se acercan al punto de pagos equilibrados con mejores ratios de convergencia.

Mecanismos de Moulin, Guo y Conitzer

Otro de los trabajos más relevantes y que abren una gran familia de mecanismos son los propuestos independientemente por Moulin y por Guo y Conitzer. En este trabajo, los autores proponen una solución que denominan *mecanismo VCG de redistribución lineal* (´´Linear VCG Redistribution Mechanism") [GC07] y citeMoulin07efficient. Como veremos la idea es construir un sistema lineal de redistribución que puede ser optimizado mediante herramientas de optimización matemática tradicionales (programación linear matemática).

La idea de su trabajo se basa en la misma idea de que el mecanismo "devuelve" parte del pago a los jugadores buscando un sistema de pagos balanceado. En este caso, los autores siguen un enfoque similar a los planteados con el estudio del precio de la anarquía. Es decir, intentan estudiar las propiedades del mecanismo cuando ocurre el peor escenario posible ("worst-case") y demuestran que en ese caso, su propuesta es óptima.

La cantidad que devuelve el mecanismo al jugador i se puede expresar como

$$z_i = c_0 + c_1\theta_1 + \cdots + c_{i-1}\theta_{i-1} + c_{i+1}\theta_{i+1} + \cdots + c_n\theta_n$$

Es decir, un nodo consumidor ve reducido su pago en una cantidad que viene determinado por un vector de constantes $c_0, c_1, ...$ multiplicado por los valores que han dicho los otros jugadores. Obsérvese que no se incluye el valor declarado por el jugador.

Un punto interesante que demuestran sus autores es que el mecanismo de Bailey-Cavallo se puede entender como un caso particular de la expresión anterior.

Para que no exista déficit la suma de la redistribución de pagos no debe superar el exceso del pago ("surplus"). Con estas consideraciones, los autores proponen un sistema de programación lineal en el que se desea minimizar la cantidad de pagos no redistribuidos sujeto a las restricciones comentadas (no existe deficit, todos los jugadores tienen utilidades positivas, es strategy-proof, etc.). La solución a este sistema

de programación lineal aporta los valores de las constantes c_i:

$$c_i^* = \frac{(-1)^{i+m-1}(n-m)\binom{n-1}{m-1}}{i \cdot \sum_{j=m}^{n-1}\binom{n-1}{j}} \frac{\sum_{j=i}^{n-1}\binom{n-1}{j}}{\binom{n-1}{i}} \quad i = m+1, ..., n-1$$

Con estos resultados,y cuando $n \geq m+2$ los autores demuestran que su mecanismo es óptimo en el escenario peor y respeta las condiciones impuestas.

Mecanismos con k-justicia

Estos mecanismos, propuestos por Atlamaz y Yegin en [AY08], buscan el reparto del pago de forma justa. Dado que no existe una noción unificada de justicia, ellos proponen utilizar un concepto de justicia parametrizada basada en el trabajo de Porter et al [PST04]. A este concepto le llaman *k-justicia* ("k-Fair") donde k es un valor entre 1 y n y lo que expresan es el hecho de que la utilidad final de un agente siempre será mayor que una parte proporcional del valor declarado por el agente que ocupa la posición k más alta. La idea de utilizar una noción de justicia parametrizada fue introducida inicialmente por Porter. Formalmente, en su trabajo definen k-justicia (k-Fairness) cuando para cada posible vector de costes $c \in C$ y para cada $i \in N$ se cumple que la utilidad del jugador $u(\psi_i(c); c_i) \geq \frac{-c[k]}{n}$

Mecanismos óptimos en esperanza: OEL

Guo y Conitzer proponen en [Guo08] un tipo de mecanismos conocidos como OEL o "Optimal-in-Expectation Linear Mechanism". De nuevo, el funcionamiento y esquema del mecanismo es muy similar a los anteriores. En este caso el objetivo es hacer que el mecanismo reparta el máximo de los pagos en esperanza. Para ello se presupone conocida una distribución de probabilidad conjunta f. Aunque realmente esta función de distribución se utiliza a efectos formales, el mecanismo únicamente presupone conocidos la esperanza de pagos de cada jugador $\mathbb{E}[\theta_i]$. No es necesario que los jugadores sigan distribuciones independientes. Otro requisito que impone el

mecanismo para su formulación es presuponer que los valores declarados están acotados superior e inferiormente $\forall\theta_i,\ U \geq \theta_i \geq L$.

El mecanismo es complejo en su formulación (y en su demostración), pero abordable computacionalmente. Siguiendo su propuesta, definamos primero algunos valores relacionados al mecanismo:

$$o_0 = U - \mathbb{E}[\theta_1] \tag{3.3.4}$$

$$o_i = \mathbb{E}[\theta_i] - \mathbb{E}[\theta_{i+1}] \ \ \forall i \in [1, n-1] \tag{3.3.5}$$

$$o_n = \mathbb{E}[\theta_n] - L \tag{3.3.6}$$

$$\tag{3.3.7}$$

Definamos las funciones G y H según:

$$G(n,m,i) = \frac{\binom{n-i-1}{n-m-1}}{\binom{m-1}{i-1}} \quad \forall i \leq m \tag{3.3.8}$$

$$H(n,m,i) = \frac{\binom{i-1}{m-1}}{\binom{n-m-1}{n-i-1}} \quad \forall i > m \tag{3.3.9}$$

$$\tag{3.3.10}$$

Finalmente sea k un entero cualquiera según:

$$k \in \underset{i\in\{0,1,\ldots,n\}}{\textit{argmin}} \ \{o_i m \frac{\binom{n-1}{n}}{\binom{n}{i}} | i - m \text{ es impar}\}$$

Con estas definiciones en mente ahora podemos definir el mecanismo según los siguientes casos:

1. Si $0 < k \leq m$, entonces

$$c_i = \begin{cases} m/n - \sum_{j=k+1}^{m} (-1)^{m-j} G(n,m,j) & \text{si } i = k, \\ (-1)^{m-i} G(n,m,i) & \text{si } i = k+1, ..., m, \\ 0 & \text{para cualquier otro } i \end{cases}$$

2. Si $k = 0$, entonces

$$c_i = \begin{cases} Um/n - U\sum_{j=1}^{m} (-1)^{m-j} G(n,m,j) & \text{si } i = k = 0, \\ (-1)^{m-i} G(n,m,i) & \text{si } i = 1, 2, ..., m, \\ 0 & \text{para cualquier otro } i \end{cases}$$

3. Si $m+1 \leq k < n$, entonces

$$c_i = \begin{cases} m/n - \sum_{j=m+1}^{k-1}(-1)^{m-j-1}H(n,m,j) & \text{si } i = k, \\ (-1)^{m-i-1}H(n,m,i) & \text{si } i = m+1,...,n-1, \\ 0 & \text{para cualquier otro } i \end{cases}$$

4. Si $k = n$, entonces

$$c_i = \begin{cases} Lm/n - L\sum_{j=m+1}^{n-1}(-1)^{m-j-1}H(n,m,j) & \text{si } i = 0, \\ (-1)^{m-i-1}H(n,m,i) & \text{si } i = m+1,...,n-1, \\ 0 & \text{para cualquier otro } i \end{cases}$$

Este mecanismo consigue redistribuir, en esperanza, todos los pagos excepto la siguiente cantidad:

$$o_k m \frac{\binom{n-1}{m}}{\binom{n}{k}}$$

Este tipo de mecanismos basados en esperanza pueden ser muy interesantes y de aplicación en nuestro caso. Suponiendo conocida la distribución de los valores declarados por los nodos, y siendo ésta una distribución uniforme con soporte en el intervalo unidad, entonces $L = 0,\ U = 1, y\ \ \mathbb{E}[\theta_i] = \frac{n+1-i}{n+1}$.

3.4. Conclusiones del capítulo

A lo largo del presente capítulo hemos visto cómo es posible construir varios mecanismos que utilizan pagos y que podrían ser útiles en ciertos escenarios. Hemos visto como aplicar los mecanismos VCG a casos tan sencillos como un juego de dos jugadores y hemos avanzado en la complejidad hasta llegar a ver juegos con varios agentes y con la posibilidad de asignar la tarea a múltiples nodos.

Desgraciadamente, y por lo que hemos comentado hasta la fecha, consideramos que todos estos mecanismos sólo resuelven parcialmente el problema planteado en la tesis. El interés de incluir este material en la tesis ha sido múltiple. Por una parte, nos ha permitido introducir y aclarar muchos de los conceptos del modelo, así como observar que los mecanismos basados en VCG tienen propiedades muy interesantes.

Por otra parte, nos han de servir de base para el estudio futuro de la conexión que estos mecanismos tienen con los mecanismos de enlace.

Entre las ventajas habituales de estos mecanismos (eficiente, robusto a las mentiras, etc.), queremos hacer notar una característica relevante para su aplicación en sistemas distribuidos: no es necesario un papel central organizador. Obsérvese que todos los nodos pueden determinar si deben ser ellos o no los responsables de ejecutar la tarea. No es necesario disponer de un agente central que haga la selección. Los mecanismos son fácilmente computables de forma independiente por cada nodo. Únicamente en el caso de un empate sería necesario ampliar el mecanismo para que el desempate se pudiera hacer de forma distribuida. Más adelante veremos que una posibilidad es que los nodos generen un código hash con los valores declarados por los demás y optar por el menor. Esta característica hace muy atractivos estos mecanismos para los sistemas distribuidos sin una organización centralizada.

Como se verá, el hecho de que las decisiones se puedan computar independientemente por cada participante, hace que sea posible una visión muy interesante para nuestro trabajo futuro. En él, siempre pensaremos como un juego de dos jugadores: un jugador y el sistema. Es decir, un jugador cualquiera verá el juego como él contra un jugador ficticio que representa el sistema, formado por la agregación del resto de los jugadores.

Otra de las ventajas de estos mecanismos es que la ejecución del algoritmo suele ser computacionalmente aceptable. La complejidad de los mismos suele ser lineal. De hecho, considero que muchos de estos algoritmos pueden ser mejorados fácilmente si se disponen a los jugadores en agrupaciones jerárquicas, lo que reduciría el número de mensajes a transmitir.

En general, los mecanismos VCG sufren las consecuencias de algunos de los teoremas de imposibilidad que son ampliamente conocidos en la literatura. Desgraciadamente, y a mi parecer, una de las propiedades más negativas para su implantación en sistemas distribuidos, es que el sistema no dispone de pagos balanceados, por lo que sería necesario subvencionar a los jugadores en cada ronda. Existen variantes del mecanismo VCG que permiten balancear los pagos en esperanza (por ejemplo res-

tando del pago el valor esperado del mismo). Por lo tanto, si el juego fuera repetido se podría pensar en aplicar dichos mecanismos ya que garantizarían un pago balanceado en el tiempo, a la vez que se mantienen algunas de las propiedades buscadas.

Creemos que es difícil aceptar un sistema de pagos en sistemas distribuidos no controlados y anárquicos. Por ello, nuestra búsqueda seguirá explorando soluciones en las que los pagos no sean necesarios. Únicamente nos queda adentrarnos en la primera versión del mecanismo propuesto en la tesis. Este mecanismo busca obtener resultados similares a los obtenidos anteriormente, pero sin necesidad de utilizar la transferencia de utilidad. Es decir, buscamos obtener las mismas propiedades (sistema óptimo, resistente a las mentiras, etc.) pero que sustituya los pagos por un reparto de carga en el tiempo. Lo que se propone es que un jugador pueda "pagar" realizando un trabajo para los demás. Como se ha visto anteriormente, no todas las soluciones "razonables" son realmente correctas y es necesario un tratamiento formal del problema.

Capítulo 4

Algoritmo Quid Pro Quo básico.

"Work out your own salvation. Do not depend on others."

Buddha

4.1. Introducción

En los próximos capítulos presentaremos algoritmos que, a nuestro parecer, responden a los requisitos expresados anteriormente. La principal diferencia entre ellos es la variación en el grado de independencia entre los valores declarados por los jugadores.

Como veremos, todos los algoritmos propuestos utilizarán la idea de sustituir los pagos por un sistema de juego repetido en el que los jugadores puedan "devolver" los favores recibidos. Es por ello que se ha optado por denominarlos Algoritmos o *Mecanismos Quid Pro Quo* (o QPQ). El nombre del algoritmo proviene de una expresión latina muy utilizada en el ámbito legal y que se suele traducir por "*Esto por eso*","*Una cosa por otra*" o "*Hoy por ti, mañana por mí*". Esta expresión se suele utilizar cuando alguien realiza un trabajo o favor y espera en compensación un trabajo o favor equivalente. Hemos utilizado esta expresión ya que refleja el espíritu del algoritmo. Debido a la inexistencia de pagos, los nodos trabajan por los demás

con la esperanza de que los demás trabajen para él en un futuro.

La idea de QPQ se basa en una variación de un tipo de mecanismo que no utiliza la noción de pagos (transferencia de utilidad). Jackson y Sonnenschein [JS03, JS04, JS07] han propuesto un nuevo e interesante tipo de mecanismo denominado *Mecanismo de enlace* ("*Linking Mechanism*"). En lugar de ofrecer incentivos monetarios (o pagos) a los jugadores, estos mecanismos limitan el espectro de respuestas de los jugadores a una distribución de probabilidad conocida por el diseñador del juego.

En este capítulo abordaremos una primera versión de QPQ, en la que se ha supuesto que los jugadores tienen valoraciones independientes entre si. En los capítulos sucesivos se irá relajado esta restricción, generando en el camino otras variantes del algoritmo básico. El lector podría considerar que, en el fondo, unos son versiones ampliadas de los siguientes, y que por lo tanto se debería exponer únicamente la versión completa del mismo. Sin embargo, las consideraciones prácticas (y matemáticas) que se introducen en cada variación hacen que, a mi parecer, sea recomendable introducir paulatinamente dicha complejidad.

La estructura del presente capítulo es la siguiente. En la siguiente sección se exponen los fundamentos de los mecanismos de enlace de Jackson y Sonnenschein. A continuación, en la sección 4.3, pasaremos a preparar el camino hacia QPQ, comentando los cambios necesarios a la propuesta inicial de Jackson y Sonnenschein. Finalmente, se expondrá el Algoritmo QPQ, analizando sus propiedades en la sección 4.4 y las simulaciones realizadas en la sección 4.6.

4.2. Mecanismos de Jackson y Sonnenschein

Como se ha comentado anteriormente, Jackson y Sonnenschein han propuesto un tipo de mecanismo que se ha denominado "*Linking Mechanism*" y que en nuestro trabajo hemos traducido como *Mecanismo de enlace* o *Mecanismo que enlaza*.

La idea subyacente a este tipo de mecanismos se basa en que, cuando el juego consiste en varias copias del mismo problema básico, es posible restringir las respuestas del jugador de forma que el número de veces que el jugador asigna un valor

se corresponde con cierta probabilidad conocida de antemano. Como veremos, estos mecanismos tienen la propiedad de que se pueden eliminar los incentivos basados en la transferencia de utilidad.

Supongamos el siguiente ejemplo aclaratorio. Imagínese que en una determinada organización es necesario preparar un informe diariamente. Hay dos personas interesadas en que dicho informe se elabore y ambas están capacitadas para su realización. Cada persona y según el día, tiene una carga de trabajo diferente, lo que hace que el esfuerzo para realizar el informe siga una determinada distribución de probabilidad. Supongamos por ejemplo que un 60 % del tiempo, cada persona estará muy ocupada, mientras el 40 % restante estará más disponible. El mecanismo propuesto por Jackson y Sonnenschein lo que hace es obligar a que cada persona declare de media un 60 % de valores altos y un 40 % de valores bajos, controlando que los valores declarados se corresponden con la distribución de probabilidad ya conocida. Si una persona declara de forma continua costes altos, se descartan sus valores por considerarlos poco creíbles. En realidad, este tipo de mecanismos suelen ser lo que, de forma intuitiva, se aplica en nuestra vida cotidiana.

Otro artículo similar que explora la idea del mecanismo de enlace es el de Robert Ferenc ([Ves05]). En este artículo Ferenc propone un mecanismo que limita las respuestas del jugador usando para ello los dos primeros momentos (media y varianza) de la distribución de probabilidad, que se supone conocida por el diseñador del mecanismo.

En el artículo original, Jackson y Sonnenschein demuestran que este tipo de mecanismo funciona cuando el jugador tiene una probabilidad simple (no es continua). En ese mismo artículo, los autores comentan que se podría utilizar este mecanismo para juegos repetidos. La demostración para el juego repetido es efectivamente evidente y se trasluce del articulo, pero dicho trabajo no ofrece un método para la construcción de mecanismos cuando el juego es repetido y con distribuciones de probabilidad continuas (y desconocidas por el diseñador del mecanismo).

A continuación, repasaremos el principal teorema de Jackson y Sonnenschein sobre el que se basa nuestro trabajo. En esta sección se recordará la formulación

original del mismo y verán las principales conclusiones que resultan de su aplicación.

En su trabajo, Jackson y Sonnenschein demuestran que:

Teorema 4.2.1. *Sea un problema de decisión D y una función de decisión social f definida sobre D y que es ex-ante Pareto eficiente. Existe una secuencia de mecanismos de enlace (M^K, g^K) sobre versiones enlazadas del problema de decisión tal que:*

1. *Existe una secuencia de equilibrios bayesianos que son aproximadamente ciertos.*

2. *La secuencia de mecanismos de enlace junto con esa secuencia de equilibrios, se aproxima a f.*

3. *Cualquier secuencia de estrategias aproximadamente ciertas de un agente i, le asegura un nivel de utilidad que converge al nivel objetivo ex-ante $\mathbb{E}[u_i(f(u))]$ ("ex-ante target level").*

4. *Todas las secuencias de equilibrios Bayesianos del mecanismo de enlace producen utilidades esperadas que convergen hacia un perfil ex-ante eficiente.*

5. *Para cualquier secuencia de equilibrios bayesianos y para cualquier secuencia de desviaciones producidas por coaliciones, la máxima ganancia que obtiene el agente en la coalición se desvanece según avanza la secuencia.*

En resumen, lo que nos proponen Jackson y Sonnenschein en su trabajo es lo siguiente. Supongamos que el diseñador del mecanismo se encuentra con un problema formado por una colección de decisiones. El diseñador de mecanismos conoce que la distribución de valores que define el tipo de cada jugador. A continuación, se pide a cada jugador que emita una declaración sobre su tipo (supuestamente es información

privada). Se observa si las valoraciones de un jugador concreto respeta la distribución del jugador. Ante la sospecha de que los valores no se correspondan con una distribución correcta, el sistema elimina los valores erróneos y propone otros en su lugar. Cuando el número de valores y decisiones crece, la función de decisión se aproxima a la eficiencia y los valores declarados son aproximadamente ciertos.

Dado que estos mecanismos requieren de un número de instancias del juego básico (declaraciones y decisiones), el teorema necesariamente habla de secuencias que evolucionan conforme crece dicho número. Lo importante es que si el juego es suficientemente repetido, el teorema nos confirma que podríamos obtener declaraciones (aproximadamente) ciertas por parte de los jugadores, y por tanto, el sistema sería (aproximadamente) eficiente.

Aunque la idea central de QPQ se basa en este tipo de mecanismos, no incluimos la demostración original del teorema de Jackson y Sonnenschein. La demostración del teorema en su versión original se realiza suponiendo que la distribución de probabilidad sobre las utilidades es simple. Dado que en nuestro algoritmo se suponen condiciones diferentes, se remite al lector al trabajo de Jackson y Sonnenschein en la que se justifica su aplicación. En cualquier caso, como veremos para el caso de QPQ, los cambios introducidos en nuestra solución hacen que sea posible realizar una demostración más adaptada a nuestras necesidades.

Aunque la solución propuesta es una variante del mecanismo de Jackson y Sonnenschein, nos enfrentamos a dos problemas que lo hacen diferente. Por una parte no deseamos suponer un conocimiento previo de la distribución de probabilidad de los jugadores, y por otra deseamos implementar una noción de justicia en el reparto de tareas.

Según nuestro mejor conocimiento, **éste es el primer trabajo que propone una aplicación del mecanismo de enlace con las propiedades mencionadas (justicia, etc.)**. En la siguiente sección veremos como es posible alterar el mecanismo de enlace para poder resolver estos requisitos.

4.3. Aplicación de los mecanismos de enlace

Veamos a continuación como QPQ aborda estas características. Primero propondremos una versión de QPQ aplicando directamente el mecanismo de enlace. Veremos que esta primera versión aparecen aspectos que serán determinantes a la hora de definir correctamente nuestra solución. Nuestra primera versión del algoritmo sería la expuesta en el algoritmo 5.

Algorithm 5 Simple linking mechanism (code for node i)

1: Estimate and publish the cost c_i of the task
2: Wait to receive the costs c_j from the other players
3: **for all** $j \in N$ **do**
4: **if** not $Accepted(c_j, Historic_j)$ **then**
5: $c_j \leftarrow Random(c_{-j})$
6: **end if**
7: $Historic_j \leftarrow Historic_j \cup \{c_j\}$
8: **end for**
9: $d \leftarrow \underset{j \in N}{argmin}\ c_j$
10: **if** $d = i$ **then**
11: execute the task
12: **else**
13: do nothing (node d will execute the task)
14: **end if**

Este algoritmo reproduce el funcionamiento de los mecanismos de enlace y nos ofrece una intuición sobre el funcionamiento de tales mecanismos. Obsérvese, que el algoritmo es el mismo para todos los participantes del juego y que por lo tanto no es necesario un agente o nodo central. El algoritmo se aplica para cada ronda o tarea

que surja en el tiempo. Es decir, este algoritmo lo aplican todos los nodos para cada ronda o tarea. Durante todas las rondas (tareas sucesivas) se pide a los jugadores que declaren su tipo (en nuestro caso, su coste c_i). Cada nodo espera a tener todas las declaraciones de todos los demás participantes. En ese momento podrá determinar si el valor declarado por un contrario c_j es aceptable o no, y para ello comprueba si la muestra puede proceder de la distribución de probabilidad de ese jugador. En nuestro algoritmo, ese proceso lo hemos representado por la función $Accepted$. Si el valor no procede de esa distribución, se generar un valor alternativo. Finalmente, el jugador determina a quien le corresponde ejecutar la tarea (se determina por la función $argmin$).

Aunque puede ser útil para llegar a comprender el funcionamiento del mecanismo de enlace, esta solución tiene unos aspectos que lo hacen desaconsajable para su implantación según nuestros requisitos. Claramente contradice muchos de nuestros puntos de partida. En concreto, nuestra primera version no garantiza una asignación de trabajo justa (medida como reparto equitativo de tareas o realización de un esfuerzo equivalente). Además, dado que se ha supuesto que los jugadores podrían tener una distribución de probabilidad arbitraría, sería complejo determinar la función de aceptación de las muestras declaradas por los jugadores. Veamos estos aspectos en detalle y la solución propuesta para abordarlos.

4.3.1. Transformación de la función de utilidad

Desafortunadamente y aunque la existencia de la función de utilidad se garantiza por el teorema 2.1.1, ésta no es única. Esto implica que no puedan hacerse comparaciones interpersonales de los niveles de utilidad. Veamos a través de un ejemplo que significa la afirmación anterior. En las gráficas 4.1 y 4.2 se pueden observar los costes declarados por dos hipotéticos jugadores. En ellas se observan cómo el primer jugador tiene una función de utilidad que sigue una distribución de probabilidad completamente diferente a la del segundo jugador. De hecho, aunque el soporte sea similar, la información suministrada por ambos jugadores es muy diferente.

Esto puede suceder por muchas razones. Puede ser causado por el hecho de que cada jugador mide su utilidad según un criterio diferente. Por ejemplo, supongamos que el primer jugador mide su coste según un criterio basado en el consumo de memoria. El segundo puede interpretar el coste según el consumo de ancho de banda. Y aunque ambos utilicen el mismo criterio, sea éste el que sea, el problema puede surgir simplemente por encontrarse en un contexto diferente (por ejemplo, un nodo es un ordenador personal y el otro es un ordenador de una empresa).

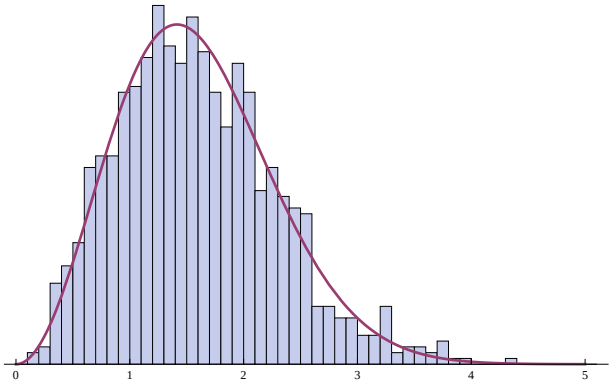

Figura 4.1: Ejemplo 1 de distribución de coste.

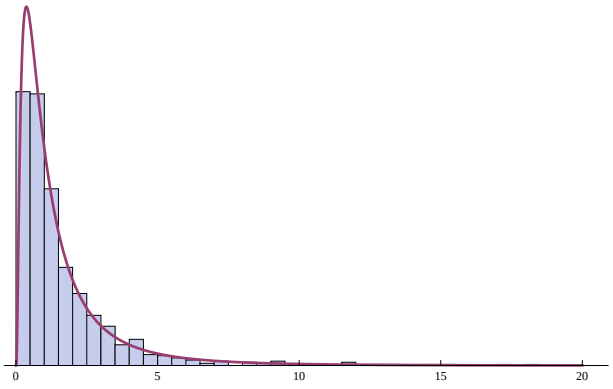

Figura 4.2: Ejemplo 2 de distribución de coste.

Además, existe otro problema derivado de la propia función de utilidad. Por la teoría de utilidad esperada (2.1.1) se sabe que cualquier transformación lineal del

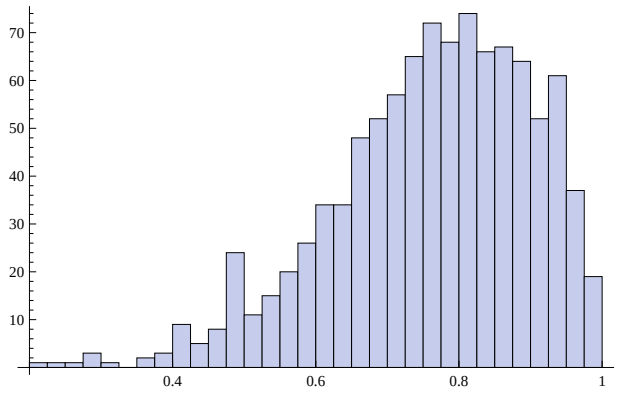

Figura 4.3: Ejemplo de distribución.

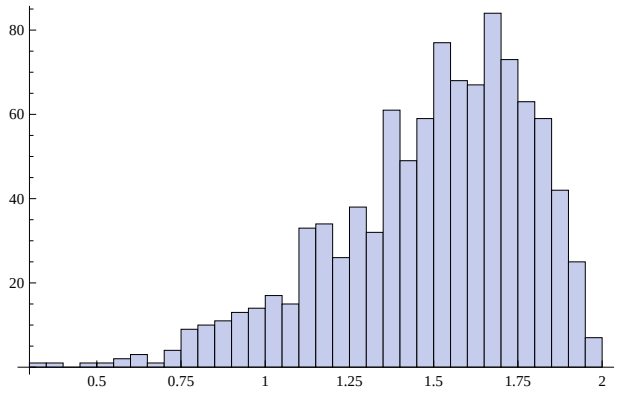

Figura 4.4: Ejemplo de distribución escalada.

tipo:

$$u_i^n(\cdot) = \alpha u_i(\cdot) + \beta$$

genera una función de utilidad equivalente. Esto justificaría que en algunos casos los jugadores tengan medias muy diferentes, provocando que un jugador terminara realizando más tareas que los demás. Por ejemplo, los jugadores de las gráficas 4.3 y 4.4 tienen funciones de utilidad similares, pero tienen escalas diferentes.

Al plantear el juego entre los jugadores de las gráficas anteriores, debemos tener en cuenta que la situación podría resultar muy injusta si no disponemos de los incentivos (pagos) correspondientes, dado que uno de los jugadores siempre haría más

trabajo, y no se vería recompensado por la falta de pagos.

Ninguno de estos problemas se plantean en el trabajo original de Jackson y Sonnenschein, posiblemente debido a que se sitúa en un contexto económico muy concreto. Pero estos problemas son claves si deseamos encontrar una solución en el ámbito de redes distribuidas. En un primer momento, se puede pensar que una solución sería obligar a que todos los participantes midieran sus costes en una misma unidad monetaria. Y aunque pudiera ser factible realizar tal suposición, creemos que es mucho más serio pensar que todos los participantes tienen una distribución conocida de antemano, y similar a los demás. Dos nodos podrían medir sus costes en una misma unidad, pero lo más probable es que sigan distribuciones desconocidas, incluso por ellos mismos.

La idea que propone el algoritmo QPQ es **normalizar los valores, de tal forma que sea factible comparar** los costes de ambos jugadores. Lo ideal es realizar un planteamiento centrado únicamente en el jugador. Estamos suponiendo que el jugador es racional y que puede responder a preguntas del estilo: ¿Qué parte de todas tus tareas te gustaría realizar si te correspondiera un determinado porcentaje de trabajo?.

En esta pregunta estamos suponiendo que el jugador puede establecer preferencias entre dos subconjuntos de una muestra de las tareas. La muestra de las tareas es el conjunto de tareas que aparece en el tiempo. Si el jugador pudiera conocer ese conjunto (y de hecho podría tener un conocimiento objetivo o subjetivo de las probabilidades que definen ese conjunto), éste podría determinar qué parte de las misma le gustaría realizar si le fijamos un porcentaje. En este sentido, decimos que el jugador es racional.

Por lo tanto, la pregunta "¿Cuánto te cuesta hacer la tarea?" debemos transformarla en otra parecida a "¿Qué probabilidad asignarías a que esta nueva tarea pertenezca al conjunto de tareas que te gustaría realizar?" o quizás "¿Qué porcentaje de tareas de las que desearías hacer estarían por debajo de esta tarea?".

Realmente, todas estas preguntas son muy similares, pero las últimas están normalizadas en el intervalo $[0, 1]$ lo que supone una gran ventaja. Además, como se verá en breve, esa respuesta debe seguir una distribución uniforme ya que realmente

estamos aplicando una transformada con nombre propio.

En la estadística existe una transformada conocida por la *transformada de la integral de probabilidad* (en adelante, nos referiremos a ella como **PIT** de las siglas en inglés de "*Probability Integral Transformation*").

Dado que la transformada PIT utiliza el concepto de función de distribución acumulada de probabilidad, recordemos brevemente la definición:

Definición 4.3.1 (Cumulative Distribution Function)**.** Sea X una variable aleatoria real definida sobre un espacio de probabilidad (Ω, Σ, Pr), se define la función de distribución acumulada de probabilidad (o CDF según sus siglas en ingles, "cumulative distribution function") como $F_X(x) = Pr\{\omega : X(\omega) \leq x\}, x \in (-\infty, \infty)$.

Formalmente, la transformada de probabilidad se define como:

Definición 4.3.2 (Probability Integral Transformation)**.** Dada una variable aleatoria continua X con una distribución acumulada de probabilidad (o CDF según sus siglas en ingles, "cumulative distribution function") igual a F; que usualmente se denota por $X \sim F$. Entonces, la transformada integral de probabilidad define una nueva variable aleatoria Y definida según: $Y = F(X)$.

Esta transformada tiene como propiedad que cualquier función de distribución de probabilidad acumulada sigue una distribución uniforme. Este hecho es ampliamente conocido en estadística. Recordemos algunos conceptos básicos que utilizaremos en nuestro trabajo.

Definición 4.3.3 (Distribución Uniforme)**.** Una variable aleatoria continua Y se dice que es una variable aleatoria uniforme en el intervalo $[\alpha, \beta]$ si, y sólo si, su función de densidad de probabilidad sigue:

$$f(y) = \begin{cases} \frac{1}{\beta - \alpha} & \text{si } \beta \leq y \leq \alpha, \\ 0 & \text{en caso contrario.} \end{cases}$$

La función de distribución acumulada (CDF) de una variable aleatoria uniforme es

$$F(y) = \begin{cases} 0 & \text{si } y \leq \alpha, \\ \frac{y-\alpha}{\beta-\alpha} & \text{si } \beta \leq y \leq \alpha, \\ 1 & \text{si } y \geq \beta. \end{cases}$$

La importancia de utilizar la transformación integral en nuestro trabajo se basa en la propiedad anteriormente mencionada, y que refleja el siguiente lema:

Lema 4.3.1 (Transformación integral de probabilidad)**.** *Dada una variable aleatoria continua X con soporte en $[\alpha, \beta] \subseteq \mathbb{R}$, su función de distribución acumulada $F(x)$ sigue una distribución uniforme.*

Todas las definiciones anteriores suponen que la variable aleatoria X es continua, pero en el caso de que los valores del jugador siga una distribución discreta, siempre es posible realizar una transformada similar. Para ello, se deben elegir valores aleatorios uniformes dentro de los "escalones" discretos que se forman en la función de distribución acumulada. Esta variante de la transformada se conoce como la transformada distribucional generalizada ("generalized distributional transform").:

Definición 4.3.4 (Generalized Distributional Transform)**.** Sea X una variable aleatoria (no necesariamente continua) con una distribución acumulada de probabilidad igual a F y sea $V \sim U(0, 1)$ con distribución uniforme en $(0, 1)$ e independiente de X. La *función de distribución modificada* $F(x, \lambda)$ se define conforme a:

$$F(x, \lambda) = P(X < x) + \lambda P(X = x).$$

Definimos la transformada distribucional generalizada de X según:

$$Y = F(X, V)$$

Al igual que la transformada PIT, esta transformada generalizada sigue una distribución uniforme en el intervalo unidad. Una demostración de esta propiedad se

puede encontrar en Fergunson [Fer67]. Son múltiples los trabajos que en economía que utilizan esta definición y sus propiedades; por ejemplo [BR06] o [LR09].

El algoritmo QPQ utiliza esta transformación para los valores declarados por los nodos. En lugar de publicar los valores provenientes de su distribución de probabilidad, deben publicar los valores normalizados mediante esta transformación. Las gráficas 4.5 y 4.6 ilustra este proceso. Estas gráficas representan la acumulada (CDF) de dos jugadores, cada uno con una distribución diferente. El jugador 1 ha declarado un coste de $0{,}8$ mientras que el jugador 2 ha manifestado que su coste sería $2{,}0$. Sin la transformada, la tarea debería ser adjudicada al jugador de menor coste (esto es al jugador 1), pero a la larga, se produciría una injusticia con este jugador. En cambio, al aplicar la transformada PIT a los valores declarados, obtenemos unos nuevos costes normalizados, que están representados por las flechas. En este caso, la tarea es realmente adjudicada al jugador 2. Como veremos en las propiedades de QPQ, esta solución produce justicia y equilibrio entre los jugadores.

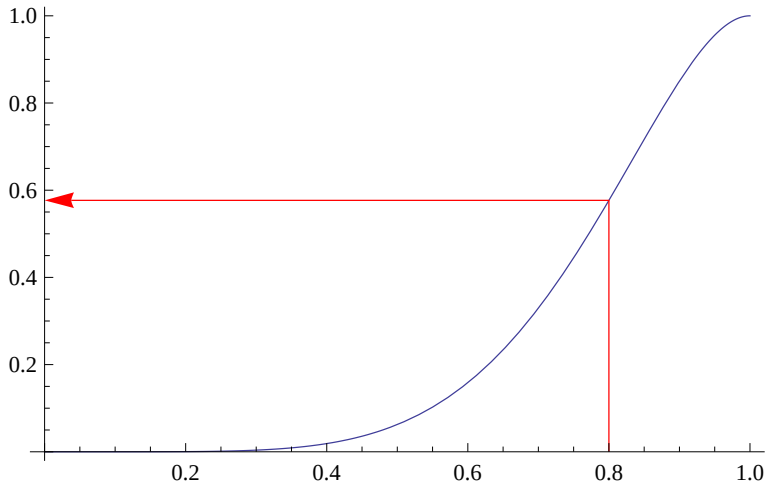

Figura 4.5: **Ejemplo transformada PIT (I)**. El jugador declara un valor de $0{,}8$ pero el sistema trabajrá con un valor normalizado cercano a $0{,}6$.

Obsérvese que es posible aplicar la transformación PIT incluso cuando los jugadores no conocen a priori su distribución de probabilidad. Es decir, se podría construir

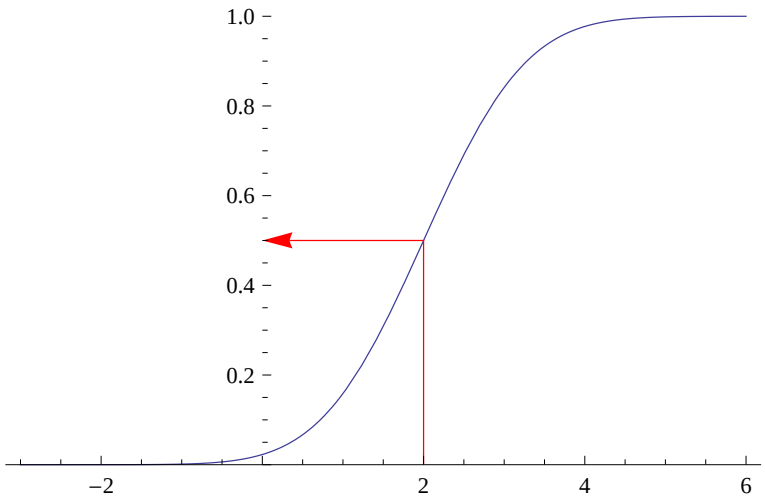

Figura 4.6: **Ejemplo transformada PIT (II)**. El jugador declara un valor de 2 pero el sistema trabajrá con un valor normalizado cercano a $0{,}5$.

utilizando para ello las sucesivas muestras que dicha distribución genere al jugar. Este proceso se denomina en estadística la función empírica de distribución acumulada de probabilidad (en adelante ECDF de sus siglas en inglés "Empirical Cumulative Distribution Function"). Formalmente este concepto se define como:

Definición 4.3.5 (Empirical Cumulative Distribution Function)**.** La función de distribución acumulada empírica (o ecdf de sus siglas en inglés "empirical cumulative distribution function') F_n definida para n muestras y_i se define como:

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} \begin{cases} 1 & \text{si } y_i \leq x, \\ 0 & \text{en caso contrario.} \end{cases}$$

Evidentemente este proceso tiene un error que tiende a cero si el número de muestras crece (según aumente las rondas en las que participa el jugador). Este hecho se conoce por el teorema de Glivenko-Cantelli, que nos dice que:

$$\sup_{n\in\mathbb{R}} |F_n(x) - F(x)| \to 0 \text{ casi seguro.}$$

Volveremos a este concepto cuando tratemos la implantación práctica de QPQ. Por el momento, y con el objetivo de realizar un análisis de las propiedades de QPQ, supondremos que conocemos de forma exacta la función $F(x)$.

Llegados a este punto, creo que merece la pena repasar con el lector lo conseguido gracias a la transformada PIT. Recordemos que uno de los problemas que teníamos para aplicar los mecanismos de enlace era que el diseñador de mecanismos debía conocer de antemano la distribución de probabilidad. Ahora, no existe tal problema. Todos los jugadores seguirán una *distribución uniforme*, independientemente de su distribución inicial de costes. Además, se ha conseguido otro aspecto interesante: es posible tratar a todos los jugadores por igual. Esto es, tenemos las herramientas para buscar justicia en el reparto. Gracias a la transformada PIT, todos los jugadores son iguales, van a declarar valores (normalizados) en el mismo rango $[0, 1]$ y todos siguen la misma distribución uniforme. Pero lo más importante y fundamental es que ahora tenemos la herramienta para aplicar el mecanismo de enlace según nuestros requisitos. Ahora existe una "moneda única" que puede ser utilizada para intercambiar trabajo presente por futuro. Es decir, las unidades de la transformada PIT nos servirá como "moneda" para medir cuanto trabajo debe un nodo a sus similares. Pero además, como veremos en las propiedades de QPQ, la transformada PIT posibilita que los nodos sin conocimientos previos de su propia capacidad (o limitados racionalmente) puedan participar en el mecanismo mejorando su utilidad. La transformada PIT es una herramienta fundamental de QPQ.

4.3.2. Aceptación de valores

Hasta el momento, hemos supuesto que los jugadores son honestos (racionales o no) y han decidido calcular correctamente su transformada PIT. Realmente, nuestro algoritmo debe aportar una solución a cualquier manipulación a tal respecto. Quizás el lector piense que la solución a tal problema pasa por hacer que sea el sistema (los otros nodo) quienes realicen tal computo. En realidad, eso no serviría de mucho ya que entonces el jugador podría alterar los valores declarados de tal forma que al hacer

el cálculo de la transformada, resulten los valores que le interese al jugador.

En los trabajos publicados sobre mecanismos de enlace se supone que las instancias del juego (rondas) son simultaneas en el tiempo. En este caso es posible, y relativamente fácil, restringir las declaraciones de los jugadores. Pero si las tareas aparecen en el tiempo, siempre sería posible tal alteración. Por ejemplo, en el caso de que un jugador intente ahorrarse la ejecución de las tareas, una estrategia posible sería decir valores de coste cada vez mayores, de forma que la transformada PIT siempre genere valores cercanos a la unidad. Este tipo de comportamiento debe ser rechazado, aunque surgen varias dudas.

Por un lado, está el problema de qué se debe hacer para penalizar al jugador mentiroso y, por otro, existe la cuestión de cómo detectar tal comportamiento. Esta sección tratará sobre la solución a estos dos problemas: buscar el castigo adecuado y encontrar las herramientas de detección.

Actuación ante la mentira

Veamos inicialmente el problema sobre el "castigo" a imponer a nuestro jugador deshonesto. Podría pensarse en obligar al nodo a ejecutar la tarea cuando se detecta una mentira. Desgraciadamente, esta política podría obligar al jugador a realizar tareas en casos de falsos positivos y por otra parte, se pierde el control sobre el comportamiento del sistema.

Por lo tanto, y siguiendo los mecanismos de enlace, lo que se propone es rechazar el valor declarado por el jugador y generar un valor aleatorio que sustituya el declarado según la distribución uniforme en $[0, 1]$. Cierto es que parece un "castigo" muy pobre ya que siempre habrá una probabilidad de que el jugador salga bien librado de su mentira. Y aparentemente, parece que sería recomendable una acción punitiva más rotunda. Pero curiosamente, este "castigo" tan aparentemente leve, es suficiente para desincentivar a los jugadores deshonestos y evitar así las mentiras.

Como veremos, lo ideal es que, incluso frente a una mentira, el sistema siga siendo un juego de distribuciones uniformes. Este último punto es crucial. Dado que deseamos construir un sistema strategy-proof, debemos encontrar una forma de que

los jugadores no tengan en cuenta la estrategia de los demás para calcular su mejor línea de acción. Es decir, buscamos una solución que tenga estrategias dominantes.

Un punto a tener en cuenta es la forma de generar el valor aleatorio cuando el sistema detecte una mentira. Evidentemente no se puede esperar a que sea el propio nodo el que debe generar dicho valor, ya que esto estaría sujeto a manipulaciones. Además debe ser un mecanismo determinista para que los nodos restantes puedan calcular y comprobar el nuevo valor. En la sección 4.5 se tratará la implementación práctica de nuestro algoritmo y pasaremos a proponer algunas alternativas a la generación de dicho valor.

Herramientas de control

Con respecto a nuestro problema de control, éste se limita a encontrar la herramienta que detecta desviaciones de una distribución dada. Uno de los artículos comentados sugiere limitar las declaraciones a los valores que fijan la media y la varianza. Para nuestro problema eso no es posible ya que el jugador vería limitado el margen de maniobra para responder. Por ejemplo, si le decimos que nos diga un primer valor que genere una media igual a $0{,}5$, el único valor que puede decir es $0{,}5$. Algo parecido le ocurrirá en los valores sucesivos. Podríamos aceptar un cierto margen de error de tal forma que el jugador tenga un rango más amplio de valores. Por supuesto, ese error tendría que tender a cero según se incremente el número de rondas.

En el fondo, una solución de ese estilo lo que hace es comparar los valores que está diciendo el jugador con una cierta distribución de probabilidad. En nuestro caso, y después de ver el proceso de transformación PIT, la distribución que nos interesa comprobar es la uniforme. El problema se reduce pues a encontrar un sistema de comprobación que nos permita asegurarnos que el jugador está realizando sus declaración según la distribución uniforme.

En estadística existen un amplio abanico de tests que nos permiten comprobar si un conjunto de muestras se corresponden a una distribución dada. Estos tests se denominan bondad-de-ajuste Gof (de las siglas en inglés “goodness-of-fit”).

Todo test de ajuste o de bondad mide la probabilidad de que una serie muestral proceda de una determinada distribución. Se dice que un test acepta un conjunto de muestras cuando las muestras superan un determinado umbral de probabilidad. La fuerza de un test se suele medir contrastando la hipótesis de que las muestras proceden de la distribución objetivo (llamada distribución nula) frente a la hipótesis de que proceden de otra distribución contraste.

En principio, QPQ podría utilizar cualquier test Gof presente o futuro que permita detectar desviaciones significativas de la distribución uniforme. Volveremos a este concepto cuano tratemos la implantación práctica de QPQ. Por el momento, aceptemos la hipotesis de que existe un test de aceptación, idealmente perfecto.

Visto todo lo anterior, ahora estamos en situación de introducir una versión teórica de nuestra solución y que será presentada en la siguiente sección.

4.4. Algoritmo Quid Pro Quo (QPQ) Básico

La primera versión del *Mecanismo Quid Pro Quo* (QPQ) que proponemos en nuestro trabajo es el descrito en 6.

Antes de pasar a analizar las propiedades de este algoritmo, nos gustaría resaltar el funcionamiento del mismo. Cuando la tarea se da a conocer, el nodo puede estimar su coste y publicar el valor normalizado según PIT. El valor del nodo se transmite hacia todos los demás jugadores. Cuando el jugador tiene todos los valores de los demás jugadores, puede proceder a determinar si la tarea le corresponde a él o no. Para ello, comprueba si alguno de los jugadores tiene un valor sospecho, aplicando el test de bondad y cambiando éste si es falso según lo descrito anteriormente. Con los valores ya revisados, el jugador procede a determinar si su valor es el mínimo, en cuyo caso ejecuta la tarea, publicando el resultado de la misma si procediera.

Obsérvese que si un jugador no sigue el protocolo, el sistema generará valores de forma aleatoria y por tanto el jugador termina reduciendo su beneficio ya que le tocará realizar tareas que no desea. En los apartados siguientes se estudiará el daño esperado (o la reducción del beneficio) que este comportamiento puede llegar a

Algorithm 6 Quid Pro Quo mechanism (code for node i, and a generic task t)

Estimate the cost c_i of the task
Publish the normalized cost $\bar{c}_i = PIT(c_i)$
Wait to receive the normalized costs $\bar{c}_j$ from the other players
for all $j \in N$ **do**
    if not $GoF_Test(\bar{c}_j, Historic_j, p\text{-}th_j)$ **then**
        $\bar{c}_j \leftarrow Random(\bar{c}_{-j})$
    end if
    $Historic_j \leftarrow Historic_j \cup \{\bar{c}_j\}$
end for
Let $d = \underset{j \in N}{argmin}\ \bar{c}_j$
if $d = i$ **then**
    execute the task
else
    do nothing (node d will execute the task)
end if

provocar. En cualquier caso, esta pérdida de beneficio será comparable a la que tenga un jugador que le sea indiferente la tarea a realizar. Con este límite garantizamos que todos los nodos tienen un beneficio esperado nunca inferior a un juego en el que la asignación de las tareas se realice de forma aleatoria. Esta propiedad es muy útil en el caso de que el nodo no sea capaz de evaluar sus costes (es no-racional).

Otra conclusión importante del sistema propuesto es que garantiza un cierto comportamiento y beneficio a todo el sistema, incluso si uno o varios jugadores son no-racionales o son racionalmente limitados. Es fácil comprobar que todos los jugadores actúan y juegan como si la totalidad de los jugadores son racionales. Es decir, un comportamiento erróneo de un grupo de jugadores no afecta a la estrategia de los

jugadores correctos.

Todos estos aspectos se analizarán y demostrarán en la siguiente sección.

4.4.1. Aspectos formales del algoritmo

Aunque, con las modificaciones sugeridas, la demostración de los mecanismos de enlace sería aplicable al problema tratado en la presente tesis, creemos interesante repasar los puntos más importantes de la demostración de dicho teorema, haciendo los cambios apropiados para el caso del algoritmo QPQ. Esta sección estará dedicada a la demostración matemática de las propiedades de QPQ en su versión básica. Recordemos que la principal diferencia entre las versiones de QPQ viene dada por el grado de independencia entre jugadores. En esta primera versión, hemos supuesto que los valores de coste de los jugadores siguen distribuciones de probabilidad independientes entre ellos (incluso cuando los jugadores manipulan sus valores). Este requisito se relajará en los capítulos sucesivos.

Entremos por tanto, en analizar las propiedades de QPQ Básico. Suponiendo que el número de rondas (tareas) es suficientemente alto y que los jugadores son independientes entre ellos, en esta sección demostraremos las siguiente propiedades de nuestro algoritmo:

1. QPQ es óptimo en el sentido de que minimiza el trabajo normalizado total cuando los jugadores son honestos.

2. Para cualquier jugador, el resto de los participantes se puede ver como un jugador único que representa al sistema. Ese jugador publica como valor el menor valor de todos sus miembros. Ese valor sigue una distribución Beta de probabilidad.

3. Un jugador determina su mejor estrategia con independencia de lo que hagan el resto de los jugadores.

4. La estrategía que optimiza la utilidad normalizada de un jugador es la correspondiente a declarar su valor real (ser honesto). Utilizando la terminología del

diseño de mecanimos, diremos que QPQ es strategy-proof (dentro del espacio de estrategias definido).

5. Todos los jugadores *siempre* obtienen una utilidad positiva (en esperanza), que únicamente depende del número de jugadores.

6. Los jugadores irracionales o racionalmente limitados también obtendrán un beneficio positivo por su participación.

7. El sistema es justo tanto si lo medimos en el número de tareas asignadas como si lo medimos en el esfuerzo normalizado. Esto es, todos los jugadores realizarán un número de tareas similar y, en caso de ser honestos, realizarán un esfuerzo normalizado similar. Nótese, que esta propiedad no contradice la anterior.

8. Cuando el número de jugadores es relativamente alto, QPQ obtiene un rendimiento muy atractivo.

Para el análisis se ha supuesto que tanto la estimación de la función de distribución, la transformada PIT como el funcionamiento del test de aceptación son perfectos. De hecho, suponiendo un número suficientemente alto de muestras, estos procesos tienen errores cercanos a cero. Otro aspecto que simplificará nuestro análisis es la idea de *jugador agregado*. Evaluaremos el rendimiento de un nodo jugando contra un jugador "imaginario" que agrega las respuestas de todos los demás agentes. Este jugador publica siempre el menor valor (normalizado) de todos los jugadores a los que agrupa. Este enfoque es compatible con todas las suposiciones de nuestro modelo y nos será de ayuda dado que simplifica el análisis.

Para aclarar nuestra notación, dada una tarea, usaremos x para denotar el coste normalizado real $\bar{c}_i$ del jugador i para esa tarea, mientras que X o X_i se referirán a las variables aleatorias de dicho valor. Cuando los jugadores participen en QPQ podrán optar por publicar sus valores reales (normalizados) $x = \bar{c}_i$ o mentir y declarar valores falso. En ese caso, utilizaremos z para referirnos a ese valor falso $\hat{c}_i$ y, abusando de la notación, al valor que el sistema regenera cuando el test Gof ha detectado una mentira. Suponemos que estos valores z son realizaciones de alguna variable aleatoria

Z. Dada una tarea con coste c_i, el jugador obtiene una utilidad normalizada igual a $\bar{u}_i = \bar{c}_i$ si no ejecuta la tarea (idependientemente del valor declarado) y realiza un esfuerzo normalizado igual a $\bar{w}_i = \bar{c}_i$ si debe ejecutar la tarea (donde $\bar{W}_i$ se refiere a la variable aleatoria asociada). Además, utilizaremos $y = \theta_{-i}$ para referirnos a menor valor publicado por el jugador agregado $y = \textit{min}\ \{\bar{c}_{-i}\}$. Utilizando la notación del diseño de mecanismos, decimos que la función de decisión social d de QPQ es:

$$d = \underset{i \in N}{\textit{argmin}}\ \bar{c}_i.$$

Definimos $Pr[d = i]$ como la probabilidad de que el jugador i declare el menor valor (y que se le asigne la tarea). Cuando nos referimos al jugador agregado, Y es un vector de variables aleatorias y utilizaremos $Pr[Y \leq y]$ para referirnos a la probabilidad de que al menos uno de los elementos de Y, digamos j, asegure $Y_j \leq y$.

Con esta notación podemos demostrar que para todo jugador i, la esperanza de los costes declarados es igual a la utilidad esperada más el coste esperado. Esta suma es una constante,

$$E[\bar{C}_i] = E[\bar{U}_i + \bar{W}_i] = E[\bar{U}_i] + E[\bar{W}_i] = \int_0^1 \bar{c}_i d\bar{c}_i = \frac{1}{2}. \tag{4.4.1}$$

Este hecho nos garantiza que cuando el jugador intenta minimizar su coste está maximizando su utilidad, para cualquier entorno de juego (función de decisión, número de jugadores, distribución de probabilidad, etc.). A lo largo de nuestro trabajo, utilizaremos este hecho.

Distribución de costes normalizados Nuestro primer paso será observar que el sistema siempre trabaja con una única distribución de probabilidad para cada agente, independientemente de lo que suceda en el sistema. De hecho, el sistema siempre trabajará con una distribución uniforme en $[0, 1]$,

Cuando los jugadores son honestos, y por las propiedades de la transformada PIT, sus valores deben seguir la distribución uniforme en $[0, 1]$. Por otra parte, si el jugador es deshonesto, éste puede decidir cambiar la distribución de sus valores, pero en ese caso hemos supuesto que el test GoF detectará ese comportamiento y por

tanto se reemplazarán los valores declarados por otros seudoaleatorios que han sido obtenidos de una distribución independiente y uniforme en $[0, 1]$. Un último caso sería cuando el jugador declara valores normalizados que siguen una distribución uniforme e independiente de los demás jugadores y por tanto pasará el test GoF. En este caso, el valor $\bar{c}_i(t)$ para una tarea t es independiente de los valores $\bar{c}_j(t)$ declarados por los otros jugadores. Por lo tanto.

Proposición 4.4.1. *El conjunto de valores de coste normalizados considerados en la línea* 10 *del Algoritmo 6 siguen una distribución independiente e identica (iid) y según la distribución uniforme en* $[0, 1]$.

Optimalidad El siguiente paso será observar que QPQ siempre asigna las tareas de forma óptima: cuando los jugadores son honestos, QPQ minimiza el coste normalizado total.

Proposición 4.4.2. *Supongamos que los jugadores son honestos. Para un conjunto de tareas* T*, no existe nigún mecanismo* M *que cumpla* $E[\sum_{i=1}^{n} \bar{W}_i^M] < E[\sum_{i=1}^{n} \bar{W}_i]$*, donde* $\bar{W}_i^M$ *es la variable aleatoria asociada al trabajo normalizado realizado por el jugador* i *cuando participa del mecanismo* M.

Demostración. La demostración es inmediata usando contradicción. Supongamos que tal mecanismo M existe, y por lo tanto debe existir, al menos una tarea para la que se cumpla $\bar{w}^M < \bar{w}$. Pero la función de decisión social de QPQ siempre selecciona al jugador con el mínimo valor declarado, por lo tanto no es posible que M pueda seleccionar a otro jugador capaz de ejecutar la tarea con un coste menor. Por lo tanto, podemos concluir que no puede existir dicho mecanismo M. □

Jugador agregado Dado que los costes normalizados de un jugador proceden de una distribución uniforme e independiente en $[0, 1]$, podemos asegurar que la función

de densidad de cada jugador i es $f_i(\bar{c}_i) = 1$, definida en dicho intervalo. Por lo tanto, los costes de un jugador agregado de $n-1$ nodos sigue una distribución de probabilidad igual a $Beta(y; 1, n-1)$, según la siguiente demostración.

Proposición 4.4.3. *Los costes Y del jugador agregado de $n-1$ i.i.d. jugadores (con distribuciones uniformes en $[0,1]$) sigue una distribución $Beta(y; 1, n-1)$, con función de densidad $f(y) = (n-1)(1-y)^{(n-2)}$ y CDF $F[y] = Pr[Y \leq y] = 1-(1-y)^{n-1}$.*

Demostración. Dado que el coste del jugador agregado se corresponde con el valor mínimo declarado por los jugadores agregados (considerando los valores normalizados), la correspondiente CDF $F(\cdot)$ tendrá la siguiente forma.

$$F[y] = Pr[Y \leq y] = 1 - Pr[Y > y] \tag{4.4.2}$$

$$= 1 - \prod_{j=1}^{n-1} Pr[Y_j > y] \tag{4.4.3}$$

$$= 1 - \left(\int_y^1 1 d\varphi \right)^{n-1} \tag{4.4.4}$$

$$= 1 - (1-y)^{n-1} \tag{4.4.5}$$

Donde Y_j es la variable aleatoria asociada al valor normalizado del nodo j. Por lo tanto la función de densidad de probabilidad es

$$f(y) = (n-1)(1-y)^{(n-2)}.$$

La distribución $Beta$ se define [EHP00] según lo siguiente.

$$Beta(y; 1, n-1) = \frac{1}{B(1, n-1)} y^{1-1}(1-y)^{(n-1)-1} \tag{4.4.6}$$

$$= \frac{(1+(n-1)-1)!}{((n-1)-1)!}(1-y)^{n-2}, \tag{4.4.7}$$

donde $B(\cdot,\cdot)$ es la función Beta. Ahora es fácil comprobar que

$$f(y) = Beta(y; 1, n-1) \tag{4.4.8}$$

□

Estrategia del jugador Nuestro siguiente paso consiste en observar que el beneficio de un jugador depende únicamente de su estrategia. Todo jugador racional sabe que el resto de los jugadores seguirán una distribución uniforme e independiente. Un jugador egoísta puede desear seguir alguna estrategia que le ofrezca mejores resultados. Una posibilidad es intentar declarar valores normalizados que no sigan la distribución uniforme. En ese caso, el test de bondad (aceptado como perfecto) detectará y rechazará los valores declarados, sustituyendolos por otros regenerados por el sistema que siguen la distribución uniforme (por la forma de construir el algoritmo). Por otra parte, el jugador podría declarar valores siguiendo una distribución uniforme que no se corresponda exactamente con los reales (de alguna forma están correlados los valores declarados con los reales). En ese caso, QPQ no puede saber si los valores declarados son reales (es información privada) y utiliza éstos para decidir la asignación de la tarea. Tanto si son valores falsos como valores regenerados por el sistema, la variable aleatoria que los representa Z debe seguir una distribución uniforme.

Pasaremos a demostrar que en ambos casos, el jugador obtiene peores resultados cuando miente que cuando se comporta honestamente. Es decir, el jugador no tiene ningún incentivo para mentir. En primer lugar, evaluemos la esperanza de trabajo a realizar por los jugadores honestos.

Proposición 4.4.4. *La esperanza de trabajo normalizado $E[\bar{W}_i]$ realizado por un jugador honesto i es*

$$\frac{1}{n+n^2} \tag{4.4.9}$$

Demostración. Recordemos que un jugador i puede observar el sistema como un juego entre él y un jugador agregado de $n-1$ nodos. Por lo tanto, $Pr[d=i]$ es la

probabilidad de que el jugador i publique un valor menor que el valor declarado por el jugador agregador (en ambos casos, hablamos de valores normalizados).

$$E[\bar{W}_i] = \int_0^1 xPr[d = i]dx \tag{4.4.10}$$

$$= \int_0^1 x \int_x^1 (n-1)(1-y)^{(n-2)}dydx \tag{4.4.11}$$

$$= \frac{1}{n+n^2} \tag{4.4.12}$$

En esta demostración hemos utilizado el hecho de que el jugador agregado sigue la distribución propuesta en la Proposición 4.4.3. □

Proposición 4.4.5. *El coste normalizado total del trabajo realizado por un jugador agregado j (agregando a $n-1$ jugadores), con costes $x = \bar{c}_j$, no cambia cuando el jugador i (no incluido en el conjunto agregado) declara valores falsos $z = \hat{c}_i$.*

Demostración. Abusando de la notación, utilizaremos z para identificar los valores declarados por i tanto si pasa el test GoF como si ha sido regenerado por el sistema cuando no lo pasa. Sea Z la variable aleatoria asociada a estos valores. Existe una distribución de probabilidad bi-variada con densidad $f_{x,z}(x,z)$ que relaciona ambos valores x y z. Por las características del algoritmo, la distribución marginal de z debe ser uniforme, y por lo tanto, podemos asegurar que,

$$f_x(x) = \int_0^1 f_{x,z}(x,z)dz = 1 \tag{4.4.13}$$

$$f_z(z) = \int_0^1 f_{x,z}(x,z)dx = 1 \tag{4.4.14}$$

$$\int_0^1 \int_0^1 f_{x,z}(x,z)dzdx = 1. \tag{4.4.15}$$

Por otra parte, la esperanza de trabajo realizada por j es

$$E[\hat{W}_j] = \int_0^1 yPr[d = j]dy \tag{4.4.16}$$

$$= \int_0^1 y(n-1)(1-y)^{(n-2)} \int_0^1 \int_y^1 f_{x,z}(x,z)dzdxdy \tag{4.4.17}$$

$$= \int_0^1 y(n-1)(1-y)^{(n-2)} \int_y^1 \int_0^1 f_{x,z}(x,z)dxdzdy, \tag{4.4.18}$$

donde $E[\hat{W}_j]$ es la esperanza de trabajo realizada por el jugador agregado j cuando el jugador i miente. Pero, dado que tenemos marginales uniformes, la expresión anterior se convierte en,

$$E[\hat{W}_j] = \int_0^1 y(n-1)(1-y)^{(n-2)} \int_y^1 1dzdy \tag{4.4.19}$$

$$= \int_0^1 y(n-1)(1-y)^{(n-2)}(1-y)dy \tag{4.4.20}$$

$$= \int_0^1 y(n-1)(1-y)^{(n-1)}dy \tag{4.4.21}$$

$$= \frac{n-1}{n+n^2}. \tag{4.4.22}$$

Que es igual al esfuerzo realizado por el jugador agregado j cuando i es honesto:

$$\int_0^1 y(n-1)(1-y)^{(n-2)} \int_y^1 1dxdy = \int_0^1 y(n-1)(1-y)^{(n-1)}dy \tag{4.4.23}$$

$$= \frac{n-1}{n+n^2}. \tag{4.4.24}$$

□

Es decir, el jugador j realizará la misma cantidad de trabajo en esperanza independientemente del comportamiento del jugador contrario i (no incluido en la agregación). El que i sea honesto o mentiroso no afecta por tanto al trabajo realizado por j. Esto nos permite demostrar que la mejor estrategia de un jugador es decir la verdad.

Proposición 4.4.6. *Un jugador i nunca realiza más trabajo (normalizado y en esperanza) cuando es honesto. Es decir,*

$$E[\bar{W}_i] \leq E[\hat{W}_i], \tag{4.4.25}$$

donde $E[\hat{W}_i]$ es el trabajo realizado por el jugador i cuando es mentiroso.

Demostración. Supongamos, por contradicción, que dicha proposición es falsa. Por lo tanto, debe haber un conjunto de tareas para las cuales, si i miente, éste realizará menos trabajo (en esperanza) y por tanto, $E[\bar{W}_i] > E[\hat{W}_i]$. Además, y utilizando la Proposición 4.4.5, se sabe que el jugador agregado j, realizará la misma cantidad de trabajo, $E[\bar{W}_j] = E[\hat{W}_j]$. De aquí se concluye que: $E[\bar{W}_i]+E[\bar{W}_j] > E[\hat{W}_i]+E[\bar{W}_j]$. Pero si la ecuación anterior fuera cierta, entonces se concluye que QPQ no sería óptimo, y por lo tanto existía un mecanismo que podría asignar las tareas simulando la asignación que se produce cuando i miente (pero sin la necesidad de que i mintiera). Pero esto claramente contradice la proposición 4.4.2. Por lo tanto, la mejor estrategia (en el sentido de que mejora su coste normalizado) de un jugador es decir la verdad. □

Este resultado se puede complementar con la siguiente propiedad.

Proposición 4.4.7. *Cuando un jugador i publica valores falsos siguiendo una distribución de valores no uniforme o una distribución independiente de sus valores reales (la variable Z es independiente de X), entonces el jugador realiza un esfuerzo igual a $E[\hat{W}_i] = \frac{1}{2n}$.*

Demostración. Los valores de z utilizados por el sistema para la asignación de las tareas, siguen una distribución independiente de los costes reales del jugador i, y por

lo tanto,

$$E[\hat{W}_i] = \int_0^1 xPr[d=i]dx \tag{4.4.26}$$

$$= \int_0^1 x \int_0^1 \int_z^1 (n-1)(1-y)^{(n-2)} dydzdx \tag{4.4.27}$$

$$= \frac{1}{2n} \tag{4.4.28}$$

□

De este resultado, proposición 4.4.4, y Eq. 4.4.1, se puede afirmar el siguiente teorema.

Teorema 4.4.8. *Dado que $n \geq 2$, se puede afirmar que $E[\bar{W}_i] = \frac{1}{n+n^2} < \frac{1}{2n} = E[\hat{W}_i]$. Por lo tanto, dado que la suma de trabajo en esperanza más la utilidad esperada es una constante igual a $\frac{1}{2}$, todos los jugadores obtienen mejores resultados normalizados cuando son honestos.*

Utilidad esperada real Nótese que el trabajo normalizado esperado para un jugador es igual a $\frac{1}{n+n^2}$. Pero podríamos preguntarnos sobre la cantidad de trabajo real (no normalizada) que realizará dicho jugador. Esta cantidad se puede calcular fácilmente según lo siguiente.

Teorema 4.4.9. *Para cada jugador i, la utilidad real obtenida es*

$$E[U_i] = \int_\Omega x f_i(x)(1-(1-F_i(x))^{n-1})dx. \tag{4.4.29}$$

donde el coste real del jugador i sigue una variable aleatoria continua con soporte en Ω, función de densidad $f_i(\cdot)$, y CDF $F_i(\cdot)$.

Demostración. Sea $Y_j = PIT(X_j) = F_j(X_j)$ la variable aleatoria que representa los valores normalizados de coste para el jugador $j \neq i$ y $Y = \min_{j \neq i} \{Y_j\}$. Entonces la

esperanza de la utilidad (real) del jugador i es:

$$E[U_i] = \int_\Omega x f_i(x) Pr(Y \leq F_i(x)) dx \tag{4.4.30}$$

$$= \int_\Omega x f_i(x)(1 - (1 - F_i(x))^{n-1}) dx, \tag{4.4.31}$$

donde se ha utilizado el hecho de que $Pr[Y \leq y] = 1-(1-y)^{n-1}$ (Proposición 4.4.3).

□

Justicia

El siguiente resultado, combinado con la Proposición 4.4.1, nos permitirá demostrar que todos los jugadores ejecutarán en esperanza una cantidad similar de tareas, incluso cuando alguno o todos los jugadores son deshonestos o irracionales.

Proposición 4.4.10. *Sean $X_1, X_2, \ldots, X_n$ i.i.d. n variables aleatorias continuas, entonces:*

$$Pr(X_i \leq \min_{j \neq i} \{X_j\}) = \frac{1}{n}.$$

Demostración.

$$Pr(X_i \leq \min_{j \neq i} \{X_j\}) = \int_{-\infty}^{\infty} f(y)[\int_{-\infty}^{y} f(x)dx]^{(n-1)} dy \tag{4.4.32}$$

$$= \int_{-\infty}^{\infty} f(y)[F(y)]^{(n-1)} dy \tag{4.4.33}$$

$$= \int_{-\infty}^{\infty} [F(y)]^{(n-1)} dF(y) \tag{4.4.34}$$

$$= \left[\frac{\{F(y)\}^n}{n}\right]_{-\infty}^{\infty} \tag{4.4.35}$$

$$= \frac{1}{n} \tag{4.4.36}$$

□

Dado que el sistema asigna las tareas utilizando los costes normalizado y éstos siguen n i.i.d. variables aleatorias (Proposición 4.4.1), QPQ asegura que la cantidad de tareas ejecutadas por cada jugador es $|T|/n$, donde T es el conjunto de tareas a realizar y n es el número de jugadores.

Por lo tanto, QPQ no sólo ofrece una gantaría de utilidad para los jugadores honestos. Además proporciona propiedades interesantes en situaciones en las que los jugadores pueden tener dificultades para evaluar correctamente los costes (irracionalidad). Esto es así, dado que QPQ garantiza que las tareas se repartirán de forma equitativa incluso en situaciones en las que los nodos son irracionales o deshonestos. QPQ asigna el trabajo de forma justa y óptima con respecto a los valores de coste normalizados. Claramente, un jugador no racional podrá ejecutar un esfuerzo mayor al que le correspondería si fuera honesto, pero siempre su utilidad será mayor o igual que si las tareas se le hubieran asignado de forma aleatoria. Esto hace que los jugadores irracionales tengan el interés en participar en QPQ si lo comparan con un mecanismo de asignación aleatoria. Además, la reducción de utilidad no afecta a los demás jugadores y es consecuencia únicamente de su propia ignorancia.

Corolario 4.4.11. *En QPQ, los jugadores realizarán un porcentaje de tareas (en esperanza) igual a* $\frac{1}{n}$ *y por lo tanto realizarán un total de* $\frac{|T|}{n}$ *tareas.*

Demostración. Los valores declarados siguen distribuciones continuas e identicamente distribuidas (según la uniforme) en $[0, 1]$, de la Proposición 4.4.1, y aplicando la Proposición 4.4.10, cada jugador ejecutará una proporción de tareas igual a $\frac{1}{n}$. □

Cotas

Finalmente, creemos interesante definir alguna métrica que mida la eficiencia del sistema y que nos pueda servir como guía para determinar si QPQ se degrada ante comportamientos egoistas. Utilizando concepto similares al “precio de la anarquía”(en inglés “price of anarchy”) [KP09], definimos la *eficiencia* como el ratio entre

la utilidad en algún tipo de equilibrio (normalmente será el "peor equilibrio" posible) y la utilidad según la solución propuesta.

Es evidente que la utilidad normalizada de un jugador debe encontrarse entre 0, cuando el nodo ejecuta todas las tareas, y $\frac{1}{2}$, situación que se produce cuando el jugador no ejectua ninguna tarea. Pero existen otros límites que pueden ser más interesantes para su utilización como referencia de bondad del algoritmo. Por una parte, cuando el nodo ejecuta $\frac{1}{n}$ tareas completamente al azar su trabajo esperado es igual a $\frac{1}{2n}$. Por otra parte, el máximo benecifio que un jugador i puede ocurrir cuando las tareas se corresponden exactamente con sus tareas más baratas. En ese caso, la utilidad normalizada debe ser igual a:

$$E[\bar{U}_i^*] = \frac{1}{2} - E[\bar{W}_i^*] \tag{4.4.37}$$

$$= \frac{1}{2} - \int_0^{\frac{1}{n}} x dx \tag{4.4.38}$$

$$= \frac{1}{2} - \frac{1}{2n^2}. \tag{4.4.39}$$

Aunque este último caso tiene una probabilidad nula (es imposible que pueda suceder) proponemos utilizarlo como base para definir nuesta noción de eficiencia.

Definición 4.4.1 (Medida de eficiencia)**.** . Definimos la medida de *eficiencia* de un algoritmo M para la asignación de tareas en entornos egoistas como la relación entre la utilidad esperada bajo cierto equilibrio y $E[\bar{U}_i^*] = \frac{1}{2} - \frac{1}{2n^2} = \frac{n^2-1}{2n^2}$. Es decir,

$$\text{Eficiencia} = \frac{E[\bar{U}^M]}{E[\bar{U}_i^*]} = \frac{2n^2 E[\bar{U}^M]}{n^2-1}. \tag{4.4.40}$$

Utilizando dicha definición, podemos proceder a calcular la eficiencia de QPQ:

$$\text{Eficiencia de QPQ} = \frac{2n^2(\frac{1}{2} - \frac{1}{n+n^2})}{n^2-1} = \frac{n^2 - (\frac{2n}{n+1})}{n^2-1} > \frac{n^2-2}{n^2-1}. \tag{4.4.41}$$

Según esta definición,la eficiencia de QPQ se acerca mucho a la unidad 1 cuando el número de jugadores es relativamente alto. Por ejemplo, con sólo 10 nodos, la eficiencia de QPQ es $0{,}991$.

4.5. Aspectos prácticos de QPQ

En esta sección nuestro objetivo es analizar las restricciones de QPQ en su implementación práctica en entornos reales. De lo visto anteriormente, creemos que las capacidades de computación y de comunicaciones requeridas por el algoritmo son abordables por el estado de la tecnologia actual. No aseguramos que la implementación sea una tarea sencilla ya que es necesario enfrentarse a múltiples retos tecnológicos. Algunos trabajos [MYCH10] hacen una relación de estos retos. En cualquier caso, creemos que, aunque pueda resultar complejo sería posible implementar de forma práctica QPQ.

Hasta ahora, hemos supuesto que las distribuciones podían ser conocidas de forma perfecta y que los tests de bondad eran ideales. Para nuestro siguiente paso debemos distinguir entre esa situación ideal y una aproximada. Uno de los primeros conceptos que deben ser redefinido es el de *estrategia*. De hecho se puede hablar de *estrategia completamente cierta* y una *estrategia aproximadamente cierta*. Ambas definiciones se pueden encontrar en los trabajos de Jackson y Sonnenschein [JS07].

Cuando los juegos se encuentran enlazados, puede no existir una estrategia completamente cierta ya que será necesario realizar aproximaciones a la probabilidad del jugador si ésta fuera discreta. En el caso del algoritmo QPQ existe una estrategia completamente cierta (esto es, decir siempre la verdad), pero existe una cierta probabilidad de que el test de bondad rechace el valor declarado (falso negativo) y lo sustituya por otro. El sistema no podrá distinguir si la estrategia del jugador es o no completamente cierta y por ello hablamos de estrategia aproximadamente cierta. Lo relevante, en cualquier caso, es que la estrategia aproximadamente cierta se aproxima todo cuanto queramos a la completamente cierta si aumentamos el número de rondas (y la longitud del histórico) lo suficiente. Esto es debido a que el test de bondad o ajuste reduce la probabilidad de rechazar los valores cuando el número de muestras aumenta.

Con estas consideraciones, nuestro siguiente paso es definir de forma práctica QPQ incorporando las modificaciones y herramientas estadísticas oportunas. Por la dificultad matemática que implican dichas herramientas, hemos comprobado median-

te simulación que cualquier *estrategia aproximadamente cierta* produce un nivel de utilidad que converge al ideal cuando el número de muestras crece.

4.5.1. Generación del valor alternativo

Anteriormente se ha comentado que una forma de "castigar" a un jugador mentiroso es generar un valor aleatorio que sustituya al valor declarado por dicho jugador. Este valor debe ser desconocido de antemano por todos los jugadores, pero una vez generado debe ser único y fácilmente comprobable por el resto de los jugadores. Además, sería recomendable que no sean necesarios nuevos mensajes que puedan hacer del protocolo un proceso lento o complejo.

Una posibilidad para la generación de los valores aleatorios que sustituyan a los valores falsos de los mentirosos es utilizar una función hash indexada con los costes de los demás nodos. Alternativamente, se puede solicitar un valor aleatorio alternativo a cada jugador y aplicar la función hash a estos valores (exceptuando el valor del jugador en cuestión).

Otra posibilidad es utilizar técnicas similares a los procedimientos propuestos por Aumann, Maschler y Stearns [AMS95] para generar loterías controladas conjuntamente ("jointly controlled lotteries"). Por ejemplo, para dos jugadores, se puede generar pidiendo un valor aleatorio a cada jugador, y sustituir el valor del mentiroso por la suma de dichos números si dicha suma es menor que 1 o uno menos la suma en caso contrario. Es fácil de demostrar que si uno de los jugadores declara valores aleatorios según una distribución uniforme, entonces el proceso genera valores aleatorios según una distribución uniforme sobre $[0, 1]$, independientemente de lo que haga el otro jugador. En consecuencia, ninguno de los jugadores podría afectar el valor a utilizar cuando falla el test de bondad.

4.5.2. Aplicación práctica del test de bondad

Para que QPQ detecte si un jugador miente sobre su transformada es necesario realizar un test de bondad (Gof) a las últimas muestras declaradas (histórico). El juego

repetido se construye pues pidiendo un valor al jugador y comprobando que el valor declarado, junto con su histórico, se corresponde con una determinada probabilidad a la distribución uniforme. Es realmente impresionante la cantidad de tests que se han propuesto a lo largo de la historia. Algunos de ellos se aplican a distribuciones simples y otros a distribuciones continuas. Seguramente, con el tiempo, existirán test de aceptación muy elaborados y con propiedades muy interesantes. Nuestra propuesta, sin embargo, se basa en un test ampliamente conocido y utilizado. Se trata del test de Kolmogorov-Smirnov (en adelante KS) ([Kol33, Smi39a]).

El test KS calcula la mayor distancia entre la función de distribución empírica y la función de distribución acumulada que queremos comprobar (en inglés se denomina "null distribution"). El test de KS viene definido por la siguiente expresión:

$$D = max_{1 \leq i \leq n}(F(x_i) - \frac{i-1}{n}, \frac{i}{n} - F(x_i))$$

Donde $F(\cdot)$ es la función de distribución teórica que se desea comprobar, n es el número de muestras disponibles y ordenadas $(x_1, x_2, \cdots, x_n)$.

Lo que hace el test de KS tan versátil es que la distribución de la distancia D no depende de la distribución de probabilidad teórica (hipótesis nula). Varios autores tales como Smirnov (1939) [Smi39b], BirnBaum y Tingey (1951) [BT51], Massey (1951) [Mas51] y Miller (1956) [Mil56] han obtenido expresiones exactas y aproximadas para la función de distribución de la variable aleatoria D en función del número de muestras disponibles. Debido a la complejidad de dichas expresiones se suelen utilizar tablas con los percentiles más habituales.

Otros GoF están diseñados especialmente para las distribuciones uniformes. Por ejemplo, hay una familia completa de test basados en la distancia entre muestras. Siguiendo la notación anterior, supongamos que los valores de las muestras están ordenadas $x_1 \leq x_2 \leq \cdots \leq x_n$. Estos tests calculan las diferencias de primer nivel:

$$d_i = x_i - x_{i-1}, i = 1, 2, \cdots, n+1$$

donde $x_0 = 0$ y $x_{n+1} = 1$.

Es fácil comprobar que si las muestras provienen de una distribución uniforme, las distancias se distribuyen según la distribución de Dirichlet para la dimensión $n+1$

dada por:

$$D_{n+1}(\mathbf{d}; 1, \cdots, 1) = n!,$$

donde $\mathbf{d} = (d_1, d_2, \cdots, d_{n+1}), 0 < d_i < 1, i = 1, \cdots, n+1$ y $\sum_{i=1}^{n+1} d_i = 1$.

También es conocido que la esperanza, varianza y covarianza de las distancias de una muestra de tamaño n se corresponde con:

$$E(d_i) = \frac{1}{n+1},$$
$$Var(d_i) = \frac{n}{(n+1)^2(n+2)},$$
$$Cov(d_i, d_j) = \frac{-1}{n+1)^2(n+2)}, i \neq j$$

Igualmente, se puede demostrar que:

$$Pr[d_1 > t_1, \cdots, d_j > t_j] = (1 - t_1 - \cdots - t_j)^n$$

Para $t_j \geq 0$, $j = 1, \cdots, i$, y $t_1 + \cdots + t_j < 1$.

Numerosos test de bondad (GoF) han sido propuestos basándose en dichas distancias $(d_1, d_2, \cdots, d_{n+1})$. Pyke [Pyk65] ofrece una revisión muy completa sobre la teoría y aplicación de estos tests. Son numerosos los trabajos al respecto (sin ser exhaustivos, se remite al lector a [JG04], [MB01], [TO69]).

Todos los tests Gof tienen la propiedad de que el error en rechazar una muestra válida decrece con el tamaño de la muestra. Si un jugador es correcto y declara sus valores según sus preferencias correctas, es evidente que al aplicar la transformada PIT se generarán valores según la distribución uniforme, y con alta probabilidad el test Gof aplicado aceptará el valor.

Para la elaboración de las simulaciones se han considerado otras alternativas posibles, tales como:

1. $D_{min} = min(d_1, d_2, \cdots, d_{n+1})$
2. $D_{max} = max(d_1, d_2, \cdots, d_{n+1})$
3. $D_{KS} = \sum_{i=1}^{n+1} |d_i - \frac{1}{n+1}|$

4. $D_{GW} = \sum_{i=1}^{n+1}(d_i - \frac{1}{n+1})^2$

El test D_{KS} se conoce por el test de Kendall y Sherman. El test de D_{GW} es un trabajo de Greenwood. Estos tests son ampliamente conocidos y están descritos en obras generales de estadística, por ejemplo [MB01].

El test D_{min} rechaza las muestras cuando el valor de D_{min} es suficientemente bajo. En cambio los tests D_{max}, D_{KS} y D_{GW} rechazan las muestras para valores altos. Aunque se han demostrado suficientemente potentes, la elección inicial de los tests se realizó principalmente por su sencillez en la formulación.

Surge la cuestión práctica sobre el valor del umbral de aceptación que debemos asignar al test de bondad para que resulte adecuado. Técnicamente este concepto está relacionado con lo que en estadística se conoce por "p-value", véase por ejemplo [Muk00]. Como veremos en las simulaciones, un test demasiado estricto podría rechazar valores correctos con frecuencia, y por lo tanto penalizar a los jugadores honestos. Por el contrario, un test demasiado abierto haría que los mentirosos pudieran manipular sus valores y engañar al mecanismo. Esta reflexión nos lleva a la conclusión de que el umbral debería ser adaptativo. La idea que proponemos en el algoritmo QPQ es construir una secuencia de valores de aceptación que inicialmente sea restrictivo, pero que aumente según el número de rondas y tienda asintóticamente a $100\,\%$. Una posibilidad sería construir la secuencia de valores de aceptación definida por

$$\text{p-Thr}_k = 1 - \frac{1}{log(k+1)} \tag{4.5.1}$$

Recuérdese que k representa la ronda o el instante de tiempo.

Pero esta solución implicaría una longitud de muestras sin límite. En la práctica, esta solución sería inviable ya que todos los nodos deberían almacenar un histórico con tamaño creciente de todos los demás jugadores. Es un volumen de información prohibitivo y nada realizable.

En nuestros estudios hemos intentado reducir el histórico de control. El problema al reducir este histórico es que pequeñas modificaciones a la distribución uniforme serían indetectables en la práctica. Por ejemplo, un nodo egoísta podría simular datos siguiendo un distribución $Beta(1, 0{,}9)$ y no ser detectado con gran probabilidad si se

reduce el histórico a 50 o 100 valores. A la larga, este nodo mentiroso conseguiría un pequeño beneficio que se acumularía con el tiempo.

La solución por tanto es utilizar el conocimiento que se tiene sobre el beneficio acumulado del jugador para abrir o cerrar el criterio de aceptación. Cuando un jugador tenga un beneficio acumulado superior a la media esperada, se puede cerrar el umbral de aceptación (más restrictivo). Si por el contrario, un jugador obtiene una media ligeramente inferior al valor esperado, se puede abrir el umbral y aceptar con más facilidad los valores declarados por el jugador.

Si un jugador es correcto, y declara sus valores según sus preferencias correctas, es evidente que al aplicar la transformada PIT se generarán valores según la distribución uniforme, y con alta probabilidad el test aplicado aceptará el valor. Si, además, su beneficio acumulado tiene un valor razonable, el test aceptará con bastante facilidad los valores declarados. Por el contrario, cuando un jugador manipula sus valores, el test lo podrá detectar mediante dos controles. Por una parte el test de bondad rechazará los valores claramente erróneos. Además, irá cerrando el valor de aceptación si el jugador acumula más beneficio de lo habitual.

Para ello, se propone una ligera modificación a la secuencia de control. Se fija la longitud del histórico de control a un valor reducido (por ejemplo de entre 50 o 100 valores). Es decir, cada nodo aplica el test de bondad de ajuste utilizando únicamente un histórico reducido (los últimos 50 o 100 valores). El umbral de aceptación se modifica según la siguiente expresión:

$$\text{p-Thr}_k = \frac{1}{\log(k+1)^{\delta*(1-(\mu_k-\mu)\cdot\sqrt{k})}}, \tag{4.5.2}$$

donde δ es un parámetro de ajuste, μ es la utilidad esperada (normalizada) de los jugadores y μ_k es la utilidad normalizada real que ha obtenido el jugador en la ronda k. Para ilustrar la forma de esta función se remite al lector a la figura 4.7, que representa el umbral de aceptación como una función del número de jugadores y su utilidad esperada téórica. La forma de dicha función es claramente empírica, aunque las simulaciones realizadas han demostrado un comportamiento muy razonable según los objetivos marcados en nuestro trabajo.

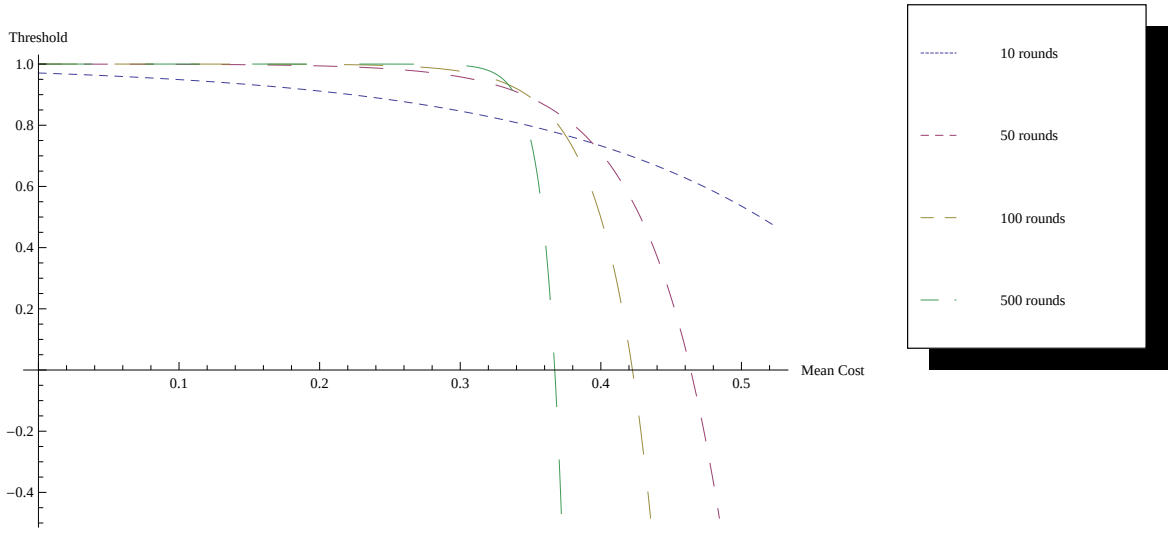

Figura 4.7: Umbral de aceptación a varias rondas. Según se incrementa el número de rondas, el sistema dispone de más información y por tanto tiende a creer más a los jugadores.

Una de las razones que nos ha llevado a proponer dicha función de umbral ha sido la búsqueda de un cierto "pago" al inicio del sistema. En Internet es posible obtener identidades nuevas con un coste muy bajo. Eso significaría que los jugadores podrían entrar y salir del sistema casi sin coste. Esa situación podría ser utilizada por los jugadores si pueden obtener un benecifio rápido. La idea era por tanto que inicialmente QPQ penalice a los jugadores noveles. Nuestra propuesta fuerza a que el sistema trate a los jugadores noveles con asignaciones aleatorias (no cree en los valores declarados con mayor probabilidad) y, conforme el sistema tenga más información del jugador, confiará más en los valores declarados. De esta forma, el sistema obliga a que los jugadores "paguen" al principio para evitar los incentivos de entrar y salir. Obsérvese que esta política no afecta a los jugadores veteranos ya que se produciría una situación similar a la estudiada anteriormente en la que QPQ trataba a los jugadores honestos que juegan contra jugadores irracionales.

4.5.3. Formulación práctica de QPQ

Con estos aspectos prácticos en mente, el algoritmo final de QPQ para jugadores independientes es el representado por el algoritmo 7.

Algorithm 7 Implementable Quid Pro Quo mechanism (code for node i, and a generic task t, omitted)

Estimate the cost c_i of the task
Publish the normalized cost $\bar{c}_i = PIT(c_i)$
Wait to receive the normalized costs $\bar{c}_j$ from the other players
for all $j \in N$ **do**
    Let $p\text{-}th_j \Leftarrow \frac{1}{\log(k+1)^{\delta(1-(\mu_{j,k}-\mu)\cdot\sqrt{k})}}$
    if not $KSTest(\bar{c}_j, Historic_j, p\text{-}th_j)$ **then**
        $\bar{c}_j \leftarrow Random(\bar{c}_{-j})$
    end if
    $Historic_j \leftarrow Historic_j \cup \{\bar{c}_j\}$
end for
Let $d = \underset{j \in N}{argmin}\ \bar{c}_j$
if $d = i$ **then**
    execute the task
else
    do nothing (node d will execute the task)
end if
for all $j \in N$ **do**
    recompute $\mu_{j,k+1}$
end for

4.6. Resultados experimentales

Aunque las propiedades de QPQ básico se han podido comprobar de forma analítica, se ha acudido a simulaciones para estudiar su comportamiento en situaciones concretas. Dado que las expresiones analíticas asociadas a los test de bondad son relativamente complejas, y en general no son tratables de forma analítica, se ha optado por estudiar su comportamiento mediante simulaciones. La idea es que dichas simulaciones puedan aportar una visión estadística del comportamiento de QPQ.

Para la realización de las simulaciones se ha tenido que fijar el mecanismo de aceptación del test de bondad. Como se recordará, existe el aspecto práctico de cómo limitar el valor de aceptación. Se propuso realizar un test adaptativo que se abriera o cerrara según el beneficio del nodo. Es decir, cuando un nodo está ganando más de lo esperado, el test Gof se vuelve más restrictivo y viceversa.

Para la realización de las simulaciones se ha utilizado la herramienta de software libre llamada R-Project. Es un entorno para la simulación y estudio estadístico muy completo y que puede correr en diferentes sistemas operativos. En particular se ha deseado comprobar varios aspectos concretos. A saber:

1. Hasta qué punto el test de bondad podía afectar al beneficio de los jugadores. La cuestión que deseamos estudiar es si los falsos negativos del test KS podrían afectar seriamente a la utilidad de los jugadores honestos.

2. Comprobar la potencia del test para detectar diferentes niveles de mentiras. Para ello se han simulado jugadores mentirosos con distribuciones muy cercanas a la uniforme.

3. Comprobar que un jugador no racional obtiene una utilidad equivalente a un algoritmo en el que las decisiones sean aleatorias.

4. Estudiar el comportamiento de QPQ cuando se modifican los parámetros básicos, tales como la longitud del histórico, el factor del test KS, etc.

Con este fin, la estrategia se ha basado en simular y analizar los siguientes escenarios:

1. Se simula un juego en el que no se aplica el test de aceptación. Esta simulación se repite mediante los parámetros de ajuste. Los resultados de esta simulación servirán como elemento de control.

2. Se repetirán las simulaciones anteriores pero aplicando el test de aceptación. Los resultados así obtenidos se comprobarán con los de control con el objetivo de analizar si existe o no una pérdida de beneficio por falsos rechazos.

3. Ambas pruebas se repetirán bajo diferentes formas de mentira, analizando de nuevo la bondad de QPQ.

Condiciones de la simulación

Cada simulación se ha repetido un determinado número de veces. El número de repeticiones realizadas era un parámetro del experimento. Las gráficas y los valores tabulados que se incluyen en este apartado se corresponden con un valor de 3.000 repeticiones. La longitud del histórico es de 50, aunque la simulación ha contemplado diferentes longitudes de juego. Es decir, aunque el histórico se mantenía fijo en 50, el juego proseguía durante un número mayor de rondas (100, 500, .., 1000 rondas). La razón de separar ambos conceptos es que se deseaba conocer el comportamiento de QPQ a la largo del tiempo, aunque se almacenara un número limitado de muestras. Como veremos, esto permite comprobar que pequeñas variaciones en las distribuciones uniformes no pasan desapercibidas por QPQ, aunque no sean detectables por el test de KS con un histórico pequeño.

Para cada simulación, los datos se representan según la distribución empírica de los valores obtenidos, así como su media.

Comportamiento del test KS

El primer punto a estudiar responde a la duda de si el test de KS puede generar un daño significativo al jugador como consecuencia de provocar falsos negativos. Para ello se ha simulado una serie de juegos en el que dos jugadores declaran sus valores

correctamente (siguiendo la uniforme) y en el que QPQ aplica o no el test de KS. Cuando QPQ no aplica el test de KS, simplemente se determina el menor valor de ambos jugadores, pero cuando se aplica el test de KS, el sistema comprueba si los valores anunciados pueden prevenir de una uniforme. Evidentemente, la potencia del test KS viene dada como función del histórico, pero éste está limitado a 50 valores.

Las gráficas 4.6 y 4.6 representan la distribución y el boxplot del coste medio que dedica un jugador en un juego de 100 rondas. Ninguno de los dos jugadores es un nodo mentiroso. Esta gráfica nos sirve de elemento de control y nos permite hacer un contraste con el mismo juego pero aplicando el test de KS. Esta situación se representa en las gráficas 4.6 y 4.6.

Como puede comprobarse, la pérdida de rendimiento causada por falsos negativos es mínima y apenas apreciable en estos escenarios. El mismo escenario se ha repetido para un número superior de rondas (pero manteniendo el histórico a 50), observando cómo el test adaptativo de KS influye en la utilidad de los jugadores de forma poco apreciable. Esta situación se puede observar en las gráficas 4.6, 4.6, 4.6 y 4.6, correspondientes a dos jugadores sinceros que juegan durante 1.000 rondas sin y con control.

Las tablas o cuadros 4.1 y 4.2 resumen los valores medios obtenidos por los dos jugadores a lo largo de varias rondas, siendo respectivamente los obtenidos sin aplicar el control KS y con el control.

n	100 rondas	300 rondas	600 rondas	1000 rondas
1	0.3341908	0.3336160	0.3338520	0.3331433
2	0.3326126	0.3334818	0.3326952	0.3335234

Cuadro 4.1: Medias de coste de 2 jugadores sinceros sin control

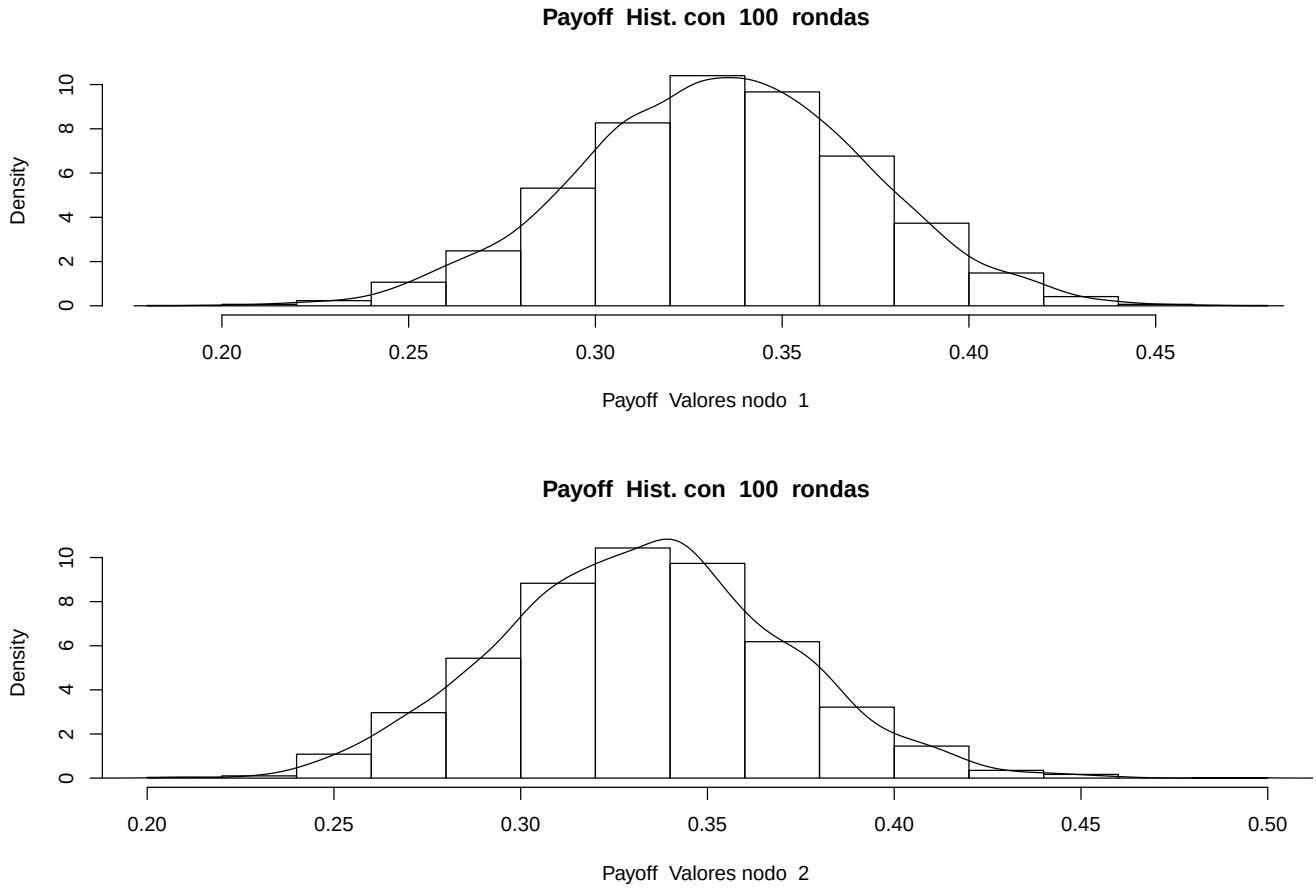

Figura 4.8: Dos jugadores honestos sin control (100 rondas).

Comportamiento según el número de jugadores

Cuando el número de jugadores aumenta, los falsos negativos disminuyen su impacto y hacen que el valores de la utilidad sean similares al aplicar o no el test KS. Este comportamiento es el esperado ya que un falso negativo tiene una menor probabilidad de convertirse en un coste para el jugador. Al aumentar el número de jugadores disminuye la probabilidad de ejecutar la tarea.

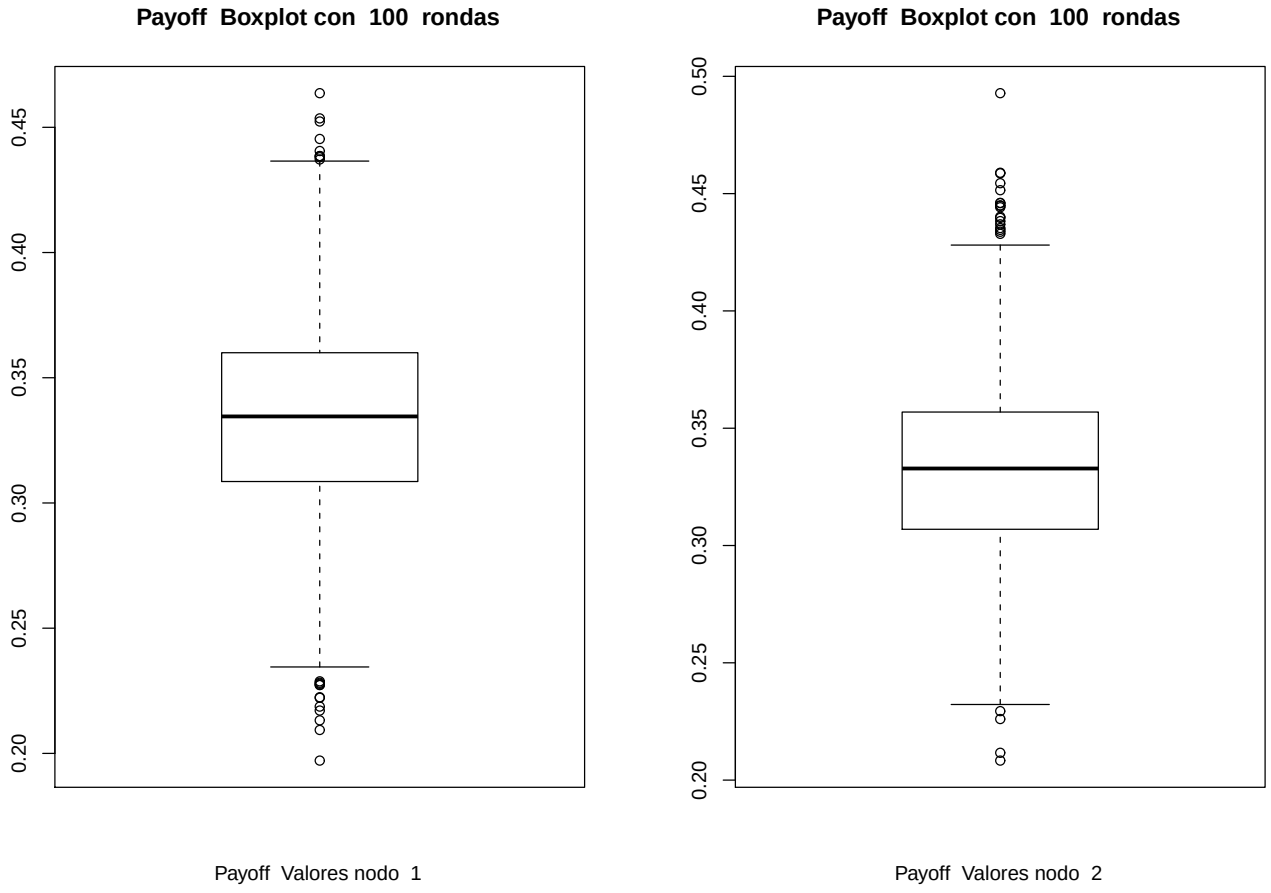

Figura 4.9: Dos jugadores honestos sin control (100 rondas).

Comportamiento ante mentirosos

La siguiente cuestión sería analizar hasta qué punto un jugador mentirosos puede manipular el algoritmo y obtener mejoras en su utilidad. Para ello se han utilizado varias distribuciones "próximas" a la uniforme y se ha observado si el jugador podía engañar a QPQ. Además, se ha estudiado si tales manipulaciones afecta a la utilidad de los jugadores honestos.

El problema que surge al intentar definir correctamente como "proximidad" a una distribución uniforme. Y es que no existe una única norma o distancia en espacios de dimensión infinita. Dado que el espacio de distribuciones de probabilidad en el

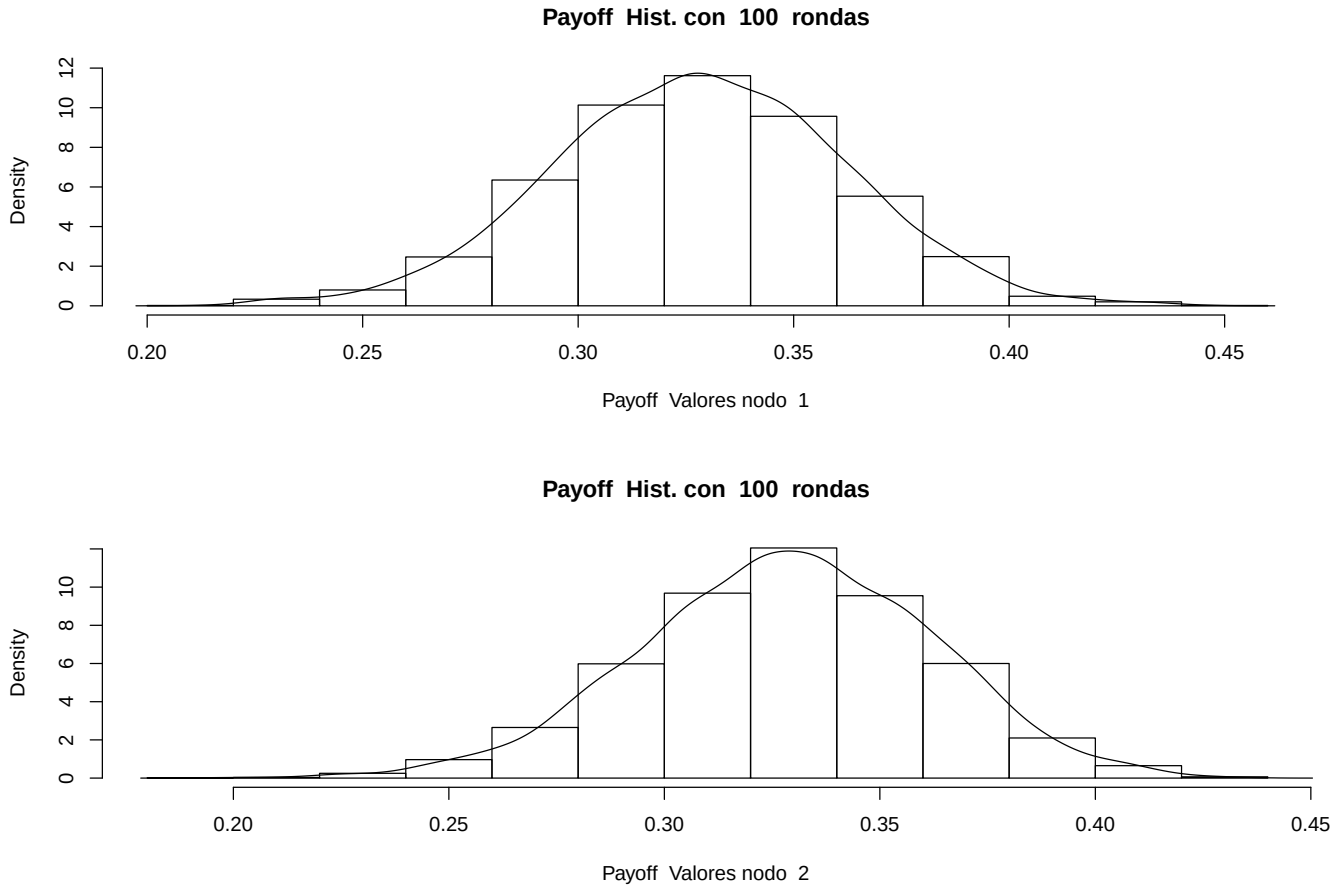

Figura 4.10: Dos jugadores honestos con control (100 rondas).

soporte $[0, 1]$ tiene dimensión infinita, sería posible utilizar diferentes nociones de distancia, y por lo tanto, tendríamos infinitas formas de establecer estrategias para nuestro jugador mentiroso.

Aprovechando pues que la distribución uniforme se puede considerar como un caso particular de la distribución Beta, en las simulaciones se ha utilizado una familia de distribuciones Beta jugando con los valores que la definen. Llegando en algunos casos a tener una similitud con la uniforme difícilmente distinguible. Por ejemplo, en la gráfica 4.6 se pueden observar los diferentes histogramas correspondientes a una $Uniforme$, a una $Beta(1, 0{,}9)$ y a $Beta(1, 0{,}7)$ con un histórico de 100 muestras.

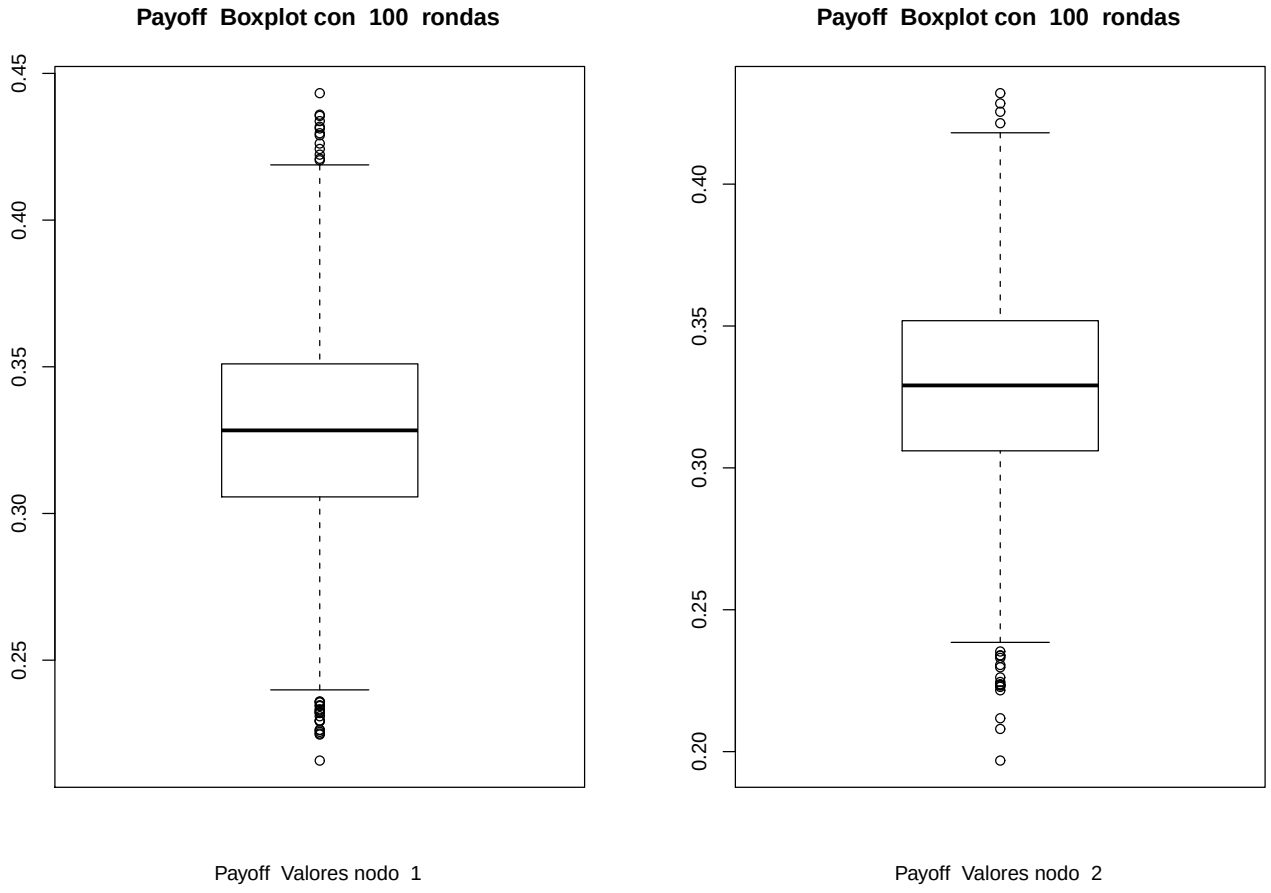

Figura 4.11: Dos jugadores honestos con control (100 rondas).

Se puede observar la gran similitud entre ellas y la dificultad a la que se enfrenta el sistema para detectar a los mentirosos. Obsérvese que el número de muestras es superior al propuesto en nuestro trabajo que se ha limitado a 50 muestras. Sin la modificación propuesta para adaptar el umbral según el beneficio acumulado, sería necesario un histórico mucho más amplio.

Las simulaciones realizadas siguen el mismo patrón ya comentado. Primeramente se ha aplicado QPQ pero sin el test de KS, en el que se puede observar un beneficio claro del jugador mentiroso, así como una pérdida del jugador honesto. Las gráficas 4.6 y 4.6 representan el beneficio de dos jugadores durante 100 rondas en el que uno

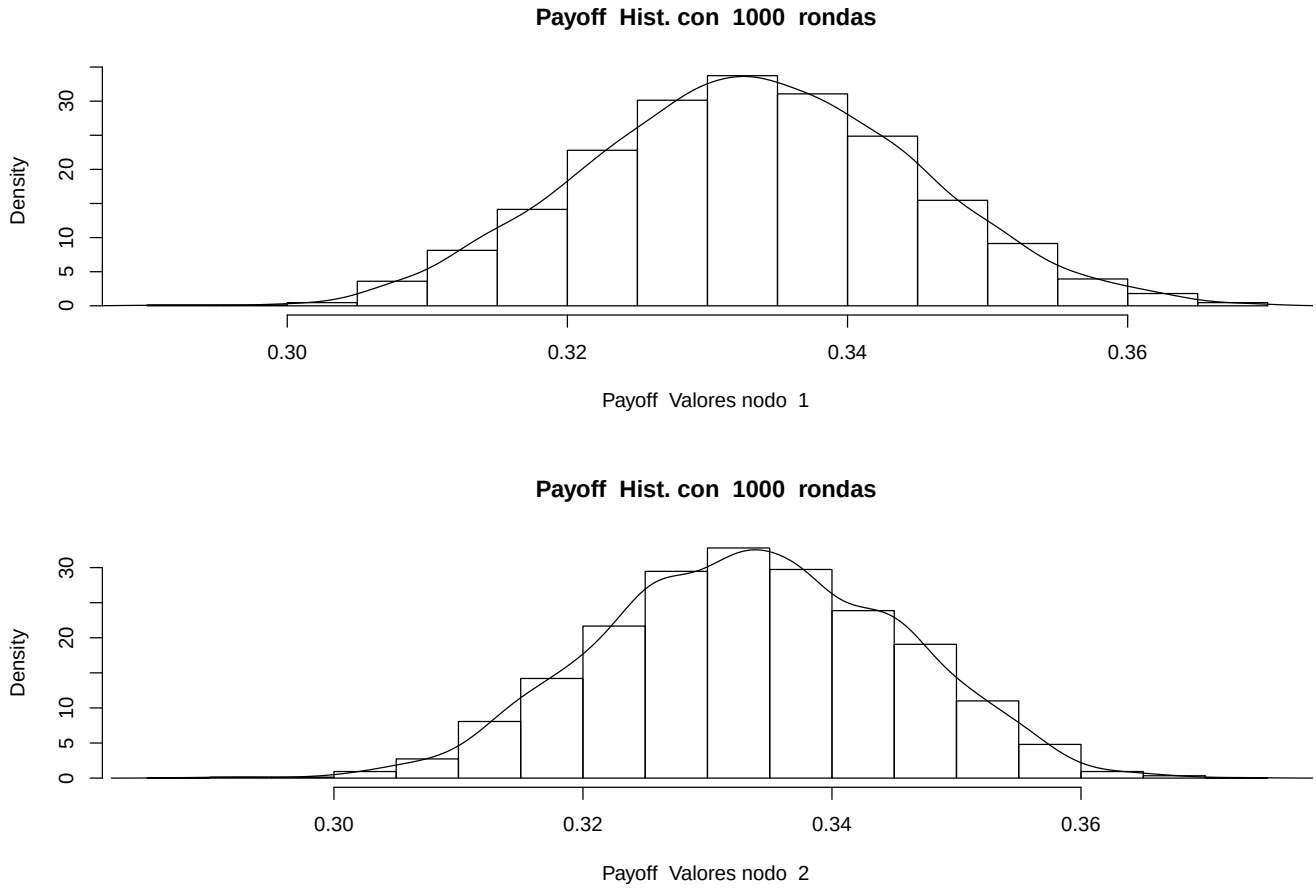

Figura 4.12: Dos jugadores honestos sin control (1.000 rondas).

de ellos está siguiendo una $Beta(1,0{,}9)$. En estas gráficas, el sistema no aplica el test de KS y se puede observar claramente como el jugador honesto sale perdiendo.

Si el número de rondas se aumenta, aunque manteniendo la misma longitud del histórico, el beneficio del honesto en media se mantiene por debajo de lo esperable si hubiera jugado contra uno honesto. Dado que el número de muestras es relativamente bajo, el mentiroso podría incluso aumentar el grado de la mentira, y en lugar de seguir una $Beta(1,0{,}9)$, éste podría atreverse incluso con $Beta(1,0{,}7)$. El sistema tendría muchas dificultades para detectar estas mentiras con un valor de histórico tan bajo sin un control adaptativo. Claramente el mentiroso hace daño al contrario. Esto se puede

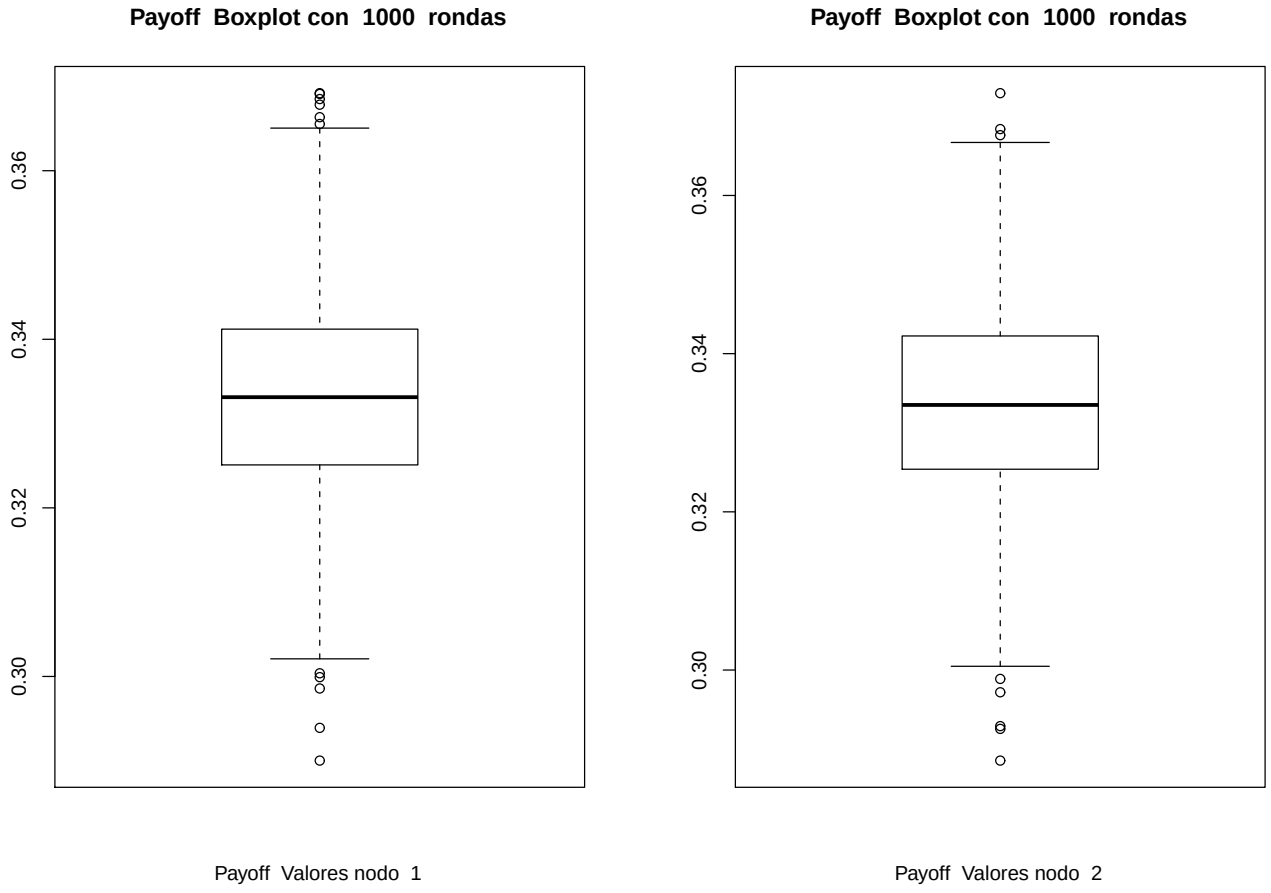

Figura 4.13: Dos jugadores honestos sin control (1.000 rondas).

comprobar en la tabla 4.3.

Las mismas simulaciones se han repetido pero aplicando el test de KS adaptativo propuesto en esta tesis. Para un factor de adaptación igual a 2, los resultados son muy prometedores. Como se puede observar en las gráficas 4.6 y 4.6 el beneficio del jugador honesto no se ve modificado.

De nuevo, es interesante observar el comportamiento del sistema cuando el número de rondas se aumenta, pero se mantiene la misma longitud del histórico. Es muy revelador poder comprobar como el beneficio del honesto apenas sufre una pérdida importante. Esto se puede comprobar en la tabla 4.4, en las que un jugador

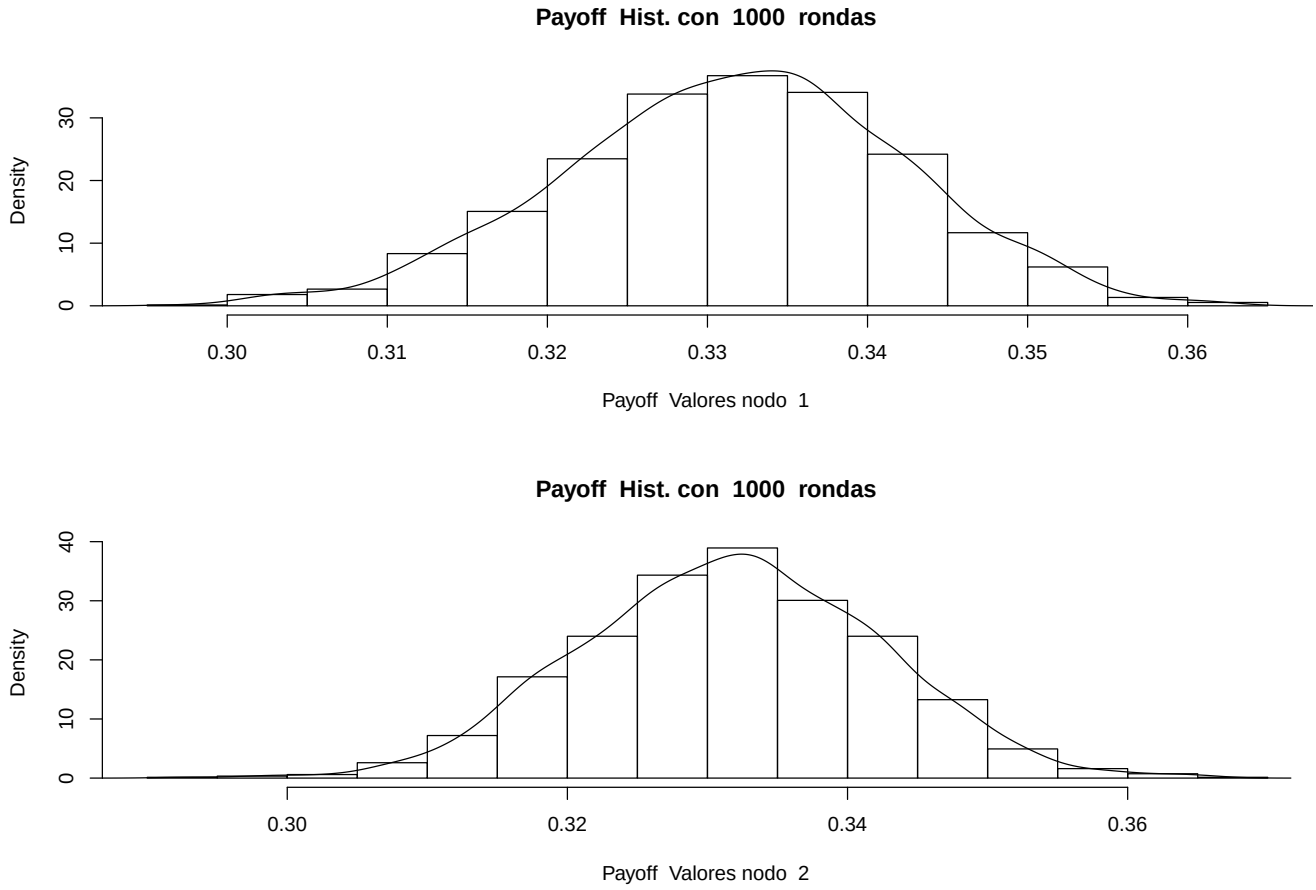

Figura 4.14: Dos jugadores honestos con control (1.000 rondas).

sincero participa con jugadores mentirosos que siguen mentiras de $Beta(1,0{,}9)$ y $Beta(1,0{,}9)$ respectivamente.

A la luz de los resultados simulados, se puede afirmar que el algoritmo QPQ tiene un comportamiento casi óptimo, y que estadísticamente se obtienen resultados muy próximos a los óptimos analizados matemáticamente en las secciones anteriores.

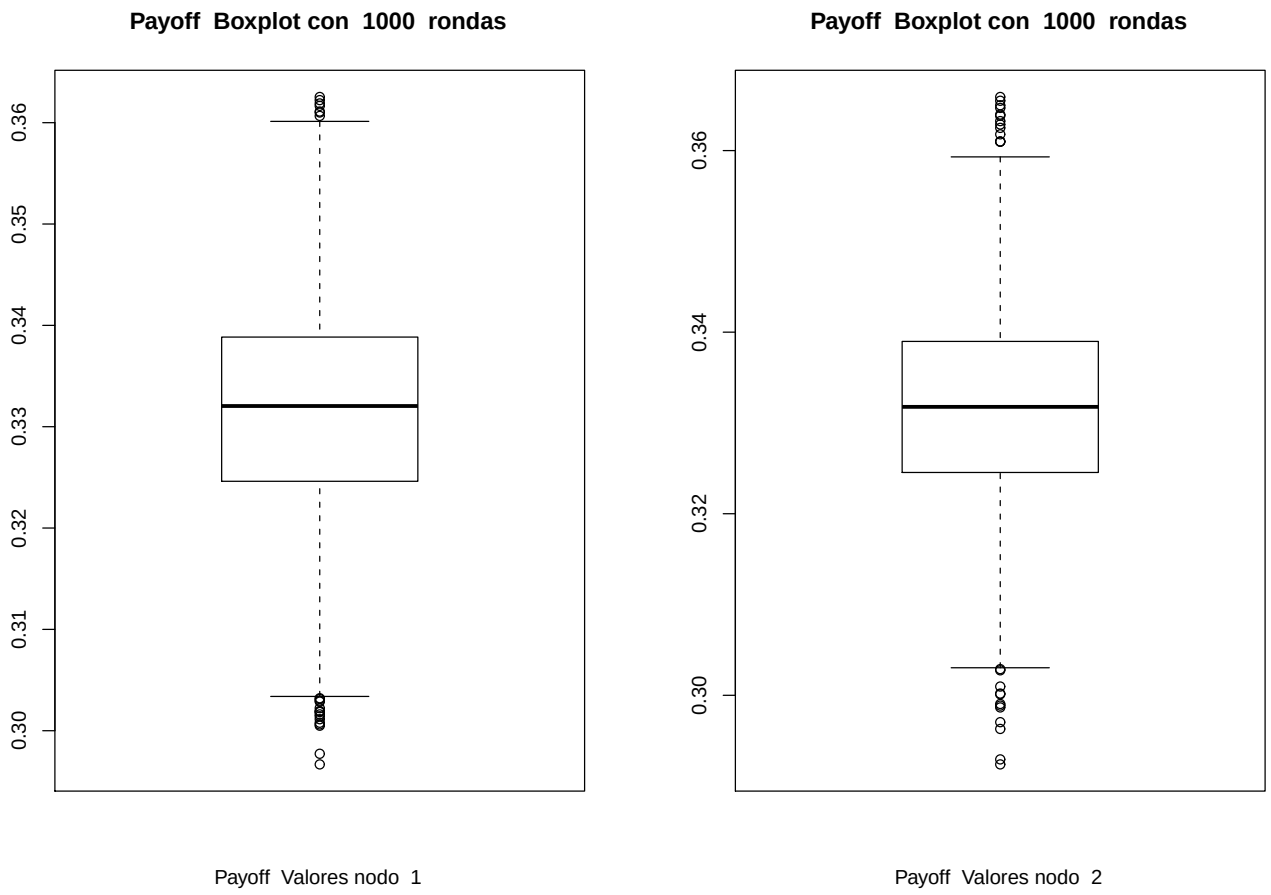

Figura 4.15: Dos jug. honestos con control (1.000 rondas).

n	100 rondas	300 rondas	600 rondas	1000 rondas
1	0.3282888	0.3303351	0.3315638	0.3316756
2	0.3283282	0.3305222	0.3310481	0.3316858

Cuadro 4.2: Medias de coste de 2 jugadores honestos con control

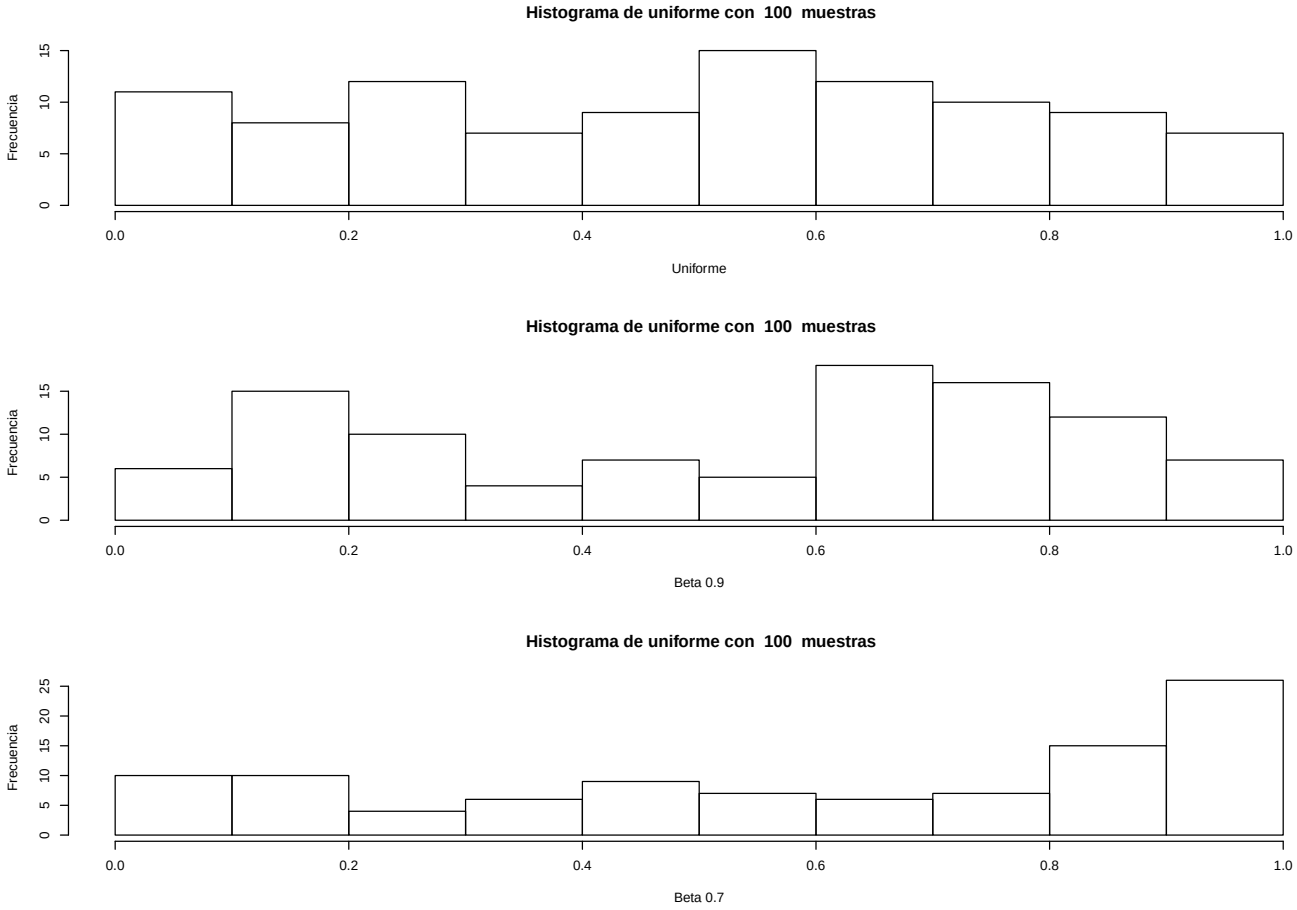

Figura 4.16: Comparativa de tres distribuciones similares (100 muestras).

Contrincante	100 rondas	300 rondas	600 rondas	1000 rondas
Honesto	0.3282888	0.3303351	0.3315638	0.3316756
Mentiroso Beta-9	0.3172515	0.3196885	0.3209979	0.3213058
Mentiroso Beta-7	0.3017784	0.3068946	0.3108102	0.3143198

Cuadro 4.3: Comparativa de utilidad con diferentes contrincantes sin control

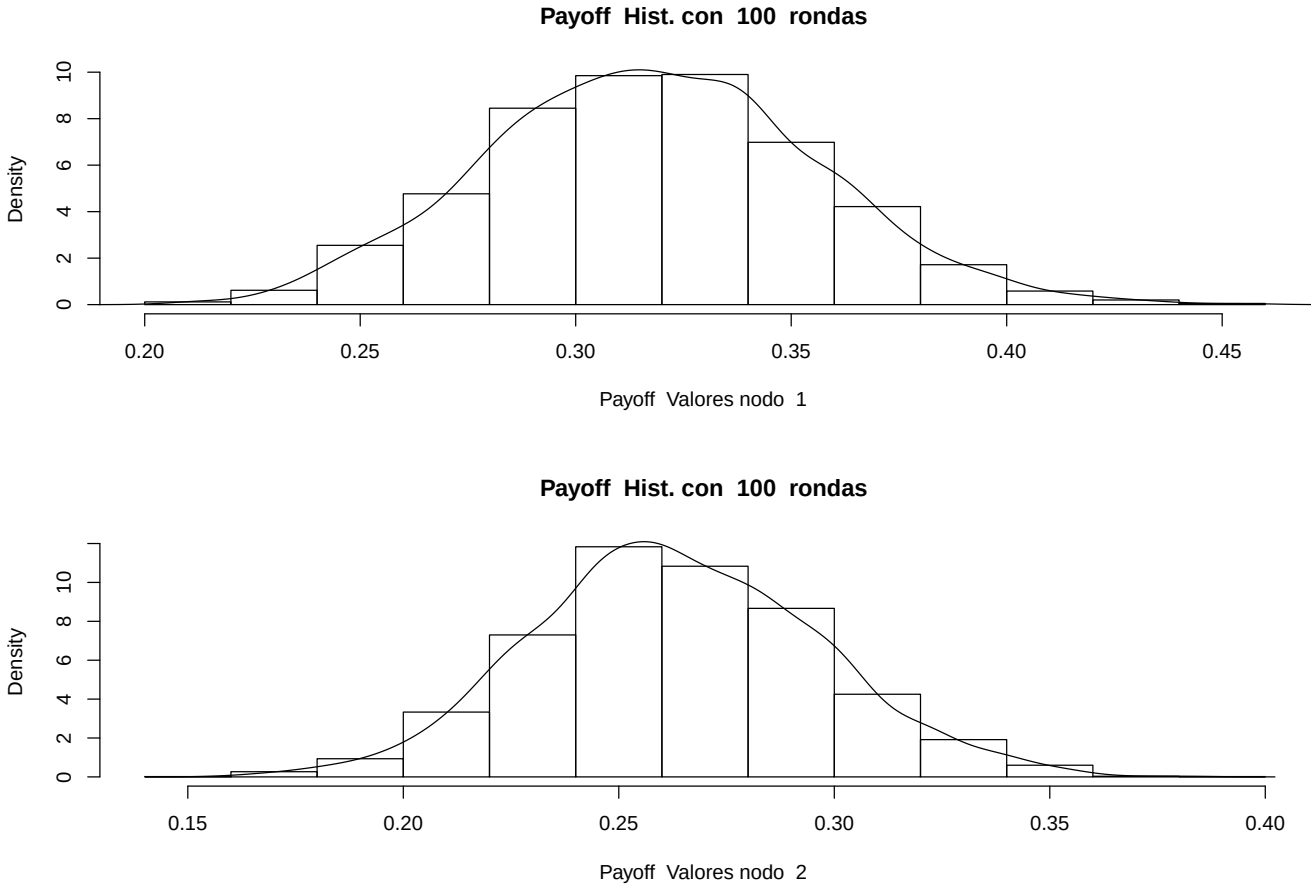

Figura 4.17: Dos jugadores (sincero vs mentiroso Beta-9) sin control I.

Contrincante	100 rondas	300 rondas	600 rondas	1000 rondas
Honesto	0.3326126	0.3334818	0.3326952	0.3335234
Mentiroso Beta-9	0.3176976	0.3180095	0.3186636	0.3187127
Mentiroso Beta-7	0.2815161	0.2818364	0.2817011	0.2817494

Cuadro 4.4: Comparativa de utilidad con diferentes contrincantes con control

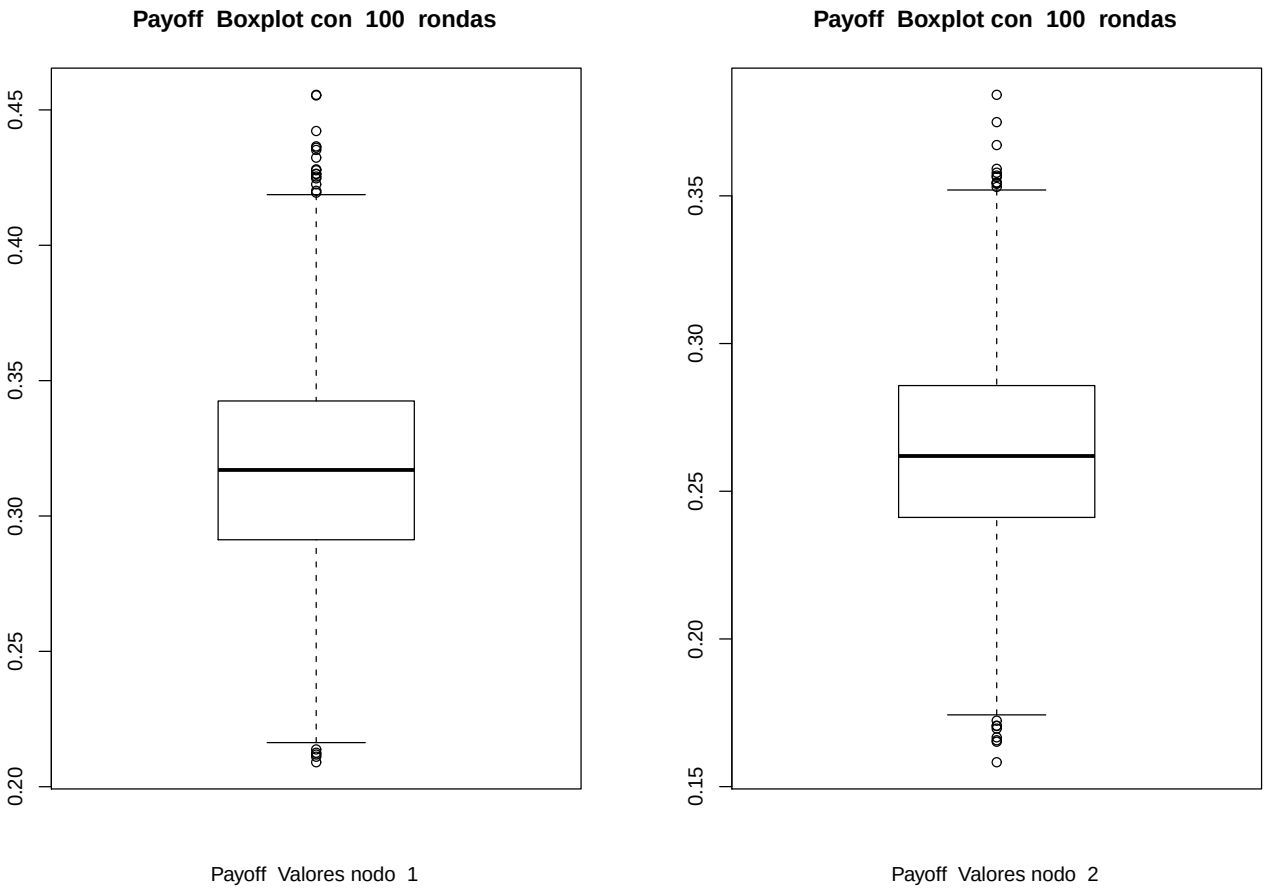

Figura 4.18: Dos jugadores (sincero vs mentiroso Beta-9) sin control II.

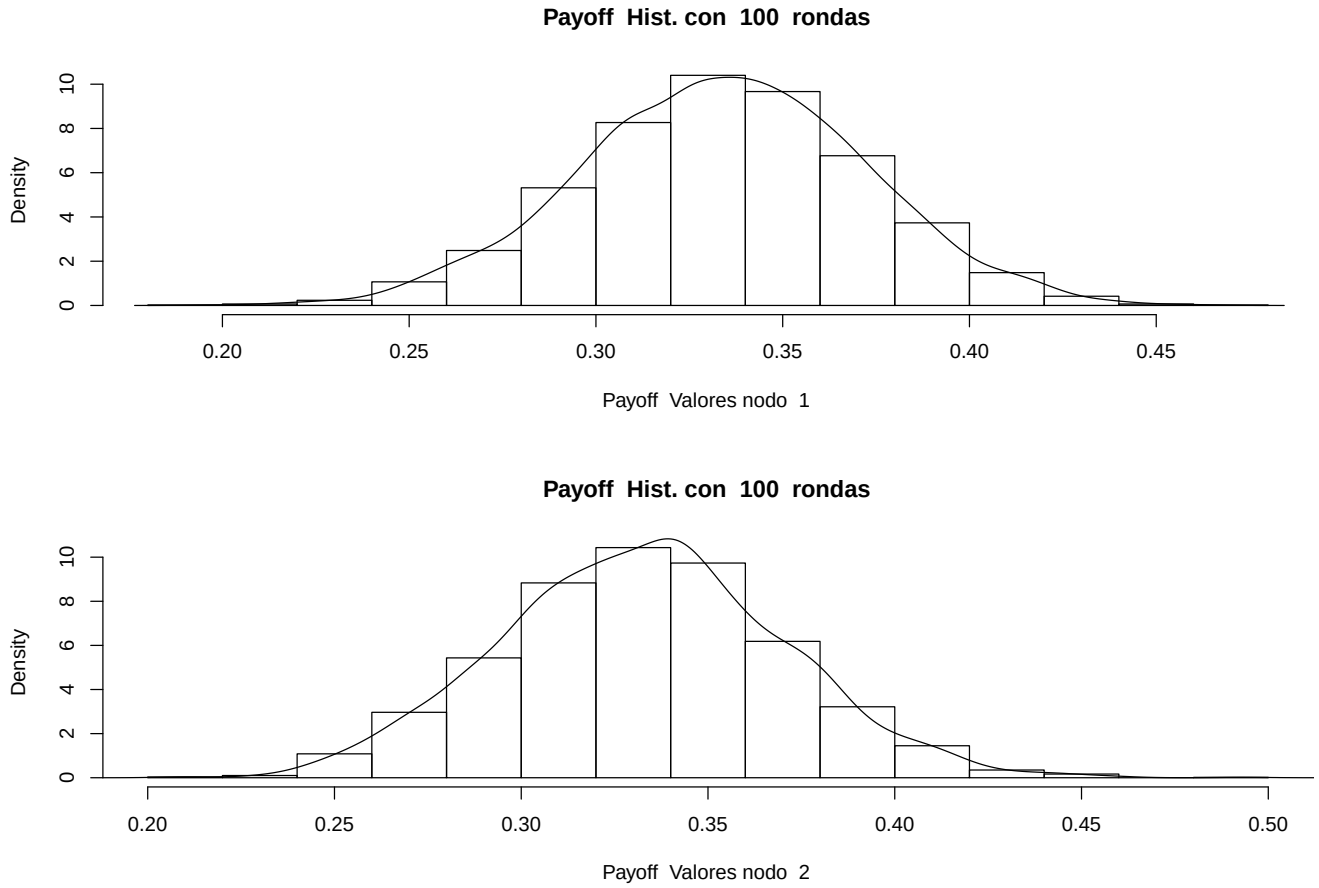

Figura 4.19: Dos jugadores (sincero vs mentiroso Beta-9) con control (100 rondas) I.

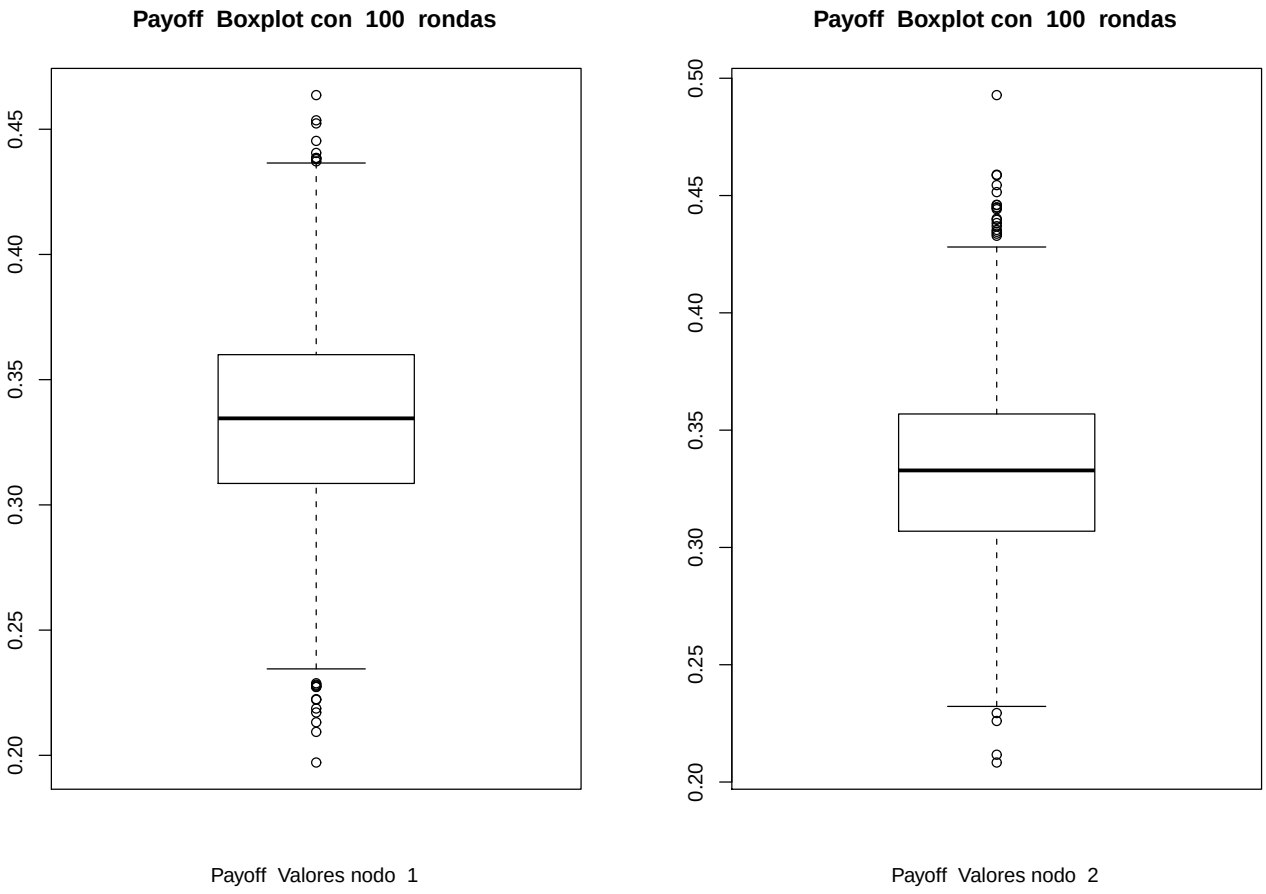

Figura 4.20: Dos jugadores (sincero vs mentiroso Beta-9) con control (100 rondas) II.

Capítulo 5

QPQ - Jugadores Correlados

> "In matters of truth and justice, there is no difference between large and small problems, for issues concerning the treatment of people are all the same."
>
> Albert Einstein

5.1. Introducción

Hasta el momento hemos visto como construir un mecanismo que asigna las tareas a los jugadores de forma óptima y con las propiedades buscadas. Pero hemos hecho una suposición difícil de cumplir en la vida real y seguramente en Internet. Si seguimos con nuestro ejemplo de un piso de estudiantes, y les pedimos que valoren la tarea de limpiar la casa un viernes por la tarde, seguramente nos encontremos que todos los estudiantes estarán pensando en salir con los amigos. ¡Los costes declarados estarán fuertemente correlados y seguramente serán todos muy altos! En Internet es probable que nos encontremos situaciones similares.

Para entender el problema que esto puede generar, supongamos la asignación de tareas a tres jugadores. Dos de ellos están fuertemente correlados (son por ejemplo gemelos) mientras que el tercero es independiente. Si asignamos la tarea siguiendo el

criterio de tomar el menor de los tres valores, lo que nos sucederá es que el jugador independiente hará un mayor volumen de trabajo. Es decir, los jugadores correlados se repartirán los valores más bajos, mientras que el jugador independiente tendrá que ejecutar él solo todas las tareas que a él le parezcan bajas. En el caso más extremo, los jugadores correlados ejecutarán una cuarta parte de las tareas, mientras que el jugador independiente hará la mitad.

Un ejemplo de esta situación se puede observar en las gráficas 5.1 y 5.2. En la primera gráfica se pueden observar los histogramas de los tres jugadores. Aparentemente los tres jugadores están declarando sus valores conforme a lo establecido. Es decir, declaran sus valores según una distribución uniforme. Nuestro mecanismo QPQ original no parece detectar valores conflictivos. La gráfica segunda (5.2) nos muestra que existe una cierta correlación entre los valores declarados por los dos primeros jugadores. En este ejemplo, los dos primeros jugadores mantienen una cierta correlación y están jugando contra un tercer jugador independiente.

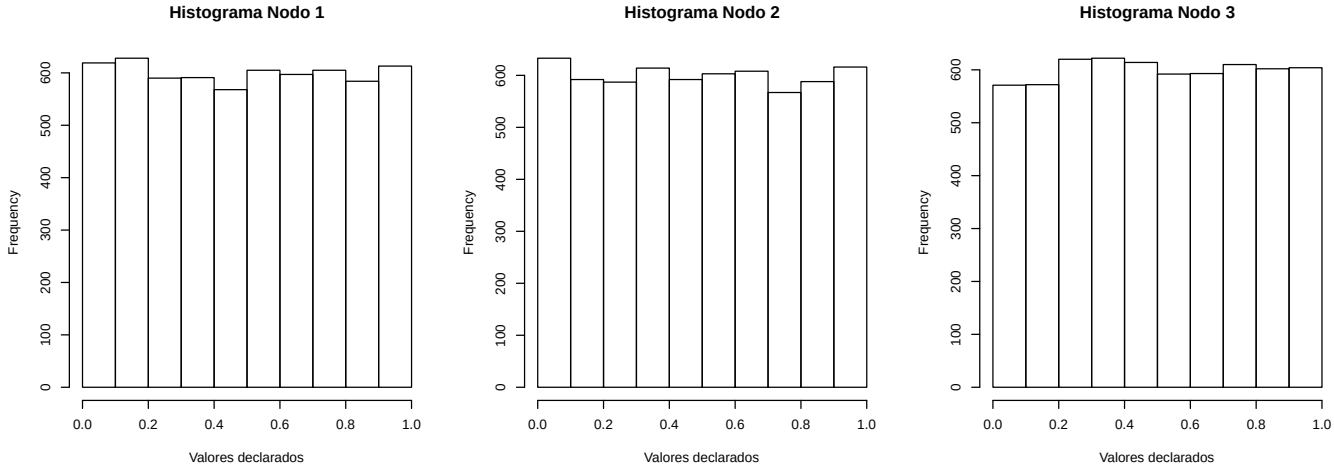

Figura 5.1: **Visión de QPQ Básico**. La versión básica de QPQ no distingue entre los 3 jugadores. Todos los participantes tienen distribuciones uniformes.

Si aplicamos una función de decisión basada en el menor valor estamos pro-

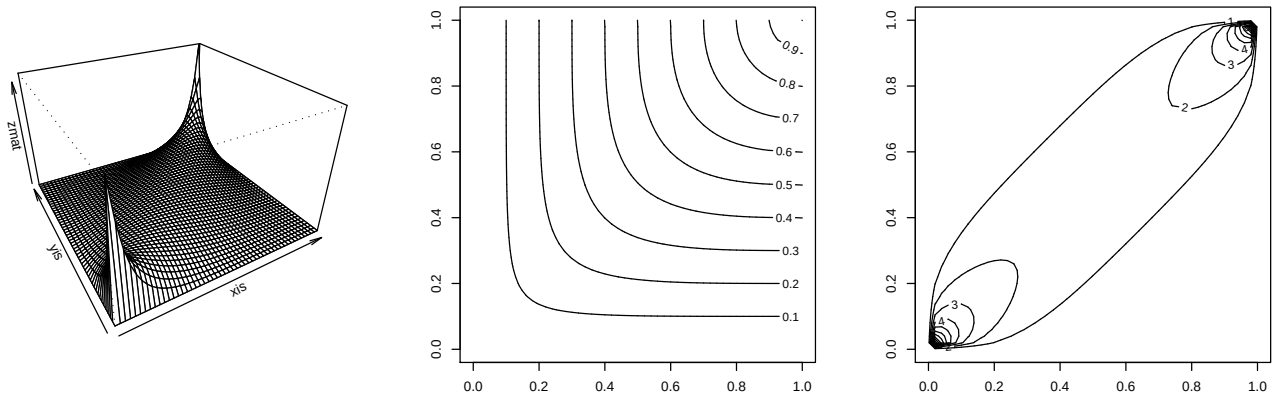

Figura 5.2: Los dos primeros jugadores están correlados.

vocando que uno de los jugadores realice un mayor esfuerzo y un mayor volumen de tareas. Esto se puede observar en la gráfica 5.3 que representa la evolución en el tiempo del beneficio (ahorro de coste) y del número de tareas de cada uno de los tres jugadores. El jugador independiente se ha representado en verde. Claramente, sale perdiendo frente a los otros dos jugadores.

Esta situación se puede invertir si la correlación entre los dos primeros jugadores fuera negativa. Este caso lo representa las gráficas 5.4 y 5.5. Como se puede observar, aquí el ganador es el jugador independiente.

Casos generales de correlación entre dos jugadores se muestran en la gráfica 5.6. En estas gráficas se ha representado con tonos de color la densidad de la función bivariada. Esta respresentación nos puede servir de base para entender que la función de corte de QPQ Básico (la línea de corte $y = x$) no va a dividir el área de juego en dos zonas "justas". También obsérvese que aunque la distribución bivariada tome estas formas, las marginales deben ser uniformes cuando se corresponden al juego de dos jugadores.

En principio, podríamos pensar en utilizar QPQ en su forma básica según lo

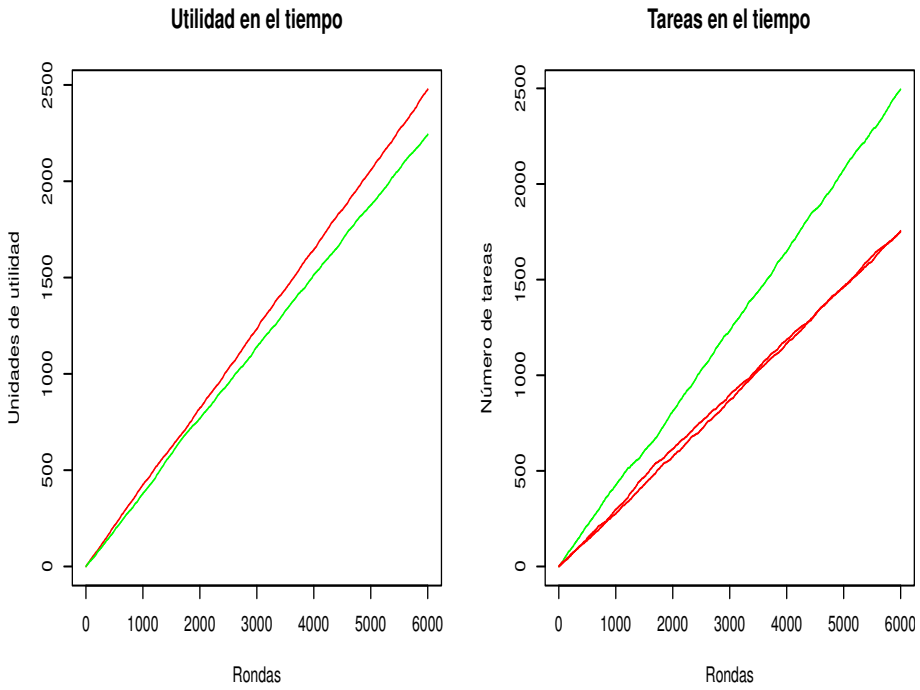

Figura 5.3: Esfuerzo y trabajo de los tres jugadores.

expuesto en el capítulo anterior, pero no podríamos garantizar que el sistema cumpla con uno de los requisitos buscados: **justicia en el reparto**. Basta con revisar muchas de las demostraciones utilizadas para comprobar que el requisito de independencia era la piedra angular del sistema.

Puede ser que esta pérdida de justicia tenga sentido en algunos de los problemas reales en el que se aplique QPQ, pero creemos que sería una limitación seria en otros casos. Por ello, hemos querido avanzar y proponer alternativas en las que se pueda conseguir justicia entre jugadores.

Al abrir la puerta a que los jugadores tengan distribuciones no independientes, damos paso a una dificultad añadida, ya que el espacio de estrategias sobre el que trabaja el mecanismo se amplía. Pero en este punto podemos abrir dos caminos de estudio según consideremos si el jugador puede o no modificar la correlación como parte de su espacio de estrategias. Antes, el espacio de estrategias que tenía el jugador era encontrar una distribución de probabilidad que fuera uniforme. El sistema podía

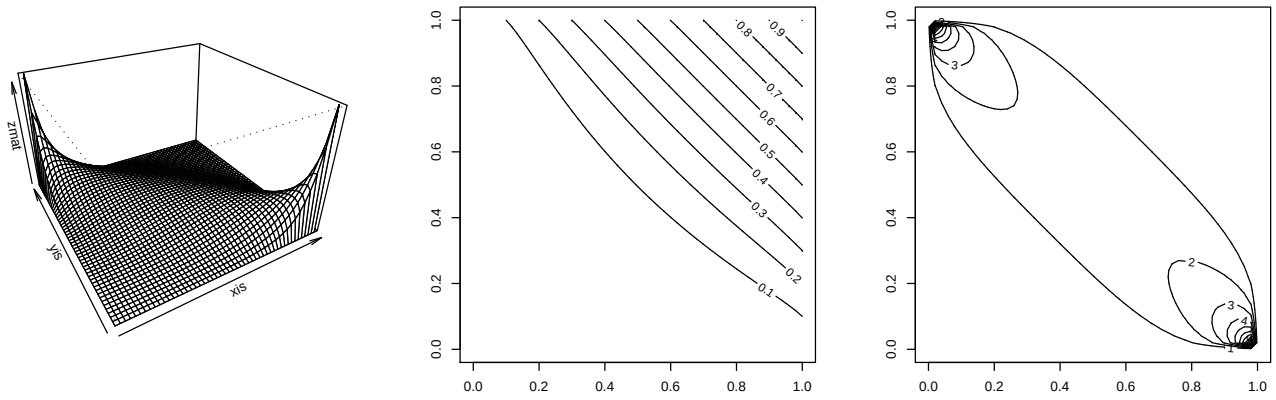

Figura 5.4: Los dos primeros jugadores están correlados (II).

comprobar dicha posibilidad sin excesiva complicación. En esta nueva variante, el jugador tiene un espacio de estrategias más amplio. Ahora puede decidir su propia distribución de probabilidad y hasta que grado desea correlarse con los contrarios.

Por ejemplo, un jugador puede tener una distribución *poco correlada* cuando juega en contra de un contrincante y *fuertemente correlada* cuando juega con otro. Y ser ciertas las dos. Pero nuestro mentiroso puede "simular" que su distribución es una de éstas si con ello puede ganar. Bajo esta perspectiva, el jugador tiene que buscar una estrategia que incluye la posibilidad de mentir usando información o estimaciones estadísticas de los valores que pueda anunciar el contrario. El jugador puede aprender o tener conocimiento de los contrarios y usarlo a su favor. El problema es encontrar la forma de distinguir entre las situaciones verdaderas y la "simulada" por el mentiroso y extraer la información del jugador mediante los adecuados "premios" o "castigos".

Observe que la dificultad para el diseñador de mecanismos estriba en que potencialmente todas las distribuciones reales o falsas de un jugador son plausibles. Y, desgraciadamente para nuestro caso, es difícil suponer que podríamos conocer la correlación entre jugadores. En el trabajo inicial de mecanismos de enlace de Jackson

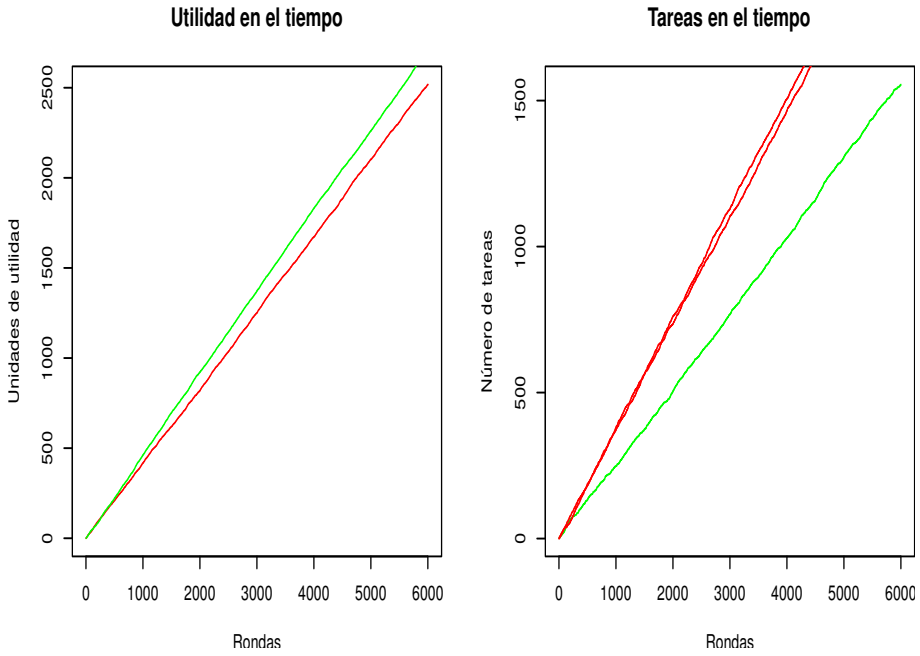

Figura 5.5: Esfuerzo y trabajo de los tres jugadores (II).

y Sonnenschein, se supone que el diseñador de mecanismos conoce de antemano el grado de correlación entre los jugadores y que puede aplicar algún tipo de control de aceptación si los jugadores deciden salirse del marco establecido.

Para abordar el problema de jugadores correlados hemos supuesto dos posibles escenarios, cada uno de ellos genera un posible modelo interesante de estudio. El primero es el escenario en el que los jugadores pueden tener una cierta correlación, pero suponemos que ésta no es conocida a priori y que los jugadores no tienen capacidad para alterarla. El segundo escenario es mucho más complejo, ya que supone que la correlación podría ser una información disponible a uno o varios de los jugadores, y que éstos tienen mecanismos para alterarla si con ello se benefician. Es decir, nos enfrentamos a dos posibles mecanismos o familias de ellos. En ambos casos debemos suponer que **no se conoce la correlación** con antelación al diseño e implantación del mecanismo.

De nuevo, nos gustaría partir del trabajo de Jackson y Sonnenschein, y plantear alternativas que nos permitan construir mecanismos de enlace sin tener que suponer

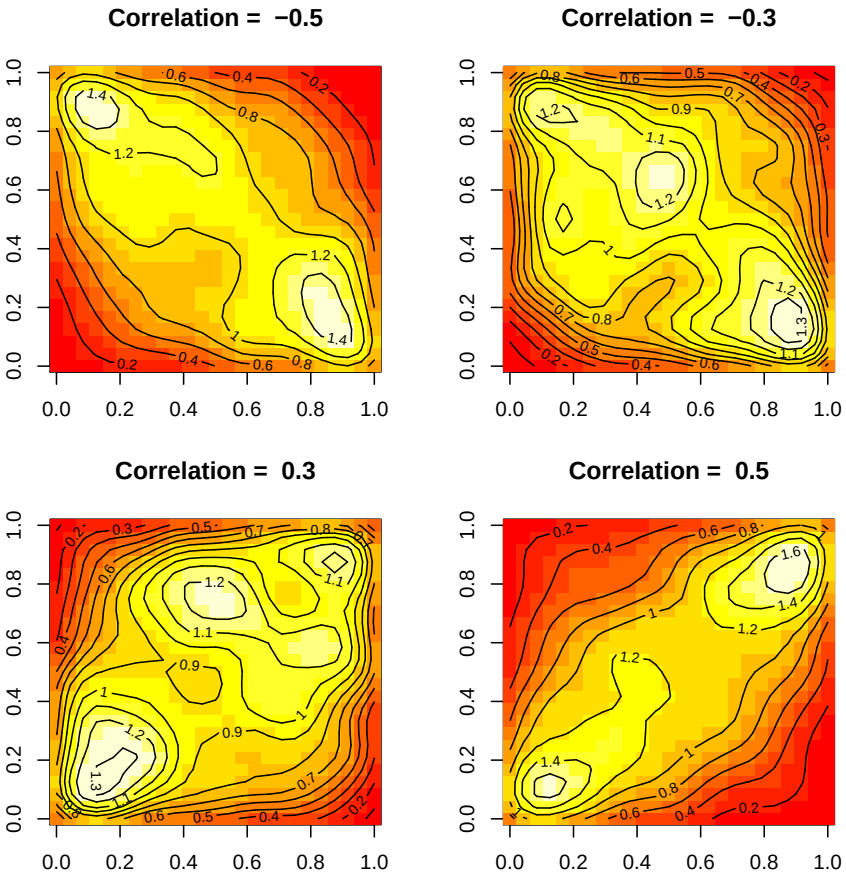

Figura 5.6: Varios ejemplos de correlación entre jugadores.

una correlación conocida por el diseñador de mecanismos. Según nuestro conocimiento, y hasta donde hemos podido comprobar, las variantes de QPQ que se incluyen en este capítulo son innovadoras y aportan múltiples novedades a los mecanismos de enlace.

A partir de este punto, supondremos que tal correlación existe pero que es desconocida por el diseñador de mecanismos. Es este capítulo abordaremos el primer escenario, es decir supondremos que los jugadores no podrán alterar la correlación entre ellos. En su espacio de estrategias no se contemplará la posibilidad de alterar la cópula de las distribuciones conjuntas. Resolveremos el problema de construir un

mecanismo que asigne tareas sin la restricción de que los valores declarados tengan distribuciones independientes, pero con la limitación anteriormente mencionada.

En realidad, esta limitación puede estar justificada en muchos casos. Por ejemplo, si los valores son declarados al inicio del juego. Si antes de iniciar el juego, el jugador debe declarar los valores de todas las tareas en cuestión y sin conocer los valores de los demás jugadores. En este caso, el jugador no podrá afectar la distribución conjunta.

La estructura del presente capítulo es la siguiente. Primero repasaremos algunos de los conceptos ya vistos, pero que tienen una visión más amplia bajo este nuevo planteamiento de distribuciones no independientes. Es por ello que se revisa la forma de la función de decisión y el concepto de justicia. A continuación se plantea una solución al problema que hemos denominado **QPQ Correlados** para distinguirlo del mecanismo presentado anteriormente. Ofrecemos dos variantes de este mecanismo, consecuencia de buscar diferentes nociones de justicia.

Empezamos pues, revisando algunos de los conceptos utilizados en la construcción de nuestro *QPQ Básico*. Conceptos que hemos propuesto y usado sin detenernos en su implicación.

5.2. Sobre la noción de justicia

¿Qué entendemos por un sistema justo de reparto de tareas?. Cuando intentamos responder a esta pregunta tendemos a reflejar vivencias e ideas de la vida que nos rodea. Y seguramente encontremos tantas respuestas como interlocutores. En la preparación de este trabajo me he encontrado con opiniones muy diversas sobre la justicia. ¿Es justo un sistema que reparte el número de tareas de forma que todos los jugadores realizan el mismo número de tareas? ¿Es mejor el sistema que reparte las tareas de forma que todos los jugadores realizan un mismo esfuerzo? ¿O quizás el sistema justo es aquel que ahorra a todos los jugadores una misma cantidad de esfuerzo?

Supongamos de nuevo el ejemplo anterior de tres jugadores, dos de ellos son ge-

melos y por lo tanto se pueden considerar completamente correlados (siempre dicen el mismo valor). Si aplicamos el mecanismo QPQ básico y pedimos las valoraciones a los jugadores, nos encontraremos con tres valores, siendo dos de ellos idénticos. El mecanismo asigna la tarea al valor más bajo y si hay empate lo reparte de forma aleatoria. Eso significa que nuestros gemelos trabajarán una cuarta parte de las veces cada uno (ejecutan una cuarta parte de las tareas) mientras que nuestro jugador solitario hará la mitad del trabajo (medido sobre el número de tareas). Si lo medimos como el esfuerzo normalizado (el coste normalizado -PIT- de cada tarea por la probabilidad de ejecutar esa tarea) resulta que nuestro amigo hará un trabajo de $1/6$ mientras que cada gemelo gastará $1/12$ de su coste. Algo similar sucedería si medimos el número de tareas que ejecuta cada jugador. Los gemelos ejecutarán un porcentaje menor de tareas que el jugador solitario.

Bajo nuestros criterios habituales podríamos decir que el sistema no parece justo, tanto si lo medimos como el número de tareas que realiza cada jugador, como si lo medimos en el coste de ejecutar dichas tareas. Con este ejemplo lo que queremos es abrir la puerta a una discusión importante: existen muchas y muy variadas definiciones de justicia aplicables, todas ellas razonables y posiblemente no aplicables simultáneamente.

Entonces, quizás las preguntas relevantes deberían ser: ¿Existe alguna situación en la no es posible obtener simultáneamente todos esos conceptos de justicia? ¿Qué sucede si no fuera posible? ¿Por cual debemos optar?

En QPQ básico para jugadores independientes no hemos tenido que distinguir entre un tipo u otro de justicia ya que al obtener una, ¡teníamos muchas de las demás en el mismo paquete!. Cuando los jugadores son independientes estas nociones de justicia son equivalentes. Esto no es cierto cuando a los jugadores se les permite tener costes no independientes entre sí.

Quizás el lector recuerde que en los mecanismos VCG era difícil obtener un reparto justo del excedente, y que se abría la puerta a diferentes conceptos de justicia. Siendo, en la mayoría de los casos, imposible obtener todos los conceptos simultáneamente. Nuestro principal objetivo en la preparación de este capítulo ha sido

la de explorar las posibilidades de QPQ, y en general de los mecanismos de enlace, para aportar estas nociones de justicia. Este punto es, por lo tanto, una parte relevante del estudio al que queremos enfrentarnos. Veremos cómo modificar QPQ para poder implementar algunas de las nociones de justicia mencionadas. También veremos que en general, y si queremos eficiencia del sistema, no podremos conseguir todas las nociones a la vez. Si conseguimos un tipo de justicia, perderemos otra.

Es posible que el lector se sorprenda que conceptos tan básicos de nuestra vida real, sean tan diferentes y muchas veces contradictorios si los estudiamos con la lupa de las matemáticas. Pero, realmente no debe sorprendernos, ya que es conocido en la teoría de juegos que no existe una noción única de justicia. Generalmente, se entiende por justo ("fair") que el jugador no se arrepiente del juego ("regret"). De forma informal, este concepto lo que mide es hasta que punto un jugador puede preferir otro sistema de juego que le mejore su utilidad.

En la nueva versión de QPQ distinguiremos entre diferentes nociones de justicia y veremos que implica cada una de las siguientes alternativas:

1. El mecanismo asigna tareas de forma que la esperanza en el número de tareas ejecutadas sea el mismo para todos los jugadores. Es decir, todos los jugadores ejecutarán una proporción de $1/n$ tareas. Dado que cada jugador tiene una función de distribución diferente, cada jugador realizará un esfuerzo diferente, incluso si el número de tareas sea el mismo.

2. El mecanismo asigna tareas de forma que la esperanza en el esfuerzo para la ejecución de las tareas sea la misma para todos los jugadores. Es decir, todos los jugadores tendrán un coste o esfuerzo similar, aunque el número de tareas sea diferente.

Claramente hemos dejado atrás muchas otras posibilidades de justicia que podrían ser relevantes en ciertos contextos. Hemos considerado que con estos ejemplos se cubre un amplio espectro de situaciones y sobre todo, esperamos que las herramientas matemáticas utilizadas nos puedan ser de utilidad para otras nociones de justicia si fueran necesarias en el futuro.

5.3. Sobre la función de decisión

Antes de continuar, quisiéramos aclarar que, aunque nuestro objetivo sigue siendo la interacción de varios jugadores, a lo largo del presente capítulo trabajaremos con un juego de dos jugadores. El lector recordará que, por la forma de calcular la función de decisión, el sistema siempre se puede ver como un juego formado por un jugador contra el sistema (agregación de los demás jugadores). Para ilustrar gráficamente y simplificar muchos de los conceptos presentados en este capítulo se ha considerado útil tan suposición. En este sentido, utilizaremos las variables x e y indistintamente de c_1 y c_2 o de c_x y c_y, representando en todos estos casos las variables aleatorias del coste de los jugadores. Sin perder generalidad, hablaremos del jugador 1 o i como el jugador en cuestión y del jugador 2 o j como el sistema agregado. Esperamos que con este pequeño cambio de notación, todos los cálculos matemáticos (principalmente integrales y funcionales) se vean simplificados. Por otra parte, y en un primer momento, nuestro estudio se centrará en la noción de justicia entendida como reparto equitativo de tareas.

En la construcción de los mecanismos de enlace, el diseñador del mismo debe centrar su atención principalmente en la función de decisión. Matemáticamente, el diseñador de QPQ se enfrenta a resolver un problema de optimización cuyo planteamiento matemático podría ser similar al siguiente:

$$\min_{d\in D} \left[\int_0^1 x \int_0^1 f(x,y) Pr[d(x,y)=1] dy\, dx + \int_0^1 y \int_0^1 f(x,y) Pr[d(x,y)=2] dx\, dy\right]$$

sujeto a condiciones adicionales, por ejemplo,

$$\int_0^1 \int_0^1 f(x,y) Pr[d(x,y)=1] dy\, dx = \tfrac{1}{n} \tag{5.3.1}$$

Lo que esta expresión quiere representar es el hecho de que el diseñador de mecanismos está buscando optimizar (hacer eficiente) la asignación de tareas. Ahora el concepto de eficiencia es más restrictivo que en su versión básica de QPQ, ya que impondremos alguna ligadura (restricción) al sistema. Estas restricciones harán posible que el reparto respete alguna de las nociones de justicias estudiadas. En el ejemplo,

representado por el sistema de ecuaciones anteriores (5.3.1), la restricción impuesta garantiza que los jugadores realizarán un número de tareas proporcional. Evidentemente, dependiendo de la justicia que se quiera imponer, se utilizarán diferentes ligaduras.

Dado que el incentivo a decir la verdad proviene del hecho de que se obliga a respetar una determinada distribución de valores declarados, el diseñador de mecanismos se enfrenta al problema de resolver un sistema de ecuaciones similar al expresado anteriormente. Obsérvese que la solución a este sistema de ecuaciones a optimizar es una función de decisión d.

Sin más información sobre la naturaleza de la función d es imposible resolver tal sistema de ecuaciones en su forma más general. Nuestro recorrido empezará por tanto, estudiando y definiendo la naturaleza de la función de decisión.

El sistema de ecuaciones anteriores (5.3.1) han supuesto una construcción muy general de la función d, ya que tiene la forma de una función aleatoria. Aunque la intuición parezca indicar que la función de decisión puede seguir siendo una función determinista, creemos que es necesario reflexionar sobre la naturaleza aleatoria o determinista de nuestra futura función de decisión.

Pero nuestra reflexión deberá llevarnos más lejos. Incluso suponiendo que la función de decisión sea determinista, siguen abiertas muchas incógnitas. No conocemos nada sobre las propiedades del conjunto de tareas que de forma determinista son asignados a un jugador concreto. Es de esperar que la forma, tamaño y estructura de dicho conjunto dependa de la función conjunta de probabilidad de los costes que tienen los jugadores. Pero, por ejemplo, desconocemos si las tareas asignadas al jugador son puntos aislados de forma más o menos caótica. Quizás formen conjuntos conectados o se repartan en varias "islas". Y sobre todo, lo que más útil nos será es determinar si existe una frontera entre los conjuntos de dos jugadores que podamos utilizar para nuestros propósitos. Si existe una frontera que pueda ser definida por una línea o función, el problema de encontrar el nuevo QPQ tendría una naturaleza claramente definida.

En el capítulo anterior hemos supuesto que la mejor forma de asignar las tareas

a los jugadores se basa en un sistema determinista. Es decir, siempre podremos decir cosas tales como: "si coste asociado a la tarea es mayor que ... entonces ..." o "Asignar la tarea al menor de tal o cual valor". Son decisiones deterministas que utilizan el coste de la tarea para designar al jugador. Aunque podría parecer obvio en la versión más básica de QPQ, podría no ser tan evidente en este nuevo escenario. De hecho, en la forma más básica de QPQ la decisión podría considerarse como no determinista ya que el sistema genera un valor aleatorio cuando dos o más jugadores declaran valores idénticos. Realmente esa situación tiene una probabilidad nula de ocurrir. También se podría pensar que el mecanismo es aleatorio ya que asigna valores aleatorios para el jugador que parece estar mintiendo. Cuando todos los jugadores declaran sus costes siguiendo una distribución uniforme, el sistema utiliza un sistema determinista para la asignación de las tareas (el menor valor declarado), pero cuando uno de los jugadores intenta manipular el sistema, el mecanismo decide de forma aleatoria. Decidir de forma aleatoria en *QPQ Básico* es una forma de penalizar al jugador. Al provocar una decisión no determinista, el sistema toma decisiones no óptimas y eso, junto a la independencia de los jugadores, hace que el jugador mentiroso pierda utilidad.

Lo que queremos en este apartado es plantear la cuestión de si debemos limitar nuestra búsqueda de mecanismos a aquellos que tengan una función de decisión determinista. Y, según mi opinión, esta pregunta tiene bastante sentido cuando se permite una cierta correlación entre jugadores, ya que en algunos casos extremos de correlación parece no existir una función determinista. Por ejemplo, cuando todos los jugadores dicen siempre el mismo valor (aunque todos siguen una uniforme, todos anuncian el mismo valor). Este caso no podría suceder en la versión anterior de QPQ dado que se había supuesto que los jugadores tenían distribuciones independientes. Si insistimos en utilizar la misma función de decisión en este caso, nos encontraríamos que todas las tareas generan un empate y el sistema tendría que resolver la decisión basándose en asignar tareas de forma aleatoria.

Cuando tratamos con jugadores que no respetan la independencia de costes, las soluciones implementables podrían ser de una naturaleza muy diferente a lo que hemos tratado hasta la fecha. Recordemos que cuando diseñamos un mecanismo bus-

camos que las soluciones puedan ser obtenidas bajo algún criterio de equilibrio. Esto es, el diseñador del mecanismo busca la forma de que los jugadores siguiendo las reglas del juego terminen en un punto de equilibrio que sea del interés del diseñador. Y en nuestro caso, lo que buscamos es una solución en la que decir la verdad sea una solución óptima independientemente de lo declarado por los demás jugadores (strategy-proof).

El lector podría preguntarse en qué nos afecta este último comentario cuando hablamos de la función de decisión. La razón de ello vuelve a ser que el sistema debe trabajar con un espacio de tipos más amplio. Supongamos que hemos encontrado un mecanismo que sea óptimo bajo un determinado equilibrio. Supongamos que este mecanismo tiene una función de decisión determinista y que toma como parámetros cualquier valor declarado por los jugadores, incluyendo las posibles mentiras. Si ese sistema existiera y fuera viable, siempre podríamos construir otro que respete las decisiones del original pero tomando los valores ciertos como referencia. Ese nuevo mecanismo seguiría siendo óptimo bajo el mismo punto de equilibrio, pero al hacer el cambio lo que tendríamos sería una nueva función de decisión que podría ser llegar a ser aleatoria.

Como veremos en breve, el espacio de estrategias del jugador es el conjunto de funciones de distribución que liguen tres variables aleatorias: la que representa su mentira, el valor cierto y el valor del contrario. Si transformamos una función de decisión que toma valores falsos para construir una que toma valores ciertos, la nueva función de decisión incluirá todas las funciones de distribución utilizadas por los jugadores en sus estrategias. Lo que hemos mencionado no deja de una aplicación de la revelación directa que ya se ha comentado en otros capítulos.

Antes de pasar al siguiente apartado, hagamos una breve recopilación de lo que debemos estudiar de la función de decisión. Por un lado, tendremos que estudiar si la futura función de decisión podría ser aleatoria o determinista. Hemos visto que aunque la intuición o los ejemplos anteriores indiquen que es determinista, hay razones para estudiar la alternativa aleatoria. Por otra parte, es importante estudiar la estructura del conjunto de tareas asignadas a un jugador.

5.4. Primeros aspectos formales

Volvamos por un segundo a nuestro *QPQ básico*, y observemos cómo la función de decisión divide el espacio de valores declarados por dos jugadores.

Recordemos que la función de decisión de QPQ para dos jugadores era una función $d : [0, 1]x[0, 1] \rightarrow \{1, 2\}$ definida según:

$$\mathrm{d}(c_i, c_j) = \begin{cases} i & c_i < c_j \\ j & c_i > c_j \\ random & c_i = c_j \end{cases}$$

Es decir, asigna la tarea al jugador que tenga el menor coste y aleatoriamente si hubiera un empate. Dado que la probabilidad de que los dos jugadores independientes declaren valores idénticos es nula, la proporción del reparto aleatorio es indiferente a efectos prácticos. El dominio de la función de decisión es el cuadro de valores reales $[0, 1]x[0, 1]$. Recordar que esto es así debido a que cada jugador debe declarar valores según una distribución uniforme con soporte en $[0, 1]$, por ser el resultado de aplicar la transformada PIT sobre sus valores reales. Por lo tanto, la función de decisión en la versión básica de QPQ divide el cuadro según la gráfica 5.7.

La gráfica representa en color blanco el conjunto de puntos que la función de decisión asigna al jugador 1 y en tono azul las correspondientes al jugador 2. Podemos observar que son dos áreas separadas por una línea y que, salvo en la línea de separación, no existe ninguna decisión aleatoria.

Como hemos visto anteriormente, en la nueva versión deseamos encontrar una función de decisión social que nos permita asignar trabajos de forma eficiente y que, además, se ajuste a ciertas restricciones tales como que todos los jugadores ejecuten el mismo número de tareas. La función en su forma más general es una función que asigna una probabilidad a cada decisión por cada punto (c_i, c_j). Lo que esto significa es que cuando los dos jugadores declaran los valores de (c_i, c_j) podrá optar por asignar la tarea de forma aleatoria a uno u otro jugador. La probabilidad de asignar la tarea puede variar según algún parámetro o condicionante del algoritmo. Pero como vamos a demostrar para nuestro caso, y como quizás nos dicte nuestra intuición, el

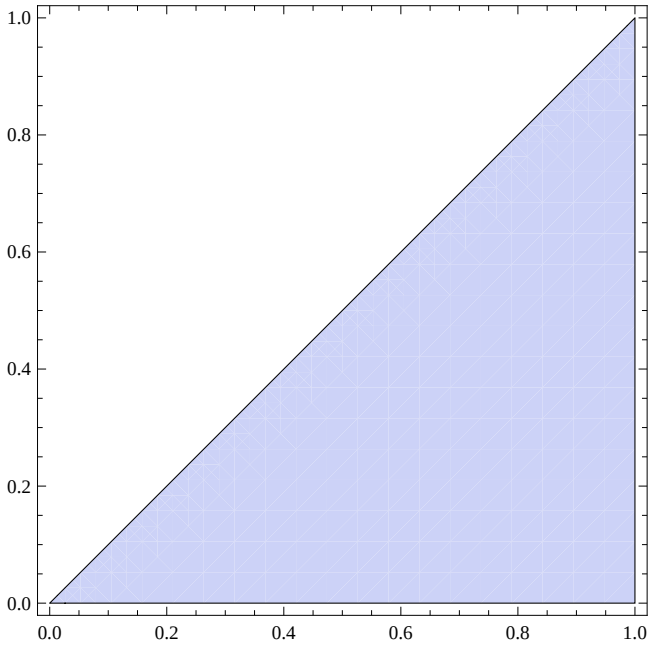

Figura 5.7: Función de decisión en QPQ básico.

sistema consigue optimizar la asignación de tareas cuando la función de decisión es determinista.

Si Φ representa el conjunto de todas las probabilidades definidas sobre $\{1, 2\}$ (también representados por la notación genérica $\{i, j\}$), la función de decisión en su forma más general sería: $d_r : [0, 1]x[0, 1] \rightarrow \Phi$.
O bien $d_r : C \rightarrow \Phi$, siendo C el conjunto $[0, 1]x[0, 1]$. En el caso de *QPQ básico* es la función de decisión se puede reescribir como:

$$\text{d}(c_i, c_j) = \begin{cases} Pr[d = i] = 1 & c_i < c_j \\ Pr[d = j] = 1 & c_i > c_j \\ Pr[d = i] = random & c_i = c_j \end{cases}$$

El primer paso de nuestro recorrido será observar que la asignación de tareas en *QPQ básico* se hace de forma determinista a.e. (“almost everywere”). Basta con observar que $F(\cdot, \cdot)$ asigna probabilidad nula al subconjunto de puntos en $C =$

$[0,1]x[0,1]$ que tienen decisiones aleatorias. Es decir, la medida o probabilidad de que los jugadores anuncien valores en dicho conjunto es cero. El resto de los puntos tienen asociada una decisión determinista (la distribución de probabilidad es $Pr[d=1]=1$ o $Pr[d=2]=1$. Por lo tanto el conjunto de puntos con decisiones deterministas es un conjunto no vacío y con medida diferente a cero.

La pregunta sería si esta propiedad se conserva en la siguiente versión de QPQ. Es decir, ¿Es posible construir una función determinista a.e para nuestro QPQ con justicia en el volumen de tareas? ¿Es posible para otras nociones de justicia?

En general, creemos que la respuesta a estas preguntas depende de las restricciones que se impongan al sistema 5.3.1. Una de las aportaciones de nuestro trabajo es la conclusión de que, cuando la restricción o ligadura del sistema es un reparto equitativo de tareas, entonces existe una función de decisión determinista a.e. Es decir, la decisión es determinista salvo en algún subconjunto de valores con medida nula. En el caso de QPQ con reparto de tareas equitativo ese subconjunto podría llegar a ser un punto del cuadro $C=[0,1]x[0,1]$.

Lo que estamos afirmando es que para cualquier función de probabilidad bivariada $F(\cdot,\cdot)$, si existe una función de decisión d_r que hace óptima la utilidad del sistema y reparte equitativamente las tareas a los jugadores, entonces existe una función de decisión d determinista a.e ("almost everyware") con las mismas propiedades (óptima y equitativa). Además, la función de decisión divide el cuadro $C=[0,1]x[0,1]$ en dos subconjuntos conexos con la topología habitual. Estos puntos son relevantes ya que todo esto implica que la función de decisión se podrá representar mediante una función de corte, como se verá en breve.

Aunque la demostración formal se ofrece más adelante, queremos ofrecer una breve intuición del razonamiento. Supongamos un subconjunto no vacío en C en el que d_r no sea determinista. Siempre se puede reordenar la asignación de tareas y construir una d nueva, haciendo que cada jugador tenga asignadas las tareas de forma determinista. Además, esa nueva función sería más eficiente que la anterior si en la reasignación damos más probabilidad a los puntos más eficientes (los más bajos o los más altos según sea el caso).

Denotamos por $Pr[d_r(x,y)=1]$ la probabilidad que d_r asigne la tarea al jugador 1. Sea L_+ el conjunto de puntos (x,y) en los que la función de decisión d_r asigna la tarea al jugador 1 de forma determinista (con probabilidad = 1). Definimos el conjunto L_- de igual forma para el jugador 2. Es decir:

$$L_+ = \{(x,y) \in C | Pr[d_r(x,y)=1] = 1\}$$

y

$$L_- = \{(x,y) \in C | Pr[d_r(x,y)=2] = 1\}$$

Claramente, la intersección de L_+ y L_- es el conjunto vacío. El conjunto de puntos en los que la decisión es aleatoria es por lo tanto: $L_0 = C - (L_+ \cup L_-)$. Lo que queremos probar es que $F(x,y)$ asigna probabilidad cero a dicho conjunto. Es decir, la medida o probabilidad de que los jugadores anuncien valores en dicho conjunto es cero. Supongamos que dicho L_0 existe y tiene medida no nula. En la gráfica 5.8 se muestra un posible ejemplo de tal situación. En esta gráfica se ha representado en color gris las áreas en las que el sistema asigna las tareas de forma aleatoria, esto es el conjunto que hemos denominado L_0. En ese caso, podríamos construir una nueva asignación en L_0 de la siguiente forma.

Sea t_+ la esperanza del número de tareas que deberá ejecutar el jugador 1 en el área L_0 y sea t_- el correspondiente al jugador 2. Ambas cantidades deben ser distintas de cero ya que hemos supuesto que había una asignación aleatoria. Ahora dividamos a L_0 en dos áreas de tal forma que cada uno de los subconjuntos creados tenga t_+ y t_- de medida respectivamente. Esta construcción siempre será posible. De hecho se podría dividir utilizando una recta $y = x + b$ y hacer que todos los puntos por encima de esa recta se asignen a un jugador y los que estén por debajo al otro. Ya que las masas de cada área se puede expresar en función del parámetro b de la recta. Además, por ser el resultado de una integral, una será creciente y la otra decreciente, pero siempre sumando la misma cantidad e igual a $t_+ + t_-$.

Un ejemplo de tal situación se muestra en la gráfica 5.9 en la que el conjunto L_0 se ha dividido en dos. El gráfico muestra las zonas asignadas a cada jugador en colores diferentes.

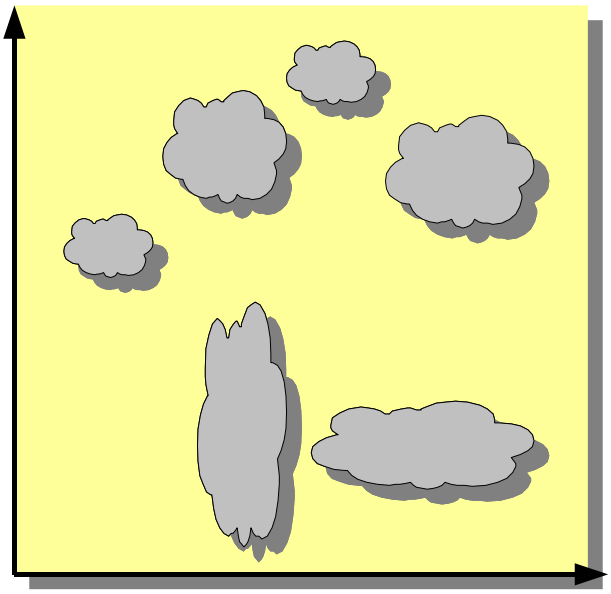

Figura 5.8: Función de decisión con áreas aleatorias.

Las dos áreas creadas de esta forma, respetan la ligadura relativa a la proporción de trabajo y además será más eficiente que la anterior. Basta observar que antes habría cierta cantidad de tareas que eran asignadas al jugador 2 y que estaban por encima de la recta. Igualmente, habría puntos que asignaban las tareas al jugador 1 y que estaban por debajo de la recta. Todas esas tareas se han permutado por asignaciones al jugador más eficiente.

Para continuar con nuestra construcción, ahora queremos hacer notar que las áreas L_+ y L_- forman dos conjuntos conexos del cuadro C bajo la topología habitual. Primero es necesario observar que no hay ninguna área asociada al jugador 2 por encima de la(s) áreas asignadas al jugador 1. De igual forma, no existe ningún área asignada al jugador 1 que esté a la izquierda de la(s) áreas asignadas al jugador 2. Si esto no fuera cierto, siempre podríamos construir una tira en el eje X (respectivamente en el eje Y) y transformar la función de decisión según el procedimiento de intercambiar áreas de trabajo. La forma de transformar la función es muy similar y se llega a la conclusión que si eso fuera posible, entonces la función de decisión no era óptima. Veamos un ejemplo de tal transformación.

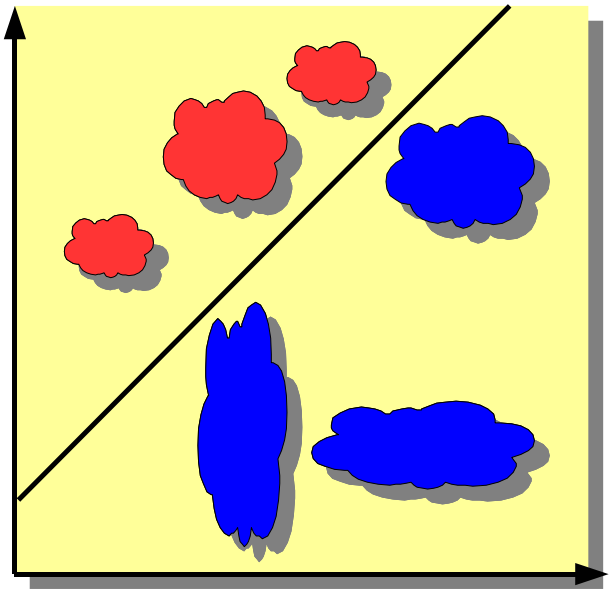

Figura 5.9: Función de decisión sin áreas aleatorias.

En la figura 5.10 se muestra una situación en la que aparecen áreas de asignación no convexas. Hay una zona asignada al jugador 2 (azul) a la izquierda de la correspondiente al jugador 1 (rojo). En esa gráfica se muestra una zona del jugador 1 aislada y por encima del área asignada al jugador 2.

En este ejemplo, siempre se podrá buscar zonas asignadas al contrario, que manteniendo la carga de trabajo, se puedan desplazar hacia el área aislada. En la gráfica 5.11 se muestra el proceso.

Obsérvese que el sistema mantiene la restricción impuesta sobre el número de tareas, pero consigue mejorar la utilidad de sistema. En este ejemplo, el jugador 1 no ha visto modificada su utilidad por el desplazamiento en vertical de la pequeña isla roja, ya que el volumen de tareas asignadas en esa tira vertical no se ha visto modificada. En cambio, el jugador Y ha mejorado su utilidad ya que ahora le tocan tareas más baratas. Un razonamiento similar se puede decir del área azul que se ha desplazado horizontalmente. El jugador 2 no ha visto modificada su utilidad, mientras que el jugador 1 ha mejorado su utilidad. Una posible situación final sería la representada por la gráfica 5.12.

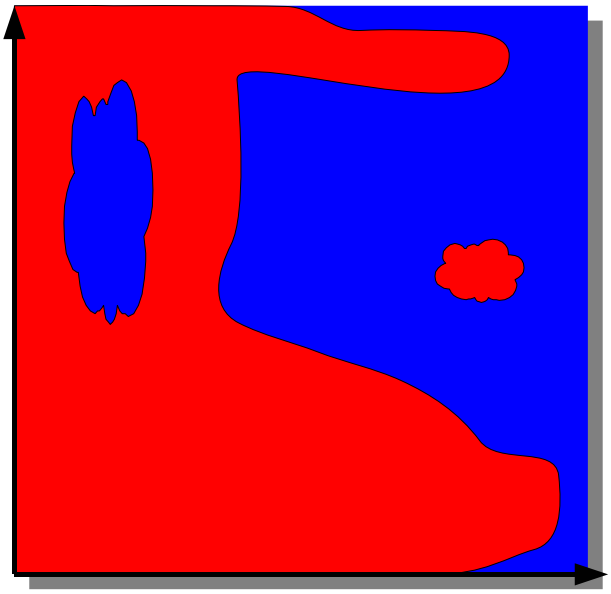

Figura 5.10: Función de decisión con áreas aisladas.

Supongamos que la función de decisión asigna a un jugador las tareas en dos áreas disjuntas y no vacías. Sin perder generalidad, supongamos que es el jugador 1. Por lo visto anteriormente, ambas áreas deben estar pegadas al eje X ya que no puede haber una zona asociada al jugador 2 por debajo. La única posibilidad de que no estuvieran pegadas al eje X es que por debajo de dichas áreas existiera una zona aleatoria. Pero por lo visto anteriormente, esa zona tiene medida nula y por lo tanto no altera ni la utilidad ni el esfuerzo si cambiamos la decisión de aleatoria a determinista para el jugador 1. En resumen, si existen más de dos áreas asociadas al jugador 1, estás deben estar pegadas al eje X. Eso significa que entre las dos áreas debe existir una separación que corresponde a una zona del jugador 2 (si fueran aleatorias, se transforman en jugador 1). Pero eso contradice que no puede haber una zona del jugador 1 a la izquierda del jugador 2. Por lo tanto, no pueden existir dos conjuntos disjuntos y no vacíos para el jugador 1 y en consecuencia, el conjunto es conexo como queríamos demostrar.

Podemos pensar hasta qué punto este razonamiento se podría aplicar al caso en los dos jugadores dicen exactamente el mismo valor (su correlación es máxima).

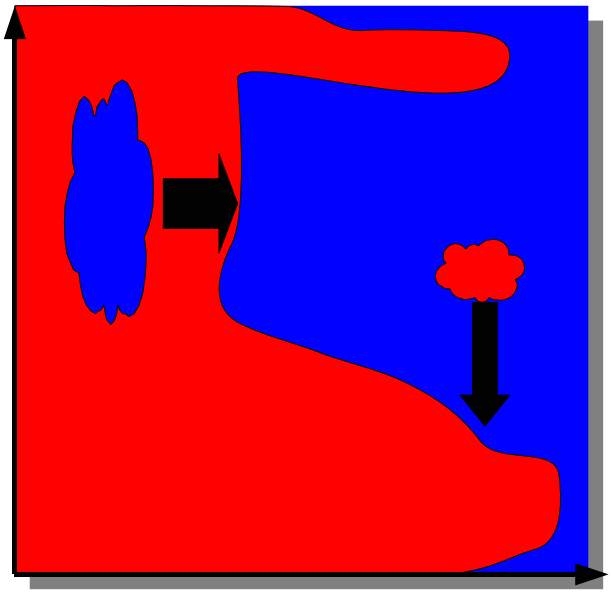

Figura 5.11: Desplazamiento de las áreas.

Pero incluso este caso, se puede transformar en determinista. Basta con hacer que la función de decisión asigne la tarea al jugador 1 en el intervalo $[0, 1/n]$ y el jugador agregado el resto, esto es en $[1/n, 1]$. Únicamente tendría una decisión aleatoria en el punto $(1/n, 1/n)$. Pero en ese caso, este punto tendría medida nula. Esta función sería por lo tanto determinista a.e, sigue siendo óptima y conserva la ligadura.

La conclusión de todo este razonamiento es que no perdemos generalidad si nos centramos en la búsqueda una función de decisión que venga definida por una función que hemos denominado de corte o de transformación:

$$d(x,y) = \begin{cases} 1 & \psi(x) < y \\ 2 & \psi(x) > y \\ random & \psi(x) = y \end{cases}$$

Recordemos que debemos entender esta función de decisión de forma que comparamos el valor de un jugador con el valor modificado del otro y asignamos la tarea al jugador que tenga el menor valor. Hemos especificado “random” cuando el valor de un jugador es idéntico al modificado del otro jugador. En realidad queremos espe-

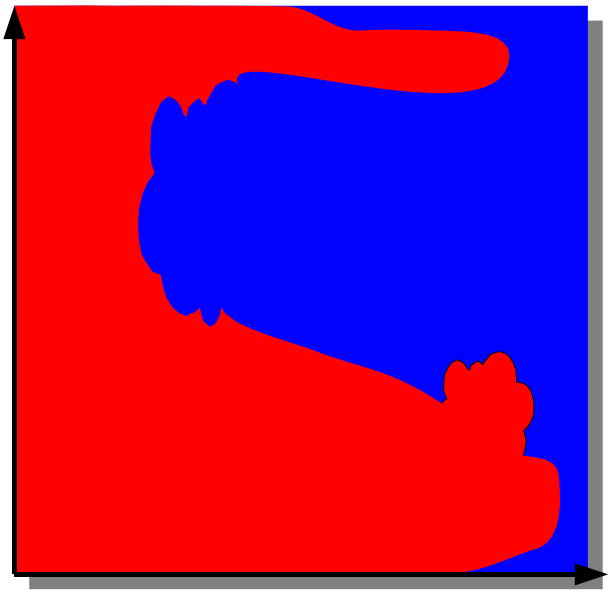

Figura 5.12: Función de decisión con áreas separadas.

cificar una función de asignación determinista o aleatoria según lo comentado en este apartado.

Como se verá más adelante la forma y existencia de esta función de corte viene determinada por la restricción de justicia que se le impone al sistema. Realmente esta restricción es una ligadura del sistema de ecuaciones a optimizar. En el caso de que se desee mantener una proporción de tareas por cada jugador, la ecuación de la ligadura tendrá la forma de un funcional. No está claro qué tipo de funcionales se pueden utilizar en la restricción para que, con carácter general, se pueda asegurar que existe la función de corte.

5.5. Función de corte

Hemos visto que podemos utilizar una función de corte para definir las áreas de asignación de tareas. En la gráfica 5.13 se muestra un posible ejemplo de lo que sería una función de corte arbitraria. No constituye un ejemplo real. La zona marcada en

blanco serían las tareas que le corresponden al jugador 1, mientras que la zona de color azul serían las tareas asignadas al jugador 2.

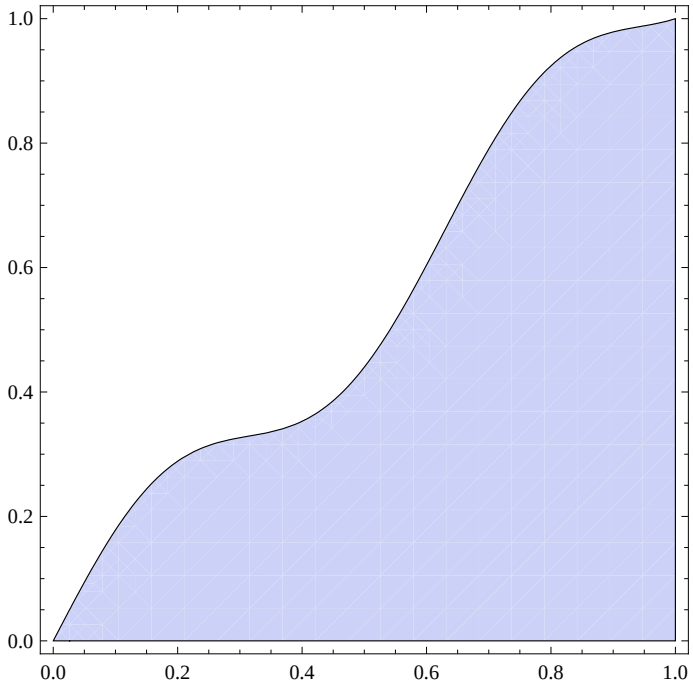

Figura 5.13: Ejemplo no real de función de corte.

En esta sección estudiaremos la forma de dicha función de corte. En general, la función de corte vendrá determinada por la solución a la optimización del sistema y opcionalmente una o varias restricciones o ligaduras. Por ejemplo, si el sistema únicamente busca optimizar el coste de ejecución, la función de corte será la recta $y = \psi(x) = x$. Este ha sido el caso del QPQ básico anteriormente analizado.

En este capítulo vamos a introducir las dos nociones de justicia anteriormente comentadas y estudiaremos cómo afecta a la función de corte. Empezaremos por buscar el concepto de justicia basada en la ejecución de un número de tareas equitativa; es decir, todos los jugadores realizarán en esperanza un número similar de tareas (e igual a $1/n$) y más adelante analizaremos el caso que busca asignar las tareas manteniendo una determinada proporción en el esfuerzo.

Por todo lo visto hasta el momento, suponemos que la función de corte para la

asignación de tareas equitativa es una función monotona creciente, tanto si la observamos desde el punto de vista del jugador 1 como si es observada por el jugador 2. Esto significa que la función ψ tiene inversa ψ^{-1} y podemos utilizar expresiones tales como $x = \psi^{-1}(y)$.

5.6. Caso 1: Justicia en el número de tareas

Este apartado estudia la forma de QPQ para el caso en el que el diseñador de mecanismos desee mantener una igualdad en el número de tareas ejecutadas por cada jugador. Para ello lo que buscamos es optimizar el coste del sistema, pero sujeto a una ligadura relacionada con el número de tareas.

Retomando el sistema de ecuaciones original 5.3.1:

$$\min_{d \in D} [\int_0^1 x \int_0^1 f(x,y) Pr[d(x,y) = 1] dy\, dx + \int_0^1 y \int_0^1 f(x,y) Pr[d(x,y) = 2] dx\, dy]$$

sujeto a condiciones adicionales, por ejemplo,

$$\int_0^1 \int_0^1 f(x,y) Pr[d(x,y) = 1] dy\, dx = \tfrac{1}{n} \tag{5.6.1}$$

pero aplicando nuestro conocimiento sobre la función de decisión, el nuevo sistema sería el siguiente:

$$\min_{\psi} (\int_0^1 x \int_{\psi(x)}^1 f(x,y) dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y) dx\, dy)$$

sujeto a, (5.6.2)

$$\int_0^1 \int_{\psi(x)}^1 f(x,y) dy\, dx = \tfrac{1}{n}$$

Recordemos que estamos buscando como solución una función de decisión d que puede ser expresada mediante otra función, denominada de corte, y que hemos denominado ψ. El sistema anterior se puede resolver utilizando las técnicas matemáticas que nos aporta el cálculo de variaciones.

Si definimos:

$$F_x(x,w) = \int_w^1 f(x,y)dy \tag{5.6.3}$$

$$F_y(y,w) = \int_w^1 f(x,y)dx \tag{5.6.4}$$

La funciones anteriores representan la probabilidad de asignación de una tarea dado el valor de x y el valor de corte z (y de igual forma para la correspondiente al jugador 2).

Con estos valores, el sistema a optimizar (5.6.2) queda:

$$\begin{aligned} &\min_{\psi} \left(\int_0^1 x\ F_x(x,\psi(x))dx + \int_0^1 y\ F_y(y,\psi^{-1}(y))dy\right) \\ &\text{sujeto a,} \\ &\int_0^1\ F_x(x,\psi(x))dx = \tfrac{1}{n} \end{aligned} \tag{5.6.5}$$

Usando un parámetro de Lagrange, este sistema es equivalente a optimizar:

$$\int_0^1 x\ F_x(x,\psi(x))dx + \int_0^1 y\ F_y(y,\psi^{-1}(y))dy + \lambda \int_0^1 F_x(x,\psi(x))dx \tag{5.6.6}$$

La ecuación anterior (5.6.6) define un funcional a ser optimizado con la forma:

$$\begin{aligned} &\int_0^1 x\ I(x,\psi,\psi')dx \\ &\text{donde} \\ &I(x,\psi,\psi') = (x+\lambda)F_x(x,\psi) + \psi F_y(\psi,x)\psi' \end{aligned} \tag{5.6.7}$$

De la ecuación de Euler-Lagrange podemos encontrar una condición necesaria para el extremo. La condición necesaria es similar a lo que sucedía cuando se optimiza una función, igualando una derivada a cero. Para determinar la naturaleza del extremo (mínimo, máximo, punto de inflexión) es necesario realizar la segunda derivada. En el caso de la optimización de los funcionales, no existe un procedimiento tan simple y habitualmente se aplica conocimiento externos al problema (p. e. por características geométricas) para determinar si la solución es o no un mínimo. Como veremos más adelante, en nuestro caso es evidente que nos genera una función que minimiza.

Recuérdese que la ecuación de Euler-Lagrange es:

$$\partial_\psi I(x,\psi,\psi^{'}) - \frac{\mathrm{d}}{\mathrm{d}x}\partial_{\psi'} I(x,\psi,\psi^{'}) = 0 \tag{5.6.8}$$

Utilizando la definición de funcional anterior (5.6.7) y resolviendo cada uno de los términos de la ecuación de Euler-Lagrange (5.6.8) obtenemos:

$$\partial_\psi I(x,\psi,\psi^{'}) = (x+\lambda)\partial_\psi F_x(x,\psi) + \psi^{'} F_y(\psi,x) + \psi^{'}\psi\partial_\psi F_y(\psi,x) \tag{5.6.9}$$

$$\partial_{\psi'} I(x,\psi,\psi^{'}) = \psi F_y(\psi,x) \tag{5.6.10}$$

$$\frac{\mathrm{d}}{\mathrm{d}x}\partial_{\psi'} I(x,\psi,\psi^{'}) = \frac{\mathrm{d}}{\mathrm{d}x}\psi F_y(\psi,x) = \psi^{'} F_y(\psi,x) + \psi\partial_\psi F_x(x,\psi) + \psi^{'}\psi\partial_\psi F_y(\psi,x) \tag{5.6.11}$$

Por lo tanto, la ecuación de Euler-Lagrange queda definitivamente:

$$(x+\lambda)\partial_\psi F_x(x,\psi) + \psi^{'} F_y(\psi,x) + \tag{5.6.12}$$

$$\psi^{'}\psi\partial_\psi F_y(\psi,x) - \psi^{'} F_y(\psi,x) - \psi\partial_x F_y(\psi,x) - \psi^{'}\psi\partial_\psi F_y(\psi,x) = 0 \tag{5.6.13}$$

que simplificada nos da:

$$(x+\lambda)\partial_\psi F_x(x,\psi) = \psi\partial_x F_y(\psi,x) \tag{5.6.14}$$

El siguiente paso es simplificar la ecuación anterior. Para ello aplicamos la regla de Leibniz ("Leibniz integral rule") que nos da una fórmula para diferenciar una integral defina cuyos límites dependen de la variable diferencial. La regla de Leibniz nos dice:

$$\frac{\partial}{\partial z}\int_{a(z)}^{b(z)} f(x,y)\,dx = \tag{5.6.15}$$

$$\int_{a(z)}^{b(z)} \frac{\partial}{\partial z} f(x,y)\,dx + f(b(z),z)\frac{\partial}{\partial z}b(z) - f(a(z),z)\frac{\partial}{\partial z}a(z)$$

Si lo aplicamos a nuestro caso:

$$\partial_\psi F_x(x,\psi) = \frac{\partial}{\partial\psi}\int_x^1 f(x,y)\,dx = \tag{5.6.16}$$

$$\int_x^1 \frac{\partial}{\partial\psi} f(x,y)\,dx + f(x,1)\frac{\partial}{\partial\psi}1 - f(x,\psi(x))\frac{\partial}{\partial\psi}\psi(x) = -f(x,\psi(x)) \tag{5.6.17}$$

y

$$\partial_x F_y(\psi,x) = \frac{\partial}{\partial x}\int_x^1 f(x,y)\,dx = \tag{5.6.18}$$

$$\int_x^1 \frac{\partial}{\partial x} f(x,y)\,dx + f(1,y)\frac{\partial}{\partial x}1 - f(x,y)\frac{\partial}{\partial x}x = -f(x,\psi(x)) \tag{5.6.19}$$

Y por lo tanto la ecuación anterior (5.6.14) se simplifica a:

$$(x+\lambda)\dot{(} - f(x,\psi(x))) = \psi\dot{(} - f(x,\psi(x))) \tag{5.6.20}$$

Despejando y resolviendo $\psi(x)$ nos queda finalmente la siguiente solución:

$$\psi(x) = x + \lambda \tag{5.6.21}$$

Sorprendentemente, ¡la función de transformación o de corte es realmente simple! Ahora podemos afirmar que las dos zonas de asignación son dos áreas de un cuadrado separado por una línea. Un ejemplo de tal división se representa en la gráfica 5.14.

Nos podría quedar la duda de si esta función es realmente un mínimo. Recordemos que la ecuación de Euler-Lagrange es una condición necesaria para obtener un extremo en el funcional, pero no nos dice nada sobre el carácter del mismo. De hecho, la línea de corte podría suponer tanto un mínimo como un máximo ya que se podrían asignar las tareas en un sentido o en otro. Es decir, se podrían asignar las tareas al jugador con valores a la derecha o a la izquierda de la recta. Por la forma de construir la asignación de tareas, parece obvio que la recta nos da el mínimo.En cualquier caso, incluimos una demostración sugerida por D. Juan Antonio Almendral que confirma estos resultados y que, además, aporta una visión muy intuitiva del funcionamiento.

Veamos, pues esta segunda demostración:

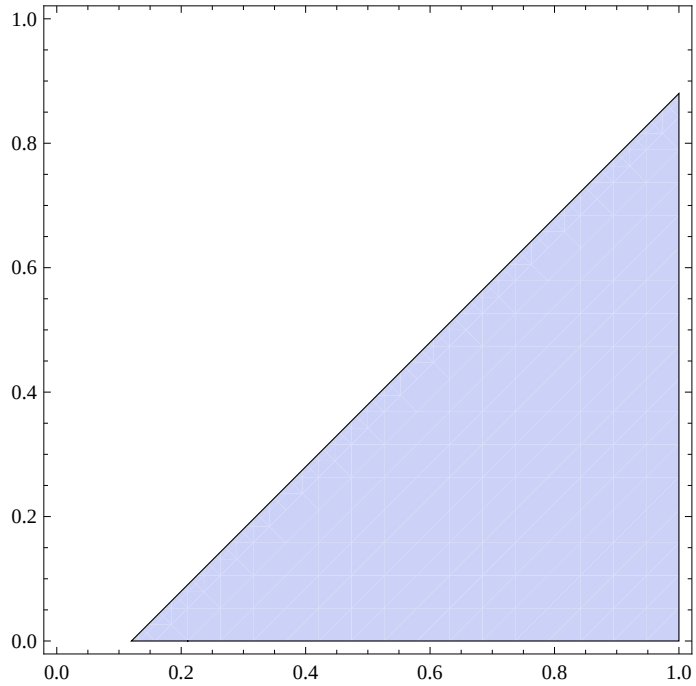

Figura 5.14: Ejemplo real de función de corte.

Demostración. Supongamos que existen dos áreas denominadas S_+ y S_- en las que el sistema asigna de forma determinista las tareas al jugador 1 y al jugador 2 respectivamente. En el resto del cuadro S_0 el sistema asigna de forma aleatoria tareas al jugador 1 según una probabilidad ξ_x y al jugador 2 según ξ_y. La unión de las tres áreas $S_+ \cup S_- \cup S_0 = S = [0,1]x[0,1]$ Sabemos que debe por las restricciones impuestas al sistema, ambos jugadores deben repartirse las tareas, y por lo tanto debe cumplirse que:

$$\int_{S_+} dF + \int_{S_0} \xi_x dF = \frac{1}{n}$$

y que

$$\int_{S_-} dF + \int_{S_0} \xi_y dF = 1 - \frac{1}{n}$$

Nótese que hemos cambiado ligeramente la notación ya que estamos utilizando

de las integrales de Lebesgue. Además, se trabaja con rangos de integración generales y no suponemos que exista la función de densidad f.

Si sumamos ambas expresiones, debemos obtener la totalidad de las tareas a ejecutar:

$$\int_{S_+} dF + \int_{S_-} dF + \int_{S_0} (\xi_y + \xi_y) dF = 1$$

$$\int_{S_0} (\xi_y + \xi_y) dF = 1 - \int_{S_+} dF - \int_{S_-} dF = \int_{S_0} dF$$

Como esto debe cumplirse para $\forall S_0$, es evidente que $\xi_y + \xi_y = 1$. Es decir, en la zona de asignación aleatoria, el sistema deberá asignar la tarea bien al jugador 1 o bien al jugador 2.

Por otra parte, el mecanismo desea optimizar el coste total del sistema, entendido como la suma de los costes de ambos jugadores:

$$\int_{S_+} x dF + \int_{S_0} x\xi_x dF + \int_{S_-} y dF + \int_{S_0} y\xi_y dF$$

Sabemos por otra parte que:

$$\int_{S_0} y\xi_y dF = \int_{S_0} y(1 - \xi_x) dF = \int_{S_0} y dF - \int_{S_0} y\xi_x dF$$

y que:

$$\int_{S_+} y dF + \int_{S_-} y dF + \int_{S_0} y dF = \int_{S} y dF$$

Luego, el sistema debe optimizar:

$$\int_{S_+} x dF + \int_{S} y dF - \int_{S_+} y dF + \int_{S_0} x\xi_x dF - \int_{S_0} y\xi_x dF$$

Dado que $\int_S y dF$ es constante (es el esfuerzo total del jugador en el área completa), el sistema debe minimizar:

$$\int_{S_+} (x - y) dF + \int_{S_0} (x - y)\xi_x dF$$

con la ligadura:

$$\int_{S_+} dF + \int_{S_0} \xi_x dF = \frac{1}{n}$$

Si ahora definimos la función de densidad:

$$g(x,y) = \begin{cases} f(x,y) & (x,y) \in S_+ \\ \xi_x(x,y) f(x,y) & (x,y) \in S_0 \\ 0 & (x,y) \in S_- \end{cases}$$

Entonces el problema se reduce a minimizar:

$$\int_{S_+} (x-y) dF + \int_{S_0} (x-y) \xi_x dF = \int_S (x-y) dG$$

con la condición:

$$\int_S dG = \frac{1}{n}$$

Obsérvese que si llamamos $h(x,y) = ng(x,y)$, y dado que n no afecta al proceso de buscar el mínimo, el sistema es equivalente a

$$\min \int_S (x-y) dH$$

sujeto a:

$$\int_S dH = 1$$

En otras palabras, estamos buscando una distribución de probabilidad h definida en $S = [0,1]x[0,1]$ tal que (x-y) sea mínimo. Si ahora realizamos un cambio de variables, haciendo que:

$$p = x + y$$

y

$$q = x - y$$

Definimos $m(p, q)$ la distribución de probabilidad resultante de aplicar el cambio de variables sobre $h(x, y)$. Nótese que el jacobiano es una constante, y por lo tanto, el proceso de optimización se convierte en:

$$\min \int_S q dM$$

sujeto a:

$$\int_S dM = 1$$

Dado que la integral a minimizar depende únicamente de q podemos simplificar la formulación reduciendo el área de integración al rango de $q \in [-1, 1]$. De la misma forma, podemos utilizar la marginal de q en lugar de la distribución M. El problema queda reducido al búsqueda de cierta F_q (considerada como la marginal de q en m) que minimice,

$$\min \int_{-1}^{1} q F_q(q) dq$$

sujeto a:

$$\int_{-1}^{1} F_q(q) dq = 1$$

La demostración pasa por observar que existe una asignación óptima que es determinista. Supongamos por contradicción, que existe una asignación no determinista y que, con esta asignación, el sistema tiene el menor coste. En esa asignación no determinista, podemos encontrar tres tipos de regiones a las que llamaremos L_+, L_0, y L_-. El jugador 1 ejecutará las tareas de forma determinista en las regiones L_+, el jugador 2 ejecutará las tareas de forma determinista en las regiones L_-, mientras que se decidirá aleatoriamente en las regiones L_0. La gráfica 5.15 representa un posible ejemplo de estas áreas. En el eje de las abscisas o eje horizontal se representa los posibles valores de q, mientras que en el eje de las ordenadas o eje vertical se representa la

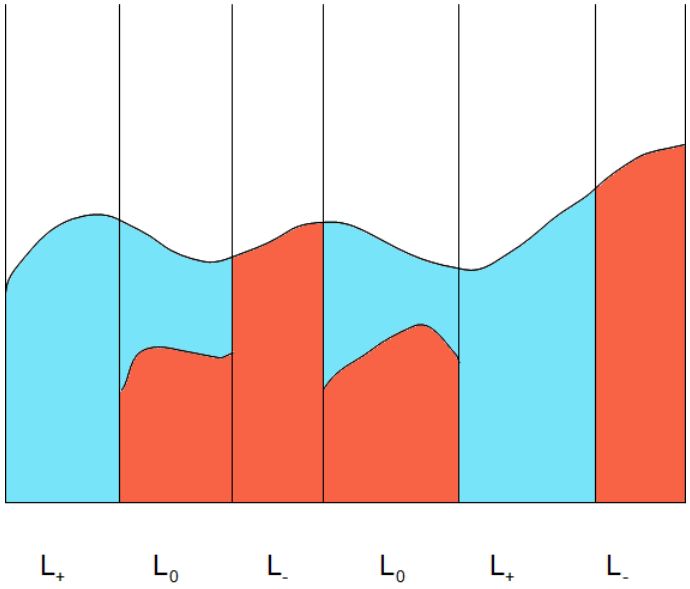

Figura 5.15: Reparto de áreas de decisión.

densidad de la función F_q anterior. Las zonas azules representan las áreas (masa de probabilidad) que le corresponden al jugador 1 (L_+). Las zonas de color rojo son las correspondientes al jugador 2 (L_-). En las zonas L_0 se ha supuesto que hay una cierta probabilidad de asignar las tareas al jugador 1 y otra de asignar las tareas al jugador 2. Por esa razón aparecen zonas azules o rojas en L_0. En dicha gráfica se ilustra un posible ejemplo en el que existen una o varias áreas de tipo L_0, ya que, por contradicción se ha supuesto que la asignación es aleatoria.

Supongamos ahora, que cada área de tipo L_0 la desdoblamos en dos áreas que denominaremos $\hat{L}_0$ y $\hat{L}_0^c$, y en las que las decisiones son deterministas en ellas y en las que suponemos que $\hat{L}_0$ se encuentra a la izquierda de $\hat{L}_0^c$. Esta construcción es evidente y podemos afirmar que siempre realizable cuando el área L_0 tiene un cierta anchura (no es una línea). Otra posibilidad es que existan varias líneas de tipo L_0. El número

de líneas de tipo L_0 puede ser finito o infinito. Si es finito, podemos volver a hacer la construcción mencionada anteriormente ($\hat{L}_0$ y $\hat{L}_0^c$) ya que bastaría con unir las líneas tipo L_0 con las áreas deterministas situadas entre ellas. Dado que hemos supuesto condiciones habituales de continuidad en las distribuciones originales (distribuciones uniformes correladas) no se contemplan alternativas patológicas o más complejas derivadas de situaciones en las que existen infinitas (contables o incontables) líneas del tipo L_0.

Sea la asignación determinista:

$$\chi = \begin{cases} 1 & \text{si } q \in \hat{L}_0, \\ 0 & \text{si } q \notin \hat{L}_0 \end{cases}$$

Veamos que $\int_{L_0} q\chi dF$ con $\int_{L_0} \chi dF = \int_{L_0} \xi dF$ es mejor (tiene menor coste) que la asignación aleatoria inicial. Es decir, queremos demostrar que

$$\int_{L_0} q\chi dF < \int_{L_0} q\xi dF$$

con la restricción anterior de conservación de trabajo.

$$\int_{L_0} \chi dF = \int_{L_0} \xi dF$$

Utilizando esa restricción podemos afirmar que:

$$\int_{\hat{L}_0} (1-\xi) dF = \int_{\hat{L}_0^c} \xi dF$$

Además, podemos afirmar que:

$$\int_{L_0} q\chi dF = \int_{\hat{L}_0} q dF$$

y que:

$$\int_{L_0} q\xi dF = \int_{\hat{L}_0} q\xi dF + \int_{\hat{L}_0^c} q\xi dF$$

pero:

$$\int_{\hat{L}_0} q(1-\xi)dF < \int_{\hat{L}_0} \max_{\hat{L}_0}\{q\}(1-\xi)dF = \max_{\hat{L}_0}\{q\}\int_{\hat{L}_0}(1-\xi)dF$$

y

$$\int_{\hat{L}_0^c} q\xi dF > \int_{\hat{L}_0^c} \min_{\hat{L}_0^c}\{q\}\xi dF = \min_{\hat{L}_0^c}\{q\}\int_{\hat{L}_0^c}\xi dF$$

Nótese que $\max_{\hat{L}_0}\{q\} \leq \min_{\hat{L}_0^c}\{q\}$, porque $\hat{L}_0$ se encuentra a la izquierda de $\hat{L}_0^c$ por construcción.

Por lo tanto, y usando $\int_{\hat{L}_0}(1-\xi)dF = \int_{\hat{L}_0^c}\xi dF$, tenemos que:

$$\int_{\hat{L}_0} q(1-\xi)dF < \int_{\hat{L}_0^c} q\xi dF$$

y por lo tanto,

$$\int_{L_0} q\chi dF < \int_{L_0} q\xi dF$$

Luego la asignación determinista es mejor y reducimos el problema a considerar L_+ y L_- ya que L_0 se ve reducido a un punto.

Por el mismo argumento, llegamos a que la conclusión de que la asignación óptima es la que acumula L_+ a la izquierda y L_- a la derecha (véase el gráfico 5.16), de modo que L_0 queda reducido a la frontera entre L_+ y L_-. □

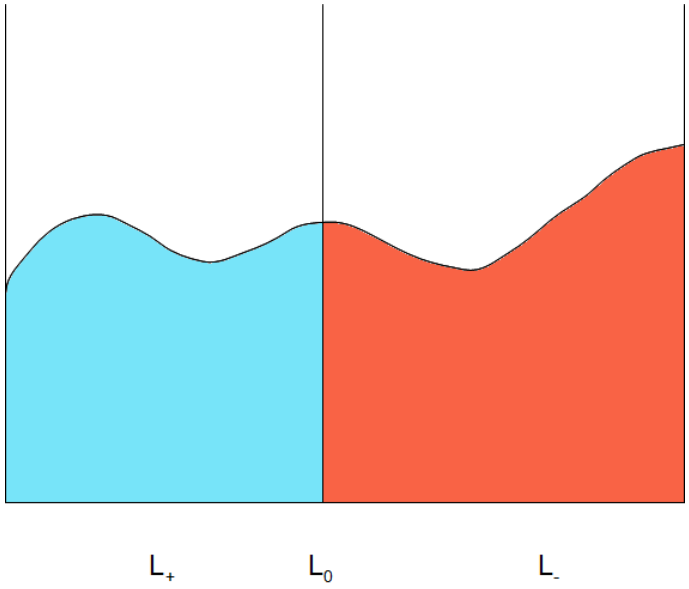

Figura 5.16: Reparto de áreas de decisión (II).

5.6.1. QPQ Básico como caso particular

Con todo lo anterior, podríamos preguntarnos hasta que punto nuestro algoritmo QPQ básico puede ser considerado como un caso particular del tratamiento matemático visto anteriormente. En este apartado, queremos volver a repasar la argumentación matemática pero en este caso, aplicándola al caso de varios jugadores independientes.

Primero, debemos recordar que cuando los jugadores son independientes, el sistema observa un jugador agregado con una distribución de probabilidad idéntica a una $Beta(1, m)$. Por lo tanto, cuando un jugador juega contra otro jugador agregado de $(n-1)$ jugadores, el sistema observa la siguiente función de densidad:

$$f(x, y) = (n-1)(1-y)^{(n-2)}$$

En este caso, la probabilidad de ejecutar la tarea según la hemos definido anteriormente (5.6.3 y 5.6.4) sería:

$$F_x(x, z) = \int_z^1 (n-1)(1-y)^{(n-2)} dy = (1-z)^{n-1} \tag{5.6.22}$$

$$F_y(w,y) = \int_w^1 (n-1)(1-y)^{(n-2)}dx = (n-1)(1-w)(1-y)^{(n-2)} \quad (5.6.23)$$

Por lo tanto calcular la ecuación de Euler-Lagrange sería:

$$(x+\lambda)(1-\psi(x))^{(n-1)} + (n-1)(1-x)^{(n-2)}(1-\psi(x))\psi(x)\psi'(x) \quad (5.6.24)$$

Es decir:

$$(-1+n)(1-\psi(x))^{(n-1)}(-x-\lambda+\psi(x)) = 0 \quad (5.6.25)$$

Despejando $\psi(x)$ nos queda la solución esperada de:

$$\psi(x) = x + \lambda \quad (5.6.26)$$

Sería útil realizar una última comprobación y calcular el valor de λ que haga que los jugadores realicen un número idéntico de tareas:

$$\int_0^1 F_x(x, x+\lambda)\, dx = \frac{(1-\lambda)^n - (-\lambda)^n}{n} \quad (5.6.27)$$

Y por lo tanto cuando $\lambda = 0$ el jugador 1 ejecutará $\frac{1}{n}$ tareas como queríamos calcular.

5.6.2. Sobre el cálculo de λ

La solución encontrada reduce el problema a encontrar el valor de λ que ajusta las tareas realizadas por el jugador. En principio, podemos solicitar a los jugadores que declaren la distribución conjunta y calcular dicho parámetro en consecuencia. Pero en general, no debemos esperar encontrar una ecuación analítica. Es decir, es posible que $f(\cdot,\cdot)$ o su acumulada $F(\cdot,\cdot)$ no tenga una forma analítica, o incluso aún existiendo tal forma, es posible que los jugadores no la conozcan y el mecanismo deba estimarla de forma empírica. Existen múltiples métodos de estimación, tanto paramétricos como no paramétricos. La mayor dificultad de estos sistemas es la velocidad de convergencia haciendo que sea necesario un número elevado de muestras. Existe una relación entre la dimensión de la función y el número de muestras necesarias. En nuestro caso, la dimensión vendría dada por el número de jugadores. Afortunadamente, cada jugador puede calcular el mecanismo QPQ haciendo que el sistema sólo tenga que trabajar con dos dimensiones (el resto se agrega en un jugador).

De todas formas, el mecanismo no necesita conocer realmente la función de densidad conjunta, nos basta con conocer la función $T(\cdot)$ que nos indica el número de tareas ejecutadas (en esperanza) dado un parámetro λ. Indicamos por $T(\lambda)$ el número de tareas que ejecutará el jugador cuando el valor de corte viene determinado por el parámetro λ. De nuevo, no podemos esperar que está función T tenga una forma analítica. Pero, bajo las condiciones adecuadas, podemos aproximar λ usando métodos de aproximación estocástica. Por las características de la función de transformación y observando en cómo influye en el número de tareas, podemos esperar que la función $T(\lambda)$ sea continua y decreciente (o creciente según el sentido de λ). Es decir, siempre existirá un valor de λ que nos proporcione el porcentaje de número de tareas deseado.

A modo de ejemplo, se muestra en la gráfica 5.17 el comportamiento de la ecuación 5.6.27 cuando se modifica λ y en el que cada curva representa a un número de jugadores diferente (de 2 a 6). Recordemos que dicha ecuación nos da el valor de tareas estimadas en función de λ y del número de jugadores n.

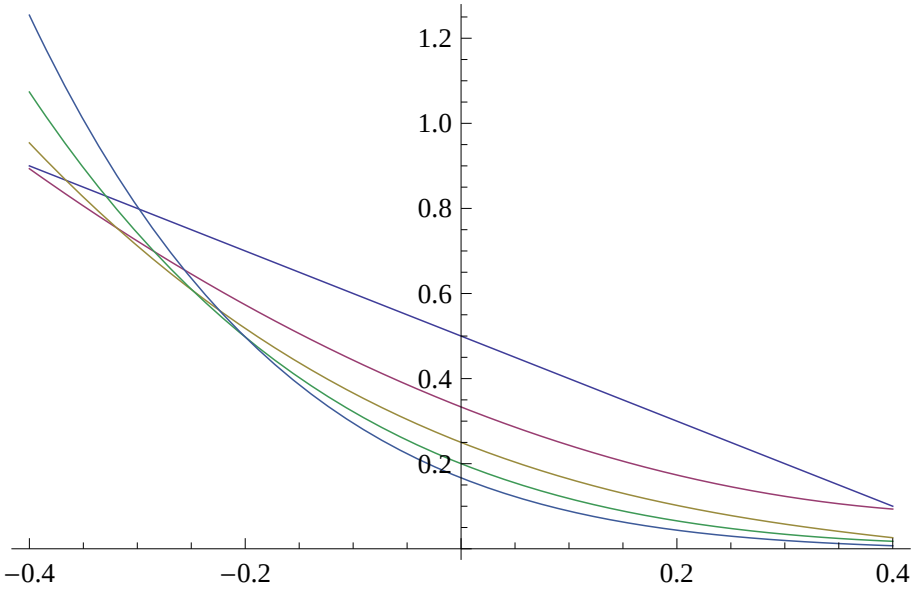

Figura 5.17: Volumen trabajo en función de λ.

Al observar la gráfica 5.14 podemos comprobar la razón de que $T(\cdot)$ sea cre-

ciente. Obsérvese que al desplazar λ se descubre una nueva área cuya masa de probabilidad se sumará a la anterior (la probabilidad siempre será positiva). Recordemos que una función es creciente en un intervalo (a, b) si para todo x_1 y para todo x_2 pertenecientes a dicho intervalo y tales que $x1 < x2$, se cumple que $f(x1) \leq f(x2)$.

Aprovechando esta característica, nuestra propuesta es aproximar λ mediante una secuencia de $\lambda_0, \lambda_1, \lambda_2, \lambda_3, ... \rightarrow \lambda$ construidas mediante algún método de aproximación estocástica. El más conocido es quizás el procedimiento de "Robbins-Monro" [RM51] aunque no es el único. Otro de los algoritmos utilizados en la teoría de control es el denominado Kiefer-Wolfowitz [KW52]. Existe una amplia bibliografía al respecto [KY03, KY97].

En su trabajo original, Robbins y Monro estudiaron la forma de obtener la raíz de una función $g(\cdot)$ de variable real. Aunque la función puede no tener una forma analítica conocida, se supone que un agente puede observar la respuesta de la función a determinados valores. Además, las observaciones se encuentran sujetas a "ruido" aleatorio que dificultan la estimación de la función.

Dada la naturaleza de nuestro problema, parece evidente aplicar el procedimiento de Robbins-Monro para la estimación de λ según se obtengan nuevos valores de costes de los jugadores, aunque para ello debemos suponer las condiciones necesarias para su aplicación, tales como una cierta continuidad en la función de trabajo $T(.)$ y que su estimación sea uniformemente acotada ("uniformly bounded"). Las funciones $T(\lambda)$ no tienen forma analítica conocida, aunque podemos experimentar con diferentes valores de λ y encontrar la solución de $T(\lambda) = 1/n$. En concreto, el algoritmo propuesto sería:

$$\lambda_{k+1} = \lambda_k + \epsilon_k(T_k - 1/n) \tag{5.6.28}$$

Siendo T_k una estimación del número medio de tareas realizadas por el jugador y donde ϵ_k es una secuencia de valores que cumple: $\epsilon_k > 0$, $\epsilon_k \rightarrow 0$, $\sum_k \epsilon_k = \infty$. En esta notación, k representa la ronda.

Para la estimación T_k se puede utilizar un histórico de muestras en la que cada jugador almacene un número de rondas limitado. En nuestra simulación hemos uti-

lizado de 500 a $1{,}000$ valores, descartando los valores más antiguos según aparecen los nuevos. En realidad, cada jugador sólo debe almacenar un par de valores por cada ronda, su propio valor y el mínimo de todos los valores declarados por el resto de los jugadores (el agregado del sistema). Con este histórico, se puede calcular el número estimado de tareas a ejecutar dado un λ.

Como futuro trabajo, sería interesante estudiar otras formas de estimación que respetando los requisitos del algoritmo, puedan reducir el computo o el número de valores almacenados. Una forma podría ser almacenar únicamente el último valor de tareas realizadas T_k y actualizar dicho valor en cada ronda, pero sin mantener ni utilizar para ello el histórico de valores. Dado que λ cambia lentamente, lo mismo hará T_k y podría servir como estimador. Esta hipótesis de trabajo ha sido simulada con resultados positivos, pero sería necesario demostrar que tal construcción cumple con los requisitos para asegurar la correcta aplicación de la aproximación estocástica.

Existen múltiples propuestas para la generación de ϵ_k que, dependiendo del caso, pueden acelerar la velocidad de convergencia a la solución buscada. En nuestras simulaciones hemos utilizado dos de sus formas más básicas $\epsilon_k = \frac{1}{k}$ y $\epsilon_k = \frac{1}{Log(k)}$, aunque ambas han arrojado resultados muy similares. En las simulaciones realizadas, el valor inicial de λ_0 era igual a cero.

5.6.3. Algoritmo QPQ con justicia en volumen de tareas

Con todo el material anterior, ahora estamos en disposición de definir la nueva versión del algoritmo QPQ y que viene representado por el algoritmo 8.

5.6.4. Propiedades

Uno de los aspectos más interesantes de nuestro nuevo mecanismo es que conserva las propiedades de su forma más básica. De hecho, esta versión de QPQ tiene las siguientes propiedades:

1. QPQ con justicia en tareas es **Individual-rationality** (Cada jugador tiene una

Algorithm 8 QPQ Correlated mechanism Fairness in Tasks (code for node i)

1: Estimate the cost c_i of the task
2: Publish the normalized cost $\bar{c}_i = PIT(c_i)$
3: Wait to receive the normalized costs $\bar{c}_j$ from the other players
4: **for all** $j \in N$ **do**
5: **if** not $GoF_Test(\bar{c}_j, Historic_j, p\text{-}th_j)$ **then**
6: $\bar{c}_j \leftarrow Random(\bar{c}_{-j})$
7: **end if**
8: $Historic_j \leftarrow Historic_j \cup \{\bar{c}_j\}$
9: **end for**
10: Let $d = \underset{j \in N}{argmin} \{\bar{c}_j + \lambda_j\}$
11: **if** $d = i$ **then**
12: execute the task
13: **else**
14: do nothing (node d will execute the task)
15: **end if**
16: Update $\lambda_j, \forall j \in N$: $\lambda_{k+1,j} = \lambda_{k,j} + \epsilon_k(T_{k,j} - 1/n)$.

utilidad positiva)

2. QPQ con justicia en tareas es **Allocative-efficient** (El mecanismo optimiza la suma de utilidades de todos los agentes, dada la ligadura de igualdad de trabajo)

3. La mejor estrategia para un jugador en QPQ con justicia en tareas es decir la verdad (revelar los costes reales es óptimo para cada jugador). Si el espacio de estrategias no incluye la posibilidad de modificar la correlación, entonces podemos decir que QPQ con justicia en tareas es **Strategy-proof**.

Las dos primeras propiedades son evidentes por la forma de construir el mecanismo. La primera propiedad (**Individual-rationality**) nos dice que todo jugador estará interesado en participar en el juego, ya que obtiene una utilidad positiva. Recordemos que el jugador obtiene una utilidad igual al coste de las tareas que realiza un contrario. Dado que el mecanismo asigna tareas de forma proporcional a cada jugador, todos los jugadores habrán ejecutado una parte del total y por lo tanto su ahorro es evidente.

Quizás sea más interesante observar que el jugador obtiene una utilidad mayor que si hubiera ejecutado la misma proporción de tareas pero asignadas de forma aleatoria. Es decir, con QPQ el jugador reduce su coste frente a otros sistemas de asignación aleatoria. En el único caso en el que QPQ podría asignar las tareas aleatoriamente (o al menos con una utilidad equivalente) sería cuando los jugadores están completamente correlados y todos dicen siempre los mismos valores. Si los jugadores reducen su correlación, el sistema solo puede mejorar. Esta propiedad es también consecuencia de la optimización del sistema.

La segunda propiedad (**Allocative-efficient**) es consecuencia por todo lo visto en los apartados anteriores. Recordemos que el algoritmo se ha construido como una solución a la ecuación (5.6.2). Efectivamente el sistema minimiza el coste del sistema entendido como la suma de todos los costes de los jugadores, aunque teniendo en cuenta la restricción de que cada jugador ejecuta una proporción de tareas dadas. Claramente, sin dicha restricción el sistema óptimo sería repartir las tareas al menor coste (es decir con un parámetro $\lambda = 0$) en cuyo caso volvemos al QPQ básico. Si repasamos la definición de **Allocative-efficient** aportada en 2.2.1, se puede comprobar que se habla de toda $d, d' \in D$. Definiendo D como el conjunto de todas las funciones de decisión que respetan la ligadura impuesta (igualdad de tareas), y recordando la construcción del mismo (cálculo variacional, etc.), es evidente que nuestro algoritmo es Allocative-efficient.

Una última duda sobre la eficiencia de QPQ se abre cuando nos planteamos si realmente nuestro algoritmo reproduce las ecuaciones obtenidas en al análisis matemático anterior (la solución obtenida con el cálculo de variaciones). En el algorit-

mo propuesto, cada i tiene su propia $\lambda_{k,i}$, y realmente lo que hacemos es aplicar el algoritmo para un vector de $\lambda_k = (\lambda_{k,1}, \lambda_{k,2}, \cdots, \lambda_{k,n})$.

En el cálculo de variaciones, se obtuvo una solución en la que sólo existía una función de corte definida para un único jugador. Realmente ambos sistemas son similares, y podemos convertir uno en otro fácilmente. Veamos cómo.

La función de decisión que aplicaría cada jugador sería:

$$
\begin{aligned}
d_1 &= argmin(c_1 + \lambda_{k,1}, c_2, \cdots, c_i, \cdots, c_n) \\
d_2 &= argmin(c_1, c_2 + \lambda_{k,2}, \cdots, c_i, \cdots, c_n) \\
&\cdots \\
d_i &= argmin(c_1, c_2, \cdots, c_i + \lambda_{k,i}, \cdots, c_n) \\
&\cdots \\
d_n &= argmin(c_1, c_2, \cdots, c_i, \cdots, c_n + \lambda_{k,n})
\end{aligned}
$$

Pero dado que el sistema debe generar la misma decisión independientemente del jugador que realice el cálculo, si sumamos todas las funciones de decisión anteriores, la decisión se conserva. Por lo tanto, la siguiente función de decisión deberá ser equivalente:

$$d = argmin(c_1 + \lambda_{k,1}, c_2 + \lambda_{k,2}, \cdots, c_i + \lambda_{k,i}, \cdots, c_n + \lambda_{k,n})$$

Es decir, en nuestra función de decisión se convierte en $d = argmin(c_i + \lambda_{k,i})$ que es equivalente a $d_i = argmin(c_{-i}, c_i + \lambda_{k,i} - \sum_{j \neq i} \lambda_{k,j})$ (recordemos que representamos al coste del jugador i por c_i y al coste de su contrario agregado por c_{-i}. Lo que estamos afirmando es que el cálculo de la decisión se puede hacer de forma distribuida y sigue siendo coherente con el mecanismo formal que hemos demostrado en los apartados anteriores.

La tercera propiedad (revelar los costes reales es una estrategia óptima) es la más interesante de todas. Teniendo en cuenta que hemos impuesto la condición de que el jugador no puede modificar la correlación de la probabilidad conjunta que pueda

existir entre jugadores, esta propiedad nos dice que la mejor estrategia del jugador es declarar sus costes reales. Es decir, el jugador cuando analiza sus posibilidades, decide decir la verdad sobre sus costes ya que en cualquier otro caso, su utilidad sería menor.

La restricción de que el jugador no puede alterar la correlación entre jugadores implica que el sistema observa una distribución de probabilidad $\hat{f}(\cdot,\cdot)$ que debe ser idéntica a la real. Es decir, el sistema observa una $\hat{f}(\cdot,\cdot) = f(\cdot,\cdot)$. Esta restricción no implica que el jugador 1 no pueda mentir sobres sus valores. Lo que estamos diciendo es que la densidad de probabilidad de que el sistema observe un determinado par de valores (x, y) es igual a la mentira (z, y). El sistema percibe la misma distribución de probabilidad conjunta, aunque el jugador 1 está realmente mintiendo. Matemáticamente, lo que tenemos son tres variables aleatorias que se corresponden con los valores reales x e y, más el valor del coste z que falsamente declara el jugador 1. Es decir, cuando tenemos mentiras, lo que tenemos es una función de densidad que liga las tres variables: $h(x, y, z)$. La restricción que imponemos al jugador 1 es que

$$f(x,y) = \int_0^1 h(x,y,z)dz = \int_0^1 h(x,y,z)dx = \hat{f}(z,y)$$

Cuando sucede esto, la ligadura impuesta al mecanismo que garantiza un determinado volumen de tareas hace que tanto con o sin mentiras, el parámetro λ sea el mismo.

Dado que el sistema calcula λ para que la restricción (ligadura) sea cierta, tiene que cumplirse que cuando el jugador 1 dice la verdad, el jugador 2 verá:

$$\int_0^1 \int_{y+\lambda}^1 f(x,y)dx\ dy = \frac{n-1}{n}$$

Cuando el jugador 1 dice mentiras, el sistema calculará un nuevo $\hat{\lambda}$ para el jugador 2 que cumpla:

$$\int_0^1 \int_{y+\hat{\lambda}}^1 \hat{f}(z,y)dx\ dz = \frac{n-1}{n}$$

Pero dado que $f(x, y) = \hat{f}(z, y)$, y que los rangos de integración son idénticos, se tiene que cumplir que $\lambda = \hat{\lambda}$. Es decir, el jugador 2 no observa ningún efecto en

la función de corte cuando el contrario miente. Y por lo tanto, el jugador 2 no ve modificada la probabilidad condicional de que se le asigne la tarea cuando anuncia un determinado valor. Es decir, la probabilidad de que se asigne la tarea al jugador 2 cuando éste dice el valor y, $Pr[d = 2 \mid y]$ es idéntica tanto si la calculamos utilizando $\hat{f}(\cdot,\cdot)$ o $f(\cdot,\cdot)$. Obsérvese que el coste del jugador 2 viene fijado por dicha probabilidad y por lo tanto por la forma de $f(\cdot,\cdot)$ (o de $\hat{f}(\cdot,\cdot)$):

$$coste = \int_0^1 yPr[d = 2 \mid y]dy$$

Es decir, el coste del jugador 2 no se ve alterado por las mentiras del jugador 1. Por lo tanto, y dado que el sistema era óptimo, no puede ser que el jugador 1 gane al mentir ya que eso implicaría que la suma de ambos esfuerzos sería menor. Pero eso contradice que el sistema era óptimo y por lo tanto anula la hipótesis de que el jugador 1 gana al mentir.

Nótese que en nuestra demostración hemos utilizado la propiedad de que el jugador no puede alterar el grado de correlación, o lo que es similar, que $f(x, y) = \hat{f}(z, y)$. Como se verá en el próximo capítulo, nuestro mecanismo *no es Strategy-proof* cuando relajamos esta suposición.

5.6.5. Comentarios finales

Nos gustaría cerrar esta versión del algoritmo comentando algunas de las propiedades del QPQ básico que no son aplicables directamente a QPQ con reparto justo de tareas.

Por una parte, nos encontramos que no es posible conocer de antermano la utilidad de los jugadores, aunque dado un cierto nivel de correlación QPQ garantizará una utilidad. Si recordamos QPQ básico, el diseñador de mecanismos y los jugadores conocían la utilidad esperada de cada jugador, que venía fijada con anterioridad al inicio del juego. Independientemente del contrario, un jugador podía conocer cuando obtendría de beneficio. Ahora, en cambio, la utilidad del jugador depende del grado de correlación que mantiene con el contrario. Como se ha visto en la demostración an-

terior, el contrario no podrá alterar la probabilidad de asignación de tareas (que viene dada por grado de correlación), y por lo tanto el jugador tendrá garantizado un nivel de utilidad. Aunque ese nivel de utilidad no se puede alterar, no es posible conocerlo antes del inicio del juego. Realmente, esto no debería suponer un inconveniente para que el jugador decida participar o no en el juego, ya que éste siempre saldrá beneficiado de su participación en el juego. Ni tampoco le influye en su decisión de ser honesto o deshonesto. Simplemente, no podrá predecir cuanta utilidad obtendrá antes de iniciar el juego.

Otro aspecto relevante sobre este punto se refiere al umbral de cierre del test de bondad. Antes, con QPQ básico, el umbral estaba en función de la desviación que existía entre la esperanza prevista de beneficio y el beneficio real obtenido. Si el jugador ganaba más de lo esperado, el umbral de aceptación se cerraba haciendo más difícil la mentira. Dado que, en nuestro caso, es imposible determinar la esperanza de beneficio antes de iniciar el juego, no es posible aplicar un test dinámico de bondad similar al descrito con QPQ Básico. Si deseamos mantener una garantía de honestidad, una solución posible es ampliar la longitud de muestras.

Finalmente, existe el problema de cómo generar el valor de sustitución cuando el sistema cree que un jugador miente. Antes, se generaba un valor siguiendo una distribución uniforme e independiente del jugador. Ahora, el valor generado debe seguir una distribución de probabilidad idéntica a la probabilidad de ejecutar la tarea condicionada al valor declarado por el jugador honesto. Es decir, es necesario mantener información estadística sobre dicha probabilidad mediante técnicas de regresión. Este punto es relevante ya que deseamos que el jugador honesto no vea modificada su utilidad, y para ello debemos asegurarle la misma probabilidad de ejecutar la tarea.

A un nivel práctico, todos estos elementos implican una cierta dificultad estadística ya que el número de muestras tiene necesariamente que aumentar, así como mantener información tanto del histórico del jugador, como de la probabilidad de ejecutar la tarea. Esa información puede no estar presente al inicio del juego, por lo que muchas de las propiedades no serán aplicables hasta que el juego haya avanzado y tenga suficiente información cómo para garantizar un nivel suficientemente aproxi-

mado de beneficios y honestidad.

En las simulaciones realizadas se han utilizado longitudes de muestras de unos 100 a 300 valores, consiguiendo unos resultados aceptables con dos jugadores. La probabilidad de ejecutar una tarea se ha estimado considerando niveles discretos (en nuestra simulación se han considerado 10 niveles). En cualquier caso, creemos que con técnicas de regresión más avanzadas, se podrían mejorar significativamente los resultados obtenidos. Téngase en cuenta que, cuando el jugador es honesto, la única información relevante es la necesaria para aplicar el test de bondad y en ese caso, un nivel de 50 a 100 muestras parece razonable. En cambio, cuando el jugador es deshonesto, se necesita información para generar el valor alternativo y en ese caso, es necesario un número significativamente mayor de muestras.

5.7. Caso 2: Justicia en el esfuerzo

En este apartado estudiaremos un nuevo concepto de justicia basado en un reparto equitativo de esfuerzo. En lugar de intentar que el mecanismo determine el número de tareas que debe ejecutar cada jugador, en este caso, el mecanismo decide la forma de repartir las tareas, aunque imponiendo la restricción de que la suma de los costes sea equitativo. Lo que estamos buscando es un mecanismo en el que los jugadores realizan un esfuerzo equitativo según un porcentaje de reparto que está definido por un parámetro del algoritmo que llamaremos δ. Es decir $\mathbb{E}[coste_1] = \delta\mathbb{E}[coste_2]$. Si todos los jugadores realizan un mismo esfuerzo, el jugador agregado realizará un esfuerzo $(n-1)$ veces mayor que el jugador aislado. Eso nos lleva a que un valor justo de δ sea por ejemplo $\frac{1}{n-1}$. Es decir, el jugador independiente hace un esfuerzo similar a los $n-1$ jugadores agregados. Como en el mecanismo anterior, se supone que todos lo jugadores conocen las reglas del juego y que conocen el parámetro δ antes de iniciar el algoritmo. Al inicio se pide a los jugadores que declaren la distribución de probabilidad conjunta con la que van a participar en el juego. Es decir, se les pide que declaren la función de distribución conjunta $f(.,.)$ (o su correspondiente acumulada F). Más adelante vamos a obtener dicha función de forma empírica, pero por el

momento supongamos que se nos revela al comenzar el algoritmo.

Por lo tanto, lo que estamos buscando es un mecanismo que optimice el siguiente sistema:

$$\min_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right)$$

sujeto a, (5.7.1)

$$\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx = \delta \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy$$

Con la definición de funciones F_x y F_y anteriores (5.6.3 y 5.6.3), el sistema a optimizar queda:

$$\min_{\psi} \left(\int_0^1 x\, F_x(x,\psi(x))dx + \int_0^1 y\, F_y(y,\psi^{-1}(y))dy\right)$$

sujeto a, (5.7.2)

$$\int_0^1 x\, F_x(x,\psi(x))dx = \delta \int_0^1 y\, F_y(y,\psi^{-1}(y))dy$$

Al igual que hicimos anteriormente para el caso de justicia en el número de tareas, ahora construimos un funcional utilizando un parámetro de Lagrange. Este sistema es equivalente a optimizar:

$$\int_0^1 x\, F_x(x,\psi(x))dx + \int_0^1 y\, F_y(y,\psi^{-1}(y))dy + \lambda\left(\int_0^1 x\, F_x(x,\psi(x))dx - \delta \int_0^1 y\, F_y(y,\psi^{-1}(y))dy\right) \tag{5.7.3}$$

La ecuación anterior (5.7.3) define un funcional a ser optimizado con la forma:

$$\int_0^1 x\, I(x,\psi,\psi')dx$$

donde (5.7.4)

$$I(x,\psi,\psi') = x(1+\lambda)F_x(x,\psi) + \psi(1-\lambda\delta)F_y(\psi,x)\psi'$$

Volviendo a aplicar la ecuación de Euler-Lagrange, obtenemos la siguiente solución:

$$x(1+\lambda)\partial_\psi F_x(x,\psi) = (1-\lambda\delta)\psi\partial_x F_y(\psi,x) \tag{5.7.5}$$

Usando las mismas técnicas anteriormente vistas, la ecuación anterior (5.7.5) se simplifica:

$$x(1+\lambda)f(x,\psi) = (1-\lambda\delta)\psi f(x,\psi) \tag{5.7.6}$$

y por lo tanto, la función de corte $\psi(x)$ tendrá la forma:

$$\psi(x) = \frac{1+\lambda}{1-\lambda\delta}x \tag{5.7.7}$$

La solución es muy similar al caso anterior ya que la función de corte sigue siendo una línea, aunque ahora el parámetro de ajuste λ determina la pendiente de la misma.

5.7.1. Sobre el nuevo cálculo de lambda

De nuevo nos enfrentamos al problema de calcular el parámetro de Lagrange λ dada la distribución de probabilidad declarada por los jugadores. En este caso, la ligadura del sistema viene dada por el cociente entre esfuerzos. Denominamos por $R(.)$ a la función que dado un valor de λ nos devuelve el cociente entre esfuerzos. Es decir,

$$R[\lambda] = \frac{Esfuerzo_x[\lambda]}{Esfuerzo_y[\lambda]}$$

es la función que determina el cociente entre esfuerzos dado un valor de corte determinado por la recta 5.7.7.

Suponiendo las condiciones apropiadas de continuidad y que la estimación $R_k = R(\lambda)$ cumple con los requisitos de Robbins-Monro, podemos volver a aplicar la aproximación estocástica definida por:

$$\lambda_{k+1} = \lambda_k + \epsilon_k(R_k - \delta)$$

Siendo R_k el cociente estimado entre esfuerzos de jugadores individual y agregado y donde ϵ_k es una secuencia de valores que cumple las condiciones anteriormente mencionadas (convergencia lenta hacia cero). De nuevo, k representa la ronda. Al igual que en el caso 1, la estimación se puede realizar manteniendo un histórico de valores.

Por lo tanto el algoritmo sería el siguiente. Para cada jugador i calcula su λ_i siguiendo el algoritmo de Robbins-Monro. Cada jugador tendrá su propia λ_i . Por lo tanto, lo que hacemos es aplicar el algoritmo para un vector de $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_n)$. Observar que la función de decisión se convierte en $d = argmin(c_j, (1 + \frac{1+\lambda_i}{1-\lambda_i\delta})c_i)$. Esta función de decisión es la que calcula cada jugador i, siendo j el resto de los jugadores.

Al igual que vimos en la versión de QPQ con reparto justo de tareas, se puede comprobar que la función de decisión distribuida es equivalente a la obtenida por las herramientas matemáticas. En la función de decisión distribuida, cada jugador i tendrá su propia $\lambda_{k,i}$. Por lo tanto, lo que hacemos es aplicar el algoritmo de Robbins-Monro para un vector de $\lambda_k = (\lambda_{k,1}, \lambda_{k,2}, \cdots, \lambda_{k,n})$.

La función de decisión que aplicaría cada jugador sería:

$$d_i = argmin(c_1, c_2, \cdots, \frac{1+\lambda_i}{1-\lambda_i\delta}c_i, \cdots, c_n)$$

Suponiendo que $\frac{1}{1-\lambda_i\delta}$ sea un valor positivo distinto de cero, podemos multiplicar todos los miembros de la ecuación anterior manteniendo la decisión. Es decir, el jugador obtendría la misma decisión con la siguiente ecuación:

$$d_i = argmin((1-\lambda_i\delta)c_1, (1-\lambda_i\delta)c_2, \cdots, (1+\lambda_i)c_i, \cdots, (1-\lambda_i\delta)c_n)$$

Pero cada jugador, estaría aplicando una ecuación similar, por lo tanto:

$$d_1 = argmin((1+\lambda_1)c_1, (1-\lambda_1\delta)c_2, \cdots, (1-\lambda_1\delta)c_i, \cdots, (1-\lambda_1\delta)c_n)$$
$$d_2 = argmin((1-\lambda_2\delta)c_1, (1+\lambda_2)c_2, \cdots, (1-\lambda_2\delta)c_i, \cdots, (1-\lambda_2\delta)c_n)$$
$$\cdots$$
$$d_i = argmin((1-\lambda_i\delta)c_1, (1-\lambda_i\delta)c_2, \cdots, (1+\lambda_i)c_i, \cdots, (1-\lambda_i\delta)c_n)$$
$$\cdots$$
$$d_n = argmin((1-\lambda_n\delta)c_1, (1-\lambda_n\delta)c_2, \cdots, (1-\lambda_n\delta)c_i, \cdots, (1+\lambda_n)c_n)$$

Todas las funciones anteriores deben ser equivalentes, ya que la decisión de asignación es la misma, con independencia de quien la calcule. Por lo tanto, sumando los

elementos de cada función de decisión nos dará otra función de decisión que sigue siendo equivalente:

$$d = argmin((n + \lambda_1 - \sum_{j \neq 1} \delta)c_1, \cdots, ((n + \lambda_n - \sum_{j \neq n} \delta)c_n)$$

Que operando y simplificando nos queda:

$$d = argmin(\lambda_1^{'} c_1, \cdots, \lambda_n^{'} c_n)$$

donde:$\lambda_1^{'} = 1 + \frac{\lambda_n - \sum_{j \neq n} \delta}{n}$

Lo que nos confirma que existe una relación entre las λ de los jugadores, el parámetro δ y la inclinación de la recta obtenida por el cálculo de variaciones.

5.7.2. Algoritmo QPQ con justicia en costes

Con todo lo comentado anteriormente, ya estamos en disposición de definir otra versión del algoritmo QPQ, y que viene representado por 8. Como era de esperar, esta nueva versión es muy similar a la versión de QPQ con justicia en el reparto de tareas. Realmente, lo único que se ve alterado es la función de transformación y el cálculo de los parámetros que lo definen.

5.7.3. Propiedades

De nuevo, podemos afirmar que esta nueva versión de QPQ tiene las propiedades habituales:

1. QPQ con justicia en coste es **Individual-rationality** (Cada jugador tiene una utilidad positiva)

2. QPQ con justicia en coste es **Allocative-efficient** (El mecanismo optimiza la suma de utilidades de todos los agentes, dada la ligadura de igualdad de trabajo)

3. Los jugadores tienen incentivos para declarar sus costes reales con QPQ con justicia en coste. De nuevo, si limitamos el espacio de estrategias para que no

Algorithm 9 QPQ Correlated mechanism Fairness in effort (code for node i)

Estimate the cost c_i of the task
Publish the normalized cost $\bar{c}_i = PIT(c_i)$
Wait to receive the normalized costs $\bar{c}_j$ from the other players
for all $j \in N$ **do**
    if not $GoF_Test(\bar{c}_j, Historic_j, p\text{-}th_j)$ **then**
        $\bar{c}_j \leftarrow Random(\bar{c}_{-j})$
    end if
    $Historic_j \leftarrow Historic_j \cup \{\bar{c}_j\}$
end for
Let $d = \underset{j \in N}{argmin} \{\lambda_j \bar{c}_j\}$
if $d = i$ **then**
    execute the task
else
    do nothing (node d will execute the task)
end if
Update $\lambda_j, \forall j \in N$: $\lambda_{k+1,j} = \lambda_{k,j} + \epsilon_k(R_{k,j} - \delta)$.

puedan cambiar la correlacion, entonces QPQ con justicia en coste es **Strategy-proof**.

Demostración. Las dos primeras propiedades son evidentes a la luz de todo lo anterior. Dado que los jugadores tiene un coste de ejecución proporcional al numero de jugadores, menor que el total correspondiente a todas las tareas, es evidente que todos los jugadores obtienen un mayor beneficio participando del juego que si no participan. Además, y por la forma de obtener la función de corte, es también evidente que

la función optimiza el coste en el reparto de las tareas, aunque siempre teniendo en cuenta que es un óptimo con las restricciones o ligaduras impuestas.

Finalmente, quizás siga siendo interesante repasar los argumentos que afirman que el jugador tiene interés en declarar los valores reales. De nuevo suponemos que

$$f(x,y) = \int_0^1 h(x,y,z)dz = \int_0^1 h(x,y,z)dx = \hat{f}(z,y)$$

Cuando sucede esto, la ligadura impuesta al mecanismo garantiza un determinado porcentaje de trabajo. Dado que el sistema calcula λ para que la restricción (ligadura) sea cierta, tiene que cumplirse que cuando el jugador 1 dice la verdad, el jugador 2 verá:

$$\int_0^1 y \int_{\lambda y}^1 f(x,y)dx\, dy = \frac{1}{\delta}\int_0^1 x \int_x^1 f(x,y)dy\, dx$$

Cuando el jugador 1 dice mentiras, el sistema calculará un nuevo $\hat{\lambda}$ que respete las ligaduras y que cumpla:

$$\int_0^1 y \int_{\hat{\lambda} y}^1 \hat{f}(z,y)dz\, dy = \frac{1}{\delta}\int_0^1 z \int_z^1 \hat{f}(z,y)dy\, dz$$

Pero dado que $f(x,y) = \hat{f}(z,y)$, y que el rango de integración son idénticos, se tiene que cumplir que $\lambda = \hat{\lambda}$. Es decir, el jugador 2 no observa ningún efecto en la función de corte cuando el contrario miente.

Es decir, el coste del jugador 2 no se ve alterado por las mentiras del jugador 1. Por lo tanto no puede ser que el jugador 1 gane al mentir ya que eso implicaría que la suma de ambos esfuerzos sería menor. Pero eso contradice que el sistema era óptimo y por lo tanto anula la hipótesis de que el jugador 1 gana al mentir. □

Nótese que, de nuevo, en nuestra demostración hemos utilizado la propiedad de que el jugador no puede alterar el grado de correlación, o lo que es similar, que $f(x, y) = \hat{f}(z, y)$. Y, como el lector podrá sospechar, de nuevo mostraremos que nuestro mecanismo *no es strategy-proof* si relajamos esta suposición.

Al igual que vimos con la versión de QPQ con justicia en el número de tareas, nuestra nueva versión de QPQ no puede asegurar un nivel de beneficio sin conocer previamente el nivel de correlación entre los jugadores. Pero dado que hemos supuesto que el jugador deshonesto no puede alterar la correlación, la esperanza de beneficio estará fijada, aunque sea desconocida de antemano. En general, se aplican los mismos comentarios realizados con respecto al test de bondad, generación de valores ante la mentira y las consideraciones estadísticas de la implantación.

Capítulo 6

Otros aspectos de QPQ y los mecanismos VCG

> "Selfishness is not living as one wishes to live, it is asking others to live as one wishes to live."
>
> Oscar Wilde

6.1. Introducción

A lo largo de la presente tesis hemos analizado un mismo problema desde diferentes puntos de vista y hemos enriquecido el modelo suponiendo restricciones cada vez más débiles. Hemos pasado de un modelo en el que los pagos se podrían realizar hasta un sistema en el que no existían pagos y los jugadores podrían tener distribuciones correladas, aunque sin poder modificar la correlación entre ellos. En este capítulo trataremos de analizar las limitaciones existentes, cómo ampliar el modelo actual y preparar el camino para las futuras vías de investigación.

Existen dos limitaciones en el modelo que, en opinión del autor, deberían ser analizadas con más profundidad en trabajos futuros. Por una parte, se ha supuesto que los jugadores no pueden afectar al grado de correlación que tienen entre ellos. Veremos en el presente capítulo que si los jugadores disponen de esas herramientas

(ampliando, por tanto, el espacio de estrategias), los mecanismos QPQ definidos hasta la fecha dejan de ser *strategy-proof*. Por otra parte, se ha supuesto que el número de jugadores es conocido. Pero los jugadores pueden realizar agrupaciones y simular ser un único jugador frente a los demás. Los mecanismos de justicia que hemos visto hasta la fecha no serían válidos ya que suponen conocer el número de jugadores y repartir el trabajo de forma equitativa.

Este capítulo aborda ambos problemas y, aunque somos conscientes de que las soluciones aportadas tienen aún muchos caminos por explorar, creemos que la información tratada en estos apartados puede ser de gran utilidad para futuras líneas de investigación.

El esquema del presente capítulo es el siguiente. En la siguiente sección, se demuestra mediante contraejemplos que los mecanismos QPQ propuestos no son strategy-proof cuando los jugadores pueden alterar la correlación. La resolución de este problema no parece ser evidente. En este apartado se exponen algunas de las soluciones exploradas con anterioridad y las principales conclusiones obtenidas. A continuación, se analizarán alternativas al concepto de número real de jugadores, su impacto en QPQ y las soluciones propuestas.

6.2. Modificación de la correlación

Hasta ahora, hemos evitado la posibilidad de que un jugador pueda alterar la distribución de probabilidad conjunta, y no únicamente la suya propia. Los algoritmos hasta ahora presentados han supuesto que el jugador podía alterar su propia distribución de probabilidad con la restricción de que siguiera teniendo la apariencia de una uniforme. Es decir puede modificar su distribución pero su marginal con respecto a los demás jugadores debía seguir siendo uniforme. Lo que nos gustaría proponer es una alternativa al modelo y planteamos la posibilidad de que el jugador pudiera alterar la correlación entre los jugadores.

En este apartado veremos en qué medida se ven afectados los mecanismos propuestos, demostrando mediante contraejemplos que las dos versiones de QPQ con

justicia no son strategy-proof cuando ampliamos el espacio de estrategia de los jugadores.

En general, todos los ejemplos que proponemos en este capítulo se basan en la idea de que el jugador mentiroso (supuestamente el jugador 1) puede alterar la correlación de la distribución mediante una técnica que hemos denominado "*la mentira de los cuadros*". Supongamos que el jugador conoce su propio valor y tiene un conocimiento del valor que diría el contrario. La gráfica 6.1 muestra un posible ejemplo de esta técnica. La idea es que el jugador desplace parte de la probabilidad de una zona a otras. Debido a que las marginales deben seguir siendo las mismas, el jugador mentiroso debe encontrar zonas a la misma altura y desplazamiento en el eje x, que compense la modificación. Es decir, si un jugador reduce la probabilidad de decir ciertos valores en una zona, debe asegurarse que ha ampliado otra de masa similar a la misma altura. Además debe asegurarse que hay otra zona con el mismo desplazamiento en x que compensa la reducción.

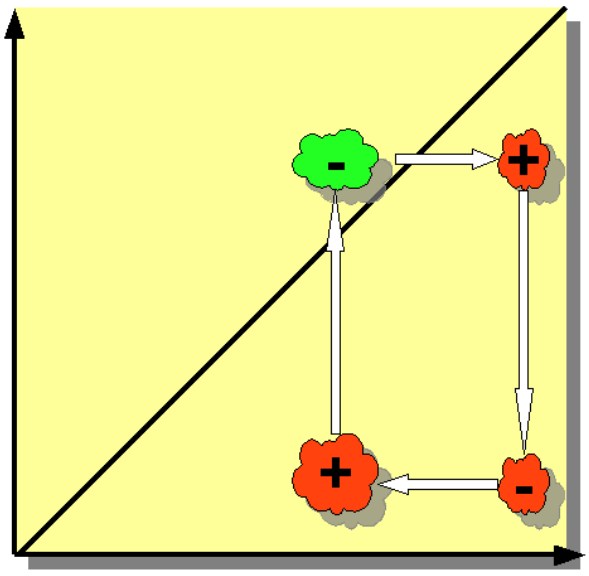

Figura 6.1: Estrategia de desplazamiento de la probabilidad.

6.2.1. Modificación correlación en QPQ Básico

Veamos un ejemplo sencillo y que nos aporte una visión del funcionamiento de la mentira. Supongamos dos jugadores que en realidad son independientes, siendo su distribución conjunta: $f(x, y) = 1$, ambos con marginales uniformes $f_x(x) = 1$ y $f_y(y) = 1$.

Ahora el jugador 1 altera la distribución de probabilidad de tal forma que simula un desplazamiento de la probabilidad de una zona a otra, haciéndose pasar por un jugador menos efectivo. Dado que las marginales deben seguir pareciendo uniformes, el desplazamiento de la masa de probabilidad debe ser compensada en otras áreas. Gráficamente, las gráficas 6.2 y 6.3 representan un posible ejemplo de tal estrategia. La primera gráfica en 2D representa la función de densidad de probabilidad f, donde cada eje representa el soporte de cada uno de los jugadores. Dado que ambos utilizan la transformación PIT, en ambos casos, el soporte es el intervalo unidad $[0, 1]$. El color de la gráfica representa el valor de la función (la densidad de probabilidad). La diferencia de tonalidad en los "cuadros" es un aumento (más claro) o disminución (más oscuro) de la densidad en esa área. En la gráfica 6.3 se representan los mismos conceptos pero en tres dimensiones. Se puede observar cómo el jugador 1 ha desplazado parte de la densidad de un área a otra, pero siempre manteniendo las restricciones del sistema. La línea azul $y = x$ es la línea de corte que utiliza QPQ básico para la función de decisión. El área superior es la zona de todas las tareas asignadas al jugador 1, mientras que la zona inferior se corresponde con las tareas del jugador 2.

Es fácil comprobar que esta estrategia mantiene las restricciones del sistema, es decir, que se mantienen las marginales: tanto el jugador 1 como el jugador 2 siguen ofreciendo una marginal uniforme.

De las gráficas anteriores podemos observar que el jugador 1 ha reducido la densidad en el cuadro superior izquierdo (zona azul oscuro). Para compensar esa pérdida de densidad ha tenido que aumentar la densidad en el cuadro a la derecha y abajo (zonas azul claro). Obsérvese que el jugador 1 ha procurado que la reducción de la densidad se sitúe en un único cuadro por encima de la línea de decisión $y = x$, mientras que las tres restantes se encuentran en la zona del jugador 2.

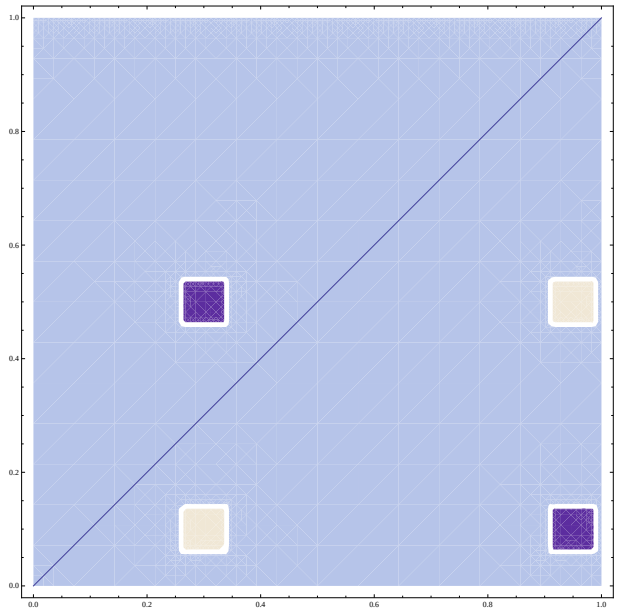

Figura 6.2: 2 Jugadores correlados.

Con esta manipulación, el jugador 1 realiza menos tareas en el cuadro superior izquierda y, por lo tanto, realizará menos esfuerzo. En realidad, se puede comprobar que para este ejemplo el jugador 1 realizará un esfuerzo de $0{,}16609$ unidades en lugar del valor previsto de $1{,}6666$ unidades (normalizadas).

El algoritmo QPQ Básico compensa esta situación comparando el coste del jugador con el teórico previsto y adaptando el nivel de confianza del test de Kolmogorov. Si el jugador intenta engañar al sistema, el algoritmo rechazará valores de coste generando nuevos valores aleatorios. Como se ha visto en los capítulos anteriores eso provoca un mayor coste para el jugador 1, mientras mantiene el coste del jugador 2 dentro de lo esperado.

6.2.2. QPQ con justicia de tareas

Cuando se trata del algoritmo QPQ con restricción en el número de tareas, existe también una estrategia que mejora la utilidad del jugador. El procedimiento es muy

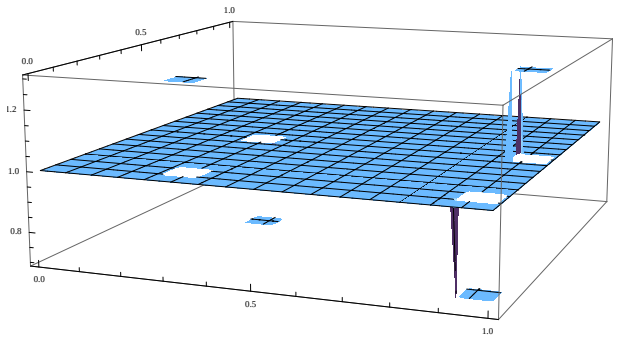

Figura 6.3: 2 Jugadores correlados, Vista en 3D.

similar a lo visto anteriormente, aunque ahora debe cumplirse la condición de que el jugador debe tener una apropiada distribución de probabilidad. Veamos en primer lugar la intuición del funcionamiento de la estrategia a realizar por el jugador 1, para ver al final del apartado un ejemplo concreto del mismo.

El jugador 1 decide reducir el número de tareas que le corresponden. Para ello altera sus valores en la zona coloreada de color verde en la gráfica 6.1. Dado que debe conservar su distribución marginal como uniforme, decide elevar la probabilidad de los valores que hay en la zona roja que existe en el mismo rango de valores de x y que se encuentra justamente en la parte inferior. De igual forma, y dado que tiene que conservar la marginal del otro jugador, es necesario compensar la alteración de probabilidad. Se han marcado con áreas rojas todas las compensaciones y se indica con un signo positivo o negativo si la densidad aumenta o disminuye en dicha zona.

Dado que las zonas rojas no están asignadas al jugador 1, aumentar la probabilidad de esas zonas no afecta a su utilidad. Por lo que el jugador 1 tendrá un menor volumen de tareas a ejecutar mientras que el jugador 2 tendrá un mayor volumen. Pero con la restricción de igual volumen de tareas a ejecutar, esa situación no puede darse ya que el sistema ajustará el parámetro λ adaptándolo en consecuencia. En este caso, la λ deberá moverse hacia abajo, asignando al jugador un área de trabajo más amplia. Si resulta que ese jugador tiene una mayor masa de probabilidad en las áreas

bajas de su coste, es razonable pensar que se beneficiará de dicha estrategia. En la gráfica 6.4 se puede observar el resultado de esta acción.

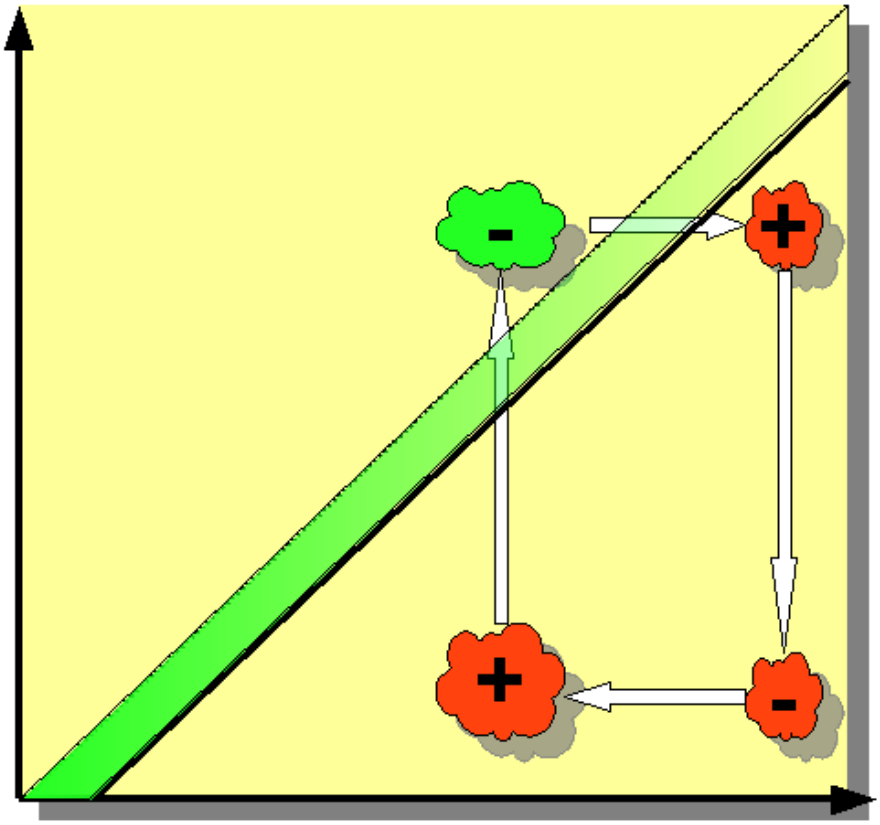

Figura 6.4: Estrategia de desplazamiento de la probabilidad (II).

Es evidente que los volúmenes de las dos zonas verdes de la gráfica 6.4 deben ser iguales. Es decir, el número de tareas que se han reducido en la burbuja inicial debe ser igual al que se ha ampliado con la tira resultante de mover el parámetro λ. Si se diera el caso de que el jugador 1 tuviera una mayor densidad cerca del origen, podría resultar que hay más tareas que en la burbuja. En ese caso su utilidad será menor mintiendo que si dice la verdad. Las tareas situadas cercanas del origen cuestan menos.

Veamos un ejemplo concreto en el que dicha estrategia resulta claramente gana-

dora. Supongamos dos jugadores con una función de densidad conjunta igual a:

$$\mathrm{f(x,y)} = \frac{1}{0{,}25}\begin{cases}(x-0{,}5)(1-y)+(0{,}5-(x-0{,}5))\cdot y & x \geq 0{,}5\\ (-x+0{,}5)(1-y)+(0{,}5-(-x+0{,}5))\cdot y & x < 0{,}5\end{cases} \tag{6.2.1}$$

La gráfica 6.5 representa esta distribución en tres dimensiones, mientras que la gráfica 6.6 representa la vista de contorno de la misma distribución.

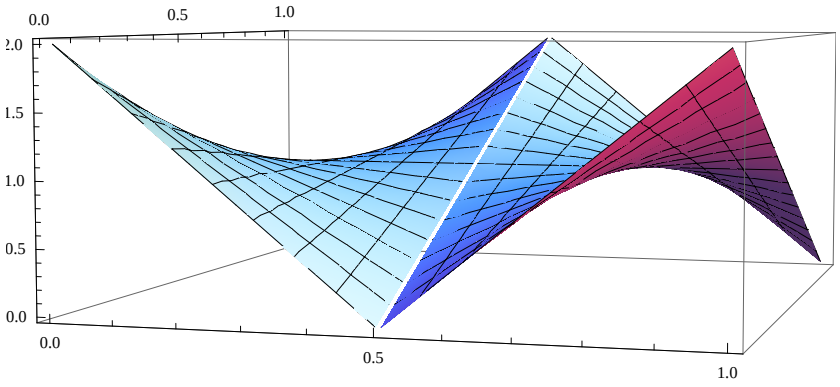

Figura 6.5: Contraejemplo 1, vista 3D.

No es difícil comprobar que es una distribución conforme a nuestros requisitos, dado que las marginales de los jugadores siguen siendo uniformes. Es decir, las siguientes integrales son iguales a la unidad:

$$f_x(x) = \int_0^1 f(x,y)dy = 1$$

y que:

$$f_y(y) = \int_0^1 f(x,y)dx = 1$$

Unos cálculos simples, pueden demostrar que cuando el parámetro λ toma el valor $\lambda = 0{,}04166$, ambos jugadores realizan un número de tareas similar (es decir, ejecutan el 50 % de las tareas). Manteniendo este valor de λ el coste esperado para el jugador 1 será de $0{,}170163$ unidades.

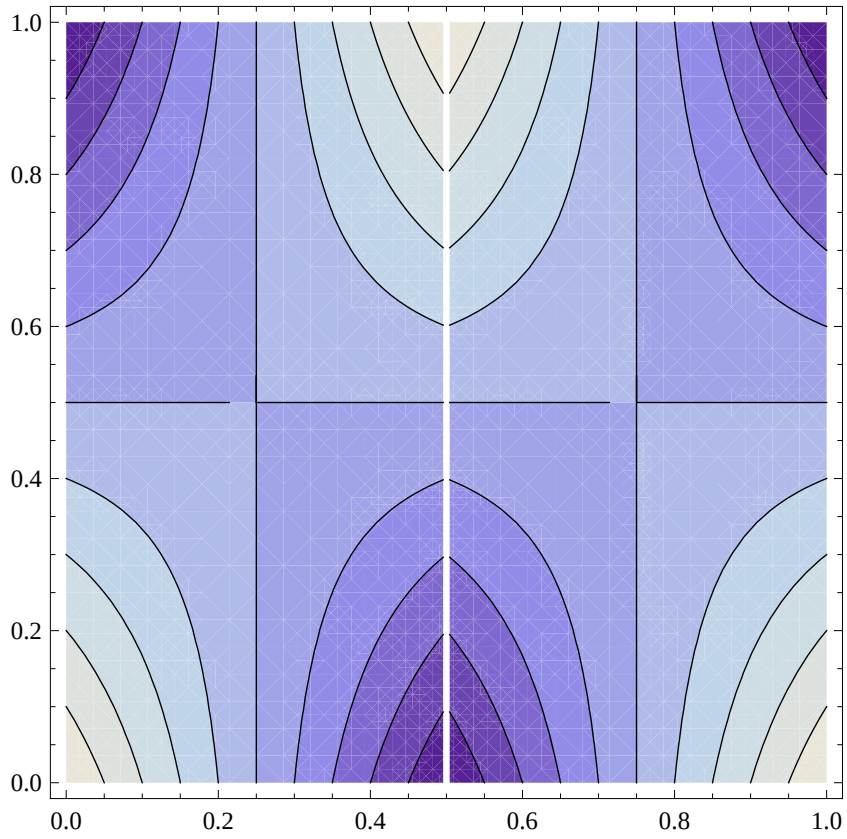

Figura 6.6: Contraejemplo 1, vista de contorno.

Supongamos ahora que el jugador 1 decide publicar sus valores según el esquema representados en las gráficas 6.7 y 6.8. En este contraejemplo concreto, el jugador 1 ha creado cuadros con una anchura igual a $0{,}04$ y centrados en los puntos $(0.6, 0.1), (0.95, 0.1), (0.6, 0.9)$ y $(0.95,\ 0.9)$, el valor que suma o resta a la densidad de probabilidad es de $0{,}35$.

Ahora el sistema observa que el jugador 1 no realiza el 50 % de las tareas supuestas, por lo que se debe modificar el valor de λ. En este ejemplo concreto, el valor aproximado de λ que reestablece la equidad de trabajo es $\lambda = 0{,}0327$, pero con ese valor resulta que el jugador 1 ahora realiza un menor esfuerzo, siendo éste igual a $0{,}168051$ unidades, frente a $0{,}170163$ que hubiera gastado de no decir mentiras. Claramente, el jugador ha salido ganando de su estrategia; y esto ha sido posible gracias

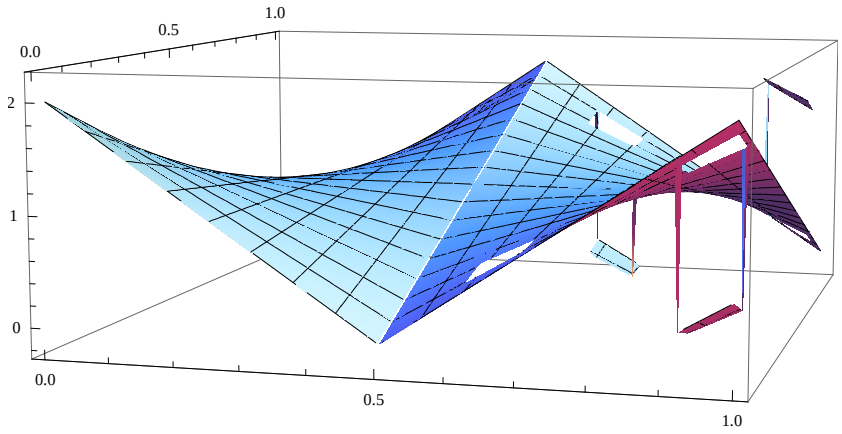

Figura 6.7: Jugador 1 mintiendo, vista 3D.

a que ha conseguido cambiar un número de tareas costosas por el mismo número de tareas pero menos costosas.

Este contraejemplo muestra claramente que, en ciertos casos, el algoritmo de QPQ con justicia en número de tareas no es strategy-proof si los jugadores pueden alterar la correlación.

6.2.3. QPQ con justicia en coste

Al igual que sucedía con la versión de QPQ anterior, existe un contraejemplo en el que un jugador puede modificar la distribución de probabilidad y ganar cuando aplicamos el algoritmo de QPQ con justicia en costes. El procedimiento sigue siendo muy parecido y se basa en el movimiento de la probabilidad haciendo "cuadros". Partimos pues, de la misma distribución del apartado anterior (6.2.1) que ya sabemos que es una distribución válida (tiene las marginales apropiadas).

Si el jugador 1 no miente, el sistema adaptará el parámetro de λ para hacer que ambos jugadores realicen un esfuerzo similar. Los cálculos son tediosos, aunque sencillos, siendo este valor igual a $\lambda = 1{,}08372$. Con este valor, ambos jugadores realizan un esfuerzo igual a $0{,}167009$.

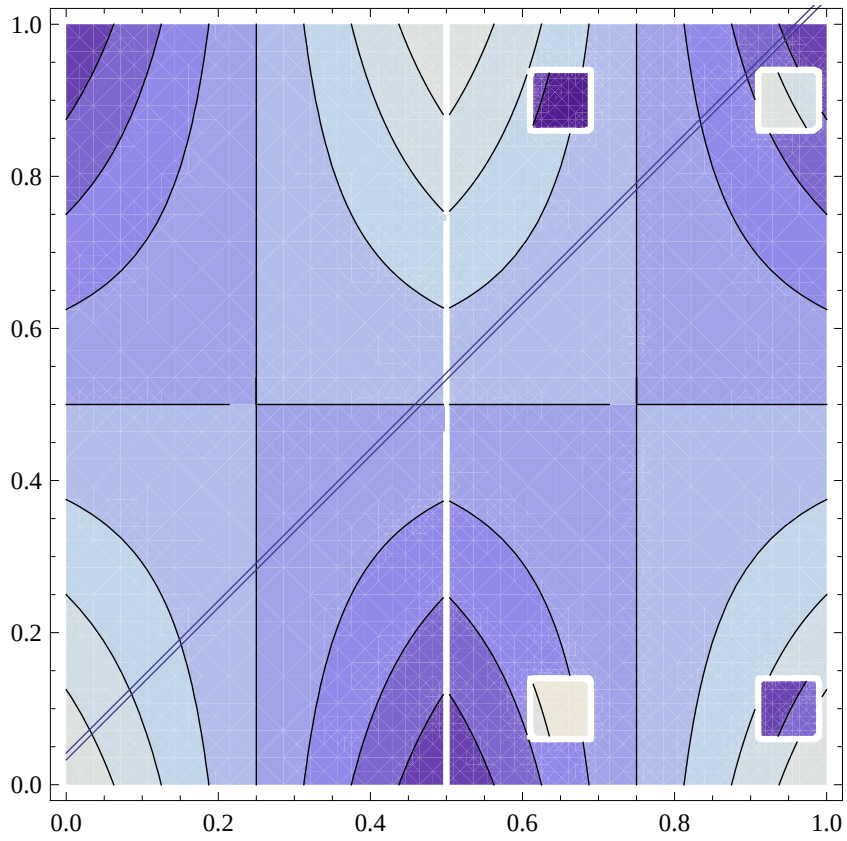

Figura 6.8: Jugador 1 mintiendo, vista de contorno.

Ahora la idea es que el jugador 1 mienta y cree los cuadros según las gráficas 6.9 y 6.10 que se corresponden a cuadros con una anchura igual a $0{,}04$ y centrados en los puntos $(0.2, 0.1), (0.6, 0.1), (0.2, 0.9)$ y $(0.6, 0.9)$, el valor que suma o resta a la densidad de probabilidad es de $0{,}2$. Nótese que en este caso, los dos cuadros superiores se encuentran en la zona del jugador 1, a diferencia del ejemplo anterior que intentaba poner un cuadro en cada zona.

Cuando el sistema se encuentra con estas declaraciones adapta y calcula λ de nuevo, intentando respetar las restricciones impuestas. El nuevo valor será, por tanto, $\lambda = 1{,}08787$. Con este valor de λ el sistema cree que el jugador 1 realiza un esfuerzo igual a $0{,}168064$, que se corresponde con el realizado por el jugador 2. Pero en realidad, el jugador 1 está realizando un esfuerzo igual a $0{,}166016$ que claramente es

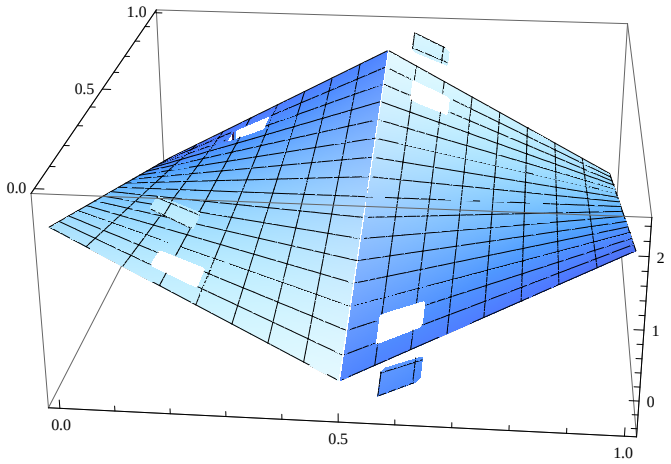

Figura 6.9: Jugador 2 mintiendo, vista 3D.

inferior al que hubiera realizado de no haber mentido ($0{,}168064$), y en consecuencia el jugador 2 realiza por lo tanto más trabajo.

Una vez más, hemos demostrado que si se ofrece la posibilidad de modificar el grado de correlación entre jugadores, éstos pueden encontrar estrategias que mejoran su utilidad.

Tanto en este caso como en el anterior, lo que está ocurriendo es que el diseñador de mecanismos no tiene herramientas en QPQ para distinguir entre una correlación válida y otra manipulada. Cuando ambas son posibles, el diseñador no puede identificarlas. Existe una circunstancia muy similar a ésta en los mecanismos bayesianos con balance de pagos ex-post. Es conocido ([KS08]) que en dichos mecanismos pueden implementar reglas de decisión social (socialmente racionales ex-ante) si la distribución de probabilidad es "identificable" y cumple con la condición de Crémer-McLean [CM88]. En el trabajo de G. Kosenok y S. Severinov se demuestra que ambas propiedades son fundamentales para la construcción de tales mecanismos. Es relevante el lema 2 de su trabajo en el que se demuestra que una distribución de probabilidad es "identificable" si cumple cierta independencia lineal entre sus probabilidades condicionales.

Aunque nuestros mecanismos QPQ son mecanismos de una naturaleza similar a

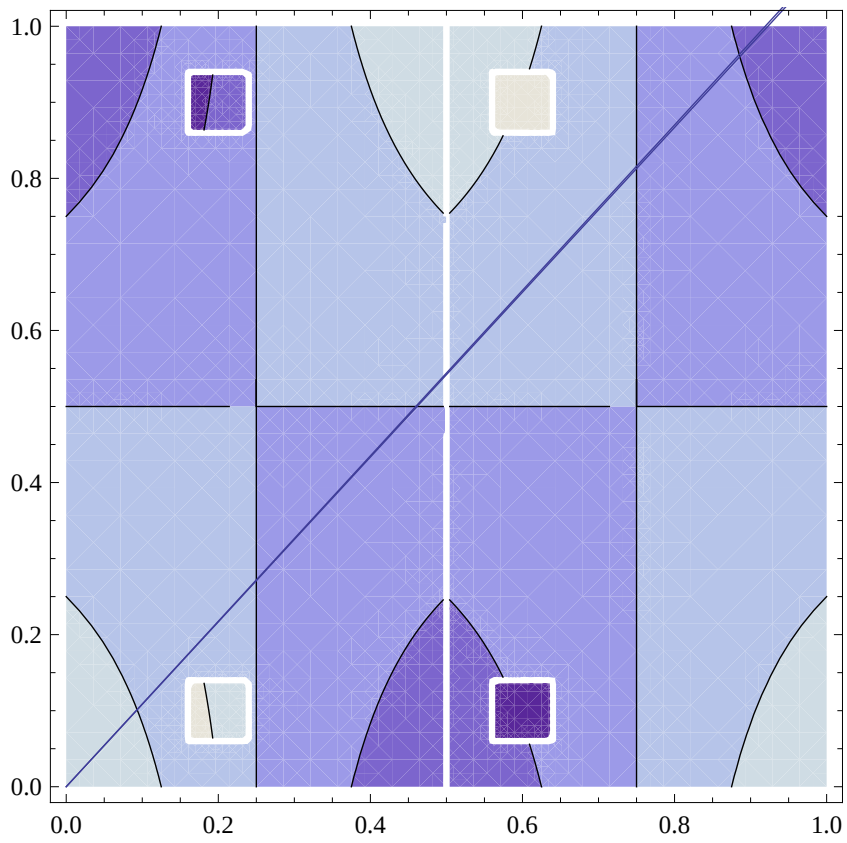

Figura 6.10: Jugador 2 mintiendo, vista de contorno.

los descritos en el trabajo de G. Kosenok y S. Severinov, creemos que la restricción de no tener pagos y de cambiar la noción de eficiencia en la regla de decisión social hace que su aplicación no sea inmediata. Se abre por tanto una línea de investigación que consideramos interesante.

6.2.4. Relación entre trabajo y esfuerzo

Si observamos los contraejemplos anteriores, lo que están haciendo los jugadores mentirosos es "acumular" o alterar el número de tareas en la parte baja de sus costes, incluso manteniendo las restricciones impuestas (ligadura del sistema y marginales). Para entender mejor lo que podría pasar, vamos a estudiar la conexión que

hay entre esos conceptos: volumen de trabajo y utilidad.

Observe que la utilidad de un jugador para QPQ Correlado se puede escribir según:

$$U_x = \int_0^1 x \int_0^{\psi(x)} f(x,y)dydx = \int_0^1 x(1-p(x))dx = \frac{1}{2} - \int_0^1 x\, p(x)dx$$

Donde $p(x)$ representa la probabilidad de que el jugador 1 ejecute la tarea cuando el valor declarado es x. La función acumulada de $p(x)$ se representa por $P(x)$.

Ahora, integrando por partes la expresión de esfuerzo se llega a:

$$\int_0^1 x\, p(x)dx = [xP(x)]_0^1 - \int_0^1 P(x)dx = P(1) - \int_0^1 P(x)dx = \int_0^1 \int_{\psi(x)}^1 f(x,y)dydx - \int_0^1 P(x)dx$$

Es decir, el volumen de trabajo medido como número de tareas es igual a la suma del esfuerzo más una cantidad que está determinada por la forma de $P(x)$:

$$\text{Trabajo}_x = \int_0^1 \int_{\psi(x)}^1 f(x,y)dydx = \int_0^1 x\, p(x)dx + \int_0^1 P(x)dx = \text{Esfuerzo}_x + \int_0^1 P(x)dx$$

De igual forma, se pueden expresar las ecuaciones del jugador 2:

$$U_y = \int_0^1 y \int_0^{\psi^{-1}(y)} f(x,y)dxdy = \int_0^1 y(1-q(y))dy = \frac{1}{2} - \int_0^1 y\, q(y)dy$$

$$\text{Trabajo}_y = \int_0^1 \int_{\psi^{-1}(y)}^1 f(x,y)dxdy = \int_0^1 y\, q(y)dy + \int_0^1 Q(y)dy = \text{Esfuerzo}_y + \int_0^1 Q(y)dy$$

Conociendo estas relaciones, ahora podemos analizar el comportamiento de QPQ cuando los jugadores modifican la correlación. El jugador podía reducir su esfuerzo mientras aumentaba $\int_0^1 P(x)dx$. En las gráficas del apartado anterior lo que pasaba era justo eso. El jugador acumulaba tareas en la zona más baja y eso lo que provoca es el aumento de $\int_0^1 P(x)dx$.

Cuando los jugadores seguían distribuciones independientes o no tenían la libertad de modificar la correlación, entonces no era posible modificar $\int_0^1 P(x)dx$ y por

lo tanto, el jugador 2 siempre tenía el mismo esfuerzo con independencia de que el jugador 1 mintiera o no.

Con un planteamiento general en el que fuera posible modificar la correlación, el diseñador de mecanismos se enfrenta al hecho de que sólo puede auditar el número de tareas (en las ecuaciones anteriores lo que hemos denominado $trabajo_x$ y $trabajo_y$) ya que la utilidad que percibe el sistema puede no ser la real. Con estos datos en mente, una posibilidad sería construir un nuevo mecanismo QPQ que tenga como ligadura dos ecuaciones: por un lado deberá fijar el número de tareas y por otro el valor de $P(\cdot)$ y $Q(\cdot)$ o expresar $\int_0^1 P(x)dx$ en función de $\int_0^1 Q(y)dy$. La idea sería fijar de tal forma el esfuerzo que cualquier estrategia de engaño provoque una transferencia de utilidad hacia el jugador honesto (medida como esperanza de trabajo normalizado). Uno de los puntos abiertos de nuestro trabajo es precisamente el estudio de cómo deberán ser esas ligaduras que garanticen la propiedad de strategy-proof en condiciones generales.

De hecho, existe un punto interesante de análisis futuro sobre qué tipo de ligaduras tienen solución para cualquier función de probabilidad f. Desgraciadamente, no todas las ligaduras imaginables tienen solución. Por ejemplo, en el apartado siguiente se presenta un posible sistema de justicia (basado en fijar cierta utilidad del contrario) y en el que se demuestra que no tiene solución.

6.2.5. Fijando la utilidad del contrario

Si repasamos las demostraciones de strategy-proof aplicadas a QPQ podemos comprobar que todas ellas utilizan la idea de que el jugador 2 no ve alterada su utilidad por el hecho de que el jugador 1 mienta. Esto, unido al hecho de que el sistema era óptimo, demostraba que el jugador 1 no podía mejorar su utilidad al mentir.

Cuando permitimos la posibilidad de que un jugador pueda alterar la correlación, surge la duda de si podemos construir un mecanismo con la seguridad de que el jugador 2 no vea afectada su utilidad si el contrario miente. Lo ideal sería encontrar un mecanismo que pueda asignar tareas a los jugadores asegurando una utilidad a

cada uno de ellos. El problema al que nos enfrentamos es la dificultad de asegurar la probabilidad de asignación de tareas a cada jugador.

Para asegurar dicha probabilidad una posibilidad es definir el mecanismo de tal forma que la probabilidad de ejecutar la tarea, dado que se ha declarado el valor de y ($Pr[d = 2 \mid y]$) esté fijada con independencia de la función de densidad y siguiendo cierto criterio de justicia. Por ejemplo, con esta idea en mente, se podría intentar asegurar el siguiente criterio:

$$Pr[d = 2 \mid y] = (n - 1)(1 - y)^{n-1}$$

Esta sería la probabilidad de que el jugador 2 ejecute una tarea cuando los jugadores son independientes. Pero por desgracia, esta solución no siempre será posible. Como demostración de esta afirmación, veamos un contraejemplo con un juego de dos jugadores. En ese caso, el sistema desea asegurar la probabilidad de que cada jugador ejecute la tarea. Para este caso, lo que estamos forzando es que:

$$Pr[d = 1 \mid x] = 1 - x$$

y que

$$Pr[d = 2 \mid y] = 1 - y$$

Si esto es cierto, debe existir una línea de corte que cumpla con esas ecuaciones. Pero eso significa que la probabilidad acumulada en un punto (x, y) de la línea de corte viene dada por la ecuación:

$$F(x, y) = \int_0^x (1 - Pr[d = 1 \mid x])dx + \int_0^y (1 - Pr[d = 2 \mid y])dy = 1/2(x^2 + y^2)$$

Y por lo tanto, para que exista la curva de corte, ésta debe ser la solución a la siguiente ecuación:

$$F(x, y) = 1/2(x^2 + y^2)$$

Por ejemplo, para $F(x, y) = x * y$ que corresponde a una distribución de jugadores independientes, la solución es $y = x$ que es la línea de corte de QPQ Básico como se esperaba.

Todas las funciones $f(x, y)$ con cierta simetría tienen la misma solución, pero es posible encontrar un contraejemplo que no tiene tal solución. Volvamos a nuestro anterior ejemplo de dos jugadores 6.2.1. Esta función tiene la siguiente acumulada:

$$\mathrm{F(x,y)} = \begin{cases} y(1-y+x(-2+x(2-2y)+3y)) & x \geq 0{,}5 \\ xy(2-y+x(-2+2y)) & x < 0{,}5 \end{cases} \tag{6.2.2}$$

En la gráfica 6.11 se representa la función $F(x, y)$ (en verde) y la función $\frac{1}{2}(x^2 + y^2)$ (en azul).

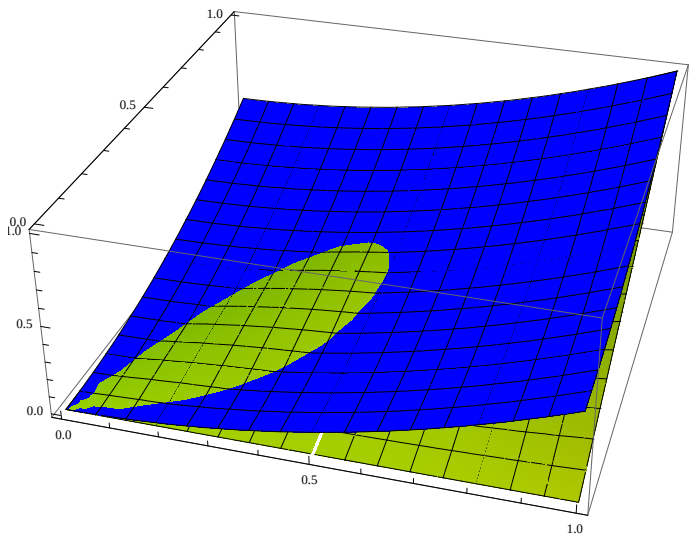

Figura 6.11: Imposibilidad de fijar la probabilidad, vista 3D.

Como se puede comprobar, no es posible trazar una línea de corte y que tenga las propiedades buscadas. Efectivamente, ambas curvas se cruzan pero el conjunto de puntos que constituyen no es una solución válida.

La conclusión es que no es posible fijar la probabilidad de ejecutar la tarea con independencia de la función de densidad y que cualquier solución que exista debe tener en cuenta tal concepto. Pero eso nos abre la puerta a otra conclusión más relevante aún. Lo que nos dice este contraejemplo es que no todas las nociones de justicia tienen solución o son implementables. Si entendemos que la justicia se puede plasmar como una restricción (ligadura) en el sistema de ecuaciones, lo que nos muestra

este contraejemplo es que hay ligaduras que no tienen solución y por lo tanto no pueden ser implementadas. La pregunta sería: ¿Qué justicias (ligaduras) pueden ser implementadas? y ¿Qué nociones de justicia pueden garantizar ciertas propiedades similares a las obtenidas hasta la fecha?

6.3. Un nuevo concepto de justicia en QPQ

En esta sección exponemos una versión de QPQ que integra los mecanismos clásicos con pagos con nuestra familia de mecanismos. La solución propuesta pasa por utilizar los mecanismos de Groves como sistema de justicia. Como ya vimos en su momento, estos mecanismos utilizan pagos como herramienta para evitar el comportamiento deshonesto.

Lo que nos proponemos realizar es la búsqueda de una construcción de QPQ basada en los mecanismos de Groves, pero sin la necesidad de utilizar pagos. Lo que deseamos en este apartado es establecer una posible conexión entre ambos mecanismos, manteniendo en la medida de lo posible las mismas propiedades de QPQ con jugadores correlados.

Hemos dividido nuestra exposición en dos apartados. En un primer momento se tratará de explorar qué tipo de solución sería viable. Esta primera parte es una reflexión sobre el camino explorado y los razonamientos que nos han guiado. En una segunda parte, formalizamos los conceptos y la solución encontrada.

En nuestro análisis nos hemos limitado a estudiar el juego de dos jugadores, ya que entendemos que podría ser generalizado a situaciones con más jugadores. Recordemos que el sistema podemos analizarlo siempre desde el punto de vista del jugador contra el sistema. Tratamos al resto de los jugadores como un jugador que agrega las preferencias de los contrincantes.

Otro aspecto que queremos destacar antes de continuar, es que seguimos suponiendo que los jugadores no pueden alterar la distribución conjunta observada por el sistema y, por tanto, tampoco sus marginales. Teniendo en cuenta que la transformada PIT nos garantiza unas marginales concretas, nos parece un supuesto razonable.

De todas formas, y cómo veremos más adelante, parece existir una relación entre la información que nos aportan las marginales y la naturaleza de un grupo de jugadores (número de jugadores, efectividad del mismo, etc.) Si el diseñador del juego tiene información sobre el grupo de jugadores (en particular del número de jugadores) se podría relajar la limitación impuesta a las marginales.

Sin más dilación, pasemos a explorar cómo debería ser la conexión entre los mecanismos de Groves y nuestros mecanismos de QPQ.

6.3.1. Análisis y exploración de la solución

Supongamos que existe un mecanismo con pagos siguiendo el esquema clásico. Recuérdese que los mecanismos de Groves tenían la forma descrita en el teorema 3.2.1 y que resumimos en este punto para mayor claridad:

Dada la función de decisión social $f(\theta) = (d^*(\theta), t_1(\theta), ..., t_n(\theta))$ y siendo ésta eficiente, entonces $f(\cdot)$ es implementable bajo estrategia dominante si satisface el siguiente esquema de transferencias:

$$t_i(\theta) = [\sum_{i \neq j} v_j(d^*(\theta), \theta_j)] + h_i(\theta_{-i}) \ \ \forall i = 1, ..., n$$

donde $h_i : \Theta_{-i} \rightarrow \mathbb{R}$ es cualquier función arbitraria que no depende de los valores declarados por el jugador i.

Es conocido que todos los mecanismos VCG o de Groves son mecanismos compatibles con incentivos Bayesianos ("Bayesian incentive compatible mechanisms"). Si se cumple que:

$$u_i(d(\theta_i, \theta_{-i}), \theta_i) \geq u_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i)$$

$\forall \hat{\theta}_i \neq \theta_i, \forall \theta_{-i} \in \Theta_{-i}$ entonces podemos afirmar que se cumplirá que el jugador obtenga mayor beneficio en esperanza al decir la verdad. Es decir, si calculamos la esperanza de la utilidad, el jugador obtiene un mayor beneficio cuando dice la verdad, independientemente de la distribución que siga en la mentira. Si para cada realización de la variable aleatoria θ_i gana diciendo la verdad, entonces ganará para cualquier

distribución de la misma. Es decir:

$$\mathbb{E}\left[u_i(d(\theta_i, \theta_{-i}), \theta_i)\right] \geq \mathbb{E}\left[u_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i)\right]$$

Para simplificar la notación hemos supuesto un mecanismo entre dos jugadores i, j en el que se ha definido un mecanismo $M_c = (d, t)$ donde d representa la decisión y t el vector de pagos, además los valores de utilidad y coste se encuentran normalizados según la PIT. Las transferencias t incluyen una posible la familia de funciones h_i. Lo que estamos buscando es un mecanismo ligado de tipo QPQ $M_l = (\psi)$ donde, para una cierta función de transferencia h y una distribución conjunta f, se busca una función de decisión ψ que sea la solución a:

$$\begin{array}{l} \mathbb{E}\left[v_i(\psi(\theta_i, \theta_{-i}), \theta_i)\right] = \mathbb{E}\left[u_i(d(\theta_i, \theta_{-i}), \theta_i)\right] + \mathbb{E}\left[h_i(\theta_{-i})\right] \\ \mathbb{E}\left[v_j(\psi(\theta_j, \theta_{-j}), \theta_j)\right] = \mathbb{E}\left[u_j(d(\theta_j, \theta_{-j}), \theta_j)\right] + \mathbb{E}\left[h_j(\theta_{-j})\right] \end{array} \tag{6.3.1}$$

En las ecuaciones anteriores se han incluido expresiones de la esperanza de la utilidad en ambos mecanismos M_c y M_l. La clave para distinguir una u otra está en que la expresión hace referencia a la función de decisión d o ψ respectivamente. Por otra parte, la esperanza de la utilidad con el mecanismo M_l viene dada por la utilidad del jugador v_i (no tiene transferencias), mientras que en el mecanismo M_c viene dada por la utilidad u_i definida como la suma de la utilidad más las transferencias: $u_i = v_i + t_i$.

Lo que estamos diciendo es que el mecanismo ligado busca la función de decisión ψ de tal forma que garantice una utilidad esperada idéntica a la podría generar el mecanismo con pagos con valores normalizados (incluyendo la función h en la transferencia). El mecanismo se adapta de tal forma que genera la misma utilidad que M_c(salvo un factor de traslación determinado por h). Imaginemos que le pedimos a los jugadores la función de distribución que van a seguir (es decir, la función de densidad $f(.,.)$). Con esta función de densidad el mecanismo calcula la esperanza de utilidad que tendrán los jugadores con el mecanismo M_c y a continuación, se calcula la forma de ψ para que los jugadores tengan la misma utilidad con M_l. Dado que el mecanismo M_l garantiza la misma utilidad que M_c, entendemos que hemos implementado el mismo concepto de justicia que tenía originalmente el mecanismo M_c. A

continuación comprobaremos si este nuevo mecanismo M_l existe y si conserva las propiedades de QPQ Correlado, entre ellas la resistencia a las mentiras.

Nótese que el factor de traslación definido por $\mathbb{E}\left[h_i(\theta_{-i})\right]$ depende de la distribución marginal del jugador contrario. Si suponemos que los jugadores no pueden alterar el nivel de correlación ni las marginales (propias o del jugador contrario), este factor no influye en la determinación de su mejor estrategia. Por lo tanto, es razonable afirmar que M_l es también resistente a las mentiras, como demostramos a continuación.

Proposición 6.3.1. *Dado un mecanismo de Groves de la forma $M_c = (d, t)$ y un mecanismo QPQ $M_l(f) = (\psi)$. Para cierta f, si existe una cierta h que hace que $M_l(f)$ respete las ecuaciones anteriores 6.3.1 para dicha función de distribución conjunta f (arbitraria pero fijada de antemano) y que maximiza el beneficio de los jugadores (la suma de las utilidades), entonces el mecanismo QPQ $M_l(f)$ es resistente a las mentiras (sujeto a que los jugadores no pueden alterar la correlación).*

Demostración. Supongamos que nuestra afirmación no fuera cierta, esto es, que el jugador i puede declarar una serie de valores falsos $\hat{\theta}_i$ que cuando son declarados por dicho jugador hace que éste gane más que diciendo la verdad θ_i. Dado que el sistema sigue percibiendo las mismas marginales y la misma distribución f, el jugador honesto sigue viendo la misma función de decisión o corte $\hat{\psi}$ pero el jugador mentiroso la ve cómo si el mecanismo la hubiera alterado. Llamamos a esta nueva función de decisión $\hat{\psi}$. Por las restricciones impuestas al mecanismo, originalmente el sistema observado debe cumplir las ecuaciones siguientes:

$$\begin{aligned}
\mathbb{E}\left[v_i(\hat{\psi}(\hat{\theta}_i, \theta_{-i}), \hat{\theta}_i)\right] &= \mathbb{E}\left[u_i(d(\hat{\theta}_i, \theta_{-i}), \hat{\theta}_i)\right] + \mathbb{E}\left[h_i(\theta_{-i})\right] \\
\mathbb{E}\left[v_j(\hat{\psi}(\theta_j, \hat{\theta}_{-j}), \theta_j)\right] &= \mathbb{E}\left[u_j(d(\theta_j, \hat{\theta}_{-j}), \theta_j)\right] + \mathbb{E}\left[h_j(\hat{\theta}_{-j})\right]
\end{aligned} \tag{6.3.2}$$

Recuérdese que se supone que el jugador no puede alterar su propia marginal o la del contrario. Por ello se cumple que:

$$\mathbb{E}\left[h_j(\hat{\theta}_{-j})\right] = \mathbb{E}\left[h_j(\theta_{-j})\right]$$

Además, el jugador honesto percibe la misma función de corte y por lo tanto se cumple que:

$$\mathbb{E}\left[v_j(\hat{\psi}(\theta_j, \hat{\theta}_{-j}), \theta_j)\right] = \mathbb{E}\left[v_j(\psi(\theta_j, \hat{\theta}_{-j}), \theta_j)\right]$$

Además, y dado que no cambia la distribución conjunta percibida por el jugador honesto se cumple que

$$f(\theta_i, \theta_j) = \hat{f}(\theta_i, \hat{\theta}_j)$$

Esto significa que el jugador honesto no percibe un cambio en su utilidad cuando el contrario miente, es decir:

$$\mathbb{E}\left[v_j(\psi(\theta_j, \theta_{-j}), \theta_j)\right] = \mathbb{E}\left[v_j(\psi(\theta_j, \hat{\theta}_{-j}), \theta_j)\right] = \mathbb{E}\left[v_j(\hat{\psi}(\theta_j, \hat{\theta}_{-j}), \theta_j)\right] \tag{6.3.3}$$

Por nuestro supuesto de que el jugador puede ganar con la mentira, lo que afirmamos es que:

$$\mathbb{E}\left[v_i(\psi(\theta_i, \theta_{-i}), \theta_i)\right] < \mathbb{E}\left[v_i(\hat{\psi}(\hat{\theta}_i, \theta_{-i}), \theta_i)\right] \tag{6.3.4}$$

Pero por las restricciones impuestas al mecanismo M_l se debe garantizar tanto 6.3.1 como 6.3.2 y por lo tanto, debe ser cierto que:

$$\mathbb{E}\left[u_i(d(\theta_i, \theta_{-i}), \theta_i)\right] + \mathbb{E}\left[h_i(\theta_{-i})\right] < \mathbb{E}\left[u_i(d(\hat{\theta}_i, \theta_{-i}), \theta_i)\right] + \mathbb{E}\left[h_i(\theta_{-i})\right]$$

Pero esto contradice que el mecanismo M_c sea strategy-proof, quedando pues demostrada nuestra proposición. □

Aunque este primer resultado puede animarnos a explorar la conexión entre mecanismos ligados y mecanismos clásicos, aún quedan pendientes muchas preguntas. Principalmente, si siempre nos será posible encontrar un mecanismo ligado para cualquier mecanismo de Groves. No menos importante es la cuestión sobre qué método deberíamos poner en marcha para determinar el mecanismo M_l.

En este punto, es relevante aclarar la razón de que se haya incluido una función h_i en la definición y construcción del mecanismo. La explicación viene por el hecho de que no habría garantía de encontrar una solución a las ecuaciones definidas por M_l, salvo que incluyamos un factor de ajuste dado por h. Por la forma en la que se construye el mecanismo ligado, la suma de las utilidades nunca puede ser mayor de un límite que está fijado por la ejecución de tareas y, por lo tanto, no todos los valores son posibles. En el mecanismo ligado no hay inyección de dinero por el sistema.

Es evidente que, si no incluimos h, y dependiendo de la función de transferencia t puede o no existir una solución a las ecuaciones M_l. Basta con buscar funciones de transferencia suficientemente altas (o bajas) para que no sea posible encontrar un reparto de tareas cuya utilidad sume las cantidades marcadas por t. Ahora bien, al incluir un factor definido por las funciones h es posible garantizar una solución a M_l ya que "trasladamos" de forma apropiada la utilidad hasta un punto en el que exista una solución.

Visto todo lo anterior, el siguiente problema es decidir cuál es la mejor forma de construir h_i para garantizar una solución. Para nuestro estudio, reduciremos el problema a nuestra visión, ya habitual, de dos jugadores: un jugador siempre juega contra el jugador agregado. Por otra parte, y a modo de ejemplo, vamos a explorar un subconjunto de la familia de funciones h. En concreto nos concentraremos en las funciones h lineales. Como veremos en breve, estas funciones nos permitirán establecer relaciones simples entre las marginales de los jugadores, lo que nos permitirá simplificar el sistema a optimizar.

En general, no podemos suponer que la esperanza de h_1 sea igual a la esperanza de h_2, ya que ésta última representa al sistema agregado. Si los jugadores son independientes, es de esperar que la esperanza de estas funciones sea función únicamente

de la marginal del jugador y por tanto incluye información del número de jugadores.

Es decir, si los jugadores son independientes, se puede expresar $\mathbb{E}[h_2]$ como una función de $\mathbb{E}[h_1]$ y del número de jugadores: $\mathbb{E}[h_2] = (n-1)\mathbb{E}[h_1]$.

Si suponemos que ambos jugadores son independientes y simétricos, entendiendo como tal el hecho de que tienen marginales idénticas, entonces significa que $\mathbb{E}[h_i(\theta_{-i})] = \mathbb{E}[h_j(\theta_{-j})]$. Este factor se puede tratar como una constante para el sistema de ecuaciones. Siendo una constante que no depende de la estrategia de los jugadores y por lo tanto se puede conocer a priori dado el mecanismo M_c. La solución a las ecuaciones anteriores debería conducirnos de nuevo, a nuestra solución de QPQ con jugadores independientes. Veamos brevemente el razonamiento de tal afirmación.

En nuestro caso, y suponiendo que son independientes y simétricos (dos jugadores independientes con marginales uniformes) el sistema de ecuaciones que define M_l se puede reducir a:

$$\begin{array}{l} \mathbb{E}[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)] = \mathbb{E}[u_i(d(\theta_i,\theta_{-i}),\theta_i)] + cte \\ \mathbb{E}[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)] = \mathbb{E}[u_j(d(\theta_j,\theta_{-j}),\theta_j)] + cte \end{array} \tag{6.3.5}$$

Obsérvese que el sistema se puede reescribir de la siguiente forma:

$$\begin{array}{l} \mathbb{E}[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)] - \mathbb{E}[u_i(d(\theta_i,\theta_{-i}),\theta_i)] = \\ \mathbb{E}[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)] - \mathbb{E}[u_j(d(\theta_j,\theta_{-j}),\theta_j)] \end{array} \tag{6.3.6}$$

Con este planteamiento, la siguiente cuestión es la relativa a la forma que pueden tener las soluciones ψ de tales mecanismos. A la luz de lo analizado en versiones anteriores, suponemos que la función de decisión que se puede expresar como una función de corte determinista, y de hecho, si tal función existe, la función ψ debe ser la solución al siguiente sistema de ecuaciones:

$$\begin{array}{l} \max{}_{\psi}(\mathbb{E}[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)] + \mathbb{E}[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)]) \\ \\ \text{sujeto a,} \\ \\ \mathbb{E}[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)] - \mathbb{E}[u_i(d(\theta_i,\theta_{-i}),\theta_i)] = \\ \mathbb{E}[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)] - \mathbb{E}[u_j(d(\theta_j,\theta_{-j}),\theta_j)] \end{array} \tag{6.3.7}$$

Utilizando técnicas similares a las vistas en los capítulos anteriores, podemos

decir que maximizar la utilidad es minimizar el coste, sujeto a la restricción anterior, y por lo tanto, lo que estamos buscando es una solución a:

$$\begin{aligned} &\min_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right) \\ &\text{sujeto a,} \\ &\mathbb{E}\left[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)\right] - \mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right] = \\ &\mathbb{E}\left[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)\right] - \mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right] \end{aligned} \tag{6.3.8}$$

y trabajando en la expresión de la ligadura, este sistema es similar a:

$$\begin{aligned} &\min_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right) \\ &\text{sujeto a,} \\ &\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx - \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy = \\ &\mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right] - \mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right] \end{aligned} \tag{6.3.9}$$

Fijado un mecanismo VCG y fijada la función f, la cantidad $\mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right] - \mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right]$ es una constante para el algoritmo, por lo que podemos reescribir el sistema anterior según:

$$\begin{aligned} &\min_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right) \\ &\text{sujeto a,} \\ &\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx = \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy + \beta \end{aligned} \tag{6.3.10}$$

Este sistema de ecuaciones es muy similar al que vimos cuando tratamos de encontrar un QPQ con justicia en el reparto equitativo de costes. La principal diferencia es que ahora, el parámetro $\delta = 1$. La solución a este sistema ya la conocemos y es igual a la recta $y = x$ como ya se esperaba.

Estos resultados previos nos animan a explorar soluciones algo más complejas. Supongamos ahora que los dos jugadores no son independientes ni tienen marginales

idénticas. En ese caso las esperanzas de h_1 y h_2 serán diferentes. De nuevo, se trata de buscar una solución a las ecuaciones que relacionan el mecanismo ligado con el mecanismo VCG. Es decir, intentemos encontrar la función de corte ψ que sea solución a tal sistema. Suponiendo y restringiendo nuestra atención a una función determinista de corte, podemos pensar que la solución a tal sistema se puede expresar como un problema de optimización:

$$\begin{aligned} &\max{}_{\psi}(\mathbb{E}\left[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)\right]+\mathbb{E}\left[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)\right]) \\ &\text{sujeto a,} \\ &\mathbb{E}\left[v_i(\psi(\theta_i,\theta_{-i}),\theta_i)\right]=\mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right]+\mathbb{E}\left[h_i(\theta_{-i})\right] \\ &\mathbb{E}\left[v_j(\psi(\theta_j,\theta_{-j}),\theta_j)\right]=\mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right]+\mathbb{E}\left[h_j(\theta_{-j})\right] \end{aligned} \tag{6.3.11}$$

Nótese que la esperanza de las funciones h dependen de las marginales de cada jugador y del grado de correlación que existe entre ellos. Seguimos suponiendo que los jugadores no pueden alterar esos parámetros y que son conocidos por el diseñador del mecanismo. Dado que éste tiene una amplia libertad para la elección de h, podemos pensar en definir una familia de funciones que aporten información sobre la estructura de los jugadores (número de los mismos y grado de correlación). Nótese que la esperanza de dichas funciones estará determinada por la función de decisión original del mecanismo VCG de referencia y, por lo tanto, el grado de correlación se podría "medir" por la distribución de probabilidad que hay a cada lado de la recta $y = x$. Más adelante veremos que se podría definir una noción de "número virtual" de jugadores como un indicador de la potencia que tiene un jugador agregado.

Restringimos nuestra atención a la familia de funciones h_i, tales que para toda función de densidad f, la aplicación del mecanismo VCG sobre f hace que $\mathbb{E}\left[h_i(\theta_{-i})\right] = \frac{1}{\delta}\mathbb{E}\left[h_j(\theta_{-j})\right]$. Y donde δ es un factor que representa la potencia de un jugador con respecto al agregado. En caso de jugadores independientes, se podría definir según $\delta = n - 1$. Estas funciones siempre existen ya que basta con definirla utilizando la esperanza del esfuerzo que tendría el jugador aislado escalado por algún parámetro. Es decir, podríamos definirla según: $\mathbb{E}\left[h_i(\theta_{-i})\right] = \alpha\mathbb{E}\left[\theta_{-i}\right]$.

Con estas consideraciones, y equipados con la definición anterior, procederemos a la solución del sistema anterior:

$$\begin{array}{l} \min\limits_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right) \\ \\ \text{sujeto a,} \\ \\ \int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx = \mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right] + \mathbb{E}\left[h_i(\theta_{-i})\right] \\ \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy = \mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right] + \delta\mathbb{E}\left[h_i(\theta_{-i})\right] \end{array} \tag{6.3.12}$$

Despejando $\mathbb{E}\left[h_i(\theta_{-i})\right]$, el sistema queda reducido a:

$$\begin{array}{l} \min\limits_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right) \\ \\ \text{sujeto a,} \\ \\ \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy - \mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right] = \\ \delta(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx - \mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right]) \end{array} \tag{6.3.13}$$

Simplificando y agrupando las constantes, llegamos al siguiente sistema:

$$\begin{array}{l} \min\limits_{\psi} \left(\int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx + \int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy\right) \\ \\ \text{sujeto a,} \\ \\ \int_0^1 x \int_{\psi(x)}^1 f(x,y)dy\, dx = \frac{1}{\delta}\int_0^1 y \int_{\psi^{-1}(y)}^1 f(x,y)dx\, dy + \beta \end{array} \tag{6.3.14}$$

En esta última ecuación, el parámetro β es un valor que aglutina las esperanzas de utilidad de los jugadores con el mecanimo VCG y que es el resultado de operar las expresiones anteriores. Es decir:

$$\beta = -\frac{1}{\delta}\mathbb{E}\left[u_j(d(\theta_j,\theta_{-j}),\theta_j)\right] + \mathbb{E}\left[u_i(d(\theta_i,\theta_{-i}),\theta_i)\right])$$

Este sistema de ecuaciones a optimizar tiene una solución ya conocida. Las ecuaciones anteriores son similares al sistema de ecuaciones de QPQ con justicia de trabajo. La solución es, por tanto, una recta $y = \lambda x$. Donde λ es el nuevo parámetro de Lagrange que debe ser calculado para respetar la ligadura.

La principal diferencia es que ahora aparecen dos parámetros de ajuste que debe establecer el diseñador del mecanismo: el parámetro con información sobre el número de jugadores δ y un término nuevo β que "regula" la justicia conforme a lo que se podría esperar si se aplicara un mecanismo VCG. Obsérvese que, dado un mecanismo VCG, y ante la declaración de los jugadores ambos parámetros están fijados (δ y β). De estos parámetros, únicamente β es susceptible de ser alterado por la estrategia de los jugadores, pero por lo que hemos visto anteriormente, el jugador tendrá como estrategia óptima declarar la verdad, independientemente de lo que decida su contrario.

6.3.2. Sobre el cálculo de la función de corte

Al igual que sucedía con las versiones anteriores de QPQ, nos enfrentamos al cálculo de la línea $y = \lambda x$ que respeta la restricción del sistema. De nuevo, nos encontramos el problema de que no siempre será posible obtener esa solución de forma analítica. Incluso si el diseñador de mecanismo tiene información sobre la función probabilidad f, es posible que ésta no tenga un tratamiento analítico sencillo. Además, es de suponer que esa información no sea conocida a priori, y que conforme evolucione la interacción entre jugadores se pueda ir generando la distribución empírica. En ambos casos, tanto si la información es conocida o empírica, el cálculo de la recta puede requerir métodos de aproximación, por lo que puede ser de utilidad recurrir de nuevo a métodos de aproximación estocástica, y en concreto al algoritmo de Robbins-Monro.

El algoritmo deberá, por tanto, estimar el nuevo parámetro de Lagrange λ según se obtenga información de los jugadores. Cuando se dispongan de un número suficiente de muestras, es posible estimar los dos parámetros del mecanismo: δ y β. El primero es una relación entre las marginales de los jugadores y el segundo es una ex-

presión que relaciona las utilidades que tendrían los jugadores si estuvieran jugando a un mecanismo clásico VCG. Ninguno de esos valores depende del valor de λ y es de suponer que, conforme el número de muestras crezca, ambos parámetros converjan hacia valores concretos.

En este caso, la ligadura del sistema viene dada por la diferencia entre esfuerzos. Denominamos por $B(.)$ a la función que dado un valor de λ nos devuelve la diferencia entre esfuerzos. Es decir, $B[\lambda] = Esfuerzo_x[\lambda] - \frac{1}{\delta} Esfuerzo_y[\lambda]$. Suponiendo las condiciones apropiadas de continuidad y que la estimación $B_k = B(\lambda)$ cumple con los requisitos de Robbins-Monro, podemos volver a aplicar la aproximación estocástica definida por:

$$\lambda_{k+1} = \lambda_k + \epsilon_k(B_k - \beta)$$

Con esto en mente, y aplicando las herramientas de los algoritmos previos de QPQ, nuestro nuevo algoritmo de QPQ quedaría definido.

Tendríamos pendiente para futuras investigaciones estudiar las condiciones en las que el algoritmo converge y analizar mediante simulaciones su comportamiento. Además, hemos reducido el estudio a una familia muy concreta de funciones h, y queda pendiente el estudio de si sería posible construir el mecanismo ligado con otra familia de funciones.

6.4. Número de jugadores

Hasta ahora hemos dado por supuesto que conocemos el número de jugadores (que hemos denominado n a lo largo del presente trabajo). Si queremos aplicar el algoritmo en ciertos contextos, dicha suposición podría ser difícil de mantener. Esto es debido principalmente a los siguientes aspectos:

1. Cuando el número de jugadores es muy alto, el volumen de mensajes entre los nodos sería tan elevado que haría impracticable el algoritmo. La formulación actual del algoritmo presupone que todos los nodos reciben mensajes de todos los demás nodos y aunque la complejidad del sistema es casi lineal, si

aumentamos considerablemente el volumen de nodos, el rendimiento se vería perjudicado.

2. Los jugadores pueden agruparse entre ellos haciendo grupos ("clusters"), definir un representante ante el sistema y repartirse el trabajo que el sistema le asigne. De esta forma, el grupo podría llegar a trabajar menos que el resto.

3. Pueden entrar y salir del sistema. Esto es un hecho muy frecuente en las redes actuales. Este problema se complica si además suponemos que la identidad de los jugadores puede cambiar si deciden salir y volver a entrar al sistema. Siempre pueden presentarse con una nueva identidad.

La solución al primer punto es que el sistema pudiera organizarse en agrupaciones, posiblemente de forma jerárquica. Esta organización podría ser controlada por el algoritmo o completamente anárquica dejando que los propios nodos se organicen. Una posibilidad sería contemplar que un mismo jugador participe en varias agrupaciones (de forma intencionada o accidental). En cualquier caso, llegamos al punto que describe el segundo problema; es decir, que los nodos se agrupen y no declaren el número real de jugadores.

En los casos más sencillos, una posibilidad sería utilizar el número de mensajes recibidos/emitidos como indicador del número de jugadores. Esto supondría una limitación muy seria. Puede estar sujeto a engaños, pérdidas de mensajes o simplemente puede variar con el tiempo.

Veamos con detenimiento cada uno de estos problemas y la solución propuesta.

6.4.1. Número de jugadores variable en el tiempo

Cuando suponemos que los jugadores pueden entrar y salir del sistema nos enfrentamos a un problema nuevo. En esta situación el valor de n puede variar según la ronda. Proponemos varias soluciones que podrían aplicarse:

Si suponemos que es conocido el valor de n para cada ronda, es suficiente aplicar los algoritmos QPQ en cada ronda según el valor de n en esa ronda. Por una parte,

si suponemos que n no se puede determinar podemos tratar a n como una variable aleatoria y utilizar el valor medio como referencia al algoritmo. En QPQ General se han utilizado valores de corte (λ o δ) que hemos supuesto ligados a n pero el algoritmo sigue siendo válido para valores arbitrarios, siempre que la carga se reparta de forma equitativa.

Cuando el problema se limita a la variación del número de jugadores en el tiempo, el algoritmo puede adaptar su λ incluso si cambian las circunstancias. Según la literatura estudiada, la solución es hacer que el valor de ϵ_k utilizado en el algoritmo de Robbins-Monro no llegue a ser cero nunca e incluso hacer que sea una constante muy pequeña [KY97]. Unas simulaciones preliminares parecen indicar que los resultados serían bastante positivos. No se han observado grandes variaciones frente al algoritmo con número fijo de jugadores. Es decir, todos los jugadores obtenían niveles de esfuerzo similares al que hubieran obtenido si hubieran jugado contra un sistema donde el número de jugadores era fijo e igual al número medio de jugadores.

6.4.2. Elevado número de jugadores

En un sistema en el que el número de jugadores sea muy alto nos enfrentamos a un problema añadido de rendimiento. El sistema debe recibir, procesar y calcular un alto número de mensajes.

Obsérvese que QPQ podría tener una complejidad lineal. Es decir, el tiempo de procesamiento es una función de n. Claramente influye el tamaño del histórico o el tiempo de computo de operaciones básicas como el test de Kolmogorov, la transformada PIT, etc. Pero suponiendo que estas operaciones son atómicas es fácil comprobar que el algoritmo es lineal ya que en realidad lo que hace el algoritmo es recibir, transformar y comparar $n - 1$ valores con el propio valor. Aunque la complejidad y por tanto su rendimiento puede parecer muy razonable, no creemos que, a efectos prácticos, fuera viable (retardos en las comunicaciones, pérdida de mensajes, falta de coherencia, etc.).

La solución viene de nuestro concepto de jugador agregado. Dado que los ju-

gadores siempre juegan contra un jugador agregado, siempre es posible crear agrupaciones jerárquicas de nodos y que, uno o varios representantes puedan agregar a los demás. Como se ha demostrado anteriormente, dos agrupaciones ("clusters") de jugadores independientes siempre tienen una distribución Beta, en el que los valores de la misma son funciones del tamaño de dichas agrupaciones.

La idea es por lo tanto bastante simple, basta que varios jugadores se agrupen y definan a uno de ellos como su representante. Este jugador actuará como el "jugador agregado" frente al sistema. Independientemente del tamaño de la agrupación y de la forma práctica en el que se realice, las propiedades del algoritmo QPQ siguen siendo válidas. Aunque la principal diferencia es que la complejidad del mismo se vería reducida.

6.4.3. Número de jugadores virtuales

Uno de los problemas que ha quedado pendiente de estudio futuro es el que se produce cuando no se conoce el número exacto de jugadores. Suponer que conocemos el número de jugadores puede parecer una condición razonable en ciertos contextos, pero debemos tener en cuenta que los jugadores pueden "ocultarse" detrás de otros. En realidad, esta circunstancia no es visible para el diseñador de mecanismos. Dado que estamos trabajando con la PIT e independientemente del número de jugadores, el "representante" siempre declara la distribución uniforme y por lo tanto el sistema puede no conocer el número real de jugadores. Incluso, si eliminamos la restricción de que los jugadores publiquen una uniforme, el sistema no podría detectar el número real de jugadores únicamente basándose en los valores declarados(o en su distribución). Un ejemplo ilustrativo sería el siguiente. Supongamos que hay dos o más jugadores fuertemente correlados (dicen siempre el mismo valor). Si estos jugadores se agrupan, sería inviable saber el número exacto de jugadores y de hecho, se confundirían con un jugador único.

Este problema se transforma en otro bien distinto cuando permitimos y legalizamos la agrupación de jugadores según lo descrito en el apartado anterior. En ese

caso, los jugadores agregados declaran distribuciones diferentes a la uniforme (por ejemplo una Beta si fueran independientes) y ese momento el sistema puede utilizar dicha distribución para estimar el número de jugadores reales.

Una posibilidad que nos parece muy interesante, es definir el número de jugadores según la capacidad de una agrupación para ejecutar tareas. Nótese que la eficiencia del sistema viene determinada por el número de jugadores. A más jugadores, más eficiente es el sistema y menor es el coste de ejecutar la totalidad de las tarea. Cuando los jugadores son independientes, existe una relación muy simple entre el número de jugadores y el esfuerzo total del sistema en ejecutar las tareas, que viene expresada por la siguiente ecuación:

$$n = \frac{1}{\sum_i \text{Cost}} - 1$$

Donde n representa nuestro número de jugadores y $Cost$ representa el esfuerzo total en ejecutar la totalidad de las tareas, calculando el esfuerzo sobre el coste mínimo declarado.

En la tabla 6.1 se puede observar dicha relación. Un jugador 1 participa en un sistema en el que el jugador 2 representa a un determinado número de jugadores agregados $(n-1)$.

Por lo tanto, una posibilidad sería definir el concepto de "*número de jugadores virtuales*" según la capacidad de producción que tiene el sistema. Para ello se propone eliminar la limitación de que los jugadores declaren sus valores según una uniforme, aunque manteniendo la restricción de que sus valores se encuentren en el rango $[0, 1]$. Cada jugador representa a un número de jugadores igual a su potencial de trabajo, según la fórmula anterior.

Para aclarar este punto, veamos una aplicación de este concepto sobre un ejemplo ya conocido. Supongamos que tenemos una combinación de jugadores según la siguiente distribución $f(x, y)$:

$$f(x,y) = \begin{cases} 0{,}3 + (-1+n)(1-y)^{-2+n} & \text{si c1} \\ -0{,}3 + (-1+n)(1-y)^{-2+n} & \text{si c2} \\ (-1+n)(1-y)^{-2+n} & \text{en el resto} \end{cases}$$

Jugadores reales	Esf. jugador 1	Esf. jugador 2	Esf. Total	Jug. virtuales
2 Jugadores:	0,1666667	0,166667	0,333333	2
3 Jugadores:	0,0833333	0,166667	0,25	3
4 Jugadores:	0,0500000	0,150000	0,2	4
5 Jugadores:	0,0333333	0,133333	0,166667	5
6 Jugadores:	0,0238095	0,119048	0,142857	6
7 Jugadores:	0,0178571	0,107143	0,125	7
8 Jugadores:	0,0138889	0,097222	0,111111	8

Cuadro 6.1: Esfuerzos de jugadores agrupados

Donde el caso $c1)$ se aplica cuando

$$(|-0{,}3+x|<0{,}04 \wedge |-0{,}1+y|<0{,}04)\& \vee (|-0{,}95+x|<0{,}04 \wedge |-0{,}5+y|<0{,}04)$$

La condición $c2)$ es:

$$(|-0{,}95+x|<0{,}04 \wedge |-0{,}1+y|<0{,}04) \vee (|-0{,}3+x|<0{,}04 \wedge |-0{,}5+y|<0{,}04)$$

Que gráficamente tiene la forma representada por la gráfica 6.12.

Es fácil comprobar que, en este caso, un jugador tiene una distribución marginal idéntica a la uniforme, mientras que el otro parece representar a una agrupación de $n-1$ jugadores. Como se ha visto en los capítulos anteriores, el jugador agregado tiene una distribución Beta. Es decir la marginal del segundo jugador es:

$$f_y(y) = (n-1)(1-y)^{(n-2)}$$

Para este ejemplo, se puede demostrar que el esfuerzo que debería realizar el jugador 1 si estuviera aislado sería igual a $1/2$, mientras que el segundo jugador tendría que realizar un esfuerzo en solitario igual a $1/n$, como era de esperar según

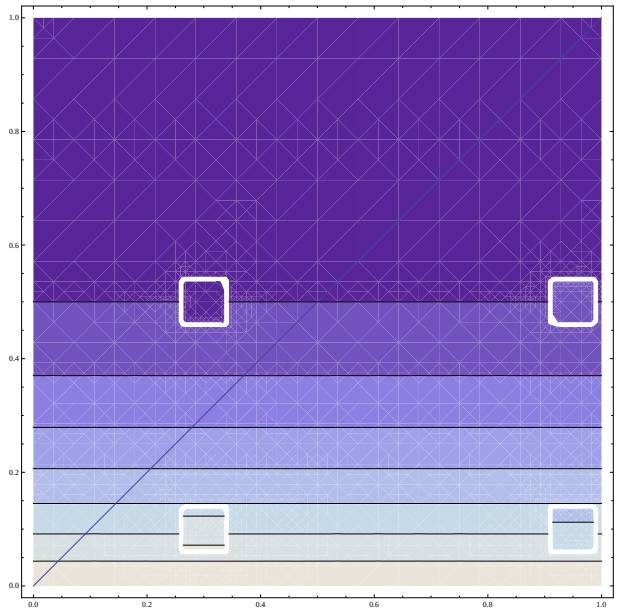

Figura 6.12: Jugadores correlados.

sus marginales. Es decir, el primer jugador representa a un único *jugador virtual*, mientras que el segundo representa a $n-1$ *jugadores virtuales*. Pero cuando trabajan en combinación, el sistema visto como un jugador agregado no trabaja como si fueran $n = (n-1)+1$ jugadores, ya que al existir una correlación entre ellos se altera el rendimiento. Para nuestro caso, el número de *jugadores virtuales* es algo inferior al valor teórico de n.

En la tabla 6.2 se representa el esfuerzo conjunto de ambos jugadores y el número de *jugadores virtuales* a los que representan:

Otra posibilidad es cambiar el sentido de la correlación, por ejemplo con:

$$f(x,y) = \begin{cases} -0{,}3 + (-1+n)(1-y)^{-2+n} & \text{si c1} \\ +0{,}3 + (-1+n)(1-y)^{-2+n} & \text{si c2} \\ (-1+n)(1-y)^{-2+n} & \text{en caso contrario} \end{cases}$$

De nuevo las condiciones que determinan la función $f(x,y)$ son para :

$$(|-0{,}3+x| < 0{,}04 \;\&\&\; |-0{,}1+y| < 0{,}04) \;\|\; (|-0{,}95+x| < 0{,}04 \;\&\&\; |-0{,}5+y| < 0{,}04)$$

Jugadores reales	Esf. jugador 1	Esf. jugador 2	Esf. Total	Jug. virtuales
2 Jugadores:	0,166091	0,167627	0,333717	1,99655
3 Jugadores:	0,0827573	0,167627	0,250384	2,99387
4 Jugadores:	0,049424	0,15096	0,200384	3,99042
5 Jugadores:	0,0327573	0,134293	0,167051	4,98621
6 Jugadores:	0,0232335	0,120008	0,143241	5,98123
7 Jugadores:	0,0172811	0,108103	0,125384	6,9755
8 Jugadores:	0,0133129	0,0981822	0,111495	7,969

Cuadro 6.2: Esfuerzos de jugadores correlados agrupados

para $c1$), y

$$(|-0{,}95+x|<0{,}04 \;\&\&\; |-0{,}1+y|<0{,}04) \,\|\, (|-0{,}3+x|<0{,}04 \;\&\&\; |-0{,}5+y|<0{,}04)$$

para $c2$).

Con este ejemplo, se puede comprobar que el número virtual de jugadores habrá cambiado.

Por lo que hemos visto, sería posible relacionar la capacidad de trabajos que tiene el sistema con el número de jugadores que el sistema "observa". Esa relación nos ayuda a definir un *número virtual de jugadores* como el número de jugadores que tendría un sistema formado por jugadores independientes y que dediquen el mismo coste a ejecutar las tareas que si estuvieran aislados. Es el resultado de observar el coste que tienen que realizar en esperanza y calcular el número de jugadores virtuales como $\hat{n}_s = \frac{1}{Cost_s} - 1$.

Relacionado con este tema, existe también un área por explorar y que el autor considera relevante. Aunque el mecanismo utilice esta noción de jugadores virtuales, sigue sin estar claro si la creación de subgrupos o clusters dentro del sistema

produciría los mismos resultados globales. Para entender este punto, supongamos un sistema que tenga tres jugadores, todos ellos representan una posible agrupación. El sistema observa para cada uno de ellos el correspondiente "numero virtual" ($\hat{n}_i$), construyendo por tanto el vector $(\hat{n}_1, \hat{n}_2, \hat{n}_3)$. Si ahora dos de los jugadores se agrupan y ocultan su identidad, el sistema trabajaría sobre un nuevo vector de dos jugadores de la forma $(\hat{n}_1^{'}, \hat{n}_a^{'})$. La pregunta sería bajo qué condiciones se puede afirmar que el reparto de tareas o de esfuerzo sería el mismo utilizando $\hat{n}_1$ o $\hat{n}_1^{'}$ y si esto es cierto para cualquier agrupación posible dentro del mecanismo. Consideramos relevante esta pregunta, ya que todos los mecanismos que respeten esa propiedad, serían "justos" en cualquiera de las connotaciones anteriormente vistas, incluso si los jugadores se agrupan ocultando su identidad.

Con todo este material, consideramos interesante añadir una nueva propiedad de los mecanismos. Un mecanismo QPQ es *resistente a las agrupaciones* si para cualquier agrupación posible, cada uno de los jugadores realiza el mismo volumen de trabajo (medido en coste o en número de tareas).

Revisemos de nuevo el ejemplo anterior en el que el sistema asigna tareas a la agrupación $(\hat{n}_1^{'}, \hat{n}_a^{'})$. Si basamos la noción de justicia que realiza el reparto de las tareas según la proporción habitual: $\frac{\hat{n}_1^{'}}{\hat{n}_1^{'}+\hat{n}_a^{'}}$ o $\frac{\hat{n}_j}{\hat{n}_1^{'}+\hat{n}_a^{'}}$, veremos que tenemos problemas al desagregar el sistema. Si ahora deseamos que todos nuestros jugadores participen en el juego, este reparto no tendría sentido. Para que el jugador 1 tenga idéntica proporción de trabajo, sería necesario que $\hat{n}_a^{'} = \hat{n}_2 + \hat{n}_3$. Pero esto, en general, no será cierto.

En su versión original, los mecanismos QPQ con justicia parecen no ser resistentes a las agrupaciones, pero no está claro si es posible modificar de alguna forma la noción de justicia y cumplir con dicha propiedad. Lo que proponemos es estudiar si cambiar la noción de justicia y utilizar el número virtual de jugadores pudiera ser una posible solución a este problema. En lugar de repartir las tareas o el esfuerzo basándonos en el número real de jugadores, podemos aplicar una proporción que refleje el número virtual de los mismos. Por ejemplo, el volumen de tareas se podría repartir según el porcentaje en el que participa cada jugador en el sistema. Algo simi-

lar se podría plantear para un reparto equitativo de esfuerzo.

Una forma de enfocar el problema sería utilizando métodos de reparto de coste para definir el concepto de número de jugadores virtuales. Un ejemplo interesante sería utilizar un método de reparto basado en el valor de Shapley.

Recordemos que el valor de Shapley fue introducido inicialmente por Lloyd Shapley [Sha53] como un sistema para el reparto justo entre jugadores cuando existen coaliciones. Partiendo de un conjunto de propiedades deseables, Shapley estableció el sistema de reparto que lleva su nombre. Dichas propiedades son:

1. *"Individual fairness"*: Cada jugador, cuando participa del reparto del coste, obtiene una utilidad mayor o igual que si no hubiera participado.

2. *"Efficiency"*: Todo el coste se reparte sin necesidad de añadir o retirar fondos.

3. *"Symmetry"*: Si dos jugadores son equivalentes, en el sentido de que aportan por igual a cualquier coalición, entonces ambos tienen igual reparto de coste.

4. *"Additivity"*: Si combinamos dos coaliciones, el reparto que tiene cada jugador esa coalición será la suma de lo que le hubiera correspondido en cada coalición por separado.

5. *"Null player"*. Un jugador que no aporta valor a una coalición no paga nada.

Además, Shapley demostró que su sistema era el único sistema que respeta todas las propiedades anteriores, convirtiéndolo en uno de los sistemas más utilizados para el reparto "justo" de los costes.

Para entender la definición y el funcionamiento del valor de Shapley, tenemos que definir un concepto llamado el "valor de una coalición". Este valor indica cuánto "vale" la agrupación de un subconjunto de jugadores. El valor de Shapley se crea considerando todas las posibles combinaciones de coaliciones y repartiendo el coste haciendo la media de lo que aporta un jugador en cada una de esas coaliciones. Este valor se define para un nodo i que pertenece a una coalición N según:

$$\Phi_i(N) = \sum_{S \subseteq N_{-i}} \frac{|S|!(|N| - |S| - 1)!}{|N|!}[V(S \cup \{i\}) - V(S)] \tag{6.4.1}$$

En nuestro caso, proponemos definir el valor de la coalición como el número virtual de jugadores definido anteriormente:

$$V(S) = \frac{1}{Cost_S} - 1. \tag{6.4.2}$$

Siendo $Cost_S$ el coste de ejecutar las tareas utilizando para ello el mínimo de los valores declarados por la agrupación S.

Para entender el funcionamiento del valor de Shapley aplicado al número virtual de jugadores, proponemos un ejemplo aclaratorio. En la tabla 6.3 se muestran valores de jugadores virtuales para cada una de las coaliciones posibles. Los datos son totalmente ficticios, pero esperamos que con ellos podamos explicar el funcionamiento del valor de Shapley.

Con estos valores es fácil calcular el valor de Shapley. A modo de ejemplo, se incluye el detalle del cálculo de uno de los nodos:

$Grupo$	$V(S)$
{1}	1
{2}	1
{3}	1
{4}	1
{1, 2}	1.2
{1, 3}	1.3
{1, 4}	1.8
{2, 3}	1.7
{2, 4}	1.8
{3, 4}	1.9
{1, 2, 3}	2.6
{1, 2, 4}	2.7
{1, 3, 4}	2.6
{2, 3, 4}	2.7
{1, 2, 3, 4}	3.5

Cuadro 6.3: Valor de una coalición

$$
\begin{aligned}
\Phi_1 =& \frac{1!2!}{4!}[V(\{1,2\}) - V(\{2\})] + \\
& \frac{1!2!}{4!}[V(\{1,3\}) - V(\{3\})] + \\
& \frac{1!2!}{4!}[V(\{1,4\}) - V(\{4\})] + \\
& \frac{2!1!}{4!}[V(\{1,2,3\}) - V(\{2,3\})] + \\
& \frac{2!1!}{4!}[V(\{1,2,4\}) - V(\{2,4\})] + \\
& \frac{2!1!}{4!}[V(\{1,3,4\}) - V(\{3,4\})] + \\
& \frac{3!0!}{4!}[V(\{1,2,3,4\}) - V(\{2,3,4\})] + \frac{3!0!}{4!}[V(\{1\})] = \\
& \frac{2}{24}(1{,}2 - 1) + \frac{2}{24}(1{,}3 - 1) + \frac{2}{24}(1{,}8 - 1) + \\
& \frac{2}{24}(2{,}6 - 1{,}7) + \frac{2}{24}(2{,}7 - 1{,}8) + \frac{2}{24}(2{,}6 - 1{,}9) + \\
& \frac{6}{24}(3{,}5 - 2{,}7) + \frac{6}{24}(1) = 0{,}766
\end{aligned}
$$

Finalmente en la tabla 6.4 se incluye un resumen con todos los valores de Shapley de este ejemplo. Utilizando este valor, el mecanismo repartirá las tareas o el esfuerzo .

Según el ejemplo, el jugador 1 ejecutaría un volumen de tareas según la proporción $\frac{\Phi_1}{3{,}5} = \frac{0{,}766}{3{,}5}$ y el sistema agregado el resto, es decir $\frac{2{,}74}{3{,}5}$.

Revisemos ahora un ejemplo extremo que parece injusto según la solución habitual con un reparto de trabajo basado en el número de jugadores reales. Supongamos tres jugadores, dos de los cuales están fuertemente correlados (supongamos que son los jugadores 2 y 3), mientras que el tercero es independiente (identificado con el número 1). Si los jugadores 2 y 3 se agrupan, el sistema repartirá la mitad de las tareas a dicha agrupación y la otra mitad al jugador 1. De esta forma, el jugador 1 ejecutará la mitad de las tareas, mientras que los otros dos jugadores ejecutarán una cuarta parte. Por el contrario, si los jugadores trabajan de forma individual, cada uno de ellos ejecutará la tercera parte del trabajo. Es decir, con esta noción de justicia

n	Φ_i
1	0,766
2	0,875
3	0,866
4	1,000
Total	3,500

Cuadro 6.4: Ejemplo de valores de Shapley

nos encontraremos que los jugadores 2 y 3 deseen agruparse y engañar al sistema. En cambio, con la propuesta de justicia basada en el valor de Shapley y número virtual de jugadores se obtendrían unos resultados similares. Recordemos que los valores de coalición de este ejemplo serían los expresados en la tabla 6.5.

Aplicando el valor de Shapley a este ejemplo nos daría un reparto de trabajo similar tanto si los jugadores se agrupan cómo si trabajan de forma individual. En este ejemplo podemos comprobar que los jugadores no tendrían interés en ocultar su identidad, lo que hace de esta noción de justicia algo muy interesante para un estudio más detallado.

Antes de finalizar, nos gustaría resaltar algunos de los puntos que consideremos clave para formalizar el concepto de jugadores virtuales. El primero de ellos es que el tratamiento matemático de QPQ presupone una cierta relación de justicia entre los jugadores. Nos referimos a que hemos fijado una relación de justicia basada en el número de jugadores (n) o el valor de Shapley aplicado al número virtuales de jugadores. No está claro bajo qué condiciones los algoritmos seguirían siendo strategy-proof si los jugadores pudieran manipular esos datos.

Otro de los aspectos críticos del planteamiento anterior es que el número de jugadores virtuales depende del valor mínimo declarado y no del valor de corte aplicado. Lo ideal sería que el número virtual dependiera realmente del valor de ejecución de-

$Grupo$	$V(S)$
$\{1\}$	1
$\{2\}$	1
$\{3\}$	1
$\{1, 2\}$	2
$\{1, 3\}$	2
$\{2, 3\}$	1
$\{1, 2, 3\}$	2

Cuadro 6.5: Nuevo ejemplo de coalición

finido por la función de decisión. Para ello sería necesario modificar el algoritmo y recalcular continuamente el valor de Shapley según se adapta el valor de corte. La dificultad de esta propuesta radica en el hecho de que tenemos una definición recursiva de mecanismo.

Matemáticamente, podemos entender el mecanismo como un operador que trabaja sobre un vector de n entradas, en el que cada elemento del vector es el número virtual de cada jugador. Es decir, supongamos que el vector $v = (\hat{n}_1, \hat{n}_2, \ldots)$ representa el número virtuales de cada jugador. Cada mecanismo $\mathfrak{M}$ intenta repartir la carga según ese vector, lo que resulta en un coste global. Pero el problema es que con el coste final del sistema, podríamos obtener un nuevo reparto de trabajo y por lo tanto, podríamos obtener un nuevo vector. Se podría representar como:

$$\mathfrak{M} \cdot v_k = v_{k+1}$$

Si este proceso converge, el sistema tendría muchas de las propiedades deseables. Algunas simulaciones previas realizadas con QPQ Correlado parecen apuntar este hecho. Sería muy recomendable ampliar estas simulaciones y demostrar matemáticamente que efectivamente se cumplen las propiedades de convergencia y de

resistencia a las agrupaciones.

Por último, nos gustaría resaltar la importancia de la utilización de un número virtual de jugadores y de la propiedad de resistencia a las agrupaciones. Nótese que si un algoritmo cumple con esta última propiedad, los problemas que hemos citado al comienzo de este apartado tendrían una naturaleza muy diferente. Recordemos los tres problemas iniciales:

1. Reducción del volumen de mensajes: Si un mecanismo cumple con la propiedad anterior, el sistema podría crear agrupaciones de nodos de forma jerárquica sin que por ello los jugadores vean modificados sus esfuerzos. Un jugador no notará la diferencia entre jugar contra un jugador agregado o participar de la agrupación.

2. Aunque los jugadores puedan mentir sobre el número de elementos que componen una agrupación, el sistema construiría un reparto basado en la capacidad de trabajo real. Al igual que el punto anterior, si un mecanismo cumple con la citada propiedad, que lo jugadores se agrupen y mientan sobre su composición no afectaría al sistema.

3. El número de jugadores puede cambiar en el tiempo. Cualquier mecanismo con la ya mencionada propiedad podrá volver a converger hacia un nuevo punto de reparto y evolucionar según los jugadores entren o salgan del sistema.

Esperamos que futuras líneas de investigación puedan servir para arrojar algo de luz sobre estos aspectos.

Capítulo 7

Conclusiones y trabajos futuros

> "It is good to have an end to journey toward, but it is the journey that matters in the end."
>
> Ursula K. LeGuin

7.1. Conclusiones

La presente tesis ha abordado una situación que, a nuestro parecer, supone un reto para la construcción de sistemas distribuidos. Nos referimos al carácter manipulador y egoísta de los nodos participantes. La computación distribuida en Internet se enfrenta, cada día más, a los problemas que ocasionan los agentes egoístas. Muchos algoritmos y sistemas dejan de funcionar o pierden rendimiento cuando los participantes manipulan las reglas del juego para su propio provecho. Continuamente aparecen nuevas versiones y parches de programas que intentan luchar contra la última manipulación, dando lugar a lo que parece ser una guerra sin fin. Sistemas como BitTorrent ya se han enfrentado a este tipo de problemas, muchos de ellos producidos por lo que se conoce como "free-riding": el jugador quiere aprovecharse de las ventajas del sistema sin tener que colaborar en el mismo.

En este contexto, nuestro trabajo ha buscado aportar soluciones en un frente muy

concreto. Nos referimos a la asignación de tareas en redes distribuidas. Nuestro objetivo era explorar modelos y soluciones que hicieran posible aprovechar al máximo los beneficios de la colaboración en la red, reduciendo al mínimo los efectos negativos del egoísmo.

A lo largo del desarrollo de la presente tesis se han ido cubriendo los principales retos y objetivos marcados. Inicialmente, se exploraron diversos modelos matemáticos, se analizaron los límites y se describieron las futuras variantes. En este punto, resultó de gran ayuda las herramientas de la teoría de juegos.

Existen numeros trabajos que utilizan la teoría de juegos para la construcción y análisis de redes y, en general, de otros ámbitos informáticos. Nuestro trabajo utiliza algunos de esos modelos, pero los hemos ampliado o modificado para adaptarlos a nuestra visión de lo que podrían ser las redes distribuidas en entornos agresivos similares a Internet. Creemos que nuestras aportaciones en este sentido, han sido múltiples. Según nuestro conocimiento, hemos propuesto un esquema de trabajo novedoso que permite eliminar algunos de los requisitos expresados habitualmente: la existencia de nodos centrales que controlan el funcionamiento y la suposición de que es posible realizar pagos. Pero nuestro trabajo no se ha limitado a la eliminación de estos requisitos, también hemos querido relajar, o incluso reducir al mínimo, otras suposiciones que no son fácilmente aplicables en las redes informáticas. Estamos hablando de elementos tales como la suposición de que los nodos son racionales o de que tenemos capacidad de cómputo suficiente. Somos conscientes que los modelos presentados aún tienen muchas alternativas para ser analizadas en el futuro, pero esperamos que nuestra propuesta sea lo suficientemente atractiva como para servir de referencia en aplicaciones prácticas futuras.

Partiendo del modelo expuesto, nuestro trabajo avanzó hacia el desarrollo de varios algoritmos que, entendemos, han resuelto el problema planteado en la tesis. A esta familia de algoritmos los hemos denominado Mecanismos QPQ. Desde una versión básica hemos introducido nuevas variantes hasta llegar a un QPQ con carácter general. Aunque los algoritmos más básicos son matemáticamente casos particulares de los más generales, en la práctica tienen implementaciones muy diferentes, por lo

que hemos expuesto todas las variantes estudiadas. En todas ellas, se han analizado matemáticamente las principales propiedades y limitaciones de nuestro algoritmo. Hemos hecho especial énfasis en la propiedad que implica decir la verdad como estrategia dominante. Es una propiedad muy atractiva ya que, gracias a ella, podemos afirmar que los nodos actuarán de forma honesta, que el sistema podrá garantizar unos resultados estadísticamente óptimos y que los nodos mentirosos, o incluso los no-racionales, no afectarán a los honestos -o existirá un impacto limitado-. Esta propiedad hace que el aprendizaje de los nodos no sea relevante. Los nodos decidirán ser honestos o no, independientemente de lo que pudieran hacer los demás. Aprender el comportamiento de los contrarios no ayudará a tomar la decisión de ser honesto.

Otro de los aspectos relevantes de nuestra investigación es que las propiedades se han estudiado bajo un escenario con poca o ninguna información previa. Todas las ventajas de QPQ se han analizado suponiendo que esa información no era fácil de obtener o incluso, que podía ser objeto de manipulación. Matemáticamente, estas propiedades se han analizado utilizando conceptos de estadística, pero siempre se ha buscado que los jugadores pudieran obtener beneficios para todas las situaciones posibles, incluyendo aquellas en la que disponer de información previa (o la falta de ella) podía ser ventajosa para los nodos deshonestos. En la teoría de juegos, algunas de estas suposiciones se estudian como conceptos “ex-ante”.

Pero también se han realizado simulaciones para estudiar el comportamiento de QPQ básico y, en menor medida, el QPQ Correlado. Ciertas propiedades no han sido estudiadas con herramientas matemáticas, principalmente las relativas al comportamiento dinámico del sistema tales como la velocidad de convergencia, tamaño idóneo de las muestras o la tolerancia al daño de los deshonestos. Sin embargo, se han realizado numerosas simulaciones que han arrojado cierta luz sobre el comportamiento del sistema si fuera implementado en la realidad. Evidentemente, si ampliamos la longitud de la información histórica, se podrían obtener resultados estadísticamente cercanos al óptimo buscado, pero hemos visto cómo sería posible tener sistemas funcionando con poca información histórica (30 a 100 muestras) y sin que el sistema se muestre inestable o ineficiente. Somos conscientes de que para la simulación de QPQ

se han utilizado herramientas matemáticas muy concretas que, seguramente podrán ser mejoradas en el futuro. Estamos hablando de las herramientas de regresión, test de bondad, aproximación estocástica, etc. La utilización de unas u otras no alteran de forma significativa las propiedades de QPQ, pero sí pueden mejorar la implementación práctica. Gracias a las simulaciones realizadas, y con una primera realización del algoritmo, podemos afirmar que QPQ podría ser aplicable en contextos reales. Sólo nos queda por explorar la forma de mejorar su comportamiento con otras herramientas estadísticas alternativas, existentes o futuras.

Finalmente, queremos destacar un resultado de nuestro trabajo que consideramos relevante. Aunque el contexto inicial de la tesis eran los sistemas distribuidos en informática, creemos que nuestro trabajo podría tener una aplicación mucho más amplía. Hemos trabajado con conceptos matemáticos que se originaron en el contexto económico, pero que han demostrado ser de gran utilidad en muchos otros ámbitos del conocimiento humano (biología, ingeniería, psicología, etc.). No hay nada en nuestro modelo que impida su aplicación en otros campos de investigación diferentes a las redes informáticas.

Las soluciones planteadas con los mecanismos de QPQ nos han permitido arrojar luz sobre aspectos poco estudiados en la aplicación de la teoría de juegos al ámbito informático. En nuestro trabajo hemos introducido nuevas ideas que han demostrado ser herramientas muy útiles en el diseño práctico de mecanismos sin pagos. Estamos hablando de los elementos de diseño básicos de QPQ, tales como incluir la transformada PIT, o la posibilidad de utilizar funciones de decisión basadas en líneas de corte. Finalmente, hemos visto en el capítulo anterior, la posibilidad de combinar mecanismos de enlace con los mecanismos clásicos basados en VCG. La potencia de esta idea está aún por descubrir, pero las posibilidades prometen ser inmensas. Si existiera un procedimiento general de convertir un mecanismos de Groves (mecanismos VCG) en otro enlazado, el diseño de mecanismos enlazado se vería enriquecido por toda la base teórica existente. Nuestro trabajo sólo ha sido un primer paso en el estudio de esta teoría, pero estamos convencidos de que los avances en ese sentido podrían tener un gran impacto en el diseño de sistemas distribuidos.

7.2. Validación o refutación de las hipótesis

A la luz del trabajo realizado, pasamos a evaluar la validez de las hipótesis iniciales objeto de la presente tesis.

Las hipótesis de trabajo que se propusieron en esta tesis eran las siguientes:

Hipótesis 1: *Es posible formalizar matemáticamente un sistema distribuido en el que los nodos tienen información privada, que pueden revelar de forma correcta o incorrecta con el objetivo de manipular la asignación de tareas en su propio beneficio.*

La base teórica desarrollada por la teoría de juegos y en concreto, por el diseño de mecanismos, aportan herramientas matemáticas de gran utilidad para el estudio de este tipo de problemas. Aunque no es la primera vez que un trabajo en redes distribuidas utiliza estas herramientas para un estudio formal del problema, creemos que nuestro trabajo las ha utilizado desde un planteamiento novedoso. Por lo que hemos podido comprobar, es el primer trabajo que aplica mecanismos de enlace a un problema relacionado con los sistemas distribuidos. Por todo lo anterior, consideramos que esta primera hipótesis se encuentra plenamente validada y confirmada.

Hipótesis 2: *Existen algoritmos capaces de asignar tareas en un sistema distribuido que incentivan a los participantes a revelar correctamente su información privada.*

A lo largo de nuestro trabajo hemos propuesto una familia de algoritmos, que hemos denominado QPQ, y que han demostrado ser tolerantes a las mentiras. Esto es, hemos demostrado que bajo ciertas condiciones los jugadores no tienen incentivos en mentir cuando revelan su información privada. Cada uno de estos algoritmos son resistentes a las mentiras cuando se restringen las condiciones de trabajo. En QPQ Básico los jugadores tienen valoraciones independientes, y bajo tal suposición, el mecanismo es strategy-proof. Con los algoritmos de QPQ Correlados nos sucede algo similar. Estos mecanismos suponen que los jugadores tienen cierta correlación entre sus valoraciones pero no tienen capacidad para alterarla. De nuevo, estos mecanismos son strategy-proof aunque nuestro trabajo ha demostrado que pierden esa propiedad si ampliamos el espectro de acciones de los agentes.

Por lo tanto, podemos afirmar que la segunda hipótesis se encuentra confirmada, aunque bajo las suposiciones apropiadas. Cuando trabajamos con un modelo completamente general (los jugadores pueden alterar el nivel de correlación entre ellos), nuestra investigación demuestra que los algoritmos básicos pueden ser sensibles a la manipulación.

Hipótesis 3: *Tales algoritmos existen en condiciones que puedan hacer viable su implementación en entornos reales, tales como sistemas de intercambio de ficheros en redes P2P o computación distribuida (Grid Computing).*

En nuestro trabajo hemos propuesto un conjunto de algoritmos que han demostrado tener propiedades muy interesantes. Mediante simulaciones hemos podido comprobar que es posible aplicar este mecanismo sin apenas información de los nodos y manteniendo un histórico muy bajo de muestras. El algoritmo asociado al mecanismo ha resultado ser sencillo de implementar, con una complejidad de cómputo abordable en redes distribuidas. Aunque el número de mensajes a intercambiar podría ser alto, se ha visto que es posible agrupar a los jugadores en conjuntos y reducir, de esta forma, el tráfico resultante.

La segunda familia de algoritmos (QPQ Correlados), aún manteniendo en esencia las mismas propiedades matemáticas que el algoritmo QPQ Básico, pueden ser más complejos de implementar. Esto es debido principalmente al tratamiento estadístico de la regresión. Desgraciadamente, para obtener esta información es necesario disponer de un mayor número de muestras lo que implica dificultades en su realización práctica. De todas formas, es una dificultad relacionada con el tratamiento estadístico y creemos que, salvo por esta característica, estos algoritmos son razonablemente sencillos para ser implementados de forma práctica en sistemas distribuidos.

Hipótesis 4: *Que estos algoritmos tienen las características adecuadas para que los nodos participantes no deseen cambiar las reglas y propongan sistemas alternativos. En particular, realizarán una asignación razonablemente justa de las tareas a los participantes.*

Nuestra última hipótesis tiene una fuerte componente de lo que, en teoría de juegos, se denomina “arrepentimiento” (“regret”). En esta teoría, una forma de justicia

está basada en el arrepentimiento de los jugadores, entendiendo como tal cuando un jugador desea un modelo alternativo de juego si con ello espera poder obtener un mayor beneficio. Pero no es la única forma de medir la noción de justicia. Toda nuestra investigación ha tenido en mente el concepto de justicia. Hemos podido estudiar las implicaciones de algunas de las posibles variantes de un reparto "justo" de tareas. En nuestro trabajo, la transformada PIT ha resultado una herramienta clave para igualar las posibilidades de trabajo del nodo, comparando siempre su trabajo en unidades medidas según su propio criterio. En QPQ Correlado hemos profundizado en este concepto y hemos definido dos variantes según dos nociones de justicia.

La principal conclusión de nuestro trabajo con respecto a este último punto es que la justicia no es única y que, en realidad, se puede expresar como una ligadura en un sistema de ecuaciones a optimizar. Nuestro trabajo aporta un marco en el que se pueden definir y estudiar estos sistemas. Bajo esta perspectiva podemos afirmar que nuestra última hipótesis ha sido verificada.

7.3. Investigación futura y problemas abiertos

A pesar de los resultados obtenidos en la presente tesis, son múltiples las incógnitas que se han abierto y que merecen ser objeto de un estudio más amplio. Las alternativas a nuestro modelo se nos antojan innumerables y todas son de gran interés. Existe una gran multitud de herramientas matemáticas (principalmente estadísticas) que merecen ser probadas y estudiado su impacto en la implantación de QPQ. Pero, según mi opinión, hay una serie de puntos que han quedado abiertos y que podrían suponer un gran avance en el desarrollo de QPQ. En esta sección veremos algunas de las líneas de investigación que consideramos prometedoras, y que hemos agrupado en cuatro grandes bloques:

1. **Elementos a mejorar en las implementaciones de QPQ**. A lo largo del presente trabajo hemos visto que existen múltiples alternativas matemáticas (principalmente estadísticas), que podrían ser de interés para mejorar el rendimiento práctico de QPQ.

2. **Elementos teóricos de QPQ**. Muchas de las demostraciones realizadas han impuesto ciertas condiciones tales como continuidad. Muchas de las investigaciones futuras tendrían que explorar si es posible generalizar los resultados obtenidos.

3. **Nuevos modelos y problemas en los sistemas distribuidos**. La tesis ha abordado un problema relacionado con la ejecución de tareas, pero no se han estudiado modelos alternativos: existencia de fallos, asincronía, comportamiento bizantino, número variable de jugadores, etc..

4. **Comportamiento dinámico de QPQ**. Sería interesante conocer el comportamiento dinámico de QPQ. Nuestros algoritmos QPQ con justicia no son estrategy-proof si los jugadores pueden modificar la correlación, pero es posible que existan puntos de equilibrio bayesianos que sean de interés.

Veamos estos puntos con más detalle.

7.3.1. Implementación de QPQ

Las implementaciones concretas de QPQ que se han utilizado para realizar las simulaciones se basan en una serie de herramientas matemáticas tales como el test de bondad de Kolmogorov-Smirnov en QPQ Básico o en el algoritmo de aproximación de Robbins-Monro. Las simulaciones han confirmado que estas herramientas son de gran ayuda para la implementación de QPQ. Pero no son las únicas herramientas posibles y seguramente existan alternativas para su implementación. Nuestra intención en este apartado es repasar brevemente algunas de estas alternativas, aunque sin buscar una revisión completa del estado del arte en este ámbito.

Sobre el test de bondad. Es evidente que la potencia del test de bondad tiene un fuerte impacto en el rendimiento de QPQ. Esta potencia debemos entenderla como la capacidad que tiene el test de detectar desviaciones de la distribución uniforme con un número bajo de muestras y el impacto que suponen los falsos positivos/negativos.

En el capítulo cuarto, cuando introdujimos QPQ, mencionamos la existencia de otros test de bondad, tales como el test de Greenwood o el test de Kendall y Sherman. Sería recomendable establecer estudios comparativos que permitan detectar la bondad de tales tests aplicados a QPQ. Se propone realizar un estudio del daño que los mentirosos pudieran ocasionar y del beneficio de los jugadores para una longitud de histórico dado. La idea sería buscar el test de bondad que funcione correctamente por debajo de 30 muestras, mientras que se mantienen la misma utilidad esperada, incluso ante posibles mentirosos. Una posibilidad de mejora podría ser la utilización de dos o más tests simultáneos y combinar las probabilidades de todos.

Sobre el umbral de aceptación. En QPQ básico hemos visto que es recomendable adaptar el umbral de aceptación del test de bondad. Una posibilidad ha sido abrir o cerrar ese umbral en función del beneficio del jugador. Un jugador deshonesto que incrementa su beneficio mediante mentiras, terminará cerrando el margen de aceptación de sus valores y tenderá a que el sistema no acepte los valores declarados. La fórmula propuesta en QPQ Básico ha sido obtenida de forma experimental que, mediante simulaciones, ha demostrado ser conveniente.

Recuérdese que, con cierta probabilidad, a veces el sistema rechaza los valores auténticos. Esos falsos negativos producen un daño al jugador honesto. Pensamos que sería interesante buscar una alternativa que, mediante una demostración formal, nos asegure el mayor beneficio de los jugadores honestos, mientras que penaliza a los mentirosos.

Una propuesta a investigar sería mezclar una combinación de test de Gof con un acumulado de pagos hipotéticos (virtuales ya que no se realizan en la práctica). Si el nodo acumula mucha deuda virtual se le cierra la ventana de confianza (el valor de P del test Gof). Si el nodo se encuentra en el rango esperado de deuda, le dejamos una ventana de confianza más abierta.

Sobre la aproximación estocástica. En las versiones de QPQ con justicia se ha propuesto la utilización el algoritmo de Robbins-Monro en su forma más básica. Explorando la literatura al respecto, el lector podrá observar que existen múltiples

variantes y alternativas, muchas de ellas muy prometedoras. Por ejemplo, una posibilidad es la utilización del algoritmo de Kiefer-Wolfowitz.

Dado que la velocidad de convergencia de los parámetros λ es un factor clave para garantizar los niveles de beneficios, se propone estudiar en el futuro otros algoritmos que permitan acelerar esa convergencia. La idea es que se puedan obtener resultados equivalentes aunque con un número reducido de muestras.

Sobre el error estadístico. Por otra parte, de nuestras simulaciones no hemos podido obtener elementos de juicio que nos permitan valorar la bondad del algoritmo cuando el número de jugadores crece. Aunque matemáticamente está claro que los jugadores ganarán más cuanto mayor sea la población de jugadores, creemos que el error estadístico se verá influenciando por el número de nodos.

El tamaño de la población influye en la dimensión de las ecuaciones y afecta claramente a las estimaciones estadísticas. Nuestro planteamiento de QPQ se basa en el hecho de que un jugador siempre compite contra el sistema agregado de los jugadores. Pero si el jugador agregado representa a un número amplio de jugadores, la probabilidad de ejecutar la tarea se reduce, y por lo tanto la posibilidad de ajustar algunos de los parámetros estadísticos que rigen a QPQ.

No disponemos de datos suficientes para valorar si ese error estadístico llega a ser significativo, ya que seguramente se vea compensado por los beneficios de participar en un juego de estas características. Sería necesario estudiar con mayor profundidad si es cierta esta hipótesis, y de ser cierta, buscar alternativas que no dependan del número de jugadores.

7.3.2. Elementos teóricos

Nuestras demostraciones de QPQ se basan en muchas de las suposiciones habituales de continuidad, diferenciabilidad, etc. Hemos utilizado principalmente herramientas de trabajo basadas en la integral de Riemman, y en general hemos supuesto que nuestras funciones de densidad de probabilidad existían. Creemos que sería interesante estudiar si todos los resultados obtenidos se pueden demostrar en contextos

más generales. En concreto, creemos que sería interesante explorar los puntos siguientes.

Espacios de probabilidad. Prácticamente en todo nuestro trabajo hemos supuesto las condiciones habituales de continuidad y derivabilidad. Pero sería interesante revisar todas las conclusiones obtenidas en un contexto lo más amplio posible. Algunas de las herramientas utilizadas no suelen ser de aplicación directa si relajamos las suposiciones anteriores. Sin embargo, y por lo que hemos podido comprobar en capítulos anteriores, se puede pensar en demostraciones más generales que no impliquen esas suposiciones.

Lo que proponemos es la generalización de las conclusiones obtenidas, de tal forma que se puedan utilizar espacios de probabilidad generales y que no sean necesarias las suposiciones de continuidad, de derivabilidad, etc.

Existencia de la función de corte. Hemos demostrado que para QPQ con justicia se puede obtener un sistema óptimo si utilizamos una función de decisión determinista y basado en la línea de corte. En cambio, queda abierto el problema si esta función de corte se puede aplicar a otros conceptos de justicia y, en concreto, si esta línea de corte es válida para la solución de QPQ con conexión con los mecanismos VCG.

Por otra parte, algunas de las propiedades de la línea de corte observadas no tienen que cumplirse necesariamente para ligaduras más generales de justicia. Nos referimos al hecho de que la línea de corte se puede expresar como $y = \psi(x)$. Quizás en algunos casos, sea necesario trabajar con decisiones basadas en funciones paramétricas, por ejemplo del estilo de *argmin* $(\psi_i(\theta))$.

Relación con mecanismos VCG. Hemos propuesto una primera versión de mecanismo enlazado que utiliza la potencia de los mecanismos VCG. Pero nuestro desarrollo teórico se ha realizado para un ejemplo concreto de mecanismo: la asignación de tareas entre jugadores. Además, se han utilizado dos aspectos que podrían ser objeto de una investigación posterior: la línea de corte en esquemas más generales y la

justicia basada en la "potencia" de los jugadores, medida como número de jugadores virtuales.

7.3.3. Nuevos modelos

A lo largo de la presente tesis se ha comentado la existencia de modelos alternativos que pudieran ser interesantes de estudio. Las posibilidades son enormes, pero quizás merezca la pena resaltar algunos de los modelos que el autor desea abordar en investigaciones futuras.

Número de jugadores. Nos gustaría resaltar la línea de trabajo relacionada con un número de jugadores variable y desconocido. Los elementos de justicia explorados son susceptibles de ser manipulados cuando los jugadores pueden trabajar en agregaciones invisibles al diseñador de mecanismos. La entrada y salida dinámica y anónima de los jugadores en el juego (típico de redes P2P) genera situaciones complejas de estudio. Las soluciones planteadas en el capítulo anterior deberían ser analizadas matemáticamente y/o simuladas.

Generalizar el pérfil de los Jugadores. Alterar la formulación de QPQ e introducir conceptos tales como adversion al riesgo o el factor de descuento en el tiempo. Hemos supuesto que los jugadores son neutrales al riesgo e infinitamente pacientes. Si alteramos estas condiciones, un jugador podría intentar ganar antes de que la partida se terminara. O aprovechar para acumular beneficios en el presente, aunque a futuro tenga que perder. Igualmente, sería interesante estudiar en qué medida los jugadores alteran sus estrategias cuando son propensos o adversos al riesgo.

Múltiples productores. Nuestro modelo actual presupone que un único agente se hace responsable de la ejecución de la tarea. Es decir, hemos supuesto un único productor y varios consumidores. Pero esta situación podría generalizarse cuando pensamos un número variable de productores y de consumidores. Por ejemplo, podemos pensar en que varios jugadores realicen la tarea de forma paralela para ahorrar tiempo.

Otra posibilidad podría ser que se ejecuten por varios nodos para evitar situaciones de errores o fallos aleatorios.

Pensamos que una posibilidad sería estudiar el problema bajo la óptica de los mecanismos VCG (por ejemplo, utilizando mecanismos "multi-unit single-demand") y adaptar los resultados a QPQ, utilizando para ello esquemas similares a los expuestos en el capítulo anterior.

7.3.4. Comportamiento dinámico de QPQ

Con los contrajemplos vistos en capítulo anterior, es evidente que QPQ tiene limitaciones cuando los jugadores tienen la posibilidad de alterar la correlación de probabilidad. Este aspecto nos abre nuevos campos de investigación que podrían añadirse a los ya vistos anteriormente en QPQ. Citamos aquí algunos de los que consideramos más interesantes:

1. Estudio de puntos de equilibrio bayesiano de QPQ con conservación de tareas. Los trabajos previos parecen indicar que podrían existir puntos de equilibrio bayesiano aunque la naturaleza o número de éstos es desconocido en la actualidad.

2. Estudio de la dinámica del juego de QPQ con justicia y jugadores capaces de alterar la correlación. Esto abre la posibilidad a que los jugadores puedan aprender unos de otros y adaptar sus valores en consecuencia. Este tipo de juego crearía un sistema dinámico de gran interés para su estudio: existencia de puntos de equilibrio, estabilidad, convergencia, etc.

Bibliografía

[AE03] Luzi Anderegg and Stephan Eidenbenz. Ad hoc-vcg: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 245–259, New York, NY, USA, 2003. ACM Press.

[AER93] H.H. Ali and H. El-Rewini. Task allocation in distributed sytems: A split graph model. *J. Computer Math. Combination Computing, vol. 14*, 1993.

[AGLS11] A. Fernández Anta, C. Georgiou, L. López, and A. Santos. Reliable internet-based master-worker computing in the presence of malicious workers. *Parallel Processing Letters*, 2011.

[AGM10] Antonio Fernández Anta, Chryssis Georgiou, and Miguel A. Mosteiro. Algorithmic mechanisms for internet-based master-worker computing with untrusted and selfish workers. In *IPDPS*, pages 1–11. IEEE, 2010.

[Ami] Artur Czumaj Amir. On the expected payment of mechanisms for task allocation.

[AMS95] R.J. Aumann, M. Maschler, and R.E. Stearns. *Repeated Games with Incomplete Information*. MIT Press, 1995.

[AR88] Dilip Abreu and Ariel Rubinstein. The structure of nash equilibrium in repeated games with finite automata. *Econometrica*, 56(6):1259–81, November 1988.

[AS92] Robert J. Aumann and Lloyd S. Shapley. Long term competition-a game theoretic analysis. UCLA Economics Working Papers 676, UCLA Department of Economics, September 1992.

[ASB05] D. Austen-Smith and J.S. Banks. *Positive political theory II: strategy and structure*. Michigan studies in political analysis. University of Michigan Press, 2005.

[AY08] Murat Atlamaz and Duygu Yengin. Fair groves mechanisms. *Social Choice and Welfare*, 31(4):573–587, December 2008.

[Bai97] Martin J Bailey. The demand revealing process: To distribute the surplus. *Public Choice*, 91(2):107–26, April 1997.

[Bak74] K.R. Baker. Introduction to sequencing scheduling. *John Wiley & Sons*, 1974.

[BCS92] A. Billionnet, M.-C. Costa, and A. Sutter. An efficient algorithm for a task allocation problem. *J. ACM, vol. 39*, 1992.

[Ber05] J. Bergin. *Microeconomic theory: a concise course*. Oxford scholarship online. Oxford University Press, 2005.

[BFB06] Steven Bauer, Peyman Faratin, and Robert Beverly. Assessing the assumptions underlying mechanism design for the internet. In *Economics of Networked Systems*, June 2006.

[BHS04a] S. Barbera, P.J. Hammond, and C. Seidl. *Handbook of Utility Theory*. Kluwer Academic Pub, 2004.

[BHS04b] S. Barberá, P.J. Hammond, and C. Seidl. *Handbook of Utility Theory: Extensions*. Handbook of Utility Theory. Kluwer Academic Publishers, 2004.

[BK98] Jean-Pierre Benoît and Vijay Krishna. The folk theorems for repeated games: A synthesis, 1998.

[Bok93] S.H. Bokhari. A network flow model for load balancing in circuit-switched multicomputers. *IEEE Trans. Parallel and Distributed Systems, vol. 4*, 1993.

[BR06] Christian Burgert and Ludger Rüschendorf. On the optimal risk allocation problem. *Statistics and Decisions*, 24:153–171, 2006.

[BS08] Sandeep Baliga and Tomas Sjöström. mechanism design (new developments), 2008.

[BT51] Z. W. Birnbaum and Fred H. Tingey. One-sided confidence contours for probability distribution functions. *The Annals of Mathematical Statistics*, 4:592–596, 1951.

[BT01] Shlomo Benartzi and Richard H. Thaler. Naive diversification strategies in defined contribution saving plans. *American Economic Review*, 91(1):79–98, March 2001.

[Cav06] Ruggiero Cavallo. Optimal decision-making with minimal waste: strategyproof redistribution of vcg payments. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, AAMAS '06, pages 882–889, New York, NY, USA, 2006. ACM.

[CH01] Enrica Carbone and John Hey. A test of the principle of optimality. Discussion papers, Department of Economics, University of York, 2001.

[Chu80] W.W. Chu. Task allocation in distributed data processing. *Computer, vol. 13, pp. 57-69*, 1980.

[CI05] Vincent P. Crawford and Nagore Iriberri. Level-k auctions: Can a non-equilibrium model of strategic thinking explain the winner's curse and overbidding in private-value auctions? University of California at San Diego, Economics Working Paper Series 128088, Department of Economics, UC San Diego, November 2005.

[Cih08] Martin Cihak. The 2007 nobel prize in economics: Mechanism design theory. *Czech Journal of Economics and Finance (Finance a uver)*, 58(01-02):82–89, 2008.

[CJK88] Thomas L. Casavant, Jon, and G. Kuhl. A taxonomy of scheduling in general-purpose distributed computing systems. *IEEE Transactions on Software Engineering*, 14:141–154, 1988.

[CL87] W.W. Chu and L.M. Lan. Task allocation and precedence relations for distributed real-time system. *IEEE Trans. Computers*, 1987.

[Cla71] Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 18:19–33, 1971.

[CM88] Jacques Cremer and Richard P McLean. Full extraction of the surplus in bayesian and dominant strategy auctions. *Econometrica*, 56(6):1247–57, November 1988.

[Com07] Nobel Prize Committee. Leonid hurwicz, eric s. maskin and roger b. myerson: Mechanism design theory. Nobel Prize in Economics documents 2007-2, Nobel Prize Committee, October 2007.

[Cor07] Luis C. Corchon. The theory of implementation : what did we learn? Economics Working Papers we081207, Universidad Carlos III, Departamento de Economía, December 2007.

[CR02] Gajiy Charness and Matthew Rabin. Understanding social preferences with simple tests. *Journal of Economics, August*, pages 817–869, 2002.

[Daw76] Richard Dawkins. *The Selfish Gene*. Oxford University Press, New York, 1976.

[Deb59] Gerard. Debreu. *Theory of value : an axiomatic analysis of economic equilibrium*. Wiley, N.Y. :, 1959.

[Deb72] Gerand Debreu. *Theory of Value: An Axiomatic Analysis of Economic Equilibrium*. Yale University Press, May 1972.

[DJP03] Rajdeep K. Dash, Nicholas R. Jennings, and David C. Parkes. Computational-mechanism design: A call to arms, 2003.

[ea82] P.Y.R. Ma et al. A task allocation model for distributed computing systems. *IEEE Trans. Computers, vol. 31*, 1982.

[EHP00] M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*. JOHN WILEY & SONS, INC, New York, 2000.

[Fer67] Thomas S. Ferguson. *Mathematical Statistics: A Decision Theoretic Approach*. Academic Press, 1st edition, July 1967.

[Fis70] Peter C Fishburn. *Utility Theory for Decision Making*. Wiley, 1970.

[Fis10] P.C. Fishburn. *The Foundations of Expected Utility*. Theory and Decision Library. Springer, 2010.

[FLSG06] Antonio Fernández, Luis López, Agustín Santos, and Chryssis Georgiou. Reliably executing tasks in the presence of untrusted entities. In *SRDS*, pages 39–50. IEEE Computer Society, 2006.

[FPS01] Joan Feigenbaum, Christos H. Papadimitriou, and Scott Shenker. Sharing the cost of multicast transmissions. *Journal of Computer and System Sciences*, 63:21–41, 2001.

[FPSS02] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A bgp-based mechanism for lowest-cost routing, 2002.

[Fre82] S. French. Sequencing and scheduling. *Halsted Press*, 1982.

[FS02] J. Feigenbaum and S. Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 1–13. ACM Press, New York, 2002.

[FS06] A.M. Feldman and R. Serrano. *Welfare economics and social choice theory*. Springer, 2006.

[GC07] Mingyu Guo and Vincent Conitzer. Worst-case optimal redistribution of vcg payments. In *Proceedings of the 8th ACM conference on Electronic commerce*, EC '07, pages 30–39, New York, NY, USA, 2007. ACM.

[GL79] Jerry R. Green and Jean-Jacques Laffont. Incentives in public decision-making. *New York: North Holland*, 1979.

[Gro73] Theodore Groves. Incentives in teams. *Econometrica*, 41(4):617–31, July 1973.

[Guo08] Mingyu Guo. Optimal-in-expectation redistribution mechanisms. In *In International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS*, 2008.

[HM53] I. N. Herstein and John Milnor. An Axiomatic Approach to Measurable Utility. *Econometrica*, 21(2):291–297, 1953.

[Hur73] Leonid Hurwicz. The design of mechanisms for resource allocation. *American Economic Review*, 63(2):1–30, May 1973.

[Jac99] Matthew O. Jackson. A crash course in implementation theory. Working Papers 1076, California Institute of Technology, Division of the Humanities and Social Sciences, July 1999.

[Jac03] M. Jackson. Mechanism theory, 2003.

[JG04] S.R.S. Rao Jammalamadaka and M. N. Goria. A test of goodness-of-fit based on gini's index of spacings. *Statistics & Probability Letters*, 68(2):177–187, June 2004.

[JS03] Matthew O. Jackson and Hugo F. Sonnenschein. The linking of collective decisions and efficiency. *Microeconomics*, 0 2003.

[JS04] Matthew O. Jackson and Hugo F. Sonnenschein. Overcoming incentive constraints by linking decisions (extended version), 2004.

[JS07] Matthew O Jackson and Hugo F Sonnenschein. Overcoming incentive constraints by linking decisions. *Econometrica*, 75(1):241–257, 01 2007.

[Kol33] A. N. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *Giornale dellIstituto Italiano degli Attuari*, 4:83–91, 1933.

[KP09] Elias Koutsoupias and Christos H. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3(2):65–69, 2009.

[Kri02] Vijay Krishna. *Auction Theory*. Academic Press, March 2002.

[KS08] Grigory Kosenok and Sergei Severinov. Individually rational, budget-balanced mechanisms and allocation of surplus. *Journal of Economic Theory*, 140(1):126 – 161, 2008.

[KW52] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23:462–466, 1952.

[KY97] H.J. Kushner and G. Yin. *Stochastic Approximation Algorithms and Applications*. Applications of Mathematics. Springer, 1997.

[KY03] H. Kushner and G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.

[Lab11] Linden Lab. Second life. `http://secondlife.com/`, October 2011.

[Lo88] V.M. Lo. Heuristic algorithms for task assignment in distrubted systems. *IEEE Trans. Computers, vol. 37*, 1988.

[LR09] Ludger and Rüschendorf. On the distributional transform, sklar's theorem, and the empirical copula process. *Journal of Statistical Planning and Inference*, 139(11):3921 – 3927, 2009.

[Mas51] Frank J Massey. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951.

[Mas08] Eric S. Maskin. Mechanism design: How to implement social goals. *American Economic Review*, 98(3):567–76, June 2008.

[MB01] P.W. Mielke and K.J. Berry. *Permutation methods: a distance function approach*. Springer series in statistics. Springer, 2001.

[MCWG95] A. Mas-Colell, M. Whinston, and J. Green. *Microeconomic Theory*. Oxford University Press: Oxford, 1995.

[Mil56] L. H. Miller. Table of percentage points of kolmogorov statistics. *Journal of the American Statistical Association*, 51:111–121, 1956.

[MMS08] Eiichi Miyagawa, Yasuyuki Miyahara, and Tadashi Sekiguchi. The folk theorem for repeated games with observation costs. *J. Econ. Theory*, pages 192–221, 2008.

[Mou10] Hervé Moulin. Auctioning or assigning an object: some remarkable vcg mechanisms. 2010.

[MS06] George J. Mailath and Larry Samuelson. *Repeated Games and Reputations: Long-Run Relationships*. Oxford University Press, USA, September 2006.

[Muk00] Nitis Mukhopadhyay. *Probability and statistical inference*. Marcel Dekker, Inc., 270 Madison Avenue, New York, NY 10016, 200.

[MYCH10] Derek G. Murray, Eiko Yoneki, Jon Crowcroft, and Steven Hand. The case for crowd computing. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, MobiHeld '10, pages 39–44, New York, NY, USA, 2010. ACM.

[Mye08] Roger B. Myerson. Perspectives on mechanism design in economic theory. *American Economic Review*, 98(3):586–603, June 2008.

[NGNP09] Y. Narahari, Dinesh Garg, Ramasuri Narayanam, and Hastagiri Prakash. *Game Theoretic Problems in Network Economics and Mechanism Design Solutions*. Springer, 1 edition, February 2009.

[NR99] Noam Nisan and Amir Ronen. Algorithmic mechanism design. In *Games and Economic Behavior*, pages 129–140, 1999.

[NRTV07] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani. *Algorithmic mechanism design*. Cambridge University Press, 2007.

[OR94] M.J. Osborne and A. Rubinstein. *A course in game theory*. MIT Press, 1994.

[OR98] Martin J. Osborne and Ariel Rubinstein. Games with procedurally rational players, 1998.

[Osb04] Martin J. Osborne. *An Introduction to Game Theory*. Oxford University Press, USA, August 2004.

[Pap01] Christos H. Papadimitriou. Algorithms, games, and the internet. In *In STOC*, pages 749–753. ACM Press, 2001.

[Pap11] Christos H. Papadimitriou. Games, algorithms, and the internet. In Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar, editors, *WWW*, pages 5–6. ACM, 2011.

[Pra06] V. Hastagiri Prakash. A mechanism design approach to resource procurement in computational grids with rational resource providers, 2006.

[PST04] Ryan Porter, Yoav Shoham, and Moshe Tennenholtz. Fair imposition. *Journal of Economic Theory*, 118(2):209–228, October 2004.

[PY94] Christos H. Papadimitriou and Mihalis Yannakakis. On bounded rationality and computational complexity. Technical report, Indiana University, 1994.

[Pyk65] R. Pyke. Spacings. *Journal of the Royal Statistical Society, Series B*, 27:395–449, 1965.

[R.91] Myerson R. *Game Theory*. Cambridge, Harvard University Press, 1991.

[Ras07] E. Rasmusen. *Games and information: an introduction to game theory*. Blackwell Pub., 2007.

[RLEM94] Forsythe Robert, Horowitz Joel L., Savin N. E., and Sefton Martin. Fairness in simple bargaining experiments. *Games and Economic Behavior*, 6(3):347–369, May 1994.

[RM51] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[Rou05] Tim Roughgarden. *Selfish Routing and the Price of Anarchy*. The MIT Press, 2005.

[RZ94] Jeffrey S. Rosenschein and Gilad Zlotkin. Rules of encounter - designing conventions for automated negotiation among computers. In *Rules of Encounter - Designing Conventions for Automated Negotiation among Computers*, 1994.

[SB78] H.S. Stone and S.H. Bokhari. Control of distributed processes. *Computer, vol. 11*, 1978.

[Sch82] Paul J H Schoemaker. The expected utility model: Its variants, purposes, evidence and limitations. *Journal of Economic Literature*, 20(2):529–63, June 1982.

[Sha53] Lloyd S. Shapley. A value for n-person games. In Harold W. Kuhn and Albert W. Tucker, editors, *Contributions to the Theory of Games II*, volume 28 of *Annals of Mathematics Studies*, pages 307–317. Princeton University Press, Princeton, 1953.

[Sim56] Herbert A. Simon. Rational choice and the structure of the environment. *Psychological Review*, 63(2):129–38, 1956.

[Sin87] J.B. Sinclair. Efficient computation of optimal assignments for distributed tasks. *J. Parallel and Distributed Computing, vol. 4*, 1987.

[Sin07] O. Sinnen. *Task scheduling for parallel systems*. Wiley series on parallel and distributed computing. Wiley-Interscience, 2007.

[SM94] S. Selvakumar and C.S.R. Murthy. Static task allocation of concurrent programs for distributed computing systems with processor and resource heterogeneity. *J. Parallel Computing, vol. 20*, 1994.

[Smi39a] N. V. Smirnov. On the estimation of the discrepancy between empirical curves of distribution for two independent samples. *Bulletin of Moscow University.*, 2:3–16, 1939.

[Smi39b] N.V. Smirnov. Estímate of deviation between empirical distribution functions in two independent samples. *Bulletin of Moscow University 2, 3-16*, 1939.

[ST85] C.C. Shen and W.H. Tsai. A graph matching approach to optimal task assignment in distributed computing systems using a minimax criterion. *IEEE Trans. Computers, vol. 34*, 1985.

[Sto77] H.S. Stone. Multiprocessor scheduling with the aid of network flow algorithms. *IEEE Trans. Software Eng., vol. 3*, 1977.

[TO69] Applied Probability Trust. and JSTOR (Organization). Advances in applied probability [electronic resource], 1969.

[TT76] T Nicolaus Tideman and Gordon Tullock. A new and superior process for making social choices. *Journal of Political Economy*, 84(6):1145–59, December 1976.

[Var95] Hal R. Varian. Economic mechanism design for computerized agents. In *In USENIX workshop on Electronic Commerce*, pages 13–21, 1995.

[Ves05] Robert Ferenc Veszteg. Linking decisions with moments. Faculty Working Papers 10/05, School of Economics and Business Administration, University of Navarra, August 2005. available at http://ideas.repec.org/p/una/unccee/wp1005.html.

[Vic61] William Vickrey. Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance*, 16(1):8–37, 1961.

[VNM44] John Von Neumann and Oskar Morgenstern. *Theory of games and economic behavior / by John Von Neumann and Oskar Morgenstern*. Princeton University Press, Princeton :, 1944.

[WS74] R.E.D. Woolsey and H.S. Swanson. Operations research for immediate applications: A quick and dirty manual. *Harper and Row*, 1974.