UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA MATEMÁTICA

COTAS PARA EL PRECIO DE LA ANARQUÍA DE JUEGOS DE SCHEDULING.

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO.

ORLANDO LUIS RIVERA LETELIER

PROFESOR GUÍA:
JOSÉ RAFAEL CORREA HAEUSSLER

MIEMBROS DE LA COMISIÓN:
MARCOS ABRAHAM KIWI KRAUSKOPF
ALEJANDRO JOFRÉ CÁCERES

SANTIAGO DE CHILE
JULIO 2012

RESUMEN DE LA MEMORIA
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL MATEMÁTICO
POR: ORLANDO RIVERA LETELIER
FECHA: 31/07/2012
PROF. JOSE R. CORREA

COTAS PARA EL PRECIO DE LA ANARQUÍA DE JUEGOS DE *SCHEDULING*.

El objetivo principal del presente trabajo de memoria de título es el cálculo de cotas para precio de la anarquía de algunos juegos asociados a problemas de *scheduling*.

Se comienza realizando una revisión general de lo que son los problemas de *scheduling*, un algoritmo de aproximación y la relación que existe entre teoría de juegos y los problemas de *scheduling*. Ahí se identifica el cuociente de aproximación del algoritmo de Smith para problemas de *scheduling*, con el precio de la anarquía de un juego asociado. Se realiza también una revisión de los principales resultados conocidos útiles para el presente trabajo.

Más adelante se calcula el precio de la anarquía para ciertos juegos de *scheduling* donde la función objetivo es la suma ponderada de los tiempos de completación. Se demuestra que en el caso de máquinas idénticas, el precio de la anarquía en estrategias mixtas es 3/2. Se demuestra también que en máquinas paralelas con velocidades, el precio de la anarquía es mayor o igual a 2. Por último, se prueba acá que en el caso en que todos los trabajos tienen el mismo tamaño, el precio de la anarquía del juego de *scheduling* en máquinas paralelas con velocidades y suma ponderada de los tiempos de completación como función objetivo es 1.

Para seguir se estudia el juego asociado al problema de *scheduling* en el cual la función objetivo es la suma de los tiempos de completación, y las máquinas son todas excepto una idénticas entre sí, la máquina restante es de velocidad mayor a las demás, y las máquinas que son más lentas son una cantidad suficientemente grande. Para este juego de *scheduling* se demuestra que el precio de la anarquía es $e/(e-1)$.

Después se estudia el juego mencionado anteriormente en su caso más general, en el cual la cantidad de máquinas lentas no está restringida a ser suficientemente grande. Para este problema se demuestra que el precio de la anarquía está acotado superiormente por 5/3. Se muestra además un problema de programación lineal cuyo óptimo acota superiormente el precio de la anarquía del juego de *scheduling* de máquinas paralelas con velocidades y suma de los tiempos de completación como función objetivo.

Finalmente, se plantea como conjetura que el precio de la anarquía del juego asociado al problema de *scheduling* más general antes mencionado es efectivamente $e/(e-1)$, y se muestran pruebas computacionales que fueron realizadas, con las cuales se justifica el plantear esta conjetura.

AGRADECIMIENTOS

Agradezco en primer lugar a Teresa, mi madre, que ha sido el pilar fundamental en mi vida, y me ha apoyado siempre tanto como ha podido. A Orlando mi padre por haberme hecho confiar en mí mismo desde pequeño, y a Lucho por haber sido mi segundo padre. Lo que soy se lo debo a usted, y terminar este proceso es un logro tanto mío como de ustedes.

Quiero agradecer luego a mi profesor guía José R. Correa, por haberme dado la oportunidad de trabajar con él e involucrarse activamente en esta memoria. Le agradezco por sus muchas ideas y consejos, y especialmente por la paciencia y confianza de permitirme trabajar a mi ritmo. Más que el solo aporte a este trabajo, me apoyó constantemente durante todo este proceso, y me permitió conocer el mundo de la investigación.

A Tjark Vredeveld, quien me recibió en mi estadía en Maastricht durante Septiembre y Octubre de 2011. Sus colaboraciones hicieron posible obtener muchos resultados de este trabajo.

A los profesores Marcos Kiwi y Alejandro Jofré, por haber aceptado ser miembros de mi comisión, y haberme aportado con importantes correcciones para lograr la forma final de esta memoria.

A los académicos Patricio Felmer, Roberto Cominetti, Jaime San Martín y nuevamente Marcos Kiwi, pues cuando me hicieron clases aportaron a mi formación mucho más que las materias de sus cursos, sino que me motivaron profundamente en diversas etapas de la carrera.

A Matías *Pasti* Godoy, quien siempre estuvo ahí en cada momento que necesité a alguien, cada vez que pude utilizar un consejo y no puedo estar más agradecido de haberlo tenido como amigo todo este tiempo.

A Emilio Vilches, con quien hemos surgido de orígenes tan similares que nos podemos comprender como nadie más, y tener una amistad tan sincera como se puede desear.

A Nicolás Hernández, con quien descubrimos juntos buena parte de la maravilla de las matemáticas, siempre me ayudó en mi vida personal y es para mí sujeto de admiración y un ejemplo a seguir como persona.

A mis amigos de la vida, que han estado conmigo en tantos buenos momentos, y apoyándome también en los malos: Gaby Covarrubias, Natalia Bravo, Benjamín Obando, Antonio Lizama, Jocelyn Naranjo, Víctor Carmi, Francisca Montolio, Andrés Fielbaum, Sergio Araneda, Eduardo Zamora y Francisco Muñoz.

A mis demás compañeros del DIM, con quienes compartimos nuestra formación como estudiantes, y mucho más. Destaco especialmente con quienes más compartí: Gonzalo Muñoz, Felipe Serrano, Gonzalo Mena, Gonzalo Contador, Sebastián Donoso, Andrés Zúñiga, Nikolas Tapia, Miguel Romero, Ernesto Araya, Adolfo Henriquez, Christopher Hermosilla, Raimundo Briceño, Pedro Montealegre, Mauro Escobar, Francisco Unda, Italo Cipriano, Víctor Riquelme, Cristobal Guzmán y Omar Larré.

A la Fundación Luksic, a International Association of Chile y a la Liga Protectora de Estudiantes de Santiago por haberme apoyado económicamente durante diversos momentos de mis estudios.

Al profesor Luis Arancibia del Instituto Nacional y a David Painequeo, quienes me presentaron el mundo de las olimpiadas de matemáticas, que terminaron motivando mi pasión por esta área del conocimiento.

Finalmente, agradezco al profesor Jaime Ortega, por su buena voluntad y simpatía como Jefe de Carrera, y a los funcionarios del DIM que han hecho todo lo posible por facilitar cada paso de este proceso, especialmente a: Gladys Cavallone, Eterin Jana, Oscar Mori y Luis Mella.

Índice general

Capítulo 1

Introducción

El presente trabajo de título consiste principalmente en el cálculo del rendimiento de *algoritmos de aproximación* para diversos problemas de *scheduling*. Se considera principalmente el caso en que no existe una administración central de los recursos asociados al problema de *scheduling*, caso en el cual es posible plantear un juego asociado, en el que cada trabajo actúa como un agente independiente tratando de maximizar su propio beneficio. En este caso, se busca calcular una medida de la pérdida de eficiencia debido a la falta de administración central, conocida como *precio de la anarquía* del juego.

1.1. Problemas de *scheduling*

Un problema de *scheduling* es un problema de optimización que consiste en asignar una cantidad de recursos para procesar cierta cantidad de trabajo de forma eficiente. Este tipo de problemas aparecen en situaciones diversas como por ejemplo en el caso de una CPU que debe procesar diversas tareas exigidas por el usuario, o una fábrica que debe ensamblar distintos tipos de productos con un conjunto limitado de líneas de producción, o una planta embotelladora que debe envasar distintos tipos de líquidos usando las mismas máquinas para todos ellos.

1.1.1. Clasificación de los Problemas de *scheduling*

Existe una diversidad de problemas de *scheduling*, dependiendo de las características de los trabajos, del ambiente y propiedades de las máquinas, de lo que se desea optimizar, entre otras propiedades. Por ejemplo, un trabajo podría ser interrumpido para seguir siendo procesado más tarde en la misma u otra máquina, podría ser dividido entre un conjunto de máquinas que lo procesen al mismo tiempo, podría tener restricciones sobre el momento en el que se puede empezar a procesar, o una fecha límite antes de la cual debe ser procesado. Así también, las máquinas podrían estar restringidas a poder procesar solo un subconjunto

de todos los trabajos, podrían ser idénticas entre sí, tener distintas velocidades, o no tener ninguna relación en el tiempo que demoran en procesar los trabajos. Como función objetivo del problema de *scheduling* podría quererse minimizar el tiempo en el cual se terminaron de procesar todos los trabajos, el tiempo promedio en el que los trabajos estuvieron listos, la cantidad de trabajos procesados dentro de la fecha límite requerida, entre otras.

Existe una notación estándar para problemas de *scheduling*, la llamada notación de tres campos, introducida por Grahams, Lawler, Lenstra y Rinnooy Kan en 1979 [3]. En esta notación, un problema de *scheduling* se representa por una expresión de la forma $\alpha|\beta|\gamma$, donde los campos α, β y γ dependerán de las características propias del problema de *scheduling* considerado.

Campo α

El primer campo representa el ambiente de máquinas en el que se está trabajando. Algunos de los valores posibles para este campo son:

- $\alpha = 1$: *Una sola máquina.* Una sola máquina debe procesar todos los trabajos, y lo que se debe decidir es únicamente el orden en el que se procesarán los trabajos.
- $\alpha = P$: *Máquinas paralelas idénticas.* Todas las máquinas son indistinguibles entre ellas, y cada trabajo tiene asociado el tiempo que demora en ser procesado en cualquiera de las máquinas.
- $\alpha = Q$: *Máquinas paralelas relacionadas.* Cada máquina posee una velocidad a la que trabaja y cada trabajo posee un tamaño. El tiempo que demora un trabajo en ser procesado en una máquina es su tamaño dividido en la velocidad de la máquina.
- $\alpha = R$: *Máquinas paralelas no relacionadas.* Cada trabajo tiene una lista que incluye el tiempo que demoraría en ser procesado en cada máquina, en caso de ser asignado ahí.

Campo β

El segundo campo representa las características de los trabajos. El campo podría dejarse vacío, así como también podría tener varios parámetros juntos, caso en el cual los trabajos cumplen todas las características descritas:

- $\beta = \text{pmtn}$: *Preemtion.* Los trabajos pueden ser interrumpidos para ser procesados más tarde, y también pueden ser divididos, para ser procesados en varias máquinas a la vez.
- $\beta = r_j$: *Tiempos de disponibilidad.* Cada trabajo tiene un tiempo a partir del cual puede ser comenzado a procesar, antes del cual está a la espera.

- $\beta =$ prec: *Precedencias.* Existe un orden parcial sobre los trabajos, de modo que si un trabajo es posterior a otro bajo ese orden, no puede ser comenzado a procesar antes de que el otro haya terminado.
- $\beta = p_j = 1$: *Trabajos iguales.* Todos los trabajos tienen el mismo tamaño, que sin pérdida de generalidad se puede asumir que es igual a 1.

Campo γ

El tercer campo representa la función objetivo a minimizar en el problema de *scheduling*. Algunos ejemplos comunes:

- $\gamma = C_{\text{máx}}$: *Makespan.* El objetivo a minimizar es el tiempo en el cual todos los trabajos ya han terminado de ser procesados. Es una de las funciones más comúnmente considerada.
- $\gamma = \sum C_j$: *Suma de los tiempos de completación.* El objetivo a minimizar es la suma de los tiempos en los cuales cada trabajo está listo. Equivalentemente, se minimiza el tiempo promedio en el que los trabajos fueron terminados de procesar.
- $\gamma = \sum w_j C_j$: *Suma ponderada de tiempos de completación.* Cada trabajo tiene un peso positivo asociado, y el objetivo es minimizar la suma de los tiempos en que los trabajos son completados, ponderado por el peso de cada trabajo.

1.1.2. Definiciones

El caso tratado en esta memoria es aquél en el que hay un conjunto de máquinas que trabajando en paralelo deben procesar un conjunto de trabajos de forma secuencial. Cada máquina puede procesar a lo más un trabajo a la vez, y cada trabajo debe ser procesado en una sola máquina sin poder ser interrumpido ni subdividido para ser procesado en distintas máquinas.

Formalmente, una instancia de un problema de *scheduling* consiste en un conjunto de n trabajos $\mathcal{J}$ que deben ser procesado en un conjunto de m máquinas $\mathcal{M}$. Cada trabajo $J_j \in \mathcal{J}$ tiene una cantidad $p_j > 0$ de trabajo a ser procesado en alguna de las máquinas y un peso $w_j > 0$, el cual es una medida de la importancia de ese trabajo. Cada máquina M_i tendrá una velocidad s_i a la que trabajará, de modo que si el trabajo J_j es procesado en la máquina M_i, el tiempo que tardará la máquina en procesar este trabajo será $\frac{p_j}{s_i}$.

Definición 1.1. Una instancia de un problema de *scheduling* se representará por un par ordenado $I = (\mathcal{J}, \mathcal{M})$, donde la primera coordenada representa el conjunto de trabajos $\mathcal{J} = (J_1, J_2, \ldots, J_n)$, y la segunda coordenada representará el conjunto de máquinas

$\mathcal{M} = (M_1, M_2, \ldots, M_m)$. Cada trabajo será representado por un par ordenado consistente en su tamaño y su peso $J_j = (p_j, w_j)$, y cada máquina será representada por su velocidad $M_i = s_i$.

Definición 1.2. Un *schedule* S de una instancia $I = (\mathcal{J}, \mathcal{M})$ de un problema de *scheduling*, será una función que asigne a cada trabajo la máquina en que será procesado y el tiempo en el cual comenzará a ser procesado, de modo que no se viole la restricción de que cada máquina puede procesar a lo más un trabajo a la vez. Se asumirá que en las máquinas no habrán tiempos muertos, es decir, cada máquina procesará los trabajos de forma continuada hasta terminar con todos los que tenía asignados. Se denotará por $S(j)$ a la máquina en la que se procesará el trabajo J_j en el *schedule* S.

El tiempo en el cual el trabajo J_j termina de ser procesado en el *schedule* S se conoce como tiempo de completación del trabajo, y se denotará por ${C_j}^S(I)$, o simplemente C_j cuando se subentienda el *schedule* y la instancia de la que se trate.

Definición 1.3. Dada una instancia $I = (\mathcal{J}, \mathcal{M})$ de un problema de *scheduling* con función objetivo $\sum w_j C_j$, el costo de un *schedule* S de la instancia será:

$$C_S(I) := \sum_{J_j \in \mathcal{J}} w_j {C_j}^S.$$

Se denominará por $C_{opt}(I)$ al mínimo costo posible de alcanzar por un *schedule* de la instancia I. Notar que este valor es alcanzado por algún *schedule* pues existe una cantidad finita de *schedules* posibles de una instancia.

1.2. Algoritmos de aproximación

En ocasiones, para un problema de optimización puede no requerirse un algoritmo que encuentre el óptimo del problema, ya sea porque los algoritmos óptimos tienen un mal tiempo de ejecución (por ejemplo en problemas NP-duros), o por ser los algoritmos óptimos difíciles de encontrar. En esos casos, puede ser útil usar *algoritmos de aproximación*.

Definición 1.4. Un algoritmo de aproximación para un problema de optimización es un algoritmo que entrega una solución factible, pero no necesariamente óptima.

Idealmente, un buen algoritmo de aproximación debe entregar una solución que cercana al óptimo. Una medida de la calidad de un algoritmo de aproximación es que tan lejos del óptimo se encuentra el resultado entregado por el algoritmo.

Definición 1.5. Dado un problema de minimización con una función de costos c, un algoritmo $\mathcal{A}$ es una ρ-aproximación del problema si para toda instancia I del problema se tiene que el resultado entregado por el algoritmo $\mathcal{A}(I)$ cumple que:

$$c(\mathcal{A}(I)) \leq \rho \cdot OPT(I),$$

donde $OPT(I)$ es el costo óptimo de la instancia.

Al menor valor ρ que cumple esto se le llama *garantía*, *factor de aproximación* o *cuociente de aproximación* del algoritmo.

Ejemplo 1.1. Un problema clásico de optimización combinatorial es *Vertex Cover*. Este es un problema NP-completo [7], por lo cual no se conoce un algoritmo óptimo que se ejecute en tiempo polinomial.

La versión como problema de optimización de *Vertex Cover* es como sigue: dado un grafo, encontrar un recubrimiento de nodos de tamaño minimal, donde un recubrimiento de nodos de un grafo es un subconjunto C de vértices, de modo que todo arco del grafo sea adyacente a algún vértice de C.

Para este problema existe un algoritmo de aproximación simple, que consiste en lo siguiente: se empieza con C vacío, se toma una arista cualquiera y se agregan sus dos vértices a C. Luego se borran del grafo esos dos vértices y todas las aristas que sean adyacente a alguno de ellos, para después iterar hasta que no queden aristas restantes en el grafo. Esto entregará como resultado un recubrimiento de nodos, que no necesariamente será de tamaño minimal pero cuyo tamaño será a lo más 2 veces el tamaño de un recubrimiento de nodos óptimo. Para ver esto, basta observar que cada vez que se agregan los dos vértices de una arista uno de ellos tiene que estar en el vertex cover óptimo, pues aquella arista debe ser cubierta por algún vértice de ese vertex cover.

Ejemplo 1.2. Para el problema de *scheduling* $Q||\sum w_j C_j$, se considera el siguiente algoritmo, conocido como algoritmo de Smith [11]. Primero se ordenan de mayor a menor los trabajos por su *radio de Smith* $r_j := \frac{w_j}{p_j}$. Sin pérdida de generalidad, se asumirá que luego de ordenar los trabajos quedaron de forma tal que el primer trabajo es J_1, seguido de J_2, hasta el último trabajo que será J_n. Luego se toma el primer trabajo que aun no ha sido asignado a una máquina, llámese J_k, para cada máquina M_i se determina el siguiente número:

$$\ell_i^k := \frac{1}{s_i} \sum_{j=1}^{k-1} p_j x_{i,j} + \frac{p_k}{s_i},$$

donde $x_{i,j}$ será un número que tomará el valor 1 si el trabajo j fue asignado a la máquina i, y tomará el valor 0 en caso que no. El trabajo J_k es asignado a la máquina que minimice el valor

ℓ_i^k. Finalmente, cada máquina procesará los trabajos que le han sido asignados según el orden mayor radio primero. Este algoritmo es natural de utilizar debido a que en el caso en que se trabaja en un ambiente con una sola máquina el algoritmo es óptimo. Ver en el Teorema 2.1

1.3. Juego asociado a un problema de *scheduling*

Dada una instancia de un problema de *scheduling* $I = (\mathcal{J}, \mathcal{M})$, se puede considerar un juego asociado a él. En este juego, cada trabajo representa a un jugador (por lo que $\mathcal{J}$ será el conjunto de jugadores), queriendo minimizar su propio costo. El conjunto de estrategias puras de cada jugador será $\mathcal{M}$, es decir, elegirá una máquina donde ser procesado. Cada máquina tendrá una lista de prioridades (un orden total sobre todos los trabajos), de modo que si un conjunto de trabajos elige una misma máquina, la máquina procesará esos trabajos en el orden de aquella lista. Esta lista será conocida por los jugadores y afectará en la estrategia que elijan. En principio los radios de Smith de distintos trabajos podrían ser iguales, sin embargo se asume implícitamente que existe una forma de romper empates, la cual comparten todas las máquinas.

Durante la presente memoria se asumirá que todas las máquinas tendrán la misma lista de prioridades. Se denotará por $a \prec b$ cuando el jugador J_a tenga una mejor prioridad que J_b. Se asumirá también que la lista que comparten todas las máquinas será consistente con el orden *WSPT* (*weighted shortest processing time*), esto quiere decir que si $j_1 \prec j_2$, entonces $r_{j_1} \leq r_{j_2}$.

Sin pérdida de generalidad, se asumirá que las instancias serán tales que:

$$1 \prec 2 \prec 3 \prec \ldots \prec n.$$

Definición 1.6. Se definirá un perfil de estrategias puras de los jugadores como un vector $S = (S_1, S_2, \ldots, S_n)$ en el cuál S_j es la máquina elegida por el jugador J_j.

Observación 1.1. Cada perfil de estrategias puras S tiene asociado un *schedule*, en el cual el trabajo J_j es procesado en la máquina S_j, de modo que cada máquina M procesa todos los trabajos J_j tales que $S_j = M$ según el orden *WSPT*.

Además, dado un *schedule* S de una instancia I de un problema de *scheduling* con función objetivo $\sum w_j C_j$, en el que cada máquina procesa los trabajos asociado en un orden consistente con *WSPT*, existe un orden total sobre los trabajos, consistente con *WSPT* (una

forma de romper empates entre trabajos distintos con igual radio de Smith), de modo que $(S(1), S(2), \ldots, S(n))$ es un perfil de estrategias que genera exactamente el *schedule* S. Es decir, existe una correspondencia entre *schedule*s donde cada máquina ordena respecto a *WSPT* y perfiles de estrategias de la instancia del juego asociado.

En particular, en un *schedule* óptimo, el orden en que los trabajos son procesados en cada máquina debe ser óptimo, lo cual se verá en la Sección 2.1 que debe corresponder a ordenar por *WSPT*. Por esto, todo óptimo corresponde también a un perfil de estrategias.

Nota 1.2. Dado un perfil de estrategias S de una instancia de un juego de *scheduling*, se hará abuso de notación, y se usará indiferentemente S para denotar también al *schedule* asociado a ese perfil de estrategias.

Definición 1.7. Se define el costo individual del jugador J_j en un perfil de estrategias S como el tiempo de completación ponderado del trabajo en el *schedule* asociado a S. Es decir, cada jugador buscará minimizar el valor de $w_j {C_j}^S$.

Observación 1.3. Para un jugador es equivalente minimzar $w_j {C_j}^S$ o ${C_j}^S$, pues w_j es una constante para él. Sin embargo, se usará $w_j {C_j}^S$ como costo del jugador J_j pues así la función objetivo del problema de *scheduling* coincide con la suma de los costos individuales de los jugadores en el juego asociado, la cual es una función de bienestar comúnmente utilizada en economía.

Observación 1.4. Dado que todas las máquinas tienen el mismo orden de prioridades consistente con *WSPT* (que sin pérdida de generalidad asumiremos es $1 \prec 2 \prec \ldots \prec n$), el costo del jugador J_j depende únicamente de su estrategia y de las estrategias elegidas por los jugadores $J_1, \ldots, J_{j-1}$.

Definición 1.8. Un perfil de estrategias N se dirá equilibrio de Nash en estrategias puras (o simplemente equilibrio de Nash cuando se subentienda), si se cumple que para todo jugador J_j se tiene que la estrategia $N(j)$ minimiza su costo individual, dado que todos los otros jugadores j' eligieron la estrategia $N(j')$. Es decir, ningún jugador tiene incentivos a cambiar de estrategia si todos los otros jugadores se mantienen en N.
Se denotará por $\mathcal{N}(I)$ al conjunto de equilibrios de Nash de una instancia I.

Definición 1.9. Dado un perfil de estrategias S de una instancia I del juego $Q||\sum w_j C_j$, se define el costo social de S, denotado por $C_S(I)$, como la suma de los costos individuales

de los jugadores cuando juegan con las estrategias de S, es decir:

$$C_S(I) := \sum_{j=1}^{n} w_j {C_j}^S(I).$$

Se definirá a continuación una noción de la pérdida de eficiencia por jugar en un equilibrio en vez de forma óptima. Este concepto fue introducido por primera vez por Koutsoupias y Papadimitriou en 1999 [9].

Definición 1.10. El precio de la anarquía en estrategias puras (o simplemente precio de la anarquía cuando se entienda que se toman solo estrategias puras) de una instancia I de un juego, denotado $\mathbf{PPoA}(I)$, es la razón entre el mayor costo posible de obtener por un equilibrio de Nash, y el costo óptimo de la instancia. Es decir:

$$\mathbf{PPoA}(I) := \max_{N \in \mathcal{N}} \frac{C_N(I)}{C_{opt}(I)}.$$

Definición 1.11. El precio de la anarquía en estrategias puras de un conjunto de instancias $\mathcal{I}$ de un juego es:

$$\mathbf{PPoA}(\mathcal{I}) := \sup_{I \in \mathcal{I}} \mathbf{PPoA}(I).$$

Observación 1.5. Existe una correspondencia trivial entre *schedules* generados por el algoritmo de Smith de una instancia de un problema de *scheduling* y equilibrios de Nash de la instancia del juego asociado. Además, el costo de un *schedule* de una instancia del juego, coincide con el costo social del perfil de estrategias asociado. De esta forma es claro que si consideramos $\mathcal{I}$ como todas las instancias del juego de *scheduling*, se tendrá que $\mathbf{PPoA}(\mathcal{I})$ será igual al radio de aproximación del algoritmo de Smith en el problema de *scheduling* considerado.

Dado un juego, se puede considerar también para un jugador un conjunto de estrategias mixtas. Cuando se juegan estrategias mixtas, cada jugador elige una probabilidad de jugar cada una de sus estrategias puras de modo que la suma de estas sea 1, y luego cada jugador forma de forma aleatoria una estrategia pura elegida con la distribución de probabilidad definida en su estrategia mixta.

Definición 1.12. Se define el conjunto de estrategias mixtas de un jugador, como el conjunto de vectores de probabilidad sobre el conjunto de estrategias puras de ese jugador. Un vector de probabilidad es un vector de números entre 0 y 1, tal que la suma de todas sus componentes es 1.

Cabe notar que las estrategias mixtas generalizan a las estrategias puras, en el sentido de que una estrategia pura se puede recuperar al poner probabilidad 1 en una coordenada, y probabilidad 0 en todas las demás.

Un perfil de estrategias mixtas será un vector con las estrategias mixtas de todos los jugadores. Dado un perfil de estrategias mixtas el costo de un jugador será el la esperanza del costo sobre las posibles realizaciones de estrategias puras, definidas por las probabilidades de las estrategias mixtas.

El costo de un perfil de estrategias mixtas será la suma ponderada de los costos individuales de los jugadores, o simplemente la suma de los costos de los jugadores, dependiendo de la función objetivo que se esté usando en el juego.

Un perfil de estrategias mixtas será un equilibrio de Nash si es que fijando las estrategias jugadas por los demás jugadores, cada jugador está minimiza su costo esperado. Esto es equivalente a que si se fijan las estrategias de los demás jugadores, cada jugador juegue con probabilidad mayor que 0 en las estrategias puras que minimizarían su costo. Es decir, si una estrategia pura es dominada por otra dado lo que los demás jugadores eligieron (tiene un costo estrictamente mayor que la otra), entonces la que tiene el mayor costo debe ser jugada con probabilidad 0 en la estrategia mixta.

Definición 1.13. Se definirá el precio de la anarquía en estrategias mixtas de una instancia I, denotado simplemente por $\mathbf{PoA}(I)$, como el cuociente entre el mayor costo posible de obtener por un equilibrio de Nash en estrategias mixtas, y el costo óptimo del juego. Es decir, que si llamamos $\mathcal{N}_{mix}$ al conjunto de equilibrios de Nash en estrategias mixtas del juego, entonces se tendrá que:

$$\mathbf{PoA}(I) := \max_{N \in \mathcal{N}_{mix}} \frac{\mathbb{E}(C_N(I))}{C_{opt}(I)}.$$

Observación 1.6. Cabe notar que cada equilibrio de Nash en estrategias puras es también un equilibrio de Nash en estrategias mixtas (al poner probabilidad 1 en la estrategia jugada y 0 en las demás), por lo cual el precio de la anarquía en estrategias mixtas será siempre mayor o igual al precio de la anarquía en estrategias puras.

Capítulo 2

Resultados preliminares

En este capítulo se presentan algunos resultados previos íntimamente relacionados con el trabajo de esta memoria.

2.1. Scheduling en una sola máquina

En el Ejemplo 1.2 se vio el algoritmo de Smith, el cual es el principal algoritmo de aproximación acá estudiado.

Teorema 2.1 (Smith 1956 [11])**.** *El algoritmo de Smith es óptimo para el problema de* scheduling $1||\sum w_j C_j$.

Esto muestra que efectivamente todo *schedule* óptimo de una instancia de un problema de *scheduling* debe ordenar en cada máquina los trabajos según el orden *WSPT*, por lo que corresponde a un perfil de estrategias de la instancia del juego asociado al problema de *scheduling*, definiendo apropiadamente la regla para romper empate que usan las máquinas.

2.2. Algoritmo óptimo para $Q||\sum C_j$

Cuando se considera el caso en que $w_j = 1$ para todos los trabajos, es decir, no hay pesos en la función objetivo, es posible encontrar un *schedule* óptimo de la instancia en tiempo polinomial. El algoritmo *MFT* (*minimun mean flow time*), propuesto por Horowitz y Sahni en 1976 [6], toma tiempo $\mathcal{O}(n \log mn)$ en ejecutarse en una instancia de n trabajos y m máquinas. Este algoritmo debe elegir únicamente la máquina en que será procesado cada trabajo, pues dentro de cada máquina lo óptimo es ordenar los trabajos según *WSPT*, que en este caso al no haber pesos se convierte simplemente en procesar los trabajos en el orden de más cortos primero (*SPT*).

Se asume primero que los trabajos están ordenados de forma que $1 \prec 2 \ldots \prec n$, lo que es equivalente a $p_1 \leq p_2 \leq \ldots \leq p_n$. El algoritmo *MFT* en esa instancia hará lo siguiente:

- Definir para cada máquina M_i el valor z_i como el número de trabajos asignados hasta el momento.
- Asignar el trabajo más grande que aun no ha sido asignado, a una máquina M_i que minimice el valor $\frac{1+z_i}{s_i}$.
- Actualizar los valores de z_i, e iterar hasta que todos los trabajos sean asignados.
- Procesar los trabajos en cada máquina según el orden *SPT*.

Teorema 2.2 (Horowitz y Sahni 1976)**.** *El algoritmo* MFT *es óptimo para el problema* $Q||\sum C_j$*. Más aun, todo* schedule *óptimo de una instancia puede ser obtenido a través del algoritmo* MFT*, eligiendo de alguna forma indicada a qué máquina asignar un trabajo cuando hay un empate en el valor de* $\frac{1+z_i}{s_i}$*.*

La demostración de la optimalidad del algoritmo puede ser encontrada en [6]. La demostración de que todo *schedule* óptimo se puede obtener como salida del algoritmo *MFT* puede ser encontrada en [5].

Observación 2.1. En este algoritmo, la máquina en la que un trabajo es procesado y la posición donde se procesa en esa máquina depende exclusivamente del lugar que ocupa el trabajo al ser ordenados por tamaño, y no del tamaño mismo del trabajo.

2.3. Cotas del precio de la anarquía para $Q||\sum C_j$

Se muestra en esta Sección un conjunto de instancias que proveen una cota inferior en el precio de la anarquía. Estas instancias son fundamentales a través de todo el trabajo.

Teorema 2.3. *El precio de la anarquía del juego de* scheduling $Q||\sum C_j$ *es al menos* $\frac{e}{e-1}$*.*

Demostración. Se mostrará un conjunto de instancias, de modo que el supremo en el precio de la anarquía de esas instancias es $\frac{e}{e-1}$. De este modo, el precio de la anarquía no puede ser inferior a ese número. Las instancias que se mostrarán, fueron propuestas por Hoeksma y Uetz en 2010 [4].

Se considera el conjunto de instancias $\left\{I_{n,s}\right\}_{n,s\in\mathbb{N},n>s}$, cada una con n trabajos y $n-s+1$ máquinas. La máquina M_1 tiene velocidad $s_1 = s$, y para $j \geq 2$, las máquinas M_j tienen velocidad $s_j = 1$. Los tiempos de proceso de los trabajos serán:

$$p_j = \begin{cases} 1, & j \leq s, \\ \left(\frac{s}{s-1}\right)^{j-s}, & s+1 \leq j \leq n. \end{cases}$$

Así, aplicando el algoritmo *MFT*, se tendrá que los s trabajos más grandes serán asignados a M_1, mientras que los demás trabajos serán cada uno el único trabajo procesado en una de las máquinas de velocidad 1.

El schedule óptimo se ilustra en la siguiente figura:

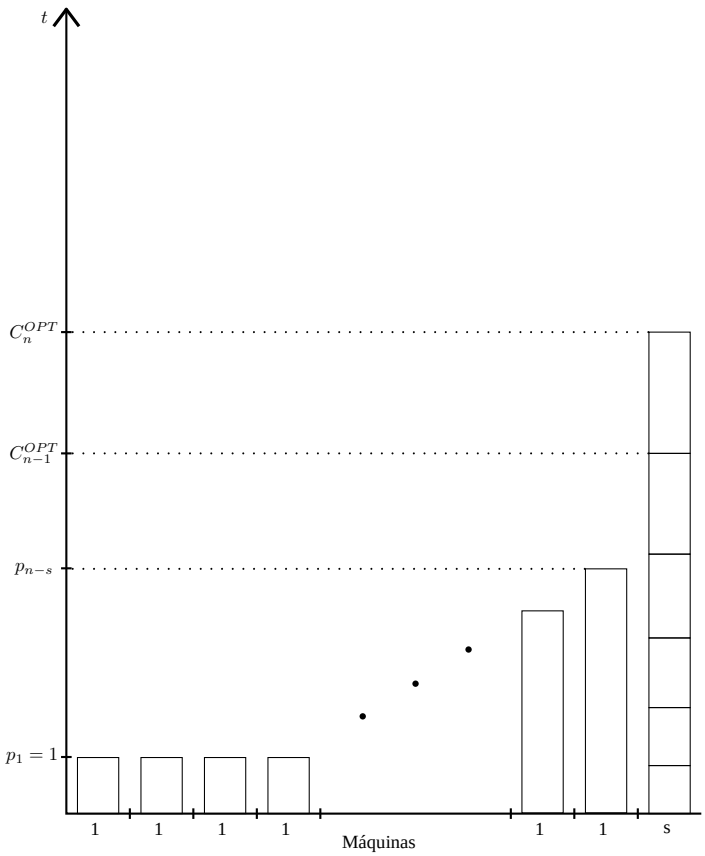


Figura 2.1: Schedule Óptimo.

De este modo, en el óptimo el tiempo de completación de los primeros $n - s$ trabajos será igual a su tiempo de proceso, es decir:

$$C_j{}^{OPT} = p_j, \qquad \text{para todo } 1 \leq j \leq n - s.$$

Los últimos s trabajos serán procesados en M_1, de menor a mayor. Como los trabajos ya están ordenados de menor a mayor, se tendrá que:

$$C_j^{OPT} = \frac{1}{s} \sum_{k=n-s+1}^{j} p_k, \qquad \text{para todo } n-s+1 \leq j \leq n.$$

Así, el costo del óptimo será:

$$\begin{aligned} C_{opt}(I_{n,s}) &= \sum_{j=1}^{s} 1 + \sum_{j=s+1}^{n-s} \left(\frac{s}{s-1}\right)^{j-s} + \sum_{j=n-s+1}^{n} \frac{1}{s} \sum_{k=n-s+1}^{j} \left(\frac{s}{s-1}\right)^{k-s} \\ &= s\left(\frac{s}{s-1}\right)^{n-s} - s\left(\frac{s}{s-1}\right)^{n-2s}. \end{aligned}$$

Se verá ahora que el perfil de estrategias N en el cual todos los trabajos juegan en M_1 es un equilibrio de Nash. En efecto, para los trabajos J_j tales que $1 \leq j \leq s$ el costo en esta estrategia será $C_j^N = \frac{j}{s}$, que es menor o igual a 1 (el costo de estar en una de las máquinas de velocidad 1).

Para los trabajos J_j tales que $j \geq s+1$ se tendrá que:

$$\begin{aligned} C_j^N &= 1 + \sum_{k=s+1}^{j} \frac{p_k}{s} \\ &= 1 + \frac{1}{s} \sum_{k=s+1}^{j} \left(\frac{s}{s-1}\right)^{k-s} \\ &= 1 + \frac{1}{s} \frac{\left(\frac{s}{s-1}\right)^{j+1-s} - \left(\frac{s}{s-1}\right)^{s+1-s}}{\left(\frac{s}{s-1}\right) - 1} \\ &= \left(\frac{s}{s-1}\right)^{j-s} \\ &= p_j. \end{aligned}$$

Luego esos trabajos son indiferentes a la máquina en la que estén, por lo cual el perfil de estrategias N es un equilibrio de Nash.

El equilibrio de Nash N se verá en las máquinas como en la siguiente figura:

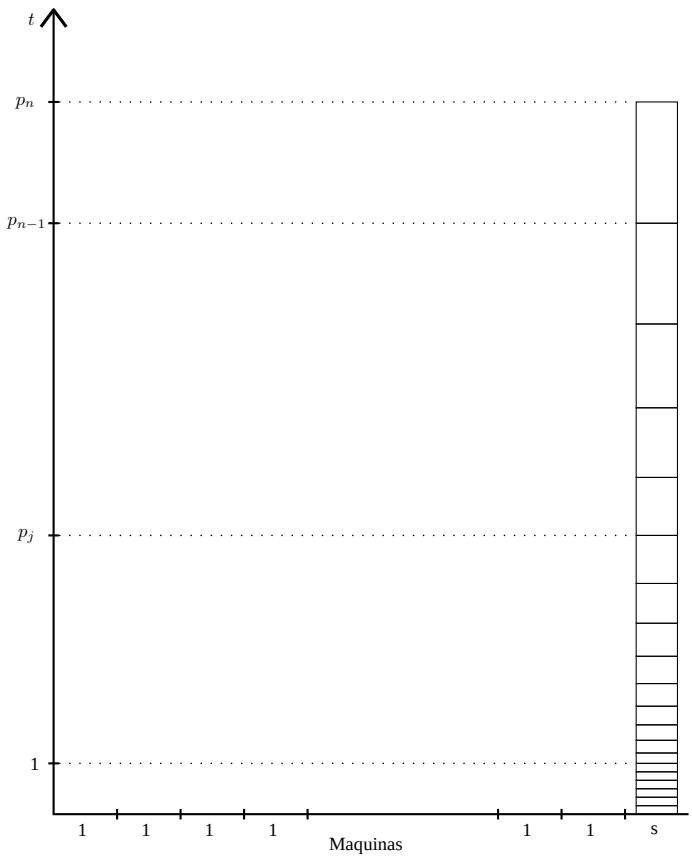


Figura 2.2: Equilibrio de Nash

Así, el costo de N será:

$$\begin{aligned} C_N(I_{n,s}) &= \sum_{j=1}^{N} {C_j}^N \\ &= \sum_{j=1}^{s} \frac{j}{s} + \sum_{j=s+1}^{n} p_j \\ &= \frac{s+1}{2} + \sum_{j=s+1}^{n} \left(\frac{s}{s-1}\right)^{j-s} \end{aligned}$$

$$
\begin{aligned}
&= \frac{s+1}{2} + \frac{\left(\frac{s}{s-1}\right)^{n-s+1} - \frac{s}{s-1}}{\frac{s}{s-1} - 1} \\
&= \frac{s+1}{2} + s\left(\frac{s}{s-1}\right)^{n-s} - s \\
&= s\left(\frac{s}{s-1}\right)^{n-s} + \frac{1-s}{2}.
\end{aligned}
$$

De este modo se tendrá que

$$
\begin{aligned}
\mathbf{PPoA}(I_{n,s}) = \frac{C_N(I_{n,s})}{C_{opt}(I_{n,s})} &= \frac{s\left(\frac{s}{s-1}\right)^{n-s} + \frac{1-s}{2}}{s\left(\frac{s}{s-1}\right)^{n-s} - s\left(\frac{s}{s-1}\right)^{n-2s}} \\
&\leq \frac{s\left(\frac{s}{s-1}\right)^{n-s}}{s\left(\frac{s}{s-1}\right)^{n-s} - s\left(\frac{s}{s-1}\right)^{n-2s}} \\
&= \frac{\left(\frac{s}{s-1}\right)^{-s}}{\left(\frac{s}{s-1}\right)^{-s} - \left(\frac{s}{s-1}\right)^{-2s}} \\
&\xrightarrow{s\to\infty} \frac{e}{e-1}
\end{aligned}
$$

□

Teorema 2.4 (Hoeksma y Uetz 2011)**.** *El precio de la anarquía en estrategias mixtas del juego de* scheduling $Q||\sum C_j$ *es a lo más 2.*

La demostración puede ser encontrada en [5].

2.4. Peor caso para el juego $Q||\sum w_jC_j$

El siguiente resultado dice que cuando queremos calcular el precio de la anarquía de un conjunto de instancias suficientemente grande en el juego asociado al problema de *scheduling* $Q||\sum w_jC_j$, nos podemos restringir a encontrar el precio de la anarquía de aquellas instancias en las cuales todos los trabajos cumplen que $p_j = w_j$. Más aún, re-escalando los trabajos, podemos asumir que para todo trabajo se tiene que $p_j = w_j = 1$.

Lema 2.5. *Sea $\mathcal{I}$ un conjunto de instancias del problema $Q||\sum w_jC_j$, cerrado bajo la operación de tomar un subconjunto de trabajos. Luego, si $\mathbf{PPoA}(\mathcal{I}) \leq \alpha$ para la función de costo social $\sum p_jC_j$, se tiene también que $\mathbf{PPoA}(\mathcal{I}) \leq \alpha$ para la función objetivo $\sum w_jC_j$.*

Demostración. Es posible encontrar una versión más general de este lema, en el que se considera el caso de máquinas no relacionadas, en [1] por Correa y Queyranne. También es posible encontrar una demostración en [8] o en [10]. □

2.5. Precio de la anarquía para $P||\sum w_jC_j$

El problema de *scheduling* $P||\sum w_jp_j$ es NP-completo [2], por lo cual se vuelve más interesante aun encontrar un algoritmo con un radio de aproximación bajo. Este es el caso del algoritmo de Smith, que para este problema da un factor de aproximación de un poco más que 1,207.

Teorema 2.6 (Kawaguchi y Kyan 1986 [8])**.** *El algoritmo de Smith es una* $\frac{1+\sqrt{2}}{2}$*-aproximación del problema* $P||\sum w_jC_j$*.*

Además de la demostración original en [8], existe una demostración más simple y reciente debida a Schwiegelshohn [10].

Se puede notar que esto dice que el precio de la anarquía en estrategias puras del juego asociado al problema de *scheduling* $P||\sum w_jC_j$ es $\frac{1+\sqrt{2}}{2} \approx 1{,}2071$. Esto es interesante, pues cuando se consideran estrategias mixtas el precio de la anarquía aumenta a $\frac{3}{2}$.

Capítulo 3

Juegos de *scheduling* con función objetivo $\sum w_j C_j$

En este capítulo se presenta una serie de resultados de problemas de *scheduling* en los que la función objetivo es $\sum w_j C_j$.

3.1. El caso de máquinas idénticas

El siguiente resultado mostrará que en el caso de máquinas idénticas se logra efectivamente una diferencia en el precio de la anarquía si es que se consideran también estrategias mixtas. Esto pues el resultado de Kawaguchi y Kyan [8] muestra que si solo se consideran estrategias puras el precio de la anarquía es aproximadamente 1.207, mientras que aumenta a 1.5 si se consideran estrategias mixtas.

Teorema 3.1. *El precio de la anarquía en estrategias mixtas para el juego asociado al problema de* scheduling $P||\sum w_j C_j$ *es* $\frac{3}{2}$.

Demostración. Se probará primero que el precio de la anarquía es mayor o igual a $\frac{3}{2}$. Para esto, sea I una instancia del juego con n máquinas y n trabajos, cada uno de ellos cumpliendo $p_j = w_j = 1$. Consideremos el perfil de estrategias mixtas N en el cual cada jugador elige con probabilidad $\frac{1}{n}$ cada una de las máquinas. La estrategia N es un equilibrio de Nash pues para cada jugador el costo esperado de jugar en cualquier máquina es el mismo, ya que todos los jugadores con mejor prioridad han jugado de forma indiferente entre todas las máquinas. Por esto, en este equilibrio de Nash, el costo esperado del jugador j será $1 + \frac{j-1}{n}$. Luego, el costo total de este equilibrio de Nash será:

$$C_N(I) = \sum_{j=1}^{n} 1 + \frac{j-1}{n} = n + \frac{1}{n}\frac{(n-1)n}{2} = \frac{3n-1}{2}.$$

El óptimo de esta instancia se dará cuando los n trabajos sean asignados a n máquinas distintas, caso en el cual el costo de cada trabajo será 1, por lo que el costo óptimo será n. Por esto, el precio de la anarquía de la instancia I será:

$$\mathbf{PPoA}(I) = \frac{C_N(I)}{C_{opt}(I)} = \frac{\frac{3n-1}{2}}{n} = \frac{3}{2} - \frac{1}{2n}.$$

Al hacer n tan grande como se desee, se muestra que el precio de la anarquía del juego no puede ser menor a $\frac{3}{2}$.

Se probará ahora, que el precio de la anarquía de este juego es a lo más $\frac{3}{2}$. Para esto, observemos que por el resultado visto en el Lema 2.5, nos podemos restringir a instancias en las que los trabajos cumplen que $p_j = w_j$. Sea $I = (\mathcal{J}, \mathcal{M})$ una instancia de este tipo, y suponemos además sin pérdida de generalidad que la lista de preferencias que las máquinas tienen sobre los trabajos es: $1 \prec 2 \prec \ldots \prec n$.

Sea N_{mix} el peor equilibrio de Nash en estrategias mixtas para la instancia I. Denotamos por $x_{i,j}$ a la probabilidad con que el jugador J_i juega en la máquina M_i en N_{mix}. A partir de este equilibrio de Nash N_{mix}, definimos una nueva instancia $\widetilde{I}$ de la siguiente forma: Por cada trabajo J_j y por cada máquina M_i se crea un trabajo denominado (i, j) con $w_{i,j} = p_{i,j} = x_{i,j} p_j$. Finalmente, de los mn trabajos creados de esta forma se eliminan todos los trabajos (i, j) tales que $p_{i,j} = 0$.

Definimos además una lista de prioridades sobre los jugadores (como $w_{i,j} = p_{i,j}$, cualquier orden sobre los trabajos es consistente con $WSPT$). Esta lista es tal que si $j \prec j'$, entonces para cualquier i, i' se tendrá que $(j, i) \prec (j', i')$. No se necesitará comparar trabajos que tengan igual la primera coordenada, por lo que en esos casos se pueden definir de cualquier forma.

Definimos también un perfil de estrategias puras N de modo tal que cada jugador (i, j) juega en estrategias puras en la máquina M_i. Es decir, $N(i, j) = i$. Observamos que esto es un equilibrio de Nash pues si un jugador (i, j) tiene incentivos a cambiar de máquina, entonces en la instancia original el jugador j no podía haber jugado con probabilidad positiva en la máquina i. Esto porque la carga esperada de cada máquina en N_{mix} cuando ya han jugado $J_1, J_2, \ldots, J_{j-1}$ es igual a la carga en esa máquina en N cuando ya han jugado todos los jugadores cuya segunda coordenada es $1, 2, \ldots, j-1$. Luego, como J_j juega con probabilidad mayor que 0 en la máquina M_i, esa máquina está entre las que tiene menor carga hasta ese momento, por lo que (i, j) puede jugar ahí.

Denotamos por $\widetilde{C}_{i,j}$ al costo del jugador (i, j) en el equilibrio de Nash N y denotamos por $\mathbb{E}(S_j)$ a la carga esperada que tienen las máquinas donde el jugador J_j juega en N_{mix} con probabilidad mayor que 0, antes de que J_j haya jugado. De esta definición es directo que $\mathbb{E}(S_j) + p_j \mathbb{E}(C_j)$, y también es claro que $\mathbb{E}(S_j) + p_{i,j} = \widetilde{C}_{i,j}$ para todo trabajo (i, j).

Así observamos que:

$$\begin{aligned}
C_N\left(\widetilde{I}\right) &= \mathbb{E}\left(\sum_{j=1}^{n}\sum_{i=1}^{m} p_{i,j}\widetilde{C}_{i,j}\right) \\
&= \sum_{j=1}^{n}\sum_{i=1}^{m} p_{i,j}(\mathbb{E}(S_j) + p_{i,j}) \\
&= \sum_{j=1}^{n}\left(\sum_{i=1}^{m} p_{i,j}(\mathbb{E}(S_j) + p_j) + \sum_{i=1}^{m} p_{i,j}(p_{i,j} - p_j)\right) \\
&= \sum_{j=1}^{n}\left(\mathbb{E}(C_j)\sum_{i=1}^{m} p_{i,j} + \sum_{i=1}^{m} p_{i,j}^2 - p_j\sum_{i=1}^{m} p_{i,j}\right) \\
&= \sum_{j=1}^{n}\left(\mathbb{E}(C_j)p_j + \sum_{i=1}^{m} p_{i,j}^2 - p_j^2\right) \\
&= C_{N_{mix}}(I) - \sum_{j=1}^{n}\left(p_j^2 - \sum_{i=1}^{m}(x_{i,j}p_j)^2\right).
\end{aligned}$$

Por otra parte:

$$\begin{aligned}
C_{opt}\left(\widetilde{I}\right) &\leq \frac{\|L_{OPT}(I)\|^2}{2} + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{m} p_{i,j}^2 \\
&= C_{opt}(I) - \frac{1}{2}\sum_{j=1}^{n}\left(p_j^2 - \sum_{i=1}^{m}(x_{j,i}p_j)^2\right)
\end{aligned}$$

Pero veamos que en un peor equilibrio de Nash mixto todas las máquinas tienen la misma carga.

En efecto, $\mathbb{E}(C_j) = \mathbb{E}(S_j) + p_j$, y además $S_j \leq \frac{1}{m}\sum_{k=1}^{j-1} p_k$ pues las máquinas con menor carga deben tener carga menor al promedio. Luego, como la cota se alcanza cuando todas las máquinas tienen la misma carga, un peor equilibrio de Nash mixto debe tener todas las máquinas con igual carga.

Así, como el costo de un equilibrio de Nash en estrategias puras depende únicamente del perfil de carga de las máquinas, se tiene que $C_N(\widetilde{I}) = C_{opt}(\widetilde{I})$.

Sigue que:

$$C_{N_{mix}}(I) - \sum_{j=1}^{n}\left(p_j^2 - \sum_{i=1}^{m}(x_{j,i}p_j)^2\right) \leq C_{opt}(I) - \frac{1}{2}\sum_{j=1}^{n}\left(p_j^2 - \sum_{i=1}^{m}(x_{j,i}p_j)^2\right)$$

Desarrollando esto:

$$\begin{aligned} C_{N_{mix}}(I) &\leq C_{opt}(I) + \frac{1}{2}\sum_{j=1}^{n}\left(p_j^2 - \sum_{i=1}^{m}(x_{j,i}p_j)^2\right) \\ &\leq C_{opt}(I) + \frac{1}{2}\sum_{j=1}^{n} p_j^2 \\ &\leq C_{opt}(I) + \frac{1}{2}C_{opt}(I) \end{aligned}$$

De donde se concluye que:

$$\frac{C_{N_{mix}}(I)}{C_{opt}(I)} \leq \frac{3}{2}$$

□

A continuación se muestra una cota inferior para el juego $Q||\sum w_j C_j$, la cual establece que el precio de la anarquía aumenta cuando se pueden imponer velocidades en las máquinas, incluso si solo consideramos estrategias puras en el juego con máquinas con velocidades, y estrategias mixtas en el juego con máquinas idénticas.

3.2. Cota inferior para $Q||\sum w_j C_j$

Teorema 3.2. *El precio de la anarquía en estrategias puras para el juego $Q||\sum w_j C_j$ es mayor o igual que 2.*

Demostración. Consideremos una instancia del juego I con n máquinas y n trabajos. En esta instancia, tendremos que la máquina M_1 tendrá velocidad $s_1 = s$, y que todas las otras máquinas tendrán velocidad $s_i = 1$. Para todos los trabajos J_j se tendrá que $w_j = p_j$. Se denotará $x = \frac{s}{s-1}$. Los tiempos de proceso serán:

$$p_j = \begin{cases} 1, & j \leq s, \\ \left(\frac{s}{s-1}\right)^{j-s}, & s+1 \leq j \leq n. \end{cases}$$

Como $w_j = p_j$, cualquier lista de prioridades es *WSPT*, por lo que consideramos que todas las máquinas tienen la lista: $1 \prec 2 \prec \ldots \prec n$.

En el ejemplo dado en el Teorema 2.3 se mostró que, con el mismo conjunto de trabajos y máquinas y donde las máquinas tenían las mismas prioridades, el perfil de estrategias donde todos los trabajos juegan en M_1 es un equilibrio de Nash. Llamamos N a este equilibrio

El costo óptimo de la instancia será menor o igual al costo de cualquier perfil de estrategias de la instancia. Por esto, sea k un entero y consideremos el perfil de estrategias S^k donde los trabajos J_j tales que $1 \leq j \leq n-k$ están cada uno en una máquina de velocidad 1, y los restantes trabajos están en la máquina M_1 (de velocidad s), de modo que se procesa primero el trabajo J_{n-k+1}, y luego se procesan en orden creciente para terminar con el trabajo J_n. Así, el costo para los trabajos J_j tales que $1 \leq j \leq n-k$ será ${C_j}^{S^k} = p_j$, y el costo para los trabajos J_j tales que $j \geq n-k+1$ será:

$$\begin{aligned} {C_j}^{S^k} &= \sum_{i=n+k+1}^{j} \frac{p_i}{s} \\ &= \frac{1}{s} \sum_{i=n+k+1}^{j} \left(\frac{s}{s-1}\right)^{i-s} \\ &= \left(\frac{s}{s-1}\right)^{j-s} - \left(\frac{s}{s-1}\right)^{n-k-s} \end{aligned}$$

Por esto tenemos, tomando $k = \alpha s$:

$$\begin{aligned} \textbf{PPoA} &\geq \frac{\sum_{j=1}^{n} p_j {C_j}^{N}}{\sum_{j=1}^{n} p_j {C_j}^{S^k}} \\ &= \frac{\sum_{j=1}^{s} \frac{j}{s} + \sum_{j=1}^{s} \left(\frac{s}{s-1}\right)^{j-s} \left(\frac{s}{s-1}\right)^{j-s}}{\sum_{j=1}^{s} 1 + \sum_{j=s+1}^{n-k} \left(\frac{s}{s-1}\right)^{j-s} \left(\frac{s}{s-1}\right)^{j-s} + \sum_{j=n-k+1}^{n} \left(\frac{s}{s-1}\right)^{j-s} \left(\left(\frac{s}{s-1}\right)^{j-s} - \left(\frac{s}{s-1}\right)^{n-k-s}\right)} \end{aligned}$$

$$
\begin{aligned}
&= \frac{\frac{s+1}{2} + \frac{\left(\frac{s}{s-1}\right)^{2n-2s+2} - \left(\frac{s}{s-1}\right)^2}{\left(\frac{s}{s-1}\right)^2 - 1}}{s + \frac{\left(\frac{s}{s-1}\right)^{2n-2s+2} - \left(\frac{s}{s-1}\right)^2}{\left(\frac{s}{s-1}\right)^2 - 1} - \left(\frac{s}{s-1}\right)^{n-k-s} \sum_{j=n-k+1}^{n} \left(\frac{s}{s-1}\right)^{j-s}} \\
&= \frac{\frac{s+1}{2} + \frac{\left(\frac{s}{s-1}\right)^{2n-2s+2} - \left(\frac{s}{s-1}\right)^2}{\left(\frac{s}{s-1}\right)^2 - 1}}{s + \frac{\left(\frac{s}{s-1}\right)^{2n-2s+2} - \left(\frac{s}{s-1}\right)^2}{\left(\frac{s}{s-1}\right)^2 - 1} - \frac{\left(\frac{s}{s-1}\right)^{2n-2s-k+1} - \left(\frac{s}{s-1}\right)^{2n-2s-2k+1}}{\left(\frac{s}{s-1}\right) - 1}} \\
&\xrightarrow{n\to\infty} \frac{\frac{\left(\frac{s}{s-1}\right)^{-2s+2}}{\left(\frac{s}{s-1}\right)^2 - 1}}{\frac{\left(\frac{s}{s-1}\right)^{-2s+2}}{\left(\frac{s}{s-1}\right)^2 - 1} - \frac{\left(\frac{s}{s-1}\right)^{-2s-k+1} - \left(\frac{s}{s-1}\right)^{-2s-2k+1}}{\left(\frac{s}{s-1}\right) - 1}} \\
&= \frac{\left(\frac{s}{s-1}\right)^{-2s+2}}{\left(\frac{s}{s-1}\right)^{-2s+2} - \left(\left(\frac{s}{s-1}\right) + 1\right)\left(\left(\frac{s}{s-1}\right)^{-2s-k+1} - \left(\frac{s}{s-1}\right)^{-2s-2k+1}\right)} \\
&= \frac{\left(\frac{s}{s-1}\right)^{2k+2}}{\left(\frac{s}{s-1}\right)^{2k+2} - \left(\left(\frac{s}{s-1}\right) + 1\right)\left(\left(\frac{s}{s-1}\right)^{k+1} - \left(\frac{s}{s-1}\right)\right)} \\
&\xrightarrow{s\to\infty} \frac{e^{2\alpha}}{e^{2\alpha} - 2(e^{2\alpha} - 1)}
\end{aligned}
$$

El valor máximo de este último término es 2, el cual se obtiene para $\alpha = \ln 2$. Por esto, tomando $\alpha = \frac{k}{s}$ convergiendo a $\ln 2$ se obtiene que **PPoA** $(Q||\sum w_j C_j) \geq 2$.

□

3.3. Juegos con todos los trabajos iguales

El siguiente es un ejemplo de uno de los casos más generales en el que el algoritmo de Smith es óptimo.

Teorema 3.3. *El precio de la anarquía en estrategias puras para el juego* $Q|p_j = 1|\sum w_j C_j$ *es 1.*

Demostración. Usando el Lema 2.5, nos podemos restringir a instancias donde $w_j = 1$, de donde surge una demostración simple. Se mostrará acá una demostración alternativa usando la estructura del problema.

Sea $I = (\mathcal{J}, \mathcal{M})$ una instancia del juego, con m máquinas y n trabajos. Sin pérdida de generalidad se asumirá que los trabajos cumplen $w_1 \geq w_2 \geq w_3 \geq \ldots \geq w_n$. Luego, la lista de prioridades será tal que: $1 \prec 2 \prec \ldots n$. Dado un *schedule* S, definimos L_i^S como la

cantidad de trabajos que juegan en la máquina M_i en el perfil de estrategias S. Definimos también el perfil de máquinas de un *schedule* S como el vector

$$L^S = \left(L_1^S, L_2^S, \ldots, L_m^S\right).$$

Sea $\mathcal{O}$ el conjunto de *schedules* óptimos de la instancia I. Se probará primero que para todo equilibrio de Nash N, existe un *schedule* $O \in \mathcal{O}$, de modo que $L^N = L^O$, es decir todo equilibrio de Nash tiene el mismo perfil de máquinas que algún óptimo. En efecto, sea N un equilibrio de Nash de la instancia. Para todo *schedule* S, definimos el valor $\Delta_N(S)$ como:

$$\Delta_N(S) := \sum_{i=1}^{m} \left|L_i^N - L_i^S\right|.$$

Notar que $L^N = L^S \iff \Delta_N(S) = 0$. Sea $O \in \mathcal{O}$ tal que:

$$\Delta_N(O) = \min_{S \in \mathcal{O}} \Delta_N(S)$$

Es decir, O es un *schedule* que minimiza el valor de Δ_N dentro de los *schedules* óptimos.

Si $\Delta_N(O) = 0$, entonces N tiene el mismo perfil de máquinas que O, que es lo que se desea probar. Supongamos que $\Delta_N(O) > 0$. Luego existirán dos máquinas i^+ e i^-, tales que $L_{i^+}^N > L_{i^+}^O$ y $L_{i^-}^N < L_{i^-}^O$. Sea J^- el último trabajo procesado en la máquina i^- en O. Consideremos el *schedule* O', que es idéntico a O, exceptuando que J^- es procesado al último en la máquina i^+.

Como el último trabajo procesado en la máquina i^+ en el *schedule* N eligió la máquina i^+ en vez de la i^-, el trabajo J^- redujo su costo al hacer esto (pues el costo de un trabajo es el número de trabajos que hay hasta el en la máquina que eligió), y como todos los otros trabajos mantuvieron su costo, se tendrá que $C'_O(I) \leq C_O(I)$. Por esto O' es un *scheduling* óptimo, pero además $\Delta_N(O') < \Delta_N(O)$ lo cual es una contradicción con la definición de O. Por lo tanto $\Delta_N(O) = 0$.

Veremos ahora que todo equilibrio de Nash es un *schedule* óptimo. Sea N un equilibrio de Nash. Notemos que si existe un *schedule* óptimo O que cumple que para todo trabajo J_j se tiene $N(j) = O(j)$, entonces N es un *scheduling* óptimo, pues en una sola máquina ordenar por *WSPT* es óptimo. Supongamos entonces que N no es óptimo. Dado un *schedule* $S \neq N$, definimos $f_N(S)$ como:

$$f_N(S) := \min\left\{j \in \{1, 2, \ldots n\} \,\middle|\, S(j) \neq N(j)\right\}.$$

Es decir, el trabajo $J_{f_N(S)}$ es el primero que juega en S en una máquina distinta a la que juega en N.

Sea $\mathcal{O}_1 = \{S \in \mathcal{O} \big| \Delta_N(O) = 0\}$. Sea $O \in \mathcal{O}_1$ tal que:

$$f_N(O) = \max_{S \in \mathcal{O}_1} f_N(S).$$

Es decir, O es un *schedule* óptimo con el mismo perfil de máquinas que N que maximiza el índice del primer trabajo que va en una máquina distinta en N.

Sean

$$\begin{aligned} d &:= {C_{f_N O}}^N = \left|\{j \leq f_N(O) \big| N(j) = N(f_N(O))\}\right|, \\ e &:= {C_{f_N(O)}}^O = \left|\{j \leq f_N(O) \big| O(j) = O(f_N(O))\}\right|. \end{aligned}$$

Como N y O tienen el mismo perfil de máquinas, entonces el *schedule* O asigna al menos d trabajos a la máquina $N(f_N(O))$.

Sea k el trabajo que ocupa la posición d en la máquina $N(f_N(O))$ en el *schedule* O. Se tendrá que $k > f_N(O)$ pues todos los trabajos entre J_1 y $J_{f_N(O)-1}$ fueron procesados en otras máquinas o fueron procesados antes que $J_{f_N(O)}$ en esa misma máquina.

Sea O' el *schedule* que resulta de intercambiar en O los trabajos $J_{f_N(O)}$ y J_k. Observamos que O' tiene el mismo perfil de máquinas que O pues solo se hizo un intercambio entre dos trabajos, luego también tiene el mismo perfil de máquinas que N. Además $f_N(O') > f_N(O)$, pues ahora también el trabajo $J_{f_N(O)}$ juega en la misma máquina en N y en O'.

Pero observemos que:

$$\begin{aligned} C_{O'}(I) - C_O(I) &= w_{f_N(O)} {C_{f_N(O)}}^N + w_k {C_{f_N(O)}}^O - w_{f_N(O)} {C_{f_N(O)}}^O - w_k {C_{f_N(O)}}^N \\ &= (w_{f_N(O)} - w_k)({C_{f_N(O)}}^N - {C_{f_N(O)}}^O) \\ &\leq 0 \end{aligned}$$

Esta última desigualdad se debe a que $f_N(O) < k$. Luego, $w_{f_N(O)} \geq w_k$. Además ${C_{f_N(O)}}^N \leq {C_{f_N(O)}}^O$ pues hasta antes del trabajo $f_N(O)$ todas las máquinas están iguales tanto en N como en O, y como N es equilibrio de Nash, $J_{f_N(O)}$ jugará en la máquina que minimice su costo.

Pero $f_N(O') > f_N(O)$, y además $C_{O'}(I) \leq C_O(I)$, por lo que O' también es un *schedule* óptimo. Esto es una contradicción con la definición de O. Por lo tanto, N es un *schedule* óptimo. □

Capítulo 4

Costo promedio como función social con un número infinito máquinas

El precio de la anarquía del problema de *scheduling* $Q||\sum C_j$ está acotado entre $\frac{e}{e-1}$ y 2 [5]. Las instancias donde se consigue la cota inferior se logran en un ambiente particular de máquinas, en donde todas las máquinas son idénticas, exceptuando una que es más rápida que el resto. Se conjetura acá que en ese contexto se da el peor precio de la anarquía, por lo cual es interesante estudiar el juego en esas instancias en particular.

Se denotará por $Q_{s,1,\ldots,1}||\sum C_j$ al problema de *scheduling* de máquinas paralelas con velocidades en el cual todas las máquinas son idénticas exceptuando por una que es más rápida. Es decir, si $I = (\mathcal{J}, \mathcal{M})$ es una instancia del problema de *scheduling* $Q_{s,1,\ldots,1}||\sum C_j$, asumiremos sin pérdida de generalidad que $s_1 \geq 1$ y $s_2 = s_3 = \ldots = s_m = 1$, donde s_i denota la velocidad de la máquina M_i.

En una instancia $I = (\mathcal{J}, \mathcal{M})$ del problema $Q_{s,1,\ldots,1}||\sum C_j$, denotaremos por $n(I) = |\mathcal{J}|$, $m(I) = |\mathcal{M}|$ y $s(I) = s_1$. Cuando no haya confusión se usará simplemente n, m y s.

En las instancias que dan la cota inferior del precio de la anarquía del juego asociado al problema $Q||\sum C_j$ el número de máquinas considerado es suficientemente grande para que en el óptimo cada máquina de velocidad 1 deba procesar a lo más un trabajo. Por eso, estas instancias bastan para obtener una cota inferior de $\frac{e}{e-1}$, por lo que se estudiará el precio de la anarquía en este caso particular.

Se denotará por $Q_{s,1,\ldots\infty\ldots,1}||\sum C_j$ al problema de *scheduling* en el que todas las instancias $I = (\mathcal{J}, \mathcal{M})$ son instancias del problema $Q_{s,1,\ldots,1}||\sum C_j$ y donde además se tendrá que $m + s > n$.

En una instancia del problema de *scheduling* $Q_{s,1,\ldots,1}||\sum C_j$, en cualquier equilibrio de Nash; los primeros $\lfloor s \rfloor$ trabajos estarán en la máquina M_1. Así también, el algoritmo *MFT*

asigna los últimos $\lfloor s \rfloor$ trabajos a la máquina M_1. Dado que los únicos *schedules* que nos interesa estudiar son los equilibrios de Nash y los *schedules* óptimos, es natural imaginar una instancia de $Q_{s,1,\ldots\infty\ldots,1}|| \sum C_j$ como si tuviera un número infinito de máquinas de velocidad 1, lo cual también justifica la notación usada.

En el presente capítulo se estudia el precio de la anarquía del problema $Q_{s,1,\ldots\infty\ldots,1}|| \sum C_j$, y se demuestra que su valor exacto es de $\frac{e}{e-1}$. Este es el valor de la cota inferior ya conocida, sin embargo acá se muestra de dos formas distintas que también es una cota superior.

4.1. Demostración combinatorial de la cota con un número infinito de máquinas

Se demostrará primero que el valor del precio de la anarquía es $\frac{e}{e-1}$ usando las propiedades combinatoriales del problema. La estructura de demostración puede resumirse de la siguiente forma:

- Se probará primero que las instancias que alcanzan el peor precio de la anarquía, son aquellas en que en el peor de los equilibrios de Nash de esa instancia, los primeros $n-s$ trabajos son asignados a la máquina más rápida.
- Se probará luego, para las peores instancias, el tamaño de esos primeros $n-s$ trabajos siguen la misma estructura que el ejemplo de peor caso conocido del precio de la anarquía para el problema $Q|| \sum C_j$, dado por Hoeksma y Uetz, y mostrado acá en la Sección 2.3. Esto es, se probará que en esas instancias, salvo escalamiento, se tiene que $1 = p_1 = p_2 = \ldots = p_s$, y que para $j \geq s$, $p_j = \left(\frac{s}{s-1}\right)^{j-s}$.
- Se probará a continuación que en las peores instancias, dentro de los últimos s trabajos, aquellos que no estén en la máquina rápida en el peor equilibrio de Nash, deberán tener un tamaño igual a uno de los trabajos que sí están en la máquina rápida. Esto permitirá probar a continuación que en el peor equilibrio de Nash de las peores instancias, todos los trabajos están en la máquina rápida.
- Se concluirá que el precio de la anarquía se obtendrá para las instancias que tienen la misma estructura que el ejemplo de Hoeksma y Uetz, para las cuales se sabe que el precio de la anarquía está acotado superiormente por $\frac{e}{e-1}$.

Teorema 4.1. *Se tiene que*

$$\textbf{PPoA}\left(Q_{s,1,\ldots\infty\ldots,1}|| \sum C_j\right) = \frac{e}{e-1}.$$

Para probar este teorema, se introducirán algunas definiciones y se establecerán lemas previos.

Definición 4.2. Dada una instancia I del juego de *scheduling* $Q||\sum C_j$, se denotará por $\mathcal{W}(I)$ al conjunto de equilibrios de Nash de I con mayor costo. Cuando no haya confusión respecto a la instancia con que se esté trabajando se denotará simplemente por $\mathcal{W}$.

Definición 4.3. Se denotará por $\mathcal{I}$ al conjunto de instancias del problema $Q_{s,1,\ldots\infty\ldots,1}\big|\big|\sum C_j$, y para $s \geq 1$, se define $\mathcal{I}^s := \big\{I \in \mathcal{I}\big|s(I) = s\big\}$. Es decir, $\mathcal{I}^s$ es el conjunto de instancias de $\mathcal{I}$, tales que la máquina rápida tiene velocidad s.

Lema 4.4. *Definimos*

$$\mathcal{I}_1^s := \big\{I \in \mathcal{I}^s\big|\exists W \in \mathcal{W}(I),\ W(1) = W(2) = \ldots = W(n-s) = M_1\big\}.$$

Entonces, $\mathbf{PPoA}(\mathcal{I}_1^s) = \mathbf{PPoA}(\mathcal{I}^s)$.

Demostración. Probaremos que para cada instancia $I \in \mathcal{I}^s$ existe una instancia $I' \in \mathcal{I}_1^s$ tal que $\mathbf{PPoA}(I') \geq \mathbf{PPoA}(I)$. Por contradicción, supongamos que para toda instancia $I' \in \mathcal{I}^s$ tal que $\mathbf{PPoA}(I') \geq \mathbf{PPoA}(I)$, cada uno de los peores equilibrios de Nash tiene uno de los $n(I') - s(I') = n(I') - s$ trabajos más chicos asignados en una máquina de velocidad 1.

Llamemos $I = \big((p_1, p_2, \ldots, p_n), \mathcal{M}\big)$. Consideremos W un peor equilibrio de Nash para I. Por nuestra suposición, sabemos que existe un conjunto de trabajos J, con $J \subseteq \{J_1, J_2, \ldots, J_{n-s}\}$, asignados a una de las máquinas de velocidad 1. Consideremos I' la instancia en la cual removimos los trabajos de J, y mantenemos las mismas máquinas. Es claro que si asignamos cada trabajo en I' a la misma máquina en la que está asignado en W, es un peor equilibrio de Nash. Entonces:

$$C_W(I') = C_W(I) - \sum_{j \in J} p_j.$$

Por el algoritmo *MFT*, en la asignación óptima de I y I', cada uno de los s trabajos más grandes esta en la máquina de velocidad s, y cada uno de los otros trabajos está solo en una máquina de velocidad 1. Luego:

$$C_{opt}(I') = C_{opt}(I) - \sum_{j \in J} p_j.$$

Usando esto se tiene que:

$$\mathbf{PPoA}(I') = \frac{C_W(I) - \sum_{j\in J} p_j}{C_{opt}(I) - \sum_{j\in J} p_j} \geq \frac{C_W(I)}{C_{opt}(I)} = \mathbf{PPoA}(I).$$

Luego tenemos una contradicción, pues I' admite un peor equilibrio de Nash que tiene cada trabajo asignado a la máquina de velocidad s. □

Lema 4.5. *Definimos*

$$\mathcal{I}_2^s := \left\{ I \in \mathcal{I}_1^s \middle| \textit{se cumple } (4.1) \right\}.$$

Donde

$$\left.\begin{aligned} p_j &= p_1 & \forall j &\in \{1,\dots,s\} \\ p_{s+j} &= p_1 \left(\frac{s}{s-1}\right)^j & \forall j &\in \{1,\dots,n-2s\} \end{aligned}\right\} \tag{4.1}$$

Entonces, $\mathbf{PPoA}(\mathcal{I}_2^s) = \mathbf{PPoA}(\mathcal{I}_1^s)$

Demostración. Por contradicción, asumamos que $\mathbf{PPoA}(\mathcal{I}_1^s) > \mathbf{PPoA}(\mathcal{I}_2^s)$, luego existe $I = (\mathcal{J}, \mathcal{M}) \in \mathcal{I}_1^s$ tal que $\mathbf{PPoA}(I) > \mathbf{PPoA}(\mathcal{I}_2^s)$. Como $I \in \mathcal{I}_1^s$, existe $W \in \mathcal{W}(I)$, tal que los primeros $n - s$ trabajos son asignados a la máquina M_1 en W. Como W es un equilibrio de Nash, se debe tener que:

$$\left.\begin{aligned} p_j &\geq p_1 & \forall j &\in \{1,\dots,s\} \\ p_{s+j} &\geq p_1 \left(\frac{s}{s-1}\right)^j & \forall j &\in \{1,\dots,n-2s\} \end{aligned}\right\} \tag{4.2}$$

Sea $I' \in \mathcal{I}_1^s$ una instancia tal que $\mathbf{PPoA}(I') \geq \mathbf{PPoA}(I)$, y tal que minimiza el número de desigualdades estrictas de (4.2), y en caso de empates, tomamos una que minimiza el primer índice de una desigualdad estricta. Luego, tomamos la primera desigualdad estricta, digamos la $j^{\text{ésima}}$ desigualdad, y definimos $I'' = (\mathcal{J}'', \mathcal{M}')$, con

$$\mathcal{J}'' = (p_1'(1+\varepsilon), p_2'(1+\varepsilon), \dots, p_j'(1+\varepsilon), p_{j+1}' - \varepsilon(p_1' + p_2' + \cdots + p_j'), p_{j+2}', \dots, p_n')$$

Claramente $\mathbf{PPoA}(I'') > \mathbf{PPoA}(I')$, y el número de desigualdades estrictas es el mismo, pero ahora el primer índice de una desigualdas estricta es mayor, lo que es una contradicción. □

Lema 4.6. *Definamos*

$$\mathcal{I}_3^s := \left\{ I \in \mathcal{I}_2^s \middle| p_{n(I)} > p_{n(I)-1} \right\}.$$

Entonces, $\mathbf{PPoA}(\mathcal{I}_3^s) = \mathbf{PPoA}(\mathcal{I}_2^s)$

Demostración. Dada una instancia $I = (\mathcal{J}, \mathcal{M})$, definamos $\eta(I) = \left|\left\{j \in \mathcal{J} \middle| p_j = p_{n(I)}\right\}\right|$.

Dada $I \in \mathcal{I}_2^s$, definamos:

$$\upsilon = \min\left\{\eta(I') \middle| I' \in \mathcal{I}_2^s, \mathbf{PPoA}(I') \geq \mathbf{PPoA}(I)\right\},$$

y sea $I' \in \mathcal{I}_2^s$ tal que $\eta(I') = \upsilon$ y $\mathbf{PPoA}(I') \geq \mathbf{PPoA}(I)$.

Si $\upsilon = 1$, entonces $I' \in \mathcal{I}_3^s$, y estaríamos listos. Supongamos que $\upsilon > 1$. Luego existen al menos 2 trabajos con tiempos de proceso p_n, lo que lleva a una contradicción con la definición de $\mathcal{I}_3^s$. □

Lema 4.7. *Definamos $\mathcal{I}_4^s$ como el conjunto de instancias $I \in \mathcal{I}_3^s$ tales que existe un peor equilibrio de Nash para I en el cual cada trabajo que no está en la máquina de velocidad s, tiene el mismo tiempo de proceso de uno de los trabajos que está en la máquina de velocidad s, y tal que cada uno de los últimos trabajos en la máquina de velocidad s (los que están después de los primeros s trabajos) es indiferente entre esa máquina y una de velocidad 1. entonces,* $\mathbf{PPoA}(\mathcal{I}_4^s) = \mathbf{PPoA}(\mathcal{I}_3^s)$

Demostración. Fijamos $n \in \mathbb{N}$. Sea I_0 una instancia que maximiza **PPoA** entre todas las instancias de $\mathcal{I}_3^s$ que tienen a lo más n trabajos. Sea $W_0 \in \mathcal{W}(I)$. Sean $J_{j_1}, J_{j_2}, \ldots, J_{j_l}$ los trabajos que no están entre los primeros $n - s$ que son asignados a la máquina de velocidad s en W_0. Notemos que el trabajo J_n es indiferente entre la máquina de velocidad s y una máquina de velocidad 1, pues si el trabajo J_n prefiriera estrictamente una máquina de velocidad 1, entonces podemos tomar una instancia I' que solo cambia p_n por $p_n + \varepsilon$, y para ε suficientemente pequeño se tiene

$$\mathbf{PPoA}(I') = \frac{C_n(I_0) + \varepsilon}{C_{opt}(I) + \frac{\varepsilon}{s}} > \mathbf{PPoA}(\mathcal{I}_0)$$

Si J_n prefiriera estrictamente la máquina de velocidad s, entonces podemos cambiar p_n por $p_n - \varepsilon$, y para ε suficientemente pequeño se tendrá que

$$\mathbf{PPoA}(I') = \frac{C_n(I_0) - \varepsilon}{C_{opt}(I) - \varepsilon} > \mathbf{PPoA}(I_0)$$

Luego, en el peor equilibrio de Nash, el trabajo más grande es indiferente entre la máquina de velocidad s y una máquina vacía de velocidad 1, por lo que asumimos que $J_{j_l} = J_n$, es decir que el trabajo J_n está en la máquina de velocidad s.

Ahora, definimos el bloque B_k como el conjunto de trabajos J_t que son asignados a una máquina de velocidad 1, y tales que $p_{j_{k-1}} \leq p_t < p_{j_k}$. Aquí llamamos $J_{j_0} = J_{n-s}$. Es claro que cada trabajo que está en una máquina de velocidad 1 está en exactamente uno de los bloques.

Ahora probaremos que el último bloque B_l tiene a lo más 1 trabajo con tiempo de proceso en el intervalo $]p_{j_{l-1}}, p_{j_l}[$. Si hubiera más de un trabajo ahí, tomemos la instancia I', que

cambia el trabajo más pequeño dentro J_{t_1} y lo hace ε más pequeño, y cambia el trabajo más grande ahí dentro J_{t_2}, y lo hace ε más grande. Para ε suficientemente pequeño, esto será un equilibrio de Nash todavía, basta ver que con esto ambos trabajos tienen aun su tiempo de proceso en $[p_{j_{l-1}}, p_{j_l}]$. Es claro que $C_N(I') = C_N(I)$, y que el tiempo de completación en el *scheduling* óptimo de los trabajos entre J_{t_1} and J_{t_2-1} es ahora más pequeño en una cantidad $\frac{\varepsilon}{s}$, y todos los otros tiempos de completación se mantienen iguales. Luego, tenemos que $\mathbf{PPoA}(I') > \mathbf{PPoA}(I)$, lo que contradice la maximalidad de I.

Ahora probaremos que cada trabajo J_{j_k} es indiferente entre la máquina de velocidad s y una máquina de velocidad 1, y que cada bloque B_k tiene a lo más 1 trabajo con tiempo de proceso en $]p_{j_{k-1}}, p_{j_k}[$.

Probaremos por inducción sobre p, que para cada $p = 0, 1, 2, \ldots, l-1$, el trabajo $J_{j_{l-p}}$ es indiferente entre la máquina de velocidad s y una máquina de velocidad 1, y que cada bloque B_{l-p} tiene a lo más 1 trabajo en $]p_{j_{l-p-1}}, p_{j_{l-p}}[$. Esto ya fue probado para $p = 0$. Asumamos que se cumple para cada número $0, 1, \ldots, p-1$, y probemos que se cumple para p.

Probaremos primero que el trabajo $J_{j_{l-p}}$ is indiferente entre la máquina de velocidad s y una máquina de velocidad 1. Supongamos que esto no es verdad, entonces prefiere estrictamente la máquina de velocidad s. Tomemos una instancia I' que cambia $p_{j_{l-p}}$ por $p_{j_{l-p}} - \varepsilon\epsilon$. Debemos ver entonces que para ε suficientemente pequeño esto es un equilibrio de Nash. Cada trabajo que tiene un tiempo de proceso menor a $p_{j_{l-p}}$ es indiferente con este cambio. cada trabajo que está en la máquina de velocidad s y tiene tiempo de proceso mayor a $p_{j_{l-p}}$ aun quiere estar en la máquina de velocidad s, pues ahora tiene menor tiempo de completación en esa máquina. Sea J_d un trabajo con tiempo de proceso mayor o igual que $p_{j_{l-p}}$, en una máquina de velocidad 1. Sea B_k el bloque al cual J_d pertenece. Entonces, si ahora prefiriera la máquina de velocidad s, iría justo después del trabajo $J_{j_{k-1}}$. Pero por hipótesis de inducción, el trabajo J_{j_k} es indiferente entre cualquier máquina, por lo que su tiempo de completación es p_{j_k}, luego si el trabajo J_d fuera en la máquina de velocidad s, su tiempo de completación sería $C'_d = p_{j_k} - \frac{p_{j_k}}{s} + \frac{p_d}{s} - \varepsilon$. Como $p_d < p_{j_k}$, podemos tomar $\varepsilon < (p_{j_k} - p_d)\left(1 - \frac{1}{s}\right)$ que es positivo, y luego $C'_d > p_d$. Entonces, para ε suficientemente pequeño tal que eso se tiene para cada trabajo en una máquina de velocidad 1, se tendrá que es un equilibrio de Nash. Luego, $C_N(I') = C_N(I) - \varepsilon\frac{p+1}{s}$, pues $p+1$ cambiaron su tiempo de completación en $\frac{\varepsilon}{s}$. Y $C_{opt}(I') = C_{opt}(I) - \varepsilon\frac{n-j_{l-p}+1}{s}$ pues para cada $k = 1, 2, \ldots, l,\ j_k \leq n-k+l$se tiene $\frac{p+1}{s} \leq \frac{n-j_{l-p}+1}{s}$, y por lo tanto $\mathbf{PPoA}(I') \geq \mathbf{PPoA}(I)$, que contradice la maximalidad de I.

Ahora probaremos que el bloque B_{l-p} tiene a lo más un trabajo con tiempo de proceso en $]p_{j_{l-p-1}}, p_{j_{l-p}}[$. Si hubiera más de un trabajo ahí, tomamos la instancia I' que cambia el trabajo más pequeño ahí dentro J_{t_1} y lo hace ε más pequeño, y cambia el trabajo más grande ahí dentro J_{t_2}, y lo hace ε más grande. Para ε suficientemente pequeño esto aun es un equilibrio de Nash, para ε suficientemente pequeño tal que $p_{t_2} + \varepsilon < p_{j_{l-p}}$, pues el trabajo

$J_{j_{l-p}}$ es indiferente entre cada máquina, por lo que el tiempo de completación del trabajo $J_{j_{l-p}}$ es $p_{j_{l-p}}$, y luego, el tiempo de completación del trabajo J_{t_2} si lo ponemos en la máquina de velocidad s sería:

$$C'_{t_2} = p_{j_{l-p}} - \frac{p_{j_{l-p}}}{s} + \frac{p_{t_2}+\varepsilon}{s} = p_{t_2} + \varepsilon + (p_{j_{l-p}} - p_{t_2} - \varepsilon)\left(1 - \frac{1}{s}\right) > p_{t_2} + \varepsilon.$$

Es claro que $C_N(I') = C_N(I)$, y que el tiempo de completación de los trabajos entre J_{t_1} and J_{t_2-1} es más pequeño por $\frac{\varepsilon}{s}$, y cada uno de los otros tiempos de completación se mantiene igual. Luego tenemos que $\mathbf{PPoA}(I') > \mathbf{PPoA}(I)$, lo que contradice la maximalidad de I. Concluyendo así la demostración por inducción.

Basta probar que no hay trabajos dentro de un bloque B_k que tengan tiempo de completación en $]p_{j_{k-1}}, p_{j_k}[$. Supongamos que hay un trabajo J_t con $p_{j_{k-1}} < p_t < p_{j_k}$. Para ε suficientemente pequeño, podemos aumentar o disminuir p_t en ε y eso será aun un equilibrio de Nash. Y para ε suficientemente pequeño, la estructura del *scheduling* óptimo no cambiara, implicando que cada trabajo se mantiene en su misma máquina, y la máquina de velocidad s tendrá los trabajos en el mismo orden.

Pero luego, el precio de la anarquía de la instancia será:

$$\begin{aligned}
\mathbf{PPoA}(I') &= \frac{\sum_{j=1}^{n} C'_j}{\sum_{j=1}^{s} p'_j + \sum_{j=s+1}^{n} \frac{n+1-j}{s} p'_j} \\
&= \frac{\sum_{\substack{j=1 \\ j\neq t}}^{n} C'_j + p'_t}{\sum_{j=1}^{s} p_j + \sum_{\substack{j=s+1 \\ j\neq t}}^{n} \frac{n+1-j}{s} p_j + p'_t \frac{n+1-t}{s}} \\
&= \frac{\sum_{\substack{j=1 \\ j\neq t}}^{n} C'_j - \frac{\sum_{j=1}^{s} p_j + \sum_{\substack{j=s+1 \\ j\neq t}}^{n} \frac{n+1-j}{s} p_j}{\frac{n+1-t}{s}}}{\sum_{j=1}^{s} p_j + \sum_{\substack{j=s+1 \\ j\neq t}}^{n} \frac{n+1-j}{s} p_j + p'_t \frac{n+1-t}{s}} + \frac{1}{\frac{n+1-t}{s}}
\end{aligned}$$

Como esto es una función monótona de p'_t (o es constante), y esto se tiene para cada $p'_t \in]p_t - \varepsilon, p_t + \varepsilon[$, entonces cambiamos la instancia I' para aumentar el precio de la anarquía. En el caso en que el precio de la anarquía es constante como función de p'_t, simplemente

tomamos $p'_t = p_{j_{k-1}}$

□

Observación 4.1. Como cada trabajo $J_{j_1}, J_{j_2}, \ldots, J_{j_l}$ es indiferente entre cada máquina, tenemos que $J_{j_k} = p_1 \left(\frac{s}{s-1}\right)^{n-2s+k}$.

Lema 4.8. *En la peor instancia I, ningún trabajo está en una máquina de velocidad 1.*

Demostración. Sea I la peor instancia, cuya estructura está dada por el Lema 4.7. Si un trabajo está en una máquina de velocidad 1, entonces existe k tal que B_k tiene al menos un trabajo.

Si aumentamos un trabajo en B_k en ε, tendremos que

$$\mathbf{PPoA}(I') = \frac{C_N(I) + \varepsilon}{C_{opt}(I) + \varepsilon\frac{n+1-(j_k-1)}{s}}$$

Si disminuimos un trabajo en B_k en ε, tendremos que

$$\mathbf{PPoA}(I') = \frac{C_N(I) - \varepsilon}{C_{opt}(I) - \varepsilon\frac{n+1-j_{k-1}}{s}}$$

Como I es la peor instancia, $\mathbf{PPoA}(I') \leq \mathbf{PPoA}(I)$ para ambos casos. Luego,

$$\frac{C_N(I) + \varepsilon}{C_{opt}(I) + \varepsilon\left(\frac{n+1-(j_k-1)}{s}\right)} \leq \frac{C_N(I)}{C_{opt}(I)},$$

lo cual es equivalente a que

$$C_{opt}(I) \leq C_N(I)\left(\frac{n+2-j_k}{s}\right),$$

de donde se concluye que

$$j_k \leq n + 2 - \frac{sC_{opt}(I)}{C_N(I)}.$$

Analogamente, se tiene que

$$\frac{C_N(I) - \varepsilon}{C_{opt}(I) - \varepsilon\left(\frac{n+1-j_{k-1}}{s}\right)} \leq \frac{C_N(I)}{C_{opt}(I)}$$

lo cual implica que

$$C_{opt}(I) \geq C_N(I)\left(\frac{n+1-j_{k-1}}{s}\right)$$

de donde se deduce que

$$j_{k-1} + 1 \geq n + 2 - \frac{sC_{opt}(I)}{C_N(I)}$$

Por lo tanto, deberíamos tener que $j_{k-1} + 1 \geq j_k$. Esto implica que B_k tiene 0 trabajos, y luego cada trabajo está en la máquina de velocidad s, lo que concluye la demostración. □

4.2. Problema Lineal que acota el precio de la anarquía con un número infinito de máquinas

En esta sección se muestra un conjunto de problemas de programación lineal que acotan superiormente el precio de la anarquía. Así, basta encontrar una cota para el óptimo de estos problemas de programación lineal para obtener una cota en el precio de la anarquía de $Q_{s,1,\dots\infty\dots,1}||\sum C_j$. Estos problemas de programación lineal, son fundamentales en la segunda demostración de que el precio de la anarquía de $Q_{s,1,\dots\infty\dots,1}||\sum C_j$ es $\frac{e}{e-1}$. Específicamente, sea $\mathcal{I}_{n,s}$ el conjunto de instancias de $Q_{s,1,\dots\infty\dots,1}||\sum C_j$ con n trabajos y tales que $s_1 = s$, se tiene el siguiente resultado:

Teorema 4.9. *Para todo $n \in \mathbb{N}$ y $s \geq 1$, se tiene que* $\mathbf{PPoA}(\mathcal{I}_{n,s}) \leq r_{n,s}$*, donde:*

$$\left(\mathcal{P}_{n,s}\right) \quad r_{n,s} = \text{máx} \quad \sum_{j=1}^{n} X_j \tag{4.3}$$

$$\text{s.a.} \quad 0 \leq p_1, \tag{4.4}$$

$$p_{j-1} \leq p_j, \qquad \forall j = 2, \dots, \lfloor s \rfloor, \tag{4.5}$$

$$X_1 \leq \frac{p_1}{s}, \tag{4.6}$$

$$X_j \leq X_{j-1} + \frac{p_j}{s}, \qquad \forall j = 2, \dots, n, \tag{4.7}$$

$$X_j \leq p_j, \qquad \forall j = \lfloor s \rfloor + 1, \dots, n, \tag{4.8}$$

$$\sum_{j=1}^{n} \alpha_j p_j = 1, \tag{4.9}$$

donde $\alpha_j = \text{mín}\{\frac{n+1-j}{s}, 1\}$.

Demostración. Dado $I \in \mathcal{I}_{n,s}$, con trabajos de tamaño $p_1, p_2, \dots, p_n$, sea N un equilibrio de Nash de I, definimos:

$$\widetilde{p}_k = \frac{p_k}{\sum_{j=1}^{n} \alpha_j p_j},$$

y

$$\widetilde{C}_k = \frac{{C_k}^N}{\sum_{j=1}^{n} \alpha_j p_j}.$$

Se verá que $(\widetilde{p}_1, \widetilde{p}_2, \ldots, \widetilde{p}_n, \widetilde{C}_1, \widetilde{C}_2, \ldots, \widetilde{C}_n)$ es un punto factible del problema de programación lineal $(\mathcal{P}_{n,s})$. En efecto, en las instancias definimos los trabajos ordenados por tamaño, y tienen tamaño positivo, por lo que será claro que $0 \leq \widetilde{p}_1$ y que $p_{j-1} \leq p_j$ para $j = 2, \ldots, n$.

El primer trabajo siempre elegirá la máquina más rápida, por lo cual ${C_k}^N = \frac{p_1}{s}$, de donde se tiene que $\widetilde{C}_k = \frac{\widetilde{p}_1}{s}$.

Para cualquier trabajo J_j, su tiempo de completación en N será menor o igual que el tiempo de completación que tendría si elige la máquina M_1. Además, la carga de la máquina M_1 antes de asignar J_j será menor o igual que ${C_{j-1}}^N$. Esto pues de no ser así, el último trabajo procesado en M_1 podría haber elegido $N(j-1)$ en vez de M_1, y esto reduciría su costo. Luego, ${C_j}^N \leq {C_{j-1}}^N + \frac{p_j}{s}$, por lo que $\widetilde{C}_j \leq \widetilde{C}_{j-1} + \frac{\widetilde{p}_j}{s}$.

Dado que estamos trabajando en una instancia de $Q_{s,1,\ldots\infty\ldots,1}\big|\big|\sum C_j$, para cualquier trabajo J_j existirá una máquina de velocidad 1 vacía disponible como estrategia. El costo de jugar en esa máquina es a lo más p_j, por lo que ${C_j}^N \leq p_j$, de donde $\widetilde{C}_j \leq \widetilde{p}_j$.

Con esto concluye la prueba de que $(\widetilde{p}_1, \widetilde{p}_2, \ldots, \widetilde{p}_n, \widetilde{C}_1, \widetilde{C}_2, \ldots, \widetilde{C}_n)$ es factible en $(\mathcal{P}_{n,s})$.

Pero, el óptimo de la instancia $I_{n,s}$ tiene costo

$$C_{opt}(I_{n,s}) = \sum_{j=1}^{n} \alpha_j p_j.$$

Por esto, se tendrá que

$$\mathbf{PPoA}(I_{n,s}) = \frac{\sum_{j=1}^{n} {C_j}^N}{\sum_{j=1}^{n} \alpha_j p_j} = \sum_{j=1}^{n} \widetilde{C}_j.$$

Este último valor es el costo de un punto factible de $(\mathcal{P}_{n,s})$, por lo que es menor o igual al óptimo. Por lo tanto:

$$\mathbf{PPoA}(I_{n,s}) \leq r_{n,s}.$$

□

4.3. Demostración con PL de la cota con un número infinito de máquinas

En esta sección se acotará superiormente por $\frac{e}{e-1}$ el valor óptimo de los problemas de programación lineal de la Sección 4.2. Esto demostrará que $\mathbf{PPoA}\left(Q_{s,1,\dots\infty\dots,1}\middle|\middle|\sum C_j\right) = \frac{e}{e-1}$, debido a que esos problemas acotan aquél precio de la anarquía.

Teorema 4.10. *Para todo $n \in \mathbb{N}$ y $s \geq 1$, se tiene que $r_{n,s} \leq \frac{e}{e-1}$.*

Demostración. La estrategia de esta demostración es la siguiente:

- Mostrar que el problema $(\mathcal{P}_{n,s})$ es acotado.
- Mostrar que lo anterior implica que todas las restricciones, excepto una están en igualdad.
- Calcular el valor exacto del óptimo, usando las igualdades antes mostradas, y acotar dicho valor por $\frac{e}{e-1}$.

Se probará primero que existe una solución factible del dual de $(\mathcal{P}_{n,s})$, con todas las variables positivas. Por el principio de dualidad débil, esto probará que el problema está acotado. En efecto, el dual de $(\mathcal{P}_{n,s})$ es:

$$
\begin{aligned}
\left(\mathcal{D}_{n,s}\right) \quad \text{mín} \quad & w && && (4.10)\\
\text{s.a.} \quad & y_2 - \frac{z_1}{s} + \alpha_1 w \geq 0, && && (4.11)\\
& y_{j+1} - y_j - \frac{z_j}{s} + \alpha_j w \geq 0, && \forall j = 2, \dots, \lfloor s \rfloor - 1, && (4.12)\\
& -y_{\lfloor s \rfloor} - \frac{z_{\lfloor s \rfloor}}{s} + \alpha_{\lfloor s \rfloor} w = 0, && && (4.13)\\
& -\frac{z_j}{s} - u_j + \alpha_j w = 0, && \forall j = \lfloor s \rfloor + 1, \dots, n, && (4.14)\\
& z_j - z_{j+1} = 1, && \forall j = 1, \dots, \lfloor s \rfloor, && (4.15)\\
& z_j - z_{j+1} + u_j = 1, && \forall j = \lfloor s \rfloor + 1, \dots, n-1, && (4.16)\\
& z_n + u_n = 1, && && (4.17)\\
& y_j \geq 0, && \forall j = 2 \dots \lfloor s \rfloor, && (4.18)\\
& z_j \geq 0, && \forall j = 2 \dots n, && (4.19)\\
& u_j \geq 0, && \forall j = \lfloor s \rfloor + 1, \dots, n. && (4.20)
\end{aligned}
$$

Sumando la Ecuación (4.14) para $j = n$, con la Ecuación (4.17) se obtiene:

$$-\frac{z_n}{s} + z_n + \alpha_n w = 1.$$

Luego, $z_n = \frac{s}{s-1}(1 - \alpha_n w)$.

Sumando la Ecuación (4.14) con la Ecuación (4.16), se obtiene para $j \in \{\lfloor s \rfloor + 1, \ldots, n-1\}$:

$$-\frac{z_j}{s} + z_j - z_{j+1} + \alpha_j w = 1.$$

De donde se tendrá que

$$z_j = \frac{s}{s-1}(z_{j+1}1 - \alpha_j w).$$

Finalmente, de la Ecuación (4.15), se obtiene que $z_j = 1 + z_{j+1}$, para $j \in \{1, \ldots, \lfloor s \rfloor\}$.

Definimos $a_j, b_j \in \mathbb{R}$ de modo que:

$$z_j = a_j - w b_j.$$

Luego:

$$a_j = \begin{cases} \frac{s}{s-1}, & \text{si } j = n, \\ \frac{s}{s-1}(a_{j+1} + 1), & \forall j \in \{\lfloor s \rfloor + 1, \ldots, n-1\}, \\ a_{j+1} + 1, & \forall j \in \{1, \ldots, \lfloor s \rfloor\}. \end{cases}$$

También:

$$b_j = \begin{cases} \frac{s}{s-1}\alpha_n, & \text{si } j = n, \\ \frac{s}{s-1}(b_{j+1} + \alpha_j), & \forall j \in \{\lfloor s \rfloor + 1, \ldots, n-1\}, \\ b_{j+1}, & \forall j \in \{1, \ldots, \lfloor s \rfloor\}. \end{cases}$$

De este modo se tiene:

$$a_j = \begin{cases} \sum_{t=\lfloor s \rfloor + 1}^{n} \left(\frac{s}{s-1}\right)^{t+1-j} + \lfloor s \rfloor + 1 - j, & \forall j \in \{1, \ldots, \lfloor s \rfloor\}, \\ \sum_{t=j}^{n} \left(\frac{s}{s-1}\right)^{t+1-j}, & \forall j \in \{\lfloor s \rfloor + 1, \ldots, n\}. \end{cases}$$

De donde:

$$a_j = \begin{cases} s\left(\frac{s}{s-1}\right)^{n-\lfloor s\rfloor} + \lfloor s\rfloor + 1 - s - j, & \forall j \in \{1,\dots,\lfloor s\rfloor\}, \\ s\left(\frac{s}{s-1}\right)^{n+1-j} - s, & \forall j \in \{\lfloor s\rfloor + 1,\dots,n\}, \end{cases}$$

y también

$$b_j = \begin{cases} \sum_{t=\lfloor s\rfloor+1}^{n} \alpha_t \left(\frac{s}{s-1}\right)^{t-\lfloor s\rfloor}, & \forall j \in \{1,\dots,\lfloor s\rfloor\}, \\ \sum_{t=j}^{n} \alpha_t \left(\frac{s}{s-1}\right)^{t+1-j}, & \forall j \in \{\lfloor s\rfloor + 1,\dots,n\}. \end{cases}$$

Observación 4.2. Notar que

$$a_1 \geq a_2 \geq \ldots \geq a_n = \frac{s}{s-1},$$

y que

$$b_1 \geq b_2 \geq \ldots \geq b_n = \frac{1}{s-1}.$$

Por otra parte, por la Ecuación (4.13), se tiene que:

$$y_{\lfloor s\rfloor} = -\frac{z_{\lfloor s\rfloor}}{s} + \alpha_{\lfloor s\rfloor} w,$$

y de la Ecuación (4.12), se tiene que para $j \in \{2,\dots,\lfloor s\rfloor - 1\}$ se cumple que:

$$y_j = y_{j+1} - \frac{z_j}{s} + \alpha_j w.$$

Definimos $c_j, d_j \in \mathbb{R}$ de modo que:

$$y_j = c_j + w d_j.$$

Así, se tendrá que:

$$c_j = \begin{cases} -\frac{c_{\lfloor s\rfloor}}{s}, & \text{si } j = \lfloor s\rfloor, \\ c_{j+1} - \frac{a_j}{s}, & \forall j \in \{2,\dots,\lfloor s\rfloor - 1\}, \end{cases}$$

y

$$d_j = \begin{cases} \frac{b_{\lfloor s\rfloor}}{s} + \alpha_{\lfloor s\rfloor}, & \text{si } j = \lfloor s\rfloor, \\ d_{j+1} + \frac{b_j}{s} + \alpha_j, & \forall j \in \{2,\dots,\lfloor s\rfloor - 1\}. \end{cases}$$

De lo anterior se deduce que para $j \in \{2, \ldots, \lfloor s \rfloor\}$

$$c_j = -\frac{1}{s}\sum_{t=j}^{\lfloor s \rfloor} a_t,$$
$$d_j = \frac{1}{s}\sum_{t=j}^{\lfloor s \rfloor} b_t + \sum_{t=j}^{\lfloor s \rfloor} \alpha_t.$$

Finalmente, se tendrá que:

$$u_j = \begin{cases} 1 - z_n, & \text{si } j = n, \\ 1 + z_{j+1} - z_j, & \forall j \in \{\lfloor s \rfloor, \ldots, n-1\}. \end{cases}$$

Lo que implica que:

$$u_j = \begin{cases} (1 - a_n) + wb_n, & \text{si } j = n, \\ (1 + a_{j+1} - a_j) + w(b_j - b_{j+1}), & \forall j \in \{\lfloor s \rfloor, \ldots, n-1\}. \end{cases}$$

Teniendo las variables duales calculadas, basta notar ahora que:

$$\begin{aligned} z_j &\geq 0, && \forall j \in \{2, \ldots, n\}, \\ y_j &\geq 0, && \forall j \in \{2, \ldots, \lfloor s \rfloor\}, \\ u_j &\geq 0, && \forall j \in \{\lfloor s \rfloor + 1, \ldots, n\}, \\ y_2 - \frac{z_1}{s} + \alpha_1 w &\geq 0. \end{aligned}$$

Esto prueba, por el principio de dualidad débil, que el problema $(\mathcal{P}_{n,s})$ es acotado.

Dado que el problema tiene $2n + 1$ restricciones y $2n$ variables, y como la restricción $p_1 \geq 0$ no es activa, se tendrá que todas las otras restricciones deben estar en igualdad, pues el óptimo será un punto extremo. De esta forma, se tiene que $p_1 = p_2 = \ldots = p_s$, y que para $j > s$, $p_j = \left(\frac{s}{s-1}\right)^{j-s} p_1$, y que $X_j = \sum_{k=1}^{j} \frac{p_k}{s}$. Por esto, el óptimo del problema tendrá la misma forma que el ejemplo de Hoeksma y Uetz [5], el cual tiene un valor menor a $\frac{e}{e-1}$.

□

Capítulo 5

Costo promedio como función social con un número finito de máquinas

En este capítulo se estudia el problema $Q_{s,1,\dots,1}||\sum C_j$, en el cual hay una máquina de velocidad $s \geq 1$, y el resto de máquinas tienen velocidad 1, sin asumir nada a priori sobre el número de máquinas. Sin embargo, en el capítulo anterior se calculó de forma exacta el precio de la anarquía para el caso en que $m + s > n$, por lo cual acá solo interesa el caso en que $m + s \leq n$.

Se probará en este capítulo que para el problema $Q_{s,1,\dots,1}||\sum C_j$, el precio de la anarquía está acotado superiormente por $\frac{5}{3}$, y se conjetura que ese valor es en verdad igual a $\frac{e}{e-1}$.

Por último, se mostrará un problema de programación lineal que acota superiormente el precio de la anarquía del caso más general, el problema $Q||\sum C_j$.

5.1. Problema Lineal que acota el precio de la anarquía con un número finito de máquinas

En esta sección se muestra un problema de programación lineal análogo al de la Sección 4.2 para el problema $Q_{s,1,\dots,1}||\sum C_j$. Este problema de programación lineal, tiene un óptimo que acota superiormente el precio de la anarquía para todas las instancias para cierto ambiente de máquinas y cierto número de trabajos ahí definido.

Definición 5.1. Dada una instancia I de $Q_{s,1,\dots,1}||\sum C_j$, con m máquinas, n trabajos y en la que la máquina rápida tiene velocidad $s \in \mathbb{N}$, definimos el valor de los coeficientes α_j de la siguiente forma:

$$\alpha_j := \text{máx}\left\{\frac{n+1-j}{s} + 1 - \left\lceil \frac{n+1-j}{m+s-1} \right\rceil,\ 1\right\} + \left\lceil \frac{n+1-j}{m+s-1} \right\rceil - 1.$$

Los coeficientes son definidos de esa forma, pues son el peso que tiene el trabajo en el costo óptimo. Esto se puede ver de forma más precisa en el siguiente lema.

Lema 5.2. *Dada una instancia I del problema de* scheduling $Q_{s,1,\ldots,1}|| \sum C_j$, *con m máquinas, n trabajos y en la que la máquina rápida tiene velocidad $s \in \mathbb{N}$, y donde los tamaños de los trabajos son $p_1 \leq p_2 \leq \ldots \leq p_n$, se tiene que:*

$$C_{opt}(I) = \sum_{j=1}^{n} \alpha_j p_j.$$

Demostración. En efecto, dado un *schedule* S cualquiera de I, su costo puede ser obtenido como

$$C_S(I) = \sum_{j=1}^{n} \frac{T_j^S}{s_{S(j)}} p_j,$$

donde T_j^S es el número de trabajos que son procesados en la misma máquina que el trabajo J_j y que no son procesados antes que J_j (esto incluye al mismo trabajo J_j) en el *schedule* S, y $s_{S(j)}$ es la velocidad de la máquina en la que J_j es procesado en S. Por esto, basta ver que $\frac{T_j^{OPT}}{s_{OPT(j)}} = \alpha_j$.

Por el algoritmo *MFT* se tendrá que en un *schedule* óptimo, los s trabajos más grandes irán en la máquina rápida, y los siguientes $m-1$ trabajos irán en una máquina de velocidad 1 cada uno. Luego, los siguientes s trabajos irán en la máquina rápida, y los siguientes $m-1$ irán en una máquina de velocidad 1 cada uno. Y así se irá repitiendo hasta que se acaben los trabajos por procesar.

En la última ronda de $m+s-1$ trabajos, los coeficientes que acompañarán a los trabajos en el óptimo serán $\left(1, 1, \ldots, 1, \frac{s-1}{s}, \frac{s-2}{s}, \ldots, \frac{2}{s}, \frac{1}{s}\right)$. Es decir, para los trabajos de la última ronda el coeficiente será $\alpha_j = \text{máx}\left\{\frac{n+1-j}{s},\ 1\right\}$. En la penúltima ronda, los coeficientes que acompañarán a los trabajos serán $\left(2, 2, \ldots, 2, 1+\frac{s-1}{s}, 1+\frac{s-2}{s}, \ldots, 1+\frac{2}{s}, 1+\frac{1}{s}\right)$. En general, en la ronda k desde el final hacia adelante, los coeficientes que acompañarán a los trabajos serán $\left(k, k, \ldots, k, k-1+\frac{s-1}{s}, k-1+\frac{s-2}{s}, \ldots, k-1+\frac{2}{s}, 1+\frac{1}{s}\right)$.

Dado que las rondas son de tamaño $k-1$, la ronda del trabajo J_j será $\left\lceil \frac{n+1-j}{m+s-1} \right\rceil$. De acá es claro que el coeficiente que acompaña al trabajo J_j en el *schedule* óptimo es:

$$\frac{T_j^S}{s_{S(j)}} := \text{máx}\left\{\frac{n+1-j}{s} + 1 - \left\lceil \frac{n+1-j}{m+s-1} \right\rceil,\ 1\right\} + \left\lceil \frac{n+1-j}{m+s-1} \right\rceil - 1$$

Esto prueba que $C_{opt}(I) = \sum_{j=1}^{n} \alpha_j p_j$.

□

Definición 5.3. Dada una instancia I del problema de *scheduling* $Q_{s,1,\dots,1}||\sum C_j$, con m máquinas, n trabajos y en la que la máquina rápida tiene velocidad $s \in \mathbb{N}$, definimos R como el tamaño de una ronda, es decir:

$$R := m + s - 1.$$

Observación 5.1. La definición anterior está motivada por el hecho de que en el óptimo, la estructura de como se asignan los trabajos es tal que los últimos R trabajos están asignados de la misma forma que los siguientes R trabajos, y así de forma consecutiva hasta terminar de asignarlos todos. Más aun, se tiene que

$$\alpha_{j-R} = \alpha_j + 1.$$

Lo anterior se tiene pues si el trabajo J_{j-R} está en una máquina de velocidad 1, se tiene que en el óptimo tiene exactamente un trabajo más arriba que el trabajo J_j, y si el trabajo J_{j-R} está en la máquina de velocidad s, en el óptimo tiene s trabajos más arriba que el trabajo J_j. En ambos casos, el coeficiente aumenta en 1.

A continuación se presenta el resultado principal de la sección, en el que se muestra que si se fijan el número de trabajos, el número de máquinas y la velocidad de la máquina más rápida, se tiene que el precio de la anarquía de todas las instancias que tienen esas máquinas y número de trabajos está acotado superiormente por el óptimo de un mismo problema de programación lineal.

Teorema 5.4. *El precio de la anarquía del juego $Q_{s,1,\dots,1}||\sum C_j$ está acotado superiormente por el óptimo del siguiente problema de programación lineal.*

$$r_{n,m,s} = \text{máx} \quad \sum_{j=1}^{n} X_j \tag{5.1}$$

$$\text{s.a.} \quad 0 \le p_1, \tag{5.2}$$

$$p_{j-1} \le p_j, \qquad \forall j = 2, \dots, n, \tag{5.3}$$

$$X_1 \le \frac{p_1}{s}, \tag{5.4}$$

$$X_j \le X_{j-1} + \frac{p_j}{s}, \qquad \forall j = 2, \dots, n, \tag{5.5}$$

$$X_j \le p_j, \qquad \forall j = s+1, \dots, R, \tag{5.6}$$

$$X_j \le p_j + X_{j-R}, \qquad \forall j = R+1, \dots, n, \tag{5.7}$$

$$\sum_{j=1}^{n} \alpha_j p_j = 1. \tag{5.8}$$

Demostración. Sea I una instancia del problema $Q_{s,1,\dots,1}||\sum C_j$ con n trabajos y m máquinas, de modo que $s_1 = s \in \mathbb{N}$. Sean $p_1, p_2, \dots, p_n$ los tamaños de los trabajos de la instancia I. Claramente podemos asumir sin pérdida de generalidad que $p_1 \leq p_2 \leq \dots \leq p_n$.

Dado N un equilibrio de Nash de I, definimos los siguientes valores:

$$\widetilde{p}_k = \frac{p_k}{\sum_{j=1}^{n} \alpha_j p_j},$$

y

$$\widetilde{C}_k = \frac{{C_k}^N}{\sum_{j=1}^{n} \alpha_j p_j}.$$

Se verá que que $(\widetilde{p}_1, \widetilde{p}_2, \dots, \widetilde{p}_n, \widetilde{C}_1, \widetilde{C}_2, \dots, \widetilde{C}_n)$ es un punto factible del problema de programación lineal $(\mathcal{P}_{n,s})$. En efecto, en cualquier instancia los trabajos son positivos, luego se cumple que $0 \leq \widetilde{p}_1$. Además, como asumimos sin pérdida de generalidad que los trabajos están ordenados por tamaño, se tendrá que $\widetilde{p}_{j-1} \leq \widetilde{p}_j$ para $j = 2, \dots, n$.

El primer trabajo siempre elegirá la máquina más rápida, por lo cual ${C_1}^N = \frac{p_1}{s}$, de donde se tiene que $\widetilde{C}_1 = \frac{\widetilde{p}_1}{s}$.

Para cualquier trabajo J_j, su tiempo de completación en N será menor o igual que el tiempo de completación que tendría si elige la máquina M_1. Además, la carga de la máquina M_1 antes de asignar J_j será menor o igual que ${C_{j-1}}^N$. Esto pues de no ser así, el último trabajo procesado en M_1 podría haber elegido $N(j-1)$ en vez de M_1, y esto reduciría su costo. Por esto, el costo que tendría el trabajo J_j de haber sido asignado a la máquina M_1 será menor o igual a ${C_{j-1}}^N + \frac{p_j}{s}$, de donde se tiene que ${C_j}^N \leq {C_{j-1}}^N + \frac{p_j}{s}$. Dividiendo por el costo óptimo se tiene que

$$\widetilde{C}_j \leq \widetilde{C}_{j-1} + \frac{\widetilde{p}_j}{s}.$$

Dado un trabajo J_j, con $j > R$, definimos el conjunto $T_j := \{J_{j-R+1}, J_{j-R+2}, \dots, J_{j-1}\}$ que tiene $R-1$ trabajos. Luego, dentro de este conjunto debe suceder una de dos cosas, o una máquina de velocidad 1 no tiene ningún trabajo de T_j asignado en ella, o la máquina M_1 tiene menos de s trabajos de T_j asignados a en ella. Sea M_i una máquina que cumple esto, y sea $\overline{C}_j$ el costo del trabajo J_j si fuera procesado en la máquina M_i en vez de la máquina en que está asignado en N. Como N es equilibrio de Nash, se tiene que ${C_j}^N \leq \overline{C}_j$.

Sea L_i^{j-R} la carga de la máquina M_i en el momento en el cual solo han sido asignados los trabajos $J_1, J_2, \dots, J_{j-R}$. Claramente se tiene que $L_i^{j-R} \leq {C_{j-R}}^N$.

Si M_i es una máquina de velocidad 1, se tiene que

$$\overline{C}_j = L_i^{j-R} + p_j.$$

Si M_i es la máquina de velocidad s (es decir, $i = 1$), sean $J_{j_1}, J_{j_2}, \ldots, J_{j_r}$ los trabajos de T_j que son asignados a M_i. Por definición de M_i, se tiene que $r < s$. Pero luego:

$$\begin{aligned}
\overline{C}_j &= L_i^{j-R} + \frac{1}{s}p_{j_1} + \frac{1}{s}p_{j_2} + \ldots + \frac{1}{s}p_{j_r} + \frac{1}{s}p_j \\
&\leq L_i^{j-R} + \frac{1}{s}p_j + \frac{1}{s}p_j + \ldots + \frac{1}{s}p_j + \frac{1}{s}p_j \\
&= L_i^{j-R} + \frac{r}{s}p_j \\
&\leq L_i^{j-R} + p_j.
\end{aligned}$$

Por esto, se tiene que en cualquier caso $\overline{C}_j = L_i^{j-R} + p_j$. Pero ${C_j}^N \leq \overline{C}_j$, y $L_i^{j-R} \leq {C_{j-R}}^N$, de donde se concluye que:

$${C_j}^N \leq {C_{j-R}}^N + p_j.$$

Finalmente, dividiendo por $\sum_{j=1}^{n} \alpha_j p_j$, se tiene que:

$$\widetilde{C}_j^N \leq \widetilde{C}_{j-R}^N + \widetilde{p}_j.$$

Con esto concluye la prueba de que $(\widetilde{p}_1, \widetilde{p}_2, \ldots, \widetilde{p}_n, \widetilde{C}_1, \widetilde{C}_2, \ldots, \widetilde{C}_n)$ es factible en $(\mathcal{P}_{n,s})$.

Pero, como se vio en el Lema 5.2, el óptimo de la instancia I tiene costo

$$C_{opt}(I) = \sum_{j=1}^{n} \alpha_j p_j.$$

Por esto, se tendrá que:

$$\mathbf{PPoA}(I) = \frac{\sum_{j=1}^{n} {C_j}^N}{\sum_{j=1}^{n} \alpha_j p_j} = \sum_{j=1}^{n} \widetilde{C}_j.$$

Pero este último valor es el costo de un punto factible del problema de programación lineal. Debido a esto se tendrá que $\sum_{j=1}^{n} \widetilde{C}_j$ será menor o igual al óptimo del problema, de donde se concluye que $\mathbf{PPoA}(I)$ es menor o igual al óptimo del problema de programación

lineal definido en el teorema.

□

5.2. Cota superior para un número finito de máquinas

Usando las desigualdades del problema de programación lineal de la Sección 5.1, ha sido posible acotar superiormente por $\frac{5}{3}$ el costo del precio de la anarquía de todas las instancias de $Q_{s,1,\dots,1}||\sum C_j$ que cumplen $s_1 \in \mathbb{N}$. Esto mejora la cota superior de 2 que era conocida hasta ahora para este caso, y deja el precio de la anarquía de este problema entre $\frac{e}{e-1}$ y $\frac{5}{3}$.

Teorema 5.5. *El precio de la anarquía del juego asociado al problema $Q_{s,1,\dots,1}||\sum C_j$, con $s \in \mathbb{N}$, está acotado superiormente por $\frac{5}{3}$.*

Demostración. La demostración incluye exclusivamente una manipulación de las desigualdades del problema de programación lineal dado en el Teorema 5.4, el cual cumplen los tiempos de proceso y los tiempos de completación en cualquier equilibrio de Nash.

Observemos que una instancia del problema (fijando las mismas máquinas) con n trabajos cuyos tiempos de proceso son $(p_1, p_2, \dots, p_n)$, es una instancia factible del problema con $n+1$ trabajos, cuyos tiempos de proceso son $(0, p_1, p_2, \dots, p_n)$. Luego, si acotamos por $\frac{5}{3}$ el problema de programación lineal para (n, m, s), entonces quedarán acotados por $\frac{5}{3}$ todos los problemas de programación lineal para (n', m, s), con $n' \leq n$. Por esto, será suficiente hacerlo para $n = Rk$, con $R := m + s - 1$ y $k \geq 3$.

Sea $I_{Rk,m,s}$ una instancia con Rk trabajos, m máquinas y $s_1 = s$. Se asumirá sin pérdida de generalidad que $\sum_{i=1}^{Rk} \alpha_j p_j$, para lo cual basta reescalar los tamaños de los trabajos p_j de modo que el precio de la anarquía no cambie.

Para este caso tendremos que:

$$\begin{aligned}\textbf{PPoA}\left(I_{Rk,m,s}\right) &= \sum_{j=1}^{Rk} C_j \\ &= \sum_{i=1}^{k} \left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} C_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} C_j \right).\end{aligned}$$

Pero en la suma de la derecha, iterando la Ecuación (5.5), se obtiene que

$$C_j \leq C_{(k-i)R+R-s} + \sum_{t=(k-i)R+R-s+1}^{j} \frac{p_t}{s}.$$

Luego, usando la Ecuación (5.7)

$$\begin{aligned}\mathbf{PPoA}\left(I_{Rk,m,s}\right) &\leq \sum_{i=1}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s}\left(p_j + C_{j-R}\right) + sC_{(k-i)R+R-s} + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\frac{(k-i+1)R+1-j}{s}p_j\right)\\ &\leq \sum_{i=1}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i\cdot p_j + sC_{(k-i)R+R-s} + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\frac{(k-i+1)R+1-j}{s}p_j\right).\end{aligned}$$

Además, iterando nuevamente la Ecuación (5.5), se tiene que

$$C_{(k-i)R+R-s} \leq \sum_{j=1}^{(k-i)R+R-s}\frac{p_j}{s}.$$

Sigue que

$$\begin{aligned}\mathbf{PPoA}\left(I_{Rk,m,s}\right) &\leq \sum_{i=1}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i\cdot p_j + \sum_{j=1}^{(k-i)R+R-s} p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\frac{(k-i+1)R+1-j}{s}p_j\right)\\ &\leq \sum_{i=1}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} 2i\cdot p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\left(i-1+\frac{(k-i+1)R+1-j}{s}\right)p_j\right)\\ &= \sum_{i=1}^{k}\left(2\cdot\sum_{j=(k-i)R+1}^{(k-i)R+R-s}\alpha_j p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\alpha_j p_j\right).\end{aligned}$$

La última igualdad se obtuvo de la definición de α_j. Por otra parte, se tiene que

$$\begin{aligned}\mathbf{PPoA}\left(I_{Rk,m,s}\right) &\leq \sum_{i=1}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i\cdot p_j + sC_{(k-i)R+R-s} + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\frac{(k-i+1)R+1-j}{s}p_j\right)\\ &\leq \sum_{i=1}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i\cdot p_j + i\cdot s\cdot p_{(k-i)R+R-s} + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\frac{(k-i+1)R+1-j}{s}p_j\right).\end{aligned}$$

Notemos que

$$s \cdot p_{(k-i)R+R-s} = s \underbrace{\frac{\sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} ((k-i+1)R+1-j)}{\frac{s(s+1)}{2}}}_{=1} p_{(k-i)R+R-s}$$
$$\leq 2\frac{s}{s+1} \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \tfrac{(k-i+1)R+1-j}{s} p_j$$
$$\leq 2 \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \tfrac{(k-i+1)R+1-j}{s} p_j.$$

Tenemos también que por la Ecuación (5.3)

$$s \cdot p_{(k-i)R+R-s} \leq \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} p_j.$$

Luego, se tendrá que

$$\mathbf{PPoA}\left(I_{Rk,m,s}\right) \leq \sum_{i=2}^{k} \left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i \cdot p_j + i \cdot s \cdot p_{(k-i)R+R-s} + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \tfrac{(k-i+1)R+1-j}{s} p_j \right)$$
$$+ \sum_{j=(k-1)R+1}^{kR-s} p_j + s \cdot p_{kR-s} + \sum_{j=kR-s+1}^{kR} \tfrac{kR+1-j}{s} p_j$$

$$\leq \sum_{i=2}^{k} \left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i p_j + s p_{(k-i)R+R-s} + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \left(i - 1 + \tfrac{(k-i+1)R+1-j}{s} \right) p_j \right)$$
$$+ \sum_{j=(k-1)R+1}^{kR-s} p_j + 2 \sum_{j=kR-s+1}^{kR} \tfrac{kR+1-j}{s} p_j + \sum_{j=kR-s+1}^{kR} \tfrac{kR+1-j}{s} p_j$$

$$\leq \sum_{i=2}^{k} \left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i \cdot p_j + \frac{1}{3} \cdot 2 \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \tfrac{(k-i+1)R+1-j}{s} p_j \right.$$

$$+\frac{2}{3}\sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\left(i-1+\tfrac{(k-i+1)R+1-j}{s}\right)p_j\Bigg)$$
$$+\sum_{j=(k-1)R+1}^{kR-s} p_j + 3\sum_{j=kR-s+1}^{kR}\tfrac{kR+1-j}{s}p_j$$

$$\leq \sum_{i=2}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i\cdot p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\left(i-1+\tfrac{2}{3}+\tfrac{5}{3}\tfrac{(k-i+1)R+1-j}{s}\right)p_j\right)$$
$$+\sum_{j=(k-1)R+1}^{kR-s} p_j + 3\sum_{j=kR-s+1}^{kR}\tfrac{kR+1-j}{s}p_j.$$

Pero como $i \geq 2$ se tiene que $i-1+\frac{2}{3} \leq \frac{5}{3}(i-1)$. Luego se tiene que

$$\mathbf{PPoA}\left(I_{Rk,m,s}\right) \leq \sum_{i=2}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} i\cdot p_j + \frac{5}{3}\sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\left(i-1+\tfrac{(k-i+1)R+1-j}{s}\right)p_j\right)$$
$$+\sum_{j=(k-1)R+1}^{kR-s} p_j + 3\sum_{j=kR-s+1}^{kR}\tfrac{kR+1-j}{s}p_j$$

$$=\sum_{i=2}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s}\alpha_j p_j + \frac{5}{3}\sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\alpha_j p_j\right) + \sum_{j=(k-1)R+1}^{kR-s}\alpha_j p_j$$
$$+3\sum_{j=kR-s+1}^{kR}\alpha_j p_j.$$

Finalmente, uniendo las dos desigualdades obtenidas se tendrá que:

$$\mathbf{PPoA}\left(I_{Rk,m,s}\right) \leq \frac{2}{3}\sum_{i=1}^{k}\left(2\cdot\sum_{j=(k-i)R+1}^{(k-i)R+R-s}\alpha_j p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\alpha_j p_j\right)$$
$$+\frac{1}{3}\left(\sum_{i=2}^{k}\left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s}\alpha_j p_j + \frac{5}{3}\sum_{j=(k-i)R+R-s+1}^{(k-i+1)R}\alpha_j p_j\right)\right.$$

$$+ \sum_{j=(k-1)R+1}^{kR-s} \alpha_j p_j + 3 \sum_{j=kR-s+1}^{kR} \alpha_j p_j \Bigg)$$

$$= \sum_{i=2}^{k} \left(\frac{5}{3} \sum_{j=(k-i)R+1}^{(k-i)R+R-s} \alpha_j p_j + \frac{11}{9} \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \alpha_j p_j \right) + \frac{5}{3} \sum_{j=(k-1)R+1}^{kR-s} \alpha_j p_j + \frac{5}{3} \sum_{j=kR-s+1}^{kR} \alpha_j p_j$$

$$\leq \sum_{i=2}^{k} \left(\frac{5}{3} \sum_{j=(k-i)R+1}^{(k-i)R+R-s} \alpha_j p_j + \frac{5}{3} \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \alpha_j p_j \right) + \frac{5}{3} \sum_{j=(k-1)R+1}^{kR-s} \alpha_j p_j + \frac{5}{3} \sum_{j=kR-s+1}^{kR} \alpha_j p_j$$

$$= \frac{5}{3} \sum_{i=1}^{k} \left(\sum_{j=(k-i)R+1}^{(k-i)R+R-s} \alpha_j p_j + \sum_{j=(k-i)R+R-s+1}^{(k-i+1)R} \alpha_j p_j \right)$$

$$= \frac{5}{3} \sum_{i=1}^{Rk} \alpha_j p_j$$

$$= \frac{5}{3}$$

□

5.3. Problema Lineal que acota el precio de la anarquía en máquinas con distintas velocidades

En esta sección se presenta un problema de programación lineal análogo al de la Sección 5.1, pero para una instancia cualquiera del problema $Q||\sum C_j$. Este problema tiene un número considerablemente más grande de ecuaciones, pero en general muchas de ellas serán redundantes.

Teorema 5.6. *Dada una instancia $I = (\mathcal{J}, \mathcal{M})$ del problema $Q||\sum C_j$, donde $\mathcal{J}$ tiene*

n trabajos, se tiene que $\mathbf{PPoA}(I)$ *está acotado superiormente por el óptimo del siguiente problema:*

$$\text{máx} \quad \sum_{j=1}^{n} X_j \tag{5.9}$$

$$\text{s.a.} \quad 0 \leq p_1 \tag{5.10}$$

$$p_{j-1} \leq p_j \qquad \forall j = 2, \ldots, n \tag{5.11}$$

$$X_j \leq X_{j-k} + \alpha_{n+1-k} p_j \qquad \forall j = 1, \ldots, n; \forall k = 1, \ldots, j \tag{5.12}$$

$$1 = \sum_{j=1}^{n} \alpha_j p_j \tag{5.13}$$

Demostración. Llamamos OPT al *schedule* entregado por el algoritmo óptimo MFT, con alguna regla para romper empates.

Denotamos por $h_i^k := \left|\left\{j \in \{n-k+1, \ldots, n\} \middle| OPT(j) = M_i\right\}\right|$. Es decir h_i^k es la cantidad de trabajos dentro de los últimos k que son asignados a la máquina M_i.

Sea N equilibrio de Nash de I. Se probará que el vector $\left(p_1, p_2, \ldots, p_n, {C_1}^N, {C_2}^N, \ldots, {C_n}^N\right)$ es factible para el PPL, lo cual probará que el precio de la anarquía es menor o igual que el óptimo del PPL, pues luego de reescalar los coeficientes, el precio de la anarquía de la instancia es igual a la función objetivo evaluada en ese vector.

Para ver que es factible, notemos que efectivamente $0 \leq p_1 \leq p_2 \leq \ldots \leq p_n$. Luego, solo basta probar que para $1 \leq j \leq n$ y para $1 \leq k \leq j$, se tiene que ${C_j}^N \leq {C_{j-k}}^N + \alpha_{n+1-k} p_j$.

En efecto, sea $1 \leq j \leq n$ y sea $1 \leq k \leq j$. Para cada par (j, k), definimos el conjunto $T_{k,j}$ como $T_{k,j} := \{J_{j-k+1}, J_{j-k+2}, \ldots, J_{j-1}\}$. En una de las máquinas M_i se debe tener que la cantidad de trabajos de $T_{k,j}$ asignados a M_i en el óptimo debe ser menor o igual a h_i^k, esto pues $\sum_{i=1}^{m} h_i^k = k > |T_{k,j}|$.

Sea $\widetilde{C}_j$ el costo del trabajo J_j si fuera procesado en la máquina M_i en vez de en $N(j)$. Como N es equilibrio de Nash, se tiene que ${C_j}^N \leq \widetilde{C}_j$.

Por otra parte, sea $L_i^d(N)$ la carga de la máquina M_i en el *schedule* N considerando solo los primeros d trabajos.

Claramente

$$ {C_{j-k}}^N = \max_{i \in \{1,2,\ldots,m\}} L_i^{j-k}(N). $$

Sean $J_{j_1}, J_{j_2}, \ldots, J_{j_r}$ los trabajos de $T_{k,j}$ que son asignados en N a la máquina M_i. Por definición de M_i, se tiene que $r < h_i^k$. Se tendrá que:

$$\begin{aligned}\widetilde{C}_j &\leq L_i^{j-k}(N) + \frac{1}{s_i}(p_{j_1} + p_{j_2} + \ldots + p_{j-r} + p_j) \\ &\leq L_i^{j-k} + \frac{1}{s_i} h_i^k p_j \\ &\leq {C_{j-k}}^N + \frac{h_i^k}{s_i} p_j\end{aligned}$$

Finalmente, observando que $\frac{h_i^k}{s_i} \leq \alpha_{n+1-k}$, se concluye que: ${C_j}^N \leq {C_{j-k}}^N + \alpha_{n+1-k} p_j$. □

Capítulo 6

Pruebas Computacionales

6.1. Problema a resolver

En el Capítulo 5, se vio que el valor precio de la anarquía del problema de *scheduling* $Q_{s,1,\ldots,1}||\sum C_j$ está entre $\frac{e}{e-1}$ y $\frac{5}{3}$. Sin embargo, el precio de la anarquía del problema de *scheduling* $Q_{s,1,\ldots\infty\ldots,1}||\sum C_j$ es $\frac{e}{e-1}$, y el precio de la anarquía de las peores instancias de $Q_{s,1,\ldots\infty\ldots,1}||\sum C_j$ solo puede hacerse más pequeño al disminuir el número de máquinas de velocidad 1, pues el costo del peor equilibrio de Nash se mantiene, y el costo del óptimo aumenta, es razonable suponer que el precio de la anarquía de $Q_{s,1,\ldots,1}||\sum C_j$ también es $\frac{e}{e-1}$.

Este problema continúa abierto y es la principal pregunta como trabajo futuro.

Conjetura 6.1. *El precio de la anarquía del juego de* scheduling $Q_{s,1,\ldots,1}||\sum C_j$ *es* $\frac{e}{e-1}$.

El principal argumento que sostiene esta conjetura es debido a que en pruebas computacionales realizadas se probó computacionalmente que para ciertos casos el precio de la anarquía no es superior a $\frac{e}{e-1}$.

Dado un número de trabajos n, un número de máquinas m y una velocidad $s \in \mathbb{N}$, se vio en la Sección 5.1 que el precio de la anarquía de cada una de las instancias posibles con n trabajos, m máquinas y $s_1 = s$ están acotadas superiormente por el óptimo del problema de programación lineal definido en el Teorema 5.4.

Notar además que una instancia de k trabajos, con $k < n$, puede ser representada por otra instancia de n trabajos, agregando $n - k$ trabajos de tamaño 0. Esto dice, que si $r_{n,m,s} \leq \frac{e}{e-1}$ para ciertos (n, m, s), entonces se tiene que el conjunto de instancias $\mathcal{I}$ del problema $Q_{s,1,\ldots,1}||\sum C_j$ que tienen m máquinas, $s_1 = s$, y k trabajos, con $k \leq n$ es tal que $\mathbf{PPoA}(\mathcal{I}) \leq \frac{e}{e-1}$.

$$r_{n,m,s} = \text{máx} \quad \sum_{j=1}^{n} X_j \tag{6.1}$$

$$\text{s.a.} \quad 0 \leq p_1 \tag{6.2}$$

$$p_{j-1} \leq p_j, \qquad \forall j = 2, \ldots, n, \tag{6.3}$$

$$X_1 \leq \frac{p_1}{s}, \tag{6.4}$$

$$X_j \leq X_{j-1} + \frac{p_j}{s}, \qquad \forall j = 2, \ldots, n, \tag{6.5}$$

$$X_j \leq p_j, \qquad \forall j = s+1, \ldots, R, \tag{6.6}$$

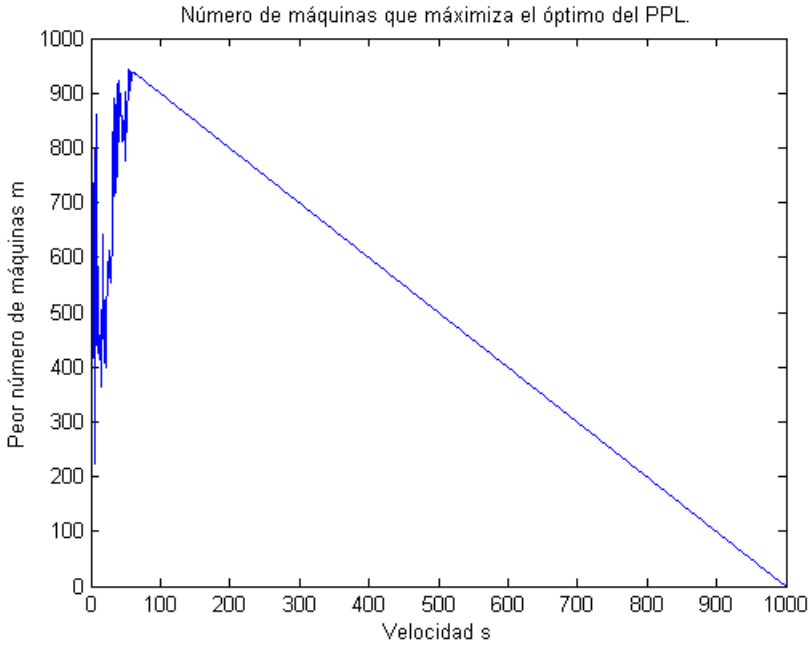
$$X_j \leq p_j + X_{j-R}, \qquad \forall j = R+1, \ldots, n, \tag{6.7}$$

$$\sum_{j=1}^{n} \alpha_j p_j = 1. \tag{6.8}$$

6.2. Pruebas realizadas

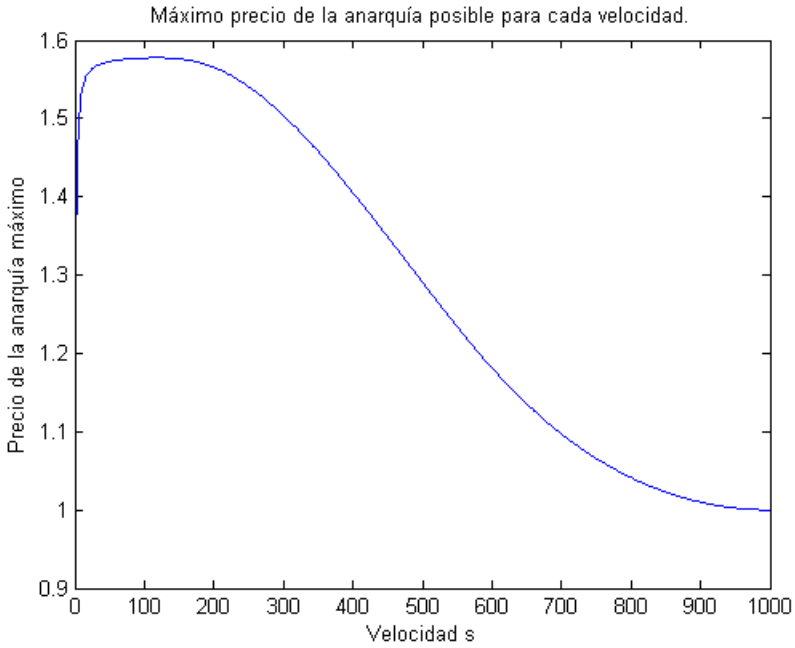
Se resolvieron computacionalmente todos los problemas de programación lineal ($\mathcal{P}_{n,m,s}$), tales que $n = 1000$, $s \geq 2$, $s \in \mathbb{N}$, $m \geq 1$ y $m + s \leq n$. Esta última condición se debe a que si $m + s > n$, la instancia pertenece a $Q_{s,1,\ldots\infty\ldots,1}||\sum C_j$, y en el Capítulo 4 se probó que $\textbf{PPoA}\left(Q_{s,1,\ldots\infty\ldots,1}||\sum C_j\right) = \frac{e}{e-1}$. Esto dio a lugar el valor óptimo para 498.501 problemas de programación lineal, cada uno con 2000 variables, los cuales fueron resueltos con el software IBM ILOG OPL-CPLEX.

El siguiente gráfico muestra para cada s, el valor de m con el que se alcanzó el mayor óptimo de alguno de los problemas de programación lineal para aquél s.



Este gráfico muestra que no se cumple necesariamente que el peor número de máquinas para cada s es el mayor m posible, lo cual solo se cumple a partir de $s \geq 66$.

El siguiente gráfico muestra el peor precio de la anarquía alcanzado para cada s, dentro de todos los valores de m posibles.



El máximo alcanzado en este gráfico es 1.577642, el cual se alcanza para $s = 119$, el cual tiene el peor precio de la anarquía para $m = 881$. Dado que $1.577642 < \frac{e}{e-1} \approx 1.582$, se concluye que para $n \leq 1000$ y $s \in \mathbb{N}$, se tiene que el precio de la anarquía es menor o igual a $\frac{e}{e-1}$.

Se detalla a continuación una tabla donde se muestra el peor precio de la anarquía para conjuntos de velocidades, divididos en intervalos de tamaño 50.

Máximo valor de PPL según intervalo para s.			
Intervalo para s	Peor s	Peor m	Valor máximo
[2,50]	50	903	1.5727
[51,100]	100	900	1.5773
[101,150]	119	881	1.5776
[151,200]	151	849	1.5761
[201,250]	201	799	1.5651
[251,300]	251	749	1.5406
[301,350]	301	699	1.5035
[351,400]	351	649	1.4569
[401,450]	401	599	1.4041
[451,500]	451	549	1.3478
[501,550]	501	499	1.2899
[551,600]	551	449	1.2329
[601,650]	601	399	1.1806
[651,700]	651	349	1.1351
[701,750]	701	299	1.0969
[751,800]	751	249	1.0658
[801,850]	801	199	1.0413
[851,900]	851	149	1.0228
[901,950]	901	99	1.0100
[951,999]	951	49	1.0025

Cabe notar que esto es una demostración computacional de que de existir alguna instancia con precio de la anarquía mayor al conjeturado y $s \in \mathbb{N}$, esta debe encontrarse con $n > 1000$.

Capítulo 7

Conclusiones y problemas abiertos

7.1. Principales aportes

Los principales aportes del presente trabajo se presentan a continuación:

Se estudió completamente el precio de la anarquía del juego de *scheduling* con máquinas idénticas, cuando la función objetivo es $\sum w_j C_j$ y también para la función objetivo $\sum C_j$. Esto se hizo en el caso de equilibrios de Nash en estrategias mixtas, lo que extiende los resultados ya conocidos en estrategias puras.

Se encontró una cota inferior de 2 para el juego de *scheduling* $Q||\sum w_j C_j$. Esto mejora la cota de $\frac{e}{e-1}$ existente para el problema de *scheduling* $Q||\sum C_j$, la cual también era aplicable a este problema.

Se presentan dos demostraciones distintas de que el precio de la anarquía del juego de *scheduling* $Q_{s,1,\ldots\infty\ldots,1}\big|\big|\sum C_j$ es exactamente $\frac{e}{e-1}$. Esto muestra que en el ambiente de máquinas ahí definido, el peor ejemplo conocido hasta ahora es efectivamente el peor ejemplo existente.

Se demostró que es posible obtener una cota superior de $\frac{5}{3}$ para el precio de la anarquía del juego de *scheduling* $Q_{s,1,\ldots,1}\big|\big|\sum C_j$.

Se mostró un problema de programación lineal cuyo óptimo acota superiormente el precio de la anarquía del conjunto de instancias del problema $Q||\sum C_j$ para un ambiente de máquinas fijo con velocidades cualquiera.

Se presentó como conjetura que el precio de la anarquía del juego $Q_{s,1,\ldots,1}\big|\big|\sum C_j$ es en realidad $\frac{e}{e-1}$ y se realizaron pruebas computacionales que, usando el problema de programación lineal, soportan la conjetura. Más aun, estas pruebas computacionales muestran que de

ser mayor el precio de la anarquía, un ejemplo debe ser buscado con un número mayor a 1000 trabajos.

7.2. Trabajo futuro

El principal problema abierto es continuar con el cálculo exacto del precio de la anarquía para el juego $Q_{s,1,\ldots,1}||\sum C_j$. Por las pruebas mostradas en el Capítulo 6 se puede esperar que el precio de la anarquía debe ser $\frac{e}{e-1}$.

El siguiente paso natural es calcular de forma exacta el precio de la anarquía del problema $Q||\sum C_j$, el cual se sabe que está entre $\frac{e}{e-1}$ y 2, pero se sospecha que es $\frac{e}{e-1}$.

Finalmente, queda por reducir el *gap* para el precio de la anarquía del juego $Q||\sum w_j C_j$, el cual por ahora solo se sabe que tiene un valor entre 2 y 4.

Bibliografía

[1] J. R. Correa and M. Queyranne. Efficiency of equilibria in restricted uniform machine scheduling with minsum social cost. 2010.

[2] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, first edition edition, January 1979.

[3] R. L. Graham, E.L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Discrete optimization: proceedings*, 1:287, 1979.

[4] R. Hoeksma. *Price of Anarchy for Machine Scheduling Games with Sum of Completion Times Objective*. PhD thesis, University of Twente, 2010.

[5] R. Hoeksma and M. Uetz. The Price of Anarchy for Minsum Related Machine Scheduling. In *Workshop on Approximation and Online Algorithms*, pages 41–52, 2011.

[6] E. Horowitz and S. Sahni. Exact and approximate algorithms for scheduling nonidentical processors. *Journal of the ACM*, 23(2):317–327, 1976.

[7] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[8] T. Kawaguchi and S. Kyan. Worst case bound of an LRF schedule for the mean weighted flow-time problem. *SIAM Journal on Computing*, 15(4):1119, 1986.

[9] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *STACS 99 (Trier)*, volume 1563 of *Lecture Notes in Comput. Sci.*, pages 404–413. Springer, Berlin, 1999.

[10] U. Schwiegelshohn. An alternative proof of the kawaguchi-kyan bound for the largest-ratio-first rule. *Oper. Res. Lett.*, 39(4):255–259, 2011.

[11] W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.