Sección de Matemáticas
Universidad de La Laguna

Ulises Pérez Romero

Teoría de juegos combinatorios

Combinatorial game theory

Trabajo Fin de Grado
Grado en Matemáticas
La Laguna, Junio de 2020

Dirigido por
Ignacio García Marco

Ignacio García Marco
Departamento de Estadística e Investigación Operativa
Universidad de La Laguna
38200 La Laguna, Tenerife

Agradecimientos

A Nacho por ser mi guía en este trabajo a lo largo del curso, por su ayuda incansable, por estar atento al whatsapp para cualquier duda y por todas esas mañanas demostrando teoremas y planificando esta obra.

A mi familia por darme todo su apoyo e impedirme que me rindiera en este duro camino que ha sido la carrera.

A mis padres quiero agradecerles también los inculcarme los valores de esfuerzo, lucha y sacrificio que me han llevado a superar todos mis obstáculos y llegar donde estoy hoy.

A Karim y a Marcos por su ayuda imprescindible en muchas asignaturas de la carrera y sin la cual me habría costado muchísmo más llegar hasta aquí.

A Antonella por sus ánimos y los momentos increíbles que hemos vivido estos años de universidad.

Y por último a todos mis amigos de Gran Canaria por ayudarme a desconectar de la carrera cuando me sentía agobiado y darme ánimos cuando más los necesitaba.

Muchas gracias a todos

Ulises Pérez Romero
La Laguna, 1 de junio de 2020

Resumen · Abstract

Resumen

En esta memoria se introducirá al lector en la Teoría de Juegos Combinatorios. La herramienta que nos permitirá modelizar los juegos combinatorios será un grafo dirigido: el grafo de estados. En particular, veremos que en estos grafos existe un conjunto de vértices característico, llamado núcleo y que será fundamental para describir estrategias ganadoras en un juego. Usaremos la teoría vista previamente para estudiar juegos como el NIM y el Chomp. A partir de dos juegos combinatorios, se puede definir uno nuevo denominado suma de los anteriores. Para el estudio de la suma de juegos veremos uno de los teoremas más importante de esta rama de las matemáticas: el Teorema de Sprague-Grundy. Generalizaremos los resultados obtenidos para la suma de dos juegos al caso de la suma de cualquier número de juegos combinatorios. Finalmente, trataremos la complejidad computacional que tiene el problema de encontrar estrategias ganadoras para los juegos combinatorios.

Palabras clave: *Juegos combinatorios – Grafo de estados – Núcleo de un grafo – Suma de juegos – Complejidad computacional*

Abstract

In this text we will introduce the reader in the Combinatorial Games Theory. The tool that will let us model combinatorial games will be a directed graph: the state graph of a game. We will see that these graphs have a distingished set of vertices, called kernel that will be fundamental to describe winning strategies in a game. We will use the theory studied before to analize games like NIM and Chomp. Starting from two combinatorial games, one can define a new game called the sum of the previous games. To study the sum of two combinatorial games we will see one of the most important theorems of this part of mathematics: the Sprague-Grundy Theorem. We will generalize the results obtained for the sum of two games to the case where we have any number of combinatorial games. Finally, we wil study the computational complexity of the problem of finding winning strategies for the combinatorial games.

Keywords: *Combinatorial games – States graph – Kernel of a graph – Sum of games – Computational complexity*

Contenido

Introducción

La Teoría de Juegos Combinatorios es una disciplina académica relativamente reciente. Los primeros análisis de juegos individuales aparecieron publicados en 1902, pero fue en 1930 cuando independientemente R. Sprague y P. M. Grundy desarrollaron una teoría para los juegos imparciales, que posteriormente fue ampliada por R. K. Guy y C. A. B. Smith. Desde entonces el interés por los juegos combinatorios va en aumento en una gran variedad de ramas: matemáticas, computación, inteligencia artificial, etc. Actualmente quedan muchos problemas abiertos y hay muchas investigaciones en este área de las matemáticas.

Los juegos combinatorios son juegos de dos personas con información perfecta, sin movimientos aleatorios y en los cuales no hay empates. En estos juegos se parte de una posición inicial y los jugadores se turnan para hacer su jugada, la cual conduce a una nueva posición. El juego prosigue hasta que se llega a una posición final, es decir, una situación en la que ningún movimiento es posible. Las reglas del juego establecen las características de un movimiento legal, el criterio de terminación y quién es el ganador en la posición final.

Para describir los juegos combinatorios y definir estrategias ganadoras haremos uso de la Teoría de Grafos. Dedicaremos el Capítulo 1 a describir los conceptos fundamentales de esta teoría que usaremos a lo largo de esta memoria. Entre estos conceptos destaca el de núcleo de un grafo dirigido, que usaremos constantemente para describir estrategias ganadoras. Otro concepto interesante que también veremos en este capítulo es el de sucesión de núcleos, y que está relacionado estrechamente con la llamada función de Sprague-Grundy, que trabajaremos en capítulos posteriores.

En el Capítulo 2 introducimos el concepto de juego combinatorio finito. A todo juego de este tipo se le asocia un grafo de estados y aquí es donde entran en juego los resultados expuestos en el Capítulo 1. Uno de los teoremas más importan-

tes de esta teoría es el Teorema de Zermelo, que afirma la existencia de estrategia ganadora en todo juego combinatorio finito. Enunciar y demostrar este teorema será nuestro cometido principal en el Capítulo 2. Además, introduciremos la función de Sprague-Grundy de un juego combinatorio, la cual asigna a cada estado de un juego combinatorio un valor numérico y que nos aportará una información adicional sobre el estado del juego que explotaremos en el Capítulo 4.

También veremos, en el Capítulo 3, algunos ejemplos de juegos combinatorios como el NIM, el Chomp en grafos y los llamados poset games, los describiremos y trataremos de diseñar una estrategia ganadora para el jugador que la posea. Todo ello se hará con los fundamentos teóricos introducidos en los dos capítulos previos.

A partir de dos juegos, se puede definir un nuevo juego denominado suma de juegos. Este es un juego en el que cada jugador puede elegir moverse en un juego u otro en cualquier punto del juego, y un jugador gana cuando su oponente no tiene movimientos posibles en ninguno de los juegos. El estudio de la suma de juegos lo llevaremos a cabo en el Capítulo 4. Para determinar qué jugador tiene una estrategia ganadora en la suma de juegos, resultará crucial el análisis de la función Sprague-Grundy introducida en el Capítulo 2, así como el análisis del juego NIM hecho en el Capítulo 3. Más concretamente, se demostrará que la suma de dos juegos es equivalente al juego de NIM con dos filas de palillos con a y b palillos, siendo a y b los valores de la función Sprague-Grundy de cada juego uno de los juegos involucrados en la suma. Finalmente, generalizaremos los resultados obtenidos a la suma de n juegos combinatorios.

En el Capítulo 5 estudiaremos el problema de determinar qué jugador tiene una estrategia ganadora como un problema de complejidad computacional. Como consecuencia de los resultados del Capítulo 3, determinar quién gana en juegos como el NIM está en la clase de complejidad P (de problemas polinomiales o fáciles). No obstante, este problema en muchos juegos combinatorios (en particular, todos los incluidos en esta memoria) se enmarcan en la clase de complejidad PSPACE. Además, se proponen ejemplos de juegos combinatorios tales que determinar qué jugador tiene una estrategia ganadora es PSPACE-completo (de los problemas más difíciles dentro de los problemas que se pueden solucionar con un tamaño de memoria polinomial).

Por último, presentaremos un código de un programa que se ha implementado en el lenguaje de programación C que determina qué jugador tiene estrategia ganadora en una partida del juego Chomp en grafos. Curiosamente, la implementación se sirve de los resultados de la Sección 3 para resolver el problema rápidamente

cuando el grafo es bipartito. En cambio, cuando el grafo no es bipartito, se sigue el algoritmo esbozado en el Capítulo 5, que si bien usa un tamaño de memoria polinomial, su tiempo de ejecución es exponencial en el tamaño de la entrada.

Las principales aportaciones en esta memoria son:

- La elección de la bibliografía y los juegos a tratar.
- La elección de la información, la distribución de la misma y la relación entre los capítulos.
- El de reescribir muchos resultados conocidos de teoría de juegos (como por ejemplo la estrategia en el juego de NIM) en términos de teoría de grafos. El objeto que permite este interfaz es el grafo de estados del juego y el concepto de núcleo de un grafo.
- Ligeras variaciones de las demostraciones de los resultados que se enuncian a lo largo del trabajo.
- Introducción del concepto de sucesión de núcleos y la Proposición 2.4.1 donde se demuestra la relación entre una sucesión de núcleos y el valor de la función Sprague-Grundy del grafo de estados de un juego.
- La implementación del código en el lenguaje de programación C de un programa que determina qué jugador tiene una estrategia ganadora para el juego de Chomp en grafos

1

Conceptos previos sobre la teoría de grafos dirigidos

Usaremos la teoría de grafos como herramienta para estudiar y modelizar los juegos combinatorios. En este primer capítulo veremos los conceptos básicos de esta teoría que utilizaremos más adelante. Los materiales de esta sección se han obtenido de [1] y [2].

1.1. Definiciones y propiedades

En esta sección estudiaremos una serie de definiciones para introducirnos en la teoría de grafos y una serie de propiedades que utilizaremos a lo largo de este trabajo.

Muchas situaciones del mundo real se pueden describir convenientemente por medio de un diagrama que consiste en un conjunto de puntos junto con líneas que unen ciertos pares de estos puntos. Por ejemplo, los puntos o vértices podrían representar personas con líneas que unen pares de amigos o ciudades conectadas por una red de carreteras. La abstracción matemática de situaciones de este tipo da lugar al concepto de grafo. Hay dos tipos de grafos en función de las líneas que unen los vértices. Cuando las líneas no tienen una dirección concreta, se les denomina aristas y dan lugar a los grafos no dirigidos. Por otro lado, cuando las líneas son unidireccionales, es decir, hay estrictamente un vértice de origen y un vértice de destino, estas líneas se llaman arcos y dan lugar a los grafos dirigidos. En este primer capítulo y en la mayor parte de este escrito trabajaremos con grafos dirigidos, pues son aquellos que nos permitirán modelizar los juegos combinatorios.

Definición 1.1 *Un grafo dirigido es un par* $G = (V, A)$ *donde* $V = \{v_1, v_2, \ldots, v_n\}$ *es un conjunto finito y* $A = \{a_1, a_2, \ldots, a_m\}$ *es un subconjunto de* $V \times V$*. Cada elemento* $(u, v) \in A$*, donde* u *y* v *son vértices del grafo, se denomina arco. En ocasiones por simplicidad suele denotarse por* $uv \in A$ *y suele representarse como una flecha que parte del vértice* u *y llega al vértice* v*, como vemos en la Figura 1.1.*

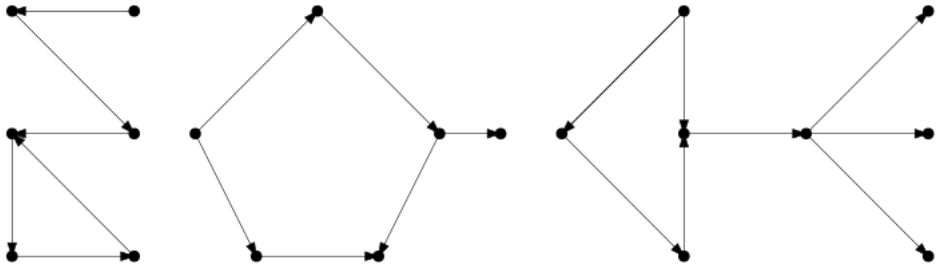

Figura 1.1. Representaciones gráficas de grafos dirigidos

Definición 1.2 *Se llama camino a una secuencia de vértices* $C = (u_0, u_1, \ldots, u_s)$ *dentro de un grafo tal que exista un arco entre cada vértice y el siguiente, pudiendo aparecer vértices o arcos repetidos. La longitud del camino es* s*, esto es, el número de arcos por los que se pasa en el camino, contando repeticiones. Se dice que un vértice* u *está conectado con un vértice* v *si existe un camino que empieza en* u *y termina en* v*. Se dice que un camino es cerrado si empieza y acaba en el mismo punto y, si solo pasa una vez por cada vértice excepto el primero, se denomina ciclo.*

Definición 1.3 *La distancia entre dos vértices* u *y* v *es la longitud del menor camino de* u *a* v*; es decir, el menor número de arcos que debemos recorrer para llegar desde* u *hasta* v*. Cuando no exista un camino de* u *a* v *diremos que* $d(u, v) = \infty$*. Cabe mencionar que no siempre se cumple que* $d(u, v) = d(v, u)$*.*

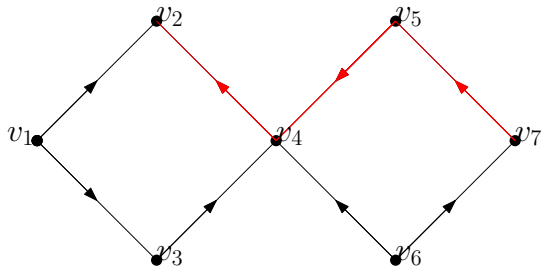

Figura 1.2. Grafo en el que destacamos en rojo la existencia de un camino de v_7 a v_2

En el grafo de la Figura 1.2 vemos que distancia de v_7 a v_2 es 3. También vemos que $d(v_1, v_7) = \infty$ ya que no existe ningún camino que comience en v_1 y acabe en v_7.

La presencia de caminos cerrados en un grafo dirigido supone la existencia de ciclos en el mismo, como veremos en el siguiente lema.

Lema 1.4 *Si un grafo $G = (V, A)$ tiene un camino cerrado, entonces tiene un ciclo. Además, si tiene un camino cerrado de longitud impar, entonces tiene un ciclo de longitud impar.*

Demostración
Sea $C = (u_0, u_1, \ldots, u_s, u_0)$ un camino cerrado. Si fuera un ciclo, ya tendríamos la prueba. De no serlo, sabemos que ese camino pasa al menos dos veces por algún vértice intermedio. Sea R uno de los vértices que se repiten, el camino es $C = (u_0, u_1, \ldots, u_{k-1}, R, u_{k+1}, \ldots, u_{h-1}, R, u_{h+1}, \ldots, u_s, u_0)$. De aquí, podemos extraer dos caminos cerrados de menor longitud: $C_1 = (u_0, u_1, \ldots, u_{k-1}, R, u_{h+1}, \ldots, u_s, u_0)$ y $C_2 = (R, u_{k+1}, u_{k+2}, \ldots, u_{h-1}, R)$. Si estos dos caminos fueran ciclos, ya tendríamos la prueba; pues la suma de sus longitudes es la longitud del camino inicial y como esta es impar, uno de los dos ciclos debería tener longitud impar. Si no fueran ciclos, aplicaríamos el razonamiento anterior, es decir, buscamos en ambos un vértice que se repita y dividimos cada camino en dos subcaminos de la forma en la que hicimos antes. El proceso se repite sucesivamente hasta que hayamos dividido el camino inicial en ciclos disjuntos. Como la suma de las longitudes de estos ciclos es igual que la del camino inicial, la cual es impar, necesariamente uno de los ciclos debe tener longitud impar.

Definición 1.5 *Sea v un vértice de un grafo, el grado de v,* $\deg(v)$*, es el número de arcos que inciden en él. En particular, se llama grado de entrada de v,* $\deg^+(v)$*, al número de arcos que tienen a v como vértice final y grado de salida,* $\deg^-(v)$*, al número de arcos que tienen a v como vértice inicial. Está claro que el grado de un vértice es la suma de su grado de entrada y de su grado de salida. Un tipo de vértice especial son aquellos que tienen grado de salida igual a cero, es decir, aquellos que no tiene vértices sucesores y que se denominan vértices terminales.*

En la Figura 1.2 vemos que $\deg^+(v_4) = 3$, ya que a este vértice llegan tres arcos, y vemos también que $\deg^-(v_4) = 1$ ya que de este vértice sale un arco.

Una propiedad característica de los grafos dirigidos acíclicos, es decir aquellos que no tienen ciclos, es la siguiente.

Lema 1.6 *Todo grafo dirigido $G = (V, A)$ acíclico tiene un vértice terminal, es decir, un vértice sin arcos salientes.*

Demostración
Supongamos por reducción al absurdo que G no tiene vértices terminales. Sea $V = \{v_1, v_2, \ldots, v_n\}$ el conjunto de vértices del grafo, como $\deg^+(v) > 1\ \forall v \in V$, existe un camino de n arcos y, por tanto, de $n+1$ vértices $C = (v_{i_1}, v_{i_2} \ldots v_{i_n}, v_{i_{n+1}})$.

Como el grafo solo tiene n vértices, en el camino anterior hay un vértice que se repite; luego, existe un camino cerrado y, por el Lema 1.4, un ciclo. Esto contradice la hipótesis de que el grafo sea acíclico y, por lo tanto, llegamos a un absurdo.

Veamos que con un subconjuntos de vértices y un subconjunto de arcos de un grafo $G = (V, A)$ podría llegar a formarse un grafo contenido en el anterior.

Definición 1.7 *Sean $G = (V, A)$ y $G' = (V', A')$ dos grafos, se dice que G' es un subgrafo de G si verifica que $V' \subset V$ y $A' \subset A$.*

Definición 1.8 *Sea $G' = (V', A')$ un subgrafo de $G = (V, A)$, se dice que G' es el subgrafo de G inducido por V', si $A' = (V' \times V') \cap A$, es decir, G' contiene todos las arcos de G que conectan vértices de V'. La notación que se utiliza es $G' = G[V']$. Además, sea $W \subset V$, denotaremos como $G - W$ al subgrafo inducido $G[V - W]$.*

Veamos ahora una familia especial de grafos determinada por la existencia de caminos entre cualquier par de vértices del mismo.

Definición 1.9 *Un grafo fuertemente conexo es aquel en el que para cualesquiera dos vértices u y v, existe un camino de u a v y uno de v a u.*
Además, las componentes fuertemente conexas de un grafo son sus subgrafos maximales fuertemente conexos. Cabe mencionar que la relación binaria uRv si, y solo si, u y v están en la misma componente conexa es una relación de equivalencia. Como consecuencia de ello, las componentes fuertemente conexas de un grafo forman una partición del conjunto de vértices.

En la Figura 1.3 se muestra un grafo en el que dado cualquier par de vértices existe un camino de ida y uno de vuelta entre ambos.

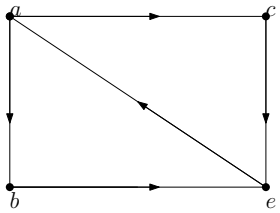

Figura 1.3. Ejemplo de grafo dirigido fuertemente conexo

Definición 1.10 *Sea $G = (V, A)$ un grafo con s componentes fuertemente conexas $F_1, F_2, \ldots, F_s$. Llamaremos grafo de componentes fuertemente conexas de G al grafo cuyos vértices son $F_1, F_2, \ldots, F_s$ y cuyos arcos son (F_i, F_j) si, y solo si, $\exists v_i \in F_i, v_j \in F_j$ tal que $(v_i, v_j) \in A$.*

Lema 1.11 *El grafo de componentes fuertemente conexas de un grafo $G = (V, A)$ es acíclico.*

Demostración
Supongamos, por reducción al absurdo, que hay ciclos en el grafo de componentes fuertemente conexas de G. Consideremos que hay un ciclo en el grafo de componentes fuertemente conexas $(F_1, F_2, \ldots, F_s, F_1)$, esto es, de s componentes fuertemente conexas que son distintas y, por lo tanto, disjuntas. Esto implica que hay un vértice de F_1 que está conectado con un vértice de F_2 y como todos lo vértices de una componente conexa están conectados por un camino, desde cualquier vértice de F_1 existe un camino de ida a cualquier vértice de F_2. Aplicando este razonamiento sucesivamente, desde cualquier vértice de F_1, existe un camino de ida a un vértice de F_s. Pero recordemos que estamos en un ciclo y F_s está conectada con la F_1 cerrando el ciclo. Por tanto, desde cualquier vértice de F_2 existe un camino de ida a cualquier vértice de F_1. Esto implica que dado un vértice cualquiera de F_1 y uno cualquiera de la F_s, existe un camino de ida y otro de vuelta. Luego, ambas componentes son iguales, es decir, tienen los mismos vértices; con lo cual llegamos a un absurdo, pues todas las componentes deberían ser disjuntas.

1.2. Núcleo de un grafo. Teorema de Richardson

En este apartado estudiaremos la definición de núcleo de un grafo dirigido y dos resultados previos que nos servirán para demostrar el Teorema de Richardson (ver [8] para la prueba original de Richardson y [2, Theorem 2.1] para una demostración más simple), que afirma la existencia de núcleo en grafos sin ciclos de longitud impar. Además, también demostraremos que todo grafo acíclico tiene un único núcleo.

Definición 1.12 *Un conjunto independiente de un grafo es un subconjunto de sus vértices tal que ninguno de ellos es adyacente a otro del mismo subconjunto. Es decir, dado $G = (V, A)$ un grafo dirigido, un subconjunto $V' \subset V$ de vértices se dice que es independiente si $\forall (u, v) \in A, u \notin V'$ ó $v \notin V'$.*

Definición 1.13 *Un conjunto absorbente de un grafo es un subconjunto de vértices del grafo tal que para cualquier vértice no perteneciente a este conjunto, existe un arco que va de ese vértice a uno de ese subconjunto. Es decir, sea $G = (V, A)$ un grado dirigido, un subconjunto $V' \subset V$ de vértices se dice que es absorbente si $\forall w \notin V', \exists u \in V'$ tal que $(w, u) \in A$.*

Definición 1.14 *Un núcleo de un grafo es un conjunto de vértices independiente y absorbente.*

El grafo de la Figura 1.2 tiene dos núcleos, que son $K_1 = \{v_1, v_4, v_7\}$ y $K_2 = \{v_2, v_3, v_5, v_6\}$, ya que ambos son subconjuntos independientes (dentro de cada subconjunto no hay ningún par de vértices conectados) y absorbente (cualquier vértice no perteneciente a uno de estos subconjuntos, está conectado con un vértice del mismo).

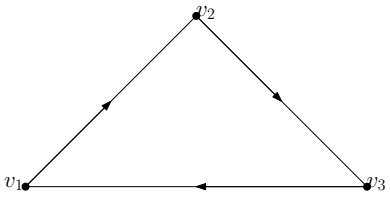

Figura 1.4. Grafo sin núcleos

Es fácil observar que no todo grafo tiene núcleos, como el grafo de la Figura 1.4. Por ejemplo, si v_1 estuviese en un núcleo K, v_2 no podría estarlo al ser sucesor de v_1. Pero v_3, al ser el único sucesor de v_2, tendría que estar en K, lo cual es absurdo porque v_1 es sucesor de v_3 y $v_1 \in K$. Este razonamiento es válido empezando desde cualquiera de los otros dos vértices dado que la situación es simétrica.

Comenzaremos demostrando la existencia de núcleos en aquellos grafos sin ciclos de longitud impar que sean fuertemente conexos, pues su prueba es más sencilla.

Lema 1.15 *Sea $G = (V, A)$ un grafo dirigido fuertemente conexo sin ciclos de longitud impar y sean $u, v \in V$, entonces $d(u, v)$ es par si, y solo si, $d(v, u)$ es par.*

Demostración
Sean $u, v \in V$ supongamos por reducción al absurdo que $d(u, v)$ es par y $d(v, u)$ es impar. Entonces, tendríamos un camino π de u a v de longitud par y un camino π' de v a u de longitud impar. Si concatenamos los caminos π y π' tendríamos un camino cerrado de longitud impar. Por el Lema 1.4, sabemos que esto implica que hay un ciclo de longitud impar, lo cual contradice la hipótesis y llegamos a un absurdo.

Proposición 1.16 *Si $G = (V, A)$ es un grafo dirigido fuertemente conexo sin ciclos de longitud impar y fijamos un vértice $u \in V$, entonces $K = \{v \in V \ / \ d(u, v) es\ par\}$ es un núcleo.*

Demostración
Veamos que K es un núcleo.

1. Sea $v \in K$, supongamos por reducción al absurdo que existe $w \in K$ tal que $(v, w) \in A$. Sea π un camino mínimo de u a v (par) y sea π' un camino mínimo de w a u, que sabemos que tiene longitud par por el Lema 1.15. Si concatenamos π con la arista (v, w) y después con el camino π', tenemos un ciclo de longitud impar. Esto contradice la hipótesis de que el grafo no posee ciclos de longitud impar. Por lo tanto, K es independiente.

2. Sea $v \in V - K$, como $d(u, v)$ es impar, por el Lema 1.15, $d(v, u)$ es impar. Sea $C = (v, w_1, w_2, \ldots, w_k = u)$ un camino mínimo de v a u. Vemos que w_1 es un seguidor de v en C y, por lo tanto, $C' = (w_1, \ldots, w_k = u)$ es un camino mínimo de w_1 a u y, por lo tanto, $d(w_1, u)$ es par. Por el Lema 1.15, $d(u, w_1)$ es par y, por lo tanto, $w_1 \in K$; luego, K es absorbente.

Definición 1.17 *Dado un grafo $G = (V, A)$, y un subconjunto de vértices $W \subset V$, denotaremos el conjunto de vértices predecesores a W como* $\text{pred}(W) = \{u \in V \ / \ \exists w \in W : (u, w) \in A\}$ *y el conjunto de sucesores a W como* $\text{suc}(W) = \{u \in V \ / \ \exists w \in W : (w, u) \in A\}$. *Por simplicidad, si $W = \{w\}$ (conjunto unitario) usaremos la siguiente notación:* $\text{suc}(W) = \text{suc}(w)$ *y* $\text{pred}(W) = \text{pred}(w)$.

Teorema 1.18 (Teorema de Richardson) *Todo grafo dirigido sin ciclos de longitud impar tiene un núcleo.*

Demostración
Sea $G = (V, A)$ un grafo dirigido. Por la Proposición 1.16, sabemos que si el grafo fuera fuertemente conexo tendría al menos un núcleo. Veamos ahora qué pasa si es grafo no fuese fuertemente conexo. Este caso procederemos a estudiarlo por inducción sobre el número de vértices del grafo.
Si $n = |V| = 1$, V es su propio núcleo.
Supongamos ahora que todo subgrafo de G tiene un núcleo. Consideramos el grafo de componentes fuertemente conexas de G, que sabemos por el Lema 1.11 que no tiene ciclos. Como consecuencia de ello, sabemos que existe, al menos, una componente fuertemente conexa que es vértice terminal (Lema 1.6). Ahora, sea C_1 una de esas componentes terminales y sea K_1 un núcleo de dicha componente (como es un subgrafo fuertemente conexo sin ciclos de longitud impar, por el Lema 1.11 sabemos que tiene un núcleo), consideramos el subgrafo $G - K_1 - \text{pred}(K_1)$, el cual, por hipótesis de inducción, tiene un núcleo, que llamaremos K_2. Afirmamos que $K = K_1 \cup K_2$ es un núcleo de G. En efecto:

1. Sea $v \in K$, entonces $v \in K_1$ ó $v \in K_2$. Separamos dos casos:

- Si $v \in K_1$, como K_1 es un núcleo de C_1 y esta componente es un vértice terminal en el grafo de componentes fuertemente conexas, todo sucesor de v debe estar en C_1 y fuera del núcleo K_1. Luego, $\nexists w \in K_1 \cup K_2$ tal que $(v, w) \in A$.
- Si $v \in K_2$ y sea $w \in G$ tal que $(v, w) \in A$. Como $v \notin \text{pred}(K_1)$ (por construcción), $w \notin K_1$. Si $w \in \text{pred}(K_1)$, entonces $w \notin K_2$. Y si $w \notin \text{pred}(K_1)$, entonces $w \in V - K_1 - \text{pred}(K_1)$ (conjunto de vértices de G') y como K_2 es un núcleo de G', entonces $w \notin K_2$. Luego, $w \notin K_1 \cup K_2$.

Por lo tanto, $\forall v \in K, \nexists w \in K$ tal que $(v, w) \in A$ (K independiente).

2. Sea $w \in V - K$ veamos que $\exists v \in K$ tal que $(w, v) \in A$.
   - Si $w \in C_1$, como K_1 es un núcleo de C_1, entonces existe $v \in K_1 \subset K$ tal que $(w, v) \in A$.
   - Si $w \in \text{pred}(K_1), \exists v \in K_1 \subset K$ tal que $(w, v) \in A$.
   - Si $w \in V - C - \text{pred}(K_1)$ (conjunto de vértices de G'), como K_2 es núcleo de G', existe $v \in K$ tal que $(w, v) \in A$.

   Por lo tanto, $\forall w \in G - K, \exists v \in K$ tal que $(w, v) \in A$ (K es absorbente).

Luego, como K es independiente y absorbente, $K = K_1 \cup K_2$ es un núcleo de G.

Observamos que la condición que exige que el grafo no tenga ciclos de longitud impar es fundamental para la demostración del teorema. Por ejemplo, en la Figura 1.4 se muestra un grafo con un ciclo de longitud impar que no tiene núcleos. Sin embargo, puede haber grafos con ciclos de longitud impar que sí tengan núcleo, como se muestra en la Figura 1.5.

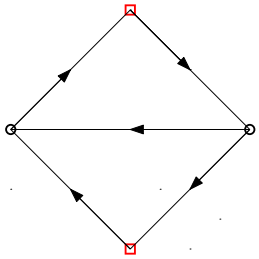

Figura 1.5. Con cuadrados aparecen representados los vértices del núcleo, ya que estos forman un subconjunto de vértices independiente y absorbente

En virtud del Teorema de Richardson los grafos acíclicos tienen núcleos. El resto de la sección la vamos a dedicar a ver que en este caso, además, el núcleo es único.

Lema 1.19 *Sea $G = (V, A)$ un grafo dirigido acíclico y $v \in V$ un vértice terminal. Definimos $V' = V - \{v\} - \text{pred}(v)$. Entonces, K es núcleo de V de si, solo si, $K' = K - \{v\}$ es un núcleo de $G' = G[V']$*

Demostración
($\Rightarrow$) Como el grafo es acíclico posee al menos un vértice terminal (Lema 1.6). Los vértices terminales han de estar en un núcleo, pues desde ellos no se puede llegar a ningún otro vértice.

1. Sea $u \in K - \{v\}$, como $u \in K$ y K es núcleo de G, $\nexists w \in K(\supset K - \{v\}))$ tal que $(u, w) \in A$. (independiente)
2. Sea $w \in V' - K' \subset V$ como K es un núcleo de G, existe $u \in K$ tal que $(w, u) \in A$. Como en V' no están los predecesores de v y $w \in V'$, entonces $u \in K - \{v\}$. (absorbente)

Luego, K' es núcleo de V'.

($\Leftarrow$) Por hipótesis $K - \{v\}$ es núcleo de V'. Veamos que K es núcleo de V.

1. Sea $u \in K$, veamos que este u no tiene sucesores en K.
   - Si $u = v$ se cumple, pues v es vértice terminal y no tiene sucesores; luego, no puede tener sucesores en el núcleo.
   - Si $u \neq v$, $u \in K - \{v\}$ y como $K - \{v\}$ es independiente, $\forall w \in V$ tal que $(u, w) \in A$ se verifica que $w \notin K - \{v\}$. Además, como $u \notin pred(v)$, $w \neq v$. Luego, $w \notin K$.

   Por tanto, K es un conjunto independiente en V.
2. Si $w \in V - K$,. Veamos que $\exists u \in K$ tal que $(w, u) \in A$.
   Si $w \notin K$, entonces $w \notin K - \{v\}$ y como $K - \{v\}$ es absorbente, $\exists u \in K - \{v\} \subset K$ tal que $(w, u) \in A$.
   Por tanto, K es un conjunto absorbente en V.

Luego, K es núcleo de V.

Teorema 1.20 (Unicidad del núcleo) *Todo grafo acíclico tiene un único núcleo.*

Demostración
Por el Teorema de Richardson ya tenemos garantizada la existencia de núcleo. Falta ver la unicidad. La probaremos por inducción sobre $|V| = n$
Para $n = 1, V$ tiene un único núcleo (el formado por el único vértice del grafo).
Supongamos cierta la propiedad para cualquier subgrafo de G con menos de n vértices y veamos que la propiedad se cumple también para G.
Sean K_1 y K_2 núcleos de V y $v \in V$ un vértice terminal, se tiene que $v \in K_1$ y $v \in K_2$. Por el lema anterior, $K_1 - \{v\}$ y $K_2 - \{v\}$ son núcleos de $V' = V - \{v\} - \text{pred}(v)$.

El subgrafo inducido en G por V' tiene como mucho $n-1$ vértices y podemos aplicar hipótesis de inducción (cualquier subgrafo de G tiene un único núcleo). Por tanto, $K_1 - \{v\} = K_2 - \{v\}$. Luego, $K_1 = K_2$, es decir, V tiene un único núcleo.

Veamos ahora un ejemplo de un grafo que tiene ciclos (de longitud par) y que tiene más de un núcleo (ver Figura 1.6).

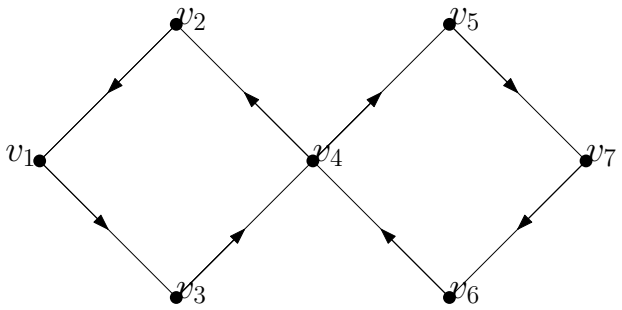

Figura 1.6. Ejemplos de grafo con ciclos y que presenta dos núcleos

Los núcleos de este grafo son $K_1 = \{v_1, v_4, v_7\}$ y $K_2 = \{v_2, v_3, v_5, v_6\}$.

1.3. Sucesión de núcleos

En este apartado introduciremos un concepto que generaliza al que hemos visto en el apartado anterior. Este es el concepto de sucesión de núcleos, el cual permitirá hacer una partición del conjunto de vértices y, posteriormente, asignar un valor numérico a cada una de las posiciones de un juego combinatorio, distinguiendo así los diferentes tipos posiciones a las que podremos movernos en un movimiento.

Definición 1.21 *Una sucesión de núcleos de un grafo $G = (V, A)$ es una partición de los vértices del grafo en subconjuntos no vacíos e independientes $K_0, K_1, \ldots, K_r$ de forma que si $w \in K_i$ con $i \in \{1, 2, \ldots, r\}$, entonces tiene arcos que van a vértices de K_j, $\forall j \in \{0, 1, \ldots, i-1\}$.*

En particular, en la definición anterior se tiene que K_0 es núcleo de G. Es más, se tiene el siguiente resultado:

Proposición 1.22 *Sea $G = (V, A)$ un grafo dirigido. Entonces, $K_0, K_1, \ldots, K_r$ es sucesión de núcleos de $V \Leftrightarrow K_i$ es núcleo de $G - (K_0 \cup K_1 \cup \ldots \cup K_{i-1}), \forall i \in 1, 2, \ldots, r$.*

Demostración
($\Rightarrow$) Veamos que K_i es núcleo de $G-(K_0 \cup K_1 \cup \ldots \cup K_{i-1}), \forall i \in \{1,2,\ldots,r\}$.

1. Por hipótesis, K_i es un conjunto independiente ($K_0, K_1, \ldots, K_r$ son todos conjuntos independientes por definición de sucesión de núcleos).
2. Sea $i \in \{0,1,2,\ldots,r-1\}$ y $w \in G-(K_0 \cup K_1 \cup \ldots \cup K_i)$, entonces $w \in K_s$ con $s \in \{i+1, i+2, \ldots, r\}$ (pues $K_0, K_1, \ldots, K_r$ forman una partición de G y, por tanto, w tiene que estar en alguno de ellos). Ahora, por hipótesis, w tiene arcos que van a $K_j, \forall j \in 0,1,2,\ldots,i$. En particular, w tiene arcos que van a K_i, esto es, $\exists u \in K_i$ tal que $(u,w) \in A$.
   Si $i = r$, entonces $G-(\cup_{j=0}^{r} K_j) = \emptyset$, por tanto, K_r también es absorbente. Por tanto, K_i es absorbente $\forall i \in \{0,1,2,\ldots,r\}$.

Luego, K_i es núcleo de $G-(K_0 \cup K_1 \cup \ldots \cup K_{i-1}), \forall i \in \{0,1,\ldots,r\}$.

($\Leftarrow$) Por hipótesis, K_i es núcleo de $G-(K_0 \cup K_1 \cup \ldots \cup K_{i-1}), \forall i \in \{1,2,\ldots,r\}$. Por lo tanto, los conjuntos K_i son no vacíos e independientes en $G-(K_0 \cup K_1 \cup \ldots \cup K_{i-1}) \subset G$. Falta ver que $\forall w \in K_i$, $\exists u_j \in K_j, \forall j \in \{0,1,\ldots,i-1\}$ tal que $(w,u_j) \in A$. Sabemos por hipótesis que $K_j, j < i$, es núcleo de $G-(K_0 \cup K_1 \cup \ldots \cup K_{j-1})$, por tanto, K_j es absorbente en $G-(K_0 \cup K_1 \cup \ldots \cup K_{j-1})$, esto es, $\forall v \in G-(K_0 \cup K_1 \cup \ldots \cup K_j)$, $\exists v_j \in K_j$ tal que $(v,v_j) \in A$. En particular, $w \in G-(K_0 \cup K_1 \cup \ldots \cup K_j)$, por lo tanto, $\exists u_j \in K_j$ tal que $(w,u_j) \in A$.
Luego, $K_0, K_1, \ldots, K_r$ es sucesión de núcleos.

Teorema 1.23 (Existencia de la sucesión de núcleos) *Todo grafo dirigido sin ciclos de longitud impar tiene una sucesión finita de núcleos.*

Demostración
Sea G un grafo en las condiciones anteriores, como G no tiene ciclos de longitud impar, por el teorema de Richardson, G tiene al menos un núcleo. Denotemos como K_0 a uno de esos núcleos. Si consideramos el grafo por $G-K_0$, vemos claramente que no tiene ciclos de longitud impar y, por el razonamiento anterior, también tiene un núcleo, denotando como K_1 a uno de ellos. Ahora, si consideramos el grafo $G-K_0-K_1$, el grafo resultante tampoco tiene ciclos de longitud impar y, por lo tanto, tiene un núcleo, que denotaremos como K_2. Repetimos el proceso hasta que $G-K_0-K_1-\cdots-K_r = \emptyset$, es decir, hasta que no queden vértices y, por lo tanto, haber construido una partición del grafo. La sucesión de grafos $G-\cup_{i=1}^{r} K_i$ tiene cada vez menos vértices y, por lo tanto, el proceso termina en un número finito de pasos, obteniendo como resultado una sucesión finita de núcleos.

Finalizamos el capítulo con el siguiente resultado, que es consecuencia directa del los Teoremas 1.20 y 1.23.

Corolario 1.24 *Si $G = (V, A)$ es un grafo dirigido acíclico, entonces tiene una única sucesión de núcleos.*

Demostración
Por el teorema de unicidad del núcleo en grafos acíclicos sabemos que G tiene un único núcleo. A partir de él construimos la sucesión de núcleos utilizando el razonamiento del teorema previo y teniendo en cuenta que cada subgrafo obtenido en el proceso es acíclico y, por ende, tiene un único núcleo.

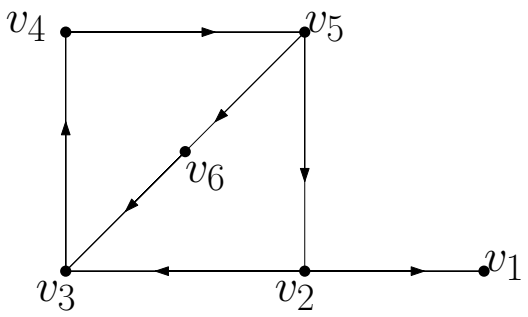

Figura 1.7. Ejemplos de grafo con ciclos pares que presenta dos sucesiones de núcleos finitas

Si G es un grafo no acíclico puede ser que tenga más de una sucesión de núcleos. En la Figura 1.7 puede verse un grafo dirigido que admite más de una sucesión de núcleos. Más concretamente, tiene dos sucesiones de núcleos que son las dos siguientes:
$K_{01} = \{v_1, v_3, v_5\}$, $K_{02} = \{v_2, v_4, v_6\}$
$K_{11} = \{v_1, v_4, v_6\}$, $K_{12} = \{v_2\}$, $K_{13} = \{v_3, v_5\}$

Los juegos combinatorios que veremos en el próximo capítulo se van a representar con grafos acíclicos y que, por lo tanto, tendrán una única sucesión de núcleos.

2

¿Qué es un juego combinatorio?

En esta sección comenzamos con la definición de juego combinatorio y de su grafo de estados que, como veremos, es un grafo dirigido acíclico y, por tanto, podemos aplicar los resultados del capítulo anterior en este contexto. También veremos el Teorema de Zermelo, que garantiza la existencia de estrategias ganadoras para estos juegos. Finalmente, introduciremos la función de Sprage-Grundy, muy importante en la búsqueda de estrategias ganadoras.

2.1. Definición

Los juegos combinatorios finitos se definen como aquellos que cumplen las siguientes propiedades:
P1. Hay dos jugadores que alternan sus movimientos, llamaremos A (Alice) al primero en jugar y B (Bob) al segundo.
P2. Hay un conjunto finito de posibles posiciones en el juego.
P3. Las reglas del juego especifican cuáles son los movimientos legales y son las mismas para los dos jugadores (es lo que se llama un juego imparcial).
P4. El juego termina cuando un jugador no puede hacer un movimiento. Aquí diferenciamos dos casos:

- Reglas normales: El último jugador en mover gana.
- Reglas misère: El último jugador en mover pierde.

P5. El juego termina en un número finito de movimientos, independientemente de las acciones de los jugadores.
P6. No se permiten movimientos al azar ni simultáneos ni escondidos.

Observación 1: Como consecuencia de P4 y P5 se tiene que no hay empates, es decir, la partida siempre finaliza y uno de los dos jugadores gana.

Observación 2: Las propiedades P2 y P6, que son condiciones de finitud, se pueden quitar, dando lugar a una familia mayor de juegos combinatorios en las que puede haber partidas infinitamente largas y/o un número infinito de estados. No obstante, este trabajo se centrará en los juegos finitos.

Observación 3: La propiedad P3 se puede relajar y se puede permitir que las reglas del juego sean diferentes para cada jugador. Esto da lugar a los juegos *no simétricos* o *partisanos*. Estos juegos se analizan con pequeñas variaciones de las herramientas que vamos a presentar en esta memoria. No obstante, en este trabajo nos centraremos en los juegos simétricos.

2.2. Grafo de estados de un juego combinatorio

Dado un juego combinatorio finito, asociamos un grafo dirigido al juego siendo cada vértice del grafo una posición o estado del juego y dos estados E_1 y E_2 están conectados por un arco si desde la posición E_1 un jugador puede pasar a la posición E_2 con un solo movimiento. Con las reglas normales, el jugador que está en una posición terminal en su turno, pierde. Por el contrario, con las reglas misère, el jugador que en su turno está en una posición terminal, gana.

Proposición 2.1 *El grafo de estados de cualquier juego combinatorio finito es acíclico.*

Demostración
En el grafo de estados, un ciclo representaría que en algún momento del juego es posible volver a una posición por la ya se ha pasado, lo cual dará lugar a la posibilidad de que el juego fuese infinito y esto contradice la propiedad P5.

Definición 2.2 (Notación) *Denotaremos la posición inicial del juego como x_0 y las posiciones finales o terminales como $x_{f_i}, \forall i \in \{1, 2, \ldots, k\}$, siendo k el número de posiciones terminales del juego. Se observa que las posiciones finales se corresponden exactamente con los vértices terminales del grafo de estados del juego.*

2.3. Teorema de Zermelo

En esta sección trataremos el Teorema de Zermelo (ver [11]), que afirma que en cualquier juego combinatorio finito existe una estrategia ganadora. Llamaremos *estrategia ganadora* a una secuencia de movimientos, de la cual dispone uno de los jugadores, que le llevará a este a la victoria independientemente de lo que haga el

otro jugador. Diremos que "A gana"(respectivamente, "que B gana") si el primer jugador (respectivamente el segundo) tiene una estrategia ganadora. Otro término importante y que usaremos con frecuencia es el de *posición ganadora o segura*; decimos que una posición p es ganadora si al modificar el juego y cambiar la posición inicial a p, entonces en el nuevo juego B tiene una estrategia ganadora. El término de *posición ganadora* viene de que esta es un estado del juego al cual se puede mover el jugador que tiene la estrategia ganadora para conservar esta condición. Veremos que el núcleo del grafo de estados de un juego combinatorio representa el conjunto de posiciones ganadoras del juego y, por lo tanto, A gana si, y solo si, el estado inicial del juego no pertenece al núcleo del grafo de estados. Luego, la estrategia ganadora, para el jugador que disponga de ella, consistirá en moverse a una posición del núcleo, que contendrá a los vértices terminales o posiciones finales, en cada uno de sus movimientos.

Teorema 2.3 (Teorema de Zermelo) *En cualquier juego combinatorio finito uno de los dos jugadores posee una estrategia ganadora.*

Demostración
Con reglas normales:
Sea $G = (V, A)$ el grafo de estados del juego y sean $x_{f_1}, x_{f_2}, \ldots, x_{f_k}$ las posiciones terminales del juego. Por la Proposición 2.1, tenemos que G es acíclico y por el Teorema 1.20 tiene un único núcleo K. Además, al ser K un núcleo, debe contener todos los vértices terminales de G. Si al principio del juego A se encuentra en un estado del juego no perteneciente al núcleo, deberá efectuar el movimiento que lo deje en una posición o vértice del núcleo (esto es posible pues el núcleo es un conjunto absorbente). En el siguiente turno, B, que se encuentra un estado del juego que pertenece al núcleo, no tendrá otra opción que moverse a una posición fuera del núcleo (pues el núcleo es un conjunto independiente). Esta situación se repetirá a lo largo del juego, por lo que B nunca tendrá la posibilidad de moverse al núcleo (donde se encuentran la posiciones ganadoras o terminales) con un movimiento suyo y será A quien la tenga. Como el juego es finito, en algún momento se llegará a una posición terminal, que será A quien acceda a ella y, por lo tanto, ganará. Por otro lado, si al principio de la partida A se encuentra en una posición del núcleo, no tendrá otra opción que irse a una posición que este fuera de este y en el siguiente turno B podrá acceder al núcleo con un movimiento, situación que, como en el caso anterior, se irá repitiendo hasta que finalice la partida, con victoria de B.

Con reglas misère:
Ahora, como G es acíclico, sabemos que $H := G - \{x_{f_1}, x_{f_2}, \ldots, x_{f_k}\}$ es acíclico y, por el Lema 1.6, que este nuevo grafo tiene vértices terminales, a los cuales denotaremos como x_i'. Estos vértices terminales de H están conectados en G con

algunos de los x_{f_i} y solamente tienen a estos como sucesores. Es decir, el jugador que se mueva a una posición x_i' obligará al otro jugador a moverse a una posición terminal en G y, por lo tanto, a perder. Hemos demostrado que el juego misère es equivalente al juego normal con grafo de estados H. Entonces, la estrategia ganadora ahora consistirá en moverse al núcleo del grafo H.

Corolario 2.4 *Sea $G = (V, A)$ el grafo de estados de un juego combinatorio y consideramos K el único núcleo de G y K' el único núcleo de $H = G - \{x_{f_1}, x_{f_2}, \ldots, x_{f_k}\}$. Entonces:*

- *Con las reglas normales,* A *gana* $\Leftrightarrow x_0 \notin K$
- *Con las reglas misère,* A *gana* $\Leftrightarrow x_0 \notin K'$

Como consecuencia se tiene que todo juego misère es equivalente a uno con reglas normales en el que ha modificado ligeramente el grafo de estados.

2.4. La función de Sprage-Grundy

Esta función fue introducida independientemente por R.P. Sprague y por P.M. Grundy en dos artículos, [10] y [5], respectivamente, para el estudio de juegos combinatorios con reglas normales. La función Sprague-Grundy de un juego con reglas normales y con grafo de estados $G = (V, A)$, que denotamos como sg, es una función definida en V y tomando valores enteros no negativos de manera que:

$$\mathrm{sg}(x) = \min\{n \in \mathbb{N} \ / \ n \neq \mathrm{sg}(y), \forall y \in \mathrm{suc}(x)\}$$

Veamos que la función sg está bien definida. Para ello, demostremos que todo vértice del grafo tiene un valor sg; esto es, $\forall x \in V, \exists n \in \mathbb{N}$ tal que $\mathrm{sg}(x) = n$

Como G es acíclico, por el Lema 1.6, tiene al menos un vértice terminal. A estos vértices terminales se les asocia el valor sg igual a 0, ya que estos vértices no tienen sucesores. A aquellos vértices que tengan como único sucesor uno de los vértices terminales, se les asigna el valor sg igual a 1. Posteriormente, a aquellos vértices cuyos sucesores ya tengan valor sg asignado, se les asocia el menor natural que no haya sido a ninguno de sus sucesores. Así, de forma recursiva se va asignando un valor sg a cada vértice hasta que todo el conjunto de vértices tengan valor sg.

Nota: Es fundamental que el grafo de estados de los juegos combinatorios sea acíclico para poder empezar a asignar valores sg a sus vértices terminales y, a partir de ellos, al resto de vértices.

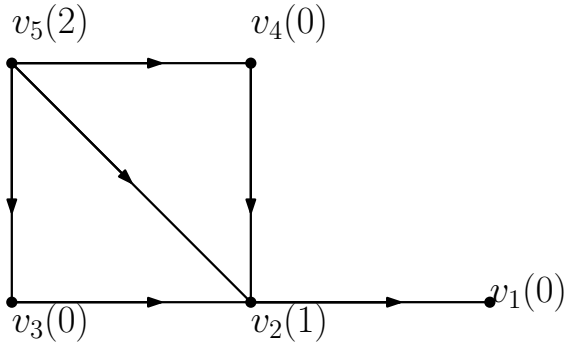

Figura 2.1. Ejemplo de asignación de la función Sprague-Grundy a un grafo dirigido acíclico

En la Figura 2.1 empezamos asignando valor sg igual a 0 al único vértice terminal del grafo, v_1. Posteriormente, asignamos valor sg igual a 1 su único predecesor, v_2. Luego, nos fijamos en aquellos que tienen a v_2 como único sucesor, que son v_3 y v_4 y les asignamos el valor sg igual 0. Por último a v_5 le asignamos valor sg igual 2, pues está conectado con vértices que tiene valor sg igual a 0 y 1, y el 2 es el menor natural aún no asignado a sus sucesores.

Como el grafo de estados de un juego combinatorio es acíclico, vimos en el Corolario 1.24 que tiene una única sucesión de núcleos. En el siguiente resultado vemos como la función Sprague-Grundy está estrechamente relacionada con la sucesión de núcleos.

Proposición 2.5 *Sea $G = (V, A)$ el grafo de estados de un juego combinatorio finito y sea $K_0, K_1, K_2, \ldots, K_r$ la única sucesión de núcleos de G. Entonces, $\forall x \in V$ $\mathrm{sg}(x) = i \Leftrightarrow x \in K_i$*

Demostración
Definimos $A_0 = \{x \in V \ / \ \mathrm{sg}(x) = 0\}$ y veamos que es el único núcleo de G.

1. Independiente. Si $\mathrm{sg}(x) = 0$, por la definición de la función sg se tiene directamente que $\nexists v \in \mathrm{suc}(x)$ tal que $\mathrm{sg}(x) = 0$.
2. Absorbente. Si $v \in V$ y $\mathrm{sg}(v) \neq 0$ ($\mathrm{sg}(v) > 0$), por definición de la función sg, $\exists w \in \mathrm{suc}(v)$ tal que $\mathrm{sg}(w) = 0$, esto es, $w \in A_0$.

Por lo tanto, A_0 es el único núcleo de V. Luego, $A_0 = K_0$

Siguiendo un argumento similar al anterior, veamos que, en general, $A_i = \{x \in V \ / \ \mathrm{sg}(x) = i\}$ es el único núcleo de $G - \cup_{j=0}^{i-1} K_j$, es decir, demostremos que $A_i = K_i$. Para ello, supongamos por inducción que $A_j = K_j, \forall j < i$ y veamos que $A_i = K_i$.

1. Independiente. Sea $u \in A_i$ ($\text{sg}(u) = i$), por la definición de la función sg, $\nexists v \in \text{suc}(u)$ tal que $\text{sg}(v) = i$; esto es, $\forall v \in suc(u), v \notin A_i$.
2. Absorbente. Sea $x \in V - \cup_{j=0}^{i-1} K_j$ y $x \notin A_i$, por hipótesis de inducción, se tiene que $x \in V - \cup_{j=0}^{i} A_j$. Por tanto, $\text{sg}(x) \notin \{0, 1, 2, \ldots, i\}$ y, por definición de función sg, $\exists v \in \text{suc}(x)$ tal que $\text{sg}(v) = i$, es decir, $v \in A_i$.

Por lo tanto, A_i es núcleo de $G - \cup_{j=0}^{i-1} K_j \ \forall i \in \{0, 1, 2, \ldots, r\}$. Luego, $A_i = K_i$ ($\text{sg}(x) = i \Leftrightarrow x \in K_i$)

Hemos visto que la estrategia ganadora de un juego combinatorio finito consiste en hacer un movimiento tal que el estado resultante del juego sea una posición del núcleo del grafo de estados. También hemos visto que los vértices de este núcleo tienen valor Sprague-Grundy igual a cero. Por tanto, la estrategia ganadora consiste en terminar después de cada movimiento en un vértice con valor Sprague-Grundy igual a cero. Esto puede comprobarse fácilmente al ver las condiciones siguientes:

1. Si x es una posición terminal, $\text{sg}(x) = 0$.
2. A la posición x para el cual $\text{sg}(x) = 0$, todo seguidor y de x es tal que $\text{sg}(y) \neq 0$.
3. A la posición x para el cual $\text{sg}(x) \neq 0$, existe al menos un seguidor y tal que $\text{sg}(y) = 0$.

Como se puede observar, para determinar las posiciones seguras, basta con saber si la función Sprague-Grundy vale cero o no. No obstante, la esta función nos da una información más fina, la cual explotaremos en el futuro cuando veamos la suma de juegos.

3

Algunos ejemplos de juegos combinatorios y búsqueda de la estrategia ganadora

En este capítulo describiremos algunos ejemplos de juegos combinatorios, como son el NIM, el Chomp en grafos y juegos del tipo "poset games", y buscaremos una estrategia ganadora para ellos. Evidentemente, aplicaremos los conocimientos presentados en los dos capítulos anteriores en la búsqueda y descripción de estrategias ganadoras para estos juegos combinatorios.

3.1. NIM

3.1.1. ¿En qué consiste el NIM?

El NIM es un juego combinatorio de dos jugadores (A y B) en el que, sobre la mesa, hay dos, tres o más filas de palillos o fichas y en el que, por turnos, los jugadores deben quitar tantos palillos como quieran de una fila que ellos escogen. En cada turno se ha de retirar al menos un palillo. El objetivo del juego es quitar el último palillo de la mesa, dejándola "limpia"; es decir, dejar a tu oponente sin movimientos posibles en el siguiente turno. Cabe recordar que este es un caso de juego combinatorio con reglas normales. El estudio original del juego de NIM y la descripción de la estrategia ganadora se pueden encontrar en [3]. Nosotros estudiaremos aquí este mismo juego con las herramientas de los capítulos anteriores.

Figura 3.1. Ejemplo de una partida de NIM

3.1.2. NIM con dos filas

Empezaremos analizando el caso en el que hay dos filas de fichas o palillos, pues es el más simple y, de hecho, es un buen ejercicio para principiantes encontrar la estrategia ganadora para este juego. Definimos $\mathrm{NIM} : \mathbb{N} \times \mathbb{N} \longrightarrow \{\mathrm{A}, \mathrm{B}\}$ donde $\mathrm{NIM}(a, b) = \mathrm{A}$ si A tiene la estrategia ganadora para el juego del NIM con a palillos en la primera fila y b palillos en la segunda. En caso contrario, $\mathrm{NIM}(a, b) = \mathrm{B}$.

En este caso, vamos a demostrar que ganará el primer jugador que tenga la posibilidad de dejar a su oponente con el mismo número de palillos en ambas filas; es decir, ganará A si al principio de la partida las dos filas tienen distinto número de palillos, y ganará B si al comienzo del juego ambas filas tienen igual número de palillos. Esto se debe a lo siguiente: si el jugador A se encuentra con las dos filas desiguales, siempre podrá quitar en la fila más numerosa los palillos suficientes para igualar el número de palillos en las dos filas, dejándole a B la única posibilidad de volver a desigualar las filas. Como A siempre se encuentra con un número desigual de palillos en las dos filas, siempre podrá jugar. No obstante, como el juego es finito, en algún momento B se encontrará ambas filas con cero palillos y, por lo tanto, perderá la partida. Concluimos este apartado con el siguiente teorema, del cual hemos esbozado la demostración, para el NIM con dos filas:

Teorema 3.1 *Sean $a, b \in \mathbb{N}$, entonces:*

$$\mathrm{NIM}(a, b) = \begin{cases} \mathrm{A}, & si\ a \neq b \\ \mathrm{B}, & si\ a = b \end{cases}$$

3.1.3. NIM con n filas

En este apartado introduciremos un teorema que determinará qué jugador tiene la estrategia ganadora en función de las condiciones iniciales del juego. Igual que en el apartado anterior, definimos $\mathrm{NIM} : \mathbb{N}^n \longrightarrow \{\mathrm{A}, \mathrm{B}\}$ donde $\mathrm{NIM}(a_1, \ldots, a_n) =$ A si A tiene la estrategia ganadora para el juego del NIM con a_1 palillos en la primera fila, a_2 palillos en la segunda, ... y a_n palillos en la n-ésima fila. En caso contrario, $\mathrm{NIM}(a_1, a_2, \ldots, a_n) = \mathrm{B}$.

Para describir la estrategia ganadora, antes vamos a definir una nueva operación en los números naturales, la suma NIM, que denotaremos con el símbolo $\oplus$. Esta suma se realiza algorítmicamente a partir de la escritura binaria de los enteros involucrados. Dado un número $z \in \mathbb{N}$, denotaremos por $z = (b_k \ldots b_0)_2$ si $(b_k \ldots b_0)$

es la estructura binaria de z, es decir, si $b_0, \ldots, b_k \in \{0,1\}$ y $z = \sum_{i=0}^{k} b_i 2^i$. Colocaremos los n números alineados por columnas de derecha a izquierda. Posteriormente, sumaremos estos números por columnas con la suma en $\mathbb{Z}_2$ sin considerar acarreos. Veamos un ejemplo de suma NIM:

Vamos a hacer la suma NIM de $6, 9$ y 15. Como se describió anteriormente, escribimos estos números en notación binaria. Se tiene que $15 = (1\ 1\ 1\ 1)_2$, $9 = (1\ 0\ 0\ 1)_2$ y $6 = (0\ 1\ 1\ 0)_2$. Posteriormente, los colocamos ordenados y hacemos la suma por columnas en $\mathbb{Z}_2$ sin acarreos.

$$\begin{array}{l} 1\,1\,1\,1 \to 15 \\ 1\,0\,0\ \ 1 \to 9 \\ \underline{0\,1\,1\ \ 0 \to 6} \\ 0\,0\,0\ \ 0 \to 0 \end{array}$$

Por lo tanto, la suma NIM de 6, 9 y 15 vale cero.

Se observa que para dos números naturales a y b la suma NIM vale cero si, y solo si, $a = b$. Teniendo en cuenta esto, podemos reescribir el Teorema 3.1 de la siguiente forma:

$$\text{NIM}(a,b) = B \Leftrightarrow a \oplus b = 0$$

Ahora, describimos el grafo de estados para este juego combinatorio. Si tenemos un juego NIM con n filas ($n \in \mathbb{N}$) y sean $a_1, a_2, \ldots, a_n$ el número de palillos en cada una de las filas, el grafo de estados de este juego es $G = (V, A)$, donde $V = \{(x_1, \ldots, x_n) \ / \ 0 \leq x_i \leq a_i\}$, $A = \{(x_1, \ldots, x_n)(y_1, \ldots, y_n) \ / \ x_i = y_i \ \forall i \neq j, y_j < x_j\}$. Se observa que el grafo de estados tiene $\prod_{i=1}^{n}(a_i + 1)$ vértices.

Como ya demostramos en el capítulo anterior, la estrategia ganadora, para el jugador que disponga de ella, consiste en moverse a una posición o estado perteneciente al núcleo del grafo de estados del juego combinatorio. Por lo tanto, lo que necesitamos para encontrar la estrategia ganadora del NIM es describir el núcleo de su grafo de estados.

Teorema 3.2 *El núcleo del grafo de estados del* NIM *está compuesto por aquellas situaciones en las que la suma* NIM *del número de palillos en cada una de las filas es igual a 0.*

Demostración
Veamos que el conjunto de vértices $K = \{(a_1, a_2, \ldots, a_n) \in V \ / \ a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0\}$ es un conjunto independiente y absorbente.

- *Independiente*. Supongamos, por reducción al absurdo, que desde una posición $(a_1, a_2, \ldots, a_n) \in K$ existe un movimiento a una posición $(a'_1, a'_2, \ldots, a'_n)$ también en K. Entonces, $\exists j \in \{1, 2, \ldots, n\}$ tal que $a_j < a'_j$ y $a'_i = a_i$, $\forall i \neq j$. Como $a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0 = a_1 \oplus \cdots \oplus a'_j \oplus \cdots \oplus a_n$, se tiene que $a_1 \oplus \cdots \oplus a'_j \oplus \cdots \oplus a_n \oplus a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0$. Entonces $a'_j \oplus a_j = 0$ y, por lo tanto, $a'_j = a_j$; lo cual es absurdo, ya que hemos considerado que eran distintos.

- *Absorbente*. De no darse que la suma NIM sea cero, quitando palillos de una fila elegida estratégicamente se puede conseguir que esto suceda. Veamos la estrategia para elegir esa fila. Recorremos las columnas de izquierda a derecha hasta encontrar en el resultado de la suma el primer 1, hecho que se produce debido a que en la columna en cuestión hay un número impar de unos. En esta columna, elegimos una de la filas que tenga un 1 y cambiamos ese 1 por un 0, obteniendo así un número par de unos y, por ende, un 0 en esa columna del resultado final. El resto de elementos de la fila elegida se irán cambiando de forma conveniente para que en todas las columnas restantes haya un número par de unos. En las columnas que no se verifique esto, de haber un 0 en la fila elegida, lo cambiaremos por un 1 y de haber 1, lo cambiaremos por un 0. Esto siempre puede hacerse, ya que se hace al quitar un determinado número positivo de palillos, lo cual es factible en cada turno.

Veamos un ejemplo de una partida de NIM que se encuentra en una situación con suma NIM distinto de cero y veamos qué fila seleccionar y cuál es el número de palillos a eliminar de la misma, esto es, veamos un movimiento ganador para este juego. Consideremos una partida de NIM con tres filas de 10, 17 y 21 palillos. La suma NIM se efectúa de la siguiente manera:

$$\begin{array}{l} 0\,1\,0\,1\,0 \to 10 \\ 1\,0\,0\,0\,1 \to 17 \\ \underline{1\,0\,1\,0\,1 \to 21} \\ 0\,1\,1\,1\,0 \to 14 \end{array}$$

Buscamos de izquierda a derecha en el resultado final la primera columna con un uno, que es la segunda columna. Vemos que de los sumandos de esta columna, solo uno de ellos es un uno y los demás son cero, por tanto, modificaremos la fila en la que se encuentra ese uno, que es la primera fila. Primero, en la segunda columna cambiaremos el uno mencionado por un cero. Revisando el resto de columnas a la derecha, vemos que en la tercera columna hay un 1 en el resultado final, por lo que cambiaremos en la primera fila el cero de la tercera columna por un uno, obteniendo un cero en el resultado final. En la cuarta columna vemos que también hay un uno en el resultado final, por lo que cambiaremos en la primera fila el uno

de la cuarta columna por un cero. Finalmente, como en la columna restante hay un cero en el resultado final, dejamos ese columna como estaba. En definitiva, lo que hemos hecho es cambiar el número $(0\ 1\ 0\ 1\ 0)_2$ de la primera fila, por el número $(0\ 0\ 1\ 0\ 0)_2$, que se traduce en pasar de tener en la primera fila 10 palillos a tener 4 palillos. Concluimos que la única estrategia ganadora en este estado del juego consiste en quitar seis palillos de la primera fila.

El Teorema 3.2 se puede reescribir de la siguiente manera:

Teorema 3.3 *Sea* $(a_1, a_2, \ldots, a_n) \in \mathbb{N}^n$. *Entonces:*

$$\text{NIM}(a_1, a_2, \ldots, a_n) = \text{B} \Leftrightarrow a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0$$

Para acabar esta sección, vamos a calcular explícitamente la función Sprague-Grundy de este juego combinatorio.

Teorema 3.4 *La función Sprage-Grundy para un juego* NIM *de n filas con un número de palillos por fila* $a_1, a_2, \ldots, a_n$ *es* $sg(a_1, a_2, \ldots, a_n) = a_1 \oplus a_2 \oplus \cdots \oplus a_n$

Demostración
Para demostrar este resultado, usaremos los siguientes tres resultados que ya hemos demostrado anteriormente:

(1) El Teorema 3.3. Es decir, $\text{NIM}(a_1, a_2, \ldots, a_n) = \text{B} \Leftrightarrow a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0$. Como ya sabemos que $\text{NIM}(a_1, a_2, \ldots, a_n) = \text{B}$, que es equivalente a que se cumpla que $sg(a_1, a_2, \ldots, a_n) = 0$, se tiene que $sg(a_1, a_2, \ldots, a_n) = 0$ si, y solo si, $a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0$.

(2) Si NIM está en el estado $(a_1, a_2, \ldots, a_n)$, entonces tras un movimiento estará en el estado $(a'_1, a'_2, \ldots, a'_n)$ donde $\exists j \in \{1, 2, \ldots, n\} : a_i = a'_i\ \forall i \in \{1, 2, \ldots, n\} - \{j\}$ y $a'_j < a_j$.

(3) Si $x, b \in \mathbb{N}$, $b \oplus x = 0 \Leftrightarrow b = x$

Hacemos la demostración por inducción.
Si $a_1 = a_2 = \ldots = a_n = 0$, entonces el resultado es claramente cierto porque $sg(a_1, a_2, \ldots, a_n) = 0$ y $a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0$.

Supongamos que el resultado es cierto para todo $(a'_1, a'_2, \ldots, a'_n)$ tal que $a'_i \leq a_i$ $\forall i \in \{1, 2, \ldots, n\}$ y $(a_1, a_2, \ldots, a_n) \neq (a'_1, a'_2, \ldots, a'_n)$ y veamos que se cumple también para $(a_1, a_2, \ldots, a_n)$.

Caso 1. Si $a_1 \oplus a_2 \oplus \cdots \oplus a_n = 0$, entonces por (1) tenemos que $sg(a_1, a_2, \ldots, a_n) = 0 = a_1 \oplus a_2 \oplus \cdots \oplus a_n$

Caso 2. Si $a_1 \oplus a_2 \oplus \cdots \oplus a_n = b \neq 0$. Para demostrar que $sg(a_1, a_2, \ldots, a_n) = b$ tenemos que ver que:

a) Después de cualquier movimiento la función Sprague-Grundy no vale b.
En efecto, supongamos por reducción al absurdo que tras un movimiento la función Sprague-Grundy vale b. Entonces, $\exists j \in \{1, 2, \ldots, n\}$ tal que $a_j < a'_j$ y $sg(a_1, \ldots, a'_j, \ldots, a_n) = b$; pero, por hipótesis de inducción $sg(a_1, \ldots, a'_j, \ldots, a_n) = a_1 \oplus \cdots \oplus a'_j \oplus \cdots \oplus a_n$.
Como $a_1 \oplus a_2 \oplus \cdots \oplus a_n = b = a_1 \oplus \cdots \oplus a'_j \oplus \cdots \oplus a_n$, se tiene que $a_1 \oplus \cdots \oplus a'_j \oplus \cdots \oplus a_n \oplus a_1 \oplus a_2 \oplus \cdots \oplus a_n = b \oplus b = 0$. Entonces $a'_j \oplus a_j = 0$ y, por lo tanto, $a'_j = a_j$; lo cual es absurdo, ya que hemos considerado que eran distintos.

b) Si $c < b$, existe un movimiento tal que la función Sprague-Grundy vale c. Consideramos el juego NIM$(a_1, a_2, \ldots, a_n, c)$, como $a_1 \oplus a_2 \oplus \cdots \oplus a_n \oplus c = b \oplus c \neq 0$ tenemos por (1) que NIM$(a_1, a_2, \ldots, a_n, c) =$ A. Entonces, existe un movimiento ganador para A. Hay dos opciones:
1) El movimiento es en una de las primeras n filas. Entonces, $\exists j \in \{1, 2, \ldots, n\}$ y $a'_j < a_j$ tal que NIM$(a_1, \ldots, a'_j, \ldots, a_n, c) =$ B y, por (3), tenemos que $a_1 \oplus \cdots \oplus a'_j \oplus \cdots \oplus a_n \oplus c = 0$. Aplicando hipótesis de inducción, tenemos que $a_1 \oplus \cdots a'_j \oplus \cdots \oplus a_n = sg(a_1, \ldots, a'_j, \ldots, a_n) = c$.
2) El movimiento es en la última fila. Veamos que esto no es posible. Si esto fuera posible, existiría $d < c$ tal que NIM$(a_1, a_2, \ldots, a_n, d) =$ B. Por (1), esto supone que $a_1 \oplus a_2 \oplus \cdots \oplus a_n \oplus d = 0$ y, por lo tanto, $a_1 \oplus a_2 \oplus \cdots \oplus a_n = b = d$, lo cual es imposible ya $d < c < b$.

3.2. Chomp en grafos

3.2.1. ¿En qué consiste el Chomp?

El Chomp es un juego de dos jugadores que juegan sobre un grafo no dirigido G. Cada uno de ellos, en su correspondiente turno, debe quitar o bien una arista o bien un vértice junto con todas sus aristas adyacentes. El objetivo final del juego es ser el jugador que quite el último vértice del grafo, dejando así a su rival sin movimientos posibles (reglas normales). Hasta ahora no se conoce una estrategia general para ganar en este juego. Sin embargo, sí son conocidos resultados que determinan el ganador y la estrategia que debe seguir para ganar la partida en el caso de grafos bipartitos, cuya definición vemos inmediatamente. El estudio del Chomp en grafos bipartitos se puede encontrar en [6].

3.2.2. Estrategia ganadora para grafos bipartitos

En este juego vamos a trabajar con grafos no dirigidos, que son grafos $G = (V, E)$, donde V es un conjunto de vértices y $E = \{\{u, v\} \ / \ u, v \in V\}$ es un conjunto de aristas.

Definición 3.5 *Un grafo se dice bipartito si existe una partición del conjunto vértices en dos subconjuntos U y W, de tal forma que si $\{u, v\} \in E$, entonces $u \in U$ y $v \in W$.*

Vemos un ejemplo de este tipo de grafos en la Figura 3.2.

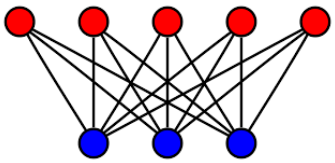

Figura 3.2. Ejemplo de grafo bipartito. Los conjuntos U y W de la partición están formados por los vértices azul y rojo, respectivamente

.

Como ya vimos en el Capítulo 2, la estrategia ganadora de un juego combinatorio, como el que estamos tratando, consiste en moverse a las posiciones del núcleo del grafo de estados del juego. Por lo tanto, para definir la estrategia ganadora, en primer lugar, debemos describir el grafo de estados de nuestro juego y, posteriormente, su núcleo.

Para describir el grafo de estados del Chomp en un grafo G, debemos tener en cuenta que al hacer un movimiento legal (quitar un vértice o una arista), lo que hacemos es pasar de un subgrafo de G a otro diferente. Por tanto, los estados del juego son cada uno de los posibles subgrafos de G. Luego, dado un grafo bipartito $G = (V, E)$, el grafo de estados del Chomp en G es $G_e = (V_e, A_e)$, donde $V_e = \{G' \ / \ G'$ es un subgrafo de $G\}$ y $A_e = \{(G'_1, G'_2) \ /$ podemos pasar de G'_1 a G'_2 quitando un vértice o una arista$\}$.

Para dar una idea del tamaño del grafo de estados de este juego, si G es el grafo completo con n vértices (es decir, cada par de vértices están conectados con una arista), entonces el grafo de estados tiene tantos vértices como subgrafos tiene G, que en este caso es $\sum_{i=0}^{n} \binom{n}{i} 2^{\binom{i}{2}}$, una cantidad exponencial en n.

Antes de enunciar el principal resultado de este apartado, en el que se describe el núcleo del grafo de estados de este juego y, por tanto, la estrategia ganadora, enunciaremos unos resultados previos que nos servirán para su demostración.

Teorema 3.6 (Teorema de los apretones de manos) *Dado un grafo $G = (V, E)$ con m aristas, se tiene $\sum_{u \in V} \deg(u) = 2m$. Además, si G es bipartito y U y W son los subconjuntos de vértices que constituyen la partición del grafo, se verifica que $\sum_{u \in U} \deg(u) = \sum_{u \in W} \deg(u) = m$.*

Demostración
Está claro que sumar los grados de los vértices es sumar las aristas con las que están conectados cada uno de ellos y que cada arista se cuenta dos veces (una por cada vértice). Luego, esa suma es el doble del número de aristas. En el caso de que G fuese bipartito, como todas las aristas del grafo conectan un vértice de U con uno de W y viceversa, se tiene que el número de aristas conectadas con vértices de U es m y el número de aristas conectadas con vértices de W es m; es decir, $\sum_{u \in U} \deg(u) = \sum_{u \in W} \deg(u) = m$.

Proposición 3.7 (Corolario de los apretones de manos) *Sea $G = (V, A)$ un grafo con un número impar de vértices, entonces existe un vértice de grado par.*

Demostración
Sea $V = \{u_1, u_2, \ldots, u_n\}$ el conjunto de vértices de G siendo $n = 2k+1$ con $k \in \mathbb{N}$ y m el número de aristas de G. Por el Teorema 3.6 sabemos que $\sum_{u \in V} \deg(u) = \sum_{i=1}^{n} \deg(u_i) = 2m$. Ahora, supongamos por reducción al absurdo que todos los vértices son de grado impar, es decir, $\deg(u_i) = 2k_i + 1, \forall i \in \{1, 2., \ldots, n\}$. Entonces, $\sum_{u \in V} \deg(u) = \deg(u_1) + \deg(u_2) + \cdots + \deg(u_n) = (2k_1+1) + (2k_2+1) + \cdots + (2k_n+1) = 2(k_1 + k_2 + \cdots + k_n) + 1(2k+1) = 2(k_1 + k_2 + \cdots + k_n + k) + 1 = 2m$ #
Llegamos a un absurdo y, por lo tanto, a la conclusión de que debe haber, al menos, un vértice de grado par.

Lema 3.8 *Sea $G = (V, E)$ un grafo bipartito con un número impar de vértices y de aristas, entonces existe un vértice de grado impar.*

Demostración
Sea m el número de aristas del grafo y supongamos por reducción al absurdo que todos los vértices son de grado par. Como G es bipartito, existen dos subconjuntos de vértices, U y W, tales que $U \cup W = V, U \cap W = \emptyset$ y si $\{u, v\} \in E$ entonces $u \in U$ y $v \in W$.
De la hipótesis de que G es bipartito, por el Teorema 3.6, tenemos que
$\sum_{u \in V} \deg(u) = \sum_{u \in U} \deg(u) + \sum_{u \in W} \deg(u) = 2m \Rightarrow \sum_{u \in U} \deg(u) = m$, lo cual es absurdo, pues por hipótesis tenemos que m es impar y hemos supuesto que todos los vértices son de grado par.
Luego, debe haber, al menos, un vértice de grado impar.

Ahora ya podemos describir qué jugador tiene una estrategia ganadora en el juego de Chomp en un grafo bipartito. Denotaremos Chomp$(G) = $ A si el jugador que empieza tiene una estrategia ganadora y Chomp$(G) = $ B en caso contrario.

Teorema 3.9 *Sea $G = (V, E)$ un grafo bipartito, el núcleo del grafo de estados del* Chomp *en G está formado por todos los subgrafos de G que tienen un número par de vértices y de aristas.*

Demostración
Veamos que el conjunto

$$K = \{G' \ / \ G' \text{ es un subgrafo de } G \text{ con un número par de vertices y aristas}\}$$

es el único núcleo del grafo de estados del Chomp en G; es decir, veamos que es un conjunto independiente y absorbente.

Independiente. Si nos encontramos en un estado del juego en el que el número de vértices y de aristas son pares, cualquiera que sea nuestro movimiento legal elegido nos lleva a romper esta paridad y, por lo tanto, a movernos a un estado que no está en el conjunto K.

Absorbente. Supongamos ahora que nos encontramos en un estado G' con n' vértices y m' aristas y que no pertenece a K. Veamos que desde G' es posible efectuar un movimiento que lleve a un estado perteneciente a K. Hay tres situaciones posibles:

1. n' impar y m' impar:
   Por el Lema 3.8, sabemos que en el grafo existe, al menos, un vértice de grado impar. Eliminando uno de estos vértices, estaríamos quitando además un número impar de aristas, quedando, por tanto, un número par de vértices y un número par de aristas.
2. n' par y m' impar:
   Eliminando una arista pasamos a un estado con un número par de vértices y aristas.
3. n' impar y m' par:
   Por la Proposición 3.7, como el número de vértices es impar, entonces ha de existir al menos un vértice de grado par. Quitando uno de esos vértices, estaríamos eliminando un número par de aristas y, por lo tanto, pasaríamos a un estado con un número par de vértices y un número par de aristas.

Finalmente, damos la estrategia ganadora en el siguiente Teorema, que es consecuencia directa del Teorema 3.9 y el Corolario 2.4.

Teorema 3.10 *Sea $G = (V, E)$ un grafo bipartito con n vértices y m aristas. Entonces:*

$$\text{Chomp}(G) = \begin{cases} \text{A}, \textit{ si } n \textit{ impar ó } m \textit{ impar} \\ \text{B}, \textit{ si } n \textit{ par y } m \textit{ par} \end{cases}$$

En la Figura 3.3 vemos un grafo bipartito con 5 vértices y 5 aristas. Por el Teorema 3.10, sabemos que A tiene la estrategia ganadora, la cual consistirá en eliminar un vértice de grado impar. Luego, las estrategias ganadoras para el jugador A son: eliminar el vértice v_2 ó eliminar el vértice v_5.

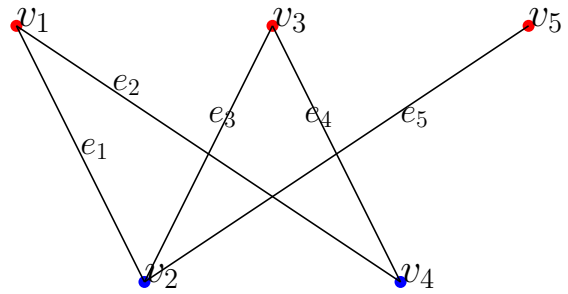

Figura 3.3. Ejemplo de Chomp en un grafo bipartito

Es un hecho sorprendente que decidir la estrategia ganadora en el juego de Chomp en el caso particular de un grafo bipartito solo depende del número de vértices y aristas y no de la estructura del grafo.

Como ya mencionamos anteriormente, para el caso del Chomp sobre grafos no bipartitos no se conoce una estrategia ganadora. Para ver en cada turno qué movimientos son ganadores, tendríamos que simular todas las posibles jugadas del juego y ver qué camino nos lleva a la victoria. Al final de esta memoria, propondremos un código de programación en lenguaje C para simular una partida de Chomp en cualquier tipo de grafo, en la que el programa sugiera al jugador un movimiento ganador si existiese un movimiento así. La idea es que el programa simula todas las posibles jugadas del juego y reconoce qué jugadas son buenas para el jugador que tiene que jugar, esto es, aquellas jugadas que dejan a su rival sin buenas opciones en el siguiente turno.

3.2.3. Función Sprague-Grundy para el juego de Chomp

En función de la paridad del número de vértices y del número de aristas, el valor de la función Sprage-Grundy en cada caso aparece reflejado en el Teorema 3.11.

Teorema 3.11 *Sea $G = (V, E)$ un grafo bipartito con n vértices y m aristas. El valor de la función Sprague-Grundy del Chomp en G en función de n y m se recoge en la siguiente tabla:*

$n \setminus m$	*par*	*impar*
par	*0*	*2*
impar	*1*	*3*

Demostración
Procedamos a demostrarlo por inducción sobre el par de valores (n, m). Claramente para el caso $(n, m) = (0, 0)$ se cumple. Ahora, supongamos el resultado cierto para cualquier grafo bipartito con $n' \leq n$ vértices y $m' \leq m$ aristas y $(n', m') \neq (n, m)$ y veamos que se cumple para G.
Veamos cada uno de los casos:

1. n par y m par
   Esta es la posición segura del juego (ya se demostró anteriormente) y está, por lo tanto, en el núcleo del grafo. Luego la función Sprage-Grundy en esta situación del juego vale 0 (ver Proposición 2.5).

2. n impar y m par.
   - Desde esta posición podemos eliminar un vértice de grado par (Lema 3.7), dejando así un número par de vértices y de aristas; es decir, una situación cuyo valor Sprage-Grundy vale 0.
   - No podemos pasar a una posición con valor sg igual a 1, ya que estas posiciones son aquellas en las que n' es impar y m' es par, igual que la que estamos estudiando, por lo que quitando un vértice o una arista pasamos a situaciones diferentes a esta.

   Como esta situación tiene sucesores con valor Sprague-Grundy igual a 0 y no se puede pasar a una con valor 1, tiene valor Sprague-Grundy igual 1.

3. n par y m impar
   - Desde esta posición podemos eliminar una arista, dejando así un número par de vértices y aristas, situación que recibe el valor sg igual 0.
   - También podemos eliminar un vértice de grado impar (Lema 3.8), dejando así un número impar de vértices y un número par de aristas, situación etiquetada en el subgrafo resultante con el valor sg igual a 1.

- No podemos pasar a una posición con valor sg igual a 2, ya que es una posición con n' par y m' impar, igual a la que estamos estudiando.

Como esta situación tiene sucesores con valores Sprague-Grundy 0 y 1 y desde esta no se puede pasar a una situación con valor 2, esta tiene valor Sprague-Grundy igual a 2

4. n impar y m impar
   Desde esta posición tenemos garantizada la existencia de tres movimientos:
   - Eliminar un vértice de grado impar (Lema 3.8), dejando así un número par de vértices y de aristas, situación que recibe el valor sg igual a 0.
   - Eliminar una arista, dejando así un número impar de vértices y un número par de aristas, situación que catalogada en el subgrafo resultante con el valor sg igual a 1.
   - Eliminar un vértice de grado par (Proposición 3.7), dejando así un número par de vértices y un número impar de aristas, situación etiquetada en el subgrafo resultante con el valor sg igual a 2.

   Como los seguidores de esta situación ya reciben los valores sg 0, 1 y 2, la situación "n y m impares"debe tener el valor Sprague-Grundy igual a 3.

3.3. Poset games

3.3.1. Introducción

Un conjunto parcialmente ordenado (también llamdo poset) formaliza el concepto intuitivo de un orden, secuencia o disposición de los elementos de un conjunto. Un poset consiste en un conjunto junto con una relación binaria que indica que, para ciertos pares de elementos del conjunto, uno de los elementos precede al otro en el orden. La relación en sí misma se llama "orden parcial". La palabra parcial en los nombres "orden parcial" y "conjunto parcialmente ordenado" se utiliza como una indicación de que no todos los pares de elementos deben ser comparables. Es decir, puede haber pares de elementos para los cuales ninguno de los elementos precede al otro en el poset. Formalmente, un orden parcial es cualquier relación binaria que es reflexiva (cada elemento es comparable a sí mismo), antisimétrica (no hay dos elementos diferentes que se precedan entre sí) y transitiva (el inicio de una cadena de relaciones de precedencia debe preceder al final de la cadena).

Los poset se representan mediante su diagrama de Hasse. Un diagrama de Hasse es una representación gráfica simplificada de un conjunto parcialmente ordenado finito. Esto se consigue eliminando información redundante. Para ello se

dibuja una arista ascendente entre dos elementos solo si uno sigue a otro sin haber otros elementos intermedios. Más adelante, cuando veamos el desarrollo de los poset games, veremos algunos ejemplos de estos juegos y sus correspondientes diagramas de Hasse.

3.3.2. Desarrollo del juego

En la teoría de juegos combinatorios, los juegos poset son juegos matemáticos de estrategia, que generalizan muchos juegos conocidos como NIM y Chomp. En tales juegos, dos jugadores comienzan con un poset y se turnan para elegir un punto en el poset, eliminarlo y todos los puntos que son mayores.

Dado un conjunto parcialmente ordenado $(P, \preceq)$, denotamos como $P_x = P - \{a | x \preceq a\}$ el poset obtenido por la eliminación de x en P. Un juego poset entre dos jugadores, A y B, es el siguiente:

- A elige un punto $x \in P$; reemplazando así P por P_x y luego le pasa el turno a B, que juega en P_x y luego le pasa el turno a A.
- Con las reglas normales, pierde el jugador que "come" el último punto, mientras que con las reglas misère un jugador pierde si en su turno ya no hay puntos para elegir.

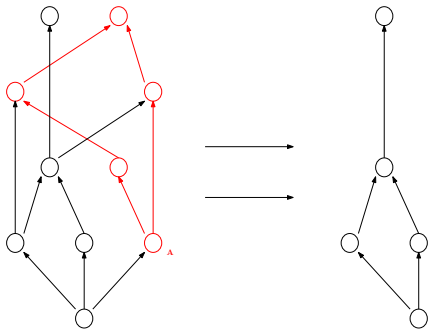

Figura 3.4. Ejemplo de poset game y movimiento

Para verificar que el NIM y el Chomp en grafos son poset games propongamos el poset en ambos casos.

- *NIM.* El poset del NIM jugado con n filas de $a_1, a_2, \ldots, a_n$ palillos es el par $(P_{NIM}, \preceq)$, donde $P_{NIM} = \{(i, j) \ / \ 0 \leq i \leq n,\ 0 \leq j \leq a_i\}$ y $(i, j) \preceq (i', j')$ si, y solo si, $i = i'$ y $j \leq j'$. El elemento $(i, j) \in P_{NIM}$ representa el palillo j-ésimo en la i-ésima fila. Por lo tanto, consiste en n cadenas de longitudes $a_1, a_2, \ldots, a_n$.

Se ve claramente que si eliminamos el palillo (i, j) también quitamos todos los palillos de la fila i-esima que están en las posiciones $k \geq j$. En la Figura 3.5 vemos el diagrama de Hasse del poset para el caso particular de NIM con tres filas, una con tres palillos, una con dos y la otra con cuatro.

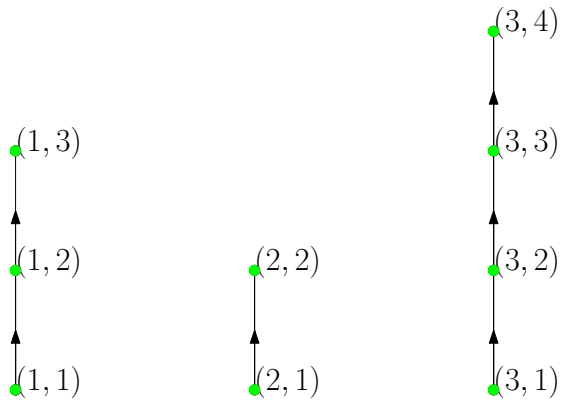

Figura 3.5. Diagrama de Hasse de NIM$(3, 2, 4)$

.

- *Chomp*. El poset del Chomp jugado en un grafo $G = (V, E)$ es un par $(P_{chomp}, \preceq)$, donde $P_{chomp} = V \cup E$ y $v_i \preceq e_j$ si, y solo si, el vértice $v_i \in V$ está conectado con la arista $e_j \in E$. En el poset de este juego vemos claramente que al eliminar un vértice, eliminamos también todas las aristas conectadas con el vértice eliminado. En la Figura 3.6 vemos el diagrama de Hasse del poset para el Chomp sobre el grafo de la Figura 3.3.

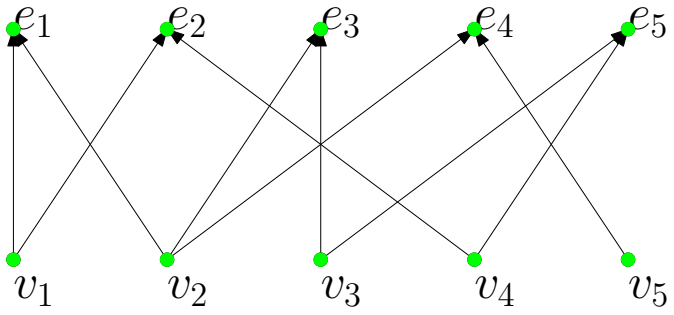

Figura 3.6. Diagrama de Hasse del poset del Chomp sobre el grafo de la Figura 3.3.

En general no se conoce una descripción de una estrategia general que funcione para todos los poset games. De hecho, en el caso particular de Chomp en grafos no se conoce una estrategia ganadora cuando el grafo en cuestión no es bipartito.

3.3.3. Poset finitos con un máximo

Como caso particular y curioso dentro de los poset games, están los poset finitos con un máximo y reglas tanto normales como misère.

Definición 3.12 *Dado un poset $(P, \leq)$, un máximo es un elemento $M \in P$ tal que para todo $x \in P$ se verifica que $x \leq M$.*

Cuando $(P, \leq)$ un máximo, veamos que siempre gana A.

Teorema 3.13 *Si $(P, \leq)$ es un poset finito con un máximo M y al menos dos vértices, entonces el jugador A tiene una estrategia ganadora para el poset game en P.*

Demostración
Supongamos por reducción al absurdo que A no tiene una estrategia ganadora, entonces es B quien la tiene. Por tanto, sea cual sea el movimiento de A, lo que queda no es una posición ganadora. Digamos que A juega el máximo M, siendo el poset restante $P_M = P - \{M\} = P'$. Ahora, como B tiene una estrategia ganadora, existe un movimiento ganador, digamos que jugar $x \in P'$. Entonces, $P'_x = P' - \{y \in P' \ / \ y \geq x\}$ es una posición segura. No obstante, se observa que que $P'_x = P_x$ porque $P' = P - \{M\}$ y M es el máximo, por tanto, $M \geq x$. Como consecuencia, se tiene que si el jugador A hubiese jugado x, estaría en una posición segura y, por tanto, sería A quien tiene una estrategia ganadora, lo cual es absurdo.

Esta demostración no es constructiva y usa la técnica del robo de estrategia (*strategy stealing* en inglés). Si bien la prueba demuestra que A tiene una estrategia ganadora, no dice cuál es ni cómo llegar a ella.

El juego de la tableta de chocolate

Un ejemplo famoso de poset game finito con un máximo y reglas misère es el llamado juego de la tableta de chocolate. Este juego se desarrolla en una tableta de chocolate rectangular con a cuadrados de alto y b cuadrados de ancho, los jugadores se suceden comiendo en la tableta y pierde el jugador que se como el último cuadrado (usualemente se dice que este cuadrado está envenenado). Cuando un jugador "muerde" un cuadradito, se come todos los cuadraditos que están en filas superiores y en la misma columna o en columnas a la derecha del cuadradito mordido.

Sea a el número de filas de la tableta de chocolate y b el número de columnas de esta, el poset de este juego es el siguiente:

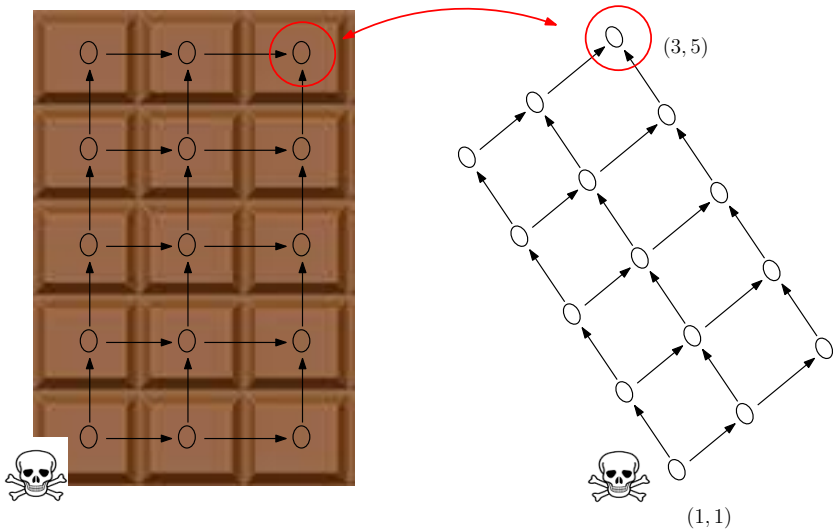

Figura 3.7. Representación gráfica del juego de la tableta de chocolate

$$P = \{(x, y) \ / \ 1 \leq x \leq a, 1 \leq y \leq b\} \ ; \ (x, y) \leq (x', y') \Leftrightarrow x \leq x' \ , \ y \leq y'$$

Se puede observar que $(1, 1)$ es el mínimo y (a, b) el máximo y por el Teorema 3.13, el jugador que empieza tiene una estrategia ganadora. No es dificíl descubrir esta estrategia cuando $b = 2$ o cuando $a = b$; no obstante, este problema está abierto en general. En particular, no se conoce una descripción del movimiento ganador cuando $b = 3$ y $a \in \mathbb{N}$.

Como curiosidad, la estrategia ganadora para el caso en el que $a = b$ consiste en comer el cuadradito $(2, 2)$, dejando así la tableta en forma de "L" simétrica (en cada lado hay el mismo número de cuadrados) y, posteriormente, se basará en "imitar" los movimientos de B; es decir, si B elimina el cuadradito $(x, 1)$, A debe eliminar el cuadradito $(1, x)$. Por su parte, para el caso en el que $b = 2$, al principio A debe comer el máximo del poset y, posteriormente, ir dejando una fila con un cuadradito más que la otra.

4

Suma de juegos combinatorios

Con varios juegos combinatorios con reglas normales se puede formar un nuevo juego combinatorio al que llamaremos suma de juegos. Dadas posiciones iniciales en cada uno de los juegos, los jugadores alternan movimientos, los cuales consisten en seleccionar cualquiera de los juegos y hacer un movimiento legal solo en el juego escogido. El juego continúa hasta que todos los juegos alcanzan una posición terminal. Gana el jugador que hace el último movimiento.

Cuando todos los juegos involucrados son finitos, el denominado Teorema de Sprague-Grundy da una bonita solución al problema de determinar qué jugador tiene una estrategia ganadora. Esta solución se basa en el valor de la función Sprague-Grundy de cada uno de los juegos involucrados y está profundamente inspirada en la solución del juego de NIM que presentamos en Teorema 3.4.

4.1. Grafo de estados para la suma de juegos

Supongamos que tenemos n juegos combinatorios finitos con reglas normales y que llamamos $J_1, \ldots, J_n$. Sea $G_i = (X_i, A_i)$ el grafo de estados de J_i para todo $i \in \{1, \ldots, n\}$. Sea $J = J_1 + \cdots + J_n$ el juego suma, vamos a describir el grafo $G = (X, A)$ de J, que denotaremos por $G = G_1 + \cdots + G_n$. El conjunto X de vértices es el producto cartesiano $X = X_1 \times \ldots \times X_n$. Para los vértices $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n) \in X$, se tiene que $(x, y) \in A$ si y solo si existe un $k \in \{1, \ldots, n\}$ tal que $(x_k, y_k) \in A_k$ $x_i = y_i$ para todo $i \neq k$.

Si y_i es la posición inicial del juego J_i, entonces $(y_1, \ldots, y_n)$ es la posición inicial de la suma de juegos. De igual forma, si $F_1, \ldots, F_n$ son los estados finales de cada uno de los juegos, el conjunto de estados finales del juego suma es el producto cartesiano de estos, es decir, $F_1 \times \ldots \times F_n$.

Dado que el grafo de estados G_i de cada juego combinatorio es finito, entonces el grafo de estados de la suma G también es finito. El número máximo de movimientos desde un estado $x = (x_1, ..., x_n)$ es la suma del máximo número de movimientos en cada uno de los juegos.

4.2. El teorema de Sprage-Grundy

Sean J_1 y J_2 dos juegos combinatorios finitos y sea $J = J_1 + J_2$ la suma de estos dos juegos. La siguiente tabla resume quién tiene estrategia ganadora en J

$J_1 \setminus J_2$	gana A	gana B
gana A	depende	gana A
gana B	gana A	gana B

Se puede dar una prueba directa de lo expuesto en esta tabla, no obstante, en esta memoria obtendremos esto como consecuencia del Teorema 4.1.

Como se puede observar en la tabla, cuando A gana tanto en J_1 como en J_2, pueden darse las dos opciones para $J = J_1 + J_2$. El Teorema de Sprague-Grundy da una solución elegante a determinar quién gana en este caso. Este resultado se basa en el estudio de la función de Sprague-Grundy que recordemos que no solo determina qué jugador gana el juego, si no que da una información más precisa: vale 0 si gana B o un valor diferente de 0 si gana A.

Ahora ya podemos enunciar y demostrar el Teorema de Sprague-Grundy.

Teorema 4.1 (Teorema de Sprague-Grungy) *Sean J_1 y J_2 dos juegos combinatorios con reglas normales y x_1 y x_2 los respectivos estados de los juegos J_1 y J_2. Entonces, para la suma de juegos: $sg(x_1, x_2) = sg(x_1) \oplus sg(x_2)$*

Demostración
Sean $sg(x_1) = a$, $sg(x_2) = b$ y $sg(x_1) \oplus sg(x_2) = a \oplus b = c$, queremos ver que $sg(x_1, x_2) = c$. Para ello, utilizaremos el método de inducción y probaremos las dos propiedades siguientes:

1. $\forall d < c$, $\exists x'$ seguidor de $x = (x_1, x_2)$ tal que $sg(x') = d$
2. No existe ningún seguidor x' de x tal que $sg(x') = c$

Veamos que la propiedad se cumple para $x_0 = (x_{10}, x_{20})$ una posición final de $J_1 + J_2$ y, por tanto, siendo x_{10} y x_{20} las posiciones terminales de los juegos 1 y 2 respectivamente. Obviamente, $sg(x_{10}) = sg(x_{20}) = 0$. Además, como en ambos juegos ya no hay movimientos posibles y nos encontramos, por tanto, en una posición terminal de $J_1 + J_2$, entonces $sg(x_{10}, x_{20}) = sg(x_{10}) \oplus sg(x_{20}) = 0$.

Ahora, supongamos cierta la igualdad para cualquier $x' = (x'_1, x'_2)$ seguidor de $x = (x_1, x_2) \in X$ y veamos que se cumple para (x_1, x_2); es decir, veamos que x satisface las propiedades 1 y 2.

(1) Sea $e = c \oplus d$ y sea k el número de dígitos de la expresión binaria de e; esto es, $2^{k-1} \leq 2^k$, entonces e tiene un 1 en la posición k (desde la derecha). Ya que $d < c$, c tiene un 1 en la posición k y d tiene un 0 ahí. Puesto que $c = sg(x_1) \oplus sg(x_2)$, existe un x_i tal que su expresión binaria tiene un 1 en la posición k. Sin pérdida de generalidad, supongamos que $i = 1$. Entonces $e \oplus sg(x_1) < sg(x_1)$. Por lo tanto, existe un movimiento de x_1 a algún x'_1, con $sg(x'_1) = e \oplus sg(x_1)$. Entonces, el movimiento de $x = (x_1, x_2)$ a $x' = (x'_1, x_2)$ es legal y utilizando la hipótesis de inducción tenemos:
$sg(x') = sg(x'_1, x_2) = sg(x'_1) \oplus sg(x_2) = e \oplus sg(x_1) \oplus sg(x_2) = e \oplus c = d$

(2) Supongamos por reducción al absurdo que existe un x' tal que $sg(x') = c$. Sin pérdida de generalidad, supongamos que este sucesor afecta al primer juego; esto es, $x' = (x'_1, x_2)$ siendo x'_1 sucesor de x_1. Ahora, utilizando la hipótesis de inducción, $c = sg(x'_1, x_2) = sg(x'_1) \oplus sg(x_2) = sg(x_1) \oplus sg(x_2)$, entonces $sg(x'_1) = sg(x_1)$. Esto es absurdo, ya que x_1 no puede tener el mismo valor sg que un sucesor.

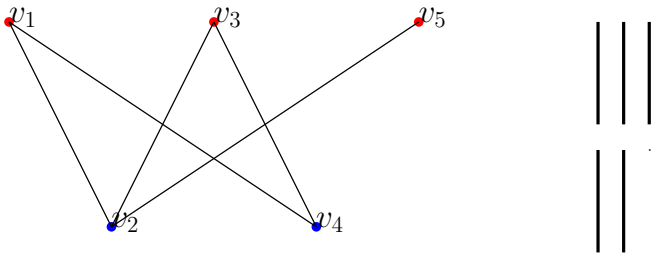

Figura 4.1. Suma de juegos del Chomp de la Figura 3.3 y un NIM(3,2).

Ahora veremos un ejemplo de suma de juegos combinatorios, determinaremos qué jugador tiene la estrategia ganadora y diremos cuáles son los movimientos ganadores. Consideremos la suma de juegos de la Figura 4.1. Lo primero que tenemos que hacer es calcular el valor de la función Sprague-Grundy para ambos juegos por separado. Para el NIM, se demostró en el Teorema 3.4 que el valor sg es la suma NIM del número de palillos en cada una de las filas; por tanto, el valor sg del NIM$(3, 2)$ es $sg(3, 2) = 3 \oplus 2 = 1$. Por su parte, en el otro juego tenemos

un Chomp con 5 vértices y 5 aristas, es decir, con un número impar de vértices y de aristas, estado para el cuál se demostró en el Teorema 3.11 que tiene valor sg igual 3. Ahora que ya tenemos el valor de la función de Sprague-Grundy de ambos juegos, para obtener el valor sg de la suma de juegos, por el Teorema 4.1, hacemos la suma NIM de los valores anteriores:

$$\begin{array}{r} 1\,1 \to 3 \\ \underline{0\,1 \to 1} \\ 1\,0 \to 2 \end{array}$$

Vemos que la suma de los juegos mencionados tiene valor sg distinto de cero. Por lo tanto, el jugador A tiene la estrategia ganadora, que consistirá en hacer un movimiento que deje el valor sg de la suma igual a 0. Para ello, al igual que hacíamos para buscar la estrategia ganadora del NIM en general, localizamos el primer uno en el resultado final de izquierda a derecha. Este se corresponde a la primera columna, en la que hay un uno, procedente del valor sg del Chomp, y un cero, procedente del valor sg del NIM. Por lo tanto, para obtener un cero en el resultado final, debemos transformar el uno de esa columna en un cero, lo cual supone cambiar el valor sg del Chomp a 1, que, por el Teorema 3.11, sabemos que esto se corresponde con una situación en la que el número de vértices es impar y el número de aristas es par. Como estamos en una situación en la que el número de vértices y de aristas son impares, la estrategia ganadora es eliminar una arista cualquiera. Luego, los movimientos ganadores para el jugador A en la suma del NIM$(3, 2)$ y el Chomp de la Figura 3.3 deben efectuarse en el juego del Chomp y son: eliminar la arista e_1, eliminar la arista e_2 ... y eliminar la arista e_5.

Ahora vamos a ver que el resultado anterior se generaliza de forma natural para la suma n juegos combinatorios. En efecto, sean $J_1, J_2, ..., J_n$ n juegos combinatorios y $x_1, x_2, ..., x_n$ las respectivas situaciones (vértices del grafo de estados) en cada uno de los juegos. Entonces, la función sg de la suma de juegos $J_1 + ... + J_n$ es:

$$sg(x) = sg(x_1, x_2, ..., x_n) = sg(x_1) \oplus sg(x_2) \oplus ... \oplus sg(x_n)$$

Este resultado se obtiene directamente por inducción; en efecto, en el Teorema 4.1 estudiamos el problema para $n = 2$ y, para valores mayores, basta con observar que el juego $J_1 + \cdots + J_n$ es el mismo que $((J_1 + J_2) + J_3) + \cdots + J_n$.

5

Complejidad

Los juegos proporcionan muchos ejemplos de problemas que, aunque en teoría son resolubles, parecen estar más allá de los límites de la computación práctica. Por ejemplo, determinar si un jugador está ganando una partida de ajedrez y qué estrategia debe seguir para acabar ganando, es un problema que aunque a priori puede llegar a ser resuelto por un ordenador, llevaría una cantidad de tiempo astronómica. En términos de complejidad computacional, uno sospecha que los juegos combinatorios son intrínsecamente difíciles y se desea obtener resultados teóricos que confirmen esta sospecha. Comparado a la abundancia de juegos que parecen ser difíciles, los resultados en esta dirección hasta ahora son bastante escasos.

5.1. Clases de complejidad

Definición 5.1 *Una clase de complejidad es el conjunto de los problemas de decisión (aquellos donde la respuesta es sí ó no) que pueden ser resueltos por una máquina M utilizando una cota superior asintótica O(f(n)) del recurso R, donde n es el tamaño de la entrada. El recurso R suele ser generalmente el tiempo de computación o el tamaño de memoria necesario.*

5.1.1. Clasificación de las clases de complejidad

No es un objetivo de este trabajo el de adentrarse formalmente en la Teoría de la Complejidad, sino introducir las clases de complejidad que nos harán falta en el desarrollo de este capítulo. Para un estudio en profundidad de la Teoría de Complejidad por medio del concepto de máquina de Turing, ver por ejemplo [7].

En nuestro contexto de teoría combinatoria de juegos, vamos a tratar el problema de decisión siguiente: dado un juego combinatorio finito ¿tiene el jugador A una estrategia ganadora? Y la entrada serán las especificaciones del juego.

Considerando la Teoría de la Complejidad Computacional, las clases de los problemas de decisión que vamos a trabajar en esta memoria son:

1. P. Contiene aquellos problemas de decisión que se pueden resolver en tiempo polinómico en el tamaño de la entrada del algoritmo. Los problemas de complejidad polinómica son tratables, es decir, en la práctica se pueden resolver en un tiempo razonable y suelen denominarse problemas *fáciles*.
2. PSPACE. Contiene aquellos problemas de decisión que pueden ser resueltos haciendo uso de una cantidad polinomial de memoria y en un tiempo ilimitado.

Claramente P $\subseteq$ PSPACE, ya que en un tiempo polinomial no se puede usar más que una cantidad polinomial de memoria. Si bien estas dos clases parecen bien diferentes, sigue siendo un problema abierto el determinar si estas dos clases son iguales o no. En el siguiente esquema (ver Figura 5.1) se presentan estas y otras clases de complejidad ordenadas por contención; los interrogantes en la figura indican que no se ha demostrado que las clases sean distintas ni iguales. No obstante, se sabe que son estrictos NL $\subsetneq$ PSPACE $\subsetneq$ EXPSPACE.

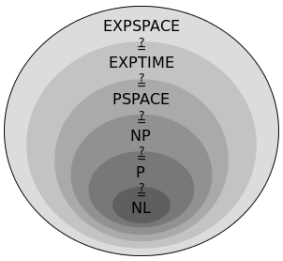

Figura 5.1. Principales clases de complejidad ordenadas por contención

Dentro de cada clase de complejidad existe una serie de problemas que los son más difíciles de la clase. A estos problemas se les llama *completos*. Por ejemplo, a los problemas PSPACE más difícles de la clase se les llama PSPACE-*completos*.

5.2. Complejidad de juegos combinatorios

En el Teorema 3.3 vimos cómo se puede decidir quién gana NIM$(a_1, \ldots, a_n)$, haciendo la suma NIM de $a_1, \ldots, a_n$. Es fácil comprobar cómo esta suma se puede hacer en tiempo polinomial. Como consecuencia se tiene que:

Proposición 5.2. *El problema de decisión de determinar qué jugador tiene una estrategia ganadora en* NIM$(a_1, \ldots, a_n)$ *está en la clase de complejidad P*

También vimos en el Teorema 3.10 cómo se puede decidir qué jugador gana en el juego de Chomp con un grafo bipartito solo mirando la paridad del número de vértices y de aristas del grafo en cuestión.

Proposición 5.3. *El problema de decisión de determinar qué jugador tiene una estrategia ganadora en el juego de Chomp en un grafo bipartito está en la clase de complejidad* P

En esta sección veremos cómo todos los juegos expuestos (y alguno más) están en la clase más general PSPACE y, en particular, que hay algunos que son PSPACE-completos. Los resultados a continuación están extraídos de [9].

5.2.1. Juego de las fórmulas booleanas

Definición 5.4 *En lógica booleana, una fórmula está en forma normal conjuntiva (*FNC*) si corresponde a una conjunción de cláusulas, donde una cláusula es una disyunción de literales.*

La entrada de este juego es un par (F, ξ), donde F es una fórmula FNC y $\xi = (\xi_1, \xi_2, ..., \xi_n)$ es una lista de variables, incluyendo todas aquellas que aparecen en F. El movimiento i-ésimo consiste en asignar a ξ_i el valor 1 (verdadero) ó 0 (falso). Tras n movimientos, el jugador A (el que empieza) gana si las asignaciones de las variables han hecho que F sea verdadero; B gana en caso contrario. Por conveniencia se suelen escribir las entradas de este juego como $(\exists\xi_1)(\forall\xi_2)(\exists\xi_3)...(Q\xi_n)F$, donde los cuantificadores son alternativamente $\exists$ y $\forall$.

Ejemplo 5.5 *Un ejemplo de una entrada de este juego es* $(\exists x_1)(\forall x_2)(\exists x_3)((x_1 \wedge (x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3))$. *No es difícil ver en este ejemplo que A puede ganar: él tiene una estrategia ganadora que consiste en dar a* x_1 *el valor 1 en el movimiento 1 y a* x_3 *el mismo valor que tiene* x_2 *en el movimiento 3.*

El problema de decidir si una fórmula booleana en FNC es cierta es el problema PSPACE completo por antonomasia (no lo demostraremos aquí), así que como consecuencia se tiene que su versión juego combinatorio también lo es.

Teorema 5.6 *El juego de las fórmulas booleanas en* FNC *es* PSPACE*-completo.*

Si bien este juego está diseñado para ser PSPACE-completo y parece poco natural como juego, veremos que hay otros juegos combinatorios que también son PSPACE-completos.

5.2.2. NODE KAYLES

La entrada del juego es un grafo no dirigido. Un movimiento consiste en eliminar un vértice del grafo (que no haya sido eliminado previamente) y sus vértices adyacentes. El primer jugador que se quede sin movimientos, pierde.

El resultado principal de este capítulo es que determinar qué jugador posee una estrategia ganadora en el juego NODE KAYLES es un problema PSPACE-completo. Para ello, primero veremos que está en la clase de complejidad PSPACE. Es más, probaremos que todos los juegos combinatorios mencionados en esta memoria: NIM, Chomp (en grafos bipartitos o no), poset games, el juego de las fórmulas booleanas y NODE KAYLES; están en la clase PSPACE. Este es un hecho, quizás, sorprendente, porque como hemos visto en cada caso, el grafo de estados del juego tiene un tamaño exponencial. Por tanto, la estrategia general de buscar un núcleo expuesta en el Capítulo 1, necesita un tamaño en memoria exponencial. En la siguiente demostración veremos cómo caracterizar si un vértice del grafo de estados está en el núcleo sin almacenar en memoria todo este grafo.

Teorema 5.7 *Sea J un juego combinatorio cualquiera de los mencionados en esta memoria. El problema de decidir qué jugador tiene una estrategia ganadora en J está en* PSPACE.

Demostración
Sea J uno de esos juegos y sea $G = (V, A)$ su grafo de estados, siendo $x_0 \in V$ el estado inicial. Sabemos que (ver Teorema 2.4) A gana si, y solo si, $x_0 \notin K$ siendo K el único núcleo de G si J es un juego con reglas normales, o el único núcleo del grafo inducido por el conjunto de vértices no terminales si J es un juego con reglas misère. Veamos que dado $y \in V$ podemos determinar usando un espacio polinomial en el tamaño de la entrada del juego si $y \in K$ o no.

- Si $\text{suc}(y) = \emptyset$, entonces $y \in K$.
- Si $\text{suc}(y) = \{z_1, z_2, \ldots, z_k\}$, entonces, $y \in K \Leftrightarrow z_i \notin K,\ \forall i \in \{1, 2, \ldots, k\}$. Por lo tanto, se puede implementar una función recursiva que verifique si cualquier vértice del grafo de estados está en el núcleo. En efecto, en cada momento solo hay que guardar en la memoria la sucesión de vértices del grafo que se ha visitado hasta llegar a uno dado. Este valor está acotado por el máximo número de movimientos en la partida, que es a lo sumo $n+m$ para el Chomp en grafos, el número de vértices para poset games, el número de variables para el juego de las fórmulas y el número de vértices del grafo para NODE KAYLES. En todos los casos, un número polinomial en el tamaño de entrada.

Ahora ya podemos demostrar el resultado más importante del capítulo:

Teorema 5.8 *Determinar qué jugador tiene una estrategia ganadora en* NODE KAYLES *es un problema* PSPACE*-completo.*

Demostración
Por el Teorema 5.7, NODE KAYLES está en PSPACE. Veamos, pues, que NODE

KAYLES es PSPACE-completo. Para ello, veremos que NODE KAYLES es *más dificil* que el juego de la fórmulas booleanas. Es decir, que si conocemos una estrategia ganadora para el juego NODE KAYLES de cualquier grafo, entonces también sabríamos ganar al juego de las fórmulas booleanas. Sea, por tanto, F una fórmula de entrada al juego de las fórmulas booleanas. Asumimos sin pérdida de generalidad que $F = (\exists\xi_n)(\forall\xi_{n-1})...(\exists\xi_1)(B_1 \wedge ...B_n)$ donde cada B_i es una disyunción de literales, n es impar y $B_1 = (x_1 \vee \overline{x_1})$. Esta última suposición es asumible pues al añadirlo a la fórmula no afecta al resultado final y además, nos va a servir como "comodín" para que n sea impar, hipótesis fundamental en el razonamiento posterior. Definimos el grafo $G = (V, E)$ como sigue $V = \cup_{i=0}^{n} X_i$, siendo

- $X_0 = \{x_{0,k} : 1 \leq k \leq m\}$
- $X_i = \{x_i, \overline{x_i}\} \cup \{y_{ij} : 0 \leq j \leq i-1\},\ i = 1, 2, \ldots, n$

$E = \cup_{0\leq i\leq n}[X_i]^2 \cup D \cup D' \cup (\cup_{0\leq j<i\leq n} C_{ij})$, donde

- $[X_i]^2 = \{\{x, y\} : x, y \in X_i, x \neq y\}$
- $D = \{\{x_i, x_{0,k}\} : x_i \textit{ aparece en } B_k\} \cup \{\{\overline{x_i}, x_{0,k}\} : \overline{x_i} \textit{ aparece en } B_k\}$
- $C_{ij} = \{\{y_{ij}, w\} : w \in \cup_{0\leq k\leq i, k\neq j} X_k\},\ i = 1, 2, \ldots n,\ j = 0, 1, \ldots, i-1$

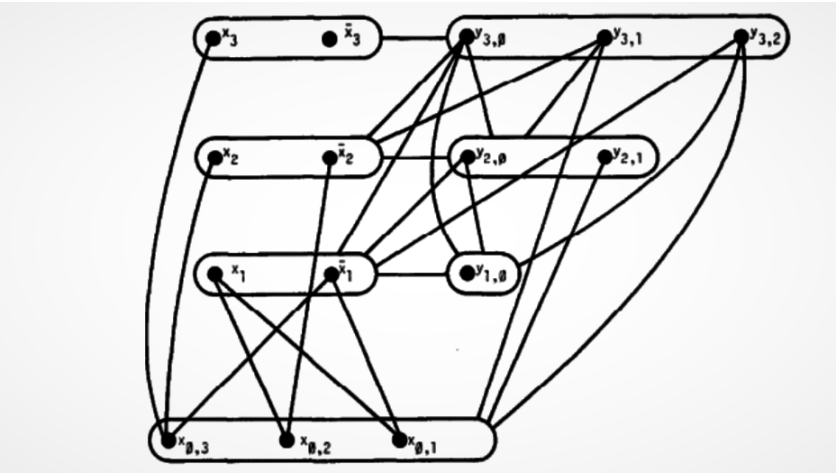

Figura 5.2. Grafo correspondiente al juego de las fórmulas FNC del Ejemplo 5.5.

Diremos que el juego NODE KAYLES es jugado legalmente si para $i = 1, 2, .., n$, el vértice jugado en el movimiento i es x_{n-i+1} ó $\overline{x_{n-i+1}}$. El grafo G que hemos construido obliga a los jugadores a jugar legalmente: si en alguno de los n primeros movimientos un jugador no juega legalmente, entonces dicho jugador perderá inmediatamente. Para ver esto, fijamos un entero i, $1 \leq i \leq n$, y suponemos que los $n - i$ primeros movimientos han sido legales. Así los nodos jugados son uno de cada par $x_n, \overline{x_n}, ..., x_{i+1}, \overline{x_{i+1}}$. Dentro de cada conjunto X_k, cualquier

par de vértices son adyacentes; por tanto, como mucho podemos jugar un vértice dentro de cada conjunto. Así todos los vértices de $X_n \cup X_{n-1} \cup ... \cup X_{i+1}$ ya no están disponibles para jugar. Supongamos que uno de los jugadores en $n - i + 1$ juega ilegalmente, esto es, juega algún vértice de $X_i \cup X_{i-1} \cup ... \cup X_1 \cup X_0$ distinto de x_i ó $\overline{x_i}$. Si juega algún vértice en X_k para $k < i$, entonces su oponente puede ganar jugando $y_{i,k}$, esto deja la partida sin vértices, puesto que todo vértice de X_k es adyacente al nodo jugado ilegalmente y cualquier nodo de $(X_0 \cup X_1 \cup ...X_i) - X_k$ es adyacente a y_{ik}. Por otra parte, si el movimiento ilegal es en X_i, debe ser en un y_{ik}, con $k < i$. En este caso, el oponente puede ganar jugando x_k si $k > 0$ ó $x_{0,1}$ si $k = 0$ (el vértice x_{01} está disponible por la definición de B_1). Nuevamente, la respuesta deja la partida sin vértices.
Completamos la prueba viendo que si A tiene la estrategia ganadora en el juego de las fórmulas FNC, entonces puede ganar en NODE KAYLES sobre G. Supongamos que A tiene una estrategia ganadora para el juego de las fórmulas con entrada F. Entonces, el jugador A puede ganar NODE KAYLES en G como sigue. En los primeros n movimientos del juego, A juega legalmente mientras B lo haga, y usa su estrategia ganadora por la correspondencia a jugar x_i a 0 ó 1 en el juego de las fórmulas correspondiente a jugar x_i ó $\overline{x_i}$ en NODE KAYLES sobre G, respectivamente. Si B juega ilegalmente alguno de esos n movimientos, A claramente gana por lo expuesto anteriormente. Si B juega siempre legalmente, tras n movimientos no hay vértices disponibles: ningún vértice de $X_1 \cup ... \cup X_n$ es jugable, puesto que un vértice de X_i ha sido elegido para $i > 0$; y dado que A tiene la estrategia ganadora, está claro que todo vértice $x_{0,k}$ de X_0 es adyacente a algún vértice jugado, es decir, todos los vértice $x_{0,k}$ han sido eliminados (si algún vértice $x_{0,k}$ no hubiera sido eliminado, entonces la disyunción B_k sería falsa, es decir, igual a 0; por lo tanto, el resultado final sería falso y eso contradice la hipótesis que afirma que A tiene la estrategia ganadora). Como en el movimiento n-ésimo se elimina el último vértice del grafo (puesto que n es impar, será el jugador A quien juegue en X_1 y, por lo tanto, quien gane.

Se razona de manera similar si el jugador B tiene la estrategia ganadora en el juego de las fórmulas para ver que entonces puede ganar en NODE KAYLES.

Como consecuencia, tenemos que hay juegos combinatorios en los que el problema computacional de decidir qué jugador tiene una estrategia ganadora es PSPACE-completo. Por ello, se cree que no habrá un algoritmo polinomial que determine qué jugador gana en este juego. Haciendo uso de que NODE KAYLES es PSPACE-completo, en [4] se demuestra que los poset games son también PSPACE-completos, no incluimos aquí esta demostración por falta de espacio.

A

Implementación del Chomp en C

Siguiendo los resultados y métodos expuestos en esta memoria, se ha implementado en el lenguaje de programación C un programa que determina qué jugador tiene una estrategia ganadora para el juego de Chomp en un grafo. En esta sección vamos a describir brevemente la implementación realizada.

El algoritmo implementado recibe como parámetro de entrada la matriz de adyacencia del grafo con el que vamos a jugar, procedente de un fichero que el programa debe leer.

Definición A.1 *La matriz de adyacencia de un grafo no dirigido $G = (V, E)$ con $V = \{v_1, \ldots, v_n\}$ es una matriz cuadrada de orden n tal que el elemento (i, j) de la matriz vale 1 si $\{v_i, v_j\} \in E$ y 0 en caso contrario.*

El algoritmo extrae todos los datos del grafo (vértices, aristas, grados de los vértices, ...) y con ellos comprueba si el grafo es bipartito o no. Si fuera bipartito, usa el Teorema 3.9 para decir qué jugador dispone de una estrategia ganadora. Por el contrario, si no fuera bipartito, usa el algoritmo recursivo que se esbozó en la demostración del Teorema 5.7. En cualquiera de los dos casos, cuando un jugador tiene al menos una estrategia ganadora, el programa también lista todos los primeros movimientos ganadores. El código se encuentra completo en https://bit.ly/2KWDBS0.

Cuando el grafo de entrada es bipartito, el algoritmo devuelve de forma inmediata qué jugador tiene una estrategia ganadora. No obstante, como cabe esperar, cuando el grafo no es bipartito, el tiempo de ejecución es mayor.

Así por ejemplo, para el grafo de Petersen (ver Figura A.1), el programa recibe como entrada el siguiente fichero:

```
10 10
0 1 0 0 1 1 0 0 0 0
1 0 1 0 0 0 1 0 0 0
0 1 0 1 0 0 0 1 0 0
0 0 1 0 1 0 0 0 1 0
1 0 0 1 0 0 0 0 0 1
1 0 0 0 0 0 0 1 1 0
0 1 0 0 0 0 0 0 1 1
0 0 1 0 0 1 0 0 0 1
0 0 0 1 0 1 1 0 0 0
0 0 0 0 1 0 1 1 0 0
```

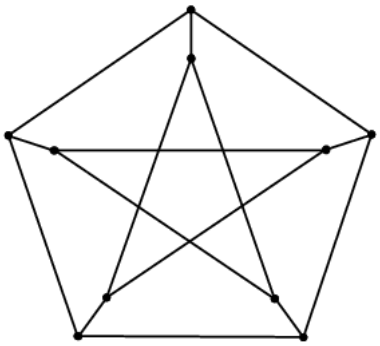

Figura A.1. Ejemplo de grafo no bipartito: grafo de Petersen.

Para el Chomp sobre este grafo, la estrategia ganadora la tiene jugador A y dispone de un total de quince movimientos ganadores en su primer turno, que consisten en quitar cualquiera de las quince aristas que tiene el grafo. El programa tarda 216 segundos en determinar el primer movimiento ganador. En total, tarda unos 40 minutos en dar todos los movimientos ganadores.

Bibliografía

[1] Bondy, J.A. & Murty, U.S.R. *Graph Theory* Graduate Text in Mathematics, Springer (2008)

[2] Boros, E. & Gurvich, V.A. *Perfect graphs, kernels and cores of coperative games* Discrete Mathematics (2006)

[3] Bouton, C.L., *NIM, a game with a mathematical theory*, The Annals of Mathematics, 2nd Ser., Vol. 3, No. 1/4. (1901-1902)

[4] Grier, D. *Deciding the winner of an arbitrary finite poset game is PSPACE-complete*. Automata, languages, and programming. Part I, pp. 497-503

[5] Grundy, P.M. *Mathematics and games* Eureka Vol. 2(1939) pp. 6-8

[6] Khadhawiw, T. & Ye, L. *Chomp on Graphs and Subsets* arXiv: 1101.2718 [math.CO]

[7] Knuth, D. M. *The art of computer programming*. Vols. 1-4, Reading, Massachusetts: Addison-Wesley

[8] Richardson, M. *Solutions of irreflexible relations* Ann. Math. 58 (1953) pp. 573-590

[9] Schaefer, T.J. *On the complexity of some two-person perfect-information games* Journal of computer and system sciences 16 (1978), no. 2, pp. 185-225

[10] Sprague, R.P. *Über Mathische Kampfspiele* Tohoku Mathematical Journal Vol. 41 (1935-1936) pp. 438-444

[11] Zermelo, E. *Über eine Anwendung der Mengenlehre auf die Theorie des Schachspiels* Cambridge (1913) pp. 501-504

Combinatorial Game Theory

Sección de Matemáticas
Universidad de La Laguna

Ulises Pérez Romero
Facultad de Ciencias · Sección de Matemáticas
Universidad de La Laguna
alu0101063683@ull.edu.es

Abstract

Combinatorial games are two-players games (Alice and Bob) with perfect information, without random movements and in which there are no ties. Some examples of combinatorial game are NIM, Chomp in graphs and poset games. These games can be modeled by a graph: the states graph of a game. In these graph there is a distinguished set of vertices, called kernel. We will see that kernel contains the information about the winning strategy of the game. In this poster we will use the game "Chomp" as an example to illustrate the results.

1. Chomp in graphs

CHOMP is a two-player game played on an undirected graph G. Each of them, in its corresponding turn, must remove either one edge, or one vertex together with all its adjacent edges. The ultimate goal of the game is to be the player who removes the last vertex of the graph.

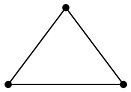

Figure 1: *Triangle graph.*

2. State graph

THE STATE GRAPH OF A GAME is a directed graph where each vertex is a position or state of the game and two states E_1 and E_2 are connected by an arc if from position E_1 a player can move to E_2 position in one movement.

Let $G = (V, E)$ an undirected graph, the state graph of Chomp on G is $G_e = (V_e, A_e)$, where $V_e = \{G' \text{ / } G' \text{ is a subgraph of } G\}$ and $A_e = \{(G'_1, G'_2) \text{ / we can go from } G'_1 \text{ to } G'_2 \text{ removing a vertex or an edge}\}$.

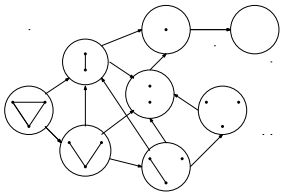

Figure 2: *States graph of Chomp on the triangle graph*

3. Kernel of a graph

Definition 1 *A kernel of a directed graph $G = (V, A)$ is a subset $K \subset V$ which is independent and absorbent.*

- *$K \subset V$ is independent if $\forall u \in K$, $\nexists v \in K$ such that $(u, v) \in A$.*
- *$K \subset V$ is absorbent if $\forall u \notin K$, $\exists v \in K$ such that $(u, v) \in A$.*

Theorem 1 *The state graph of every combinatorial game has only one kernel.*

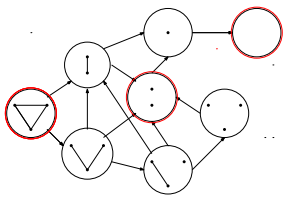

Figure 3: *Kernel (in red color) of the graph of Figure 2*

4. Winning strategy

Theorem 2 *Let $G = (V, A)$ be the state graph of a combinatorial game, $x_0 \in V$ the starting position of the game and we consider K the unique kernel of G. Then, Alice has a winning strategy $\Leftrightarrow x_0 \notin K$*

Corollary 1 (Zermelo's Theorem) *In every finite combinatorial game one of the players has a winning strategy.*

In Chomp on the triangle graph, the initial state of the game is in K, so Bob has a winning strategy. Indeed, whatever Alice moves, she will move to a state out of the kernel (because the kernel it is independent) and Bob can reply with a move to a state in the kernel (because it is absorbent). This behaviour eventually implies Bob's victory.

When the graph G where we play Chomp is bipartite there is a nice description of the kernel which leads to the following result

Theorem 3 *Let G be a bipartite graph. Bob has a winning strategy playing Chomp on G if and only if both the number of edges and vertices of G are even.*

5. Implementation of Chomp

When G is not bipartite, Theorem 3 does not hold (the example in Figure 3 shows). Nevertheless, we have the following:

Theorem 4 *Alice has a winning strategy in Chomp on G if and only if $\exists x \in V \cup E$ such that Bob has a winning strategy in Chomp on $G - \{x\}$.*

This result suggests a recursive algorithm for determining which player has a winning strategy in Chomp on a given graph.

Corollary 2 *Given a graph G, determining which player has a winning strategy in Chomp on G is PSPACE problem.*

We have implemented a C language program to simulate Chomp game and obtain winning strategies:

- If the graph is bipartite, it uses Theorem 3.
- If the graph is not bipartite, it uses Theorem 4 and simulates all possible games.

The complete code can be found at https://bit.ly/2KWDBS0.

References

[1] BONDY, J.A. & MURTY, U.S.R. *Graph Theory* Graduate Text in Mathematics, Springer (2008)

[2] BOROS, E. & GURVICH, V.A. *Perfect graphs, kernels and cores of coperative games* Discrete Mathematics (2006)

[3] KHADHAWIW, T. & YE, L. *Chomp on Graphs and Subsets* arXiv: 1101.2718 [math.CO]