












# Teoría de Juegos

# Contenido

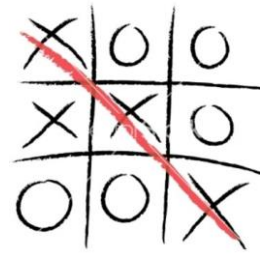
1. Conceptos básicos	
2. Posiciones Ganadoras y Perdedoras	
3. Algunas técnicas básicas	
4. Juego del Nim	
5. Suma de juegos	

# Contenido

1. Conceptos básicos	
2. Posiciones Ganadoras y Perdedoras	
3. Algunas técnicas básicas	
4. Juego del Nim	
5. Suma de juegos	

# Juego

- ❑ Actividad realizada por un conjunto de jugadores que interactúan entre sí, generalmente siguiendo determinadas reglas, y buscando lograr un objetivo (algún beneficio o ganar el juego).
- ❑ Ejemplo: Ajedrez, Damas, Monopolio, Fútbol, Póker, Battleship, Empresas competidoras, Michi, etc.



# Juegos Combinatorios

Son juegos que cumplen las siguientes propiedades:

- ☐ Son jugados por 2 jugadores que alternan movimientos
- ☐ Existe un conjunto de posibles “posiciones” o “estados” definidos (usualmente finitas)
- ☐ Hay reglas específicas que indican los posibles movimientos de cada jugador y sus objetivos
  - ☐ **Juego combinatorio imparcial:** Los jugadores tienen las mismas reglas para cualquier posición del juego. Lo único que diferencian a los jugadores son los turnos.
  - ☐ **Juego combinatorio partisano:** Hay posiciones en donde los jugadores tienen distintas reglas.
- ☐ Existe **información perfecta** (los jugadores conocen toda la información actual y pasada del juego)
- ☐ Los movimientos no dependen del **azar**

A menudo suele agregarse la siguiente propiedad:

- ☐ El juego debe terminar luego de un número finito de movimientos y no existe posibilidad de empate.

# Juego Combinatorio Partisano

- ❑ Juego Combinatorio en el cual hay una distinción entre jugadores, debido a que existen posiciones en donde los jugadores tienen distintas reglas (distintas formas de movimiento o distintos objetivos o beneficios)

Ejemplos: Ajedrez, Michi, Damas

Muchos de estos juegos pueden ser muy difíciles de resolver (como el ajedrez), es por eso que algunas propiedades que veremos solo se aplicarán a los juegos combinatorios imparciales.



# Juego Combinatorio Imparcial

- Juego combinatorio en el que los jugadores tienen las mismas reglas para cualquier posición del juego. Lo único que diferencian a los jugadores son los turnos.

Ejemplos: Subtraction Game, Nim, Chomp

## **Subtraction Game (juego de la resta):**

Hay una pila de  $n$  piedras. Dos jugadores alternan turnos. En cada turno cada jugador debe quitar  $x$  piedras, donde  $x \in S$  y  $0 < x \leq \# \text{piedras que quedan}$ . Supongamos que en este caso  $S = \{1, 2, 5\}$ . Asumiendo que ambos jugadores juegan óptimamente, ¿quién ganará?



# Juegos Combinatorios — Modos de juego






La condición de término es aquella que indica cuándo ya no es posible hacer movimientos y, por lo tanto, hace que el juego termine.

Existen 2 variantes de juegos:

- ☐ **Modo de juego normal:** Pierde el jugador que ya no puede hacer movimientos
- ☐ **Modo de juego misère:** Gana el jugador que ya no puede hacer movimientos.



# Contenido

1. Conceptos básicos	
2. Posiciones Ganadoras y Perdedoras	
3. Algunas técnicas básicas	
4. Juego del Nim	
5. Suma de juegos	

# Definición

En general, estos los juegos combinatorios imparciales se pueden representar como un DAG, debido a que no hay ciclos en el juego (si hubiera, el juego sería infinito). Cada nodo sería una posición o estado del juego y cada arista sería un posible movimiento a partir de esa posición.

Como no hay empates y ,asumiendo que los jugadores tomarán acciones racionales (es decir, tomarán decisiones óptimas tratando de ganar), podemos clasificar a cada posición como ganadora o perdedora.

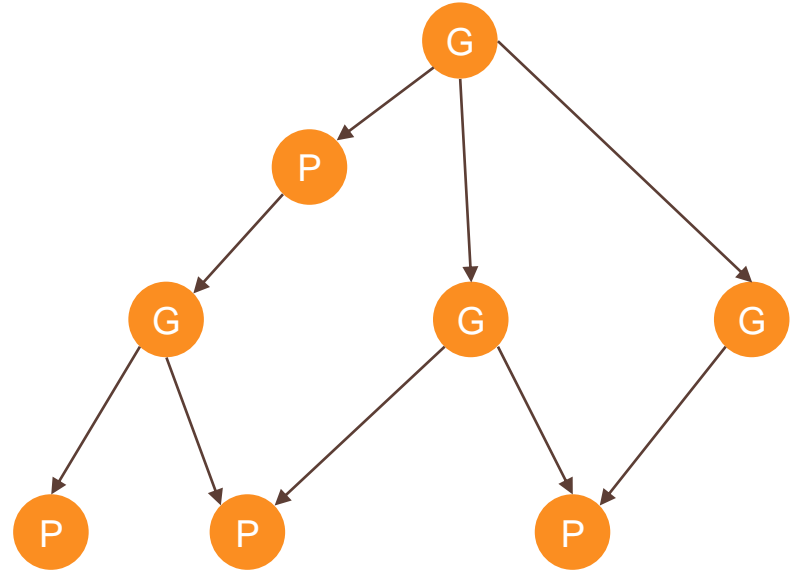
- Una posición será **perdedora** si, no importa lo que hagas en esa posición, el siguiente jugador siempre podrá ganarte mediante una secuencia de movimientos.
- Una posición será **ganadora** si existe al menos una secuencia de movimientos a partir de esa posición que te permita ganar



# Posiciones ganadoras y perdedoras

Podemos hallar recursivamente si una posición es ganadora o perdedora de la siguiente forma:

- ❑ Todas las posiciones terminales (en donde ya no se pueden mover) son perdedoras (o ganadoras si es que estás en el modo misère)
- ❑ Si estoy en una posición en donde solo me puedo mover a posiciones donde mi contrincante gana (posiciones ganadoras), entonces estoy en una posición **perdedora**
- ❑ Si estoy en una posición en donde me puedo mover al menos a una posición en donde mi contrincante pierde (posición perdedora), entonces estoy en una posición **ganadora**.



# Juego de la resta — Posiciones ganadoras y perdedoras

## Subtraction Game:

Hay una pila de  $n$  piedras. Los jugadores alternan turnos. En cada turno pueden quitar  $x$  piedras, donde  $x \in S$  y  $x \leq \# \text{piedras que quedan}$ . Supongamos que en este caso  $S = \{1, 2, 5\}$ . El que no puede hacer más movimientos pierde.

Asumiendo que ambos jugadores juegan óptimamente, ¿quién ganará?

n	0	1	2	3	4	5	6	7	8	9
G/P	P	G	G	P	G	G	P	G	G	P

Se puede demostrar por inducción que cualquier posición múltiplo de 3 es perdedora, mientras que las que no son múltiplos de 3 son ganadoras. Se puede implementar en  $O(1)$

En realidad, en este caso  $S = \{1, 2\}$  llevaría al mismo resultado. Es decir el  $x = 5$  está por gusto.

# Algoritmo Winning Losing positions

A veces es factible hallar un patrón una regla general como en el juego anterior. Pero otras veces no se puede, o es muy complicado. En esos casos una opción puede ser aplicar el algoritmo general.

El pseudocódigo sería de la siguiente forma:

```
bool isWinning(Position pos) {
    if (pos is terminal position) {
        return false;
    }
    vector<Position> moves = possible positions to which I can move;
    for (all nextPos in moves) {
        if (!isWinning(nextPos)) return true;
    }
    return false;
}
```

Nota: La posición actual podría estar compuesta de **varios parámetros**.

# Algoritmo Winning Losing positions - DP

Como el grafo es un DAG. Puede hacer el código más eficiente usando dp.

```
bool used[];
bool dp[];

bool isWinning(Position pos) {
    if (pos is terminal position) {
        return false;
    }
    if (used[pos]) {
        return dp[pos];
    }
    used[pos] = true;
    vector<Position> moves = possible positions from whic I can move;
    for (all nextPos in moves) {
        if (!isWinning(nextPos)) return dp[pos] = true;
    }
    return dp[pos] = false;
}
```

# Juego de la Resta – Solución con dp

$O(n)$ :

```
bool isWinning(int n) {
    if (n == 0) {
        return false;
    }
    if (used[n]) {
        return dp[n];
    }
    used[n] = true;
    vector<int> moves;
    if (n - 1 >= 0) moves.push_back(n - 1);
    if (n - 2 >= 0) moves.push_back(n - 2);
    if (n - 5 >= 0) moves.push_back(n - 5);
    for (int nextPos : moves) {
        if (!isWinning(nextPos)) return dp[n] = true;
    }
    return dp[n] = false;
}
```

# Posiciones ganadoras y perdedoras

El algoritmo de posiciones ganadoras y perdedoras no solo sirve para juegos combinatorios imparciales, es más general que eso. En general puede ser aplicado cuando el grafo es un DAG.

Por lo que también puede ser aplicado en ciertos juegos partisanos o incluso juegos que sean “casi” combinatorios (que jueguen 3 jugadores en vez de 2 o que hayan empates, etc.)






Sin embargo, necesitarás más parámetros en tu dp mientras más complejo sea. Por ejemplo, si el juego es partisano, probablemente necesites en el dp un parámetro adicional booleano para saber de qué jugador es el turno.

Ejemplos:

- **UVA 10111 - Find the Winning Move**
- **Topcoder LeftToRightGame**



# Contenido

1. Conceptos básicos	
2. Posiciones Ganadoras y Perdedoras	
3. Algunas técnicas básicas	
4. Juego del Nim	
5. Suma de juegos	

# Strategy-Stealing Argument

Es un argumento no constructivo que sirve para demostrar que el primer jugador tiene estrategia ganadora.

El argumento se basa en suponer que si el 2do jugador tiene estrategia ganadora, entonces el 1er jugador puede robarse esa estrategia, siempre y cuando su primer movimiento no sea una desventaja. Entonces se llega a una contradicción ya que no es posible que ambos jugadores ganen a la vez.



# Juego Chomp

## Enunciado:

Se tiene una barra de chocolate rectangular de dimensiones  $m \times n$  que ha sido dividida en cuadraditos de  $1 \times 1$ . El cuadradito de la casilla  $(1, 1)$  ha sido envenenado y debe ser evitado a toda costa. Los jugadores 1 y 2 se turnan. En cada turno el jugador selecciona una casilla  $(a, b)$  que aún no haya sido comida y debe comer todos los cuadraditos que estén abajo o a la derecha de esa casilla, es decir todas las casillas  $(x, y)$  tal que  $x \geq a, y \geq b$ . El jugador que come la casilla envenenada, pierde.

Juego online : <https://cariboutests.com/games/chomp.php>



Fuente: <https://en.wikipedia.org/wiki/Chomp>

# Juego Chomp

## Solución:

- Si el tablero es de tamaño  $1 \times 1$  gana el 2do jugador (caso obvio)
- En todos los demás casos gana el 1er jugador

## Demostración:

Supongamos que el 2do jugador tiene una estrategia ganadora. Entonces no importa la casilla que escoja el jugador 1 en su primer movimiento, siempre estaría destinado a perder. Supongamos que el jugador 1 escogiera la casilla  $(n, m)$  en su primer movimiento. Luego el jugador 2 escogería una casilla de su supuesta estrategia ganadora. Digamos que esta sea la casilla  $(u, v)$ . Sin embargo, el jugador 1 también pudo haber escogido la casilla  $(u, v)$  en su primer movimiento, llegando al mismo estado. Por lo que se deduce que el jugador 1 también tenía una estrategia ganadora, llegando a una contradicción.



# Reflejando movimientos simétricamente

En problemas donde existe **simetría**, a veces es posible ganar con la estrategia de replicar los mismos movimientos de tu contrincante, pero reflejados simétricamente.

Ejemplo:

## Pintando la barra

Se tiene una barra rectangular compuesta por  $n$  cuadrados. Dos jugadores alternan turnos. En cada turno el jugador debe escoger una cantidad de  $x$  cuadrados consecutivos que todavía no hayan sido pintados y debe pintarlos. Se debe cumplir que  $x \in S$ , donde  $S$  es un conjunto de números enteros positivos. Se garantiza que 1 y 2 pertenecen a  $S$ . El jugador que ya no pueda realizar ningún movimiento (todos los cuadrados fueron pintados) pierde.

Si ambos jugadores juegan de forma óptima, ¿quién ganará?



# Reflejando movimientos simétricamente

## Pintando la barra

### **Solución:**

El 1er jugador siempre gana

### **Demostración:**

El primer jugador puede forzar a crear simetría. Si  $n$  es impar puede pintar el cuadradito del medio y dividir el rectángulo en 2 rectángulos de igual tamaño.



Si  $n$  es par, puede pintar 2 cuadraditos del medio y dividir el rectángulo en 2 rectángulos de igual tamaño



Luego de esto, el 1er jugador solo tiene que repetir simétricamente los movimientos de su oponente (si el jugador 2 hace algo a la izquierda, el jugador 1 lo hará a la derecha, y así). Como el rectángulo es finito, en algún momento se acabará y el 1er jugador hará el último movimiento, imitando al último movimiento de su oponente.






# Reflejando movimientos simétricamente

En problemas donde existe **simetría**, a veces es posible ganar con la estrategia de replicar los mismos movimientos de tu contrincante, pero reflejados simétricamente.

Más ejemplos:

- **Hackerrank- Towers Breakers**
- **Arabella 2017 – Competitive Seagulls**
- **Codeforces 493D – Vasya and Chess**
- **Codeforces 197A – Plate Game**

# Contenido

1. Conceptos básicos	
2. Posiciones Ganadoras y Perdedoras	
3. Algunas técnicas básicas	
4. Juego del Nim	
5. Suma de juegos	

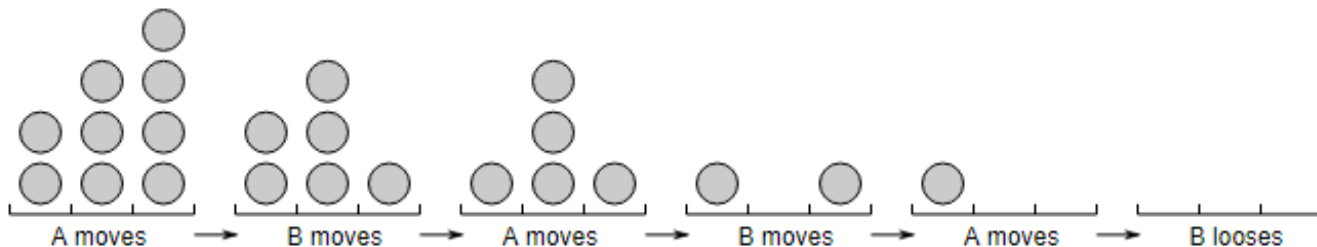


# Juego del Nim

Es el juego más importante de los combinatorios imparciales y del cual derivará un teorema muy importante.

## Enunciado:

Se tienen  $n$  pilas de piedras cuyas cantidades de piedras no necesariamente son iguales. Dos jugadores alternan turnos. En cada turno, el jugador debe escoger una pila de piedras (que todavía le queden piedras) y quitar una cantidad  $x$  de piedras ( $0 < x \leq \text{tamaño actual de la pila}$ ). Pierde el jugador que ya no pueda hacer más movimientos.



# Juego del Nim

## Algunos casos particulares fáciles de resolver

- $n = 1$

El primer jugador siempre gana cogiendo todas las piedras

- $n = 2$

**Caso 1:** Si las pilas son de igual tamaño, por simetría podemos demostrar que gana el 2do jugador imitando al 1ero.

**Caso 2:** Si las pilas son de distinto tamaño, el primer jugador gana ya que puede reducir la pila de mayor tamaño de tal forma que quede igual a la de menor tamaño y entramos al caso 1

- $n$  pilas de igual tamaño

**Caso 1:** Si  $n$  es par, entonces hay simetría y podemos demostrar que gana el 2do jugador imitando al primero.

**Caso 2 :** Si  $n$  es impar, entonces el primer jugador gana, eliminando cualquier pila y volviendo al caso 1

# Juego del Nim

## Caso general

Es posible resolverlo con dp con el algoritmo de Winning-Losing positions, pero esto sería demasiado lento para tamaños muy grandes de las pilas. Sin embargo, igual podemos usarlo para buscar algún patrón.

Intentemos para un  $n = 3$  y con distintos tamaños simular con dp cuáles son posiciones ganadoras y cuáles perdedoras. Si simulamos con nuestro programa, podemos notar que estas son las posiciones perdedoras para tamaños menores que 10.

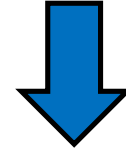
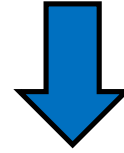
¿Ves algún patrón?

**Yo tampoco**

( 0 , 0 , 0 )
( 0 , 1 , 1 )
( 0 , 2 , 2 )
( 0 , 3 , 3 )
( 0 , 4 , 4 )
( 0 , 5 , 5 )
( 0 , 6 , 6 )
( 0 , 7 , 7 )
( 0 , 8 , 8 )
( 0 , 9 , 9 )
( 1 , 2 , 3 )
( 1 , 4 , 5 )
( 1 , 6 , 7 )
( 1 , 8 , 9 )
( 2 , 4 , 6 )
( 2 , 5 , 7 )
( 3 , 4 , 7 )
( 3 , 5 , 6 )

# Juego del Nim

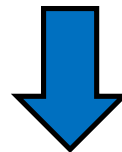
Probemos con operadores binarios



Posiciones perdedoras	Binario	AND	OR	XOR
( 0 , 0 , 0 )	( 0000 , 0000 , 0000 )	0	0	0
( 0 , 1 , 1 )	( 0000 , 0001 , 0001 )	0	1	0
( 0 , 2 , 2 )	( 0000 , 0010 , 0010 )	0	2	0
( 0 , 3 , 3 )	( 0000 , 0011 , 0011 )	0	3	0
( 0 , 4 , 4 )	( 0000 , 0100 , 0100 )	0	4	0
( 0 , 5 , 5 )	( 0000 , 0101 , 0101 )	0	5	0
( 0 , 6 , 6 )	( 0000 , 0110 , 0110 )	0	6	0
( 0 , 7 , 7 )	( 0000 , 0111 , 0111 )	0	7	0
( 0 , 8 , 8 )	( 0000 , 1000 , 1000 )	0	8	0
( 0 , 9 , 9 )	( 0000 , 1001 , 1001 )	0	9	0
( 1 , 2 , 3 )	( 0001 , 0010 , 0011 )	0	3	0
( 1 , 4 , 5 )	( 0001 , 0100 , 0101 )	0	5	0
( 1 , 6 , 7 )	( 0001 , 0110 , 0111 )	0	7	0
( 1 , 8 , 9 )	( 0001 , 1000 , 1001 )	0	9	0
( 2 , 4 , 6 )	( 0010 , 0100 , 0110 )	0	6	0
( 2 , 5 , 7 )	( 0010 , 0101 , 0111 )	0	7	0
( 3 , 4 , 7 )	( 0011 , 0100 , 0111 )	0	7	0
( 3 , 5 , 6 )	( 0011 , 0101 , 0110 )	0	7	0

# Juego del Nim

Problemas con más casos para  $n = 4$



Posiciones Perdedoras	Binario	AND	OR	XOR
( 1 , 1 , 1 , 1 )	( 00001 , 00001 , 00001 , 00001 )	1	1	0
( 1 , 1 , 2 , 2 )	( 00001 , 00001 , 00010 , 00010 )	0	3	0
( 1 , 1 , 3 , 3 )	( 00001 , 00001 , 00011 , 00011 )	1	3	0
( 1 , 1 , 4 , 4 )	( 00001 , 00001 , 00100 , 00100 )	0	5	0
( 1 , 1 , 5 , 5 )	( 00001 , 00001 , 00101 , 00101 )	1	5	0
( 2 , 2 , 2 , 2 )	( 00010 , 00010 , 00010 , 00010 )	2	2	0
( 2 , 2 , 3 , 3 )	( 00010 , 00010 , 00011 , 00011 )	2	3	0
( 2 , 2 , 4 , 4 )	( 00010 , 00010 , 00100 , 00100 )	0	6	0
( 2 , 2 , 5 , 5 )	( 00010 , 00010 , 00101 , 00101 )	0	7	0
( 2 , 3 , 4 , 5 )	( 00010 , 00011 , 00100 , 00101 )	0	7	0
( 3 , 3 , 3 , 3 )	( 00011 , 00011 , 00011 , 00011 )	3	3	0
( 3 , 3 , 4 , 4 )	( 00011 , 00011 , 00100 , 00100 )	0	7	0
( 3 , 3 , 5 , 5 )	( 00011 , 00011 , 00101 , 00101 )	1	7	0
( 4 , 4 , 4 , 4 )	( 00100 , 00100 , 00100 , 00100 )	4	4	0
( 4 , 4 , 5 , 5 )	( 00100 , 00100 , 00101 , 00101 )	4	5	0
( 5 , 5 , 5 , 5 )	( 00101 , 00101 , 00101 , 00101 )	5	5	0

# Juego del Nim

## Teorema

En el juego de Nim una posición  $(x_1, x_2, \dots, x_n)$  es perdedora si y solo si  $s = x_1 \oplus x_2 \oplus \dots \oplus x_n = 0$   
 $\oplus$  es el operador xor.

## Demostración

Demostremos por inducción.

- **Caso base:** Todas las pilas tienen tamaño 0. El xor también es 0 y efectivamente es una posición perdedora
- **Paso inductivo:** Supongamos que estamos en una posición del Nim  $(x_1, x_2, \dots, x_n)$ , y por hipótesis inductiva, todas las posiciones a las que nos podemos mover cumplen con el teorema. Entonces aquí hay que comprobar dos propiedades.
  - ☐ De una posición perdedora solo se puede ir a ganadoras
  - ☐ De una posición ganadora puedo ir, al menos, a una perdedora

# Juego del Nim

## **Demostración (continuación)**

Supongamos que estás en una posición con xor actual  $s$ . Si en tu turno escoges cambiar una pila de tamaño  $a$  y reducirla a un tamaño  $b$  ( $b < a$ ). Entonces el xor ahora será  $t = s \oplus a \oplus b$

### **□ De una posición perdedora solo se puede ir a ganadoras**

Si estás en una posición con  $s = 0$ . Entonces el xor ahora será  $t = s \oplus a \oplus b = a \oplus b$

Como  $b < a$ , entonces  $t \neq 0$ , que por inducción es una posición ganadora.

Entonces desde una posición con xor 0, solo puedo ir a posiciones que, por inducción, son ganadoras.

# Juego del Nim

## □ De una posición ganadora puedo ir, al menos, a una perdedora

Si estás en una posición con  $s \neq 0$ . Tenemos que demostrar que existe un movimiento que nos lleve a un  $t = s \oplus a \oplus b = 0$ . Para ello, revisemos la representación binaria de  $s$  y busquemos su último bit prendido (el más significativo). Denotemos esa posición como  $d$ . Entonces escogeré una pila  $a$  que cumpla que tenga el bit de la posición  $d$  (debe existir una pila de ese tipo ya que  $s$  lo tiene prendido) y decido reducir su tamaño a un tamaño  $b = a \oplus s$ . El siguiente esquema muestra que  $b < a$  y que, por lo tanto, es un movimiento válido

<i>pos</i>	<i>last</i>			$d + 1$	$d$	...
$s$	0	0	...	0	1	...
$a$	$a_{last}$	$a_{last-1}$	...	$a_{d+1}$	1	...
$b = a \oplus s$	$a_{last}$	$a_{last-1}$	...	$a_{d+1}$	0	...

Finalmente tendríamos que  $t = s \oplus a \oplus (a \oplus s) = 0$ , lo cual demuestra lo que necesitábamos



# Contenido

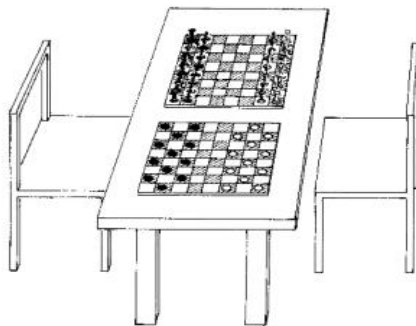
1. Conceptos básicos	
2. Posiciones Ganadoras y Perdedoras	
3. Algunas técnicas básicas	
4. Juego del Nim	
5. Suma de juegos	

# Suma de juegos

**Definición:** Sean dos juegos independientes  $A, B$  se define la suma de juegos  $G = A + B$  como el juego en donde en cada turno cada jugador elige un juego en el que aún pueda hacer movimientos y luego haga un movimiento legal en ese juego dejando **el otro juego intacto**.

Es claro que esta definición se puede extender a una suma de varios juegos.

A partir de ahora nos enfocaremos solo en los juegos combinatorios imparciales con modo de juego normal. Aunque algunas fórmulas, propiedades o teoremas también pueden aplicarse a algunos juegos en general, muchas de ellas no podrán ser aplicadas a cualquier juego.



Fuente : Winning ways for your mathematical plays

# Suma de juegos

Como ejemplo, considere al Juego del Nim, que puede considerarse como una suma de juegos en donde cada juego es una sola pila de piedras. Como el Nim es un juego especial, cada pila de piedras tiene una notación especial. Denotamos a  $*n$  como el juego del Nim de una sola pila con  $n$  piedras

Entonces, por ejemplo, el siguiente juego del Nim :



Fuente : <https://www.pinterest.com/pin/161637074099035109/>

Puede expresarse como el juego  $*1 + *2 + *3 + *4 + *3 + *2 + *1$

# Suma de juegos

## Juego Cero:

Se define al Juego 0 (cero) como aquel en donde ya no se pueden hacer más movimientos. (Es fácil notar que 0 es el mismo juego que  $* 0$ )

## Operador resultado:

Denotemos a  $res(A)$  como el resultado del juego A. Es decir  $res(A) = 1$  si el juego tiene posición inicial ganadora y  $res(A) = 0$  si el juego tiene posición inicial perdedora.

- ☐  $res(A + 0) = res(A)$
- ☐  $res(A + A) = 0$  (utilizando la táctica de replicar en simetría)

# Suma de juegos

Algo que nos interesa saber es si podemos encontrar una regla sencilla para saber el resultado de una suma de juegos en general. En una primera instancia podríamos estar tentados a pensar que si tengo un juego  $A + B$  en donde  $A, B$  están en una posición ganadora, es decir  $res(A) = 1$  y  $res(B) = 1$ , entonces  $A + B$  también lo estará. Sin embargo **esto no necesariamente se cumple**.

Contraejemplo:  $* 1$  tiene posición ganadora, pero  $* 1 + * 1$  no.

Sin embargo, algo que sí se cumple es:

□ **Lema 1:** si  $res(A) = 0$  y  $res(B) = 0$ , entonces  $res(A + B) = 0$

## Demostración:

Se puede demostrar por inducción.

- **Caso base:** La suma dos juegos en los que ya no se pueden hacer movimientos es obviamente un juego con posición perdedora
- **Paso inductivo:** Inicialmente el primer jugador está en un par de juegos con posiciones  $P - P$ . Luego de que el primer jugador haga su movimiento, el juego pasará a estar en una posición  $G - P$  (o  $P - G$ ). El 2do jugador solo tiene que escoger el mismo juego y llevarlo a una posición  $P - P$ , que por inducción es perdedora.

# Equivalencia de juegos

**Definición:** Dos juegos serán equivalentes si es que puedo intercambiarlos en cualquier suma de juegos y esto no alterará el resultado.

Es decir  $A \equiv B$  si para cualquier juego  $G$  se cumple que  $\text{res}(G + A) = \text{res}(G + B)$

Si reemplazamos  $G$  por el juego 0 en la definición, entonces tendremos que:

Si  $A \equiv B \Rightarrow \text{res}(A) = \text{res}(B)$ . (Nota: lo inverso no necesariamente se cumple)

## Propiedades:

- ☐ La suma de juegos es conmutativa:  $A + B \equiv B + A$
- ☐ La suma de juegos es asociativa:  $(A + B) + C \equiv A + (B + C)$
- ☐ El juego 0 es el elemento neutro:  $A + 0 \equiv A$
- ☐ Todo juego combinatorio imparcial es inverso de sí mismo:  $A + A \equiv 0$  (se demostrará en la siguiente diapo)
- ☐  $A \equiv B$  si y solo si  $A + C \equiv B + C$ , para cualquier juego  $C$

# Equivalencia de juegos

□ **Lema 2:**  $A \equiv 0 \Leftrightarrow res(A) = 0$

$\Rightarrow$

Si  $A \equiv 0$ , por definición  $res(A + G) = res(0 + G)$  para cualquier  $G$ . Reemplazamos  $G$  por 0  
 $res(A) = res(0) = 0$

$\Leftarrow$

Si  $res(A) = 0$ , estamos en un juego con posición perdedora. Queremos demostrar que para cualquier juego  $G$ , se cumple que  $res(G + A) = res(G)$ .

- 1) Si  $res(G) = 0$ , entonces por el Lema 1 como  $res(A) = 0$  y  $res(G) = 0$ , entonces  $res(A + G) = 0 = res(G)$
- 2) Si  $res(G) = 1$ , entonces el 1er jugador puede elegir  $G$  en  $G + A$  y seguir su estrategia ganadora en  $G$ , en ese momento el juego se moverá a un estado  $G' + A$  en donde  $res(G') = 0$  y  $res(A) = 0$ , que por el lema 1, se tiene  $res(G' + A) = 0$ . Entonces estamos demostrando que desde la posición inicial de  $G + A$  podemos ir a una posición perdedora. Es decir,  $G + A$  tiene posición ganadora.  $res(A + G) = 1 = res(G)$

Se concluye que  $res(G + A) = res(G)$  para cualquier  $G$  y entonces  $A \equiv 0$

□ **Corolario 1:**  $A + A \equiv 0$ , es decir, todo juego (combinatorio imparcial) es inverso de sí mismo

# Equivalencia de juegos

□ **Lema 3:**  $A \equiv B \Leftrightarrow A + B \equiv 0$  (juegos combinatorios imparciales)

**Demostración**

$\Rightarrow$

$$A \equiv B$$

$$A + B \equiv B + B \text{ (Sumamos } B \text{ a ambo lados)}$$

$$A + B \equiv 0 \text{ (Por el corolario anterior)}$$

$\Leftarrow$

$$A + B \equiv 0$$

$$A + B + B \equiv B + 0 \text{ (Sumamos } B \text{ a ambo lados)}$$

$$A + 0 \equiv B + 0 \text{ (Propiedad asociativa + corolario anterior)}$$

$$A \equiv B \text{ (Elemento neutro)}$$

□ **Corolario 2:**  $A \equiv B \Leftrightarrow \text{res}(A + B) = 0$  (juegos combinatorios imparciales)

**Demostración:** Usar lema 2 y 3

En conclusión, hemos encontrado otra forma para saber cuándo dos juegos serán equivalentes



# Equivalencia de juegos

□ **Lema 4 (Suma de Nims):** Sean los juegos  $*n$ ,  $*m$ , la suma  $*n + *m \equiv *(n \oplus m)$

## Demostración

Sea el juego  $G = *n + *m + *(n \oplus m)$ , sabemos que  $G$  es el juego del Nim de 3 pilas. El resultado depende del xor de los tamaños de las 3 pilas.  $s = n \oplus m \oplus (n \oplus m) = 0$

Es decir  $res(G) = 0$

Utilizando el corolario 2 :  $*n + *m \equiv *(n \oplus m)$

□ **Generalización del Lema 4:**  $*n_1 + *n_2 + \dots + *n_k = *(n_1 \oplus n_2 \oplus \dots \oplus n_k)$

# Sprague-Grundy Theorem

Roland Sprague y Patrick Grundy descubrieron de forma independiente (en 1935 y 1939) el siguiente teorema:

- ❑ **Teorema:** Todo juego combinatorio imparcial  $G$  con modo de juego normal es equivalente a un juego de Nim de una sola pila. Es decir existe un  $n$  tal que  $G \equiv *n$ . Donde  $n$  es el menor entero no negativo tal que  $G$  no tiene un movimiento que lo lleve a un juego equivalente a  $*n$ . El número  $n$  recibe el nombre de **grundy number**.



# Grundy Numbers

Antes de demostrarlo, analicemos un poco más sobre lo que nos quiere decir este teorema y sobre los Grundy numbers.

El teorema nos dice que podemos representar cualquier juego combinatorio imparcial  $G$  por una pila de tamaño  $\text{grundy}(G)$ . Donde podemos hallar recursivamente este número *grundy* de la siguiente forma:

- **Caso base:** Si  $G = 0$ , entonces  $\text{grundy}(G) = 0$  (debido a que  $0 = * 0$ )
- **Caso general:** Denotemos a los juegos a los que puede transicionar  $G$  como  $G_0, G_1, \dots, G_x$

Entonces  $\text{grundy}(G) = \text{mex}(\text{grundy}(G_0), \text{grundy}(G_1), \dots, \text{grundy}(G_x))$

Donde la función  $\text{mex}()$  es el mínimo entero no negativo excluido (del inglés *minimum excluded*)

Ejemplos del  $\text{mex}$ :

$$\text{mex}([ ]) = 0$$

$$\text{mex}([1, 2, 3]) = 0$$

$$\text{mex}([0, 2, 4, 6]) = 1$$

$$\text{mex}([0, 1, 2, 3, 8, 100, 359]) = 4$$

$$\text{mex}([0, 1, 2, 3, \dots, x]) = x+1$$

# Sprague-Grundy Theorem

## Demostración del Teorema:

Demostremos por inducción en las opciones que tiene  $G$

- **Caso base:** Si  $G$  no tiene opciones, entonces  $G \equiv * 0$  y se cumple que  $0 = \text{mex}([ ])$
- **Paso inductivo:**

Queremos demostrar que  $G \equiv * n$ , donde  $n = \text{mex}([ \text{grundy}(G_i) ])$

Si demostramos que  $\text{res}(G + * n) = 0$ , por el corolario 2 estaríamos demostrando la equivalencia.

Demostremos entonces, que  $G + * n$  tiene posición perdedora. El primer jugador tiene 2 opciones: jugar en  $G$  o en  $* n$ . Demostremos que cualquier decisión de estas lo lleva a posiciones ganadoras (lo cual demostraría que está en una perdedora).

1. Si juega en  $G$ , se moverá a un  $G_i$  que, por hipótesis inductiva,  $G_i \equiv * m$  y  $m \neq n$  (por definición de mex) entonces se habrá movido de  $G + * n$  a un juego  $* m + * n$ , que es un caso trivial del juego del Nim y tiene posición ganadora.
2. Si juega en  $* n$  se moverá a un  $* m$  tal que  $m < n$  (por las reglas del Nim). Entonces se habrá movido de  $G + * n$  a  $G + * m$ . Note que  $G + * m$  es ganadora porque, por hipótesis inductiva, puedo mover en  $G$  y pasar a un juego  $G_i = * m$  (si no fuera posible,  $n$  no sería el mex) lo cual nos llevaría a un juego  $* m + * m$  que es perdedora. Por lo que  $G + * m$  es ganadora, lo cual completa la prueba.

# Algoritmo Grundy Numbers

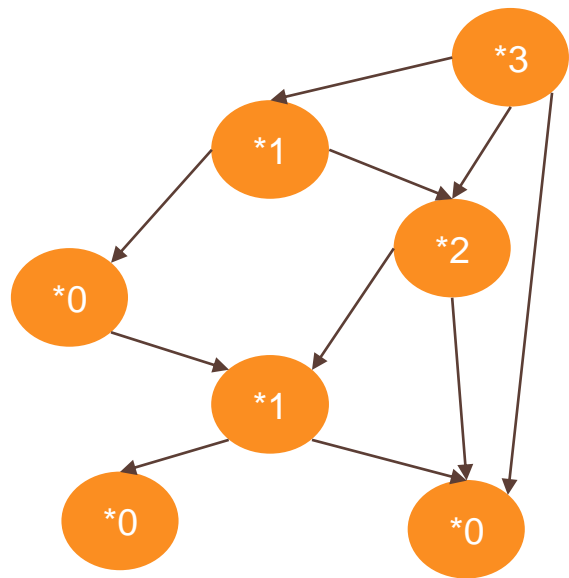
**Corolario:** Si un juego  $G$  tiene  $grundy(G) = 0$ , entonces tendrá posición perdedora, caso contrario (tiene  $grundy(G) > 0$ ) tendrá posición ganadora.

Esto quiere decir que el algoritmo para hallar los *grundy numbers* puede reemplazar al algoritmo de Winning Losing Positions. Pero como veremos en el último teorema, los *grundy numbers* son más poderosos.

## Algoritmo para hallar grundy numbers

- **Caso base:** Si  $G = 0$ , entonces  $grundy(G) = 0$  (debido a que  $0 = *0$ )
- **Caso general:** Denotemos a los juegos a los que puede transicionar  $G$  como  $G_0, G_1, \dots, G_x$

Entonces  $grundy(G) = mex(grundy(G_0))$



# Algoritmo Grundy Numbers

```
bool used[];
int dp[];

int grundy(Position pos) {
    if (pos is terminal position) {
        return 0;
    }
    if (used[pos]) {
        return dp[pos];
    }
    used[pos] = true;
    vector<Position> moves = possible positions from which I can move;
    vector<int> g;
    for (all nextPos in moves) {
        g.push_back(grundy(nextPos));
    }
    return dp[pos] = mex(g);
}
```

# Función mex

```
int mex(vector<int> &s) { //O(n)
    int n = s.size();
    vector<bool> marked(n , false);
    for(int x : s) {
        if(x < n) marked[x] = true;
    }
    for(int x = 0; x < n; x++) {
        if(!marked[x]) return x;
    }
    return n;
}
```

# Teorema de la suma de Juegos

**Teorema:** Si  $A, B$  son juegos combinatorios imparciales. Entonces  $\text{grundy}(A + B) = \text{grundy}(A) \oplus \text{grundy}(B)$   
Es decir  $A + B \equiv * (\text{grundy}(A) \oplus \text{grundy}(B))$

**Demostración:**

Se deduce directamente del teorema Sprague-Grundy y de la Suma de Nims (Lema 4)

**Generalización:** Para cualquier conjunto de juegos combinatorios imparciales  $A_1, A_2, \dots, A_n$  se cumple que  
 $\text{grundy}(A_1 + A_2 + \dots + A_n) = \text{grundy}(A_1) \oplus \text{grundy}(A_2) \oplus \dots \oplus \text{grundy}(A_n)$



# Aplicación Teorema Suma de Juegos

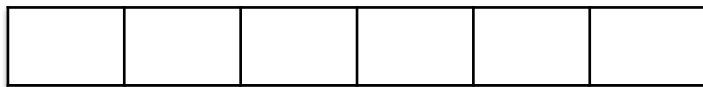
Obviamente podemos aplicarlo en problemas donde explícitamente tenemos una suma de juegos, pero también se puede aplicar en problemas donde **se inicia con un solo juego** pero existen movimientos que pueden “dividir” el juego original en varios juegos.

Ejemplo:

## Pintando la barra (versión medium)

Se tiene una barra rectangular compuesta por  $n$  cuadrados. Dos jugadores alternan turnos. En cada turno el jugador debe escoger una cantidad de 2 cuadrados consecutivos que todavía no hayan sido pintados y debe pintarlos. El jugador que ya no pueda realizar ningún movimiento pierde.

Si ambos jugadores juegan de forma óptima, ¿quién ganará?

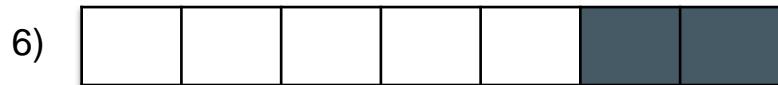
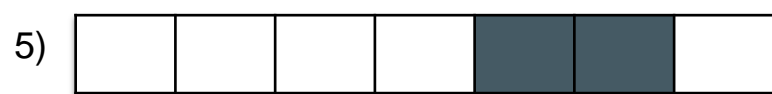


# Aplicación Teorema Suma de Juegos

## Pintando la barra (versión medium)

- **Caso 1:** Si  $n$  es par, entonces por simetría gana el primer jugador (ya lo visto antes)
- **Caso 2:** No se puede hacer simetría

Veamos una de las posibles transiciones para un  $n = 7$



Podemos observar que en las transiciones 1 y 6, se pasa a un solo juego con  $n = 5$ , sin embargo, las demás transiciones “dividen” el juego original en 2 juegos independientes (suma de juegos). Por ejemplo la transición 2 divide al juego en  $n_1 = 1$  y  $n_2 = 4$ . Por lo tanto, ya no es posible aplicar el algoritmo WL, sino que debemos aplicar los Grundy numbers

# Aplicación Teorema Suma de Juegos

## Pintando la barra (versión medium)



$$grundy(n) = mex(grundy(i) \oplus grundy(n - i - 2)), \forall i \in [0, n - 2]$$

$$O(n^2)$$

# Aplicación Teorema Suma de Juegos

Pintando la barra (versión medium)

$$\text{grundy}(n) = \text{mex}(\text{grundy}(i) \oplus \text{grundy}(n - i - 2)), \forall i \in [0, n - 2]$$

```
int grundy(int n) { //O(n^2)
    /*if (n <= 1) {
        return 0;
    }*/
    if (used[n]) {
        return dp[n];
    }
    used[n] = true;
    vector<int> g;
    for (int i = 0; i <= n - 2; i++) {
        g.push_back(grundy(i) ^ grundy(n - i - 2));
    }
    return dp[n] = mex(g);
}
```

# Aplicación Teorema Suma de Juegos

## Pintando la barra (versión HARD)

Se tiene una barra rectangular compuesta por  $n$  cuadrados. Dos jugadores alternan turnos. En cada turno el jugador debe escoger una cantidad de  $x$  cuadrados consecutivos que todavía no hayan sido pintados y debe pintarlos. Se debe cumplir que  $x \in S$ , donde  $S$  es un conjunto de números enteros positivos. El jugador que ya no pueda realizar ningún movimiento pierde.

Si ambos jugadores juegan de forma óptima, ¿quién ganará?

### **Solución:**

Se puede resolver bajo el mismo principio que la versión anterior pero aplicándolo para todos los  $x \in S$

$$grundy(n) = mex(grundy(i) \oplus grundy(n - i - x)), \forall x \in S, \forall i \in [0, n - x]$$

$$O(n^2|S|)$$

# Aplicación Teorema Suma de Juegos

## Pintando la barra (versión HARD)

```
vector<int> S;  
int Grundy(int n) { //O(|S| * n^2)  
    if (used[n]) {  
        return dp[n];  
    }  
    used[n] = true;  
    vector<int> g;  
    for (int x : S) {  
        for (int i = 0; i <= n - x; i++) {  
            g.push_back(Grundy(i) ^ Grundy(n - i - x));  
        }  
    }  
    return dp[n] = mex(g);  
}
```

# Referencias

- ❑ Topcoder: <https://www.topcoder.com/community/competitive-programming/tutorials/algorithm-games/>
- ❑ UCLA: [https://www.math.ucla.edu/~tom/Game\\_Theory/comb.pdf](https://www.math.ucla.edu/~tom/Game_Theory/comb.pdf)
- ❑ Winning ways for your mathematical plays, Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy
- ❑ CP-algorithms: [https://cp-algorithms.com/game\\_theory/sprague-grundy-nim.html](https://cp-algorithms.com/game_theory/sprague-grundy-nim.html)
- ❑ Agustín Gutiérrez – Training Camp Argentina 2019: <https://tc-arg.tk/pdfs/2019/juegos.pdf>
- ❑ University of Munster: <https://ivv5hpp.uni-muenster.de/u/baysm/teaching/3u03/notes/14-games.pdf>