Álgebra Lineal en ICPC

Franco De Rico[1]

[1]Universidad Nacional de Rosario
Don Gato

Training Camp 2025

¡Gracias sponsors!

Organizador

Diamond Plus

¡Gracias sponsors!

Platinum

FOLDER IT

INTERNATIONAL
SOFTWARE COMPANY

Gold

NeuralSoft

JS JERÁRQUICOS

¡Gracias sponsors!

Aliado

Santa Fe
PROVINCIA

Matrices

Una matriz es un tablero con números. Decimos que es de $m \times n$ si tiene m filas y n columnas.
Si una matriz A es de $m \times n$ y tiene todas sus entradas reales, decimos que A está en $\mathbb{R}^{m \times n}$.

$$A = \begin{bmatrix} 0,5 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & \frac{1}{3} & 2 \end{bmatrix} \in \mathbb{R}^{3 \times 5}$$

Producto

Para multiplicar una matriz por otra necesitamos que la cantidad de columnas de la primera sea la cantidad de filas de la segunda.
Es decir, tenemos que poner una matriz de $m \times n$ con otra de $n \times p$.

$$\begin{bmatrix} 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0,5 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & \frac{1}{3} & 2 \end{bmatrix} \begin{bmatrix} & 9 & & & & \\ & & & & & \\ & & & & & \end{bmatrix}$$

$$M_0^1 = 9$$

Producto de matriz por vector

Es multiplicar una matriz de $m \times n$ por un vector vertical, que sería una matriz de $n \times 1$.
Está bueno tener presente la noción de que el vector vertical está combinando linealmente las columnas de la matriz.

$$\begin{array}{cc} & \begin{bmatrix} a \\ b \\ c \end{bmatrix} \\ \begin{bmatrix} 1 & 5 & 3 \\ 2 & 6 & 4 \end{bmatrix} & = a \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} + b \cdot \begin{bmatrix} 5 \\ 6 \end{bmatrix} + c \cdot \begin{bmatrix} 3 \\ 4 \end{bmatrix} \end{array}$$

Sistemas de ecuaciones

Podemos plantear un sistema de ecuaciones como un producto de matrices. En general nos gusta que sean de $n \times n$, pero tranquilamente podría ser de $m \times n$ con $m \neq n$.

$$
\begin{aligned}
2a + 3b + 5c + 8d + 9e &= 3 \\
2c + 4d + 5e &= 2
\end{aligned}
$$

Sistemas de ecuaciones

Podemos plantear un sistema de ecuaciones como un producto de matrices. En general nos gusta que sean de $n \times n$, pero tranquilamente podría ser de $m \times n$ con $m \neq n$.

$$\begin{aligned} 2a + 3b + 5c + 8d + 9e &= 3 \\ 2c + 4d + 5e &= 2 \end{aligned}$$

$$\begin{array}{cc} & \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} \\ \begin{bmatrix} 2 & 3 & 5 & 8 & 9 \\ 0 & 0 & 2 & 4 & 5 \end{bmatrix} & \begin{bmatrix} 3 \\ 2 \end{bmatrix} \end{array}$$

Gauss

Este algoritmo es la forma estándar de resolver sistemas de ecuaciones. Lo que hace es triangular la matriz y en base a eso ir despejando de abajo hacia arriba.
Para triangular, tomamos la primera fila y se la restamos a las otras para dejarles la primera columna en 0. Después dividimos por el primer elemento, para que este quede en 1. Hacemos eso con todas las filas, y cada una va matando una columna.

Gauss

Este proceso de hacer combinaciones lineales entre las filas es interesante porque mantiene muchas propiedades de la matriz y del sistema de ecuaciones (en particular, su solución).

Implementación :)

(O googlear "la pandilla github")

Gauss

$$
\begin{aligned}
2a + 3b + 5c + 8d + 9e &= 3 \\
2c + 4d + 5e &= 2
\end{aligned}
$$

$$
\left[\begin{array}{ccccc|c}
2 & 3 & 5 & 8 & 9 & 3 \\
0 & 0 & 2 & 4 & 5 & 2
\end{array}\right]
$$

En el ejemplo anterior, si reducimos la matriz extendida, llegamos a esto:

Gauss

$$2a + 3b + 5c + 8d + 9e = 3$$
$$2c + 4d + 5e = 2$$

$$\begin{bmatrix} 2 & 3 & 5 & 8 & 9 & | & 3 \\ 0 & 0 & 2 & 4 & 5 & | & 2 \end{bmatrix}$$

En el ejemplo anterior, si reducimos la matriz extendida, llegamos a esto:

$$\begin{bmatrix} \mathbf{1} & 1,5 & 0 & -1 & -1,75 & | & -1 \\ 0 & 0 & \mathbf{1} & 2 & 2,5 & | & 1 \end{bmatrix}$$

Gauss

Los 1 en rojo son los que usamos para ir triangulando la matriz. Por lo tanto solo tienen 0 en su columna. Los llamamos *pivots*.

$$\begin{bmatrix} \mathbf{1} & 1,5 & 0 & -1 & -1,75 & | & -1 \\ 0 & 0 & \mathbf{1} & 2 & 2,5 & | & 1 \end{bmatrix}$$

Gauss

Los 1 en rojo son los que usamos para ir triangulando la matriz. Por lo tanto solo tienen 0 en su columna. Los llamamos *pivots*.

$$\left[\begin{array}{ccccc|c} \mathbf{1} & 1,5 & 0 & -1 & -1,75 & -1 \\ 0 & 0 & \mathbf{1} & 2 & 2,5 & 1 \end{array}\right]$$

Casi que ni nos importan los números en sí. Lo que es importante notar es que con las variables de las columnas sin pivot podemos hacer lo que queramos (es decir, asignarles cualquier número real), y por eso las llamamos *variables libres*. Pero con las que tienen pivots, no (debemos despejarlas en base a los valores de las otras).

Gauss

Los 1 en rojo son los que usamos para ir triangulando la matriz. Por lo tanto solo tienen 0 en su columna. Los llamamos *pivots*.

$$\begin{bmatrix} \mathbf{1} & 1,5 & 0 & -1 & -1,75 & | & -1 \\ 0 & 0 & \mathbf{1} & 2 & 2,5 & | & 1 \end{bmatrix}$$

Casi que ni nos importan los números en sí. Lo que es importante notar es que con las variables de las columnas sin pivot podemos hacer lo que queramos (es decir, asignarles cualquier número real), y por eso las llamamos *variables libres*. Pero con las que tienen pivots, no (debemos despejarlas en base a los valores de las otras).
Notar que en un sistema de ecuaciones con solución única nos van a quedar todos pivots. O sea, una matriz identidad, que tiene pegada la columna del término independiente. Esta nos dice cuánto va a valer cada variable.

Problemas

Los siguientes problemas los pueden encontrar en
este contest en VJudge.
La contraseña es tcAR2025
Además, cada uno tiene en el título su enlace individual.

Mudanza

Juli tenía en su casa $n \leq 1000$ números naturales, tales que ninguno tiene un divisor primo mayor a 500.

Ahora que se mudó, está decidiendo cuáles de ellos llevarse. Como su nueva casa es cuadrada, le gustaría elegir algunos tales que su producto sea un cuadrado perfecto.

¿De cuántas formas distintas puede hacerlo?

No podemos probar todos los subconjuntos, porque sería exponencial.

No podemos probar todos los subconjuntos, porque sería exponencial.
Hasta el 500 hay más o menos 100 primos.

No podemos probar todos los subconjuntos, porque sería exponencial.
Hasta el 500 hay más o menos 100 primos.
Y si hablamos de cuadrados perfectos y factores primos, solo nos importa la paridad del exponente de cada uno. Entonces podemos pensar a cada número como una lista de sus exponentes. O mejor dicho, de la paridad de sus exponentes. Esto es un vector de 0 y 1 (0 si es par, 1 si es impar).

Queremos multiplicar varios de estos números. Eso significa sumar los vectores de los exponentes.

Queremos multiplicar varios de estos números. Eso significa sumar los vectores de los exponentes. Y queremos que en cada primo el exponente sea par. O sea que queremos que la suma mod 2 en cada entrada nos de 0.

Queremos multiplicar varios de estos números. Eso significa sumar los vectores de los exponentes. Y queremos que en cada primo el exponente sea par. O sea que queremos que la suma mod 2 en cada entrada nos de 0. Si miramos bien, podemos pensar que estamos sumando todos los vectores, pero algunos de ellos (los que no elegimos) van multiplicados por un 0.

Queremos multiplicar varios de estos números. Eso significa sumar los vectores de los exponentes. Y queremos que en cada primo el exponente sea par. O sea que queremos que la suma mod 2 en cada entrada nos de 0.
Si miramos bien, podemos pensar que estamos sumando todos los vectores, pero algunos de ellos (los que no elegimos) van multiplicados por un 0.
Lo que hace que verlo de esta forma sea interesante es que pasamos a tener una combinación lineal de los vectores.
Y sé cuál quiero que sea el resultado: $0\,0\ldots0$.

Lo que estamos haciendo entonces es buscar una combinación lineal de algunos vectores de forma de obtener el vector nulo. Y esto es justo lo que vimos en las slides 7 y 8.

Lo que estamos haciendo entonces es buscar una combinación lineal de algunos vectores de forma de obtener el vector nulo. Y esto es justo lo que vimos en las slides 7 y 8.
Escribamos cada uno de los n números como vector columna, como hicimos recién, y pongámoslo en una matriz A.
Queremos un vector x con entradas $0/1$ tal que $Ax = 0$. Para eso reduzcamos esta matriz, pero módulo 2.

Por ejemplo, si tenemos los números 2, 30, 3, 15, 6, la matriz A queda:

$$\left[\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right]$$

Por ejemplo, si tenemos los números 2, 30, 3, 15, 6, la matriz A queda:

$$\left[\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right]$$

La matriz reducida A' ya no tiene los mismos números, pero mantuvo todas sus propiedades importantes:

Por ejemplo, si tenemos los números 2, 30, 3, 15, 6, la matriz A queda:

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 1 & | & 0 \\ 0 & 1 & 1 & 1 & 1 & | & 0 \\ 0 & 1 & 0 & 1 & 0 & | & 0 \end{bmatrix}$$

La matriz reducida A' ya no tiene los mismos números, pero mantuvo todas sus propiedades importantes: un conjunto de columnas es válido en A si y solo si lo es en A'.

$$\begin{bmatrix} \mathbf{1} & 0 & 0 & 1 & 1 & | & 0 \\ 0 & \mathbf{1} & 0 & 1 & 0 & | & 0 \\ 0 & 0 & \mathbf{1} & 0 & 1 & | & 0 \end{bmatrix}$$

Recordemos ahora que con las variables libres podemos hacer lo que sea. En $\mathbb{R}$ eso era elegir cualquier número real.

Recordemos ahora que con las variables libres podemos hacer lo que sea. En $\mathbb{R}$ eso era elegir cualquier número real.
Ahora, significa que podemos elegir tanto 0 como 1. Y para cualquier elección de 0 y 1 de las variables libres, las variables con pivot quedan determinadas.

Recordemos ahora que con las variables libres podemos hacer lo que sea. En $\mathbb{R}$ eso era elegir cualquier número real.
Ahora, significa que podemos elegir tanto 0 como 1. Y para cualquier elección de 0 y 1 de las variables libres, las variables con pivot quedan determinadas.
Luego, podemos contar la cantidad c de variables libres, y nuestra respuesta es 2^c (un conjunto por cada elección de $0/1$ de las variables libres).

Resumen:

- Escribir un vector de 0 y 1 con los exponentes de la factorización de cada número.
- Ponerlos en una matriz y reducirla.
- Contar la cantidad c de variables libres. La respuesta es 2^c.

Resumen:

- Escribir un vector de 0 y 1 con los exponentes de la factorización de cada número.
- Ponerlos en una matriz y reducirla.
- Contar la cantidad c de variables libres. La respuesta es 2^c.

Variación del problema: ¿qué pasaría si solo nos importan los subconjuntos con una cantidad impar de números?

Maes

Maricruz, Beto y Neto tienen $n \leq 10^5$ números naturales, todos menores a 2^{30}.

Como saben que Juli se mudó y está adornando su nueva casa con números, quieren llevarle algunos de regalo.

Es sabido que en Costa Rica tienen el mejor xor bit a bit del mundo. Es por eso que, como regalo, van a traer el subconjunto con mayor xor bit a bit posible.

Los xor tienen una propiedad interesante: si a un elemento lo podemos construir como xor de otros, entonces este elemento no lo necesitamos, porque podemos generar todos los que generábamos con él usando a sus creadores.

Los xor tienen una propiedad interesante: si a un elemento lo podemos construir como xor de otros, entonces este elemento no lo necesitamos, porque podemos generar todos los que generábamos con él usando a sus creadores.
En álgebra lineal, un conjunto de elementos donde ninguno es combinación lineal de otros es un conjunto *linealmente independiente*.
Si con un conjunto linealmente independiente C podemos generar otro conjunto D mediante combinaciones lineales, decimos que C es *base* de D.

Los xor tienen una propiedad interesante: si a un elemento lo podemos construir como xor de otros, entonces este elemento no lo necesitamos, porque podemos generar todos los que generábamos con él usando a sus creadores.
En álgebra lineal, un conjunto de elementos donde ninguno es combinación lineal de otros es un conjunto *linealmente independiente*.
Si con un conjunto linealmente independiente C podemos generar otro conjunto D mediante combinaciones lineales, decimos que C es *base* de D.
La reduccion gaussiana nos deja filas linealmente independientes. Las filas que quedan en 0 eran combinacion lineal de otras.
También nos deja columnas linealmente independientes: las que tienen pivot.

O sea que nos gustaría poder hacer Gauss, ya que por lo que dijimos antes, nos daría una base del conjunto original. Y lo bueno de una base es que sería mucho más pequeña: por ejemplo, tomar todas las potencias de 2, desde 2^0 hasta 2^{29}, es una base que genera todos los números menores a 2^{30}.

O sea que nos gustaría poder hacer Gauss, ya que por lo que dijimos antes, nos daría una base del conjunto original. Y lo bueno de una base es que sería mucho más pequeña: por ejemplo, tomar todas las potencias de 2, desde 2^0 hasta 2^{29}, es una base que genera todos los números menores a 2^{30}.

Vamos a buscar otra forma de construir la base xor: un conjunto de números que generen al conjunto original, que sean linealmente independientes, y que estén escalonados (lo que nos daría Gauss). Esto ahora significa que no haya 2 números con el mismo bit más significativo (a partir de ahora, msb).

O sea que nos gustaría poder hacer Gauss, ya que por lo que dijimos antes, nos daría una base del conjunto original. Y lo bueno de una base es que sería mucho más pequeña: por ejemplo, tomar todas las potencias de 2, desde 2^0 hasta 2^{29}, es una base que genera todos los números menores a 2^{30}.

Vamos a buscar otra forma de construir la base xor: un conjunto de números que generen al conjunto original, que sean linealmente independientes, y que estén escalonados (lo que nos daría Gauss). Esto ahora significa que no haya 2 números con el mismo bit más significativo (a partir de ahora, msb).

Vamos a mantener un conjunto que actualizamos iterativamente, y que al final será una base.

Sean A nuestro conjunto original y B el nuevo conjunto. Para cada número a de A, para ver si lo agregamos a B, tenemos que ver si es combinación lineal de los números actualmente en B.

Sean A nuestro conjunto original y B el nuevo conjunto. Para cada número a de A, para ver si lo agregamos a B, tenemos que ver si es combinación lineal de los números actualmente en B.
Notemos que para cada b número de B, podemos saber si tiene que estar en la combinación lineal que genera a a. Esto lo hacemos mirando en a el bit pivot de b. Este es importante porque solo está prendido en b.

Sean A nuestro conjunto original y B el nuevo conjunto. Para cada número a de A, para ver si lo agregamos a B, tenemos que ver si es combinación lineal de los números actualmente en B.
Notemos que para cada b número de B, podemos saber si tiene que estar en la combinación lineal que genera a a. Esto lo hacemos mirando en a el bit pivot de b. Este es importante porque solo está prendido en b.
Si este bit está prendido en a, entonces vamos a necesitar a b en nuestra combinación lineal. Si está apagado en a, forzosamente no podremos usar a b.

Lo que hacemos entonces es $a := a \oplus b$ para cada b en B que debamos usar en nuestra combinación lineal. Lo que nos queda es la información de a que no podemos representar como combinación lineal de B: si era 0, entonces es porque a era combinación lineal de B. Si es distinto de 0, tenemos que agregar este número a B.

Lo que hacemos entonces es $a := a \oplus b$ para cada b en B que debamos usar en nuestra combinación lineal. Lo que nos queda es la información de a que no podemos representar como combinación lineal de B: si era 0, entonces es porque a era combinación lineal de B. Si es distinto de 0, tenemos que agregar este número a B.
Y al agregarlo, tenemos que asegurarnos de que nuestro msb se convierta en pivot.
Esto es cuadrático, pero en el tamaño de la base. Es conocido que todas las bases tienen el mismo tamaño, así que nunca vamos a usar más de 30 elementos.

Ahora que conseguimos esta base, falta encontrar el mayor xor posible. Seguimos sin poder probar todos los subconjuntos.
Pero como nos tomamos el trabajo de mantenerla escalonada, no hace falta: el xor de todos será el máximo posible, puesto que todos prenden su msb, y en el peor de los casos apagan cosas menos significativas.

Resumen:

- Construir la base iterativamente.
- En cada paso, manetenerla escalonada.
- Hacer el xor de todas las filas.

Café

A Seba le gusta mucho el café, y por eso abre una paquete por día. Cada paquete puede traer, con probabilidad $0 < p < 1$, un sticker de Don Gato.

Seba guarda cada figurita por $n \leq 6$ días, y después se la regala a algún competidor de ICPC. Pero está esperando ansioso a llegar a tener algún día $k \leq n$ figuritas al mismo tiempo.

¿Cuál es la cantidad esperada de paquetes de café que Seba va a tener que abrir hasta conseguir las k figuritas?

Notemos que un día queda determinado por si en cada uno de los últimos n días salió o no una figurita.
Sea $abcdef$ una máscara de $n = 6$ dígitos. Que el dígito a valga 1 significa que salió una figurita hace 6 días. Que el dígito f valga 1 significa que salió una figurita el último día.

Notemos que un día queda determinado por si en cada uno de los últimos n días salió o no una figurita.
Sea $abcdef$ una máscara de $n = 6$ dígitos. Que el dígito a valga 1 significa que salió una figurita hace 6 días. Que el dígito f valga 1 significa que salió una figurita el último día.
Vamos a decir que E_{acbdef} es la cantidad esperada de tiradas, dado que en los últimos 6 días salieron o no figuritas según dice la máscara $abcdef$.
Por ejemplo, E_{000000} significa que en los últimos 6 días no salió ninguna figurita.

Podemos plantear la siguiente ecucación:

$$E_{acbdef} = p(1 + E_{bcdef1}) + (1 - p)(1 + E_{bcdef0})$$
$$E_{acbdef} = 1 + pE_{bcdef1} + (1 - p)E_{bcdef0}$$

Esto está diciendo que nos olvidamos del día más lejano (por eso del lado derecho no aparece a), y con probabilidad p nos sale una figurita: agregamos un 1 a la máscara.
Además, con probabilidad $1 - p$ no nos sale una figurita: agregamos un 0 a la máscara.

Podemos plantear la siguiente ecucación:

$$E_{acbdef} = p(1 + E_{bcdef1}) + (1 - p)(1 + E_{bcdef0})$$
$$E_{acbdef} = 1 + pE_{bcdef1} + (1 - p)E_{bcdef0}$$

Esto está diciendo que nos olvidamos del día más lejano (por eso del lado derecho no aparece a), y con probabilidad p nos sale una figurita: agregamos un 1 a la máscara.
Además, con probabilidad $1 - p$ no nos sale una figurita: agregamos un 0 a la máscara.
Si la máscara tiene k unos o más, entonces $E_{abcdef} = 0$.

Tenemos 2^n máscaras. Por cada una hay una ecuación como la anterior.
¡Es un sistema de $2^n \times 2^n$! Podemos tirarle Gauss y resolver.
La respuesta al problema es E_{000000}.

Resumen:

- Plantear la ecuación del valor esperado para cada máscara.
- Resolver con Gauss.

Dagedrez

Fede e Ivo juegan al *dagedrez*. Este se juega en un DAG con $n \leq 10^5$ vértices y $m \leq 10^5$ aristas.

Antes de arrancar, hay que poner fichas en el DAG usando un dado de n+1 caras.

Si al tirar el dado sale un número i entre 0 y $n-1$, ponen una ficha en el nodo i, y vuelven a tirar el dado.

Si sale el número n, entonces dejan de tirar el dado y comienzan a jugar.

En su turno, cada jugador puede mover una ficha en un nodo cualquiera por una arista que salga de ese nodo.

Pierde el jugador que no puede mover.

Ambos son muy rápidos para jugar, y lo hacen siempre siguiendo la mejor estrategia posible.

Fede, sabiendo que empieza, quiere encontrar la probabilidad de quedar en una posición ganadora al comenzar el juego (luego de tirar los dados).

Notemos que una vez que comienza, esto es un juego general, en DAG. La solución sería mirar el número de Grundy del juego.

Notemos que una vez que comienza, esto es un juego general, en DAG. La solución sería mirar el número de Grundy del juego.
Si no saben lo que es, pueden verlo en la sección de Teoría de Juegos del Competitive Programmer's Handbook.

Notemos que una vez que comienza, esto es un juego general, en DAG. La solución sería mirar el número de Grundy del juego.
Si no saben lo que es, pueden verlo en la sección de Teoría de Juegos del Competitive Programmer's Handbook.
Para cada grundy g, vamos a pensar en la probabilidad de ganar dado que el estado actual tiene grundy g, y todavía estoy tirando el dado. Llamemos $Q(g)$ a esta probabilidad.

Notemos que una vez que comienza, esto es un juego general, en DAG. La solución sería mirar el número de Grundy del juego.
Si no saben lo que es, pueden verlo en la sección de Teoría de Juegos del Competitive Programmer's Handbook.
Para cada grundy g, vamos a pensar en la probabilidad de ganar dado que el estado actual tiene grundy g, y todavía estoy tirando el dado. Llamemos $Q(g)$ a esta probabilidad. Queremos hallar $Q(0)$, porque inicialmente nuestro grundy es 0 (no hay fichas).

Notemos que una vez que comienza, esto es un juego general, en DAG. La solución sería mirar el número de Grundy del juego.
Si no saben lo que es, pueden verlo en la sección de Teoría de Juegos del Competitive Programmer's Handbook.
Para cada grundy g, vamos a pensar en la probabilidad de ganar dado que el estado actual tiene grundy g, y todavía estoy tirando el dado. Llamemos $Q(g)$ a esta probabilidad. Queremos hallar $Q(0)$, porque inicialmente nuestro grundy es 0 (no hay fichas).
Llamemos también $P(g \to h)$ a la probabilidad de pasar del grundy g al grundy h. Tenemos algo como

$$Q(g) = P(g \to 0) \cdot Q(0) + P(g \to 1) \cdot Q(1) + \dots$$

Notemos que una vez que comienza, esto es un juego general, en DAG. La solución sería mirar el número de Grundy del juego.
Si no saben lo que es, pueden verlo en la sección de Teoría de Juegos del Competitive Programmer's Handbook.
Para cada grundy g, vamos a pensar en la probabilidad de ganar dado que el estado actual tiene grundy g, y todavía estoy tirando el dado. Llamemos $Q(g)$ a esta probabilidad. Queremos hallar $Q(0)$, porque inicialmente nuestro grundy es 0 (no hay fichas).
Llamemos también $P(g \to h)$ a la probabilidad de pasar del grundy g al grundy h. Tenemos algo como

$$Q(g) = P(g \to 0) \cdot Q(0) + P(g \to 1) \cdot Q(1) + \dots$$

Si podemos calcular los coeficientes P, tendríamos una ecuación para cada grundy posible, con la misma cantidad de incógnitas: $Q(0), Q(1), \dots$

Fijemos un grundy g, y miremos todos los otros grundy a los que podríamos movernos:

Fijemos un grundy g, y miremos todos los otros grundy a los que podríamos movernos: si pongo una ficha en un nodo u con grundy h, voy a $g \oplus h$. Esto pasa con probabilidad $\frac{1}{n+1}$ para cada uno.

Fijemos un grundy g, y miremos todos los otros grundy a los que podríamos movernos: si pongo una ficha en un nodo u con grundy h, voy a $g \oplus h$. Esto pasa con probabilidad $\frac{1}{n+1}$ para cada uno.
O sea que podemos iterar por todos los nodos u, y acumular

$$P(g \to g \oplus \text{grundy}(u)) \ += \ \frac{1}{n+1}.$$

Fijemos un grundy g, y miremos todos los otros grundy a los que podríamos movernos: si pongo una ficha en un nodo u con grundy h, voy a $g \oplus h$. Esto pasa con probabilidad $\frac{1}{n+1}$ para cada uno.
O sea que podemos iterar por todos los nodos u, y acumular

$$P(g \to g \oplus \text{grundy}(u)) \ += \ \frac{1}{n+1}.$$

Y para $g \neq 0$, si sale n entonces empezamos, y ganamos. Así que sumamos $\frac{1}{n+1}$ de probabilidad a la ecuación de la slide anterior.

Fijemos un grundy g, y miremos todos los otros grundy a los que podríamos movernos: si pongo una ficha en un nodo u con grundy h, voy a $g \oplus h$. Esto pasa con probabilidad $\frac{1}{n+1}$ para cada uno.
O sea que podemos iterar por todos los nodos u, y acumular

$$P(g \to g \oplus \text{grundy}(u)) \mathrel{+}= \frac{1}{n+1}.$$

Y para $g \neq 0$, si sale n entonces empezamos, y ganamos. Así que sumamos $\frac{1}{n+1}$ de probabilidad a la ecuación de la slide anterior.
Ya sabemos que podemos resolver el sistema juntando todas estas ecuaciones en una matriz y corriendo Gauss, pero ¿cuántos grundys distintos hay?

Fijemos un grundy g, y miremos todos los otros grundy a los que podríamos movernos: si pongo una ficha en un nodo u con grundy h, voy a $g \oplus h$. Esto pasa con probabilidad $\frac{1}{n+1}$ para cada uno.
O sea que podemos iterar por todos los nodos u, y acumular

$$P(g \to g \oplus \text{grundy}(u)) \ += \ \frac{1}{n+1}.$$

Y para $g \neq 0$, si sale n entonces empezamos, y ganamos. Así que sumamos $\frac{1}{n+1}$ de probabilidad a la ecuación de la slide anterior.
Ya sabemos que podemos resolver el sistema juntando todas estas ecuaciones en una matriz y corriendo Gauss, pero ¿cuántos grundys distintos hay? Resulta que son a lo sumo $O\left(\sqrt{m}\right)$, ya que para que un nodo tenga grundy g, necesita al menos 1 arista que vaya de él a cada grundy $0 \leq h < g$.

Resumen:

- Calcular, con DFS, el número de Grundy de cada nodo.
- Para cada grundy g, pensar en la probabilidad $Q(g)$ de ganar dado que el estado actual tiene grundy g, y todavía se está tirando el dado.
- Calcular los coeficientes de estas ecuaciones y resolver con Gauss. La respuesta final es $Q(0)$.

:)