(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 819 888**

(51) Int. CI.:

H04L 9/32 (2006.01)
G06F 16/901 (2009.01)
G06Q 30/04 (2012.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **09.11.2016 E 16198040 (4)**

(97) Fecha y número de publicación de la concesión europea: **22.07.2020 EP 3321819**

(54) Título: **Dispositivo, método y programa para reducir de forma segura una cantidad de registros en una base de datos**

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
19.04.2021

(73) Titular/es:

INGENICO GROUP (100.0%)
28/32 Boulevard de Grenelle
75015 Paris, FR

(72) Inventor/es:

GERAUD, RÉMI y
NACCACHE, DAVID

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Dispositivo, método y programa para reducir de forma segura una cantidad de registros en una base de datos

1. Dominio

La invención se refiere a gestión de cadena de bloques. Una cadena de bloques es una base de datos distribuida que mantiene una lista en continuo crecimiento de registros llamados bloques. Estos bloques están protegidos contra manipulación y revisión mediante el uso de medios y recursos criptográficos. Cada bloque contiene una marca de tiempo y un enlace a un bloque anterior. Una cadena de bloques es, por ejemplo, un componente clave de la moneda digital bitcoin. Se utiliza como libro de contabilidad público para todas las transacciones.

2. Técnica anterior

En muchas aplicaciones de bases de datos, principalmente aquellas que realizan un seguimiento de los datos o el valor que se mueve, se almacena una lista de transacciones. Un ejemplo típico es la cadena de bloques de Bitcoin, donde se intercambian divisas entre cuentas y el historial (completo) de dichos intercambios constituye la propia cadena de bloques. A medida que pasa el tiempo, la cantidad de transacciones crece, al igual que los requisitos de almacenamiento: en 2016, la cadena de bloques de Bitcoin reclamó 90 GB (Gigabytes).

Si bien no es excesivo en términos de capacidad de almacenamiento según los estándares actuales, la cadena de bloques de Bitcoin aún ejerce una tensión considerable en la red, en particular cuando los usuarios necesitan acceder a este historial, aunque solo sea para buscar información en el mismo.

Incluso en entornos más tradicionales, tal como los libros de contabilidad de las grandes corporaciones financieras, es habitual realizar un seguimiento de las transacciones realizadas a lo largo del tiempo.

En cualquier caso, los requisitos de almacenamiento solo aumentarán. De hecho, una cadena de bloques se define vagamente como un libro de contabilidad público que contiene un historial de todas las transacciones entre usuarios. Estas transacciones se acumulan con el tiempo y constituyen una gran base de datos que necesita ser duplicada (para mantener el libro de contabilidad público) y verificada (para asegurar la validez de la transacción). Esto provoca una carga cada vez mayor en todas las operaciones. Para reducir el tamaño de la base de datos distribuida, el problema consiste en construir un libro de contabilidad (demostrablemente equivalente) más corto que el existente. Que la información nueva y la vieja coincidan debería ser fácil de verificar, y la cadena de bloques más corta ("purgada") facilita la duplicación y la verificación.

El problema de la contabilidad no es nuevo, pero solo se vuelve verdaderamente problemático en un contexto distribuido. En el entorno tradicional, el archivo completo es la solución de facto.

En un contexto distribuido, utilizando, por ejemplo, cadenas de bloques, se han realizado algunos esfuerzos para tratar de evitar la duplicación perpetua del historial, comenzando con la verificación rápida de transacciones basada en Merkle de Bitcoin. Con este esquema, es posible verificar que una transacción ha sido aceptada por la red descargando solo los encabezados de bloque correspondientes y el árbol de Merkle. Los nodos que no mantienen una cadena de bloques completa, denominados nodos de verificación de pago simplificada (SPV), utilizan las rutas de Merkle para verificar las transacciones sin descargar bloques completos.

Sin embargo, dado que los nodos de SPV no tienen la cadena de bloques completa, todavía debe haber otros nodos que realicen el archivo. Una idea que se describe a menudo consiste en excluir las transacciones antiguas: Descartar transacciones que ocurrieron antes de un límite de tiempo preestablecido, manteniendo solo las más recientes. El problema con este enfoque es que impide la auditabilidad, en la medida en que se puede perder el origen de algunas transacciones. También es incompatible con los protocolos existentes (tal como Bitcoin) y da como resultado un libro de contabilidad o criptomoneda alternativo.

El artículo científico "The Mini Blockchain Scheme" de J.D. Bruce (publicado en julio de 2014) se considera estado de la técnica relevante, ya que divulga una solución para reducir la cantidad de espacio necesario para almacenar una cadena de bloques.

Para preservar el uso eficiente de cadenas de bloques, existe la necesidad de proporcionar una técnica que permita reducir el tamaño de la base de datos distribuida mientras se preservan las propiedades de seguridad e inviolabilidad de la base de datos reducida.

3. Sumario

En la presente divulgación, se propone un método para reducir el tamaño de las bases de datos, eliminando los registros innecesarios. La técnica propuesta permite reducir los inconvenientes de la técnica anterior. Más específicamente, la técnica propuesta no necesita cálculos extensos. Adicionalmente, la técnica propuesta permite obtener una base de datos más pequeña y reducir drásticamente la transmisión de información, especialmente en el caso de bases de datos distribuidas. Más específicamente, se propone un método para reducir una cantidad de registros en una base de datos, siendo implementado dicho método por un aparato electrónico que comprende

recursos de hardware para reducir la cantidad de registros en bases de datos. De acuerdo con la divulgación, dicha base de datos está representada en la forma de un multigrafo orientado en el que una arista valorada desde un vértice de origen a un vértice de destino representa un registro de dicha base de datos, y el método comprende:

- al menos una etapa para obtener, en dicho grafo, al menos un conjunto de vértices y aristas que forman un grupo de un saldo dado para cada vértice de dicho conjunto, denominado conjunto de concatenaciones;

- al menos una etapa para eliminar dicho conjunto de concatenaciones de dicho multigrafo orientado.

De acuerdo con la divulgación, el valor de dicho saldo es igual a 0.

De acuerdo con la divulgación, dicha etapa para obtener dicho conjunto de concatenaciones comprende:

- una etapa para obtener, a partir de dicho multigrafo, al menos un subgrafo que comprende al menos un vértice y al menos una arista valorada;

- al menos una etapa para borrar, dentro del subgrafo del conjunto, al menos un vértice que está ausente de un grupo de vértices para el cual el saldo de los valores de cada vértice es nulo, entregando un subconjunto de concatenaciones.

De acuerdo con la divulgación, dicha etapa para obtener al menos un subgrafo comprende una etapa para determinar, dentro de dicho multigrafo orientado, un conjunto de subgrafos $\{G_1, ..., G_l\}$, comprendiendo cada uno un conjunto de vértices fuertemente conectados.

De acuerdo con la divulgación, dicha etapa para determinar dicho conjunto de subgrafos $\{G1, ..., Gl\}$ comprende la implementación de un módulo de Tarjan.

De acuerdo con la divulgación, dicha etapa para eliminar, dentro de dicho subgrafo, al menos un vértice, comprende, para dicho al menos un vértice:

- cuando el número de aristas de entrada o el número de aristas de salida de dicho vértice es igual a cero, eliminar dicho vértice;

- cuando el número de aristas de salida y el número de aristas de entrada es igual a uno y el valor de la arista de salida única es diferente del valor de la arista de entrada, eliminar dicho vértice;

De acuerdo con la divulgación, dicha etapa para obtener, a partir de dicho multigrafo G, dicho conjunto de concatenaciones $\tilde{G}$ comprende, antes de la obtención de dicha etapa de concatenación:

- determinar, a partir de dicho multigrafo G, un registro inicial S_R;

- a partir de dicho registro inicial S_R, generar un subgrafo SG;

- usar dicho subgrafo SG para obtener dicho conjunto de concatenaciones $\tilde{G}$;

De acuerdo con la divulgación, dicha etapa para generar dicho subgrafo SG a partir de dicho registro S_R comprende obtener al menos un nodo y derivar dicho subgrafo de dicho al menos un nodo.

De acuerdo con la divulgación, el método comprende al menos una etapa para determinar, basándose en dicho conjunto de concatenaciones, al menos un dato que representa una recompensa.

De acuerdo con la divulgación, dicha etapa para determinar al menos un dato que representa una recompensa comprende determinar una tarifa de transacción de dicho conjunto de concatenaciones.

La invención también se refiere a un dispositivo electrónico para reducir una cantidad de registros en una base de datos, comprendiendo dicho dispositivo recursos de hardware para reducir la cantidad de registros en bases de datos, caracterizado por que dicho dispositivo comprende medios para registrar dicha base de datos, en una memoria, en forma de un multigrafo orientado en el que un borde valorado a partir de un vértice de origen a un vértice de destino representa un registro de dicha base de datos, caracterizado por que dicho dispositivo comprende además:

- medios para obtener, en dicho grafo, al menos un conjunto de vértices y aristas que forman un conjunto de un saldo dado para cada vértice de dicho conjunto, denominado conjunto de concatenaciones;

- medios para eliminar dicho conjunto de concatenaciones de dicho multigrafo orientado.

También se describe un medio legible por procesador no transitorio que tiene almacenada en el mismo una base de datos reducida de este tipo.

De acuerdo con una implementación, las diferentes etapas del método para reducir el tamaño de una base de datos

como se describe aquí anteriormente son implementados por uno o más programas de software o programas de módulo de software que comprenden instrucciones de software destinadas a ser ejecutadas por un procesador de datos de un aparato para reducir la tamaño de una base de datos, estando diseñadas estas instrucciones de software para comandar la ejecución de las diferentes etapas de los métodos de acuerdo con los presentes principios.

También se describe un programa informático que puede ser ejecutado por un ordenador o por un procesador de datos, comprendiendo este programa instrucciones para ordenar la ejecución de las etapas de un método para reducir el tamaño de una base de datos como se mencionó anteriormente.

Este programa puede utilizar cualquier lenguaje de programación y estar en forma de código fuente, código objeto o código intermedio entre el código fuente y el código objeto, tal como en una forma parcialmente compilada o en cualquier otra forma deseable.

El soporte de información puede ser cualquier entidad o aparato capaz de almacenar el programa. Por ejemplo, el soporte puede comprender un medio de almacenamiento tal como una ROM, por ejemplo, un CD ROM o una ROM de circuito microelectrónico o un medio de grabación magnética, por ejemplo, un disquete o una unidad de disco duro.

De nuevo, el soporte de información puede ser un soporte transmisible, tal como una señal eléctrica u óptica, que se puede transmitir mediante un cable eléctrico u óptico, por radio o por otros medios. El programa de acuerdo con los presentes principios se puede cargar especialmente en una red de tipo Internet.

Como una alternativa, el soporte de información puede ser un circuito integrado en el que se incorpora el programa, estando el circuito adaptado para ejecutar o para ser utilizado en la ejecución de los métodos en cuestión.

De acuerdo con una realización, los métodos/aparatos pueden implementarse mediante componentes de software y/o hardware. A este respecto, el término "módulo" o "unidad" puede corresponder en el presente documento igualmente bien a un componente de software y a un componente de hardware o a un conjunto de componentes de hardware y software.

Un componente de software corresponde a uno o más programas de ordenador, uno o más subprogramas de un programa o, más generalmente, a cualquier elemento de un programa o una pieza de software capaz de implementar una función o un conjunto de funciones como se describe a continuación para el módulo correspondiente. Dicho componente de software es ejecutado por un procesador de datos de una entidad física (terminal, servidor, etc.) y es capaz de acceder a los recursos de hardware de esta entidad física (memorias, medios de grabación, buses de comunicaciones, placas electrónicas de entrada/salida, interfaces de usuario , etc.).

De la misma manera, un componente de hardware corresponde a cualquier elemento de una unidad de hardware capaz de implementar una función o un conjunto de funciones como se describe a continuación para el módulo en cuestión. Puede ser un componente de hardware programable o un componente con un procesador integrado para la ejecución de software, por ejemplo, un circuito integrado, una tarjeta inteligente, una tarjeta de memoria, una placa electrónica para la ejecución de firmware, etc.

4. Breve descripción de los dibujos

- La figura 1 ilustra la concatenación de acuerdo con la presente técnica;

- La figura 2 es un diagrama de bloques esquemático que ilustra el método para reducir el tamaño de la base de datos;

- La figura 3 divulga grafos en los que no se presentan obstrucciones de primer orden;

- La figura 4 divulga la concatenación de tres concatenaciones independientes;

- La figura 5 divulga una realización simplificada de un dispositivo de acuerdo con la presente divulgación.

5. Descripción de una realización

5.1. Principios

Como se explicó anteriormente, uno de los propósitos de la técnica propuesta es reducir el tamaño de la base de datos, que se utiliza, por ejemplo, en una solución basada en cadena de bloques. La aplicación de la técnica propuesta, sin embargo, no se limita a bases de datos de cadena de bloques, sino que puede implementarse en cualquier base de datos que comprenda el almacenamiento de registros que rastrean la transmisión de una cantidad de algo entre entidades. El uso de las presentes técnicas permite reducir el crecimiento de los requisitos de almacenamiento. Para lograr este resultado, los inventores pensaron en cómo la información puede almacenarse o limpiarse de manera eficiente, es decir, representarse. Tal representación debería preservarse semánticamente, al menos en la medida en que el efecto de ninguna transferencia (transacción) se pierda en el proceso (que es muy

diferente de "que no se pierda ninguna transacción en el proceso"). En muchos casos, algunos detalles pueden volverse irrelevantes (por ejemplo, el número preciso de transferencias o transacciones sucesivas entre dos partes o dos entidades) y es posible agrupar algunos eventos en una forma más compacta.

Una parte clave de la presente técnica radica en la representación de la información. Los inventores investigaron la forma en que la cadena de bloques, y más generalmente las transferencias sucesivas, pueden ser representadas para reducir el tamaño de la representación. De acuerdo con la técnica propuesta, la representación de la cadena de bloques es un multigrafo orientado. Se muestra, a continuación, que esta representación es matemáticamente correcta y que, considerando esta representación, es posible implementar un método de reducción de tamaño. Este método de reducción de tamaño ofrece un multigrafo orientado reducido, donde se eliminan los intercambios con un saldo nulo.

Dado su libro de contabilidad y su descripción disponibles públicamente, la red de Bitcoin es un estudio de caso concreto del presente documento. Sin embargo, la técnica también se puede aplicar a instituciones tales como bancos o similares. En el caso de bitcoin, cada cliente compatible puede conectarse a la red, enviarle nuevas transacciones, verificar transacciones y participar en la competencia para crear nuevos bloques. Considerando el método de gestión de transacciones en una cadena de bloques, los inventores tuvieron la intuición de que una representación adecuada de las transacciones podría llevar a aplicar categorías de métodos que no se han aplicado para reducir el tamaño de la cadena de bloques.

Para simplificar, se considera un conjunto de usuarios (cada uno poseyendo una cuenta para simplificar), y transacciones entre estos usuarios, que se pueden representar como aristas etiquetadas entre nodos. Por lo tanto, el libro de contabilidad de transacciones consiste en un multigrafo (ya que las aristas se pueden repetir). Nuestro objetivo es identificar información redundante, en forma de un subgrafo que se puede eliminar sin afectar el saldo de ningún usuario (la semántica de la red). Este subgrafo se llama concatenación, de acuerdo con la técnica propuesta. La ***figura 1*** expone los principios de la concatenación. El "problema de concatenación" (NCP) de cadena de bloques consiste en construir un libro de contabilidad (demostrablemente equivalente) que sea más corto que el existente. Que la información nueva y la vieja coincidan debería ser fácil de verificar, y la cadena de bloques más corta ("purgada") facilita la duplicación y la verificación.

La ***figura 2*** expone las etapas generales implementadas en la técnica propuesta. Más específicamente, se propone un método para reducir una cantidad de registros en una base de datos (DB), siendo implementado el método por un aparato electrónico que comprende recursos de hardware para reducir la cantidad de registros en bases de datos. De acuerdo con la técnica propuesta, la base de datos (DB) se representa en forma de un multigrafo orientado (G) en el que una arista valorada desde un vértice de origen a un vértice de destino representa un registro de la base de datos; de acuerdo con la técnica propuesta el método comprende:

- al menos una etapa para obtener (10), en el grafo (G), al menos un conjunto de vértices y aristas que forman un grupo de un saldo dado (b) para cada vértice del conjunto, denominado conjunto de concatenaciones ($\tilde{G}$);

- al menos una etapa para eliminar (20) el conjunto de concatenaciones ($\tilde{G}$) del multigrafo orientado (G).

Generalmente, el valor del saldo (b) es igual a cero. De esta forma, es posible identificar los grupos de registros que se cancelan a sí mismos, que son grupos de registros para los que el saldo es nulo entre un número determinado de usuarios (en un grupo). En el caso de los bitcoins, por ejemplo, este sería un número dado de transacciones entre varios usuarios, formando un grupo, para lo cual, desde el punto de vista de cada usuario del grupo, el saldo de los valores de transacción es igual a cero, lo que significa que cada usuario del grupo ha dado tanto como lo que ha recibido. Los grupos pueden ser, por ejemplo, ciclos (bucle) y/o camarillas de registros, estando estos registros fuertemente conectados entre sí.

De acuerdo con la presente técnica, la obtención del conjunto de vértices y aristas que forman un grupo se puede realizar de varias formas distintas. Sin embargo, una buena forma de realizar esta operación consiste en agrupar los registros obteniendo un subgrafo del multigrafo.

Más específicamente, en al menos una realización, la etapa para obtener el conjunto de concatenaciones ($\tilde{G}$) comprende:

- Una etapa para obtener (110), a partir del multigrafo (G) al menos un subgrafo (G_k) que comprende al menos un vértice (v_{k1}, ..., v_{kn}) y al menos una arista valorada (e_{k1},...,e_{kn});

- al menos una etapa para borrar (120), dentro del subgrafo del conjunto (G_k), al menos un vértice (v_{kx}) que está ausente de un grupo (L) de vértices para los cuales el saldo de los valores de cada vértice es nulo, entregando un subconjunto de concatenaciones ($\tilde{G}_k$).

Gracias a la implementación de este método, se volvió fácil reducir el grafo y obtener un grafo simplificado (tal como el ejemplo presentado en la figura 1)

Como se expondrá más adelante, resolver un problema de concatenación de cadena de bloques es un servicio para

la comunidad que puede ser recompensado con ... monedas, al igual que cualquier otra prueba de trabajo. En ese sentido, la perspectiva de una criptomoneda que permite a los usuarios extraer mediante concatenación es completamente implementable. Purgar transacciones pasadas mientras se preserva la semántica de la red también tiene posibles implicaciones de privacidad. Al eliminar algunas transacciones, no se crea ninguna mentira sobre el pasado (es decir, se crean transacciones virtuales que no existen) mientras se olvidan las transacciones que no necesitan ser recordadas.

En el presente, primero se propone formalizar el problema de la concatenación de cadena de bloques y ubicarlo en el contexto de transacciones financieras representables a través de multigrafos ponderados. La formulación y las pruebas correctas de esta formulación forman la base de la técnica propuesta ya que es porque el problema está bien planteado y comprobado que la solución técnica funciona. Más específicamente, se muestra, a través de una reducción al problema de suma de subconjuntos (multidimensional), ese NCP es **NP** completo.

Sobre esta base, se propone un método para encontrar la concatenación óptima cuando el problema de suma de subconjuntos subyacente tiene una densidad baja, lo que en la práctica es casi siempre el caso. El enfoque se basa en una combinación de técnicas grafo teórico y de reducción de retícula.

Dado que comprobar la validez de una solución para el NCP es fácil, como contribución separada, se propone utilizar las soluciones de NCP como pruebas de trabajo. Se presentan modelos de recompensa y también se exponen las precauciones prácticas necesarias para ajustar correctamente el incentivo resultante.

También se analizan las estrategias de trampa y se proporcionan contramedidas. La primera contramedida que se propone se adapta a los modelos de cadenas de bloques considerando las tarifas de transacción. Se ofrece una segunda solución (junto con una prueba en el modelo Oracle aleatorio) para modelos de cadena de bloques sin tarifas de transacción.

5.2. Descripción del problema de concatenación de cadena de bloques

5.2.1. anotaciones

[*n*] indica el conjunto {1, ..., n}. Para un conjunto S, se indica por $s \leftarrow_{\$} S$ la acción del muestreo *s* uniformemente al azar. |*S*| representa la cardinalidad de un conjunto. Letras en negrita (por ejemplo, $\vec{\boldsymbol{v}}, \vec{\boldsymbol{A}}$) se utilizan para representar vectores y matrices.

5.2.2. Grafos y multigrafos

Se utilizan las siguientes definiciones estándar: Un *grafo G* = (*V, E*) son los datos de un conjunto finito *V* y $E \subseteq V \times V$, llamados respectivamente los vértices y las aristas de *G*. Una secuencia de aristas $(s_1, t_1), ..., (s_n, t_n)$ tal que $t_i = s_{i+1}$ para todo $1 \leq i < n$ se llama una *trayectoria dirigida* desde s_1, a t_n. El *grado* de un vértice $v \in V$ es el número de aristas conectadas a *v*. El *grado de entrada* (resp. *grado de salida*) de *v* es el número de aristas que terminan en (resp. comenzando en) *v*.

Se puede dar la siguiente definición: Si *G* = (*V, E*) es un grafo, entonces un *componente fuertemente conectado* (o *SCC*) de *G* es un subgrafo máximo de *G* tal que para cada dos nodos distintos *x* e y, existe una trayectoria dirigida desde *x* a *y*, **y** una trayectoria dirigida desde *y* a *x*.

En particular, cualquier componente fuertemente conectado está conectado, pero no ocurre lo contrario.

Aquí, se considera una extensión de grafos donde las aristas se pueden repetir y se etiquetan: Un *multigrafo etiquetado* se indica por *G* = (*V, E, φ*) donde ahora *E* es un conjunto múltiple de parejas desde $V \times V$ (no solo un conjunto) y: $E \rightarrow Z$ da la etiqueta asociada a cada arista. Uno usará la siguiente notación: Si $e = (a, b) \in E$ y $r = \varphi(e)$, y representan la arista *e* escribiendo $a \xrightarrow{r} b$.

Las definiciones de componentes conectados y fuertemente conectados, y la definición de grado, se extienden naturalmente a los multigrafos.

5.2.3. El problema de la suma de subconjuntos

Se recuerda el conocido problema de suma de subconjuntos (SSP):

Dado un conjunto finito $A \subset \mathbb{Z}$ y un valor objetivo $t \in \mathbb{Z}$, encontrar un subconjunto $S \subseteq A$ tal que:

$$\sum_{s \in S} s = t.$$

Se sabe que el SSP es NP completo. Una definición equivalente del problema viene dada por lo siguiente:

Dado un vector $A \subset \mathbb{Z}^n$, y un valor objetivo $t \in \mathbb{Z}$, encontrar un vector $\epsilon \in \{0,1\}^n$ tal que

$$< A, \epsilon > = t,$$

donde <·, ·> indica el producto interior.

La densidad de una instancia de SSP particular se define como:

$$d = \frac{n}{\max_{a \in A} \log a}$$

Si bien las instancias de SSP genéricas son difíciles de resolver, las instancias de baja densidad se pueden resolver de manera eficiente mediante técnicas de aproximación o reducción de retícula.

5.2.4. Formalización del NCP

5.2.4.1. Una primera definición

En todo lo que sigue, se supone que el historial de transacciones forma un multigrafo G = (V, E, φ), donde los vértices V corresponden a entidades (por ejemplo, cuentas de un banco o similares, pero también otro tipo de entidad gestora) y una arista etiquetada $a \xrightarrow{u} b$ corresponde a una transferencia desde el vértice *a* al vértice *b* de una cantidad *u* (por ejemplo una cantidad *u* en bitcoins, pero también puede ser otro tipo de productos o servicios).

El saldo *b(v)* de un vértice individual (por ejemplo, entidad) *v* viene dada por la diferencia entre la transferencia entrante y la transferencia saliente, es decir:

$$b(v) = \sum_{e:\to v} \phi(e) - \sum_{f:v\to \cdot} \phi(f),$$

donde (· → *v*) indica todas las aristas de entrada, es decir, todos los elementos en E de la forma (w, v) para algunos *w* ∈ *V*; de manera similar (*v* → ·) indica todas las aristas de salida. Si b(G) indica el vector {b(v), vEV}, que es la semántica del grafo.

Dado un multigrafo *G* = (*V, E, φ*), encuentra el máximo $\tilde{E} \subseteq E$ tal que *b*(G) = *b*($\tilde{G}$), donde $\tilde{G}$ = (*V, Ẽ, φ*). ($\tilde{G}$, *G* - $\tilde{G}$) es la concatenación de G.

En otros términos, encontrar una concatenación consiste en encontrar aristas que se puedan eliminar sin afectar el saldo de nadie, es decir, que conserven la semántica del grafo. No se permite fusionar transacciones.

5.2.4.2. NCP y SSP

Una idea clave para resolver este problema es darse cuenta de que una forma similar de afirmar que las aristas en Ẽ no contribuyen al saldo de ninguna v, es que contribuyen con una cantidad de cero. En otros términos, en cada vértice v el saldo en $\tilde{G}$ es $\tilde{b}$ (v) = 0. Esto da la siguiente descripción:

[def: ncp] Sea G = (V, E, φ) un multigrafo. Escribe V = {v_1... , v_n} y representan una arista e: $v_i \xrightarrow{r} v_j$ como el vector $re_{ij} \in \mathbb{Z}^n$ donde e_{ij} es el vector de $\mathbb{Z}^n$ con 1 en la posición j, -1 en la posición i y 0 en los componentes restantes. Ahora E se entiende como una lista de m de tales vectores E = (e_1 , ..., e_m). El problema de concatenación consiste en encontrar $\epsilon \in \{0,1\}^m$ tal que

$$\sum_{i=1}^{m} \epsilon_i e_i = \langle E, \epsilon \rangle = 0,$$

donde la notación ⟨·, ·⟩ se extiende de la manera obvia. La concatenación de G se define entonces como (G - $\tilde{G}$, $\tilde{G}$), donde $\tilde{G}$ = (V, Ẽ, φ) y Ẽ = {$e_i \in E$, ϵ_i = 1}.

Esta definición aclara el paralelismo entre el problema de concatenación y la versión multidimensional del problema de suma de subconjuntos, como se describe en: Para n = 2, NCP y SSP son exactamente el mismo problema.

De hecho, más es cierto: NCP puede verse como una variante multidimensional del problema de la suma de subconjuntos, donde las entradas pertenecen a $\mathbb{Z}^{|V|}$ en vez de a $\mathbb{Z}^n$. Debe tenerse en cuenta, sin embargo, que *NCP* es un caso especial notablemente escaso de ese SSP multidimensional.

NCP es equivalente a SSP y, por lo tanto, NP completo.

De acuerdo con la discusión anterior, un oráculo SSP multidimensional proporciona una solución a cualquier ejemplo de *NCP*. Vectores de $\mathbb{Z}^{|V|}$ se pueden describir como números enteros usando codificación de base |V|. Por tanto, SSP y SSP multidimensional son equivalentes. Por lo tanto, tenemos una reducción de SSP a *NCP.*

Por el contrario, asumiendo un oráculo NCP, y en particular un oráculo n = 2, entonces, de acuerdo con la observación anterior, es exactamente un oráculo SSP.

5.2.4.3. Descripción de una primera realización: Resolver una instancia NCP genérica

Siguiendo la observación anterior, uno puede tener la tentación de aprovechar las técnicas de resolución *SSP* para abordar el *NCP.* Sin embargo, la reducción de *NCP* a *SSP* no es necesariamente interesante desde un punto de vista computacional para grandes bases de datos: los coeficientes se vuelven muy grandes, del orden de Bb^n, donde *B* es el límite superior de la representación de *E,* y b es la base elegida. Esta codificación se puede mejorar un poco si los límites $B_i^{\pm}$ para cada columna son conocidos, porque se pueden utilizar mejores representaciones.

Sin embargo, en la práctica puede volverse prohibitivo rápidamente; e incluso forzando el original *NCP* puede ser menos exigente computacionalmente: el problema de la suma de subconjuntos se puede resolver exactamente (clásicamente) en el peor de los casos

$$\mathcal{O}(2^m)$$

forzando todas las combinaciones, e incluso los algoritmos de última generación solo tienen una complejidad marginalmente mejor, es decir

$$\mathcal{O}(2^{m\cdot 0.291\ldots}).$$

Si uno desea abordar el *NCP* directamente, para n > 2, los enfoques basados en encuentro en el medio no se aplican, ya que en ese caso no hay un orden total en $\mathbb{Z}^n$. En su lugar, uno aprovechará, de acuerdo con la técnica actual, el algoritmo de reducción de retícula *LLL*. Dado como entrada una base reticular de dimensión d entera cuyos vectores tienen una norma menor que *B, LLL* produce una base reducida en el tiempo

$$\mathcal{O}(d^2 n(d+\log B)\log Bf),$$

donde *f* representa el costo de la multiplicación de d bits.

Para ver por qué la reducción de retícula resolvería el problema, primero debe tenerse en cuenta que *E* se puede representar como una matriz *n* × *m* con coeficientes racionales (o enteros). Es una matriz dispersa, que tiene (como máximo) dos entradas distintas de cero por columna, es decir (como máximo) *2m* entradas distintas de cero de *nm.* Si i_n es la matriz de identidad *n* × *n* y si **ε** = (i_n|*E*) es el resultado de concatenar los dos bloques: **ε** es una matriz *n x (n+m)*, teniendo como máximo *n+2m* elementos distintos de cero de *n(n+m).*

Ahora bien, si hay una solución al *NCP,* entonces *(0,...,0)* pertenece a la retícula generada por *E.* En particular, este es un vector corto: Si este es el vector más corto, entonces *LLL* lo encontrará con abrumadora probabilidad. La cuestión de resolver el *NCP* a partir de una solución al problema del vector más corto *(SVP)* depende de la densidad, de la topología y de la distribución probabilística de ponderaciones del multigrafo. A continuación, se elabora una prueba de la optimalidad para algunas familias de grafos, y en los casos en que no se garantiza la optimalidad, el resultado suele ser cercano al óptimo.

En la práctica, sin embargo, usar esta técnica directamente no es práctico. La principal razón es que la complejidad de *LLL* en un grafo grande está dominada por m^3, y grandes bases de datos (especialmente grandes libros de contabilidad para el encadenamiento de bloques) manejan muchas transacciones, siendo m del orden de 10^8 por día.

5.3. Descripción de una segunda realización: Resolución de NCP más rápida

Si bien el enfoque de reducción de retícula discutido anteriormente no necesariamente se puede aplicar de manera eficiente directamente en un gran multigrafo para encontrar una solución al *NCP,* puede trabajar en pequeños multigrafos. De acuerdo con la presente técnica, se describe, en esta sección, un mecanismo de corte que reduce drásticamente el tamaño del problema. Este algoritmo analiza la instancia de *NCP* en subinstancias *NCP* mucho más pequeñas, que pueden ser abordadas por *LLL.* Además, cada instancia puede tratarse de forma independiente, lo que hace que nuestra heurística sea paralelizable.

En otros términos, si bien uno podría convertir una instancia *NCP* en una instancia *SSP* y tratar de abordar la instancia *SSP* con las técnicas existentes, primero se aprovecha la forma particular de tales problemas, es decir, las propiedades relacionadas con los grafos, para reducir el tamaño del problema. Es posible reducir el tamaño del

problema gracias a las dos observaciones siguientes:

En primer lugar, solo es necesario considerar componentes fuertemente conectados. De hecho, si $v, w \in V$ pertenecen a dos componentes diferentes fuertemente conectados de G, entonces, por definición, no hay trayectoria que vaya de v a w y viceversa. Por lo tanto, no se puede devolver ninguna cantidad tomada de v, por lo que el saldo de v no se puede conservar. Por lo tanto, todas las aristas de $\tilde{E}$ están contenidas en un componente fuertemente conectado de G.

En segundo lugar, si *H* es una concatenación de *G*. Entonces *H* debe satisfacer una propiedad de "conservación de flujo local": el flujo a través de cualquier corte de *H* es cero; de manera equivalente, la entrada de cada vértice equivale a la salida. Los subgrafos que no satisfacen esta propiedad se denominan obstrucciones y pueden eliminarse de forma segura.

Un vértice $v \in V$ es una obstrucción de primer orden si se cumple lo siguiente:

- el grado de entrada y de salida de *v* son ambos iguales a 1;

- las etiquetas de la arista de entrada y de salida son desiguales.

En consecuencia, se pueden definir obstrucciones de "orden cero", donde el mínimo del grado de entrada y de salida de v es cero (pero tales vértices no existen en un componente fuertemente conectado), y obstrucciones de orden superior, donde el grado de entrada o de salida de *v* es mayor que 1, y no hay solución para la conservación local de *SSP:*

Si $v \in V$, si E_I es el conjunto múltiple de aristas de entrada, y E_O el conjunto múltiple de aristas de salida. El SSP de conservación local es el problema de encontrar $S_I \subseteq E_I$, $S_O \subseteq E_O$ tal que:

$$\sum_{e \in S_I} \phi(e) = \sum_{f \in S_O} \phi(f)$$

Este problema es exactamente el SSP en el conjunto $E_I \sqcup E_O$, y el valor objetivo 0, de ahí el nombre.

5.3.1. Componentes fuertemente conectados

Es fácil ver que una partición de *G* en *k* componentes fuertemente conectados corresponde a una partición de *E* en *(k+1)* conjuntos múltiples: Cada componente fuertemente conectado con sus aristas y un resto de aristas que no pertenecen a *SCC.* Como se explicó anteriormente, este resto no pertenece a $\tilde{E}$.

En esta realización, la partición de un grafo en componentes fuertemente conectados se puede determinar exactamente en tiempo lineal usando, por ejemplo, el algoritmo de Tarjan, implementado en un módulo llamado Tarjan. A cada componente se le asocia un descriptor (por ejemplo, un vector binario que define un subconjunto de *E*) y procesarlos en paralelo o secuencialmente, de forma independiente. También se pueden aplicar otras técnicas o algoritmos, tal como, por ejemplo, el algoritmo de Kosaraju o el algoritmo de PBSC (componente fuerte basado en la trayectoria).

Esto corresponde a reordenar *V* de manera que *E* es una matriz diagonal de bloques y trabaja en cada bloque de forma independiente.

Si bien las obstrucciones de primer orden son fáciles de caracterizar y pueden detectarse en tiempo lineal, el caso de las obstrucciones de orden superior es más desafiante. La detección de obstrucciones de orden superior requiere resolver varias (pequeñas) instancias *SSP*, que tiene un costo adicional. Como recompensa, se pueden eliminar más aristas (se describe un ejemplo en la figura 3: no hay obstrucciones de primer orden en estos grafos, que están fuertemente conectados y cuyas aristas están todas etiquetados de manera idéntica. Dichos "concentradores" se pueden detectar detectando obstrucciones de orden superior, a costa de resolver $(n + 1)$ veces una instancia de SSP de elemento *n*, donde $n = 5$ (izquierda) o 6 (derecha)).

Resolver repetidamente el *SSP* probablemente no sea óptimo, ya que hay mucho trabajo redundante.

Cualquiera que sea el método que se utilice, es interesante saber si la recompensa de detectar obstrucciones de orden superior merece el esfuerzo. Esta pregunta se detalla en la siguiente subsección. Mientras tanto, se observa que dado que detectar una obstrucción de grado n es equivalente a resolver una instancia multidimensional *SSP* con un máximo de 2n elementos.

5.3.2. El algoritmo de corte

Ahora se puede presentar el algoritmo de corte que aprovecha las observaciones de esta sección:

Datos: Multigrafo $G = (V, E, \phi)$

Resultado: Multigrafos $\{G_i = V_i, E_i, \phi_i)\}$, que ni tienen obstrucción simple

Función: Corte (G):

$\{G_1, \ldots, G\} \leftarrow$ Tarjan (G)

foreach $G_k = (V_k, E_k, \phi_k)$ **do**

foreach v 2 V_k **do**

if min (d_{+v}, d_{-v}) = 0 **then**

eliminar todas las aristas conectadas a v en E_i

else if $d_{+v} = d_{-v} = 1$ **then**

indicar e_{entrada} el vértice de entrada

indicar e_{salida} el vértice de salida

if ϕ_k ($e_{entrada}$) ≠ ϕ_k (e_{salida}) **then**

borrar $e_{entrada}$ y e_{salida} de E_k

end

end

end

end

return $\{G_1, \ldots, G\}$

end

Este algoritmo: (1) descompone el grafo en su *SCC;* luego (2) elimina las obstrucciones (de primer orden) en cada componente. La eliminación de obstrucciones puede dividir un componente fuertemente conectado en dos (se puede realizar un seguimiento de esto utilizando una estructura de datos de refinamiento de partición), por lo que se pueden repetir las etapas (1) y (2) hasta la convergencia, es decir, hasta que no se encuentre ninguna obstrucción o no se identifique ningún nuevo SCC. Esto da un algoritmo recursivo de *Corte recursivo.*

Análisis de complejidad. La complejidad de este algoritmo depende a priori del grafo que se considere y, en particular, de cuántos *SCC* uno puede esperar, cuán probable es que una obstrucción cree nuevos *SCC,* qué tan frecuentes sean las obstrucciones, etc. Si uno pone nuestra atención en el peor comportamiento del caso, de hecho, se puede considerar el multigrafo para el cual este algoritmo tomaría más tiempo en ejecutarse.

El algoritmo de Tarjan tiene complejidad temporal

$$\mathcal{O}(n+m),$$

y la eliminación de obstrucciones de primer orden tiene una complejidad de tiempo $\mathcal{O}(n)$. Así, la complejidad de el corte completo está determinada por el número de iteraciones hasta la convergencia. Por lo tanto, el peor grafo tendría una obstrucción, que al eliminarse divide su *SCC* en dos; cada *sub-SCC* tendría una obstrucción, que al ser eliminada divide el *sub-SCC* en dos, etc. Suponiendo que este comportamiento se mantenga hasta el final, hasta que solo queden nodos aislados, se ve que no puede haber más de $log_2 n$ iteraciones.

Cada iteración crea dos instancias de NCP, teniendo cada una n/2 vértices y alrededor de m/2 aristas. Por lo tanto, el algoritmo de corte completo tiene complejidad

$$\mathcal{O}((n+m)logn).$$

Debe tenerse en cuenta que, de hecho, se puede trabajar en cada subproblema de forma independiente.

Si ahora se extiende el algoritmo de corte para detectar también obstrucciones de orden superior, digamos hasta un orden fijo d, entonces la etapa de eliminación de obstrucciones cuesta

$$\mathcal{O}\left(2^d n\right) = \mathcal{O}(n)$$

ya que 2^d es una constante. Por lo tanto, la complejidad asintótica del peor de los casos no se ve afectada. Sin embargo, en la práctica, el término constante podría ser un factor limitante, especialmente porque la obstrucción de orden superior puede ser rara. Hacer de esto una declaración precisa requiere un modelo de multigrafos aleatorios. Para compensar el costo adicional de detectarlos, las obstrucciones de *orden-d* deben ser lo suficientemente frecuentes: Una conjetura, para una realización general, que este no es el caso, y que no hay ganancia en ir más allá del primer orden.

5.3.3. Resolución NCP rápida

Ahora se puede describir en su totalidad el algoritmo de resolución *NCP* rápida. Consiste en utilizar primero la técnica de corte descrita anteriormente, que produce muchas pequeñas instancias *NCP*, y luego resolver cada instancia usando el algoritmo de reducción de retícula presentado anteriormente.

Datos: Multigrafo $G = (V, E, \phi)$

Resultado: Concatenaciones $\{\tilde{G}\, i\}$

Función: Encontrar Concatenaciones (G) :

$\{G1, \ldots, G'\} \leftarrow$ Corte Recursivo (G)

foreach $G_k = (V_k, E_k, \phi_k)$ **do**

$\tilde{E}k =$ LLL $(I \mid E_k)$

$\tilde{G}k = (Vk, \tilde{E}k, \phi k)$

end

return $\{\tilde{G}1, \ldots, \tilde{G}'\}$

end

La ventaja sobre el uso directo de la reducción de retícula en la instancia grande, además del hecho obvio de que las instancias más pequeñas se tratan más rápido, es que la carga computacional ahora se puede distribuir, ya que cada instancia pequeña se puede tratar de forma independiente.

Si solo estamos interesados en la mayor concatenación conectada, como será el caso en la siguiente sección, entonces nuestro algoritmo funciona aún mejor: De hecho, solo es necesario abordar el subgrafo más grande, y uno puede descartar al otro.

5.4. Resolución NCP como prueba de trabajo

Muchas construcciones de cadena de bloques se basan en una prueba de trabajo, es decir, un problema computacionalmente difícil que sirve para demostrar (bajo algunas hipótesis razonables) que uno ha pasado mucho tiempo calculando. Calcular una prueba de trabajo requiere operaciones que, como tales, son inútiles. Este desperdicio de energía es innecesario, y se propone una extensión interesante que es compatible con las cadenas de bloques existentes. El principio es reconocer como prueba válida de trabajo el resultado de las cancelaciones del libro de contabilidad. Intuitivamente, las concatenaciones mayores serían más recompensadas, ya que requieren más trabajo. Como resultado, los usuarios pueden mantener su copia local de la cadena de bloques más pequeña; de hecho, verificar que una concatenación dada sea válida es una tarea fácil. Esta es una prueba de trabajo al servicio de la comunidad.

Concretamente, un bloque de concatenación es similar a los bloques "estándar", pero verificado de una manera diferente. En lugar de contener transacciones, los bloques de concatenación contienen una descripción de la concatenación. Los usuarios responsables de publicar bloques de concatenación son recompensados en función del tamaño de sus concatenaciones. Los usuarios que reciban bloques de concatenación los comprobarán y los aceptarán solo si son válidos.

Sin embargo, antes de que los bloques de concatenación puedan usarse como prueba de trabajo, uno debe considerar estrategias de trampa y ajustar los incentivos, de modo que el cálculo honesto sea la opción racional para los mineros. Se identifican dos estrategias de trampa: los denominados ciclos fantasma y soldadura, para los que se sugieren contramedidas. Luego se discute la cuestión de cómo recompensar las concatenaciones.

Como resumen, para utilizar NCP como prueba de trabajo, se debe:

- requerir que las concatenaciones obedezcan las reglas de cazafantasmas, es decir, pertenezcan a un subgrafo muestreado aleatoriamente de una instantánea del multigrafo de transacción;

- aceptar únicamente concatenaciones conectadas como se explica en la sección 5.4.2.;

- ser recompensado linealmente en el tamaño de la concatenación, como se describe en la sección 5.4.2.

5.4.1. Ciclos fantasma

Creación de un ciclo fantasma.

Un primer comentario es que algunos usuarios pueden crear muchas transacciones (inútiles) con la intención de facilitar la concatenación. Para facilitar la exposición, uno solo considera los ciclos, pero señala que los adversarios también pueden crear camarillas. Este "ataque" no es muy práctico, ya que los usuarios solo pueden crear transacciones desde las cuentas que controlan. Pero dado que el número de cuentas que una persona puede crear es a priori ilimitado, no se puede descartar la posibilidad de que un tramposo actúe de la siguiente manera:

1. Encuentre la trayectoria más larga de transacciones idénticas que apuntan al nodo controlado: las escribe $v_i \xrightarrow{r} v_{i+1}$, con i = 0, ..., n y $v_{(n+1)}$ siendo los nodos bajo control adversario. Debe tenerse en cuenta que r es fijo. La búsqueda de dicho ciclo se puede hacer comenzando desde $v_{(n+1)}$, y realizando una búsqueda en profundidad en el grafo de transacciones.

2. Calcule la ganancia esperada de una prueba de trabajo basada en la concatenación que elimina (n+1) transacciones: lo llama G_{n+1}. Tal cantidad sería de conocimiento público, y uno puede asumir por simplicidad que $G_n > G_m$ siempre que n > m.

3. Si $G_{n+1} > r$, realiza una nueva transacción $v_{n+1} \xrightarrow{r} v_0$; y luego envía el ciclo concatenable $\{v_0, ..., v_{n+1}\}$ como "prueba de trabajo".

Al usar varias cuentas, un usuario puede crear cadenas artificialmente largas, solo para ser "encontradas" y eliminadas inmediatamente. Uno los llama "ciclos fantasma", y esta forma de engaño es, por supuesto, muy indeseable.

Cazafantasmas.

Hay dos formas (complementarias) de combatir fantasmas. Un enfoque económico consiste en hacer que los fantasmas no sean rentables. Una contramedida técnica, llamada cazafantasmas y descrita en el presente documento, asegura que los fantasmas no se puedan aprovechar, excepto quizás con una probabilidad insignificante.

Una idea natural para combatir los ciclos fantasma podría ser restringir qué parte del grafo de transacciones se puede anular. Podría estar restringido en "tiempo" o en "espacio", pero los enfoques sencillos no son satisfactorios:

- Por ejemplo, si B_t indica la cadena de bloques en un momento dado t, solo se puede considerar un tiempo umbral *T,* y solo acepta concatenaciones para B_s, donde *t-s > T.* Sin embargo, esto no evita que un adversario cree ciclos fantasma durante un período de tiempo más largo.

- Alternativamente, se observa que dado que la transacción que "cierra el ciclo" se origina en el tramposo, se puede requerir que la concatenación no contenga este nodo. Esta contramedida se elude fácilmente creando una nueva cuenta cuyo único propósito es reclamar las recompensas de la prueba de trabajo asociada.

Lo que resaltan las observaciones anteriores es la necesidad de que la concatenación se calcule en un grafo que no esté bajo el control del adversario. Así, para evitar este control de un adversario, una técnica puede consistir en seleccionar un subgrafo a procesar, en lugar de requerir el procesamiento de todo el multigrafo. La técnica entonces comprende:

- determinar, a partir de dicho multigrafo G, un registro inicial S_R;

- a partir de dicho registro inicial S_R, generar un subgrafo SG;

- usar dicho subgrafo SG para obtener dicho conjunto de concatenaciones $\tilde{G}$;

La determinación del registro inicial S_R ventajosamente puede depender de varios parámetros tales como el tiempo (t) y/o la aleatorización. En una realización específica, es posible muestrear un subgrafo SG uniformemente en el grafo de transacciones utilizando la siguiente técnica:

Cazafantasmas (*t, Bt*):

1. Considere bt el bloque de definición en el tiempo t

2. semilla = H(bt)

3. SG = SubGrafoGen (semilla)

4. devolver SG

Este procedimiento se basa en la idea de que un bloque de la cadena contiene suficiente entropía, ya que lleva resúmenes de todos los bloques anteriores (de acuerdo con el mecanismo de la cadena de bloques). El principio del cazafantasmas es que solo se deben aceptar las eliminaciones entre los nodos de SG.

Debe tenerse en cuenta que el procedimiento de muestreo debe ser determinista, de modo que los verificadores puedan asegurarse de que la concatenación realmente pertenece al subgrafo autorizado, y para que todos los mineros aborden la misma tarea.

Aquí, se usa una función pseudoaleatoria H, para la cual es difícil calcular preimágenes, es decir, dada *y* debería ser difícil de encontrar *X* tal que *H(x)=y.* Se cree que la mayoría de las funciones hash criptográficas estándar, tal como SHA-256, satisfacen esta propiedad; sin embargo, uno debe abstenerse de usar el propio SHA-256 vanilla: La prueba de trabajo de Bitcoin consiste precisamente en diseñar bloques cuyo hash se ajuste a un formato predeterminado. Una solución sencilla es definir, por ejemplo, *H(x) = SHA-256(x||x).*

El subgrafo SG se obtiene a través de *SubGrafoGen* mediante la selección de nodos (es decir, cuentas, que pueden estar bajo control adversario, realizándose la selección a partir de registros) y todas las aristas entre estos nodos. Para ser aceptada, una concatenación solo debe contener nodos de este subgrafo.

Suponiendo que el adversario tiene control sobre *k* fuera de *n* nodos, y que el subgrafo muestreado contiene l nodos, con *k < n/2,* la probabilidad de que al menos *m ≤ l* de estos nodos estén bajo control adversario es:

$$\frac{1}{2k-n} \cdot \frac{k^m}{n^l} \left(k^{l+1-m} - (n-k)^{l+1-m}\right)$$

En el límite que *k « n,* esta probabilidad es aproximadamente $(k/n)^m$*,* que no depende de la elección de:

Se supone que H es un oráculo aleatorio. Por lo tanto, SG se muestrea perfectamente de manera uniforme en G. Por lo tanto, un nodo dado tendrá probabilidad *k*/*n* de ser controlado por un adversario. Existen *l* nodos en SG, por lo tanto, la probabilidad de elegir al menos m nodos adversarios es *0* si *m > l* y:

$$(Pr[C_{\geq m}] = Pr[C_m] + Pr[C_{m+1}] + \cdots + Pr[C_l])$$

de lo contrario, donde C_p es el evento en el que exactamente p nodos elegidos están bajo control adversario. Dado que los nodos se seleccionan uniformemente de manera aleatoria,

$$Pr[C_p] = \left(\frac{k}{n}\right)^p (1-\frac{k}{n})^{l-p}$$

Por lo tanto,

$$\begin{aligned}\Pr[C_{\geq m}] = \Pr[C_m + \cdots + \Pr[C_l]] &= \sum_{p=m}^{l} (\frac{k}{n})^p (1-\frac{k}{n})^{pl-p} \\ &= \frac{1}{2k-n}\left(k\left(\left(\frac{k}{n}\right)^m \left(1-\frac{k}{n}\right)^{l-m} + \left(\frac{k}{n}\right)^l\right) - n\left(\frac{k}{n}\right)^m \left(1-\frac{k}{n}\right)^{l-m}\right) \\ &= \frac{1}{2k-n} \cdot \frac{k^m}{n^l} \left(k^{l+1-m} - (n-k)^{l+1-m}\right)\end{aligned}$$

Asumiendo *k « n,* es posible usar una expansión en serie en k/n de lo anterior para obtener:

$$Pr[C_{\geq m}] = \left(\frac{k}{n}\right)^m (1 + \frac{k}{n}(m-l+1) + O((k/n)^2)),$$

y, en particular, sigue el resultado.

Por lo tanto, la probabilidad de que un adversario logre crear un gran ciclo de fantasmas cuando se utiliza el procedimiento de cazafantasmas se vuelve exponencialmente pequeña.

En cuanto a cómo el *"bloque de definición"* b_t debe ser elegido, solo se requiere que todos los mineros y

verificadores estén de acuerdo en un procedimiento determinista para decidir si b_t es aceptable. Uno sugiere lo siguiente: Si T_{-1} es el sello de tiempo del último bloque de concatenación en la cadena de bloques, y T_{-2} es el sello de tiempo del bloque de concatenación anterior. Entonces b_t puede ser cualquier bloque agregado a la cadena de bloques entre T_{-2} y T_{-1}.

5.4.2. Concatenaciones de soldadura

Otra pregunta interesante, motivada por el mayor número de mineros de criptomonedas que paralelizan su trabajo, es medir cuánto ayuda el cálculo paralelo para resolver el NCP. Como se describió anteriormente, el algoritmo de corte genera muchos grafos pequeños que se pueden tratar de forma independiente.

En este escenario, después de reunir suficientes concatenaciones publicadas por pares, un usuario podría ensamblarlas en una instancia única más grande y reclamar las recompensas por ello. Desde un punto de vista teórico, una concatenación grande e inconexa satisface.

Sin embargo, el incentivo sería producir rápidamente muchas pequeñas concatenaciones. Dado que comprobar la falta de inconexiones es (bastante) fácil (en términos de tiempo y recursos), se sugiere que los usuarios rechacen las concatenaciones desconectadas, es decir, solo acepten las conectadas. Esto anima a los mineros a buscar concatenaciones más grandes y también limita la eficiencia de los grupos de mineros.

Este enfoque no impide, en teoría, que los usuarios unan las concatenaciones parciales en una más grande. Teniendo en cuenta, por ejemplo, el grafo de la figura 4, donde el usuario 1 encuentra una concatenación 10-10, el usuario 2 encuentra 20-20 y el usuario 3 encuentra 30-30. Entonces pueden coludirse para generar una concatenación conectada más grande.

Sin embargo, una conjetura que es un problema difícil en general es ensamblar concatenaciones que no están inconexas en una más grande; o al menos, que esto es tan caro como calcularlas desde cero. Además, las restricciones de cazafantasmas reducen las posibilidades de colusión al evitar que los nodos controlados por el adversario participen en el grafo de concatenación.

5.4.3. Determinando la recompensa

Usando el NCP como prueba de trabajo en una situación de cadena de bloques, y por ejemplo para el sistema de gestión de transacciones de Bitcoin, los usuarios pueden ser recompensados cuando calculan una concatenación válida. La recompensa exacta debe ajustarse con precisión para proporcionar los incentivos correctos. Debe tenerse en cuenta que esto depende de si la criptomoneda aplica tarifas de transacción o no.

Tarifas de transacción.

Si se aplican tales tarifas, entonces crear un fantasma es una operación costosa desde un punto de vista contradictorio. El sistema debe establecer la recompensa para una concatenación con m aristas, denominada recompensa (m), para que sea menor o igual al costo de crear un fantasma de tamaño m, que se puede asumir que es $m \cdot c$ donde c es la tarifa de transacción. Uno puede conformarse con una recompensa (m) = $m \cdot c$. Pueden aplicarse técnicas similares cuando se dispone de un espectro más amplio de tarifas de transacción.

Debe tenerse en cuenta que el uso de una función de recompensa sublineal es contraproducente, ya que fomenta la producción de muchas pequeñas cadenas, en lugar de una grande única.

Por el contrario, el uso de una función de recompensa súper lineal, al tiempo que fomenta las concatenaciones más grandes, también hace que los fantasmas sean rentables por encima de cierto tamaño.

Sin tarifas de transacción.

Si no hay tarifas de transacción, entonces el método antes mencionado no se aplica (ya que c = 0). Para criptomonedas que no utilizan tarifas de transacción, la caza de fantasmas () limita la creación de ciclos fantasmas. En tales casos, la función de recompensa puede ser una función afín arbitraria en el tamaño de la concatenación.

5.5. Dispositivo para reducir el tamaño de la base de datos

La divulgación también propone un dispositivo para reducir el tamaño de una base de datos. El dispositivo puede diseñarse específicamente para reducir el tamaño de una base de datos o cualquier dispositivo electrónico que comprenda un medio legible por ordenador no transitorio y al menos un procesador configurado por instrucciones legibles por ordenador almacenadas en el medio legible por ordenador no transitorio para implementar cualquier método en la divulgación.

De acuerdo con una realización mostrada en la figura 5, el dispositivo para reducir el tamaño de una base de datos incluye una Unidad Central de Procesamiento (CPU) 52, una Memoria de Acceso Aleatorio (RAM) 51, una Memoria de Solo Lectura (ROM) 53, un dispositivo de almacenamiento que están conectados a través de un bus de tal manera que pueden llevar a cabo la comunicación entre los mismos.

La CPU controla la totalidad del dispositivo mediante la ejecución de un programa cargado en la RAM. La CPU también realiza varias funciones al ejecutar un(os) programa(s) (o una aplicación(es)) cargado(s) en la RAM.

La RAM almacena varios tipos de datos y/o programa(s).

La ROM también almacena varios tipos de datos y/o un(os) programa(s) (Pg).

El dispositivo de almacenamiento, tal como una unidad de disco duro, una tarjeta SD, una memoria USB, etc., también almacena varios tipos de datos y/o un(os) programa(s).

El dispositivo realiza el método para reducir el tamaño de una base de datos como resultado de que la CPU ejecuta instrucciones escritas en un(os) programa(s) cargado(s) en la RAM, el(los) programa(s) se lee(n) desde la ROM o el dispositivo de almacenamiento y se carga(n) en la RAM.

Más específicamente, el dispositivo puede ser un servidor, un ordenador, una tableta, un teléfono inteligente o una cámara.

La divulgación también se refiere a un producto de programa informático que comprende un código de programa ejecutable por ordenador grabado en un medio de almacenamiento no transitorio legible por ordenador, el código de programa ejecutable por ordenador cuando se ejecuta, realizando el método para reducir el tamaño de una base de datos. El producto de programa informático se puede grabar en un CD, un disco duro, una memoria flash o cualquier otro medio legible por ordenador adecuado. También se puede descargar de Internet e instalar en un dispositivo para reducir el tamaño de una base de datos.

REIVINDICACIONES

1. Un método para reducir una cantidad de registros en una base de datos, siendo implementado dicho método por un aparato electrónico que comprende recursos de hardware para reducir la cantidad de registros en bases de datos, estando dicho método caracterizado por que dicha base de datos está representada en forma de un multigrafo orientado (G) en el que una arista valorada desde un vértice de origen a un vértice de destino representa un registro de dicha base de datos, estando dicho método además caracterizado por que comprende:

- al menos una etapa para obtener (10), en dicho grafo (G), al menos un conjunto de vértices y aristas que forman un grupo (L) de un determinado saldo (b) para cada vértice de dicho conjunto, denominado conjunto de concatenaciones ($\tilde{G}$);

- al menos una etapa para eliminar (20) dicho conjunto de concatenaciones ($\tilde{G}$) de dicho multigrafo orientado (G).

2. El método de acuerdo con la reivindicación 1, caracterizado por que el valor de dicho saldo (b) es igual a 0.

3. El método de acuerdo con la reivindicación 1, caracterizado por que dicha etapa para obtener dicho conjunto de concatenaciones ($\tilde{G}$) comprende:

- una etapa para obtener (110), a partir de dicho multigrafo (G) al menos un subgrafo (G_k) que comprende al menos un vértice (v_{k1}, ..., v_{kn}) y al menos una arista valorada (e_{k1},...,e_{kn});

- al menos una etapa para borrar (120), dentro del subgrafo del conjunto (G_k), al menos un vértice (v_{kx}) que está ausente de un grupo (L) de vértices para los cuales el saldo de los valores de cada vértice es nulo, entregando un subconjunto de concatenaciones ($\tilde{G}_k$).

4. El método de acuerdo con la reivindicación 3, caracterizado por que dicha etapa para obtener al menos un subgrafo (G_k) comprende una etapa para determinar, dentro de dicho multigrafo orientado (G), un conjunto de subgrafos $\{G_1, ..., G_l\}$, cada uno comprendiendo un conjunto de vértices fuertemente conectados.

5. El método de acuerdo con la reivindicación 3, caracterizado por que dicha etapa para determinar dicho conjunto de subgrafos $\{G1, ..., G_l\}$ comprende la implementación de un módulo de Tarjan.

6. El método de acuerdo con la reivindicación 3, en el que dicha etapa para eliminar, dentro de dicho subgrafo (G_k), al menos un vértice (v_k), comprende, para dicho al menos un vértice (v_k):

- cuando el número de aristas de entrada o el número de aristas de salida de dicho vértice (v_k) es igual a cero, eliminar dicho vértice (v_k);

- cuando el número de aristas de salida y el número de aristas de entrada es igual a uno y el valor de la arista de salida única es diferente del valor de la arista de entrada, eliminar dicho vértice (v_k);

7. El método de acuerdo con la reivindicación 1, caracterizado por que dicha etapa para obtener, a partir de dicho multigrafo G, dicho conjunto de concatenaciones $\tilde{G}$ comprende, antes de la obtención de dicho conjunto de concatenaciones:

- determinar, a partir de dicho multigrafo G, un registro inicial S_R;

- a partir de dicho registro inicial S_R, generar un subgrafo SG;

- usar dicho subgrafo SG para obtener dicho conjunto de concatenaciones $\tilde{G}$;

8. El método de acuerdo con la reivindicación 7, caracterizado por que dicha etapa para generar dicho subgrafo SG a partir de dicho registro S_R comprende obtener al menos un nodo y derivar dicho subgrafo de dicho al menos un nodo.

9. El método de acuerdo con la reivindicación 1, caracterizado por que comprende al menos una etapa para determinar, en base a dicho conjunto de concatenaciones ($\tilde{G}$), al menos un dato que representa una recompensa.

10. El método de acuerdo con la reivindicación 9, caracterizado por que dicha etapa para determinar al menos un dato que representa una recompensa comprende determinar una tarifa de transacción de dicho conjunto de concatenaciones ($\tilde{G}$).

11. Un dispositivo electrónico para reducir una cantidad de registros en una base de datos, siendo implementado dicho método por un aparato electrónico que comprende recursos de hardware para reducir la cantidad de registros en bases de datos, caracterizado por que dicho dispositivo comprende medios para grabar dicha base de datos, en una memoria, en forma de multigrafo orientado (G) en el que una arista valorada desde un vértice de origen a un vértice de destino representa un registro de dicha base de datos, caracterizado por que dicho dispositivo comprende además:

- medios para obtener (10), en dicho grafo (G), al menos un conjunto de vértices y aristas que forman un grupo (L) de un determinado saldo (b) para cada vértice de dicho conjunto, denominado conjunto de concatenaciones ($\tilde{G}$);

- medios para eliminar (20) dicho conjunto de concatenaciones ($\tilde{G}$) de dicho multigrafo orientado (G).

12. Un producto de programa informático que comprende un código de programa ejecutable por ordenador grabado en un medio de almacenamiento no transitorio legible por ordenador, realizando el código de programa ejecutable por ordenador cuando se ejecuta, cualquier método de acuerdo con las reivindicaciones 1 a 10.

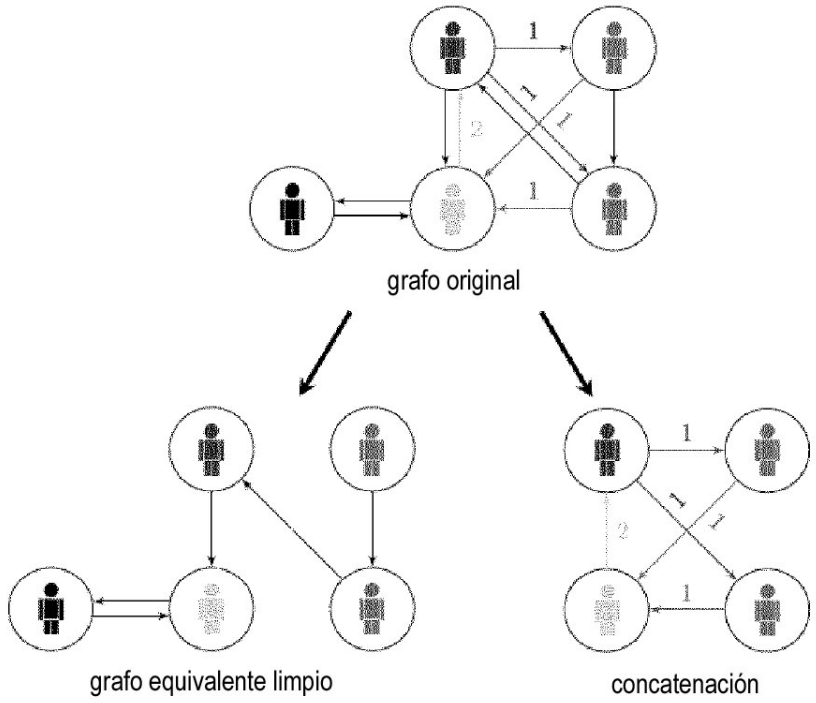

Figura 1

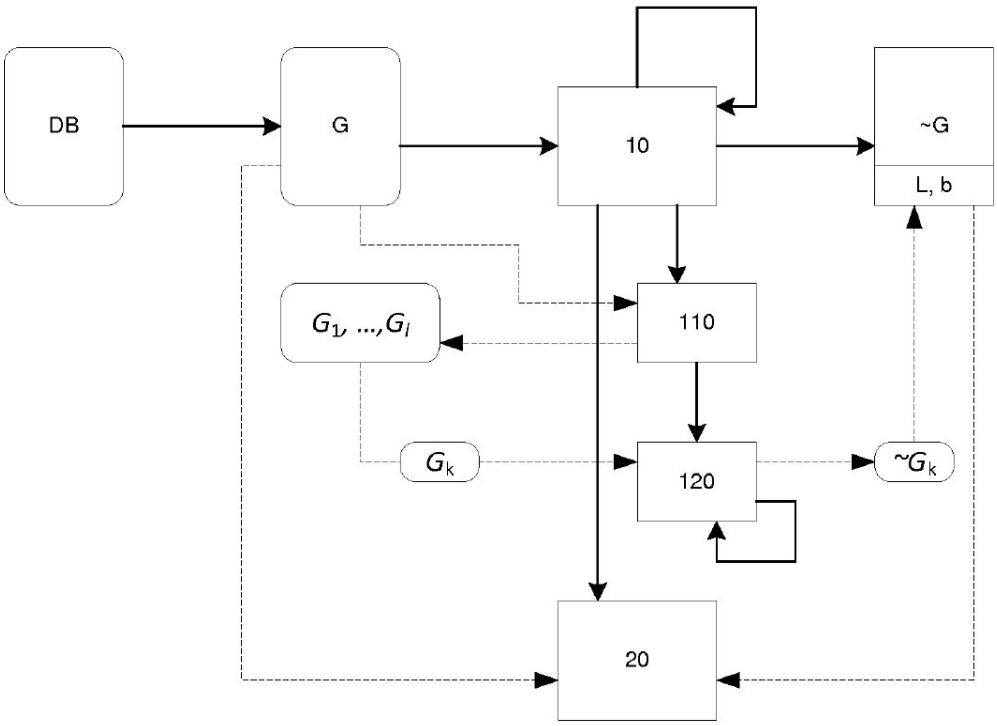

Figura 2

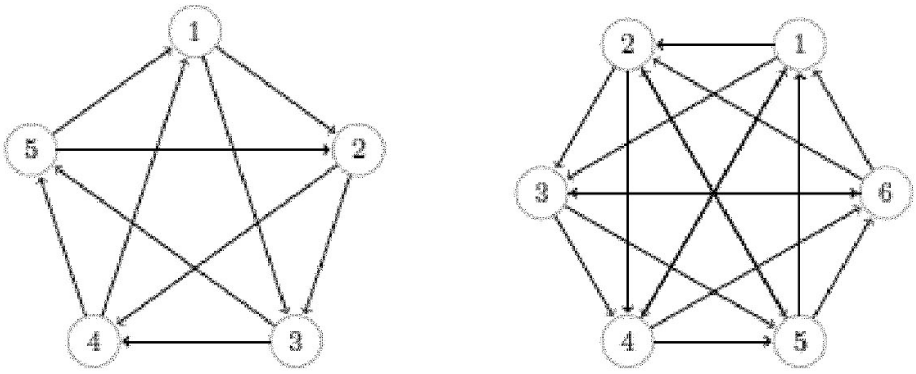

Figura 3

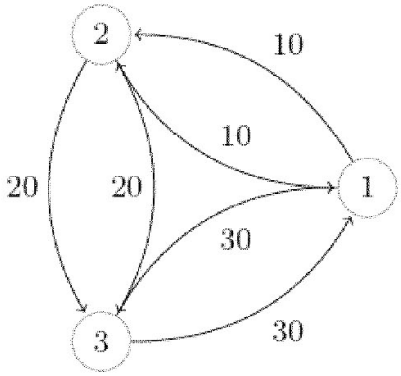

Figura 4

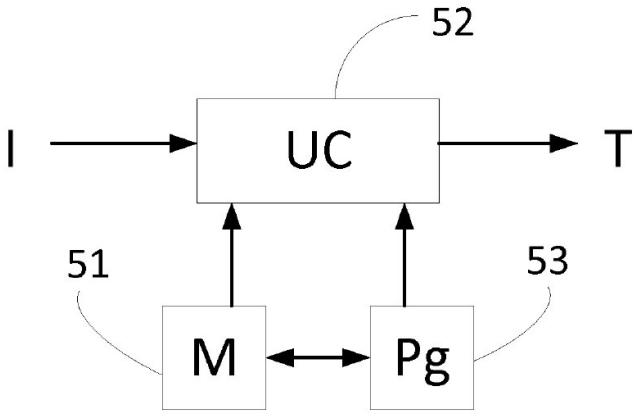

Figura 5