

CÓMPUTO DE PROGRAMAS A LARGO PLAZO PARA
TRANSFERENCIAS DE DATOS A TRAVÉS DE UNA RED DE ÁREA
AMPLIA

5 **Campo y antecedentes de la invención**

Las redes de área amplia (WAN, por sus siglas en inglés) se están volviendo universales, y a menudo se transfieren volúmenes relativamente grandes de datos entre dispositivos de cómputo en las WAN. Para soportar la transferencia de grandes volúmenes de datos, los operadores de las WAN respectivas invierten una cantidad sustancial de recursos en *hardware* de redes de cómputo que facilita la transferencia de datos entre dispositivos de cómputo en las WAN. El aumento al máximo de la utilización de este *hardware* de redes de cómputo en una WAN, por lo tanto, es deseable desde un punto de vista de eficiencia.

Desdichadamente, los solicitantes de transferencias de datos en una WAN, que generan tráfico de datos de red, convencionalmente no son conscientes unos de otros. De esta forma, un solicitante de una transferencia de datos establece una solicitud con la expectativa de que la WAN pueda manejar la solicitud, y la transferencia de datos se inicializará inmediatamente. En tanto que esto no es en gran parte problemático para volúmenes relativamente pequeños de tráfico de red, cuando las transferencias de datos que implican volúmenes relativamente grandes de datos se traslapan en el tiempo, la WAN se puede cargar excesivamente. Estas

transferencias superpuestas pueden presentarse incluso si uno o más solicitantes tienen la opción de retrasar sus transferencias a otro momento, ya que no se proporcionan mecanismos mediante los cuales se puedan diferir estas transferencias de datos. En consecuencia, no solamente esta inflexibilidad puede dar por resultado periodos de carga excesiva de red, también puede dar por resultado otros periodos de tiempo durante los cuales se infrautilizan los recursos de *hardware* de la WAN, ya que existe una cantidad relativamente pequeña de tráfico de red. En consecuencia, los operadores de las WAN a menudo se ven forzados a comprar *hardware* de redes de cómputo costoso para adaptarse a periodos de tiempo de alta demanda, cuando se transfieren grandes volúmenes de datos a través de la WAN. Convencionalmente, esto puede dar por resultado que se sobreaprovisione la WAN, de tal forma que este *hardware* de redes de cómputo, costoso permanece infrautilizado durante porciones sustanciales de su vida de servicio.

Breve descripción de la invención

Lo siguiente es una breve descripción de la materia que se describe en más detalle en la presente. No se propone que esta breve descripción sea limitante en cuanto al alcance de las reivindicaciones.

Se describen en la presente diferentes tecnologías relacionadas con el programa de transferencias de datos en una red, tal como una red de área amplia (WAN), con base en una pluralidad

de solicitudes para las transferencias de datos, donde las solicitudes tienen fechas límites respectivas, y donde las transferencias de datos se van a completar antes de las fechas límite respectivas. En una modalidad de ejemplo, la red puede comprender una pluralidad de centros de datos (cada uno que comprende una pluralidad de dispositivos de cómputo), y una solicitud para transferir datos puede ser una solicitud para transferir datos de un dispositivo de cómputo en un primer centro de datos a un dispositivo de cómputo en un segundo centro de datos. Los centros de datos en la red pueden ser incluidos en una nube pública o una nube privada. Una nube pública expone los recursos de los centros de datos (por ejemplo, aplicaciones y almacenamiento) al público general por medio de internet. Por el contrario, una nube privada expone los recursos de los centros de datos a una empresa privada que opera los centros de datos.

Además, la red comprende una pluralidad de dispositivos que facilitan la transferencia de datos entre dispositivos en la red. La pluralidad de dispositivos incluye una pluralidad de dispositivos de cómputo que se configuran para almacenar y procesar datos, donde la pluralidad de dispositivos de cómputo reside en el borde de la red. Por ejemplo, la pluralidad de dispositivos de cómputo puede incluir dispositivos de cómputo de servidor en centros de datos. De forma adicional, la pluralidad de dispositivos comprende dispositivos de infraestructura de red (por ejemplo, conmutadores, enrutadores, concentradores, puertas de enlace, etc.) que se configuran para

dirigir la transferencia de datos en medio de y entre los dispositivos de cómputo en la pluralidad de dispositivos de cómputo.

La red también incluye un dispositivo de cómputo controlador que se configura para programar las transferencias de datos a través de la red. Por ejemplo, el dispositivo de cómputo controlador puede recibir una pluralidad de solicitudes para transferir datos entre dispositivos de cómputo respectivos en la red, en donde cada solicitud puede identificar: 1) un dispositivo de cómputo fuente, respectivo (por ejemplo, desde donde se van a transferir los datos; 2) un dispositivo de cómputo receptor, respectivo (por ejemplo, al cual se van a transferir los datos; 3) una cantidad respectiva de datos que se van a transferir del dispositivo de cómputo fuente al dispositivo de cómputo receptor; y 4) una fecha límite respectiva, en donde la transferencia de datos del dispositivo de cómputo fuente al dispositivo de cómputo receptor se va a completar antes de la fecha límite. El dispositivo de cómputo controlador puede recibir estas solicitudes en horas arbitrarias, y las solicitudes pueden especificar diferentes cantidades de datos y diferentes fechas límite. Por ejemplo, se puede recibir una primera solicitud de transferencia de datos en un primer punto en el tiempo, y puede indicar que la transferencia de datos es relativamente urgente al especificar una fecha límite que es próxima en el tiempo al primer punto en el tiempo. Se puede recibir una segunda solicitud de transferencia de datos en un segundo punto en el tiempo (donde el primer punto en el tiempo y el segundo punto en el tiempo son próximos), e indicar que

la transferencia de datos no es particularmente imperiosa al especificar una fecha límite que es relativamente distal del segundo punto en el tiempo.

El dispositivo de cómputo controlador recibe estas solicitudes y
5 calcula un programa a largo plazo para la transferencia de datos en la red. El programa a largo plazo puede cubrir una pluralidad de unidades de tiempo (potencialmente uniformes), donde el flujo de datos a través de la red se define en el programa a largo plazo para cada unidad de tiempo cubierta por el programa a largo plazo. De
10 esta forma, el programa a largo plazo comprende un subprograma para cada unidad de tiempo cubierta por el programa a largo plazo. En un ejemplo no limitante, el programa a largo plazo puede cubrir doce intervalos de 5 minutos, donde las unidades de tiempo son los intervalos de 5 minutos. Para cada una de estas unidades de tiempo
15 en el programa a largo plazo, el programa a largo plazo puede incluir un subprograma respectivo. Un subprograma puede identificar cuáles dispositivos de cómputo van a actuar como dispositivos de cómputo fuente para la unidad de tiempo del subprograma, velocidades respectivas a las cuales los dispositivos de cómputo van a producir
20 los datos, rutas respectivas a través de las cuales van a viajar los datos entre los dispositivos de cómputo fuente y los dispositivos de cómputo receptores, etc. Por lo tanto, se puede demostrar que el dispositivo de cómputo controlador calcula el programa a largo plazo para facilitar la terminación de las transferencias de datos
25 solicitadas antes de sus fechas límite, respectivas en tanto que

aumenta al máximo la utilización de los recursos de red. Con ese fin, en un ejemplo, el dispositivo de cómputo controlador puede determinar si aceptar o rechazar solicitudes de transferencia de datos recién recibidas con base en el programa a largo plazo (por ejemplo, el dispositivo de cómputo controlador puede determinar si se puede completar una transferencia de datos solicitada antes de su fecha límite especificada con base en el programa a largo plazo, solicitudes previamente recibidas donde no se han completado sus transferencias de datos, y sus fechas límite respectivas). El dispositivo de cómputo controlador recalcula el programa a largo plazo con el paso del tiempo para tomar en consideración las solicitudes recién recibidas, modificaciones en la red, solicitudes cumplidas (donde se han completado las transferencias de datos respectivas), etc. En una modalidad de ejemplo, el dispositivo de cómputo controlador puede calcular el programa a largo plazo al ejecutar un proceso de optimización. Por ejemplo, el proceso de optimización puede incluir la ejecución de un algoritmo de empaquetamiento y cobertura mezclado.

El dispositivo de cómputo controlador puede calcular además un programa a corto plazo con base en el programa a largo plazo. El programa a corto plazo puede cubrir menos unidades de tiempo que el programa a largo plazo. Por ejemplo, el programa a corto plazo puede cubrir una unidad de tiempo individual que es inmediatamente posterior en el tiempo a una unidad de tiempo actual (por ejemplo, un intervalo de 5 minutos). El programa a corto plazo incluye: 1)

tablas de encaminamiento para dispositivos de infraestructura de red respectivos en la red; y 2) instrucciones para dispositivos de cómputo fuente que identifican los datos que se van a producir por los dispositivos de cómputo fuente y las velocidades respectivas a las cuales se van a producir los datos por los dispositivos de cómputo fuente. El dispositivo de cómputo controlador calcula el programa a corto plazo para facilitar la terminación de las transferencias de datos de acuerdo con el programa a largo plazo. El dispositivo de cómputo controlador transmite las tablas de encaminamiento e instrucciones en el programa a corto plazo a los dispositivos de cómputo fuente y dispositivos de infraestructura de red respectivos en la red.

En la modalidad de ejemplo donde la red soporta transferencias de datos entre dispositivos de cómputo expuestos en una nube pública, el dispositivo de cómputo controlador puede calcular un programa de fijación de precios que se expone a los clientes del operador de la nube pública. El dispositivo de cómputo controlador puede calcular el programa de fijación de precios para suavizar la demanda con el paso del tiempo, facilitando por lo tanto el aumento al máximo de la utilización de los recursos de red de la red. Por ejemplo, la lista de precios puede indicar que las solicitudes con demandas menos urgentes están asociadas con tarifas más pequeñas por unidad de datos transferida cuando se compara con las solicitudes con demandas más urgentes. En otro ejemplo, la lista de precios puede indicar que las transferencias de datos con fechas

límite asociadas dentro de un periodo de tiempo particular en general se cobran más por unidad de datos transferida cuando se comparan con las transferencias de datos con fechas límite asociadas dentro de otros periodos de tiempo (que están asociados con menos
5 demanda que el periodo de tiempo particular). Aún además, los precios para transferencias de datos pueden estar basados en los dispositivos de cómputo fuente y/o destino.

El resumen anterior presenta un resumen simplificado a fin de proporcionar un entendimiento básico de algunos aspectos de los
10 sistemas y/o métodos analizados en la presente. Este resumen no es una descripción general exhaustiva de los sistemas y/o métodos analizados en la presente. No se propone que identifique elementos clave/críticos o delimite el alcance de estos sistemas y/o métodos. Su único propósito es presentar algunos conceptos en una forma
15 simplificada como un preludio a la descripción más detallada que se presenta más adelante.

Breve descripción de los dibujos

La figura 1 es un diagrama de bloques funcional de un sistema
20 de ejemplo que facilita la programación de transferencias de datos en medio de y entre dispositivos de cómputo en una red.

La figura 2 es un programa a largo plazo de ejemplo para transferencias de datos en la red.

La figura 3 es un programa a corto plazo de ejemplo para
25 transferencias de datos en la red.

La figura 4 es un diagrama de bloques funcional de un componente programador de ejemplo que puede generar un programa a largo plazo y un programa a corto plazo para transferencias de datos en la WAN.

5 La figura 5 es un gráfico de red de ejemplo.

La figura 6 es una interfaz de usuario gráfica de ejemplo que se puede exponer a un cliente para facilitar la recepción de una solicitud para una transferencia de datos a través de la red.

10 La figura 7 es un diagrama de flujo que ilustra una metodología de ejemplo para programar transferencias de datos en la red.

La figura 8 es un diagrama de flujo que ilustra una metodología de ejemplo para calcular un programa a largo plazo para transferencias de datos en la red.

15 La figura 9 es un diagrama de flujo que ilustra una metodología de ejemplo para exponer un programa de precios que corresponde a la transferencia de datos en la red.

La figura 10 es un sistema de cómputo de ejemplo.

Descripción detallada de la invención

20 Ahora se describen diferentes tecnologías con respecto a la transferencia de datos entre dispositivos de cómputo en una red (tal como una red de área amplia (WAN)) con referencia a las figuras, en donde números de referencia similares se utilizan para hacer referencia a elementos similares en todo momento. En la siguiente
25 descripción, para propósitos de explicación, se exponen varios

detalles específicos a fin de proporcionar un entendimiento completo de uno o más aspectos. Puede ser evidente, sin embargo, que estos aspectos se pueden practicar sin estos detalles específicos. En otros casos, se muestran estructuras y dispositivos bien conocidos en forma de diagrama de bloques a fin de facilitar la descripción de uno o más aspectos. Además, se va a entender que la funcionalidad que se describe como que se lleva a cabo por un componente de sistema individual se puede realizar por múltiples componentes. De forma similar, por ejemplo, un componente individual se puede configurar para llevar a cabo la funcionalidad que se describe como que se lleva a cabo por múltiples componentes.

Además, el término "o" se propone que signifique una "o" inclusiva en lugar de una "o" exclusiva. Es decir, a menos que se especifique de otra forma, o se aclare a partir del contexto, la frase "X emplea A o B" se propone que signifique cualquiera de las permutaciones inclusivas naturales. Es decir, la frase "X emplea A o B" se satisface por cualquiera de los siguientes casos: X emplea A; X emplea B; o X emplea tanto A como B. Además, los artículos "uno" y "una", como se utilizan en esta solicitud y en las reivindicaciones anexas, se deben considerar en general que significan "uno o más" a menos que se especifique de otra forma o se aclare a partir del contexto que está dirigido a una forma singular.

Además, como se utiliza en la presente, los términos "componente" y "sistema" se propone que abarquen almacenamiento de datos leíbles por computadora que se configura con instrucciones

ejecutables por computadora que provocan que se lleve a cabo cierta funcionalidad cuando se ejecutan por un procesador. Las instrucciones ejecutables por computadora pueden incluir una rutina, una función, o similares. También se va a entender que un
5 componente o sistema se puede colocar en un dispositivo individual o distribuir a través de varios dispositivos. Además, como se utiliza en la presente, el término "de ejemplo" se propone que signifique que funciona como una ilustración o ejemplo de algo, y no se propone que indique una preferencia.

10 Con referencia ahora a la figura 1, se ilustra un dispositivo de cómputo controlador de ejemplo 100 que se configura para calcular un programa a largo plazo para transferencias de datos en una red 102. Por ejemplo, la red 102 puede ser una WAN. En tanto que el dispositivo de cómputo controlador 100 mostrado como que es
15 externo a la red, se va a entender que esto es para propósitos de ilustración, y que el dispositivo de cómputo controlador 100 está incluido en la red 102. Además, en tanto que el dispositivo de cómputo controlador 100 se muestra y se describe como que es un dispositivo de cómputo individual, el dispositivo de cómputo controlador 100 es para abarcar un sistema de control lógicamente
20 centralizado, en donde la funcionalidad descrita como que se lleva a cabo por el dispositivo de cómputo controlador 100 se puede distribuir a través de varios dispositivos de cómputo.

La red 102 incluye una pluralidad de dispositivos de cómputo
25 104-110 que residen en el borde de la red 102. En una modalidad de

ejemplo, varios dispositivos de cómputo en la pluralidad de dispositivos de cómputo 104-110 pueden ser entre diez dispositivos de cómputo y 3000 dispositivos de cómputo. En otra modalidad de ejemplo, varios dispositivos de cómputo en la pluralidad de dispositivos de cómputo 104-110 pueden ser entre diez dispositivos de cómputo y 100,000 dispositivos de cómputo. En una modalidad de ejemplo, los dispositivos de cómputo 104-110 pueden ser dispositivos de cómputo de servidor que residen en bastidores de servidores respectivos. Además, los dispositivos de cómputo 104-110, en un ejemplo, pueden estar en diferentes centros de datos respectivos, en donde cada centro de datos comprende una pluralidad respectiva de dispositivos de cómputo en comunicación el uno con el otro. Los centros de datos se pueden emplear para facilitar la provisión de una nube pública y/o privada. Una nube pública expone los recursos de los centros de datos al público (por ejemplo, para tarifas) por medio de internet, en tanto que una nube privada expone los recursos de los centros de datos a una empresa que opera los centros de datos. La red 102 comprende además una pluralidad de dispositivos de infraestructura de red 112-114 que facilitan la transferencia de datos en medio de y entre los dispositivos de cómputo 104-110. Por ejemplo, los dispositivos de infraestructura de red 112-114 pueden ser o incluir conmutadores, enrutadores, concentradores, puertas de enlace, o similares. De acuerdo con un ejemplo, varios dispositivos de infraestructura de red en la pluralidad de dispositivos de infraestructura de red pueden ser

entre 10 dispositivos y 1000 dispositivos. En otro ejemplo, varios dispositivos de infraestructura de red en la pluralidad de dispositivos de infraestructura de red pueden ser entre 100 dispositivos y 10000 dispositivos.

5 En un ejemplo, un propietario de los datos retenidos en uno de los dispositivos de cómputo 104-110 puede transferir de forma deseable estos datos a otro de los dispositivos de cómputo 104-110. De forma convencional, por ejemplo, cuando se recibe una solicitud para transferir datos del primer dispositivo de cómputo 104 al tercer
10 dispositivo de cómputo 108, está implicada muy poca programación. En su lugar, el primer dispositivo de cómputo 104 comienza a transmitir datos a una velocidad tan alta como sea posible, y los datos viajan a lo largo de una o más rutas a través de la red 102 al tercer dispositivo de cómputo 108, cargando potencialmente la
15 infraestructura de la red 102. De forma específica, los dispositivos de infraestructura de red 112-114 se configuran convencionalmente con instrucciones para equilibrar la carga transmitida a través de enlaces de red cuando se dirigen datos a lo largo del receptor propuesto (el tercer dispositivo de cómputo 108). Este enfoque es
20 factible para redes sobreaprovisionadas, donde un operador de la WAN 102 proporciona suficiente capacidad de red para satisfacer la demanda de transferencia de datos máxima. Esto, sin embargo, tiende a ser un uso ineficiente de los recursos de la red 102.

 El dispositivo de cómputo controlador 100, como se describió
25 en más detalle más adelante, calcula un programa a largo plazo para

transferencia de datos a través de la red 102, que puede dar por resultado en la suavización del volumen del tráfico a través de la red 102 con el paso del tiempo, incrementando por lo tanto la eficiencia de la utilización de la red 102. Con ese fin, el dispositivo de cómputo controlador 100, en contraste con los enfoques convencionales, contempla las fechas límite de transferencia (recibidas en diferentes horas y que tienen diferentes duraciones) cuando calcula el programa a largo plazo (plan) para transferencias de datos en la red 102.

Con más detalle con respecto al cálculo de un programa a largo plazo, el dispositivo de cómputo controlador 100 incluye un componente receptor 116 que puede recibir una pluralidad de solicitudes de transferencias de datos. Por ejemplo, la red 102 se puede configurar para soportar de forma simultánea 100,000 flujos de tráfico separados (por ejemplo, transferencias de datos entre dispositivos de cómputo en los dispositivos de cómputo 104-110). Las solicitudes de transferencias de datos recibidas por el componente receptor 116 incluyen identidades respectivas de fuentes de datos que se van a transmitir, identidades respectivas de receptores propuestos de datos que se van a transmitir, volúmenes de datos respectivos que se van a transmitir a través de la WAN 102, y fechas límite respectivas. De forma más particular, una solicitud de transferencia de datos recibida por el componente receptor 116 incluye: 1) una identidad de un dispositivo de cómputo fuente a partir del cual se van a transferir los datos; 2) una indicación de los datos

que se van a transferir; 3) una identidad de un dispositivo de cómputo receptor que va a recibir los datos que se van a transferir; 4) un volumen de los datos que se van a transferir del dispositivo de cómputo fuente al dispositivo de cómputo receptor; y 5) una fecha límite, en donde el volumen de datos se va a transferir del dispositivo de cómputo fuente al dispositivo de cómputo receptor antes de la fecha límite. Además, se va a entender que la solicitud puede incluir múltiples fecha límite potenciales, donde las tarifas para la transferencia varían dependiendo de la fecha límite. El componente receptor 116 puede recibir la solicitud de transferencia de datos del dispositivo de cómputo fuente, del dispositivo de cómputo receptor, o de otro dispositivo de cómputo operado por un propietario de los datos que se van a transferir del dispositivo de cómputo fuente al dispositivo de cómputo receptor. Las solicitudes de transferencias de datos se pueden recibir de cualquiera de los dispositivos en los dispositivos de cómputo 104-110, o se pueden recibir de un sistema o componente que identifica servidores a partir de los cuales se van a transferir los datos. En un ejemplo, el sistema o componente mencionado anteriormente no necesita identificar dispositivos de cómputo fuente y objetivo particulares en la solicitud; por lo tanto, la solicitud puede ser más general, que solicita información acerca de las rutas en la red, velocidades de transferencia, etc.

En un ejemplo no limitante, el componente receptor 116 puede recibir una primera solicitud de transferencia de datos, en donde la

primera solicitud de transferencia de datos identifica el primer dispositivo de cómputo 104 como el dispositivo de cómputo fuente, identifica los datos que se van a transferir, identifica el tercer dispositivo de cómputo 108 como el dispositivo de cómputo receptor, 5 identifica que el volumen de los datos es 50 terabytes, e identifica que esta transferencia se va a completar antes de las 4:00 pm de la fecha de la solicitud de transferencia de datos. De forma similar, el componente receptor 116 puede recibir una M-ésima solicitud de transferencia de datos, en donde la M-ésima solicitud de 10 transferencia de datos identifica el segundo dispositivo de cómputo 106 como el dispositivo de cómputo fuente, identifica los datos que se van a transferir, identifica el primer dispositivo de cómputo 104 como el dispositivo de cómputo receptor, identifica que el volumen de los datos es 5 petabytes, e identifica que esta transferencia se va 15 a completar a las 5:00 am al día siguiente.

El dispositivo de cómputo controlador 100 comprende además un componente programador 118 que está en comunicación con el componente receptor 116. El dispositivo de cómputo controlador 100 tiene acceso a un almacén de datos 120, en donde el almacén de 20 datos 120 incluye un mapa de red 122. El mapa de red 122 representa la topografía física de la red 102. Por ejemplo, el mapa de red 122 puede ser una gráfica implementada por computadora que es representativa de la red 102, en donde la gráfica incluye nodos que son representativos de los dispositivos (por ejemplo, los 25 dispositivos de cómputo 104-110 y los dispositivos de infraestructura

de red 112-114) en la red 102 y los bordes que representan enlaces de comunicaciones entre dispositivos respectivos. El mapa de red 122 puede incluir además datos que identifican limitaciones de la red 102, tal como capacidades de enlaces respectivos de la red 102. En una modalidad de ejemplo, el mapa de red 122 se puede actualizar de vez en cuando con base en los datos recibidos de los dispositivos de la red 102. Por ejemplo, se pueden recibir datos de un dispositivo en la red 102 que indican que un enlace particular está inactivo. En otro ejemplo, se pueden recibir datos de un dispositivo en la red 102 que indican que se ha restablecido un enlace particular. El mapa de red 122 se puede actualizar de vez en cuando para reflejar estos cambios en la topología de la red 102.

El componente programador 118 recibe la pluralidad de solicitudes de transferencias de datos (con el paso del tiempo) y el mapa de red 122 del repositorio de datos 120, calcula un programa a largo plazo 124 para transferencias de datos en la red 102 con base en las solicitudes de transferencias de datos y el mapa de red 122, y almacena el programa a largo plazo 124 en el repositorio de datos 120. Además, en tanto que no se muestra, el componente programador 118 puede calcular el programa a largo plazo con base en la utilización histórica de la red. Por ejemplo, si un cliente solicita de forma histórica transferencia de datos diario a una hora particular, el componente programador 118 puede reservar recursos de red para el cliente en el programa a largo plazo, incluso si el componente receptor 116 aún no ha recibido una solicitud de transferencia de

datos del cliente. Aún más, el componente programador 118 puede reservar recursos de red para solicitudes *ad hoc* cuando calcula el programa a largo plazo, en donde las solicitudes *ad hoc* convencionalmente son aquellas que no tienen ninguna fecha límite especificada y solicitan un volumen relativamente pequeño de datos que se van a transferir a través de la red 102. En un ejemplo, y como se describirá en más detalle más adelante, el componente programador 118 puede ejecutar un proceso de optimización cuando calcula el programa a largo plazo 124, en donde el proceso de optimización puede incluir la ejecución de un algoritmo de empaquetamiento y cobertura mezclado.

Ahora se exponen detalles adicionales con respecto al programa a largo plazo 124. El programa a largo plazo 124 cubre una pluralidad de unidades de tiempo que avanzan en el tiempo (por ejemplo, unidades de tiempo futuro). En una modalidad de ejemplo, las unidades de tiempo cubiertas por el programa a largo plazo 124 pueden ser uniformes, de tal forma que las unidades de tiempo tienen una duración común. En otra modalidad de ejemplo, las unidades de tiempo cubiertas por el programa a largo plazo pueden ser no uniformes, en donde la duración de una unidad de tiempo cubierta por el programa a largo plazo 124 próxima en el tiempo a la hora actual es más corta cuando se compara con una unidad de tiempo cubierta por el programa a largo plazo que es distal en el tiempo de la hora actual. De acuerdo con un ejemplo, el programa a largo plazo 124 puede cubrir 240 ventanas de tiempo de un minuto

consecutivas (por ejemplo, para una ventana de tiempo de 4 horas totales), en donde las unidades de tiempo cubiertas por el programa a largo plazo 124 son ventanas de un minuto. El programa a largo plazo 124 incluye un subprograma respectivo para cada unidad de tiempo cubierta por el programa a largo plazo 124, en donde un subprograma define los flujos de datos a través de la red 102 durante su unidad de tiempo.

Por ejemplo, el subprograma puede identificar cuáles dispositivos de cómputo en los dispositivos de cómputo 104-110 son dispositivos de cómputo fuente, cuáles dispositivos de cómputo en los dispositivos de cómputo 104-110 son dispositivos de cómputo receptores (donde un dispositivo de cómputo puede ser tanto un dispositivo de cómputo fuente como un dispositivo de cómputo receptor), las velocidades a las cuales los dispositivos de cómputo fuente respectivos van a producir los datos, y las rutas a través de las cuales se van a transferir los datos de los dispositivos de cómputo fuente a los dispositivos de cómputo receptores. El componente programador 118 recalcula el programa a largo plazo 124 conforme pasa el tiempo, de tal forma que el programa a largo plazo cubre una ventana de tiempo de una duración particular conforme el tiempo avanza. El recálculo del programa a largo plazo permite además que se contemplen además nuevas solicitudes de transferencias de datos (*ad hoc* o con fechas límite especificadas), que se contemplen cambios en la topología de la red 102, que se contemplen solicitudes completadas, etc. Más adelante se expone un

proceso de ejemplo para llevar a cabo este cálculo del programa a largo plazo 124.

El componente programador 118 puede calcular un programa a corto plazo 126 con base en el programa a largo plazo 124 y puede
5 almacenar el programa a corto plazo 126 en el repositorio de datos 120. El programa a corto plazo 126 define el flujo de datos a través de la red 102 durante una unidad de tiempo que es inmediatamente posterior a la unidad de tiempo actual (por ejemplo, la siguiente ventana de tiempo de un minuto). En consecuencia, el programa a
10 corto plazo 126 cubre menos unidades de tiempo que el programa a largo plazo 124. En una modalidad de ejemplo, el programa a corto plazo 126 puede cubrir una unidad de tiempo individual. El programa a corto plazo 126 incluye: 1) tablas de encaminamiento para los dispositivos de infraestructura de red 112-114, respectivamente; e
15 instrucciones para dispositivos de cómputo fuente en los dispositivos de cómputo 104-110 en cuanto a si producir datos particulares durante el tiempo cubierto por el programa a corto plazo 126 y una velocidad a la cual producir los datos particulares. El dispositivo de cómputo controlador 100 puede transmitir las tablas de
20 encaminamiento respectivas a los dispositivos de infraestructura de red 112-114 y puede transmitir las instrucciones respectivas a uno o más de los dispositivos de cómputo 104-110.

Por lo tanto se puede comprobar que el componente programador 118 calcula el programa a largo plazo 124 y el
25 programa a corto plazo 126 para facilitar la terminación de las

transferencias de datos solicitadas que corresponden a las solicitudes de transferencias de datos aceptadas antes de sus fechas límite respectivas. En otro ejemplo, donde no es posible la terminación de las transferencias de datos, el componente

5 programador 118 puede calcular el programa a largo plazo 124 para reducir al mínimo las pérdidas de ingresos, para permitir que se satisfaga un porcentaje de solicitudes (por ejemplo, que se satisfaga el 95% de todas las solicitudes), o similares. Además, como se señaló anteriormente, el componente programador 118 puede

10 determinar si aceptar o rechazar una solicitud de transferencia de datos recibida con base en el programa a largo plazo 124. De acuerdo con un ejemplo, el programa a largo plazo 124 puede indicar que, para una ventana de tiempo particular, se programa un cierto enlace en la red 102 para transmitir datos a máxima capacidad. Con

15 base en esta información, el componente programador 118 puede producir una indicación al solicitante que no es capaz de completar la solicitud de transferencia de datos antes de la fecha límite (puesto que el enlace no soporta la capacidad incrementada producida por la solicitud de transferencia de datos). El componente programador 118

20 puede llevar a cabo de forma adicional otro análisis cuando programa las transferencias de datos. Por ejemplo, cuando opera en una nube pública, el componente programador 118 se puede configurar para determinar si aceptar o rechazar solicitudes recibidas para aumentar al máximo las ganancias, reducir pérdida, etc.

25 El dispositivo de cómputo controlador 100 puede incluir

además un componente fijador de precios 128 que puede exponer un programa de fijación de precios a los clientes con base en el programa a largo plazo 124. Por ejemplo, cuando al menos uno de los dispositivos de cómputo 104-110 está incluido en un centro de

5 datos que expone recursos por medio de una nube pública, los clientes de la nube pública pueden pagar tarifas por transferencia de datos. El componente fijador de precios 128 puede fijar precios para transferencias de datos como una función de (duraciones de) fechas límite en las solicitudes para transferencias de datos, horas en las

10 que se van a completar las transferencias, etc. Por ejemplo, el componente fijador de precios 128 puede establecer un precio más bajo por unidad de datos transferida cuando la fecha límite de una solicitud de transferencia de datos es distal en el tiempo de una hora actual. El componente fijador de precios 128 puede calcular el

15 programa de precios para manejar la utilización de la red a un punto de operación deseado. Es decir, el componente fijador de precios 128 puede establecer precios para manipular la demanda para transferencias de datos en la red 102 con base en las solicitudes de transferencias de datos basadas en fechas límite. Por lo tanto se

20 puede incentivar a los iniciadores de solicitudes de transferencias de datos a proporcionar fechas límite más largas para lograr costo reducido. Además, el componente fijador de precios 128 se puede configurar para aceptar o rechazar ofertas de precios; por ejemplo, una solicitud puede identificar múltiples fechas límite, cada fecha

25 límite con un precio particular (por ejemplo, por unidad que se va a

transferir) relacionado con la misma. Por ejemplo, la solicitud puede indicar que si la transferencia se completa por primera vez, el solicitante está dispuesto a pagar un primer precio, si la transferencia se completa por segunda vez, el solicitante está
5 dispuesto a pagar un segundo precio, y así sucesivamente. El componente fijador de precios 128 puede actuar para seleccionar el precio (y de esta forma la fecha límite), y se puede calcular el programa a largo plazo con base en las acciones del componente fijador de precios 128.

10 Con referencia ahora a la figura 2, se muestra una representación de ejemplo del programa a largo plazo 124 calculado por el componente programador 118. Como se puede comprobar, el programa a largo plazo 124 cubre una pluralidad de unidades de tiempo (unidad de tiempo 1-unidad de tiempo P). En consecuencia, el
15 programa a largo plazo 124 se puede percibir como que comprende una pluralidad de subprogramas 202-210, un subprograma para cada unidad de tiempo. Cada subprograma en los subprogramas 202-210 puede definir cuántos datos van a fluir a través de la red 102 durante su unidad de tiempo respectiva. Por ejemplo, el primer subprograma
20 202 en el programa a largo plazo 124 puede definir cuantos datos van a fluir a través de la red 102 durante la primera unidad de tiempo (por ejemplo, durante los 5 minutos representados por la primera unidad de tiempo). En consecuencia, el primer subprograma 202 puede definir, para cada transferencia de datos entre un dispositivo
25 de cómputo fuente y un dispositivo de cómputo receptor (flujo de

tráfico), una velocidad a la cual la fuente va a producir los datos y las rutas a través de las cuales los datos producidos van a viajar para alcanzar el dispositivo de cómputo receptor. Por lo tanto, el primer subprograma 202 puede configurar (o especificar) la velocidad y las rutas de diferentes flujos de tráfico, cada flujo de tráfico que tiene su propio dispositivo de cómputo fuente y receptor, y cada flujo de tráfico que tiene rutas definidas a través de las cuales el flujo de tráfico va a viajar del dispositivo de cómputo fuente al dispositivo de cómputo receptor. Como se indicó anteriormente, las unidades de tiempo 1-P pueden tener una duración común, que, en un ejemplo, puede ser entre 1 minuto y 5 minutos. En otro ejemplo, las unidades de tiempo 1-P pueden tener diferentes duraciones. Por ejemplo, la primera unidad de tiempo puede tener una duración más corta que la duración de la P-ésima unidad de tiempo. De acuerdo con un ejemplo particular, la primera unidad de tiempo para el primer subprograma 202 puede tener una duración de tiempo de 1 minuto, en tanto que la P-ésima unidad de tiempo para el P-ésimo programa 210 puede tener una duración de tiempo de 10 minutos.

Con referencia ahora a la figura 3, se ilustra una representación de ejemplo del programa a corto plazo 126. El programa a corto plazo 126 cubre una unidad de tiempo inmediatamente posterior (unidad de tiempo 0). El programa a corto plazo 126 comprende una pluralidad de tablas de encaminamiento 302-306 que corresponden respectivamente a la pluralidad de dispositivos de infraestructura de red 112-114, y una pluralidad de

instrucciones 308-312 que corresponden respectivamente a la pluralidad de dispositivos de cómputo 104-110. La pluralidad de instrucciones 308-312 define cuáles de los dispositivos de cómputo 104-110 van a comenzar a producir datos para satisfacer una solicitud de transferencia de datos, y las velocidades a las cuales los dispositivos de cómputo van a producir estos datos. El componente programador 118 puede transmitir las tablas de encaminamiento 302-306 a los dispositivos de infraestructura de red respectivos 112-114, y el componente programador 118 puede transmitir las instrucciones 308-312 a los dispositivos de cómputo respectivos 104-110. Por lo tanto, la primera tabla de encaminamiento 302 se transmite al primer dispositivo de infraestructura de red 112, y la N-ésima tabla de encaminamiento 306 se transmite al N-ésimo dispositivo de infraestructura de red 114. El primer dispositivo de infraestructura de red 112, que responde a la recepción de la primera tabla de encaminamiento 302, se configura (para la unidad de tiempo 0) con la tabla de encaminamiento 302, y encamina los datos recibidos por el dispositivo de infraestructura de red 112 de acuerdo con los contenidos de la primera tabla de encaminamiento 302. Del mismo modo, las primeras instrucciones 310 se transmiten al primer dispositivo de cómputo 304, y las Z-ésimas instrucciones 312 se transmiten al Z-ésimo dispositivo de cómputo 110. En consecuencia, en un ejemplo, el primer dispositivo de cómputo 104, que responde a la recepción de las primeras instrucciones 310, produce datos de acuerdo con las primeras instrucciones 310.

Con referencia ahora a la figura 4, se presenta una representación detallada del componente programador 118. El componente programador 118 puede acceder al repositorio de datos 120 y recuperar el mapa de red 122. El componente programador 118
5 comprende un componente constructor de gráfico de red 401 que recibe el mapa de red 122 y construye un gráfico de red 402 con base en el mapa de red 122 y varias unidades de tiempo cubiertas por el programa a largo plazo 124. El componente programador 118 provoca que se retenga el gráfico de red 402 en el repositorio de
10 datos 120. Cuando se construye el gráfico de red 402, el componente constructor de gráfico de red 401 puede generar múltiples casos del mapa de red 122: un caso para cada unidad de tiempo cubierta por el programa a largo plazo 124. El componente constructor de gráfico de red 401 puede acoplar diferentes casos del mapa de red 122 al
15 generar bordes que acoplan nodos en los casos del mapa de red 122 que representan un mismo dispositivo.

Con referencia brevemente a la figura 5, se ilustra un gráfico de red de ejemplo 500 que se puede construir por el componente constructor de gráfico de red 401. El gráfico de red de ejemplo 500
20 comprende un primer caso del mapa de red 501a para una primera unidad de tiempo y un segundo caso del mapa de red 501b para una segunda unidad de tiempo. El primer caso 501a del mapa de red 122 comprende una primera pluralidad de nodos 502a-508a que son representativos respectivamente de los dispositivos de cómputo 104-
25 110 de la red 102 y una segunda pluralidad de nodos 510a-512a que

son representativos respectivamente de los dispositivos de infraestructura de red 112-114. El primer caso 501a del mapa de red 122 incluye bordes (mostrados en línea sólida) que representan enlaces físicos entre dispositivos en la red 102. El segundo caso 501b del mapa de red 122 incluye además estos bordes. En un ejemplo, se pueden ponderar bordes de forma diferente en los casos del mapa de red 122. Por ejemplo, puede ser deseable reservar más ancho de banda a través de un enlace particular en la red 102 para la primera unidad de tiempo cuando se compara con la segunda unidad de tiempo. Esto se puede representar en el gráfico de red 500 a través de la asignación de diferentes pesos al enlace en el primer caso 501a del mapa de red 122 y el segundo caso 501b del mapa de red 122.

La gráfico de red 402 también incluye una pluralidad de bordes 514-524 que acoplan nodos entre los casos 501a y 501b del mapa de red 122 que representan mismos dispositivos. Por ejemplo, un primer borde 514 acopla el nodo 504a con el nodo 504b, en donde tanto 504a como 504b representan el segundo dispositivo de cómputo 106. De forma similar, un segundo borde 516 acopla el nodo 502a al nodo 502b, en donde ambos nodos 502a y 502b representan el primer dispositivo de cómputo 104 en la red 102. En una modalidad de ejemplo, se pueden asignar pesos relativamente altos a los bordes 514-524. En consecuencia, la red 102 se puede representar en varias unidades de tiempo diferentes por el gráfico de red 402. Los bordes mostrados en forma sólida se pueden ponderar con base en,

por ejemplo, la capacidad de los enlaces de red físicos representados por estos bordes, demandas históricas de los enlaces físicos durante las unidades de tiempo, alguna combinación de los mismos, o similares. En tanto que no se muestra, si el gráfico de red 5 500 incluyera un tercer caso del mapa de red 122 que corresponde a una tercera unidad de tiempo, cada nodo en el tercer caso del mapa de la WAN 122 se acoplaría a todos los nodos en el gráfico de red 500 que representa el mismo dispositivo.

Regresando a la figura 4, el componente programador 118 10 puede incluir además un componente limitador de ruta 404 que recibe el gráfico de red 402 y aplica limitaciones basadas en rutas al gráfico de red 402. Por ejemplo, para reducir la complejidad del cálculo, el componente limitador de ruta 404 puede limitar un número de rutas potenciales entre una fuente y destino a algún número 15 umbral de rutas, puede limitar las rutas con base en un número de saltos de red entre un dispositivo de cómputo fuente y un dispositivo de cómputo destino (por ejemplo, no se consideran las rutas de red que tienen más de un número umbral de saltos entre una fuente y destino cuando se determinan las rutas entre la fuente y el receptor), 20 u otra limitación adecuada. En respuesta a la aplicación de limitaciones al gráfico de red 402, el componente programador 118 puede almacenar una gráfica limitada 406 en el repositorio de datos 120.

El componente programador 118 puede incluir además un 25 componente optimizador 408 que recibe la gráfica limitada 406 y los

datos con respecto a todas las solicitudes de transferencias de datos pendientes (por ejemplo, solicitudes de transferencias de datos previamente recibidas y aceptadas, donde están incompletas sus transferencias de datos respectivas), y produce el programa a largo

5 plazo 124 con base en las solicitudes de transferencias de datos pendientes, sus estados respectivos (por ejemplo, cantidades de datos que se van a transferir, cantidades de tiempo antes de las fechas límite respectivas de las solicitudes, etc.), y la gráfica limitada 406. Como se indicó anteriormente, el componente

10 optimizador 408 puede ejecutar un proceso de optimización adecuado para calcular el programa a largo plazo. En una modalidad de ejemplo, el proceso de optimización puede incluir la ejecución de un algoritmo de empaquetamiento y cobertura mezclado, en donde el componente optimizador 408 calcula el programa a largo plazo 124

15 para (por ejemplo) suavizar la utilización de la red con el paso del tiempo y satisfacer las fechas límite de las solicitudes. En una modalidad de ejemplo, el componente optimizador 408 puede ejecutar una versión paralelizada de un algoritmo de empaquetamiento y envoltura mezclado. La paralelización puede ser

20 necesaria para permitir el cálculo repetido del programa a largo plazo 124 con el paso del tiempo conforme cambian las circunstancias. Se pueden utilizar otros tipos de enfoques de optimización, que incluyen enfoques de aumento al máximo de expectativa, procesos de programación lineales, y similares para

25 generar el programa a largo plazo 124. El componente optimizador

408 puede generar el programa a corto plazo 126 con base en el programa a largo plazo 124.

Ahora se exponen los detalles adicionales relacionados con un proceso de optimización de ejemplo que se puede ejecutar por el componente optimizador 408. En general, el componente optimizador 408 recibe una gráfica implementada por computadora $G = (V, E)$, donde $|V| = n$ y $|E| = m$, y las capacidades de borde no negativas $c: E \rightarrow \mathbb{R}_+$. Se pueden definir solicitudes a largo plazo por la tupla $(\tilde{U}_i, b_i, d_i, D_i, s_i, t_i, P_i)$, donde a_i es el tiempo de percepción en el cual el componente optimizador 408 percibe la solicitud, b_i es el tiempo de inicio a partir del cual se puede programar la solicitud, d_i es la fecha límite después de la cual no se puede programar la solicitud, D_i es la demanda de la solicitud, s_i es el nodo fuente desde el cual se debe encaminar la solicitud, t_i es el nodo objetivo al cual se debe encaminar la solicitud, y P_i es la recolección de rutas admisibles de s_i a t_i de la solicitud.

Cuando comienza el paso del tiempo $\tilde{U}$ el componente optimizador 408 percibe todas las solicitudes a largo plazo i para las cuales $a_i = \tilde{U}$. Además, el componente optimizador 408 puede tener una estimación $X_{e, \tilde{U}}$ para la fracción de la capacidad c_e del borde e que puede ser necesaria por solicitudes de alta prioridad (por ejemplo, *ad hoc*) en el tiempo $\tilde{U}$. Ya que $X_{e, \tilde{U}}$ es una estimación, la fracción real de la capacidad necesaria para atender las solicitudes *ad hoc* se da a conocer al comienzo del tiempo $\tilde{U}$ (junto con solicitudes a largo plazo hechas en ese momento).

Como se indicó anteriormente, el componente optimizador 408 puede tratar el problema de programación como un problema de empaquetamiento y cobertura mezclado, y puede utilizar de esta forma limitaciones de empaquetamiento y cobertura para generar el programa. Por ejemplo, el componente optimizador 408 puede utilizar variables $f_{i,p,\tilde{U}}$ que corresponden a la cantidad de flujo asignado para la solicitud i en la ruta $p \in P_i$ en el tiempo $\tilde{U}$ donde $b_i \leq \tilde{U} \leq d_i$. Utilizando estas variables, se pueden formular desigualdades de empaquetamiento y cobertura lineales que aseguran la factibilidad de la asignación; en cada momento y para cada borde el flujo total asignado en ellos no excede su capacidad; cada solicitud a largo plazo i se asigna a al menos una fracción $\tilde{U}_i$ de su demanda D_i hasta su fecha límite.

Estas desigualdades evolucionan y cambian con el paso del tiempo por varias razones: llegan nuevas solicitudes a largo plazo; se rebela la realización estocástica de uso de alta prioridad de bordes (por ejemplo, para atender solicitudes *ad hoc*); y asignación insuficiente de bordes en el futuro lejano disminuye conforme el tiempo avanza. El componente optimizador 408 ajusta las variables $f_{i,p,\tilde{U}}$ en respuesta al cambio de las desigualdades, adaptando por lo tanto el programa. Se va a señalar que este enfoque mejora los esquemas que solamente calculan programas para el caso de hora actual, ya que el componente optimizador 408 programa los flujos relativamente lejos en el futuro, permitiendo que se reciba la salida de una $\tilde{U}_i$ prometida en la fracción de la demanda de la i -ésima

solicitud.

Además, como se indicó, el componente optimizador 408 puede corregir solicitudes de transferencias de datos de alta prioridad (*ad hoc*). $X_{e,\tilde{U}}$ es una estimación de la fracción de la capacidad del borde e necesario por solicitudes de alta prioridad en el tiempo $\tilde{U}$. Por lo tanto, cuando se formulan las desigualdades anteriores, en lugar de utilizar la capacidad c_e en el tiempo $\tilde{U}$ el componente optimizador 408 puede utilizar $(1-X_{e,\tilde{U}}) \cdot c_e$. Una vez que se ha alcanzado el tiempo $\tilde{U}$ y se observa la realización de esta fracción, el componente optimizador 408 puede descubrir que el borde e se usa en exceso, por ejemplo, el flujo total en e excede su capacidad. Cuando esto ocurre, el componente optimizador 408 puede reducir algunos flujos preprogramados $f_{i,p,\tilde{U}}$ para los cuales $e \in p$. Esto se puede realizar al formular un pequeño programa lineal (cuyo tamaño depende solamente del tamaño de la gráfica implementada por computadora original G y no de $\tilde{U}$) para reducir al mínimo el efecto en la $\tilde{U}$ prometida producida por el componente optimizador 408.

Además, el componente optimizador 408 se puede configurar para infrautilizar bordes que son lejanos en el futuro (por ejemplo, limitar la gráfica G), y disminuir esta limitación conforme el tiempo avanza. Por ejemplo, se puede utilizar un parámetro $b_{e,t,\tilde{U}}$ para determinar la fracción de $(1-X_{e,\tilde{U}}) \cdot c_e$ que se puede utilizar por el componente optimizador 408 en el tiempo $\tilde{U}$. Conforme avanza el tiempo $\tilde{U}$, $b_{e,t,\tilde{U}}$ puede incrementar (por ejemplo, disminuye la restricción de infrautilización).

En una modalidad de ejemplo, el componente optimizador 408 se puede configurar para aumentar al máximo la peor $\bar{U}$ prometida, que corresponde a un objetivo de "equidad". A pesar del aumento al máximo de $\bar{U}$, algunas porciones de la red 102 se pueden

5 infrautilizar, y se puede asignar más flujo. Para abordar este problema, el componente optimizador 408 puede emplear una función de utilidad secundaria; por ejemplo, aumentar al máximo la $\bar{u}$, prometida promedio, lo que conduce a un programa que tiene mayor utilización de red. Esta utilidad se puede formular como una

10 desigualdad de cobertura.

Se puede comprobar que las solicitudes a largo plazo que son extremadamente largas, por ejemplo, $d_i - a_i$ es muy grande, pueden retardar el tiempo de ejecución del componente optimizador 408. A fin de facilitar el cálculo de una solución relativamente rápido, el

15 componente optimizador 408 puede emplear un enfoque de ventana deslizante, donde el componente optimizador 408 considera un número fijo de pasos de tiempo en el futuro. Los solicitantes cuyas fechas límite respectivas están más allá de la ventana deslizante se pueden dividir en solicitudes más pequeñas. En este caso, el

20 componente optimizador 408 puede actualizar la demanda de una solicitud más pequeña con base en su demanda original y el flujo total ya asignado.

Ahora se exponen detalles adicionales relacionados con el diseño y la operación del componente optimizador 408. Para

25 propósitos de explicación, se derivan procesos en etapas,

comenzando desde el caso sin conexión, procediendo al caso en línea, y entonces añadiendo características adicionales tal como la suavización del tráfico por medio de los parámetros b , gracia o violación de limitaciones de capacidad, división de solicitudes debido a horizonte acotado, y manejo de diferentes prioridades. Además, en tanto que los ejemplos provistos en la presente hacen referencia al cálculo de programas de flujo de tráfico en la red, se va a entender que el enfoque de empaquetamiento y cobertura mezclado descrito en la presente se puede utilizar en otros escenarios. Por ejemplo, se contempla modificación dinámica de desigualdades conforme se reciben datos para producir una solución. Además, en general se contempla la paralelización de algoritmos de empaquetamiento y cobertura mezclados, y al hacerlos para calcular un programa a largo plazo para flujos de tráfico de red no es más que un escenario de ejemplo donde puede ser deseable paralelizar un algoritmo de empaquetamiento y cobertura mezclado. Aún más, se contempla el enfoque general de la estimación de tráfico de alta prioridad en una red y la asignación de ancho de banda restante de enlaces de red para satisfacer un objetivo. En este caso, la asignación del ancho de banda de los enlaces se puede actualizar de forma dinámica cuando el tráfico de alta prioridad real difiere de la estimación, en tanto que se presenta una mínima interrupción del objetivo.

Como se indicó anteriormente, $G = (V, E)$, donde $|V| = n$ y $|E| = m$, y capacidades de borde no negativas $c : E \rightarrow \mathbb{R}_+$. Se pueden definir solicitudes a largo plazo por la tupla $(a_i, b_i, d_i, D_i, s_i, t_i, P_i)$. Además,

a menos que se especifique lo contrario, se puede asumir que existe un tamaño absoluto T (que puede ser grande) que delimita las fechas límite respectivas de las solicitudes, por ejemplo $T \geq d_i \forall 1 \leq i \leq k$.

Ahora se describe el componente optimizador 408 en el caso sin conexión, donde se asume que el componente optimizador 408 percibe todas las solicitudes desde el comienzo, por ejemplo, que $a_i = 0$ para todas las solicitudes $1 \leq i \leq k$. Dada la gráfica G y todas las k solicitudes, puede ser deseable ubicar la α más deseable, $0 \leq \alpha \leq 1$, de tal forma que una fracción de al menos α de la demanda de cada solicitud se puede encaminar en tanto que se respetan las limitaciones de capacidad. Esto es llamado como el problema de flujo de multiservicio concurrente, un caso especial del problema de resolver de forma fraccionada un sistema de limitaciones de empaquetamiento/cobertura lineales. Dada α , el sistema de desigualdades lineales necesario para resolver el caso sin conexión (sin utilidad) por un programa lineal es como sigue:

$$\begin{aligned} \sum_{i: b_i \leq t \leq d_i} \sum_{p \in \mathcal{P}_i: e \in p} f_{i,p,t} &\leq c_e & \forall e \in E, \forall 0 \leq t \leq T \\ \sum_{t=b_i}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t} &\geq \alpha \cdot D_i & \forall 1 \leq i \leq k \\ f_{i,p,t} &\geq 0 & \forall 1 \leq i \leq k, \forall p \in \mathcal{P}_i, \forall 0 \leq t \leq T \end{aligned}$$

En el ejemplo expuesto anteriormente, el proceso de optimización busca de forma iterativa la mejor α (por ejemplo, utilizando métodos de búsqueda lineal). En cada iteración, se determina si el programa lineal es factible o no; si es factible, se

puede producir una solución factible. Se presenta más adelante un proceso de ejemplo para resolver el problema de factibilidad.

Inicialización

$$5 \quad \begin{cases} f_{i,p,t} \leftarrow 0 & \forall 1 \leq i \leq k, \forall p \in \mathcal{P}_i, \forall b_i \leq t \leq d_i \\ y_{e,t} \leftarrow 1 & \forall e \in E, \forall 0 \leq t \leq T \\ z_i \leftarrow 1 & \forall 1 \leq i \leq k \\ R \leftarrow \{1, \dots, k\} \end{cases}$$

Mientras que $\exists i \in R$ s. t. $\sum_{t=b_i}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t} < \alpha \cdot D_i$ hacer

Encontrar $i^* \in R, b_{i^*} \leq t^* \leq d_{i^*}$, y $p^* \in \mathcal{P}_{i^*}$ s. t.:

$$10 \quad \frac{\sum_{e \in p^*} \frac{y_{e,t^*}}{c_e}}{\sum_{e \in E} \sum_{t=0}^T y_{e,t}} \leq \frac{\frac{1}{\alpha D_{i^*}} z_{i^*}}{\sum_{i \in R} z_i} (*)$$

Si no existe ese i^*, t^* , y p^* , entonces

abortar y regresar que no existe ninguna solución factible

De otro modo

$$15 \quad \gamma \leftarrow \varepsilon \cdot \min \left\{ \alpha \cdot D_{i^*}, \min_{e \in p^*} \{c_e\} \right\};$$

$$f_{i^*,p^*,t^*} \leftarrow f_{i^*,p^*,t^*} + \frac{\varepsilon}{N} \cdot \gamma;$$

$$y_{e,t^*} \leftarrow y_{e,t^*} \cdot e^{\frac{\gamma}{c_e}}, \forall e \in p^*;$$

$$z_{i^*} \leftarrow z_{i^*} \cdot e^{-\frac{\gamma}{\alpha D_{i^*}}};$$

$$20 \quad \text{Si } \sum_{t=b_{i^*}}^{d_{i^*}} \sum_{p \in \mathcal{P}_{i^*}} f_{i^*,p,t} \geq \alpha \cdot D_{i^*} \text{ entonces}$$

$$R \leftarrow R \setminus \{i^*\};$$

regresar $\{f_{i,p,t} : 1 \leq i \leq k, p \in \mathcal{P}_i, b_i \leq t \leq d_i\}$

La inicialización mencionada anteriormente y la actualización de los $y_{e,t}$ s y z_i s se hace de tal forma que se cumple lo siguiente:

$$y_{e,t} = \exp\left(\frac{N}{\varepsilon \cdot c_e} \sum_{i: b_i \leq t \leq d_i} \sum_{p \in \mathcal{P}_i: e \in p} f_{i,p,t}\right) \quad (1)$$

$$z_i = \exp\left(-\frac{N}{\varepsilon \cdot \alpha \cdot D_i} \sum_{t=b_i}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t}\right) \quad (2)$$

5 En cada paso del proceso, se incrementa una variable de flujo individual (elegida de forma cuidadosa) f_{i^*,p^*,t^*} . La variable se puede elegir para que el incremento en la limitación de empaquetamiento $\tilde{\mu}_{\text{peor}}$ (por ejemplo, un enlace en la red 102 cuya capacidad está más cerca de violarse) sea menos que el incremento en la limitación

10 de cobertura $\tilde{\mu}_{\text{peor}}$ (por ejemplo, la solicitud que es la más lejana de su limitación de demanda). Procesos existentes relacionados con problemas de empaquetamiento y cobertura pueden satisfacer este requerimiento a través del uso de aproximaciones suaves de operadores $\max$ y $\min$; el requerimiento de aproximación se

15 manifiesta en la desigualdad (*), donde la $y_{e,t}$ ^s se puede ver como variables internas que representan las limitaciones de empaquetamiento, en tanto que la z_i como variables internas que representan las limitaciones de cobertura.

Las producción de un algoritmo de empaquetamiento y

20 cobertura existente garantiza que se satisfagan las limitaciones de cobertura (asumiendo que $\tilde{U}$ es factible). Debido a que (*) es la relación de las derivadas de las aproximaciones suaves de $\max$ y $\min$ (en lugar de la relación de los cambios en el valor de las limitaciones), la salida viola las limitaciones de empaquetamiento por

25 un factor multiplicativo pequeño. El factor disminuye de forma

monótona a uno conforme el parámetro de precisión γ disminuye a cero. Esta garantía se hace formal en el siguiente teorema.

Teorema 1: Si es factible el programa lineal, entonces para cada $0 < \epsilon \leq 1/2$, la salida del proceso satisface:

5

$$\begin{aligned} \sum_{i: b_i \leq t \leq d_i} \sum_{p \in \mathcal{P}_i: e \in p} f_{i,p,t} &\leq (1 + 3\epsilon) \cdot c_e & \forall e \in E, \forall 0 \leq t \leq T \\ \sum_{t=b_i}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t} &= \alpha \cdot D_i & \forall 1 \leq i \leq k \end{aligned}$$

Teorema 2: Para cada $0 < \epsilon \leq 1/2$, el proceso de comprobación de factibilidad termina después de como máximo

10

$$k + \frac{N}{\epsilon^2} \cdot (k + (1 + 3\epsilon) \cdot m \cdot (T + 1))$$

de iteraciones.

La elección de γ refleja el compromiso necesario entre la precisión del proceso (por cuánto se desvían las limitaciones de empaquetamiento) y el tiempo de ejecución. Una pequeña violación en las limitaciones de capacidad puede ser aceptable cuando los

conceptos de la suavización de tráfico por medio de los parámetros b_i , ya que la violación significa que solamente se excede una fracción preespecificada de la capacidad c_e , en lugar de la capacidad completa.

20

Como se puede determinar, el aumento al máximo de $\tilde{U}$ corresponde a un objetivo de equidad, donde se garantiza que una fracción de $\tilde{U}$ a partir de la demanda de cada solicitud se encamina por su fecha límite. Esto por separado puede ser insuficiente, puesto que es posible que una solución que logra la mejor $\tilde{U}$ puede no utilizar partes de la red y aún se puede encaminar más flujo. En

25

consecuencia, se puede utilizar una función de utilidad adicional, por

ejemplo, el promedio $\alpha: \frac{1}{k} \sum_{i=1}^k \alpha_i$, que se necesita aumentar al máximo a lo largo de todas las soluciones óptimas del objetivo de equidad original (por ejemplo, el aumento al máximo anteriormente mencionado de la $\tilde{U}$ peor).

Cualquier utilidad de la siguiente forma se puede introducir:

$$g(\vec{f}) = \sum_{i=1}^k \sum_{p \in \mathcal{P}_i} \sum_{t=b_i}^{d_i} u_{i,p,t} \cdot f_{i,p,t}, \text{ donde } u_{i,p,t} \geq 0. \text{ En}$$

particular, la utilidad de $\tilde{U}$ promedio anteriormente mencionada se

puede lograr al elegir $u_{i,p,t} = \frac{1}{k \cdot D_i}$. Se puede señalar que $g(\vec{f})$ se

puede utilizar para modelar fechas límite flexibles. Por ejemplo, se

puede fijar una solicitud i , y los coeficientes de utilidad $u_{i,p,t} = h_i(t)$ se

pueden considerar para una función $h_i(t)$ que decae de forma

exponencialmente rápida para $t > D$ (D es algún paso de tiempo).

Estas funciones modelan el hecho de que D es una fecha límite flexible para la solicitud i .

Se puede añadir la siguiente limitación al programa lineal

$$g(\vec{f}) \geq U$$

mencionado anteriormente: para alguna U de

estimación del valor máximo de g . Ya que un algoritmo de

empaquetamiento y cobertura convencional funciona para cualquier

limitación de empaquetamiento y cobertura, es posible la adición de

la limitación de cobertura lineal anterior. Dada la limitación, se

pueden realizar los siguientes cambios al proceso mencionado anteriormente: se añade una variable r interna; r se inicializa a 1; cuando se incrementa f_{i^*,p^*,t^*} , r se actualiza como sigue:

$$r \leftarrow r \cdot e^{-\frac{u_{i^*,p^*,t^*} \gamma}{F}}; \quad \text{se cambia la condición (*) a:}$$

$$\frac{\sum_{e \in p^*} \frac{y_{e,t^*}}{c_e}}{\sum_{e \in E} \sum_{t=0}^T y_{e,t}} \leq \frac{\frac{1}{\alpha \cdot D_{i^*}} z_{i^*} + \frac{u_{i^*,p^*,t^*}}{F} \cdot r}{\sum_{i \in R} z_i + r};$$

cuando se satisface la nueva limitación por primera vez, es decir, $g(\vec{f}) \geq F$, se remueve la variable interna r de (*) y ya no se actualiza más; la condición de parada del proceso es que se satisfagan todas las limitaciones de cobertura (una para cada

solicitud i y la limitación de utilidad adicional $g(\vec{f}) \geq F$); Ó se elige

por la siguiente fórmula:

$$\gamma \leftarrow \varepsilon \cdot \min \left\{ \alpha \cdot D_{i^*} \min_{e \in p^*} \{c_e\}, \frac{F}{u_{i^*,p^*,t^*}} \right\}.$$

La

variable r se inicializa como de la forma anterior de tal forma que se cumple lo siguiente:

$$r = \exp \left(\frac{N}{\varepsilon \cdot F} \sum_{i=1}^k \sum_{p \in \mathcal{P}_i} \sum_{t=b_i}^{d_i} u_{i,p,t} \cdot f_{i,p,t} \right). \quad (3)$$

Ahora se expone la operación del componente optimizador 408 en el caso en línea -donde el componente optimizador 408 recibe solicitudes y adapta el programa a largo plazo en respuesta a la

recepción de la solicitud. La suposición realizada en el caso sin conexión que el componente optimizador 408 percibe todas las solicitudes en el tiempo $t = 0$, por ejemplo, $a_i \in 0$, puede ser poco razonable. Un proceso de ejemplo que se puede emplear por el

5 componente optimizador 408 cuando se utiliza en línea tiene tres propiedades principales: permite el manejo en línea de solicitudes, hace cambios incrementales a la solución conforme llegan nuevas solicitudes, y actualiza los valores de $\tilde{U}$ prometidos de forma progresiva.

10 Con respecto a este proceso, se puede indexar el tiempo por $\tilde{U}$. Cuando inicia el paso de tiempo $\tilde{U}$ el proceso empleado por el componente optimizador 408 percibe las solicitudes i cuyo tiempo de percepción es $\tilde{U}$ por ejemplo, $a_i = \tilde{U}$. La recolección de solicitudes que son relevantes en el tiempo $\tilde{U}$ se puede denotar por $R(\tilde{U}) = \{i: a_i \leq \tilde{U} \text{ y } d_i \leq \tilde{U}\}$. En la presente la derivación de este proceso se describe en

15 etapas.

Se presenta un primer diseño, en donde el proceso es incremental en naturaleza y elige los valores de $\tilde{U}$ con avidez. El sistema de desigualdades del proceso en línea utilizado para

20 resolver en el tiempo $\tilde{U}$ por el programa lineal $LP(\tilde{U})$ se denota como sigue:

$$\sum_{\substack{i: i \in R(\tau) \\ \max\{\tau, b_i\} \leq t \leq d_i}} \sum_{\substack{p \in \mathcal{P}_i: \\ e \in p}} f_{i,p,t} \leq c_e \quad \forall e \in E, \forall \tau \leq t \leq T$$

$$F_{i,\tau-1} + \sum_{t=\max\{\tau, b_i\}}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t} \geq \alpha_{a_i} \cdot D_i \quad \forall i \in R(\tau)$$

5

$$f_{i,p,t} \geq 0 \quad \forall i \in R(\tau), \forall \max\{\tau, b_i\} \leq t \leq d_i, \forall p \in \mathcal{P}_i.$$

$$F_{i,\tau-1} = \sum_{t=b_i}^{\tau-1} \sum_{p \in \mathcal{P}_i} f_{i,p,t}$$

se puede definir como el flujo total ya

10 encaminado por el proceso antes del tiempo $\tilde{U}$ para la solicitud $i \notin R(\tilde{U})$, y por $\tilde{U}_i$, la fracción prometida de la demanda de la solicitud i que el proceso programó. Se puede señalar que $F_{i,\tilde{U}-1}$ es una constante cuando se opera en el paso de tiempo $\tilde{U}$

El proceso en línea en el tiempo $\tilde{U}$ utiliza la solución del paso
15 de tiempo anterior como su punto de inicio. Después de eso, dados los valores para $\tilde{U}_i$ s, el proceso ejecuta el algoritmo sin conexión para determinar si existe una solución factible o no para los valores de $\tilde{U}_i$ dados. El proceso puede determinar los valores de $\tilde{U}_i$, ya que las solicitudes i que no son nuevas, por ejemplo $\tilde{U}_i < \tilde{U}$ ya tienen un
20 valor de $\tilde{U}_i$ prometido del paso de tiempo anterior. Las solicitudes recién percibidas i , por ejemplo, $\tilde{U}_i = \tilde{U}$ se pueden configurar de forma inicial para tenerse en $\tilde{U}_i = 0$. El proceso puede conducir un proceso de llenado de agua, en el cual se incrementan las puntuaciones de $\tilde{U}$ más bajas siempre y cuando $LP(\tilde{U})$ sea factible.

25 Con respecto al proceso sin conexión, se pueden realizar los

siguientes cambios: se cambia la condición (*) a:

$$\frac{\sum_{e \in p^*} \frac{y_{e,t^*}}{c_e}}{\sum_{e \in E} \sum_{t=0}^T y_{e,t}} \leq \frac{\frac{1}{\alpha \cdot D_{i^*} - F_{i^*,\tau-1}} \cdot z_{i^*}}{\sum_{i \in R} z_i};$$

5 cuando se incrementa $f_{i^*,p^*,\tilde{U}^*}$ la variable z_{i^*} se actualiza como

sigue:
$$z_{i^*} \leftarrow z_{i^*} \cdot e^{-\frac{\gamma}{\alpha \cdot D_{i^*} - F_{i^*,\tau-1}}};$$
 Ó se cambia de acuerdo con

$$\gamma \leftarrow \varepsilon \cdot \min \left\{ \alpha \cdot D_{i^*} - F_{i^*,\tau-1}, \min_{e \in p^*} \{c_e\} \right\};$$

10 i^* se remueve de R cuando:

$$F_{i^*,\tau-1} + \sum_{t=\max\{\tau, b_{i^*}\}}^{d_{i^*}} \sum_{p \in \mathcal{P}_{i^*}} f_{i^*,p,t} \geq \alpha \cdot D_{i^*};$$

la condición de parada del algoritmo es que se satisfagan todas las ñuevasò limitaciones de cobertura. De forma similar a la
 15 ecuación (1) y la ecuación (2), $y_{e,t}$ y z_i se mantienen para apoyar las siguientes definiciones:

$$y_{e,t} = \exp \left(\frac{N}{\varepsilon \cdot c_e} \sum_{\substack{i: i \in R(\tau) \\ \max\{\tau, b_i\} \leq t \leq d_i}} \sum_{\substack{p \in \mathcal{P}_i \\ e \in p}} f_{i,p,t} \right) \quad (4)$$

$$20 \quad z_i = \exp \left(-\frac{N}{\varepsilon \cdot (\alpha \cdot D_i - F_{i,\tau-1})} \sum_{t=\max\{\tau, b_i\}}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t} \right). \quad (5)$$

Un problema que existe con la solución incremental en línea anterior es que se elige esencialmente los valores de $\tilde{U}$ de una forma con aidez, que puede provocar una amplia variación en la secuencia $\tilde{U}_0, \tilde{U}_1, \dots$, y algunas $\tilde{U}$ s pueden tener un valor muy pequeño.

25 A fin de sostener la equidad con el paso del tiempo, se puede

introducir el concepto de suavización de tráfico. De forma intuitiva, esto se puede realizar al limitar la capacidad que puede utilizar el proceso en línea incremental, dejando cierta capacidad vacante para solicitudes futuras. Formalmente, $b_{e,t,\tilde{U}}$ puede denotar la fracción de la capacidad de borde e en el tiempo futuro t que se puede utilizar por el proceso incremental en línea cuando se ejecuta en el paso del tiempo $\tilde{U}$. Esto cambia la limitación de capacidad de $LP(\tilde{U})$ como sigue:

$$10 \quad \sum_{\substack{i:i \in R(\tau) \\ \max\{\tau, b_i\} \leq t \leq d_i}} \sum_{\substack{p \in \mathcal{P}_i: \\ e \in p}} f_{i,p,t} \leq \beta_{e,t,\tau} c_e \quad \forall e \in E, \forall \tau \leq t \leq T$$

Dado un borde $e \notin E$ y un tiempo t , la siguiente función es no decreciente en $\tilde{U}$ $b_{e,t,\tilde{U}}[0, \epsilon, t] \in [0, 1]$. $b_{e,t,t}$ se puede establecer a 1, ya que cuando el proceso en línea alcanza el paso de tiempo $\tilde{U} = t$, se puede utilizar la capacidad completa del borde puesto que no existe ninguna razón de ahorrar capacidad vacante para usos futuros. Por ejemplo, una elección de ejemplo es $b_{e,t,\tilde{U}} = \exp(-(t-\tilde{U})/c)$ para alguna constante c .

Con base en lo anterior, se pueden implementar los siguientes cambios a la ecuación (1):

se puede actualizar una regla para y_{e,t^*} , en donde

$$y_{e,t^*} \leftarrow y_{e,t^*} \cdot e^{\frac{\gamma}{\beta_{e,t^*,\tau} c_e}} \forall e \in p^* ; \text{ la condición (*) se cambia a:}$$

$$\frac{\sum_{e \in p^*} \frac{y_{e,t^*}}{\beta_{e,t^*,\tau} \cdot c_e}}{\sum_{e \in E} \sum_{t=0}^T y_{e,t}} \leq \frac{\frac{1}{\alpha \cdot D_{i^*} - F_{i^*,\tau-1}} \cdot z_{i^*}}{\sum_{i \in R} z_i},$$

5 $y_{e,t}$ se inicializa y se actualiza para mantener la siguiente definición:

$$y_{e,t} = \exp \left(\frac{N}{\varepsilon \cdot \beta_{e,t,\tau} \cdot c_e} \sum_{\substack{i: i \in R(\tau) \\ \max\{\tau, b_i\} \leq t \leq d_i}} \sum_{\substack{p \in \mathcal{P}_i \\ e \in p}} f_{i,p,t} \right). \quad (6)$$

En el caso en línea cuando se opera en el tiempo $\tilde{U}$ se pueden
 10 mejorar los resultados obtenidos por el enfoque de suavización de tráfico. Ya que el componente optimizador 408, cuando emplea el proceso en línea, aplica el llenado de agua en cada paso de tiempo $\tilde{U}$ puede ser el caso que se pueden mejorar algunos de los valores de $\tilde{U}$. En cada paso de tiempo $\tilde{U}$ después del proceso en línea
 15 (cuando termina la suavización de tráfico), se puede aplicar el mismo enfoque algorítmico con una limitación de utilidad. $LP'(\tilde{U})$ puede denotar el sistema de desigualdades que el proceso en línea necesita resolver en el tiempo $\tilde{U}$ con la utilidad $g(\vec{f})$:

$$20 \quad \sum_{\substack{i: i \in R(\tau) \\ \max\{\tau, b_i\} \leq t \leq d_i}} \sum_{\substack{p \in \mathcal{P}_i \\ e \in p}} f_{i,p,t} \leq \beta_{e,t,\tau} c_e \quad \forall e \in E, \forall \tau \leq t \leq T$$

$$F_{i,\tau-1} + \sum_{t=\max\{\tau, b_i\}}^{d_i} \sum_{p \in \mathcal{P}_i} f_{i,p,t} \leq D_i \quad \forall i \in R(\tau)$$

$$\sum_{i \in R(\tau)} \left(U_{i,\tau-1} + \sum_{p \in \mathcal{P}_i} \sum_{t=\max\{\tau, b_i\}}^{d_i} u_{i,p,t} \cdot f_{i,p,t} \right) \geq U$$

$$25 \quad f_{i,p,t} \geq 0 \quad \forall i \in R(\tau), \forall \max\{\tau, b_i\} \leq t \leq d_i, \forall p \in \mathcal{P}_i.$$

$$U_{i,\tau-1} = \sum_{p \in \mathcal{P}_i} \sum_{t=b_i}^{\tau-1} u_{i,p,t} \cdot f_{i,p,t}$$

se puede definir como la utilidad total del flujo ya encaminado por el proceso antes del tiempo $\tilde{U}$ para la solicitud $i \notin R(\tilde{U})$. Se puede señalar que $U_{i,0-1}$ es una limitación cuando se opera en el paso de tiempo $\tilde{U}$

La segunda limitación indica que el flujo total de la solicitud i no debe exceder su demanda total D_i (esta es una limitación de empaquetamiento). La tercera limitación indica que la utilidad total es al menos algún valor F dado (ésta es una limitación de cobertura). Se puede señalar que la segunda limitación no es necesaria en LP y LP($\tilde{U}$), pero es necesaria aquí con LP'($\tilde{U}$).

Cuando se consideran estas limitaciones, se pueden incorporar los siguientes cambios en la ecuación (1) en el componente optimizador 408: se pueden añadir variables internas s_i , donde $\forall i \notin R(\tilde{U})$; cuando se incrementan $f_{i^*,p^*,\tilde{U}^*}$, s_{i^*} se actualiza como sigue:

$$s_{i^*} \leftarrow s_{i^*} \cdot e^{\frac{\gamma}{D_{i^*} - F_{i^*,\tau-1}}}; \text{ la condición (*) se cambia a:}$$

$$\frac{\sum_{e \in p^*} \frac{\gamma_{e,t^*}}{\beta_{e,t^*,\tau} \cdot c_e} + \frac{s_{i^*}}{D_{i^*} - F_{i^*,\tau-1}}}{\sum_{e \in E} \sum_{t=0}^T \gamma_{e,t} + \sum_{i \in R(\tau)} s_i} \leq \frac{u_{i^*,p^*,t^*}}{U - \sum_{i \in R(\tau)} U_{i,\tau-1}};$$

20

la condición de parada del proceso es que se satisfaga la limitación de utilidad; $\tilde{O}$ se elige para que sea

$$\gamma \leftarrow \varepsilon \cdot \min \left\{ D_{i^*} - F_{i^*,\tau-1}, \min_{e \in p^*} \{c_e\}, \frac{U - \sum_{i \in R(\tau)} U_{i,\tau-1}}{u_{i^*,p^*,t^*}} \right\}.$$

25

Las

variables internas $y_{e,t}$, s_i , y r se mantienen, de tal forma que la ecuación (6) permanece sin cambios, y se cumple lo siguiente:

$$s_i = \exp\left(-\frac{N}{\varepsilon \cdot (D_i - F_{i,\tau-1})} \sum_{p \in \mathcal{P}_i} \sum_{t=\max\{\tau, b_i\}}^{d_i} f_{i,p,t}\right) \quad (7)$$

$$5 \quad r = \exp\left(\frac{N \cdot u_{i,p,t}}{\varepsilon \cdot (U - \sum_{i \in R(\tau)} U_{i,\tau-1})} \sum_{p \in \mathcal{P}_i} \sum_{\max\{\tau, b_i\}}^{d_i} f_{i,p,t}\right) \quad (8)$$

Como se indica en el teorema 1, el componente optimizador
408 puede producir una solución que viola las limitaciones de capacidad por un pequeño factor multiplicativo de a lo mucho $(1+3')$.
10 Esto puede suceder tanto en la suavización de tráfico como la incorporación de utilidad. Esto no es un problema sin embargo, ya que se exceden las unidades de flujo $b_{e,t,\tilde{U}} \bar{C}_e$ en el borde e en el tiempo t programado por el proceso en línea en el tiempo $\tilde{U}$ y la capacidad c_e del borde. Este enfoque permite la aceptación de
15 violaciones de pequeña capacidad generadas por el componente optimizador 408 sin violar realmente la capacidad del todo en la salida final, que da por resultado valores más altos de $\tilde{U}$.

Se puede determinar que el caso en línea no tiene garantías de desempeño teórico, aunque se puede demostrar que cuando es
20 factible el sistema apropiado de desigualdades, entonces no se atascará el proceso utilizado por el componente optimizador 408; es decir, siempre existe una variable f_{i^*,p^*,t^*} que puede incrementar lo que implica que la relación de la derivada de la suavización de la peor limitación de empaquetamiento y la derivada de la suavización
25 de la peor limitación de empaquetamiento se delimita por 1. Esto es

válido para cualquier solución actual $\vec{f}$. Esto se resume en el siguiente teorema:

Teorema 3: Para cualquier $\vec{f}$, si LP, LP($\vec{U}$) y LP'($\vec{U}$) son factibles,
 5 entonces siempre existe una i^* , p^* , y t^* de tal forma que se cumple la condición (*) respectiva.

Como se mencionó anteriormente, se satisfacen de forma deseable solicitudes de alta prioridad (por ejemplo, solicitudes *ad hoc*). Se puede asumir que se dividen todas las solicitudes de alta
 10 prioridad en pequeños trozos, cada uno que tiene $a_i = b_i = d_i$, por ejemplo, cada solicitud de alta prioridad abarca exactamente un paso de tiempo individual. El proceso mencionado anteriormente tiene la estimación previa de la fracción de la capacidad c_e del borde e en el tiempo t que será necesaria para solicitudes de alta prioridad. Esta
 15 fracción se puede denotar por $X_{e,t}$. Esto incurre en la siguiente adaptación del proceso previamente mencionado: c_e se reemplaza con $(1-X_{e,t}) \cdot c_e$. Ya que X_s son estimaciones, puede ser el caso que una vez que se llega al paso de tiempo t y el proceso percibe las solicitudes de alta prioridad reales, se hace evidente que es
 20 necesaria más de una fracción $X_{e,t}$ de la capacidad de borde e . En este caso, el proceso se puede configurar para reducir algún flujo preprogramado en el borde e , que se puede lograr al formular un programa lineal pequeño que corrige el flujo preprogramado en el borde e en tanto que reduce al mínimo el efecto en la $\vec{U}$ garantizada,
 25 prometida provista por el componente optimizador 408.

En otra modalidad de ejemplo, para acelerar el tiempo de ejecución, puede ser deseable considerar el uso de un enfoque de ventana deslizante. Dado un tamaño de ventana W , si el proceso en línea es un paso de tiempo $\tilde{U}$ sólo observa W pasos de tiempo
 5 adelante en el futuro. Un problema potencial con este enfoque es que puede haber solicitudes i para las cuales $d_i - \tilde{U} > W$, por ejemplo, en el tiempo en el cual el proceso percibe la solicitud i que termina después de la ventana actual.

Una solución para este problema potencial es dividir esta
 10 solicitud en solicitudes más pequeñas. De forma específica, una vez que el proceso percibe la solicitud i como anteriormente en el tiempo $\tilde{U}$ considera que su fecha límite es la hora de finalización actual de la ventana deslizante: $\tilde{U} + W$. Además, su demanda se configura para que sea la proporción de su demanda original que se adapta en la
 15 ventana deslizante actual. Con más particularidad, la demanda se puede determinar por la siguiente fórmula cuando el proceso en línea del componente optimizador 408 está en el tiempo $\tilde{U}$

$$= D_i * \frac{W}{d_i - \tau} \quad \text{si } b_i \leq \tau \leq d_i \quad \text{y} \quad d_i > W$$

20

$$= D_i \quad \text{de otra forma .}$$

En cualquier paso de tiempo posterior el total del flujo de la solicitud i programada se puede sustraer de D_i .

Se puede determinar que el componente optimizador 408 debe
 25 operar en una escala de tiempo relativamente angosta. En

consecuencia, el proceso descrito anteriormente se puede modificar para facilitar la agilización de su ejecución. Por ejemplo, el problema de la ruta más corta se puede descomponer. El hallazgo de una tupla (i, t, p) que satisface (*) puede ser sustancialmente similar a la ejecución de un proceso de todos los pares de la ruta más corta en una gráfica expandida, que incluye una copia de la red original para cada caso de tiempo. Al hacerlo, sin embargo, puede ser demasiado costoso en vista de los casos de problemas grandes que se pueden considerar por el componente programador 118. Por lo tanto, como se aludió anteriormente, el gráfico de red 402 se puede limitar. Por ejemplo, para cada par de fuente-destino, se puede retener una lista de rutas ordenada por longitud para cada copia de la gráfica (la longitud de una ruta es la suma de las longitudes de sus bordes; la longitud de borde está dada por $\frac{y_e}{c_e}$). La ruta más corta para cada par de fuente-destino se puede obtener al tomar la ruta de longitud mínima entre todas las copias; una ventaja de esta descomposición es que se pueden buscar las rutas más cortas en gráficas mucho más pequeñas.

En vista de lo anterior, se puede observar que en cualquier iteración del proceso, se puede calcular la ruta más corta solamente para el caso de tiempo que corresponde a la última actualización (por ejemplo, donde este caso de tiempo se denota por t). Además, la ruta más corta (que corresponde al tiempo t) se necesita calcular solamente para usuarios cuya ruta más corta actual tiene bordes

comunes con la ruta cuyo flujo se ha aumentado. Esto da por resultado menos actualizaciones en cada iteración, haciendo al proceso más eficiente.

Además, el proceso expuesto anteriormente se puede
 5 paralelizar. El incremento de las variables de flujo $f_{i,p,t}$ y la actualización de las variables internas correspondientes $y_{e,t}$ y z_i se pueden hacer en paralelo, siempre y cuando no exista ninguna contención en la dimensión de tiempo, y además se asigna una solicitud a solamente un subproceso a la vez. Para obedecer estas
 10 limitaciones, se puede dividir el tiempo total en intervalos de tiempo, donde cada intervalo de tiempo se asigna a un subproceso diferente. Se pueden elegir intervalos de tiempo específicos de tal forma que se iguale aproximadamente la $\tilde{n}$ carga $\hat{o}$ (cuenta de solicitudes) en cada intervalo. La paralelización es sencilla cuando cada solicitud
 15 está $\tilde{n}$ activa $\hat{o}$ en exactamente un intervalo de tiempo. Las solicitudes, sin embargo, pueden abarcar múltiples intervalos de tiempo. Para abordar este desafío, la solicitud se puede asignar a un intervalo de tiempo específico con probabilidad igual a la masa relativa de la solicitud en ese intervalo de tiempo. Esta asignación garantiza que
 20 en efecto se fuercen las limitaciones descritas anteriormente. En una fase de dispersión, cada subproceso se ejecuta un número dado de iteraciones (que se pueden establecer -por ejemplo, 10^4). La siguiente fase de recolección entonces actualiza los valores de todas las variables. En tanto que algunas limitaciones de cobertura
 25 permanecen sin satisfacerse y permanecen posibles ediciones

factibles a $f_{i,p,t}$ con base en la condición *, el componente optimizador 408 repite las fases de dispersión y recolección anteriores. Se señala que una solicitud que abarca múltiples intervalos de tiempo se puede asignar a diferentes subprocesos en diferentes fases de dispersión. Hacia el final de la ejecución, donde el número de solicitudes no satisfechas es pequeño, el proceso puede cambiar a ejecución de un solo subproceso (por ejemplo, la sobrecarga de la paralelización se vuelve significativa).

Resumiendo al menos alguna parte de lo anterior, un algoritmo de ejemplo que se puede utilizar por el componente optimizador 408 se puede clasificar como un algoritmo de empaquetamiento y cobertura mezclado. El proceso identifica una solución al hacer pequeños cambios aun demostrablemente seguros a las variables, de tal forma que existe un límite en las violaciones de rebasamiento en la limitación de empaquetamiento. Cuando aparecen nuevas limitaciones (por ejemplo, nuevas transferencias en un siguiente paso de tiempo o un cambio en las condiciones de red), el proceso puede utilizar la etapa convergida ñantiguaò (por ejemplo, el problema no se necesita resolver desde cero, sino puede comenzar desde una etapa previamente convergida). El proceso permite que se realicen promesas a los solicitantes, donde la promesa garantiza que se satisfará una solicitud de transferencia a pesar de nuevas llegadas, solicitudes de alta prioridad (*ad hoc*), y condiciones de red variables. Eso se puede lograr al dejar de lado una cantidad particular de recursos de red futuros -en un siguiente paso de tiempo

(del tiempo actual), se pueden asignar completamente recursos, en tanto que para pasos de tiempo en el futuro, las cantidades crecientes de recursos pueden permanecer sin asignar. Además, este proceso se puede paralelizar para implementación práctica.

5 Con referencia a la figura 6, se ilustra una interfaz de usuario gráfica de ejemplo 600 que se puede exponer a un propietario de datos retenidos en al menos un dispositivo de cómputo de la red 102. La interfaz gráfica de usuario 600 incluye un campo de datos 602 que se configura para recibir identificación de datos que se van a transferir de un dispositivo de cómputo fuente a un dispositivo de
10 cómputo receptor. Por ejemplo, con base en una identificación de los datos que se menciona en el campo de datos 602, se puede determinar un volumen de los datos que se solicitan para que se transfieran del dispositivo de cómputo fuente al dispositivo de
15 cómputo receptor.

 La interfaz gráfica de usuario 600 incluye además un campo de ID de receptor 604 que identifica un dispositivo de cómputo receptor en la red 102. La interfaz gráfica de usuario 600 puede incluir de forma opcional un campo de inicio 606 que se configura para recibir
20 una identificación de una hora cuando puede comenzar la transferencia de los datos al dispositivo de cómputo receptor identificado en el campo de ID de receptor 604. Si el campo de inicio 606 se omite de la interfaz gráfica de usuario 600 o no se especifica una hora de inicio, entonces se puede hacer una suposición que
25 puede comenzar inmediatamente la transferencia de los datos. La

interfaz de gráfica de usuario 600 también incluye un campo de fecha límite 608 que puede recibir una indicación de cuándo se va a completar la transferencia de los datos del dispositivo de cómputo fuente al dispositivo de cómputo receptor identificado en el campo de ID de receptor 604. Con base en los datos expuestos en los campos 602-608, se puede presentar información de fijación de precios en un campo de fijación de precios 610. La información de fijación de precios puede cambiar, por ejemplo, conforme cambia una fecha límite expuesta en el campo de fecha límite 608, conforme cambia una cantidad de datos en el campo de datos 602, etc. En consecuencia, se puede determinar que la interfaz de gráfica de usuario 600 facilita la especificación de una fecha límite absoluta, donde se va a completar la transferencia de datos antes de la fecha límite. El componente programador 118 puede considerar la fecha límite cuando determina si aceptar o rechazar la solicitud, y puede calcular el programa a largo plazo 124 y el programa a corto plazo 126 con base en la fecha límite.

Las figuras 7-9 ilustran metodologías de ejemplo relacionadas con la programación de transferencia de datos en una red. En tanto que las metodologías se muestran y describen como una serie de actos que se llevan a cabo en una secuencia, se va a entender y apreciar que las metodologías no se limitan por el orden de la secuencia. Por ejemplo, algunos actos pueden presentarse en un orden diferente del que se describe en la presente. Además, un acto puede presentarse de forma concurrente con otro acto. Además, en

algunos casos, no todos los datos pueden ser necesarios para implementar una metodología descrita en la presente.

Además, los actos descritos en la presente pueden ser instrucciones ejecutables por computadora que se pueden
5 implementar por uno o más procesadores y/o almacenar en un medio o medios leíbles por computadora. Las instrucciones ejecutables por computadora pueden incluir una rutina, una subrutina, programas, un subproceso de ejecución, y/o similares. Aún más, los resultados de los actos de las metodologías se pueden almacenar en un medio
10 leíble por computadora, mostrar en un dispositivo de visualización, y/o similares.

Con referencia ahora a la figura 7, se ilustra una metodología de ejemplo 700 que facilita el cálculo de un programa a largo plazo. La metodología 700 comienza en 702, y en 704, se recibe una
15 solicitud para transferir datos de un primer dispositivo de cómputo en una red a un segundo dispositivo de cómputo en la red. Como se indicó anteriormente, la solicitud comprende primeros datos que identifican el segundo dispositivo de cómputo, segundos datos que identifican un volumen de datos que se van a transferir del primer
20 dispositivo de cómputo al segundo dispositivo de cómputo de acuerdo con la solicitud, y terceros datos que identifican una fecha límite, en donde la transferencia de los datos del primer dispositivo de cómputo al segundo dispositivo de cómputo se va a completar antes de la fecha límite.

25 En respuesta a la recepción de la solicitud y con base al menos

en parte en la solicitud, en 706 se genera un programa a largo plazo para transferencias de datos en la red. Como se indicó anteriormente, se utiliza un proceso de optimización para calcular el programa a largo plazo. El programa a largo plazo cubre una pluralidad de unidades de tiempo futuro y se genera para facilitar la terminación de la transferencia del volumen de los datos del primer dispositivo de cómputo al segundo dispositivo de cómputo antes de la fecha límite. Además, el programa a largo plazo identifica, para al menos una unidad de tiempo futuro en la pluralidad de unidades de tiempo futuro, una ruta en la red a través de la cual se van a transferir los datos del primer dispositivo de cómputo al segundo dispositivo de cómputo.

En 708, con base en el programa a largo plazo, se genera un programa a corto plazo para transferencia de datos en la red, en donde el programa a corto plazo se genera para facilitar la terminación de la transferencia de los datos del primer dispositivo de cómputo al segundo dispositivo de cómputo antes de la fecha límite. El programa a corto plazo comprende una tabla de encaminamiento para un dispositivo de infraestructura de red en la red. La tabla de encaminamiento identifica al menos un dispositivo al cual se van a transferir los datos recibidos por el dispositivo de infraestructura de red. En 710, la tabla de encaminamiento se transmite al dispositivo de infraestructura de red, y se completa la metodología 700 en 712.

Con referencia ahora a la figura 8, se ilustra una metodología de ejemplo 800 que facilita la generación del programa a largo plazo.

La metodología 800 comienza en 802, y en 804 se recibe un mapa de una red. Como se indicó anteriormente, el mapa de la red comprende una pluralidad de nodos que son representativos de dispositivos en la WAN y una pluralidad de bordes que son representativos de enlaces de red entre dispositivos en la red. En 806, se construye un gráfico de red con base en el mapa de red. El gráfico de red comprende una pluralidad de casos del mapa de la red: un caso del mapa de red para cada unidad de tiempo cubierta por el programa a largo plazo. En 808, se reciben limitaciones de ruta para transferencias de datos. Estas limitaciones de ruta pueden restringir las rutas a través de las cuales pueden viajar los datos que se van a transferir de un dispositivo de cómputo fuente a un dispositivo de cómputo receptor.

En 810, se recibe información acerca de solicitudes de transferencias de datos pendientes. Esta información puede incluir un volumen de datos respectivo que permanece para que se transfiera para cada solicitud, una fecha límite respectiva de la solicitud, entre otra información. En 812, se ejecuta un algoritmo de empaquetamiento y cobertura mezclado en paralelo sobre el gráfico de red con base en el gráfico de red, las limitaciones de ruta, y la información recibida en 810. Se va a entender que se contemplan otros tipos de algoritmos para el cálculo del programa a largo plazo. Por ejemplo, se puede resolver un programa lineal en conexión con el cálculo del programa a largo plazo. La metodología 800 finaliza en 814.

Volviendo ahora a la figura 9, se ilustra una metodología de ejemplo 900 que facilita la exposición de información de fijación de precios acerca de transferencias de datos a través de una red. La metodología comienza en 902, y en 904 se recibe un programa a largo plazo para transferencias de datos a través de una red. En 906, con base en el programa a largo plazo, se genera un programa de fijación de precios para transferencias de datos. Este programa de fijación de precios se puede configurar para suavizar la demanda de transferencias de datos en la red con el paso del tiempo, incrementando de esta forma la eficiencia de la utilización de recursos de *hardware* de la red. En 908, se recibe una solicitud para una transferencia de datos a través de la WAN, en donde la solicitud incluye una cantidad de datos que se van a transferir y una fecha límite antes de la cual se va a completar la transferencia. En 910, se expone la información de fijación de precios al solicitante con base en el programa de fijación de precios y la solicitud recibida. El solicitante de la transferencia de datos entonces puede aceptar el precio o modificar la solicitud con base en la información de fijación de precios. La metodología 900 se termina en 912.

Con referencia ahora a la figura 10, se ilustra una ilustración de alto nivel de un dispositivo de cómputo de ejemplo 1000 que se puede utilizar de acuerdo con los sistemas y metodologías descritas en la presente. Por ejemplo, el dispositivo de cómputo 1000 se puede utilizar en un sistema que soporta cálculo de un programa a largo plazo para transferencias de datos en una red. A manera de

otro ejemplo, el dispositivo de cómputo 1000 puede ser un dispositivo de cómputo fuente o un dispositivo de cómputo receptor en la red 102. El dispositivo de cómputo 1000 incluye al menos un procesador 1002 que ejecuta instrucciones que se almacenan en una memoria 1004. Las instrucciones pueden ser, por ejemplo, instrucciones para implementar la funcionalidad descrita como que se lleva a cabo por uno o más componentes analizados anteriormente o instrucciones para implementar uno o más de los métodos descritos anteriormente. El procesador 1002 puede acceder a la memoria 1004 por medio de un bus de sistema 1006. Además de almacenar las instrucciones ejecutables, la memoria 1004 también puede almacenar tablas de encaminamiento, programas a largo plazo, programas a corto plazo, etc.

El dispositivo de cómputo 1000 incluye además un almacén de datos 1008 que es accesible por el procesador 1002 por medio del bus del sistema 1006. El almacén de datos 1008 puede incluir instrucciones ejecutables, programas de transferencias de datos, etc. El dispositivo de cómputo 1000 también incluye una interfaz de entrada 1010 que permite que dispositivos externos se comuniquen con el dispositivo de cómputo 1000. Por ejemplo, la interfaz de entrada 1010 se puede utilizar para recibir instrucciones de un dispositivo de cómputo externo, de un usuario, etc. El dispositivo de cómputo 1000 también incluye una interfaz de salida 1012 que está interfaz con el dispositivo de cómputo 1000 con uno o más dispositivos externos. Por ejemplo, el dispositivo de cómputo 1000 puede mostrar texto,

imágenes, etc. por medio de la interfaz de salida 1012.

Se contempla que los dispositivos externos que se comunican con el dispositivo de cómputo 1000 a través de la interfaz de entrada 1010 y la interfaz de salida 1012 pueden estar incluidos en un entorno que proporciona sustancialmente cualquier tipo de interfaz de usuario con la cual un usuario puede interactuar. Ejemplos de tipos de interfaces de usuario incluyen interfaces gráficas de usuario, interfaces naturales de usuario, etc. Por ejemplo, una interfaz de usuario gráfica puede aceptar entrada de un usuario que emplea dispositivos de entrada tal como un teclado, ratón, control remoto, o similares y proporcionar salida a un dispositivo de salida tal como una pantalla. Además, una interfaz natural de usuario puede permitir que un usuario interactúe con el dispositivo de cómputo 1000 de una forma libre de limitaciones impuestas por el dispositivo de entrada tal como teclados, ratones, controles remotos, y similares. En su lugar, una interfaz natural de usuario puede basarse en el reconocimiento de voz, reconocimiento de tacto y lápiz, reconocimiento de gestos tanto en la pantalla y adyacente a la pantalla, gestos aéreos, seguimiento de cabeza y ojos, voz y dictado, visión, tacto, gestos, inteligencia artificial, etc.

Además, en tanto que se ilustra como un sistema individual, se va a entender que el dispositivo de cómputo 1000 puede ser un sistema distribuido. Por lo tanto, por ejemplo, varios dispositivos pueden estar en comunicación por medio de una conexión de red y pueden llevar a cabo de forma colectiva tareas descritas como que

se llevan a cabo por el dispositivo de cómputo 1000.

Diferentes funciones descritas en la presente se pueden implementar en *hardware*, *software*, o cualquier combinación de los mismos. Si se implementan en *software*, las funciones se pueden

5 almacenar en o transmitir a través de una o más instrucciones o código en un medio leíble por computadora. Medios leíbles por computadora incluyen medios de almacenamiento leíbles por computadora. Un medio de almacenamiento leíble por computadora puede ser cualquier medio de almacenamiento disponible al cual se

10 puede tener acceso por una computadora. A manera de ejemplo, y sin limitación, estos medios de almacenamiento leíbles por computadora pueden comprender RAM, ROM, EEPROM, CD-ROM u otro almacenamiento de disco óptico, almacenamiento de disco magnético u otros dispositivos de almacenamiento magnético, o

15 cualquier otro medio que se puede utilizar para transmitir o almacenar código de programa deseado en la forma de instrucciones o estructuras de datos y al cual se puede tener acceso por una computadora. Disco magnético y disco óptico, como se utiliza en la presente, incluyen disco compacto (CD), disco láser, disco óptico,

20 disco versátil digital (DVD), disco flexible, y disco Blu-ray (BD), donde los discos magnéticos por lo general reproducen datos de forma magnética y los discos ópticos por lo general reproducen datos de forma óptica con láser. Además, no está incluida una señal propagada dentro del alcance de medio de almacenamiento leíble por

25 computadora. Medios leíbles por computadora también incluyen

medios de comunicación que incluyen cualquier medio que facilita la transferencia de un programa de computadora de un lugar a otro. Una conexión, por ejemplo, puede ser un medio de comunicación. Por ejemplo, si el *software* se transmite de un sitio web, servidor, u

5 otra fuente remota utilizando un cable coaxial, cable de fibra óptica, par trenzado, línea de abonado digital (DSL, por sus siglas en inglés), o tecnologías inalámbricas tal como infrarrojo, radiofrecuencia, y microondas, entonces el cable coaxial, cable de fibra óptica, par trenzado, DSL, o tecnologías inalámbricas tal como

10 infrarrojo, radiofrecuencia y microondas están incluidas en la definición de medio de comunicación. Las combinaciones de los anteriores deben estar incluidas dentro del alcance de medios leíbles por computadora.

De forma alternativa, o además de, la funcionalidad descrita en

15 la presente se puede llevar a cabo, al menos en parte, por uno o más componentes lógicos de *hardware*. Por ejemplo, y sin limitación, tipos ilustrativos de componentes lógicos de *hardware* que se pueden utilizar incluyen arreglos de compuertas programables en el campo (FPGA), circuitos integrados de aplicación específica (ASIC),

20 productos estándar de aplicación específica (ASSP), sistemas en un chip (SOC), dispositivos lógicos programables complejos (CPLD), etc.

Lo que se ha descrito anteriormente incluye ejemplos de una o más modalidades. Desde luego, es imposible describir cada

25 alteración y modificación concebible de los dispositivos o

metodologías anteriores para propósitos de describir los aspectos anteriormente mencionados, pero una persona experta en la técnica puede reconocer que son posibles muchas modificaciones y permutaciones adicionales de diferentes aspectos. En consecuencia,

5 se propone que los aspectos descritos abarquen todas estas alteraciones, modificaciones, y variaciones que caen dentro del espíritu y alcance de las reivindicaciones anexas. Además en la medida que el término "incluye" se utiliza en ya sea la descripción de los detalles o las reivindicaciones, se propone que este término sea

10 inclusivo de una forma similar al término "que comprende" como "que comprende" se interpreta cuando se emplea como una palabra de transición en una reivindicación.

15

20

25