(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 741 131**

(51) Int. CI.:

G06F 9/50 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **28.10.2010 PCT/FR2010/052312**

(87) Fecha y número de publicación internacional: **19.05.2011 WO11058260**

(96) Fecha de presentación y número de la solicitud europea: **28.10.2010 E 10790584 (6)**

(97) Fecha y número de publicación de la concesión europea: **08.05.2019 EP 2499570**

(54) Título: **Procedimiento y dispositivo de optimización de ejecución de aplicaciones de software en una arquitectura multiprocesador que comprende varios controladores de entrada/salida y unidades de cálculo secundarias**

(30) Prioridad:

13.11.2009 FR 0905453

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
10.02.2020

(73) Titular/es:

BULL SAS (100.0%)
Rue Jean Jaurès
78340 Les Clayes-sous-Bois, FR

(72) Inventor/es:

DERR, SIMON;
GARRIGUES, PHILIPPE y
WELTERLEN, BENOÎT

(74) Agente/Representante:

ELZABURU, S.L.P

DESCRIPCIÓN

Procedimiento y dispositivo de optimización de ejecución de aplicaciones de software en una arquitectura multiprocesador que comprende varios controladores de entrada/salida y unidades de cálculo secundarias

La presente invención concierne a la colocación de cálculos en una arquitectura multiprocesador y, más en particular, a un procedimiento y un dispositivo de optimización de ejecución de aplicaciones de software en una arquitectura multiprocesador que comprende varios controladores de entrada/salida y unidades de cálculo secundarias, tales como procesadores gráficos.

Por causa de las restricciones físicas relacionadas con los microprocesadores y que limitan sus prestaciones, se han desarrollado arquitecturas basadas en la puesta en práctica de varios microprocesadores, que permiten efectuar cálculos paralelos. Estas arquitecturas multiprocesador facultan la ejecución de un gran número de aplicaciones y/o de aplicaciones que, segmentadas en etapas, recurren a una considerable cantidad de cálculo.

Los procesadores puestos en práctica en tales arquitecturas son, generalmente, capaces de procesar en paralelo procesos (denominados *threads* en la terminología anglosajona) complejos.

Por otro lado, se han desarrollado procesadores específicos para responder a necesidades particulares, especialmente para las operaciones de presentación visual y de manipulación de datos gráficos. Estos procesadores, denominados procesadores gráficos o GPU (sigla de *Graphics Processing Unit* en la terminología anglosajona) actúan masivamente en paralelo y especialmente permiten procesar simultáneamente un gran número de procesos simples. Son particularmente eficaces para procesar cálculos repetitivos. Sin embargo, mientras que estos procesadores se han desarrollado para responder a unas necesidades muy particulares, algunos permiten, a día de hoy, efectuar cálculos no específicos. A título de ilustración, la tecnología CUDA (CUDA es una marca), desarrollada por la firma nVidia, da respuesta a problemas de cálculos complejos.

De este modo, para mejorar las prestaciones de computadores de altas prestaciones o HPC (sigla de *High-Performance Computing* en la terminología anglosajona), se han desarrollado arquitecturas que combinan procesadores convencionales y procesadores específicos tales como procesadores gráficos.

Así, es conocido el documento de patente WO 2006/083043, que describe un dispositivo para descargar tareas de entre los nodos de un sistema distribuido. Asimismo es conocido el documento de patente WO 01/18641, que describe una arquitectura que permite redirigir una petición de una máquina cliente hacia una máquina servidor más próxima.

La puesta en práctica de estos diferentes tipos de procesadores precisa de una considerable cantidad de transferencias de datos entre ellos y la memoria. Consecuentemente, se utilizan controladores de entrada/salida. Se trata, por ejemplo, de componentes de un conjunto de chips (componentes electrónicos integrados que permiten gestionar los flujos de datos digitales entre procesadores, memoria y periféricos) que permiten servir de puente entre las interconexiones de procesadores estándar y los buses de entrada/salida tales como buses de tipo PCI-e (sigla de *Peripheral Component Interconnect Express* en la terminología anglosajona).

La figura 1 ilustra un ejemplo de tal arquitectura multiprocesador que comprende varios controladores de entrada/salida y procesadores gráficos.

Como se ilustra, el sistema 100 comprende, en este punto, cuatro procesadores principales referenciados con 105-1 a 105-4 (referenciados genéricamente con 105), por ejemplo, procesadores de tipo Nehalem (Nehalem es una marca) desarrollados por la firma Intel. Cada uno de estos procesadores comprende, en este punto, cuatro núcleos (procesadores quad-core) representados de manera esquemática. A título de ilustración, el procesador 105-1 comprende los núcleos 110-11 a 110-14.

De acuerdo con este ejemplo, cada procesador principal está unido a todos los demás procesadores principales a través de un enlace de comunicación rápida, por ejemplo, un enlace de tipo QPI (sigla de *Quick Path Interconnect* en la terminología anglosajona).

El sistema 100 comprende además dos controladores de entrada/salida 115-1 y 115-2, también denominados I/O Hub o IOH (sigla de *Input/Output Hub* en la terminología anglosajona). Cada IOH está unido, en este punto, a dos procesadores principales. Así, el IOH 115-1 está unido a los procesadores 105-1 y 105-4, en tanto que el IOH 115-2 está unido a los procesadores 105-2 y 105-3. La conexión entre IOH y procesadores es, por ejemplo, del tipo QPI.

Los IOH están además conectados cada uno de ellos a uno o varios procesadores gráficos que pueden estar especialmente agrupados sobre una placa específica. Los IOH 115-1 y 115-2 están unidos, en este punto, a los procesadores gráficos 120-1 y 120-2, respectivamente. El enlace de comunicación entre un IOH y un conjunto de procesadores gráficos es, por ejemplo, del tipo PCI-e.

Así, tal arquitectura permite a los procesadores 105-1 y 105-4 acceder directamente al procesador gráfico 120-1 y, a los procesadores 105-2 y 105-3, acceder directamente al procesador gráfico 120-2. Adicionalmente, los procesadores 105-1 y 105-4 pueden acceder al procesador gráfico 120-2, de manera indirecta, a través de la interconexión de los procesadores 105-2 y 105-3. Igualmente, los procesadores 105-2 y 105-3 pueden acceder al procesador gráfico 120-1 a través de la interconexión de los procesadores 105-1 y 105-4.

Cuando se pone en práctica tal sistema, generalmente se utiliza una biblioteca para administrar las llamadas de las aplicaciones ejecutadas por los procesadores principales a las funciones ejecutadas por los procesadores gráficos. Esta biblioteca tiene por objeto especialmente determinar el o los procesadores gráficos que deben ejecutar estas funciones.

Se observa en este punto que unos procesadores gráficos pueden ser identificados por los usuarios según algunas de sus características, tales como su rendimiento o su versión. De este modo, a través de la biblioteca utilizada para administrar las llamadas a las funciones ejecutadas por procesadores gráficos, unos usuarios pueden utilizar esta información con el fin de escoger estos últimos según las funciones que hayan de ejecutarse.

Aunque estas soluciones hayan demostrado su eficacia, existe, no obstante, una necesidad constante de mejorarlas con el fin de responder a las necesidades de cálculo, cada vez mayores, exigidas por numerosas aplicaciones, especialmente dentro del ámbito de la simulación, cosa que propone la invención descrita.

Así, la invención tiene por objeto un procedimiento para optimizar la ejecución de una aplicación de software que comprende al menos una llamada a al menos una función que debe ser ejecutada por una unidad de cálculo secundaria, siendo ejecutada dicha aplicación de software en un sistema que comprende al menos una pluralidad de procesadores principales, una pluralidad de unidades de cálculo secundarias y una pluralidad de controladores de entrada/salida, estando cada controlador de entrada/salida de dicha pluralidad de controladores de entrada/salida unido a al menos un procesador principal de dicha pluralidad de controladores principales y estando cada unidad de cálculo secundaria de dicha pluralidad de unidades de cálculo secundarias unida a un controlador de entrada/salida de dicha pluralidad de controladores de entrada/salida, comprendiendo este procedimiento las siguientes etapas:

- determinación de la topología de dicho sistema;
- interceptación de dicha al menos una llamada a dicha al menos una función que debe ser ejecutada por al menos una unidad de cálculo secundaria;
- identificación del procesador principal que ha generado dicha al menos una llamada;
- identificación de al menos una unidad de cálculo secundaria en dicha pluralidad de unidades de cálculo secundarias, identificándose dicha al menos una unidad de cálculo secundaria según dicho procesador principal identificado y según dicha topología de dicho sistema; y
- modificación de dicha al menos una llamada para forzar la ejecución de al menos una parte de dicha al menos una función en dicha al menos una unidad de cálculo secundaria identificada.

Así, el procedimiento según la invención permite elegir las unidades de cálculo secundarias en las que deben ejecutarse funciones llamadas según la topología del sistema y el lugar de los procesadores principales originarios de esas llamadas dentro de la topología, con el fin de optimizar la colocación de ejecución de las funciones.

De acuerdo con una forma particular de realización, la etapa de determinación de dicha topología de dicho sistema comprende una etapa de constitución de al menos una lista asociada a al menos uno de dichos procesadores principales, comprendiendo dicha lista al menos un identificador de al menos una de dichas unidades de cálculo secundarias y una medida de distancia entre dicho al menos uno de dichos procesadores principales y dicha al menos una de dichas unidades de cálculo secundarias correspondiente a dicho al menos un identificador. Tal lista permite identificar rápidamente una unidad de cálculo secundaria en función de un procesador principal particular y de la distancia que los separa.

De manera ventajosa, el procedimiento comprende además una etapa de prueba de disponibilidad de dicha al menos una unidad de cálculo secundaria identificada, con el fin de seleccionar una unidad de cálculo secundaria disponible para ejecutar la función llamada.

De acuerdo con una forma particular de realización, dicha al menos una unidad de cálculo secundaria identificada es la unidad de cálculo secundaria disponible más próxima a dicho procesador principal que ha generado dicha al menos una llamada. Así, se minimizan los tiempos de latencia inducidos por la distancia entre un procesador principal y una unidad de cálculo secundaria que ejecuta una función llamada por este último.

Dicha topología se determina, preferentemente, según información propia de un sistema operativo puesto en práctica en dicho sistema. Así, se puede determinar la topología del sistema sin precisar de información complementaria.

De acuerdo con una forma particular de realización, dicha etapa de determinación de dicha topología comprende las siguientes etapas:

- identificación de al menos una unidad de cálculo secundaria de dicha pluralidad de cálculo secundarias;
- identificación de al menos un bus al que está conectada dicha al menos una unidad de cálculo secundaria identificada durante dicha etapa de determinación de dicha topología; e
- identificación de al menos un procesador principal de dicha pluralidad de procesadores principales conectado a dicho al menos un bus identificado.

Así, se puede determinar la topología de un sistema a partir de información propia de un sistema operativo tal como LINUX.

Siempre de acuerdo con una forma particular de realización, dicha etapa de modificación de dicha al menos una llamada comprende una etapa de sobrecarga de una función ejecutada en un procesamiento de dicha al menos una llamada. Así, la invención se puede llevar a la práctica fácilmente sin precisar de modificaciones particulares en las aplicaciones de software ejecutadas por el sistema.

Siempre de acuerdo con una forma particular de realización, las etapas anteriormente descritas se llevan a la práctica en una biblioteca cargada dinámicamente con anterioridad a la ejecución de dicha aplicación de software. Entonces, la invención es particularmente simple en su puesta en práctica.

La invención tiene asimismo por objeto un programa de ordenador que comprende instrucciones adaptadas para la puesta en práctica de cada una de las etapas del procedimiento anteriormente descrito cuando dicho programa es ejecutado en un ordenador, así como un dispositivo que comprende medios adaptados para la puesta en práctica de cada una de las etapas del procedimiento anteriormente descrito.

Las ventajas que brindan este programa de ordenador y este dispositivo son similares a las apuntadas anteriormente.

Otras ventajas, finalidades y características de la presente invención se desprenden de la descripción detallada que sigue, llevada a cabo a título de ejemplo no limitativo, con relación a los dibujos que se acompañan, en los cuales:

- la figura 1 ilustra un ejemplo de una arquitectura multiprocesador simple que comprende varios controladores de entrada/salida y procesadores gráficos, en la que la invención puede ser llevada a la práctica;
- la figura 2 ilustra un ejemplo de una arquitectura multiprocesador compleja que comprende varios controladores de entrada/salida y procesadores gráficos, en la que la invención puede ser llevada a la práctica;
- la figura 3 ilustra esquemáticamente ciertas etapas de un ejemplo del procedimiento puesto en práctica para forzar la elección de una unidad de cálculo secundaria particular en una llamada a una función que debe ser ejecutada por una unidad de cálculo secundaria; y
l- a figura 4 ilustra más concretamente la etapa de identificación de una unidad de cálculo secundaria presentada en la figura anterior.

Cuando en un sistema de cálculo se utiliza un solo controlador de entrada/salida (o IOH, sigla de *Input/Output Hub* en la terminología anglosajona), los procesadores principales (o UCP, sigla de *Central Processing Unit* en la terminología anglosajona) unidos a este IOH acceden a la misma velocidad a las unidades de cálculo secundarias, por ejemplo a los procesadores gráficos (o GPU, sigla de *Graphics Processing Unit* en la terminología anglosajona), a las que están unidos. Sin embargo, cuando hay varios IOH presentes, es posible, en función de la topología del sistema de cálculo, que varios procesadores principales no accedan a la misma velocidad a una unidad de cálculo secundaria dada.

Por ejemplo, con referencia a la figura 1, si las UCP 105-1 y 105-4 acceden ambas a la misma velocidad a las GPU 120-1 a las que están unidas directamente a través del IOH 115-1 y las UCP 105-2 y 105-3 acceden ambas, asimismo, a la misma velocidad a las GPU 120-1 a las que están unidas indirectamente a través del IOH 115-1, las UCP 105-1 y 105-4 no acceden a la misma velocidad a las GPU 120-1 que las UCP 105-2 y 105-3.

Así, se ha observado que una aplicación o un proceso que se ejecuta en un procesador principal relacionado directamente con un IOH debe comunicarse, en la medida de lo posible, con unidades de cálculo secundarias relacionadas con este IOH para evitar rebajar las prestaciones del sistema de cálculo. Este problema es tanto más cierto en grandes sistemas tales como el ilustrado en la figura 2.

Como se ilustra en la misma, el sistema de cálculo comprende, en este punto, cuatro subsistemas que tienen una misma arquitectura y que comprenden cada uno de ellos varios procesadores principales y varias unidades de cálculo secundarias, en este punto, procesadores gráficos.

A título de ilustración, el subsistema 200-1 comprende cuatro procesadores principales referenciados con 205-1 a 205-4, por ejemplo, procesadores de tipo Nehalem. Cada procesador principal, en este punto, está unido a todos los demás procesadores principales de su subconjunto a través de un enlace de comunicación rápida, por ejemplo, un enlace de tipo QPI (sigla de *Quick Path Interconnect* en la terminología anglosajona).

El subsistema 200-1 comprende además un elemento de comunicación 210 al que están conectados todos los procesadores principales 205-1 a 205-4 y al que asimismo están conectados dos IOH 215-1 y 215-2. Tal elemento de comunicación es, por ejemplo, un componente de tipo BCS (sigla de *Bull Coherent Switch* en la terminología anglosajona).

Los IOH, adicionalmente, están conectados cada uno de ellos a uno o varios procesadores gráficos. Los IOH 215-1 y 215-2 están unidos, en este punto, a los procesadores gráficos 220-1 y 220-2, respectivamente. El enlace de comunicación entre un IOH y un procesador gráfico es, por ejemplo, del tipo PCI-e (sigla de *Peripheral Component Interconnect Express* en la terminología anglosajona).

Por otro lado, varios componentes de tipo BCS pueden estar conectados entre sí, por ejemplo según un modo de conexión punto a punto de tipo XCSI (sigla de *eXtended Common System Interface* en la terminología anglosajona). Así, es posible unir los subsistemas entre sí a través de la red 225 de tipo XCSI.

De este modo, de acuerdo con esta arquitectura, cada procesador es capaz de recurrir a las funciones de cada procesador gráfico.

Sin embargo, como se ha apuntado anteriormente, se ha observado que todas las UCP no acceden a la misma velocidad a todas las GPU. Por ejemplo, mientras que las UCP 205-1 a 205-4 acceden las cuatro a la misma velocidad a las GPU 220-1 y 220-2 a las que están unidas directamente a través de los IOH 215-1 y 215-2, respectivamente, y el elemento de comunicación 210, la UCP 205'-1 del subsistema 200-3 accede a una velocidad menor a estas GPU por causa de la latencia introducida por el elemento de comunicación 210' y la red 225.

Cuando un proceso o una aplicación ejecutado en un procesador principal de un sistema de cálculo, tal como los ilustrados en las figuras 1 y 2, llama a una función que debe ser ejecutada por una unidad de cálculo secundaria, el núcleo del sistema operativo de este sistema administra esta llamada, por ejemplo a través de una biblioteca cargada previamente. El cometido de la biblioteca es, en especial, determinar los parámetros que permiten la ejecución de la función llamada, en particular, identificar la o las unidades de cálculo secundarias que deben ejecutar la función.

La invención tiene especialmente por objeto interceptar estas llamadas para forzar la elección de la o de las unidades de cálculo secundarias que deben ejecutar la o las funciones llamadas. Dicho de otro modo, la llamada a una función procedente de un procesador principal y encaminada a reservar una unidad de cálculo secundaria para ejecutar esta función es interceptada con el fin de forzar la elección de esta unidad de cálculo secundaria, para que sea lo más próxima posible al procesador principal originario de la llamada, preferentemente unida al mismo controlador de entrada/salida que aquel al que está unido el procesador principal originario de la llamada.

La figura 3 ilustra esquemáticamente ciertas etapas de un ejemplo del procedimiento puesto en práctica para forzar la elección de una unidad de cálculo secundaria particular en una llamada a una función que debe ser ejecutada por una unidad de cálculo secundaria.

Como se ilustra, consiste una primera etapa (etapa 300) en determinar la topología del sistema de cálculo para determinar, en particular, los enlaces entre los procesadores principales, las unidades de cálculo secundarias y los controladores de entrada/salida.

Una parte de esta etapa puede consistir, en especial, en analizar mensajes de diagnóstico o el diario de ejecución del núcleo del sistema operativo puesto en práctica en el sistema de cálculo, generalmente llamados ficheros de log. Ésta puede consistir, asimismo, en explorar ciertos datos de la estructura jerárquica de los datos (sistema de ficheros) del sistema operativo.

De este modo, por ejemplo, existen, en la estructura jerárquica de los datos del sistema operativo Linux (Linux es una marca), especialmente en las ubicaciones conocidas con el nombre de */sys* y */proc*, seudoficheros que contienen información sobre el sistema. Ésta la proporciona el núcleo del sistema operativo y permite determinar la topología del sistema de cálculo.

A título de ilustración, la topología de un sistema de cálculo se puede determinar de la manera siguiente:

- identificación de las unidades de cálculo secundarias;
- análisis de los buses del sistema de cálculo para identificar los buses (y los controladores de entrada/salida)

a los que están conectadas las unidades de cálculo secundarias; e
- identificación de los procesadores principales conectados a los buses a los que están conectadas las unidades de cálculo secundarias.

La identificación de las unidades de cálculo secundarias nVidia puede realizarse, por ejemplo, a partir de la información proporcionada en la siguiente ubicación:

/proc/driver/nvidia/cards/
donde se guardan indicaciones relativas a los programas controladores de cada periférico de tipo nVidia y, consecuentemente, a estos propios periféricos.
Se observa, en este punto, que el sistema de ficheros */proc* es el directorio que contiene el seudosistema de ficheros del núcleo, que permite acceder a la información sobre el soporte físico, la configuración del núcleo y sobre los procesos que se están ejecutando.

De este modo, explorando los directorios de este tipo, es posible identificar todas las unidades de cálculo secundarias del sistema de cálculo.

De manera similar, es posible acceder a la configuración de los buses del sistema de cálculo para identificar los buses a los cuales están conectadas las unidades de cálculo secundarias identificadas con anterioridad. Este análisis puede realizarse, por ejemplo, a partir de la información proporcionada en la siguiente ubicación:

/sys/bus/pci/devices/0000:xxxxx
donde se guarda información relativa a los buses utilizados y, consecuentemente, a los controladores de entrada/salida utilizados.

Se observa, en este punto, que el sistema de ficheros /sys es el directorio que contiene, en particular, el seudosistema de ficheros de los administradores de periféricos que permite obtener información sobre el conjunto de los objetos del núcleo, en particular, sobre el conjunto de los periféricos de sistema de cálculo. Contiene información específica propia de características definidas de manera más general en el sistema de ficheros */proc*.

Finalmente, es posible determinar los procesadores principales unidos a los buses identificados anteriormente, por ejemplo a partir de la información proporcionada en la siguiente ubicación:

/proc/self/stat
donde hay presente información relativa a los procesadores utilizados por la aplicación llamante.

De manera similar, es posible determinar las conexiones entre los procesadores principales y, consecuentemente, establecer una estructura representativa de la distancia entre cada unidad de cálculo secundaria y cada procesador principal. Tal estructura puede estar, por ejemplo, guardada en una tabla. Se da un ejemplo de tal tabla en el Anexo (Tabla 1). Ésta corresponde a la topología del sistema de cálculo ilustrado en la figura 1. De este modo, como se ha indicado, el procesador principal 105-1 está conectado directamente a la unidad de cálculo secundaria 120-1 (distancia nula), en tanto que este procesador está conectado indirectamente a la unidad de cálculo secundaria 120-2, a través de un procesador principal (distancia igual a uno).

De manera ventajosa, esta tabla está organizada en forma de listas ordenadas de tal manera que, cuando se selecciona un procesador principal, sea posible identificar directamente las unidades de cálculo secundarias más próximas, estando las mismas clasificadas por distancias crecientes. Tal ejemplo de listas clasificadas, basado en la Tabla 1, se ilustra en el Anexo (Tabla 2). De este modo, de acuerdo con este ejemplo, cuando, en este punto, se selecciona el procesador principal 105-1, se evidencia inmediatamente, con la lectura de la primera línea, que la unidad de cálculo secundaria más próxima es la unidad de cálculo secundaria 120-1, siendo la siguiente la unidad de cálculo secundaria 120-2. Se pueden utilizar otros métodos para definir la topología del sistema de cálculo. En particular, la misma puede estar definida, de manera estática, en un fichero.

Una etapa siguiente (etapa 305) tiene por objeto detectar e interceptar las llamadas de los procesos o de las aplicaciones ejecutados por los procesadores principales a funciones que deben ser ejecutadas por unidades de cálculo secundarias tales como GPU.

Cuando se detecta e intercepta una llamada de este tipo, se identifica (etapa 310) el procesador principal originario de la llamada. Esta identificación se puede realizar, especialmente, consultando los datos guardados en el fichero */proc/self/stat*.

Esta etapa viene seguida de una etapa de determinación de una lista de al menos una unidad de cálculo secundaria disponible para ejecutar la o las funciones llamadas y situada a una distancia predeterminada del procesador principal anteriormente identificado, preferentemente lo más cerca posible (etapa 315).

Aunque, de manera general, el objeto sea el identificar la unidad de cálculo secundaria disponible más próxima al procesador principal originario de la llamada a una función que debe ser ejecutada por una unidad de cálculo secundaria, es posible, no obstante, que se necesiten varias unidades de cálculo secundarias. En este caso, el número de unidades de cálculo secundarias identificadas puede depender de la naturaleza de la o las funciones llamadas, es decir, del número de unidades de cálculo secundarias requerido para ejecutar la o las funciones.

Adicionalmente, se hace observar que la unidad de cálculo secundaria más próxima pueda no ser seleccionada en un momento dado, con el fin de que permanezca disponible para ejecutar una función llamada posteriormente.

La topología del sistema de cálculo tal y como se ha determinado anteriormente se utiliza para identificar, en función del identificador del procesador principal originario de la llamada, la o las unidades de cálculo secundarias que deben ser utilizadas para ejecutar la o las funciones llamadas.

Para estos fines, se identifica, en primer lugar, el procesador principal para inferir las unidades de cálculo secundarias que tiene relacionadas, con las distancias correspondientes. Puede tratarse de una lista ordenada de unidades de cálculo secundarias. Esta información se obtiene directamente a partir de la topología determinada, por ejemplo, con el concurso de una tabla similar a la Tabla 2 dada en el Anexo. De acuerdo con una forma preferida de realización, las unidades de cálculo secundarias son analizadas unas a continuación de otras, por ejemplo según el orden de la lista ordenada de las unidades de cálculo secundarias, para identificar la o las unidades de cálculo secundarias disponibles más próximas.

En la figura 4 se ilustra en detalle un ejemplo de puesta en práctica de esta etapa 315.

Después de haber inicializado a cero una variable i que representa un índice en una lista de unidades de cálculo secundarias (etapa 400), se determina (etapa 405) una lista ordenada de las unidades de cálculo secundarias accesibles por el procesador principal identificado. Tal lista, preferentemente, está predeterminada como se describe con referencia a la Tabla 2 presentada en el Anexo. Se efectúa entonces una prueba para determinar si la unidad de cálculo secundaria que tiene el índice i en la lista ordenada está disponible (etapa 410). Si no está disponible, se incrementa el índice i en uno (etapa 415) y se repite la prueba precedente. Si, por el contrario, la unidad de cálculo secundaria que tiene el índice i está disponible, ésta es seleccionada para ejecutar la función llamada por el procesador principal.

Si se necesitan varias unidades de cálculo secundarias, se repiten las etapas 410 y 415 hasta que se obtenga el número de unidades de cálculo secundarias.

Obviamente, cuando deben seleccionarse varias unidades de cálculo secundarias, éstas pueden serlo al objeto de ser las más próximas al procesador principal seleccionado, de estar todas ellas a una misma distancia, la más próxima posible, del procesador seleccionado, o de estar a una misma distancia predeterminada del procesador seleccionado.

A título de ilustración, de manera acorde con la topología definida anteriormente con referencia a la figura 1, si la UCP originaria de la llamada es la UCP 105-1, se infiere que la lista ordenada de las GPU es la siguiente: 120-1, 120-2. A partir de esta última, se efectúa una prueba para determinar si la primera GPU, es decir, la GPU 120-1 más próxima, está disponible. En caso negativo, se efectúa una prueba similar en la siguiente unidad de cálculo secundaria, es decir, en la unidad de cálculo secundaria 120-2. Si lo está, ésta es seleccionada.

Así, cuando se han determinado la o las unidades de cálculo secundarias, se modifica (etapa 320) la llamada antes de ser transmitida (etapa 325).

La modificación de una llamada consiste, en este punto, en cargar una biblioteca que sobrecarga la llamada de asignación de la unidad de cálculo secundaria, por ejemplo, recurriendo a la función *cudaSetDevice()* que sirve para seleccionar la unidad de cálculo secundaria que ejecutará la función llamada.

Más concretamente, la función *cudaSetDevice()*, en este punto, es interceptada y llamada con los parámetros que permiten asignar las unidades de cálculo secundarias identificadas, por ejemplo, las unidades de cálculo secundarias más próximas.

Las etapas anteriormente descritas (etapas 305 a 325) se repiten para procesar las siguientes llamadas para ejecutar otras funciones en otras unidades de cálculo secundarias (cuando una unidad de cálculo secundaria queda asignada a un proceso, no se vuelve a ejecutar la función descrita con referencia a la figura 3 para cada llamada a la unidad de cálculo secundaria). Este proceso se repite mientras sean susceptibles de generarse llamadas.

De acuerdo con una forma particular de realización, se crea y se carga dinámicamente una biblioteca adaptada para poner en práctica el algoritmo descrito con referencia a la figura 3, por ejemplo con el concurso de la variable de entorno LD_PRELOAD, antes de la ejecución de aplicaciones que recurran a funciones efectuadas en unidades de

cálculo secundarias. Recuérdese, en este punto, que la variable de entorno LD_PRELOAD permite forzar la carga de una biblioteca adicional en la ejecución de una aplicación de software. Tal biblioteca permite sobrecargar una llamada a una función ejecutada cuando una función debe ser ejecutada en una unidad de cálculo secundaria.

De este modo, la utilización de una biblioteca que tiene por objeto interceptar llamadas a funciones ejecutadas por unidades de cálculo secundarias y modificar estas llamadas para forzar la ubicación de ejecución de estas funciones según la topología del sistema permite acelerar la ejecución de estas aplicaciones de software sin modificarlas.

Obviamente, para satisfacer necesidades específicas, una persona perita en la materia de la invención podrá introducir modificaciones en la anterior descripción. En particular, si bien las unidades de cálculo secundarias pueden ser, en particular, procesadores gráficos, asimismo puede tratarse de circuitos particulares tales como FPGA (sigla de *Field-Programmable Gate Array* en la terminología anglosajona) o ASIC (sigla de *Application-Specific Integrated Circuit* en la terminología anglosajona).

Anexo

Tabla 1

	120-1	120-2
105-1	0	1
105-2	1	0
105-3	1	0
105-4	0	1

Tabla 2

105-1	120-1	120-2
105-2	120-2	120-1
105-3	120-2	120-1
105-4	120-1	120-2

REIVINDICACIONES

1. Procedimiento para optimizar la ejecución de una aplicación de software que comprende al menos una llamada a al menos una función que debe ser ejecutada por una unidad de cálculo secundaria, siendo ejecutada dicha aplicación de software en un sistema que comprende al menos una pluralidad de procesadores principales (105, 205), una pluralidad de unidades de cálculo secundarias (120, 220) y una pluralidad de controladores de entrada/salida (115, 215), estando cada controlador de entrada/salida de dicha pluralidad de controladores de entrada/salida unido a al menos un procesador principal de dicha pluralidad de controladores principales y estando cada unidad de cálculo secundaria de dicha pluralidad de unidades de cálculo secundarias unida a un controlador de entrada/salida de dicha pluralidad de controladores de entrada/salida, estando este procedimiento caracterizado por comprender las siguientes etapas:

- determinación (300) de la topología de dicho sistema;
- interceptación (305) de dicha al menos una llamada a dicha al menos una función que debe ser ejecutada por al menos una unidad de cálculo secundaria;
- identificación (310) del procesador principal que ha generado dicha al menos una llamada;
- identificación (315) de al menos una unidad de cálculo secundaria en dicha pluralidad de unidades de cálculo secundarias, identificándose dicha al menos una unidad de cálculo secundaria según dicho procesador principal identificado y según dicha topología de dicho sistema; y
- modificación (320) de dicha al menos una llamada para forzar la ejecución de al menos una parte de dicha al menos una función en dicha al menos una unidad de cálculo secundaria identificada.

2. Procedimiento según la reivindicación 1, según el cual dicha etapa de determinación de dicha topología de dicho sistema comprende una etapa de constitución de al menos una lista asociada a al menos uno de dichos procesadores principales, comprendiendo dicha lista al menos un identificador de al menos una de dichas unidades de cálculo secundarias y una medida de distancia entre dicho al menos uno de dichos procesadores principales y dicha al menos una de dichas unidades de cálculo secundarias correspondiente a dicho al menos un identificador.

3. Procedimiento según la reivindicación 1 o la reivindicación 2, que comprende además una etapa de prueba de disponibilidad (410) de dicha al menos una unidad de cálculo secundaria identificada.

4. Procedimiento según la reivindicación anterior, según el cual dicha al menos una unidad de cálculo secundaria identificada es la unidad de cálculo secundaria disponible más próxima a dicho procesador principal que ha generado dicha al menos una llamada.

5. Procedimiento según una cualquiera de las reivindicaciones anteriores, según el cual dicha topología se determina según información propia de un sistema operativo puesto en práctica en dicho sistema.

6. Procedimiento según la reivindicación anterior, según el cual dicha etapa de determinación de dicha topología comprende las siguientes etapas:

- identificación de al menos una unidad de cálculo secundaria de dicha pluralidad de cálculo secundarias;
- identificación de al menos un bus al que está conectada dicha al menos una unidad de cálculo secundaria identificada durante dicha etapa de determinación de dicha topología; e
- identificación de al menos un procesador principal de dicha pluralidad de procesadores principales conectado a dicho al menos un bus identificado.

7. Procedimiento según una cualquiera de las reivindicaciones anteriores, según el cual dicha etapa de modificación de dicha al menos una llamada comprende una etapa de sobrecarga de una función ejecutada en un procesamiento de dicha al menos una llamada.

8. Procedimiento según una cualquiera de las reivindicaciones anteriores, según el cual dichas etapas se llevan a la práctica en una biblioteca cargada dinámicamente con anterioridad a la ejecución de dicha aplicación de software.

9. Programa de ordenador que comprende instrucciones adaptadas para la puesta en práctica de cada una de las etapas del procedimiento según una cualquiera de las reivindicaciones anteriores cuando dicho programa es ejecutado en un ordenador.

10. Dispositivo que comprende medios adaptados para la puesta en práctica de cada una de las etapas del procedimiento según una cualquiera de las reivindicaciones 1 a 8.

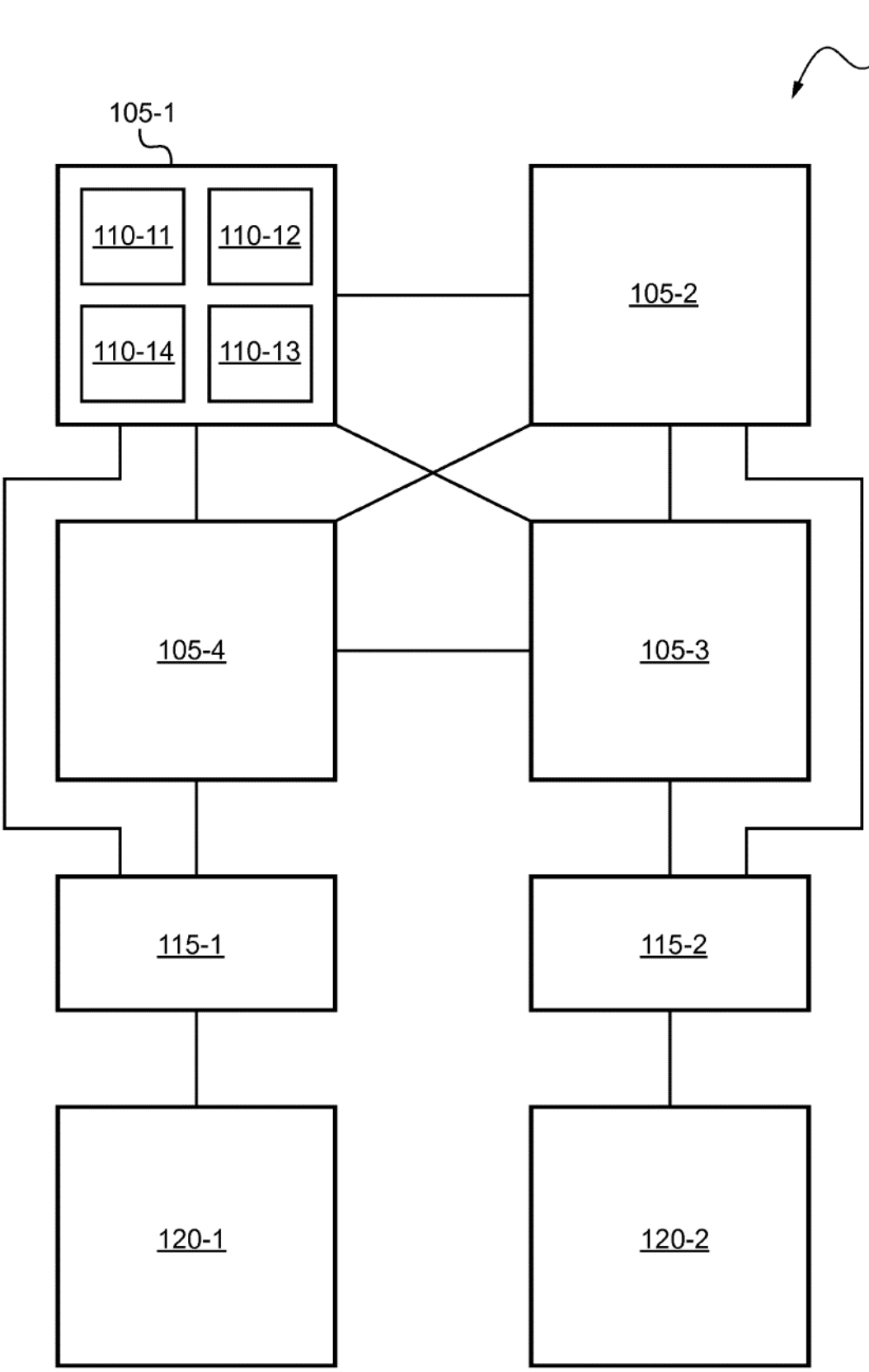
100
105-1
110-11
110-12
110-14
110-13
105-2
105-4
105-3
115-1
115-2
120-1
120-2

Fig. 1

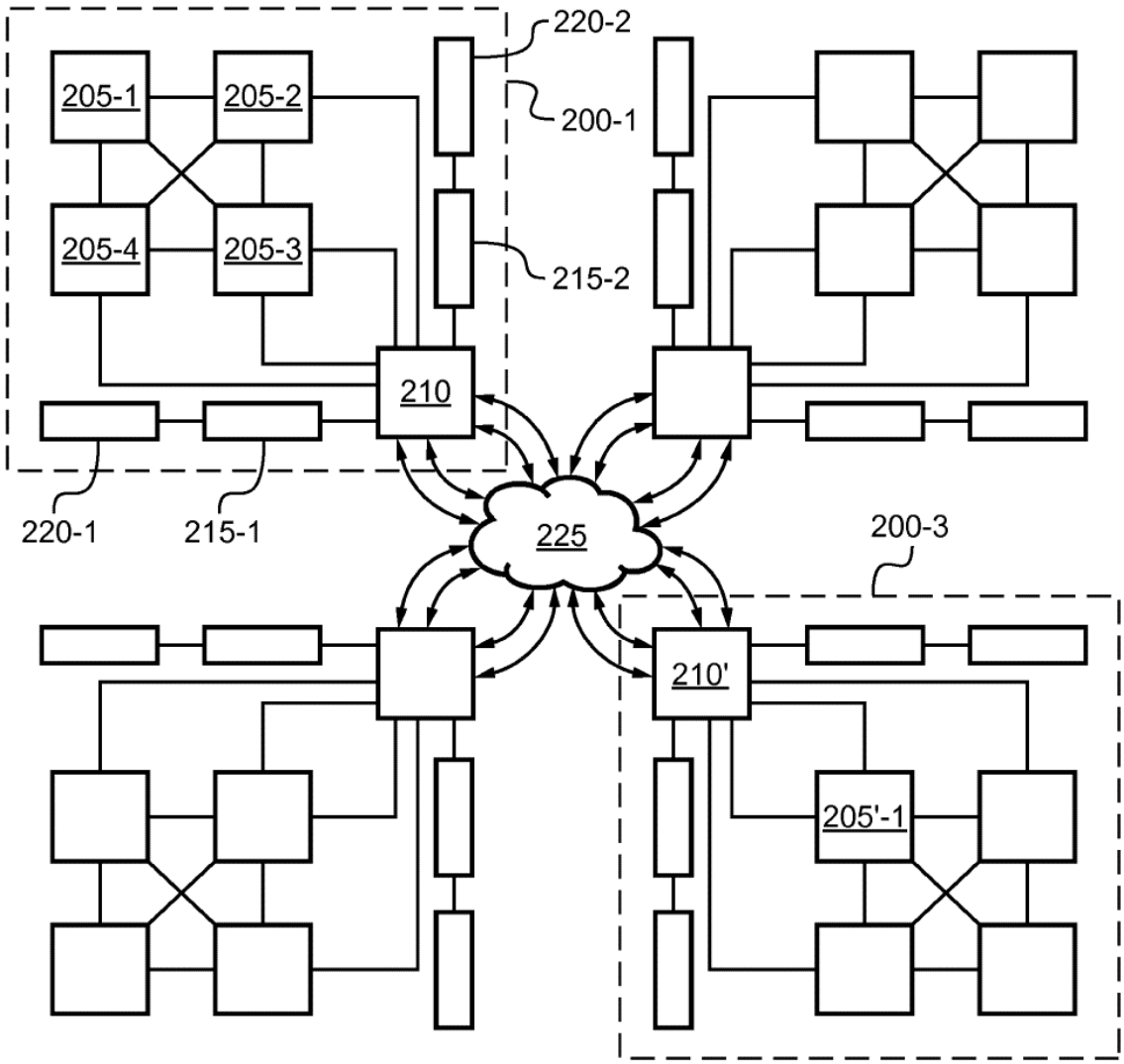
220-2
200-1
205-1
205-2
205-4
205-3
215-2
210
220-1
215-1
225
200-3
210'
205'-1

Fig. 2

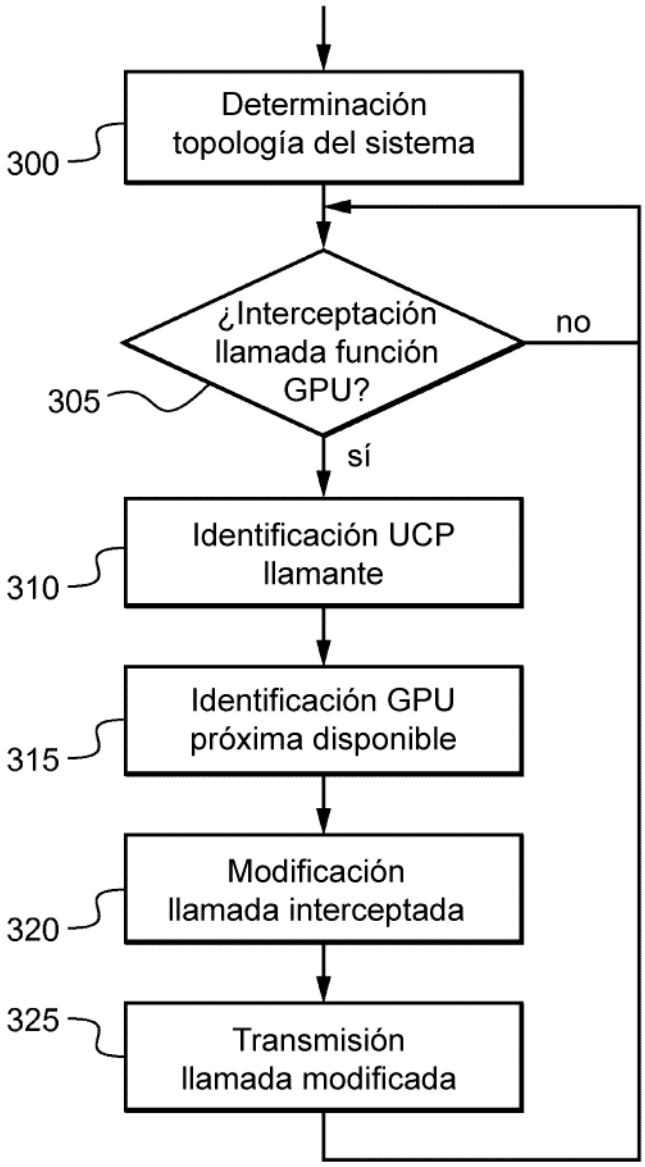
Determinación topología del sistema
300
¿Interceptación llamada función GPU?
305
no
sí
Identificación UCP llamante
310
Identificación GPU próxima disponible
315
Modificación llamada interceptada
320
Transmisión llamada modificada
325

Fig. 3

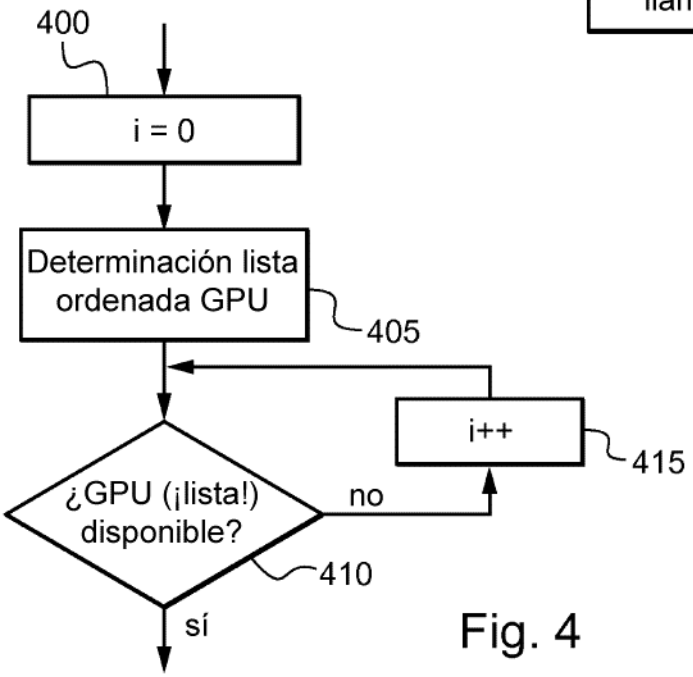
400
i = 0
Determinación lista ordenada GPU
405
i++
415
¿GPU (¡lista!) disponible?
410
no
sí

Fig. 4