(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 699 698**

(51) Int. CI.:

G06F 9/46 (2006.01)
G06F 9/48 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **27.03.2009 PCT/US2009/038658**

(87) Fecha y número de publicación internacional: **19.11.2009 WO09139966**

(96) Fecha de presentación y número de la solicitud europea: **27.03.2009 E 09747052 (0)**

(97) Fecha y número de publicación de la concesión europea: **05.09.2018 EP 2288990**

(54) Título: **Recopilaciones de programación en un programador**

(30) Prioridad:

16.05.2008 US 121794

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
12.02.2019

(73) Titular/es:

MICROSOFT TECHNOLOGY LICENSING, LLC (100.0%)
One Microsoft Way
Redmond, WA 98052, US

(72) Inventor/es:

RINGSETH, PAUL, F.;
FERNANDES, GENEVIEVE;
GUSTAFSSON, NIKLAS y
MOLLOY, RICK

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

DESCRIPCIÓN

Recopilaciones de programación en un programador

Antecedentes

Los procedimientos ejecutados en un sistema informático pueden incluir programadores de tareas que programan tareas de los procesos para su ejecución en el sistema informático. Estos programadores pueden operar con diversos algoritmos que determinan cómo se ejecutarán las tareas de un proceso. En un sistema informático con múltiples recursos de procesamiento, los recursos de procesamiento pueden competir entre sí en la búsqueda de tareas para ejecutar en un programador. La competencia tiende a reducir la eficiencia del sistema informático en la ejecución de un proceso con un programador, y la cantidad de competencia en general aumenta a medida que aumenta la cantidad de recursos de procesamiento en el sistema informático. Como resultado, la competencia de los recursos de procesamiento puede limitar la escalabilidad del programador a medida que aumenta la cantidad de recursos de procesamiento en un sistema informático.

El documento US 2008/0066072 A1 desvela un sistema y procedimiento para la asignación de procesos a una pluralidad de recursos. Uno de los conjuntos de asignación de recursos-tareas establecidos se selecciona de acuerdo con una condición de selección para el parámetro de control y los procesos se asignan a los recursos de acuerdo con el conjunto de asignaciones de recursos-tareas seleccionado. En consecuencia, se crea una asignación que define una programación precisa de los procesos y su asignación a los recursos.

Por lo tanto, es el objeto de la presente invención proporcionar un procedimiento mejorado para ejecutar tareas por una pluralidad de recursos de procesamiento y un medio de almacenamiento legible por ordenador correspondiente.

El presente objeto se resuelve por la materia objeto de las reivindicaciones independientes.

Las realizaciones preferidas están definidas por las reivindicaciones dependientes.

Breve descripción de los dibujos

Los dibujos adjuntos se incluyen para proporcionar una comprensión adicional de las realizaciones y se incorporan en y constituyen una parte de la presente memoria descriptiva. Los dibujos ilustran las realizaciones y junto con la descripción sirven para explicar los principios de las realizaciones. Otras realizaciones y muchas de las ventajas pretendidas de las realizaciones se apreciarán fácilmente a medida que se entiendan mejor haciendo referencia a la siguiente descripción detallada. Los elementos de los dibujos no están necesariamente a escala relativa entre sí. Los números de referencia similares designan partes similares correspondientes.

La figura 1 es un diagrama de bloques que ilustra una realización de un programador con recopilaciones de programación en un entorno de ejecución que incorpora la presente invención.
La figura 2 es un diagrama de flujo que ilustra una realización de un procedimiento para crear y llenar las recopilaciones de programación en un programador.
Las figuras 3A-3B son un diagrama y una tabla que ilustran una realización de una asignación de recopilaciones de programación.
La figura 4 es un diagrama de flujo que ilustra una realización de un procedimiento para seleccionar tareas de acuerdo con la presente invención.
Las figuras 5A-5B son diagramas de bloques que ilustran las realizaciones de las recopilaciones de programación.
Las figuras 6A-6B son diagramas de bloques que ilustran las realizaciones de un sistema informático configurado para implementar un entorno de ejecución que incluye un programador con recopilaciones de programación.

Descripción detallada

La figura 1 es un diagrama de bloques que ilustra una realización de un programador 22 de contexto de ejecución en un proceso 12 de un entorno 10 de tiempo de ejecución. El programador 22 incluye un conjunto de recopilaciones 40 (1)-40 (L) de programación, donde L es un entero mayor o igual que dos e indica la L-ésima recopilación 40 de programación. Cada recopilación 40 (1)-40 (L) de programación corresponde a un nodo 30 (1)-30 (L) de programación respectivo.

El entorno 10 de tiempo de ejecución representa un modo de tiempo de ejecución de la operación en un sistema informático, tales como las realizaciones 100A y 100B de un sistema 100 informático mostrado en las figuras 6A y 6B y descrito con más detalle a continuación, donde el sistema informático está ejecutando instrucciones. El sistema informático genera el entorno 10 de tiempo de ejecución a partir de una plataforma de tiempo de ejecución tal como una plataforma 122 de tiempo de ejecución mostrada en la figura 6A y descrita con más detalle a continuación.

El entorno 10 de tiempo de ejecución incluye al menos un proceso 12 invocado, una capa 14 de gestión de recursos y un conjunto de subprocesos 16 (1)-16 (M) de hardware, donde M es un número entero que es mayor o igual que dos e indica el M-ésimo subproceso 16 de hardware. El entorno 10 de tiempo de ejecución permite que se ejecuten

unas tareas a partir del proceso 12, junto con las tareas de cualquier otro procedimiento que exista conjuntamente con el proceso 12 (no mostrado), usando la capa 14 de gestión de recursos y los subprocesos 16 (1)-16 (M) de hardware. El entorno 10 de tiempo de ejecución opera junto con la capa 14 de gestión de recursos para permitir que el proceso 12 obtenga el procesador y otros recursos del sistema informático (por ejemplo, los subprocesos 16 (1)-16 (M) de hardware).

El entorno 10 de tiempo de ejecución incluye una función de programador que genera un programador 22. En una realización, la función de programador se implementa como una interfaz de programación de aplicaciones de programador (API). En otras realizaciones, la función de programador puede implementarse usando otras construcciones de programación adecuadas. Cuando se invoca, la función de programador crea el programador 22 en el proceso 12, donde el programador 22 opera para programar las tareas del proceso 12 para que sean ejecutadas por uno o más subprocesos 16 (1)-16 (M) de hardware. El entorno 10 de tiempo de ejecución puede explotar la concurrencia detallada que los desarrolladores de aplicaciones o bibliotecas expresan en sus programas (por ejemplo, el proceso 12) usando herramientas complementarias que son conscientes de las facilidades que proporciona la función de programador.

El proceso 12 incluye una asignación del procesamiento y otros recursos que albergan uno o más contextos de ejecución (es decir, subprocesos). El proceso 12 obtiene acceso al procesamiento y a otros recursos en el sistema informático (por ejemplo, los subprocesos 16 (1)-16 (M) de hardware) desde la capa 14 de gestión de recursos. El proceso 12 hace que las tareas se ejecuten usando el procesamiento y otros recursos.

El proceso 12 genera trabajo en tareas de longitud variable, donde cada tarea se asocia con un contexto de ejecución en el programador 22. Cada tarea incluye una secuencia de instrucciones que realizan una unidad de trabajo cuando se ejecuta por el sistema informático. Cada contexto de ejecución forma un subproceso que ejecuta las tareas asociadas en los recursos de procesamiento asignados. Cada contexto de ejecución incluye el estado de programa y la información de estado de máquina. Los contextos de ejecución pueden terminar cuando ya no quedan más tareas por ejecutar. Para cada tarea, el entorno 10 de tiempo de ejecución y/o el proceso 12 o bien asignan la tarea al programador 22 para que se programe para su ejecución o de otro modo hacen que la tarea se ejecute sin usar el programador 22.

El proceso 12 puede estar configurado para operar en un sistema informático basado en cualquier modelo de ejecución adecuado, tal como un modelo de pila o un modelo intérprete, y puede representar cualquier tipo adecuado de código, tal como una aplicación, una función de biblioteca, o un servicio de sistema operativo. El proceso 12 tiene un estado de programa y un estado de máquina asociados con un conjunto de recursos asignados que incluyen un espacio de direcciones de memoria definido. El proceso 12 se ejecuta de manera autónoma o sustancialmente autónoma desde cualquier proceso existente conjuntamente en el entorno 10 de tiempo de ejecución. En consecuencia, el proceso 12 no altera adversamente el estado de programa de los procesos existentes conjuntamente o el estado de máquina de los recursos asignados a los procesos existentes conjuntamente. De manera similar, los procesos existentes conjuntamente no alteran adversamente el estado de programa de proceso 12 o el estado de máquina de cualquier recurso asignado al proceso 12.

La capa 14 de gestión de recursos asigna los recursos de procesamiento al proceso 12 asignando uno o más subprocesos 16 de hardware al proceso 12. La capa 14 de gestión de recursos existe por separado de un sistema operativo del sistema informático (no mostrado en la figura 1) en la realización de figura 1. En otras realizaciones, la capa 14 de gestión de recursos o algunas o todas sus funciones pueden incluirse en el sistema operativo.

Los subprocesos 16 de hardware residen en núcleos de ejecución de un conjunto o uno o más paquetes de procesador (por ejemplo, los paquetes 102 de procesador mostrados en la figura 6 y descritos con más detalle más adelante) del sistema informático. Cada subproceso 16 de hardware está configurado para ejecutar instrucciones de manera independiente o sustancialmente independiente de los otros núcleos de ejecución e incluye un estado de máquina. Los subprocesos 16 de hardware pueden incluirse en un único paquete de procesador o pueden distribuirse en múltiples paquetes de procesador. Cada núcleo de ejecución en un paquete de procesador puede incluir uno o más subprocesos 16 de hardware.

El proceso 12 hace implícita o explícitamente que el programador 22 se cree a través de la función de programador proporcionada por el entorno 10 de tiempo de ejecución. El programador 22 puede crearse implícitamente cuando el proceso 12 usa las API disponibles en el sistema informático o en las funciones de lenguaje de programación. En respuesta a las API o a las funciones de lenguaje de programación, el entorno 10 de tiempo de ejecución crea el programador 22 con una política predeterminada. Para crear explícitamente un programador 22, el proceso 12 puede invocar la función de programador proporcionada por el entorno 10 de tiempo de ejecución y especificar una política para el programador 22.

El programador 22 interactúa con la capa 14 de gestión de recursos para negociar los recursos del sistema informático de una manera que sea transparente al proceso 12. La capa 14 de gestión de recursos asigna los subprocesos 16 de hardware al programador 22 basándose en la oferta y la demanda y en cualquier política del programador 22.

En la realización mostrada en la figura 1, el programador 22 gestiona los recursos de procesamiento creando procesadores 32 virtuales que forman una abstracción de los subprocesos 16 de hardware subyacentes. El programador 22 multiplexa los procesadores 32 virtuales sobre los subprocesos 16 de hardware asignando cada procesador 32 virtual a un subproceso 16 de hardware. El programador 22 puede asignar más de un procesador 32 virtual sobre un subproceso 16 de hardware específico, pero asigna solo un subproceso 16 de hardware a cada procesador 32 virtual. En otras realizaciones, el programador 22 gestiona los recursos de procesamiento de otras maneras adecuadas para provocar que las instrucciones del proceso 12 se ejecuten por los subprocesos 16 de hardware.

El entorno 10 de tiempo de ejecución crea un programador 22 con conocimiento de la topología subyacente del sistema informático. El entorno 10 de tiempo de ejecución proporciona la capa 14 de gestión de recursos y/o el programador 22 con información de nodo del sistema informático. La información de nodo identifica los nodos de hardware del sistema informático directamente o incluye información suficiente sobre la topología del sistema informático para permitir que la capa 14 de gestión de recursos y/o el programador 22 puedan particionar los recursos de hardware en los nodos 30 de programación basándose en una o más métricas de ejecución. Las métricas de ejecución pueden incluir una velocidad, tipo y/o configuración de los recursos de procesamiento (por ejemplo, subprocesos 16 de hardware), recursos de memoria y/u otros recursos del sistema informático.

Por ejemplo, en las realizaciones donde la topología del sistema informático incluye una arquitectura de acceso a memoria no uniforme (NUMA) coherente de caché, la información de nodo puede identificar un conjunto de dos o más nodos NUMA donde cada nodo NUMA incluye un conjunto de subprocesos 16 de hardware y una memoria local. La información de nodo también puede incluir información que describa los accesos de memoria entre los nodos NUMA (por ejemplo, las distancias NUMA o las topologías o tiempos de acceso a la memoria).

En otro ejemplo, la información de nodo puede describir la velocidad, el tipo y/o la configuración de los recursos de procesamiento (por ejemplo, los subprocesos 16 de hardware) para permitir que los recursos de procesamiento se agrupen basándose en similitudes o diferencias entre las características de los recursos de procesamiento. Estas características pueden incluir el tipo del conjunto de instrucciones de uno o más de los recursos de procesamiento para permitir que se formen diferentes nodos con conjuntos de recursos de procesamiento que tienen diferentes tipos de conjuntos de instrucciones.

El entorno 10 de tiempo de ejecución hace que el programador 22 incluya un conjunto de dos o más nodos 30 (1)-30 (L) de programación basándose en la información de nodo. Cada nodo 30 de programación incluye unos recursos de procesamiento asignados en forma de procesadores 32 virtuales y subprocesos 16 de hardware. El nodo 30 (1) de programación incluye los procesadores 30 (1)-30 (N1) virtuales que se asignan a los subprocesos 16 (1)-16 (m_1) de hardware donde N_1 es un número entero que es mayor o igual que uno e indica el (N_1)-ésimo procesador 30 virtual y m_1 es menor o igual que M e indica el (m_1)-ésimo subproceso 16 de hardware. El nodo 30 (L) de programación incluye los procesadores 30 (1)-30 (N_L) virtuales que se asignan a los subprocesos 16 (m_m)-16 (M) de hardware donde N_L es un número entero que es mayor o igual que uno e indica el (N_L)-ésimo procesador 30 virtual y m_m es menor o igual que M, mayor que m_1, e indica el (m_m)-ésimo subproceso 16 de hardware.

El programador 22 crea una recopilación 40 de programación para cada nodo 30 de programación. En consecuencia, se programan las recopilaciones 40 (1)-40 (L) para corresponder a los nodos 30 (1)-30 (L) de programación respectivos como se indica por las flechas 37 (1)-37 (L). El programador 22 asigna las recopilaciones 40 de programación en un orden de búsqueda completo o parcial basándose en una o más métricas de ejecución y usa el orden de búsqueda para buscar las tareas a ejecutar cuando los recursos de procesamiento estén disponibles, como se describirá con más detalle a continuación.

El conjunto de contextos de ejecución en el programador 22 incluye un conjunto de contextos 34 de ejecución con tareas 36 asociadas respectivas que están ejecutándose por los procesadores 32 virtuales respectivos en cada nodo 30 de programación y, en cada recopilación 40 de programación, un conjunto de cero o más contextos 38 de ejecución ejecutables y un conjunto de cero o más contextos 40 de ejecución bloqueados (es decir, dependientes de la espera). Cada contexto 34, 38 y 40 de ejecución incluye una información de estado que indica si se está ejecutando un contexto 34, 38 y 40 de ejecución, ejecutable (por ejemplo, en respuesta a desbloquearse o sumarse al programador 22), o bloqueado. Los contextos 34 de ejecución que están ejecutándose se han adjuntado a un procesador 32 virtual y se están ejecutando actualmente. Los contextos 38 de ejecución que pueden ejecutarse incluyen una tarea 39 asociada y están listos para ejecutarse por un procesador 32 virtual disponible. Los contextos 40 de ejecución que están bloqueados incluyen una tarea 41 asociada y están esperando datos, un mensaje o un evento que se está generando o se generará por otro contexto 34, 38 o 40 de ejecución.

Cada contexto 34 de ejecución que se ejecuta en un procesador 32 virtual puede generar, en el curso de su ejecución, tareas 42 adicionales, que se organizan de cualquier forma adecuada (por ejemplo, sumándose a las colas de trabajo (no mostrado en la figura 1)). El trabajo puede crearse usando o bien las interfaces de programación de aplicaciones (API) proporcionadas por el entorno 10 de tiempo de ejecución o las funciones de lenguaje de programación y las herramientas correspondientes en una realización. Cuando los recursos de procesamiento están disponibles para el programador 22, las tareas se asignan a los contextos 34 o 38 de ejecución que los ejecutan hasta completarlos en los procesadores 32 virtuales antes de recoger nuevas tareas. Un contexto 34 de ejecución

que se ejecuta en un procesador 32 virtual también puede desbloquear otros contextos 40 de ejecución generando datos, un mensaje o un evento que se usará por otro contexto 40 de ejecución.

Cada tarea en el programador 22 puede realizarse (por ejemplo, las tareas 36 y 39 realizadas), lo que indica que un contexto 34 o 38 de ejecución se ha adjuntado o se adjuntará a la tarea y la tarea está lista para ejecutarse. Las tareas realizadas normalmente incluyen contextos de ejecución desbloqueados y agentes programados. Una tarea que no se realiza se denomina no realizada. Las tareas no realizadas (por ejemplo, las tareas 42) pueden crearse como tareas secundarias generadas por la ejecución de tareas principales y pueden generarse mediante construcciones paralelas (por ejemplo, paralelas, paralelas para, comenzar y finalizar). Cada recopilación 40 de programación en el programador 22 puede organizarse en una o más recopilaciones sincronizadas (por ejemplo, una pila y/o una cola) para tareas lógicamente independientes con contextos de ejecución (es decir, tareas realizadas) junto con una lista de colas de robo de trabajo para tareas dependientes (es decir, tareas no realizadas) como se ilustra en la realización de la figura 5A descrita a continuación.

Una vez completado, el bloqueo, u otra interrupción (por ejemplo, rendimiento explícito o preferencia forzada) de un contexto 34 de ejecución que se ejecuta en un procesador 32 virtual, el procesador 32 virtual se convierte en disponible para ejecutar otra tarea 39 realizada o tarea 42 no realizada. El programador 22 busca un contexto 38 de ejecución ejecutable o una tarea 42 no realizada para adjuntarla al procesador 32 virtual disponible para su ejecución. El programador 22 continúa adjuntando los contextos 38 de ejecución a los procesadores 32 virtuales disponibles para su ejecución hasta que se hayan ejecutado todos los contextos 38 de ejecución del programador 22.

Cuando un procesador 32 virtual en un nodo 30 de programación se convierte en disponible, el procesador 32 virtual intenta localizar una tarea a ejecutar en una recopilación 40 de programación correspondiente al nodo 30 de programación. Si el procesador 32 virtual no localiza una tarea a ejecutar en la recopilación 40 de programación, el procesador 32 virtual intenta localizar una tarea a ejecutar en otras recopilaciones 40 de programación en un orden especificado por el orden de búsqueda. En una realización, el programador 22 puede incluir un parámetro de retardo configurable que hace que los procesadores 32 virtuales disponibles retrasen la búsqueda de otras recopilaciones 40 de programación para intentar minimizar la competencia con otros procesadores 32 virtuales disponibles. El parámetro de retardo también puede usarse para priorizar la búsqueda para trabajar con la recopilación 40 de programación correspondiente al nodo 30 de programación del procesador 32 virtual disponible.

La figura 2 es un diagrama de flujo que ilustra una realización de un procedimiento para crear y llenar las recopilaciones 40 de programación en el programador 22. El procedimiento de la figura 2 se describirá haciendo referencia a la realización de programador 22 en la figura 1.

En la figura 2, el entorno 10 de tiempo de ejecución, y/o la capa 14 de gestión de recursos identifican los nodos 30 de programación basándose en una o más métricas de ejecución como se indica en el bloque 52. Las métricas de ejecución pueden ser cualesquiera medidas adecuadas para ejecutar instrucciones en el sistema informático y pueden incluir la velocidad de procesamiento, el rendimiento del procesamiento y las características de latencia de memoria del procesamiento y otros recursos en el sistema informático. El uso de métricas de ejecución determinadas para diversos conjuntos de componentes del sistema informático, el entorno 10 de tiempo de ejecución y/o la capa 14 de gestión de recursos particionan el procesamiento y otros recursos del sistema informático y usan las particiones para identificar los nodos 30 de programación para el programador 22. Cada uno de los nodos 30 de programación incluye unos grupos de conjuntos similares o diferentes de procesamiento y otros recursos del sistema informático.

En un ejemplo, el sistema informático puede incluir unos procesadores que incluyan múltiples subprocesos 16 de hardware. En este ejemplo, el entorno 10 de tiempo de ejecución, y/o la capa 14 de gestión de recursos pueden particionar cada paquete de procesador en un nodo separado y crear un nodo 30 de programación para cada nodo.

En otro ejemplo, en un sistema NUMA, la diferencia en las latencias de memoria entre procesadores y diferentes partes de una memoria pueden usarse como métricas de ejecución para dividir el sistema informático en nodos NUMA y crear un nodo 30 de programación para cada nodo NUMA. Los nodos NUMA pueden tener cada uno de los mismos un conjunto de recursos de procesamiento y una memoria local donde el acceso a la memoria local por los recursos de procesamiento dentro de un nodo NUMA es más rápido que el acceso a una memoria local en otro nodo NUMA por los recursos de procesamiento.

En un ejemplo adicional, el entorno 10 de tiempo de ejecución, y/o la capa 14 de gestión de recursos pueden particionar los conjuntos arbitrarios o parcialmente arbitrarios de recursos de procesador en un sistema informático en los nodos y crear un nodo 30 de programación para cada nodo.

En otro ejemplo más, el entorno 10 de tiempo de ejecución, y/o la capa 14 de gestión de recursos pueden particionar los recursos de procesamiento de diferentes tipos o velocidades en nodos donde cada nodo incluye un número del mismo tipo o velocidad de recurso de procesamiento. El entorno 10 de tiempo de ejecución y/o la capa 14 de gestión de recursos crean un nodo 30 de programación para cada nodo.

El entorno 10 de tiempo de ejecución, la capa 14 de gestión de recursos, y/o el programador 22 crean una

recopilación 40 de programación respectiva para cada nodo 30 de programación como se indica en el bloque 54. Como se muestra en la figura 1, el programador 22 crea unas recopilaciones 40 (1)-40 (L) de programación que corresponden a los nodos 30 (1)-30 (L) de programación respectivos. Cada recopilación 40 de programación forma una estructura de datos en la memoria del sistema informático para almacenar tareas donde la estructura de datos puede buscarse por los procesadores 32 virtuales desde un nodo 30 de programación correspondiente y los procesadores 32 virtuales desde otros nodos 30 de programación.

El entorno 10 de tiempo de ejecución, la capa 14 de gestión de recursos, y/o el programador 22 asignan las recopilaciones 40 (1)-40 (L) de programación en un orden de búsqueda total o parcial basándose en una o más métricas de ejecución, como se indica en el bloque 56. El programador 22 usa las métricas de ejecución para comparar los costes de ejecución entre diferentes nodos 30 de programación. Los costes de ejecución pueden describirse en términos de distancias de nodo donde las diferentes distancias de nodo expresan diferentes características de ejecución entre un nodo 30 de programación dado y otros nodos 30 de programación. Con las distancias de nodo, los nodos 30 de programación con costes de ejecución más bajos en relación con un nodo 30 de programación dado se describen como que están más cercanos al nodo 30 de programación dado y los nodos 30 de programación con costes de ejecución más altos en relación con el nodo 30 de programación dado se describen como que están más alejados del nodo 30 de programación dado. El Programador 22 asigna las recopilaciones 40 (1)-40 (L) de programación en el orden de búsqueda completo o parcial usando las distancias de los nodos en una realización.

Para crear el orden de búsqueda, el programador de 22 agrupa el conjunto de recopilaciones 40 de programación en subconjuntos de una o más recopilaciones 40 de programación basándose en las distancias de nodo. Cada recopilación 40 de programación tiene una distancia de nodo de cero a partir de un nodo 30 de programación correspondiente. En consecuencia, cada recopilación 40 de programación forma el primer subconjunto de nivel de las recopilaciones 40 de programación (por ejemplo, un subconjunto de nivel 0) para el nodo 30 de programación correspondiente. Para el siguiente subconjunto de nivel de recopilaciones 40 de programación (por ejemplo, un subconjunto de nivel 1), el programador 22 agrupa el conjunto de una o más recopilaciones 40 de programación con un intervalo más cercano de distancias de nodo a partir del nodo 30 de programación dado. A continuación, el programador 22 agrupa el conjunto de una o más recopilaciones 40 de programación con un intervalo más próximo de distancias de nodo a partir del nodo 30 de programación dado hasta el siguiente subconjunto de nivel de las recopilaciones 40 de programación (por ejemplo, un subconjunto de nivel 2). El programador 22 continúa agrupando los conjuntos de una o más recopilaciones 40 de programación con rangos sucesivos de distancias de nodo desde el nodo 30 de programación dado hasta los subconjunto de nivel sucesivos de las recopilaciones 40 de programación hasta que todas las recopilaciones 40 de programación deseadas en el conjunto de recopilaciones 40 de programación se hayan incorporado al orden de búsqueda.

El orden de búsqueda de las recopilaciones 40 de programación se usa por los recursos de procesamiento disponibles (es decir, los procesadores 32 virtuales) en los nodos 30 de programación para buscar tareas a ejecutar. El orden de búsqueda puede especificar un orden de búsqueda parcial agrupando más de una recopilación 40 de programación en al menos algunos de los subconjuntos (por ejemplo, un subconjunto de dos o más recopilaciones 40 de programación que corresponden a un subconjunto de nodos 30 de programación que tienen la misma distancia de nodo o distancias de nodo similares desde el nodo 30 de programación dado). Cuando se especifica un orden parcial, un recurso de procesamiento puede buscar en el subconjunto de recopilaciones 40 de programación en un turno rotativo u otra orden adecuada. El orden de búsqueda también puede especificar un orden de búsqueda completo agrupando o bien solo una recopilación 40 de programación en cada subconjunto o especificando un orden de búsqueda de cada subconjunto de dos o más recopilaciones 40 de programación.

Las figuras 3A-3B son un diagrama y una tabla, respectivamente, que ilustran una realización de un orden 60 de búsqueda parcial en un sistema 61 informático NUMA con cuatro paquetes de procesador que incluyen unos conjuntos respectivos de cuatro subprocesos 16 de hardware. Una memoria local respectiva se conecta a cada procesador (no mostrado). Debido a que cada subproceso 16 de hardware en un paquete de procesador tiene métricas de ejecución similares, cada paquete de procesador forma un nodo de nivel 0 en el ejemplo de las figuras 3A-3B. En consecuencia, el nodo 30 (1) de programación de nivel 0 incluye los subprocesos 16 (1)-16 (4) de hardware, el nodo 30 (2) de programación de nivel 0 incluye los subprocesos 16 (5)-16 (8) de hardware, el nodo 30 (3) de programación de nivel 0 incluye los subprocesos 16 (9)-16 (12) de hardware, y el nodo 30 (4) de programación de nivel 0 incluye los subprocesos 16 (9)-16 (12) de hardware. Los nodos 30 (1)-30 (4) de programación de nivel 0 corresponden a los subconjuntos de nivel 0 respectivos de las recopilaciones 40 (1)-40 (4) de programación.

Como se muestra en la figura 3A, los nodos 30 (1)-30 (2) de programación comparten una interconexión 62 (1) entre los nodos 30 (1)-30 (2), los nodos 30 (1)-30 (3) de programación comparten una interconexión 62 (2) entre los nodos 30 (1)-30 (3), los nodos 30 (2)-30 (4) de programación comparten una interconexión 62 (3) entre los nodos 30 (2)-30 (4), y los nodos 30 (3)-30 (4) de programación comparten una interconexión 62 (4) entre los nodos 30 (3)-30 (4). Se supone que todas las interconexiones 62 (1)-62 (4) tienen las mismas características de velocidad y ancho de banda en el ejemplo de la figura 3A.

Las distancias de nodo entre dos nodos 30 cualquiera que comparten una interconexión 62 son menores que las

distancias de nodo entre dos nodos 30 cualquiera que no comparten una interconexión 62. Por ejemplo, el nodo 30 (1) accede al nodo 30 (4) usando o bien ambas interconexiones 62 (1) y 62 (3) o ambas interconexiones 62 (2) y 62 (4). De manera similar, el nodo 30 (2) accede al nodo 30 (3) usando o bien las interconexiones 62 (1) y 62 (2) o ambas interconexiones 62 (3) y 62 (4).

Desde el nodo 30 (1), el subconjunto de nivel 1 de recopilaciones 40 de programación incluye las recopilaciones 40 (2)-40 (3) de programación que corresponden a los nodos 30 (2)-30 (3) de programación y el subconjunto de nivel 2 de recopilaciones 40 de programación incluye la recopilación de programación 40 (4) que corresponde al nodo 30 (4) de programación.

Desde el nodo 30 (2), el subconjunto de nivel 1 de recopilaciones 40 de programación incluye las recopilaciones 40 (1)-40 (4) de programación que corresponden a los nodos 30 (1)-30 (4) de programación y el subconjunto de nivel 2 de recopilaciones 40 de programación incluye la recopilación de programación 40 (3) que corresponde al nodo 30 (3) de programación.

Desde el nodo 30 (3), el subconjunto de nivel 1 de recopilaciones 40 de programación incluye las recopilaciones 40 (1)-40 (4) de programación que corresponden a los nodos 30 (1)-30 (4) de programación y el subconjunto de nivel 2 de recopilaciones 40 de programación incluye la recopilación de programación 40 (2) que corresponde al nodo 30 (2) de programación.

Desde el nodo 30 (4), el subconjunto de nivel 1 de recopilaciones 40 de programación incluye las recopilaciones 40 (2)-40 (3) de programación que corresponden a los nodos 30 (2)-30 (3) de programación y el subconjunto de nivel 2 de recopilaciones 40 de programación incluye la recopilación de programación 40 (1) que corresponde al nodo 30 (1) de programación.

Haciendo referencia de nuevo a la figura 2, el programador 22 llena las recopilaciones 40 (1)-40 (M) de programación con los conjuntos respectivos de tareas como se indica en el bloque 58. Cada conjunto de una o más tareas presentado al programador 22 puede crearse explícitamente por el proceso 12 o implícitamente por el entorno 10 de tiempo de ejecución (por ejemplo, creando un agente sin un padre o induciendo un contexto de ejecución de sistema operativo en un contexto de ejecución del programador 22). El programador 22 inserta los conjuntos de tareas en las recopilaciones 40 de programación de acuerdo con cualquier algoritmo adecuado o de acuerdo con la topología de los nodos 30 de programación. Por ejemplo, el programador 22 puede insertar conjuntos de tareas en las recopilaciones 40 de programación en un orden de turno rotativo. Como otro ejemplo, el programador 22 puede insertar conjuntos de tareas en recopilaciones de programación correspondientes a las topologías deseadas de los nodos 30 de programación.

La figura 4 es un diagrama de flujo que ilustra una realización de un procedimiento para seleccionar tareas para su ejecución. El procedimiento de la figura 4 se describirá haciendo referencia a la realización del programador 22 en la figura 1.

El programador 22 determina si un procesador 32 virtual vuelve a estar disponible, como se indica en el bloque 72. El programador 22 puede realizar esta función continuamente mientras provoca que se ejecute el proceso 12. Una vez finalizado, un bloqueo u otra interrupción (por ejemplo, rendimiento explícito o preferencia forzada) de un contexto de ejecución 34 que se ejecuta en un procesador 32 virtual, el procesador 32 virtual estará disponible para ejecutar una nueva tarea.

Cuando el programador 22 determina que un procesador 32 virtual esté disponible, el programador 22 comienza una búsqueda de una tarea a ejecutar para el procesador 32 virtual disponible. En primer lugar, el programador 22 intenta localizar una tarea a ejecutar en un primer subconjunto de recopilaciones 40 de programación como se indica en el bloque 74. El primer subconjunto de recopilaciones 40 de programación es la recopilación 40 de programación correspondiente al nodo 30 de programación que incluye el procesador 32 virtual disponible. El programador 22 puede buscar el primer subconjunto de cualquier manera adecuada.

Si se encuentra una tarea ejecutable en el primer subconjunto, entonces el programador 22 hace que la tarea se ejecute por el procesador 32 virtual como se indica en el bloque 76. El procesador 32 virtual intenta ejecutar la tarea como una continuación de un contexto 34 de ejecución anterior. Si el procesador 32 virtual no puede ejecutar la tarea como una continuación, entonces el procesador 32 virtual realiza un cambio de contexto de sistema operativo completo al contexto de ejecución representado por la tarea.

Si no se encuentra una tarea ejecutable en el primer subconjunto, entonces el programador 22 determina si otro subconjunto de recopilaciones 40 de programación está especificado por el orden de búsqueda como se indica en el bloque 78. Si el primer subconjunto de nivel es el único subconjunto especificado por el orden de búsqueda, entonces el programador 22 continúa buscando el primer subconjunto hasta que se localice una tarea ejecutable.

Si se especifica otro subconjunto por el orden de búsqueda, entonces el programador 22 intenta localizar una tarea a ejecutar en una o más recopilaciones 40 de programación en el siguiente subconjunto como se indica en el bloque 80. Si se encuentra una tarea ejecutable en una recopilación 40 de programación en el siguiente subconjunto, entonces el programador 22 hace que la tarea se ejecute por el procesador 32 virtual como se indica en el bloque

82. Si no se encuentra una tarea ejecutable en el siguiente subconjunto de las recopilaciones 40 de programación, entonces el programador 22 repite la función del bloque 78. El programador 22 continúa buscando subconjuntos de recopilaciones 40 de programación en el orden de búsqueda especificado hasta que se encuentre una tarea ejecutable o se hayan buscado todos los subconjuntos especificados por el orden de búsqueda.

En las realizaciones anteriores, el programador 22 puede estar configurado para buscar uno o más de los subconjuntos anteriores de recopilaciones 40 de programación repetidamente antes de pasar al siguiente subconjunto. El programador 22 también puede configurarse para retrasar la búsqueda de uno o más de los subconjuntos de acuerdo con uno o más parámetros de retardo.

En las realizaciones anteriores, los nodos 30 de programación se apropian efectivamente de las recopilaciones 40 de programación correspondientes. En algún momento de la ejecución del proceso 12, todos los recursos de procesamiento de un nodo 30 de programación dado pueden estar ejecutando tareas a partir de otras recopilaciones 40 de programación que la recopilación 40 de programación que corresponde al nodo 30 de programación dado. En este escenario, la recopilación 40 de programación apropiada del nodo 30 de programación dado se convierte en la recopilación 40 de programación a partir de la que la mayoría de los recursos de procesamiento del nodo 30 de programación dado ejecutan tareas y el nodo 30 de programación dado se convierte en un nodo incoherente. Si un nodo incoherente tiene más tarde un recurso de procesamiento que está ejecutando una tarea a partir de la recopilación 40 de programación apropiada originalmente, entonces el nodo incoherente pasa de nuevo a ser el propietario de la recopilación 40 de programación apropiada originalmente.

Las figuras 5A-5B son diagramas de bloques que ilustran las realizaciones 40A y 40B respectivas de las recopilaciones 40 de programación.

La figura 5A es un diagrama de bloques que ilustra la realización 40A de la recopilación 40 de programación que incluye un conjunto de grupos 90 (1)-90 (P) de programación, donde P es un número entero mayor o igual que uno e indica el P-ésimo grupo 90 de programación. El conjunto de los grupos 90 (1)-90 (P) de programación se disponen en un anillo de programación como se indica por una flecha 96. Cada grupo 90 de programación incluye una recopilación 92 ejecutable, una cola 93 de trabajo y un conjunto de cero o más colas 94 de robo de trabajo. Cada recopilación 92 ejecutable contiene una lista de tareas ejecutables o contextos de ejecución. El programador 22 suma un contexto de ejecución a la recopilación 92 ejecutable cuando un contexto de ejecución se desbloquea o se presenta un nuevo contexto de ejecución ejecutable (posiblemente una demanda creada) al programador 22 por el proceso 12. La cola 93 de trabajo contiene una lista de colas 94 de robo de trabajo como indica un flecha 95 y realiza un seguimiento de los contextos de ejecución que están ejecutando tareas a partir de las colas 93 de trabajo. Cada cola 94 de robo de trabajo incluye una o más tareas no realizadas sin contexto de ejecución asignado.

El programador 22 llena las recopilaciones 40A de programación (figura 1) con conjuntos respectivos de cero o más grupos 90 de programación en cualquier momento (por ejemplo, en respuesta a la ejecución de otras tareas) donde cada grupo 90 de programación incluye un conjunto de tareas del proceso 12. En tales realizaciones, el programador 22 puede buscar en cada grupo 90 de programación en una recopilación 40A de programación antes de buscar una tarea en otra recopilación 40 o 40A de programación.

El programador 22 puede intentar localizar una tarea a ejecutar en el grupo 90 de programación a partir de la que un procesador 32 virtual disponible obtuvo más recientemente una tarea ejecutable o en el grupo 90 de programación se indica por un índice 97 (por ejemplo, un índice de turno rotativo). En cada grupo 90 de programación, el programador 22 puede buscar tareas realizadas en la recopilación 92 ejecutable del grupo 90 de programación antes de buscar una tarea realizada en otros grupos 90 de programación (por ejemplo, en un orden de turno rotativo). Si no se encuentra una tarea realizada, entonces el programador 22 puede buscar tareas no realizadas en las colas 94 de robo de trabajo del grupo 90 de programación antes de buscar una tarea no realizada en otros grupos 90 de programación (por ejemplo, en un orden de turno rotativo). El programador 22 puede actualizar el índice 97 para identificar un grupo 90 de programación donde se ha encontrado una tarea ejecutable.

El proceso 12 pueden usar los grupos 90 de programación en el programador 22 para proporcionar una estructura para la localidad de trabajo, la equidad, y el progreso hacia delante. Las tareas de cada grupo 90 de programación pueden agruparse debido a un trabajo lógicamente relacionado (por ejemplo, una recopilación de tareas que descienden de una tarea raíz común), a una topología de hardware (por ejemplo, una arquitectura de memoria no uniforme (NUMA)), o a una combinación de los mismos.

La figura 5B es un diagrama de bloques que ilustra la realización 40B de una recopilación 40 de programación que incluye las recopilaciones locales de las tareas 44 (1)-44 (N) correspondientes al procesador 32 (1)-32 (N) virtual respectivo.

En realizaciones donde una o más recopilaciones 40 de programación, que incluyen recopilaciones 44 locales, el conjunto de contextos de ejecución en el programador 22 también incluye conjuntos de contextos 46 (1)-46 (N) de ejecución ejecutables en las recopilaciones 44 (1)-44 (N) locales respectivas. Cada contexto 46 de ejecución tiene una tarea 47 asociada que se ha desbloqueado por la ejecución de una tarea 36 donde la tarea 36 se ha ejecutado o se está ejecutando actualmente en el procesador 32 virtual correspondiente a la recopilación 44 local que incluye el

contexto 46 de ejecución.

En primer lugar, el programador 22 puede intentar localizar una tarea en la recopilación 44 local correspondiente al procesador 32 virtual disponible antes de buscar en otra parte de la recopilación 40B de programación. Las recopilaciones 44 locales pueden permitir que el programador 22 explote la localidad de memoria y otros efectos que pueden producirse con los subprocesos 16 de hardware. Al ejecutar el proceso 12, el programador 22 puede asignar cada contexto de ejecución dependiente-en espera que se desbloquee en la recopilación 44 local correspondiente al procesador 32 virtual que provocó que el contexto de ejecución se desbloquee. Cuando un procesador 32 virtual se vuelve disponible, el procesador 32 virtual puede intentar ejecutar el contexto de ejecución sumado más recientemente en la recopilación 44 local correspondiente para tratar de aprovechar los datos almacenados en la jerarquía de memoria correspondiente al procesador 32 virtual.

Si no se encuentra una tarea ejecutable en la recopilación 44 local correspondiente al procesador 32 virtual disponible, entonces el programador 22 puede intentar localizar una tarea ejecutable en la recopilación 44 local correspondiente a otro procesador 32 virtual de un nodo 30 de programación. El programador 22 accede a las recopilaciones 44 locales correspondientes a los otros procesadores 32 virtuales en un orden de turno rotativo u otro orden adecuado y puede ejecutar el último contexto de ejecución recientemente sumado en la recopilación 44 local donde se encuentra una tarea ejecutable.

En otras realizaciones, otras recopilaciones 40 de programación pueden incluir tanto los grupos 90 de programación de la recopilación 40A de programación (figura 5A) como las recopilaciones 44 locales de la recopilación 40B de programación (figura 5B).

Las figuras 6A-6B son diagramas de bloques que ilustran las realizaciones 100A y 100B, respectivamente, de un sistema 100 informático configurado para implementar el entorno 10 de tiempo de ejecución que incluye el programador 22 con las recopilaciones 40 de programación.

Como se muestra en la figura 6A, el sistema 100A informático incluye uno o más paquetes 102 de procesador, un sistema 104 de memoria, cero o más dispositivos 106 de entrada/salida, cero o más dispositivos 108 de visualización, cero o más dispositivos 110 periféricos, y cero o más dispositivos 112 de red. Los paquetes 102 de procesador, el sistema 104 de memoria, los dispositivos 106 de entrada/salida, los dispositivos 108 de visualización, los dispositivos 110 periféricos y los dispositivos 112 de red se comunican usando un conjunto de interconexiones 114 que incluye cualquier tipo, número y configuración adecuados de controladores, buses, interfaces y/u otras conexiones cableadas o inalámbricas.

El sistema 100A informático representa cualquier dispositivo de procesamiento adecuado configurado para un fin general o un fin específico. Los ejemplos del sistema 100A informático incluyen un servidor, un ordenador personal, un ordenador portátil, una tableta, un asistente digital personal (PDA), un teléfono móvil y un dispositivo de audio/video. Los componentes del sistema 100A informático (es decir, paquetes 102 de procesador, sistema 104 de memoria, dispositivos 106 de entrada/salida, dispositivos 108 de visualización, dispositivos 110 periféricos, dispositivos 112 de red e interconexiones 114) pueden estar contenidos en una carcasa común (no mostrada) o en cualquier número adecuado de carcasas separadas (no mostradas).

Los paquetes 102 de procesador incluyen los subprocesos 16 (1)-16 (M) de hardware. Cada subproceso 16 de hardware en los paquetes 102 de procesador está configurado para acceder y ejecutar las instrucciones almacenadas en el sistema 104 de memoria. Las instrucciones pueden incluir un sistema básico de salida de entrada (BIOS) o firmware (no mostrado), un sistema 120 operativo (SO), una plataforma 122 de tiempo de ejecución, unas aplicaciones 124 y la capa 14 de gestión de recursos (también mostradas en la figura 1). Cada subproceso 16 de hardware puede ejecutar las instrucciones junto con o en respuesta a la información recibida desde los dispositivos 106 de entrada/salida, los dispositivos 108 de visualización, los dispositivos 110 periféricos y los dispositivos 112 de red.

El sistema 100A informático arranca y ejecuta el SO 120. El SO 120 incluye instrucciones ejecutables por los paquetes 102 de procesador para gestionar los componentes del sistema 100A informático y proporcionar un conjunto de funciones que permiten a las aplicaciones 124 acceder y usar los componentes. En una realización, el SO 120 es el sistema operativo Windows. En otras realizaciones, el SO 120 es otro sistema operativo adecuado para su uso con el sistema 100A informático.

La capa 14 de gestión de recursos incluye instrucciones que pueden ejecutarse junto con el SO 120 para asignar recursos del sistema 100A informático que incluye los subprocesos 16 de hardware como se ha descrito anteriormente haciendo referencia a la figura 1. La capa 14 de gestión de recursos puede incluirse en el sistema 100A informático como una biblioteca de funciones disponibles para una o más aplicaciones 124 o como parte integrada del SO 120.

La plataforma 122 de tiempo de ejecución incluye instrucciones que pueden ejecutarse junto con el SO 120 y la capa 14 de gestión de recursos para generar el entorno 10 de tiempo de ejecución y proporcionar funciones de tiempo de ejecución a las aplicaciones 124. Estas funciones de tiempo de ejecución incluyen una función de programador como se ha descrito con más detalle anteriormente haciendo referencia a figura 1. Las funciones de tiempo de

ejecución pueden incluirse en el sistema 100A informático como parte de una aplicación 124, como una biblioteca de funciones disponibles para una o más aplicaciones 124, o como parte integrada del SO 120 y/o de la capa 14 de gestión de recursos.

Cada aplicación 124 incluye instrucciones que pueden ejecutarse junto con el SO 120, la capa 14 de gestión de recursos, y/o la plataforma 122 de tiempo de ejecución para provocar las operaciones deseadas a realizar por el sistema 100A informático. Cada aplicación 124 representa uno o más procesos, tal como el proceso 12 como se ha descrito anteriormente, que puede ejecutarse con el programador 22 de acuerdo con lo provisto por la plataforma 122 de tiempo de ejecución.

El sistema 104 de memoria incluye cualquier tipo, número y configuración adecuados de dispositivos de almacenamiento volátiles o no volátiles configurados para almacenar instrucciones y datos. Los dispositivos de almacenamiento del sistema 104 de memoria representan unos medios de almacenamiento legibles por ordenador que almacenan las instrucciones ejecutables por ordenador, incluyendo el SO 120, la capa 14 de gestión de recursos, la plataforma 122 de tiempo de ejecución y las aplicaciones 124. Las instrucciones pueden ejecutarse por el sistema informático para realizar las funciones y los procedimientos del SO 120, la capa 14 de gestión de recursos, la plataforma 122 de tiempo de ejecución y las aplicaciones 124 descritas en el presente documento. Los ejemplos de dispositivos de almacenamiento en el sistema 104 de memoria incluyen unidades de disco duro, memoria de acceso aleatorio (RAM), memoria de solo lectura (ROM), unidades y tarjetas de memoria flash y discos magnéticos y ópticos.

El sistema 104 de memoria almacena instrucciones y datos recibidos desde los paquetes 102 de procesador, los dispositivos 106 de entrada/salida, los dispositivos 108 de visualización, los dispositivos 110 periféricos, y los dispositivos 112 de red. El sistema 104 de memoria proporciona instrucciones y datos almacenados a los paquetes 102 de procesador, a los dispositivos 106 de entrada/salida, a los dispositivos 108 de visualización, a los dispositivos 110 periféricos y a los dispositivos 112 de red.

Los dispositivos 106 de entrada/salida incluyen cualquier tipo, número y configuración adecuados de los dispositivos de entrada/salida configurados para introducir instrucciones o datos de un usuario al sistema 100A informático y emitir instrucciones o datos del sistema 100A informático al usuario. Los ejemplos de los dispositivos 106 de entrada/salida incluyen un teclado, un ratón, una alfombrilla táctil, una pantalla táctil, botones, diales, mandos y conmutadores.

Los dispositivos 108 de visualización incluyen cualquier tipo, número y configuración adecuados de los dispositivos de visualización configurados para emitir información de texto y/o gráfica para un usuario del sistema 100A informático. Los ejemplos de dispositivos 108 de visualización incluyen un monitor, una pantalla de visualización y un proyector.

Los dispositivos 110 periféricos incluyen cualquier tipo, número y configuración adecuados de los dispositivos periféricos configurados para operar con uno o más de otros componentes en el sistema 100A informático para realizar funciones de procesamiento generales o específicas.

Los dispositivos 112 de red incluyen cualquier tipo, número y configuración adecuados de los dispositivos de red configurados para permitir que el sistema 100A informático se comunique a través de una o más redes (no mostradas). Los dispositivos 112 de red pueden funcionar de acuerdo con cualquier protocolo y/o configuración de red adecuados para permitir que la información se transmita por el sistema 100A informático a una red o se reciba por el sistema 100A informático desde una red.

La figura 6B es un diagrama de bloques que ilustra la realización 100B del sistema 100 informático. El sistema 100B informático incluye también al menos unos paquetes 102 de procesador y el sistema 104 de memoria. Los paquetes 102 de procesador incluyen los paquetes 102(1)-102 (R) de procesador y el sistema 104 de memoria incluye los conjuntos de dispositivos 128 (1)-128 (R) de memoria donde R es un número entero mayor o igual que dos y representa el R-ésimo paquete 102 de procesador y el R-ésimo conjunto de dispositivos 128 de memoria. El SO 120, la plataforma 122 de tiempo de ejecución, la aplicaciones 124 y la capa 14 de gestión de recursos pueden almacenarse cada uno en cualquiera de los dispositivos 104 (1)-104 (R) de memoria adecuados.

En la realización de la figura 6B, cada paquete 102 (1)-102 (R) de procesador y el conjunto respectivo de dispositivos 128 (1)-128 (R) de memoria forman un nodo. Los nodos están interconectados con cualquier tipo, número y/o combinación adecuados de interconexiones 130 de nodos. La velocidad y/o el ancho de banda de las interconexiones 130 pueden variar entre los nodos.

Cada paquete 102 de procesador incluye un conjunto de subprocesos 16 (1)-16 (4) de hardware donde cada subproceso de hardware incluye una caché de L1 (nivel uno) (no mostrada). Cada paquete 102 de procesador incluye también un conjunto de cachés 132 (1)-132 (4) de L2 (nivel dos) que corresponden a los subprocesos 16 (1) (1)-16 (1) (4) de hardware respectivos. Cada paquete 102 de procesador incluye además una caché de L3 (nivel tres) disponible para el conjunto de subprocesos 16 (1)-16 (4) de hardware, una interfaz 136 de recursos de sistema, un conmutador 138 de barras cruzadas, un controlador 140 de memoria y una interfaz 142 de nodo. La interfaz 136 de recursos de sistema proporciona acceso a los recursos de nodo (no mostrado). El conmutador 138 de barras

cruzadas interconecta la interfaz 136 de recursos de sistema con el controlador 140 de memoria y la interfaz 142 de nodo. El controlador 140 de memoria se conecta a un dispositivo 128 de memoria. La interfaz 142 de nodo se conecta a una o más interconexiones 130 de nodo.

Debido a que un nodo incluye memoria local (es decir, un conjunto de dispositivos 104 de memoria), el acceso a la memoria local por los paquetes 102 de procesador en el nodo puede ser más rápido que el acceso a la memoria en otros nodos. Además, el acceso a la memoria en otros nodos puede depender de la velocidad de conexión, el ancho de banda, la topología de la caché y/o la distancia de nodo NUMA de las interconexiones 130 entre los nodos. Por ejemplo, algunos nodos pueden estar conectados con una interconexión 130 relativamente rápida, tal como un bus de hipertransporte de microdispositivos avanzado o un bus CSI de Intel, mientras que otros pueden estar conectados con una o más interconexiones 130 relativamente lentas.

En otras realizaciones, cada paquete 102 de procesador puede incluir otras configuraciones y/o números de cachés. Por ejemplo, cada subproceso 16 de hardware puede incluir dos o más cachés de L1 en otras realizaciones y las cachés de L2 y/o L3 pueden o no compartirse en otras realizaciones. Como otro ejemplo, otras realizaciones pueden incluir cachés adicionales (por ejemplo, una caché de nivel cuatro (L4)) o menos o ninguna caché.

Haciendo referencia a las realizaciones descritas anteriormente en las figuras 1-5B, las latencias de memoria y de interconexión en el sistema 100B informático proporcionan distancias de nodos que pueden considerarse por el entorno 10 de tiempo de ejecución, la capa 14 de gestión de recursos, y/o el programador 22 en la formación de los nodos 30 de programación. Por ejemplo, el entorno 10 de tiempo de ejecución, la capa 14 de gestión de recursos y/o el programador 22 pueden crear un nodo 30 de programación para cada nodo en el sistema 100B informático junto con una recopilación 40 de programación correspondiente. El entorno 10 de tiempo de ejecución, la capa 14 de gestión de recursos, y/o el programador 22 pueden asignar las recopilaciones 40 de programación en un orden de búsqueda parcial o completa basándose en la topología de interconexión entre los nodos. Por ejemplo, cualquiera de dos nodos conectados con una interconexión 130 relativamente rápida pueden agruparse en el mismo nodo de programación y el nivel de subconjunto de recopilación de programación y los nodos con interconexiones 130 relativamente lentas pueden agruparse en un nodo de programación y un nivel de subconjunto de recopilación de programación por encima del nodo de programación y del nivel de subconjunto de recopilación de programación que incluye la interconexión 130 relativamente rápida.

Mediante la búsqueda en las recopilaciones 40 de programación de tareas ejecutables en el orden de búsqueda, los recursos de procesamiento en los nodos aumentan la posibilidad de explotar los efectos de localidad de memoria en el sistema 100B informático. Es más probable que las tareas de la misma recopilación 40 de programación tengan datos comunes que estén presentes en la jerarquía de memoria local de un nodo que las tareas de otra recopilación 40 de programación.

Además de las ventajas potenciales de localidad, el uso de nodos de programación y las recopilaciones de programación en las realizaciones anteriores puede proporcionar un programador con la capacidad de reducir la competencia entre los recursos de procesamiento que están buscando tareas a ejecutar. Los recursos de procesamiento en diferentes nodos de programación inician la búsqueda de tareas ejecutables en diferentes recopilaciones de programación correspondientes. Al hacerlo, puede reducirse el número de bloqueos u otras construcciones de sincronización colocadas en las recopilaciones de tareas en el programador.

El programador también puede escalar a sistemas informáticos con un gran número de recursos de procesamiento como resultado de la búsqueda localizada de tareas ejecutables. Además, el programador puede proporcionar una localidad de trabajo al mismo tiempo que preserva la imparcialidad y el progreso hacia delante usando la búsqueda por turnos rotativos y las colas de robo de trabajo en los grupos de programación.

REIVINDICACIONES

1. Un procedimiento realizado por un programador (22) en un proceso (12) de un sistema (100A; 100B) informático,
en el que el programador incluye una pluralidad de recopilaciones (40 (1), 40 (L)) de programación,
en el que cada recopilación de programación corresponde a un nodo (30 (1), 30 (L)) de programación respectivo,
en el que la pluralidad de recopilaciones de programación comprende una primera recopilación (40 (1)) de programación y una segunda recopilación (40 (L)) de programación,
en el que las recopilaciones de programación comprenden tareas,
en el que los nodos de programación comprenden unos recursos (16 (1), 16 (m_1), 16 (m_m), 16 (M), 32 (1), 32 (N_1), 32 (N_L)) de procesamiento,
comprendiendo el procedimiento:

en respuesta a uno de una primera pluralidad de recursos (16 (1), 16 (m_1), 32 (1), 32 (N_1)) de procesamiento en un primer nodo (30 (1)) de programación que está disponible, buscar una primera tarea (39, 41, 42) en una primera recopilación (40) de programación, en el que la primera recopilación de programación corresponde al primer nodo de programación;
en respuesta a la búsqueda de la primera tarea en la primera recopilación de programación, ejecutar la primera tarea con el uno de la primera pluralidad de recursos de procesamiento; y
en respuesta a no encontrar la primera tarea en la primera recopilación de programación, ejecutar, con el uno de la primera pluralidad de recursos de procesamiento, una segunda tarea a partir de la segunda recopilación de programación, en el que la segunda recopilación de programación corresponde a un segundo nodo (30 (L)) de programación que incluye una segunda pluralidad de recursos (16 (m_m), 16 (M), 32 (1), 32 (N_L) de procesamiento.

2. El procedimiento de la reivindicación 1, que comprende además:

buscar el primer conjunto (90 (1), 90 (L)) de grupos de programación para la primera tarea (39, 41, 42) en la primera recopilación (40 (1)) de programación,
en el que el primer conjunto de grupos de programación está incluido en la primera recopilación de programación,
en el que el primer conjunto de grupos de programación comprende la primera tarea.

3. El procedimiento de la reivindicación 2, que comprende además:

buscar un segundo conjunto de grupos de programación para la segunda tarea en la segunda recopilación (40 (L)) de programación,
en el que el segundo conjunto de grupos de programación está incluido en la segunda recopilación de programación,
en el que el segundo conjunto de grupos de programación comprende la segunda tarea.

4. El procedimiento de la reivindicación 1, en el que el uno de la primera pluralidad de recursos (16 (1), 16 (m_1), 32 (1), 32 (N_1)) de procesamiento incluye un primer procesador (32 (1), 32 (N_1)) virtual y un primer subproceso (16 (1), 16 (m_1)) de hardware.

5. El procedimiento de la reivindicación 1, que comprende además:

en respuesta a no encontrar una primera tarea en la primera recopilación (40 (1)) de programación, buscar la segunda tarea en la segunda recopilación (40 (L)) de programación basándose en al menos una métrica de ejecución que relacione el primer nodo (30 (1)) de programación con el segundo nodo (30 (L)) de programación,
en el que la al menos una métrica de ejecución se usa para comparar los costes de ejecución entre el primer nodo de programación y el segundo nodo de programación,
en el que los costes de ejecución se describen en términos de distancias de nodos.

6. El procedimiento de la reivindicación 1, en el que la búsqueda de la primera tarea (39, 41, 42) en la primera recopilación (40 (1)) de programación comprende además:

buscar una tarea (39, 41) realizada en un conjunto de grupos (90) de programación en la primera recopilación de programación,
en el que el conjunto de grupos de programación está incluido en la primera recopilación de programación,
en el que el conjunto de grupos de programación incluye una recopilación (92) de ejecutables,
en el que la recopilación de ejecutables comprende la tarea realizada, y
en el que la tarea realizada comprende un contexto (38, 40) de ejecución adjunto a la tarea;
y
en respuesta a no encontrar una tarea realizada en el conjunto de grupos de programación, buscar una tarea (42) no realizada en el conjunto de grupos de programación,
en el que la tarea no realizada no tiene un contexto de ejecución asignado.

7. Un medio (104; 128 (1), 128 (S), 132 (1) (1), 132 (1) (2), 132 (1) (3), 132 (1) (4), 134 (1)) de almacenamiento legible por ordenador que almacena instrucciones ejecutables por ordenador que, cuando se ejecutan en un sistema (100A; 100B) informático, realizan el procedimiento de acuerdo con la reivindicación 1.

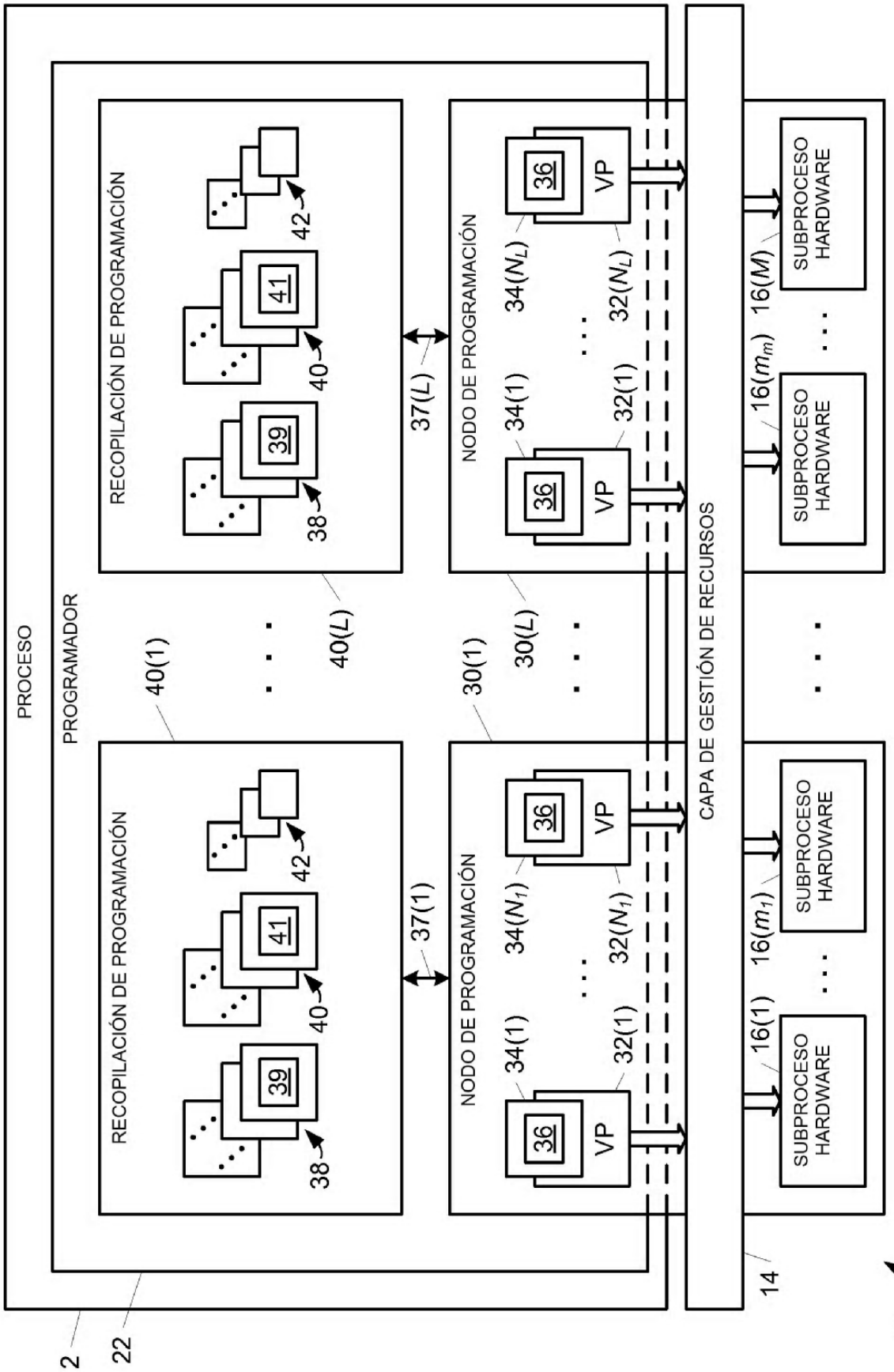
PROCESO
PROGRAMADOR
RECOPILACIÓN DE PROGRAMACIÓN
NODO DE PROGRAMACIÓN
CAPA DE GESTIÓN DE RECURSOS
SUBPROCESO HARDWARE
VP
12
22
14
10
16(1)
16(m1)
16(mm)
16(M)
30(1)
30(L)
32(1)
32(N1)
32(NL)
34(1)
34(N1)
34(NL)
36
37(1)
37(L)
38
39
40(1)
40(L)
40
41
42

Fig. 1

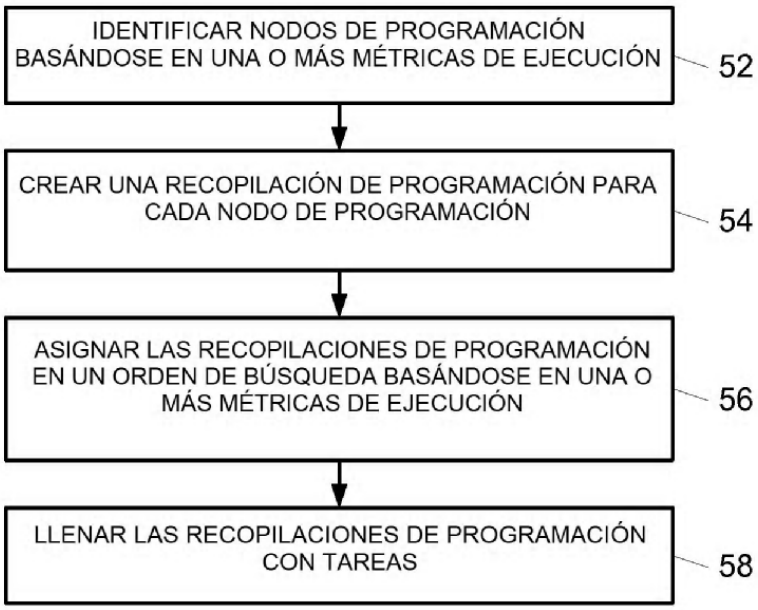

Fig. 2

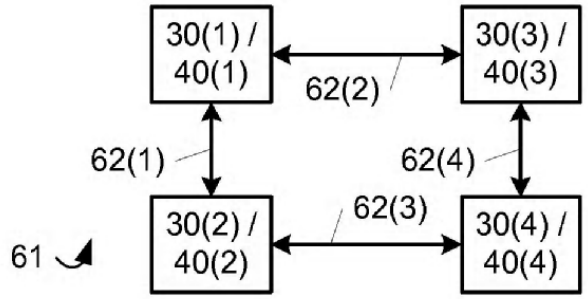

Fig. 3A

NODOS DE NIVEL 0 / SUBCONJUNTOS DE RECOPILACIONES	SUBPROCESOS DE HARDWARE	NODOS DE NIVEL 1 / SUBCONJUNTOS DE RECOPILACIONES	NODOS DE NIVEL 2 / SUBCONJUNTOS DE RECOPILACIONES
30(1) / 40(1)	16(1)-16(4)	30(2), 30(3) / 40(2), 40(3)	30(4) / 40(4)
30(2) / 40(2)	16(5)-16(8)	30(1), 30(4) / 40(1), 40(4)	30(3) / 40(3)
30(3) / 40(3)	16(9)-16(12)	30(1), 30(4) / 40(1), 40(4)	30(2) / 40(2)
30(4) / 40(4)	16(13)-16(16)	30(2), 30(3) / 40(2), 40(3)	30(1) / 40(1)

60

22

Fig. 3B

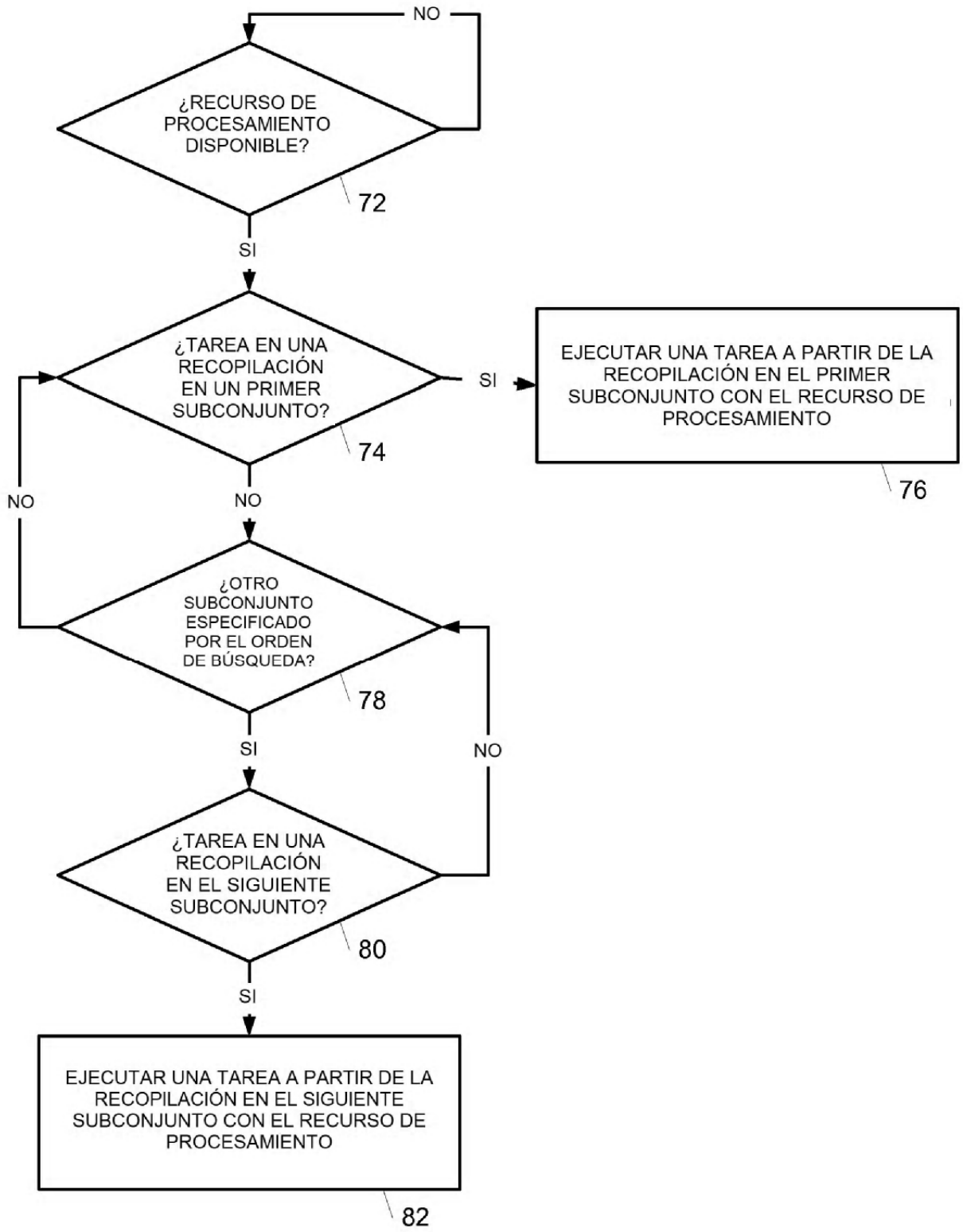
NO
¿RECURSO DE PROCESAMIENTO DISPONIBLE?
72
SI
¿TAREA EN UNA RECOPILACIÓN EN UN PRIMER SUBCONJUNTO?
74
SI
EJECUTAR UNA TAREA A PARTIR DE LA RECOPILACIÓN EN EL PRIMER SUBCONJUNTO CON EL RECURSO DE PROCESAMIENTO
76
NO
NO
¿OTRO SUBCONJUNTO ESPECIFICADO POR EL ORDEN DE BÚSQUEDA?
78
SI
NO
¿TAREA EN UNA RECOPILACIÓN EN EL SIGUIENTE SUBCONJUNTO?
80
SI
EJECUTAR UNA TAREA A PARTIR DE LA RECOPILACIÓN EN EL SIGUIENTE SUBCONJUNTO CON EL RECURSO DE PROCESAMIENTO
82

Fig. 4

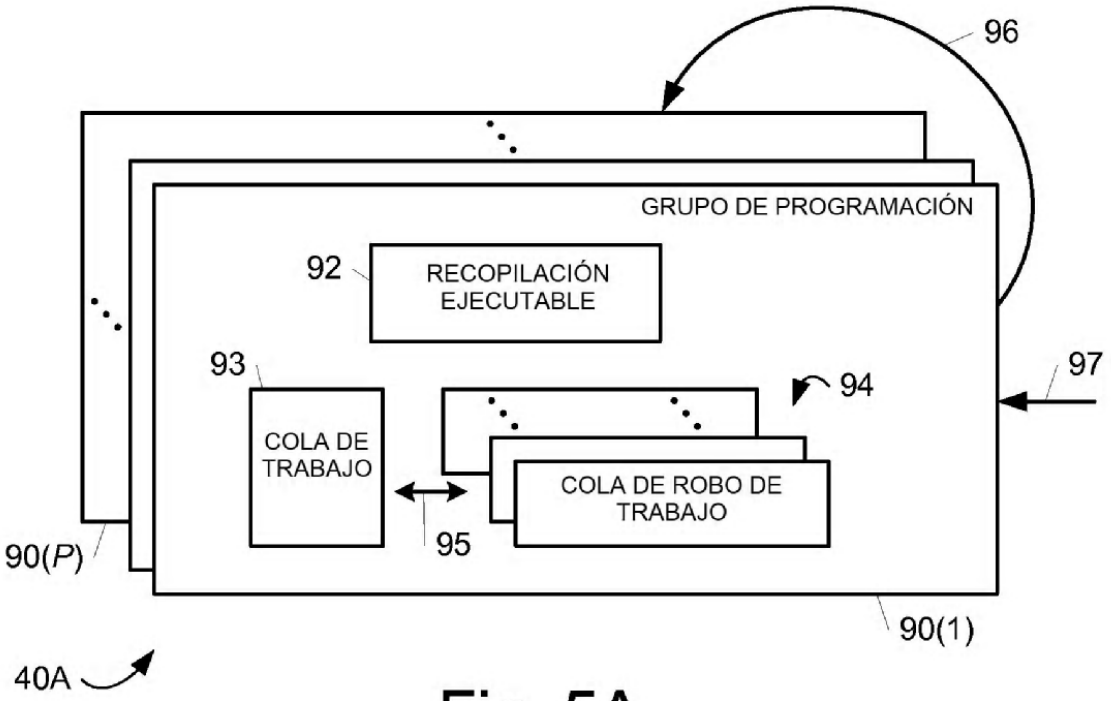
96
GRUPO DE PROGRAMACIÓN
92
RECOPILACIÓN EJECUTABLE
93
COLA DE TRABAJO
94
97
COLA DE ROBO DE TRABAJO
95
90(P)
90(1)
40A

Fig. 5A

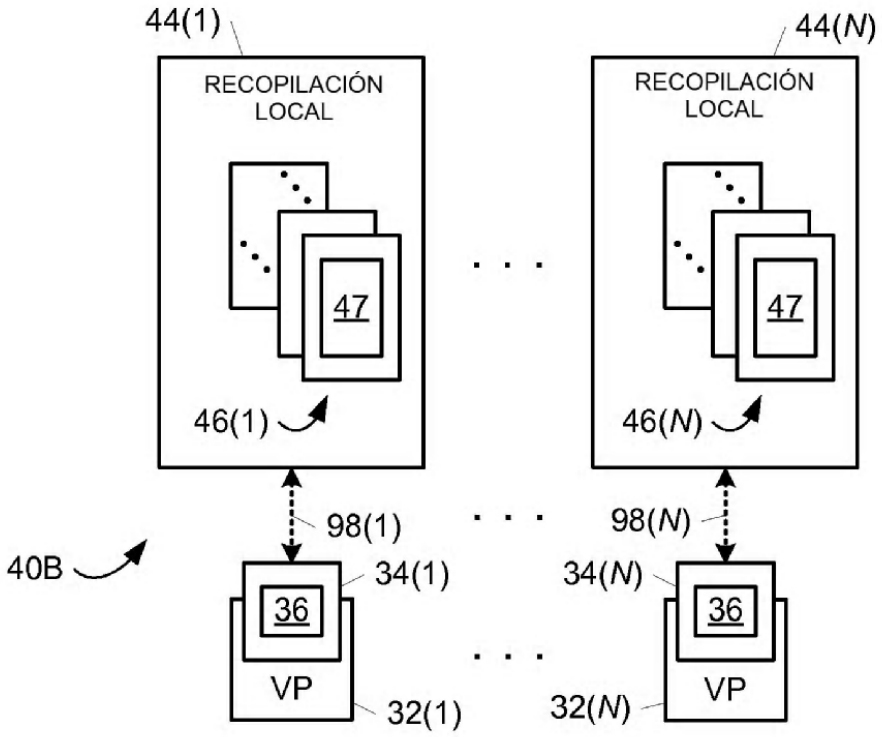
44(1)
44(N)
RECOPILACIÓN LOCAL
RECOPILACIÓN LOCAL
47
47
46(1)
46(N)
98(1)
98(N)
40B
34(1)
34(N)
36
36
VP
VP
32(1)
32(N)

Fig. 5B

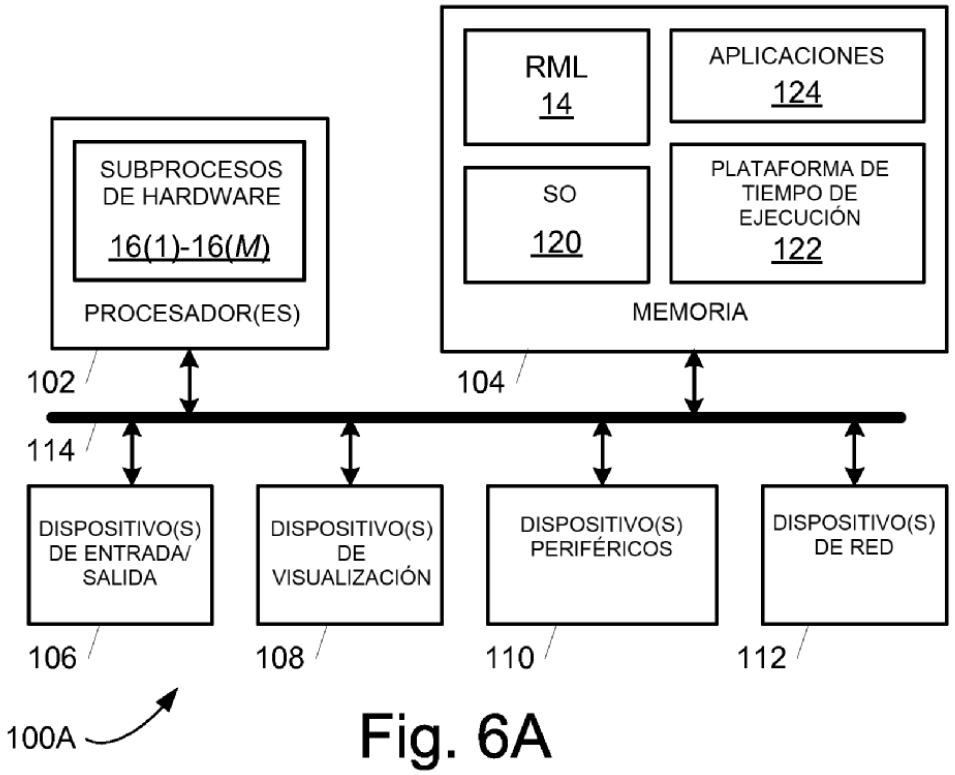
RML
14
APLICACIONES
124
SO
120
PLATAFORMA DE TIEMPO DE EJECUCIÓN
122
MEMORIA
SUBPROCESOS DE HARDWARE
16(1)-16(M)
PROCESADOR(ES)
102
104
114
DISPOSITIVO(S) DE ENTRADA/ SALIDA
DISPOSITIVO(S) DE VISUALIZACIÓN
DISPOSITIVO(S) PERIFÉRICOS
DISPOSITIVO(S) DE RED
106
108
110
112
100A

Fig. 6A

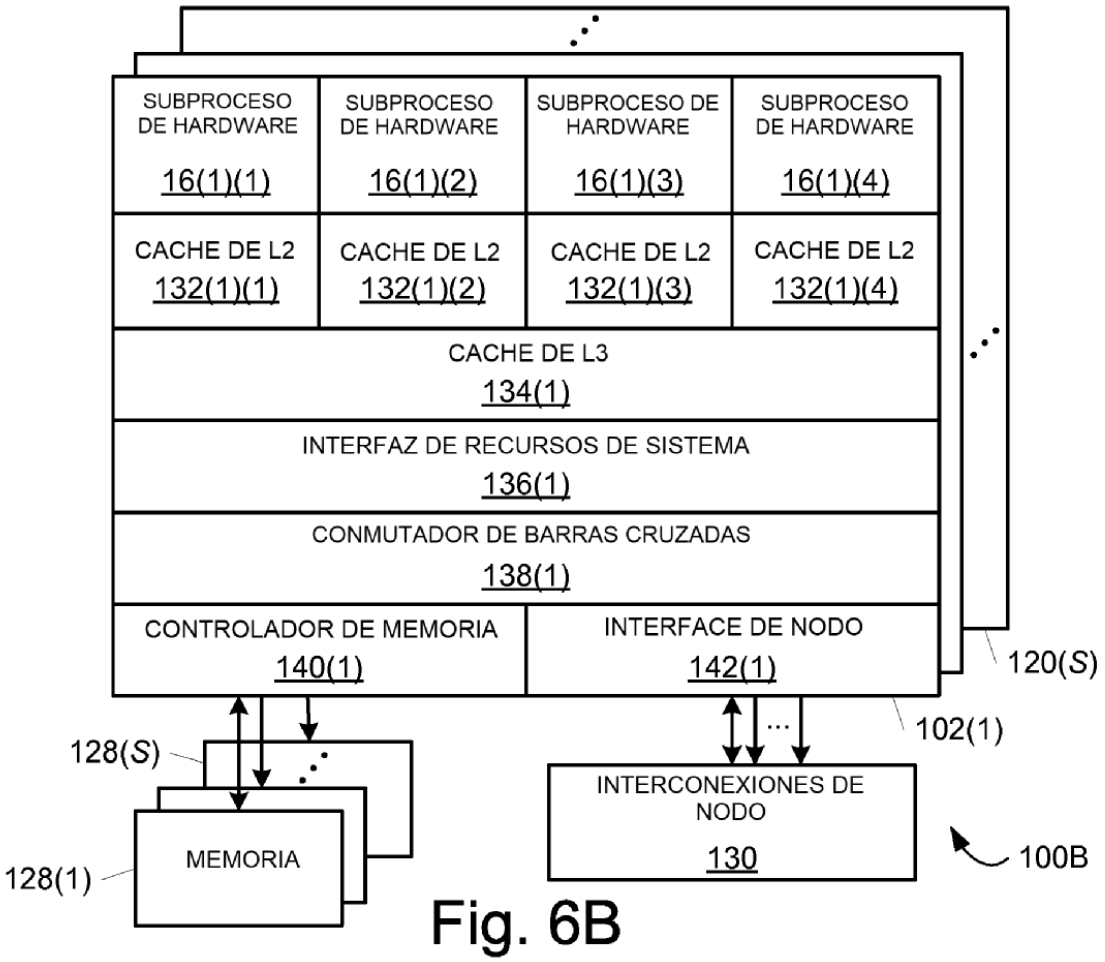
SUBPROCESO DE HARDWARE
16(1)(1)
SUBPROCESO DE HARDWARE
16(1)(2)
SUBPROCESO DE HARDWARE
16(1)(3)
SUBPROCESO DE HARDWARE
16(1)(4)
CACHE DE L2
132(1)(1)
CACHE DE L2
132(1)(2)
CACHE DE L2
132(1)(3)
CACHE DE L2
132(1)(4)
CACHE DE L3
134(1)
INTERFAZ DE RECURSOS DE SISTEMA
136(1)
CONMUTADOR DE BARRAS CRUZADAS
138(1)
CONTROLADOR DE MEMORIA
140(1)
INTERFACE DE NODO
142(1)
120(S)
102(1)
128(S)
MEMORIA
128(1)
INTERCONEXIONES DE NODO
130
100B

Fig. 6B