

(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2014 年 9 月 18 日(18.09.2014)



(10) 国際公開番号

WO 2014/141419 A1

- (51) 国際特許分類:
G06F 9/50 (2006.01) G06F 9/46 (2006.01)
- (21) 国際出願番号: PCT/JP2013/057141
- (22) 国際出願日: 2013 年 3 月 14 日(14.03.2013)
- (25) 国際出願の言語: 日本語
- (26) 国際公開の言語: 日本語
- (71) 出願人: 株式会社日立製作所 (HITACHI, LTD.)
[JP/JP]; 〒1008280 東京都千代田区丸の内一丁目
6 番 6 号 Tokyo (JP).
- (72) 発明者: 竹内 翔一 (TAKEUCHI, Shoichi); 〒
2591392 神奈川県秦野市堀山下 1 番地 株式会
社日立製作所 IT プラットフォーム事業本部
内 Kanagawa (JP). 松本 周平 (MATSUMOTO,
Shuhei); 〒2591392 神奈川県秦野市堀山下 1 番地
株式会社日立製作所 IT プラットフォーム事
業本部内 Kanagawa (JP).
- (74) 代理人: 特許業務法人藤央特許事務所 (TOU-OU
PATENT FIRM); 〒1050001 東京都港区虎ノ門一
丁目 1 6 番 4 号アーバン虎ノ門ビル Tokyo (JP).
- (81) 指定国 (表示のない限り、全ての種類の国内保
護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA,
BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN,
CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES,
FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN,
IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR,
LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX,
MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH,
PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL,
SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG,
US, UZ, VC, VN, ZA, ZM, ZW.
- (84) 指定国 (表示のない限り、全ての種類の広域保
護が可能): ARIPO (BW, GH, GM, KE, LR, LS, MW,
MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシ
ア (AM, AZ, BY, KG, KZ, RU, TJ, TM), ヨーロッパ
(AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR,
GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT,

[続葉有]

(54) Title: VIRTUAL COMPUTER SYSTEM AND SCHEDULING METHOD

(54) 発明の名称: 仮想計算機システムおよびスケジューリング方法

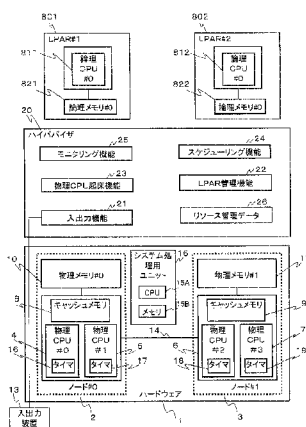


FIG. 1A:
1 Hardware
2 Node #0
3 Node #1
4 Physical CPU #0
5 Physical CPU #1
6 Physical CPU #2
7 Physical CPU #3
8, 9 Cache memory
10 Physical memory #0
11 Physical memory #1
13 Input/output device
15 System processing unit
15A CPU
15B Memory
16, 17, 18, 19 Timer
20 Hypervisor
21 Input/output function
22 LPAR management function
23 Physical CPU wakeup function
24 Scheduling function
25 Monitoring function
26 Resource management data
811, 812 Logical CPU #0
821, 822 Logical memory #0

(57) Abstract: A scheduling method whereby a virtualization unit, which has multiple nodes equipped with cache memories and containing physical CPUs and physical memories, and which operates a virtual computer by generating a logical CPU and logical partitions from the computer resources of the multiple nodes, allocates a physical CPU to the logical CPU. The multiple nodes are connected via an interconnect, and the virtualization unit selects the physical CPU to be allocated to the logical CPU, and measures performance information related to the performance when the physical memory is accessed from the logical CPU. When the performance information satisfies a prescribed threshold value the physical CPU allocated to the logical CPU is selected from the same node as that of the previously allocated physical CPU, and when the performance information does not satisfy the prescribed threshold value the physical CPU allocated to the logical CPU is selected from a different node than the node of the previously allocated physical CPU.

(57) 要約: キャッシュメモリを備えた物理CPUと物理メモリを含むノードを複数有し、複数のノードの計算機資源から論理CPUと論理区画を生成して仮想計算機を稼働させる仮想化部が、論理CPUに物理CPUを割り当てるスケジューリング方法であって、複数のノードは、インターコネクトを介して接続され、仮想化部が、論理CPUに割り当てる物理CPUを選択し、論理CPUから物理メモリへのアクセス性能に関する性能情報を測定し、性能情報が所定の閾値を満たす場合には、論理CPUに割り当てる物理CPUを、前回割り当てた物理CPUと同一ノードから選択し、性能情報が所定の閾値を満たさない場合には、論理CPUに割り当てる物理CPUを、前回割り当てた物理CPUのノードとは異なるノードから選択する。

WO 2014/141419 A1



NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI 添付公開書類:

(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, — 国際調査報告 (条約第 21 条(3))
NE, SN, TD, TG).

明 細 書

発明の名称：仮想計算機システムおよびスケジューリング方法 技術分野

[0001] 本発明は、物理計算機上の物理リソースを論理分割した論理区画（以下：L P A R）を複数生成し、複数の物理C P Uを複数の論理C P U間で時分割により共有する際のスケジューリングに関する。

背景技術

[0002] 近年、1台の物理計算機の性能が向上したため、計算機の集約率を上げて管理コストや消費電力の削減をするべく、物理計算機上に論理的な計算機を複数台構築する方法が一般的となってきた。論理的な計算機の構築方法として、ハイパバイザが物理C P U、物理メモリ、入出力装置などの計算機リソースを分割または共有して論理的な計算機であるL P A R（L o g i c a l P A R t i t i o n）に割り当て、1個以上の論理的な計算機を生成して制御する方法が挙げられる。

[0003] 一つの物理C P Uを複数のL P A Rにて共有させるには、ハイパバイザが物理C P Uをタイムスライスとよばれる単位時間にて時分割し、各々のL P A Rに割り当てられた論理C P Uの実行時間を管理して共有させる必要がある。この制御を一般的にスケジューリングとよぶ。

[0004] 一方で、物理計算機において、複数のC P Uとメモリを一つのノードとよばれる組にまとめ、ある物理C P Uから見て同じノード内のメモリへのアクセスコストが、他のノード内のメモリへのアクセスコストと比べて小さくなるような計算機構成のことをN U M A（N o n - U n i f o r m M e m o r y A c c e s s）構成と呼ぶ。このとき、同じノード内のメモリをローカルメモリ、他のノードのメモリをリモートメモリとよぶ。

[0005] 全ての物理C P Uからのメモリアクセスコストが同じとなるU M A（U n i f o r m M e m o r y A c c e s s）構成では、物理C P Uを多く搭載する場合において物理メモリアクセスが物理C P U間で競合が発生しやすく

なるのに対し、NUMA構成ではノードごとに並列して物理メモリアクセスができるため競合を減らすことができる、というメリットがある。このため、搭載CPU数が増加している近年では、NUMA構成を採用している計算機は少なくない。

[0006] 物理メモリへのアクセスコストを軽減する機構としては、物理CPUに対して物理的に近い位置にメモリ内容を一時保存する物理メモリキャッシュ（以下：キャッシュ）が存在する。物理メモリへアクセスを行なう度にキャッシュにもアクセスしたメモリ内容を保存することで、次回同じメモリ内容へアクセスする際はキャッシュへアクセスすればよくなるため、結果的に物理メモリへのアクセスコストを軽減できるのである。なお、一般的にキャッシュへのアクセス速度はローカルメモリにアクセスするよりも高速にアクセスすることができる。

[0007] 特許文献1の仮想計算機の実装方法では、システム全体がビジーのとき、より優先度の高い処理をLPARの処理能力の低下を抑制しながら物理CPU間で分散して実施する、という技術が開示されている。

先行技術文献

特許文献

[0008] 特許文献1：特開2008-186136号公報

発明の概要

発明が解決しようとする課題

[0009] しかしながら、上記特許文献1では、物理CPUがアイドルであればその物理CPU上で優先度の高い処理を繰り返し実施してしまう。これは、優先度の高い処理をLPARの処理として置き換えて考えても同様なことが言える。

[0010] NUMA構成の計算機の場合、LPARの処理を一つの物理CPUで処理させ続けてしまうと、当該LPARの処理に対してキャッシュ容量以上のメモリアクセスが伴い、かつ、処理を実行する物理CPUに対してリモートメ

メモリへのアクセスとなっていた場合、計算機システム全体がビジーにならない限りリモートメモリへのアクセスを継続し、物理メモリのアクセスコストが高い状態で処理を継続してしまう、という問題があった。

[0011] そこで本発明は、上記問題点に鑑みてなされたもので、物理メモリへのアクセスコストが高い状態が継続するのを抑制することを目的とする。

課題を解決するための手段

[0012] 本発明は、キャッシュメモリを備えた物理CPUと物理メモリを含むノードを複数有し、前記複数のノードの計算機資源を仮想計算機に割り当てる仮想化部と、を備えた仮想計算機システムであって、前記複数のノードは、インターコネクトを介して接続され、前記仮想化部は、前記仮想計算機に割り当てる計算機資源として論理区画と論理CPUを生成する論理区画管理部と、前記論理CPUを実行する前記物理CPUを選択するスケジューリング部と、前記論理CPUから前記物理メモリへのアクセス性能に関する性能情報を測定するモニタリング部と、を備え、前記スケジューリング部は、前記性能情報が所定の閾値を満たす場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記性能情報が所定の閾値を満たさない場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択する。

発明の効果

[0013] 本発明によれば、NUMA構成の計算機のように、複数の物理CPUが複数の物理メモリにアクセスする際に、物理CPUと物理メモリの位置関係に応じてメモリのアクセス性能が異なる物理計算機において、論理CPU数に対して物理CPUの数が十分にある場合には、論理CPUに対する物理メモリのアクセス性能を、1秒程度の所定の観測時間にて観測した際に、最高性能と最低性能の中間程度の性能が少なくとも得られ、さらにキャッシュヒット率が高い際にはキャッシュメモリによる高いメモリリード、ライト性能となるという効果が得られる。

図面の簡単な説明

[0014] [図1A]本発明の第1の実施例を示し、仮想計算機システムの一例を示すブロック図である。

[図1B]本発明の第1の実施例を示し、ハイパバイザが管理するリソース管理データの詳細を示すブロック図である。

[図2]本発明の第1の実施例を示し、L P A Rの設定を保持するL P A R管理テーブルの一例を示す図である。

[図3]本発明の第1の実施例を示し、論理C P Uを稼働させる物理C P Uの情報を保持する物理C P U管理テーブルの初期状態を示す図である。

[図4A]本発明の第1の実施例を示し、C P Uの状態とコードとの対応関係を示すC P U状態テーブルの一例を示す図である。

[図4B]本発明の第1の実施例を示し、C P Uの状態遷移図である。

[図5]本発明の第1の実施例を示し、物理C P Uごとに設定された論理C P Uの情報を保持する固有論理C P U管理テーブルの初期状態を示す図である。

[図6]本発明の第1の実施例を示し、全物理C P Uで稼働させることのできる論理C P Uの情報を保持する共有論理C P U管理テーブルの初期状態を示す図である。

[図7]本発明の第1の実施例を示し、論理C P Uを物理C P Uで稼働させるときに一時的に論理C P Uの情報を保持する一時保管用論理C P U管理テーブルの初期状態を示す図である。

[図8]本発明の第1の実施例を示し、論理C P Uを実行させる物理C P Uを選択する際の計算情報を一時的に保管する一時計算テーブルの初期状態を示す図である。

[図9]本発明の第1の実施例を示し、論理C P Uを実行させる物理C P Uを選択する際の判定用の閾値を保持する閾値管理テーブルの一例を示す図である。

[図10]本発明の第1の実施例を示し、ハイパバイザで行われる閾値の登録処理の一例を示すフローチャートである。

[図11A]本発明の第1の実施例を示し、ハイパバイザで行われるLPARの起動処理の一例を示すフローチャートで、前半部である。

[図11B]本発明の第1の実施例を示し、ハイパバイザで行われるLPARの起動処理の一例を示すフローチャートで、後半部である。

[図12]本発明の第1の実施例を示し、ハイパバイザで行われる物理CPUの起床処理の一例を示すフローチャートである。

[図13]本発明の第1の実施例を示し、図11A、図11BのLPARの起動処理後の物理CPU管理テーブルの一例を示す図である。

[図14]本発明の第1の実施例を示し、図11A、図11BのLPARの起動処理後の共有論理CPU管理テーブルの一例を示す図である。

[図15]本発明の第1の実施例を示し、ハイパバイザで行われるスケジューリング処理の概要を示すフローチャートである。

[図16A]本発明の第1の実施例を示し、スケジューリング処理の詳細のうち、物理CPUディスパッチ開始処理の前半部を示すフローチャートである。

[図16B]本発明の第1の実施例を示し、スケジューリング処理の詳細のうち、物理CPUディスパッチ開始処理の中間部を示すフローチャートである。

[図16C]本発明の第1の実施例を示し、スケジューリング処理の詳細のうち、物理CPUディスパッチ開始処理の後半部を示すフローチャートである。

[図16D]本発明の第1の実施例を示し、スケジューリング処理の詳細のうち、物理CPUディスパッチ開始処理で処理すべき論理CPUが存在しなかったときのフローチャートである。

[図17A]本発明の第1の実施例を示し、スケジューリング処理の際の共有論理CPU管理テーブルと一時保管用論理CPU管理テーブルのエントリの関係を示す図である。

[図17B]本発明の第1の実施例を示し、スケジューリング処理の際の一時計算テーブルを示す図である。

[図18]本発明の第1の実施例を示し、スケジューリング処理における論理CPU実行処理の一例を示すフローチャートである。

[図19]本発明の第1の実施例を示し、ゲストモード終了の要因コードテーブルの一例を示す図である。

[図20A]本発明の第1の実施例を示し、スケジューリング処理のうち、物理CPUディスパッチ終了処理の一例を示すフローチャートの前半部である。

[図20B]本発明の第1の実施例を示し、スケジューリング処理における物理CPUディスパッチ終了処理のフローチャートの中間部である。

[図20C]本発明の第1の実施例を示し、スケジューリング処理における物理CPUディスパッチ終了処理のフローチャートの後半部である。

[図21A]本発明の第1の実施例を示し、図20Aの物理CPUディスパッチ終了処理の処理を実施した際の一時保管用論理CPU管理テーブルを示す図である。

[図21B]本発明の第1の実施例を示し、図20Aの物理CPUディスパッチ終了処理の処理を実施した際の一時計算テーブルを示す図である。

[図21C]本発明の第1の実施例を示し、図20B、図20Cの処理を実施した際の一時保管用論理CPU管理テーブルエントリと共通論理CPU管理テーブルの関係を示す図である。

[図22]本発明の第1の実施例を示し、閾値を閾値管理テーブルに格納する処理の一例を示すフローチャートである。

[図23]本発明の第1の実施例を示し、論理CPUのキャッシュヒット率を測定するためのモニタ有効化処理の一例を示すフローチャートである。

[図24]本発明の第1の実施例を示し、論理CPUのキャッシュヒット率を測定するためのモニタリング処理の一例を示すフローチャートである。

[図25A]本発明の第1の実施例を示し、図23のモニタ有効化処理を実施した際の閾値管理テーブルを示す図である。

[図25B]本発明の第1の実施例を示し、物理CPUディスパッチ終了処理を実施した際の閾値管理テーブルを示す図である。

[図25C]本発明の第1の実施例を示し、図24のモニタリング処理を実施した際の閾値管理テーブルを示す図である。

[図26]本発明の第1の実施例を示し、論理CPUの実行時間とキャッシュヒット率との関係を示すグラフである。

[図27]NUMA構成の計算機のブロック図で、メモリアクセスの一例を示す図である。

[図28]従来例を示し、スケジューリング処理による物理CPU上の処理の一例を示すタイムチャートである。

[図29]本発明の第1の実施例を示し、スケジューリング処理による物理CPU上の処理の一例を示すタイムチャートで、総キャッシュヒット率閾値が0.5の場合である。

[図30]NUMA構成の計算機のブロック図で、メモリアクセスの他の一例を示す図である。

[図31]本発明の第1の実施例を示し、本発明のスケジューリング処理において計算機システムがビジー状態であった際の物理CPU上の処理の一例を示すタイムチャートである。

[図32]本発明の第2の実施例を示し、仮想計算機システムの一例を示すブロック図である。

発明を実施するための形態

[0015] 以下、本発明の一実施形態について添付図面を用いて説明する。

実施例 1

[0016] 図1A、図1Bは、本発明における仮想計算機システムの基本的な構成例を示すブロック図である。

[0017] 1以上の仮想計算機を提供する物理計算機は、物理CPU（#0～#3）4～7と、物理CPU（または物理プロセッサ）ごとに設けたタイマ16～19と、ノードごとに存在する物理キャッシュメモリ8～9と、ノード毎に接続された物理メモリ10～11と、入出力装置13と、システム処理用ユニット15を有する。

[0018] 物理CPU4～5およびタイマ16～17、物理キャッシュメモリ8、物理メモリ10はノード（#0）2に属し、物理CPU6～7およびタイマ1

8～19、物理キャッシュメモリ9、物理メモリ111はノード（#1）3にそれぞれ属しており、ノード間はインターコネクト14で接続されている。

[0019] 当該計算機のメモリの一部にロードされるハイパバイザ（仮想化部）20は、物理CPU4～7、物理メモリ10～11、入出力装置13などの物理的な計算機リソースを分割または共有して論理的な計算機であるL P A R（Logical P A R tition：論理区画）801、802に割り当て、1以上の論理的な計算機（または仮想計算機）を生成して制御する。

[0020] システム処理用ユニット15は、L P A R 801、802に割り当てず、ハイパバイザ20の処理のみを行う物理CPU15Aと物理メモリ15Bを有する。

[0021] ハイパバイザ20は、入出力装置13を介してユーザからの入力を受け付け、またユーザへ情報の出力を行なう入出力機能21と、L P A R 801、802に割り当てる計算機リソースの設定およびL P A R 801、802の生成及び起動を行うL P A R 管理機能22と、休止状態（または停止状態）の物理CPU4～7を起床させる物理CPU起床機能23と、ひとつの物理CPUを複数の論理CPU（論理プロセッサ）で時分割（タイムスライス）で共有させるスケジューリング機能24と、論理CPU811、812（または物理CPU4～7のキャッシュリード回数、キャッシュヒット回数）ごとのキャッシュヒット率を測定するモニタリング機能25と、物理計算機（ハードウェア）1のリソース情報を保持するリソース管理データ26とを含んで構成される。

[0022] また、ハイパバイザ20の上記各機能は、上述のようにシステム処理用ユニット15にて処理される。図1の例では、ハイパバイザ20は、物理計算機のハードウェア1から論理CPU811、812と論理メモリ821、822を生成し、生成したL P A R 801、802へそれぞれのリソースを割り当てる例を示す。

[0023] リソース管理データ26には、図1Bで示すように、L P A R 801、8

０２に割り当てた論理ＣＰＵ８１１、８１２と、論理メモリ８２１、８２２の容量を管理するＬＰＡＲ管理テーブル１００と、物理ＣＰＵ４～７の状態を管理する物理ＣＰＵ管理テーブル２００と、対象の物理ＣＰＵ４～７でのみ実行（または割り当て）可能な論理ＣＰＵの情報を物理ＣＰＵ４～７ごとに保持する固有論理ＣＰＵ管理テーブル３００と、全ての物理ＣＰＵ４～７にて実行させる論理ＣＰＵの情報を保持する共有論理ＣＰＵ管理テーブル４００と、物理ＣＰＵ４～７ごとに保持している一時保管用論理ＣＰＵ管理テーブルエントリ５００と、キャッシュメモリ８、９のヒット率を演算するための一時計算用テーブル６００と、論理ＣＰＵ８１１、８１２を、次のタイムスライス期間ではどの物理ＣＰＵ４～７で実行させるかを決定するための基準値を保持する閾値管理テーブル７００と、物理ＣＰＵ４～７の稼動状態を定義するＣＰＵ状態テーブル４５０と、ＬＰＡＲ８０１、８０２の仮想計算機が、ゲストモードを終了させる要因を保持する要因コードテーブル４６０と、が含まれる。

[0024] ハイパバイザ２０上のＬＰＡＲ数およびＬＰＡＲを構成する論理ＣＰＵ数の最大数はハイパバイザ２０内で定義される最大数によって決定される。当該構成図では、ハイパバイザは２台のＬＰＡＲ８０１、８０２を生成し、それぞれのＬＰＡＲ８０１、８０２には、ハードウェア１から設定した各１個の論理ＣＰＵ８１１、８１２と論理メモリ８２１、８２２がそれぞれ割り当てられた例を示している。

[0025] ハイパバイザ２０のモニタリング機能２５は、論理ＣＰＵ８１１、８１２（または物理ＣＰＵ４～７）毎にキャッシュメモリ８、９のリード回数（以下、キャッシュリード回数）と、ヒット回数（以下、キャッシュヒット回数）を測定するカウンタを備え、論理ＣＰＵ８１１、８１２（または物理ＣＰＵ４～７）ごとのキャッシュヒット率を演算する。

[0026] 本発明の概要は、次のとおりである。本発明では、論理ＣＰＵ８１１（８１２）が同じノード上の物理ＣＰＵ４（５～７）で連続実行された際に、ハイパバイザ２０にて当該論理ＣＰＵ８１１（８１２）の連続実行時間が、所

定のキャッシュヒット率となることを期待する所定時間を経過しているか否かを判定する。所定時間を経過している場合には、ハイパバイザ20は論理CPU811(812)のキャッシュヒット率が閾値よりも高いか否かを判定する。キャッシュヒット率が閾値よりも低い場合は、キャッシュメモリ8、9の恩恵を受けていないとハイパバイザ20は判断し、論理CPU811(812)が連続実行したノードとは別のノード上に休止状態の物理CPUが存在すれば当該物理CPUを起床して当該論理CPUを実行させる。

[0027] 一方、キャッシュヒット率が閾値よりも高い場合は、論理CPU811(812)を連続実行したノードと別のノード上に休止状態の物理CPUが存在しても起床させず、引き続き現在のノードの物理CPU4(または5)にて当該論理CPU811を実行させる。また、キャッシュヒット率は低い、論理CPU811を連続実行させたノードとは別のノード上に休止状態の物理CPUが存在しなかった場合に関しても、現在起床している物理CPU6(または7)にて当該論理CPU(811)を実行させる。

[0028] 物理CPU4(～7)に実行させる論理CPU811(812)は、ハイパバイザ20に設けた論理CPUのキュー(共有論理CPU管理テーブル400)にて保持する。ハイパバイザ20は、キューに入った順で論理CPU811(812)を物理CPU4(～7)にて実行させ、所定のタイムスライス時間が経過して論理CPU811(812)の実行が完了したら、当該論理CPU811を再度論理CPUのキューに入れる。

[0029] 所定のキャッシュヒット率となることを期待する論理CPUの総実行時間の閾値 β と、論理CPUがキャッシュメモリ8、9の恩恵を受けていると判定するためのキャッシュヒット率の閾値 α は、ユーザにより設定できる。なお、前者を総実行時間の閾値 β 、後者をキャッシュヒット率の閾値 α とする。

[0030] 総実行時間の閾値 β は物理メモリ10、11のメモリアクセス性能(レイテンシや転送速度など)の平均値を測定する時間T(後述)よりも小さいものとする。この理由は次の通りである。本発明において論理CPU811(

812) が連続して同じノード上の物理CPU4 (または5) で実行される時間は、総実行時間の閾値 β となり、メモリアクセス性能の平均値を測定する時間Tよりも総実行時間の閾値 β の方が大きいと、時間Tの間に一つのノードの物理CPU4 (または5) で論理CPU811が実行され続けてしまう。これは、上記発明が解決しようとする課題で述べたように、時間Tにおける論理CPU811の平均メモリアクセス性能は、リモートメモリへのアクセス性能、すなわち低い性能 (レイテンシが大) となってしまう可能性があることを意味する。このため、メモリアクセス性能の平均値を測定する時間Tよりも総実行時間の閾値 β は小さい値に設定される。

[0031] 次に、ハイパバイザ20及びハイパバイザ20のスケジューリング機能24等の各機能部はプログラムとしてシステム処理用ユニット15のメモリ15Bにロードされ、システム処理用ユニット15の物理CPU15Aによって実行される。

[0032] システム処理用ユニット15の物理CPU15Aは、各機能部のプログラムに従って動作することによって、所定の機能を実現する機能部として動作する。例えば、物理CPU15Aは、スケジューリングプログラムに従って動作することでスケジューリング機能24として機能する。他のプログラムについても同様である。さらに、システム処理用ユニット15の物理CPU15Aは、各プログラムが実行する複数の処理のそれぞれを実現する機能部としても動作する。計算機及び仮想計算機システムは、これらの機能部を含む装置及びシステムである。

[0033] ハイパバイザ20の各機能を実現するプログラム、テーブル等の情報は、図示しないストレージサブシステムや不揮発性半導体メモリ、ハードディスクドライブ、SSD (Solid State Drive) 等の記憶デバイス、または、ICカード、SDカード、DVD等の計算機読み取り可能な非一時的データ記憶媒体に格納することができる。

[0034] 図2は、LPARの設定を保持するLPAR管理テーブル100の一例を示す図である。図2に示すLPAR管理テーブル100は、LPAR801

、８０２の識別子であるＬＰＡＲ番号１０１と、ＬＰＡＲ８０１、８０２に割り当てる論理ＣＰＵ８１１、８１２の個数を格納する論理ＣＰＵ個数１０２と、論理メモリ８２１、８２２としてＬＰＡＲ８０１、８０２に割り当てる物理メモリ１０、１１の開始位置（アドレス）を格納する割り当て物理メモリ開始位置１０３と、開始位置から論理メモリ８２１、８２２へ割り当てる物理メモリ４～７のサイズを格納する物理メモリサイズ１０４と、を備える。

[0035] ＬＰＡＲ管理テーブル１００は、予め定義されたＬＰＡＲ８０１、８０２の数に応じたエントリを有する。ＬＰＡＲ管理テーブル１００の各エントリは、ユーザによるＬＰＡＲ環境の設定命令を入出力装置１３を介して、ハイパバイザ２０の入出力機能２１が受け付ける。そして、ＬＰＡＲ管理機能２２がＬＰＡＲ番号、論理ＣＰＵ数、メモリ領域の情報を読み込んで、ＬＰＡＲ管理テーブル１００の各エントリを定義する。

[0036] 図３は、論理ＣＰＵ８１１、８１２を稼働させる物理ＣＰＵ４～７の情報を保持する物理ＣＰＵ管理テーブル２００の初期状態を示す図である。

[0037] 図３に示す物理ＣＰＵ管理テーブル２００は、物理ＣＰＵ４～７の識別子である物理ＣＰＵ番号２０１、物理ＣＰＵ４～７の稼働状態を格納する状態２０２、物理ＣＰＵ４～７が属するノード番号を保持する物理ＣＰＵノード番号（２０３）を備える。図３に示す物理ＣＰＵ管理テーブル２００は、ＬＰＡＲ８０１、８０２に時分割で共有可能な物理ＣＰＵ群を保持するため、ＬＰＡＲ８０１、８０２の論理ＣＰＵ８１１、８１２を実行させる物理ＣＰＵの個数分のエントリを有する。

[0038] 図４Ａ、図４Ｂではハイパバイザ２０にて予め定義されている物理ＣＰＵ４～７及び論理ＣＰＵ８１１、８１２の状態とコードの対応テーブルと状態遷移図を示す。図４Ａは、物理ＣＰＵ４～７（論理ＣＰＵ８１１、８１２）の状態とコードとの対応関係を示すＣＰＵ状態テーブル４５０の一例を示す図である。図４Ｂは、物理ＣＰＵ４～７（論理ＣＰＵ８１１、８１２）の状態遷移図である。

[0039] 物理、論理を問わずCPUの状態の初期状態は処理を行っていない状態を示すHALT（図4Aのコード451＝3）である。処理が発生すると、HALTの状態から処理の実行がいつでも可能な状態を示すREADY（図4Aのコード451＝2）に移る。処理を開始するとREADYから処理を実行中の状態を示すRUN（図4Aのコード451＝1）に移る。処理が完了し、処理すべき内容が何もない場合は、RUNからHALTに移り、I/Oなどの待ちが発生した場合等ではRUNからREADYに移る。HALTから直接RUNに移ることや、READYからHALTに移ることはない。

[0040] 物理CPU管理テーブル200は、ハイパバイザ20が起動する際に物理CPU4～7とそのノード番号を認識し、予めハイパバイザ20で定義されていた物理CPU4～7のみ追加される。その際の物理CPU状態は全てHALTとなる。なお、HALTの状態にはスリーピングステートを加えるようにしても良い。

[0041] 図5は、物理CPU4～7ごとに設定された論理CPU811、812の情報を保持する固有論理CPU管理テーブル300の初期状態（エントリが0）を示す図である。

[0042] 図5に示す固有論理CPU管理テーブル300は、エントリ順序を示すエントリ番号301と、当該エントリのLPAR番号302と論理CPU番号303、および論理CPU状態304、当該エントリの論理CPUが前回実行した物理CPUの属していたノードの番号を意味する前回のノード番号305と、前回のノードにて実行していた際の総キャッシュリード回数306と総キャッシュヒット回数307、および総実行時間308と、LPAR801、802が起動してからノード#0の物理CPUで実行した総時間を意味するノード0の物理CPUでの実行時間309と、同様にノード#1に關しての実行時間を意味するノード#1の物理CPUでの実行時間310をひとつのエントリに備える。

[0043] 固有論理CPU管理テーブル300は物理CPU4～7ごとにハイパバイザ20で設定され、スケジューリング機能24にて、対応する物理CPU4

～7へ優先的に割り当てたい論理CPU 811、812のエントリを保持させる。この固有論理CPU管理テーブル300が、上述した本発明の概要における別のノード上の物理CPU 4、5で論理CPU 811を実行させる手法にあたる。

[0044] 図6は、全物理CPU 4～7で稼働させる論理CPU 811、812の情報を保持する共有論理CPU管理テーブル400の初期状態（エントリが0）を示す図である。

[0045] 図6に示す共有論理CPU管理テーブル400は、図5に示した固有論理CPU管理テーブル300と同じ構成のエントリで構成される。すなわち、エントリ番号401、LPAR番号402、論理CPU番号403、論理CPU状態404、前回のノード番号405、総キャッシュリード回数406、総キャッシュヒット回数407、総実行時間408、ノード0の物理CPUでの実行時間409、ノード1の物理CPUでの実行時間410でひとつのエントリを構成する。

[0046] 共有論理CPU管理テーブル400はハイパバイザ20内に一つのみ存在し、スケジューリング機能24にて、いずれの物理CPUで実行させてもよい論理CPUのエントリを保持する。

[0047] 図7は、論理CPU 811、812を物理CPU 4～7で稼働させるときに一時的に論理CPU 811、812の情報を保持する一時保管用論理CPU管理テーブル500の初期状態（エントリが0）を示す図である。図7に示す一時保管用論理CPU管理テーブルエントリ500に関しても、図5に示した固有論理CPU管理テーブル300からエントリ番号を削除した他は、同じ構成のエントリで構成される。すなわち、LPAR番号501、論理CPU番号502、論理CPU状態503、前回のノード番号504、総キャッシュリード回数505、総キャッシュヒット回数506、総実行時間507、ノード0の物理CPUでの実行時間508、ノード1の物理CPUでの実行時間509でひとつのエントリを構成する。一時保管用論理CPU管理テーブルエントリ500は、固有論理CPU管理テーブル300、共有論

理CPU管理テーブル400と異なり、エントリ番号は備えない。

[0048] 一時保管用論理CPU管理テーブルエントリ500は物理CPU4～7ごとに設定され、スケジューリング機能24は、制御対象としている物理CPU4～7で実行している論理CPU811、812のエントリを保持させる。

[0049] 図1の構成では、ノードが二つ（#0、#1）存在する構成であるため、図5のノード（#0、#1）の物理CPUでの実行時間309～310と、図6のノード（#0、#1）の物理CPUでの実行時間409～410と、図7のノード（#0、#1）の物理CPUでの実行時間508～509はそれぞれ二列（2エントリ）ずつ備えることになる。ノードの数が増えると、この列の数は増加する。

[0050] 固有論理CPU管理テーブル300と、共有論理CPU管理テーブル400の合計エントリ数の最大値は、起動しているLPAR801、802に割り当てられた論理CPU811、812の数となる。

[0051] 一時保管用論理CPU管理テーブルエントリ500は、当該物理CPU4～7にて実行中の論理CPU811、812の情報を保持するため、エントリ数は最大でも一つとなる。また、固有論理CPU管理テーブル300、共有論理CPU管理テーブル400、一時保管用論理CPU管理テーブルエントリ500の初期値は、論理CPUが一つも実行されていない状態であるため、図5、図6、図7のエントリを削除した状態、すなわち空のテーブルとなる。

[0052] 図8は、論理CPU811、812を実行させる物理CPU4～7を選択する際の計算情報を一時的に保管する一時計算テーブルの初期状態を示す図である。図8に示す一時計算用テーブル600は、総キャッシュヒット率601、論理CPUを物理CPU上で実行させた際のキャッシュリード回数602およびキャッシュヒット回数603、論理CPUの物理CPU上での実行開始時刻604と実行終了時刻605を保存する領域を備える。

[0053] 図9は、論理CPU811、812を実行させる物理CPU4～7を選択

する際の判定用の閾値を保持する閾値管理テーブル 700 の一例を示す図である。図 9 に示す閾値管理テーブル 700 は、L P A R 番号 701、論理 C P U 番号 702、当該エントリの論理 C P U を次に実行させる物理 C P U の選択の際に必要となる、総キャッシュヒット率の閾値を意味する閾値 α 703 と、総実行時間の閾値を意味する閾値 β 704 と、現在の稼動状況を監視するか否かを保持するモニタ有効フラグ 705 と、モニタした内容を書き込んだか否かを保持する書き込み完了フラグ 706 と、モニタした内容を書き込む総キャッシュヒット率 707 および総実行時間 708 からひとつのエントリを構成する。

[0054] 閾値管理テーブル 700 は、L P A R 管理機能 22 が L P A R 管理テーブル 100 のエントリを定義した後に、図 9 のエントリを定義する。図 9 に示した閾値管理テーブル 700 では、L P A R # 1 (801)、L P A R # 2 (802) が定義され、ハイパバイザ 20 内で閾値 α が “0.5”、閾値 β が “2000” と予め定義されているときにエントリが作成された直後の状態を示している。なお、閾値管理テーブル 700 において、L P A R 801、802 が定義されていないときの初期値はエントリが存在しない、すなわち空のテーブルとなる。

[0055] なお、上述したとおり、閾値 β はメモリアクセス性能の平均値を測定する時間 T (例えば、1 秒間) よりも小さいものとする。

[0056] 図 10 は、ハイパバイザ 20 で行われる閾値の登録処理の一例を示すフローチャートである。図 10 は、入出力装置 13 を介してハイパバイザ 20 の入出力機能 21 から L P A R 環境の設定命令を受け取り、L P A R 管理機能 22 が L P A R 管理テーブル 100 のエントリを追加した後に引き続き行なう処理であり、L P A R 管理機能 22 が当該 L P A R 801、802 の論理 C P U 811、812 の閾値を閾値管理テーブル 700 に登録する処理を示す。以下、当該処理の実施直前に L P A R 管理テーブル 100 に追加したエントリの L P A R 801 を当該 L P A R として、その処理について説明する。

- [0057] LPAR管理機能22は、論理CPU番号を0として、当該LPAR801の論理CPU811の数に応じてステップ903を実行し、論理CPU番号に1を加算する、というループを繰り返す(902~904)。
- [0058] ステップ903では、LPAR管理機能22が、閾値管理テーブル700の最後へ、LPAR番号701を当該LPAR番号、論理CPU番号702を現在の論理CPU番号、閾値 α 703および閾値 β 704を、ハイパバイザ20で予め定義されている値に設定し、他のフィールド705~708を“0”に設定したエントリを挿入する。
- [0059] 本フローチャートでは、LPAR管理機能22が、当該LPAR801の論理CPU811の数に応じてエントリを作成し、閾値管理テーブル700の最後へ追加している。なお、当該LPAR801の論理CPU数を減少、および当該LPAR801を削除する場合には、閾値管理テーブル700のLPAR番号701と論理CPU番号702をインデックスとして検索し、対象の論理CPUのエントリを削除すればよい。
- [0060] また、当該LPAR801の論理CPU数を増加させる場合には、閾値管理テーブル700のLPAR番号701と論理CPU番号702をインデックスとして検索し、その中で最も大きい論理CPU番号のエントリの次に、ステップ903と同様の手順でエントリを追加すればよい。
- [0061] 図11Aは、ハイパバイザ20で行われるLPAR801、802の起動処理の一例を示すフローチャートの前半部で、図11Bは後半部を示す。
- [0062] 図11A、図11Bでは、入出力装置13を介してハイパバイザ20の入出力機能21からLPAR801、802の起動命令を受け取り、LPAR管理機能22がLPAR801、802を起動させる際の処理を示す。以下、起動させるLPAR801を当該LPAR801としてその処理を示す。この処理は、入出力装置13から指令を受け付けたときにハイパバイザ20が実行する。
- [0063] ハイパバイザ20のLPAR管理機能22は、LPAR801(または802)に割り当てる論理CPU数とメモリ量をLPAR管理テーブル100

から読み出す（932）。L P A R管理機能22は、共有論理C P U管理テーブル400へ新たなエントリを挿入し、エントリ番号401として最も大きいエントリ番号に1を加算した値を設定し、L P A R番号402として対象L P A R番号を設定し、論理C P U番号403として0を設定し、論理C P U状態404としてR E A D Yを設定し、前回のノード番号405として未定義を意味する“−1”を設定し、他のフィールド406〜410は0を設定する（933）。ただし、共有論理C P U管理テーブル400が空だった場合、挿入するエントリのエントリ番号401は“1”とする。

[0064] 次に、L P A R管理機能22は、ステップ934からステップ936のループ処理を論理C P Uの数まで繰り返す。

[0065] まず、L P A R管理機能22は、論理C P U番号を1とし、ステップ935の処理を行ない、論理C P U番号に1を加算する、という処理を当該L P A R 801の論理C P U数分繰り返す（934〜936）。なお、ステップ935はステップ933において、論理C P U状態404を“H A L T”に設定し、論理C P U番号403を1回のループにて1ずつ増加させている点異なる処理となっている。

[0066] L P A R管理機能22は、物理C P U管理テーブル200にて、物理C P U状態202がH A L T状態のエントリを、物理C P U番号の小さいものから検索する（図11Bのステップ937、938、940）。H A L T状態のエントリが検出した場合、L P A R管理機能22は、当該物理C P Uを選択して起床させる（939）。

[0067] 上記フローチャートの処理によって、当該L P A R 108に割り当てられた論理C P U 811が共有論理C P U管理テーブル400に登録される。そして、登録された論理C P Uはスケジューリング機能24によって物理C P U 4〜7が割り当てられ、実行を開始する。

[0068] 図12は、ハイパバイザ20で行われる物理C P U 4〜7の起床処理の一例を示すフローチャートである。図12では、ハイパバイザ20の物理C P U起床機能23が、H A L T状態の物理C P U 4〜7の起床を行なう。以下

、起床させる物理CPU 4～7を当該物理CPU 4として、処理の一例を示す。

[0069] ハイパバイザ20の物理CPU起床機能23は、物理CPU管理テーブル200の当該物理CPU 4の物理CPU番号201の状態202を、図4AのCPU状態テーブル450のHALTからREADYに対応するコード“2”に更新する(952)。その後、物理CPU起床機能23は、物理CPU管理テーブル200の当該物理CPU 4の状態202を、RUNに対応するコード“1”に更新する(953)。

[0070] ハイパバイザ20は物理CPU起床機能23が起床した物理CPU 4を対象として、スケジューリング機能24にてスケジューリング処理を開始する(954)。

[0071] 図2、3、図5～図9の状態にてL PAR # 1 (801)を起動した際のテーブルの変化を図13、図14に示す。

[0072] 図13は、L PAR 801の起動処理後の物理CPU管理テーブル200の一例を示す図である。図14は、L PAR 801の起動処理後の共有論理CPU管理テーブル400の一例を示す図である。

[0073] L PAR # 1 (801)の論理CPU (811)の数は一つであり、図6では共有論理CPU管理テーブル400が空であるため、図11Aのステップ933の処理により共有論理CPU管理テーブル400に、エントリ番号411が“1”のエントリが新たに挿入される。

[0074] その後、図3の物理CPU管理テーブル200の物理CPU状態202は、全てのエントリがHALT状態を示す“3”となっているため、図11Bのステップ937～940の処理により最も小さい物理CPU番号403を持つ物理CPU # 0 (4)を起床させる。図12に示した物理CPU起床処理にて、物理CPU管理テーブル200の物理CPU番号211が“0”のエントリにおける物理CPU状態212は、HALT状態の“3”からREADY状態の“2”を経由して、RUN状態の“1”となる。

[0075] なお、L PAR管理テーブル100、固有論理CPU管理テーブル300

、一時保管用論理CPU管理テーブルエントリ500、一時計算用テーブル600、閾値管理テーブル700に変更はない。

[0076] 図15は、ハイパバイザ20で行われるスケジューリング処理の概要を示すフローチャートである。

[0077] この処理は、大きく分けて、物理CPU4～7にどの論理CPU811、812を実行させるかを選択する物理CPUディスパッチ開始処理（972）と、論理CPU811、812を物理CPU4～7上で実行させる論理CPU実行処理（973）と、時分割の処理が完了した論理CPU811、812を、次にどの物理CPU4～7で実行させるかを選択する物理CPUディスパッチ終了処理（974）、の三つで構成される。ハイパバイザ20のスケジューリング機能24は、上記ステップ972～974の処理を繰り返して実行する。

[0078] なお、スケジューリング機能24は、一つの物理CPU4～7に対して実施され、物理CPUが複数個存在するときは、複数のスケジューリング処理が並列して実行される。以下、スケジューリング機能24がスケジュールの対象とする物理CPU4～7を当該物理CPU4として説明する。

[0079] 図16A、図16B、図16C、図16Dは物理CPUディスパッチ開始処理（972）の処理の一例を示すフローチャートを示す。なお、図16Aは、物理CPUディスパッチ開始処理の前半部を示し、図16Bは中間部を示し、図16Cは後半部を示し、図16Dは、物理CPUディスパッチ開始処理で処理すべき論理CPUが存在しなかったときのフローチャートである。

[0080] 物理CPUディスパッチ開始処理（972）は図16Aの処理1001から開始する。スケジューリング機能24は、当該物理CPU4に対応する固有論理CPU管理テーブル300において、エントリ番号が小さいものから順に論理CPU状態304がREADYを示す“2”であるエントリを検索する（1002～1004）。スケジューリング機能24は、論理CPU状態304がREADYを示す“2”であるエントリが存在すれば処理100

5に進む。

[0081] 一方、READYを示す“2”の論理CPU状態304のエントリが固有論理CPU管理テーブル300に存在しない場合は、図16Bのステップ1007へ進む。

[0082] まず、ステップ1002～1004の検索にて、論理CPU状態304がREADYを示す“2”となっているエントリが固有論理CPU管理テーブル300に存在するときを説明する。

[0083] スケジューリング機能24は、READYとなっている論理CPU状態304のエントリにおいて、エントリ番号301以外を一時保管用論理CPU管理テーブルエントリ500にコピーし、固有論理CPU管理テーブル300から当該エントリを除去する（1005）。

[0084] スケジューリング機能24は、固有論理CPU管理テーブル300において各々のエントリが持つエントリ番号301を一ずつ減らし（1006）、ステップ1012、すなわち図16Cのステップ1014に進む。

[0085] 次に、ステップ1002～1004の検索にて論理CPU状態304がREADYを示す“2”のエントリが固有論理CPU管理テーブル300に存在しないときを説明する。

[0086] スケジューリング機能24は、共有論理CPU管理テーブル400において、エントリ番号401が小さいものから順に論理CPU状態404の値がREADYを示す“2”であるエントリを検索し（1007～1009）、ステップ1008では、論理CPU状態404の値がREADYを示す“2”であるエントリが存在すればステップ1010に進む。

[0087] READYとなっている論理CPU状態404のエントリが共有論理CPU管理テーブル400にも存在しない場合は、ステップ1013、すなわち図16Dのステップ1022に進む。

[0088] READYとなっている論理CPU状態404のエントリが共有論理CPU管理テーブル400に存在する場合は、上記ステップ1005と同様である。すなわち、スケジューリング機能24は、共有論理CPU管理テーブル

400から当該エントリのエントリ番号401以外を一時保管用論理CPU管理テーブルエントリ500にコピーし、共有論理CPU管理テーブル400から当該エントリを除去する(1010)。

[0089] その後、スケジューリング機能24は、共有論理CPU管理テーブル400において各々のエントリが持つエントリ番号401を一ずつ減らし(1011)、ステップ1012、すなわち図16Cのステップ1014に移る。

[0090] 図16Cのステップ1014以降の処理では、当該物理CPUの属するノード番号が一時保管用論理CPU管理テーブルエントリ500の、前回のノード番号504と一致するか否かを判定し(1015)、一致しなかった場合はステップ1016に進み、一致した場合はステップ1017に進む。

[0091] スケジューリング機能24は、一時保管用論理CPU管理テーブルエントリ500の前回のノード番号504に、当該物理CPU4が属するノード番号を書き込む処理と、総キャッシュリード回数505と、総キャッシュヒット回数506と、総実行時間507に“0”をそれぞれ書き込む処理を行う(1016)。

[0092] 次に、スケジューリング機能24は、一時計算用テーブル600の実行開始時刻604に現時刻と、一時保管用論理CPU管理テーブルエントリ500の論理CPU状態503にRUNをそれぞれ書き込む(1017)。

[0093] スケジューリング機能24は、モニタリング機能25が備えるカウンタをリセットする。スケジューリング機能24は、モニタリング機能25に対して起動対象の論理CPU811、812によるキャッシュリード回数とキャッシュヒット回数のカウンタをリセットし、これらのカウントを開始させる(1018)。

[0094] ハイパバイザ20で予め定義されているタイムスライスの時間(または周期)にて当該物理CPU4~7のタイマ16~19をセットする(1019)。このとき、タイムスライスの時間を経過した物理CPU4~7は、タイマ16~19によってゲストモードが終了するタイマ割り込みが発生する。なお、タイマ16~19によるタイマ割り込みにより、ホストモードに切り

替わり、ハイパバイザ20に制御が移る。

[0095] スケジューリング機能24は、ハイパバイザ20が介在しないゲストモードに当該物理CPU4〜7を切り替え、論理CPU811、812を起動し（1020）、物理CPUディスパッチ開始処理（972）は完了となる（1021）。

[0096] 図16Dの処理1022以降の処理では、スケジューリング機能24が、物理CPU管理テーブル200のうち当該物理CPU4のエントリの物理CPU状態202にHALTに対応する“3”を書き込む（1023）。

[0097] スケジューリング機能24は、当該物理CPUを休止状態にし（1024）、スケジューリングの処理終了となる（1025）。

[0098] 図16A、図16B、図16C、図16Dで示したフローチャートでは、以下の処理を行なう。

[0099] スケジューリング機能24は、固有論理CPU管理テーブル400内のエントリ番号401が小さいエントリの論理CPU811（812）を優先的に選択して物理CPU4〜7で実行させる論理CPUとする。

[0100] スケジューリング機能24は、固有論理CPU管理テーブル400内にREADYに対応する“2”の論理CPU状態404のエントリが存在しないときには、共有論理CPU管理テーブルエントリ500から同様にREADYに対応する“2”の論理CPU状態503を検索して論理CPU811（812）を選択し、それでも存在しないときには当該物理CPUにて処理する仕事が存在しないため、当該物理CPUを休止状態とする。

[0101] 物理CPUディスパッチ開始処理（972）にて上記の処理を行なうことで、スケジューリング処理はREADYとなっている論理CPU状態304の論理CPU811（812）のエントリを、当該物理CPU4の固有論理CPU管理テーブル300から優先的に選択する。そして、スケジューリング機能24は、固有論理CPU管理テーブル300に論理CPU811のエントリを新たに挿入することで、当該論理CPU811を当該物理CPU4で必ず実行させることができる。

- [0102] また、固有論理CPU管理テーブル300に実行させる論理CPUが存在しない場合は、スケジューリング処理は全物理CPU4～7にて実行可能な論理CPU811（812）のエントリを保有する共有論理CPU管理テーブル400からエントリを取得するため、READY状態の論理CPUをいずれかの物理CPU4～7にて必ず実行させることを保証できる。
- [0103] ここで処理の一例として、図2、図5、図7～14、図13、図14の状態にて図12のフローチャートのステップ954にて物理CPU#0（4）を対象としてスケジューリング機能24のスケジューラ処理を開始したとき、図16A、図16B、図16C、図16Dのフローチャートの処理の結果を図17A、図17Bに示し、このようになる理由を以下に示す。
- [0104] 図13、図14の状態では、固有論理CPU管理テーブル300にはエントリがなく、共有論理CPU管理テーブル400にエントリが1つあるため、ステップ1007～1009の処理により共有論理CPU管理テーブル400にてREADY状態の論理CPU状態のうち、最も小さいエントリ番号411のエントリにおいてエントリ番号以外412～420を、一時保管用論理CPU管理テーブルエントリ500にコピーしてから、共有論理CPU管理テーブル400の当該エントリを削除する。
- [0105] その後、共有論理CPU管理テーブル400はエントリが空になるため、図16Bのステップ1010の処理は、今回は行なわれない。
- [0106] スケジューリング機能24が、一時保管用論理CPU管理テーブルエントリ500にコピーした論理CPU#0のエントリは、ノード#0の物理CPU4、5上で初めて実行するため、図16Cのステップ1015～1016により前回のノード番号504を“0”に変更して、総キャッシュリード回数505と、総キャッシュヒット回数506と、総実行時間507を、それぞれ“0”で初期化する。
- [0107] そして、ステップ1017にて一時計算用テーブル600の実行開始時刻604に現在時刻を書き込み、一時保管用論理CPU管理テーブルエントリ500の論理CPU状態503をRUN状態に対応する値“1”に更新し、

ステップ１０１８～１０２０の処理により物理ＣＰＵ＃０（４）をゲストモードに切り替える。なお、ステップ１０１７では、図１７Ｂで示すように、処理を開始した時刻６０４を２００としている。なお、物理ＣＰＵ４～７は、仮想化支援機能を備えており、ハイパバイザ２０を実行するホストモードと、ゲストＯＳを実行するゲストモードとを備える。

[0108] 図１８は、図１５に示した論理ＣＰＵ実行処理（９７３）の処理の一例を示すフローチャートを示す。本処理の内容は、物理ＣＰＵ＃０（４）のゲストモードの終了要因が発生するまで対象の物理ＣＰＵ＃０（４）をゲストモードにて実行させ、モニタリング機能２５にて実行中の論理ＣＰＵ８１１のキャッシュリード回数と、キャッシュヒット回数をそれぞれカウントする処理を行なう（１０５２）。

[0109] ゲストモード終了要因のコードと要因、その後の論理ＣＰＵ状態は、ハイパバイザ内部で予め定められており、図１９に例を示す論理ＣＰＵ終了要因コードと論理ＣＰＵ状態対応表にて識別する。

[0110] 図１９は、ゲストモード終了の要因コードテーブル４６０の一例を示す図である。要因コードテーブル４６０は、ゲストモードが終了する際の命令を格納する要因コード４６１と、要因の名称を格納する要因４６２と、要因コードを実行した後の論理ＣＰＵの状態を格納する論理ＣＰＵ状態４６３からひとつのエントリが構成される。

[0111] 図２０Ａ、図２０Ｂ、図２０Ｃでは、図１５の物理ＣＰＵディスパッチ終了処理（９７４）で行われる処理の一例を示すフローチャートである。論理ＣＰＵ８１１がゲストモードを終了すると、物理ＣＰＵディスパッチ終了処理（９７４）は図２０Ａのステップ１０６１から開始する。図２０Ａは、スケジューリング処理の物理ＣＰＵディスパッチ終了処理の一例を示すフローチャートの前半部である。同様に、図２０Ｂは物理ＣＰＵディスパッチ終了処理の中間部を示し、図２０Ｃは、後半部を示す。

[0112] スケジューリング機能２４は、要因コードテーブル４６０を参照して、論理ＣＰＵ８１１のゲストモードの終了要因に従って、図１９の要因コードテ

ーブル460の論理CPU状態を決定し、一時保管用論理CPU管理テーブルエントリ500の論理CPU状態503に書き込む。そして、スケジューリング機能24は、当該物理CPU#0(4)にセットされていたタイマ16を停止させる(1062)。この終了要因による論理CPU状態の決定は、スケジューリング機能24が、上述した図19の要因コードテーブル460を参照して実施する。

[0113] 一時保管用論理CPU管理テーブルエントリ500の論理CPU状態503が、RUNに対応する“1”となっている場合(1063のYES)は、ハイパバイザ20の要因コードテーブル460で予め定義されたゲストモード処理要因に対応した処理を行ない(1064)、一時保管用論理CPU管理テーブルエントリ500の論理CPU状態503にREADYに対応する“2”を書き込む(1065)。なお、ステップ1064には論理CPU811がタイムスライスの時間を使い切った際の処理も含まれ、この場合は物理CPU#0(4)のタイマ割り込みに対応した処理となる。

[0114] 次に、スケジューリング機能24は、モニタリング機能25から論理CPU811のキャッシュリード回数とキャッシュヒット回数をそれぞれ読み込んで、一時計算用テーブル600の該当フィールド602、603に書き込む。また、スケジューリング機能24は、一時保管用論理CPU管理テーブルエントリ500の総キャッシュリード回数505、総キャッシュヒット回数506に、上記読み込んだキャッシュリード回数とキャッシュヒット回数をそれぞれ加算する(1066)。

[0115] スケジューリング機能24は、一時計算用テーブル600の実行終了時刻605に現在時刻を書き込み、実行終了時刻605と実行開始時刻604の差分を算出する。そして、スケジューリング機能24は、一時保管用論理CPU管理テーブルエントリ500において、論理CPU811を実行した物理CPU#0の属するノードに対応する実行時間508または509に、上記算出した差分を加算する(1067)。また、スケジューリング機能24は、閾値管理テーブル700の総実行時間708にも上記算出した差分を加

算する（１０６７）。

[0116] スケジューリング機能２４は、一時保管用論理ＣＰＵ管理テーブルエントリ５００の総キャッシュヒット回数５０６を総キャッシュリード回数５０５で除算し、結果を一時計算用テーブル６００の総キャッシュヒット率６０１に書き込む（１０６８）。

[0117] 次に、スケジューリング機能２４は、閾値管理テーブル７００から、一時保管用論理ＣＰＵ管理テーブルエントリ５００のＬＰＡＲ番号５０１と論理ＣＰＵ番号５０２が一致するエントリを取得する（図２０Ｂの１０７１）。以下、閾値管理テーブル７００から取得した本エントリを閾値管理テーブル７００の対象エントリとする。

[0118] スケジューリング機能２４は、閾値管理テーブル７００の対象エントリのモニタ有効フラグ７０５が“１”であるか否かを判定する（１０７２）。モニタ有効フラグ７０５が１であればステップ１０７３へ進み、そうでなければステップ１０７４へ進む。

[0119] スケジューリング機能２４は、閾値管理テーブル７００の対象エントリにおいて、総キャッシュヒット率７０７および総実行時間７０８に一時計算用テーブル６００の総キャッシュヒット率６０１および一時保管用論理ＣＰＵ管理テーブルエントリ５００の総実行時間５０７を書き込み、書き込み完了フラグ７０６に“１”を書き込み、モニタ有効フラグ７０５に“０”をそれぞれ書き込む（１０７３）。

[0120] スケジューリング機能２４は、一時保管用論理ＣＰＵ管理テーブルエントリ５００の総実行時間５０７が閾値管理テーブル７００の当該エントリの閾値 β ７０４より大きいかな否かを判定する（１０７４）。総実行時間５０７が閾値 β ７０４より大きい場合にはステップ１０７５へ進み、総実行時間５０７が閾値 β ７０４以下の場合にはステップ１０７８へ進む。

[0121] ステップ１０７５では、スケジューリング機能２４が、一時計算用テーブル６００の総キャッシュヒット率６０１が閾値管理テーブル７００の対象エントリの閾値 α ７０３より小さいかな否かを判定する。総キャッシュヒット率

601が閾値 α 703より小さい場合にはステップ1076へ進み、総キャッシュヒット率601が閾値 α 703以上であればステップ1078へ進む。

[0122] ステップ1076では、スケジューリング機能24が、物理CPU管理テーブル200において、当該物理CPUが属するノードとは別のノードにHALT状態の物理CPUが存在するか否かを判定する。当該物理CPUが属するノード以外でHALT状態の物理CPUが存在する場合は、ステップ1077へ進み、存在しない場合にはステップ1078へ進む。

[0123] 上記複数の判定により、一時保管用論理CPU管理テーブルエントリ500の総実行時間507が閾値 β 704より大きく、かつ、総キャッシュヒット率601が閾値 α 703より小さく、かつ、当該物理CPUが属するノード以外でHALT状態の物理CPUが存在する場合には、ステップ1077から図20Cのステップ1079へ進み、ステップ1074～1076の判定のいずれか一つでもNOの場合はステップ1078から図20Cのステップ1083へ進む。

[0124] ステップ1079はステップ1080に進み、ステップ1080～1082では下記処理を行なう。

[0125] スケジューリング機能24は、別のノードにてHALT状態で、かつ一時保管用論理CPU管理テーブルエントリ500にて実行時間（508または509）が最も短いノードの物理CPUを選択する（1080）。

[0126] スケジューリング機能24は、上記ステップ1080で選択した物理CPUに対応する固有論理CPU管理テーブル300へ、最も大きいエントリ番号301に“1”を加算したものを新たなエントリ番号301として一時保管用論理CPU管理テーブルエントリ500に挿入する（1081）。ただし、固有論理CPU管理テーブル300のエントリが空だった場合、挿入するエントリのエントリ番号301は“1”とする。

[0127] スケジューリング機能24は、ステップ1080で選択した物理CPUを、ハイパバイザ20の物理CPU起床機能23によって起床させる（108

2)。

- [0128] 一方、ステップ1074～1076の判定のいずれか一つでも“NO”の場合のステップ1083は処理1084に進む。ステップ1084では当該物理CPUに対応する共有論理CPU管理テーブル400へ、最も大きいエントリ番号401に1加算したものを新たなエントリ番号401として一時保管用論理CPU管理テーブルエントリ500を挿入する(1084)。ただし、共有論理CPU管理テーブル400のエントリが空であった場合、挿入するエントリのエントリ番号401は“1”とする。
- [0129] 上記ステップ1082またはステップ1084の処理完了を以って、物理CPUディスパッチ終了処理の終了となる(1085)。
- [0130] 図20A、図20B、図20Cの上記フローチャートでは、下記の処理を行なっている。
- [0131] 論理CPUのゲストモードが終了した後、論理CPU状態503によってはハイパバイザ20にて後処理を行なった後に、スケジューリング機能24が、モニタリング機能25から論理CPUを実行した期間中のキャッシュリード回数とキャッシュヒット回数を取得し、統計情報である総キャッシュヒット率707と総実行時間708を演算して更新する。
- [0132] スケジューリング機能24は、同じノード内の物理CPUで、総実行時間708の閾値 β 704よりも長く当該論理CPUを連続実行していた場合に、総キャッシュヒット率707が閾値 α と比較することで、キャッシュメモリ8、9を有効に活用しているかを判定する。
- [0133] 閾値 α よりも総キャッシュヒット率707が小さく、HALT状態の物理CPUが別のノード内に存在する場合は、別のノード内の物理CPUで当該論理CPU811を実行させるべく、固有論理CPU管理テーブル300に実行させた論理CPUのエントリを追加する。これにより、論理CPUを実行する物理CPUをノード間で転々と移動させることができ、閾値 β よりも長い時間にて論理CPU811のメモリアクセス性能を測定した際に、論理CPU811からみた物理メモリ10、11へのアクセスコストにばらつき

を生むことかできる。

[0134] すなわち、論理CPU 811で実行するプログラムがアクセスするデータなどが、物理CPU 4と同じノードの物理メモリ 10に格納されている場合はローカルアクセスとなり、異なるノードの物理メモリ 11に格納されている場合はリモートアクセスとなり、アクセスコスト、換言すればレイテンシが異なる。

[0135] 閾値 α よりも総キャッシュヒット率707が大きい場合は、論理CPU 811がキャッシュメモリ 8の恩恵を受けていることから同じノードの物理CPU 4、5にて引き続き論理CPUを実行してもよい。このため、共有論理CPU管理テーブル400へ実行させた論理CPU 811のエントリを追加する。

[0136] このとき、別ノードの物理CPU 6、7にHALT状態でないものが存在する場合には、別ノードの物理CPU 6、7を対象としたスケジューリング処理にて共有論理CPU管理テーブル400から当該論理CPU 811のエントリを抽出し、当該論理CPU 811を実行させることもありうる。しかしながらこのような場合は、後述するように仮想計算機システム全体がビジー状態の際のランダム性と同じく、閾値 β よりも長い時間Tにてメモリアクセス性能を測定した際に、論理CPU 811からみた物理メモリ 10、11へのアクセスコストにばらつきを生むことができる。

[0137] HALT状態の物理CPUが別のノード内に存在しない場合は、仮想計算機システム全体がビジー状態であり、後述のとおり、当該論理CPU 811を割り当てる物理CPU 4～7の割り当て先にランダム性が働く。

[0138] この特徴を利用すべく、共有論理CPU管理テーブル400に実行させた論理CPU 811のエントリを追加することで、物理CPUディスパッチ開始処理(972)の際に選ばれる論理CPU 811、812の選択がランダムとなる。そして、閾値 β よりも長い時間Tにて論理CPU 811、812のメモリアクセス性能を測定した際に、論理CPUからみた物理メモリ 10、11へのアクセスコストにばらつきを生むことができるのである。

[0139] 図17A、図17Bとなった状態から引き続き、図20A、図20B、図20Cの処理を実施した結果を図21A、図21B、図21Cに示す。このとき、論理CPU811の実行開始時刻604は、上述のように“200”、実行終了時刻が“1000”、論理CPU811の終了要因がタイマ割り込み、論理CPU811を実行した期間の当該物理CPU4のキャッシュリード回数が“1000”、キャッシュヒット回数が“300”だったものとする。図21A～図21Cは、図20Aのステップ1061～1068を実施した結果である。図21Aは、図20Aの物理CPUディスパッチ終了処理の処理を実施した際の一時保管用論理CPU管理テーブルエントリ500を示す図である。図21Bは、図20Aの物理CPUディスパッチ終了処理の処理を実施した際の一時計算用テーブル600を示す図である。図21Cは、図20B、図20Cの処理を実施した際の一時保管用論理CPU管理テーブルエントリ500と共通論理CPU管理テーブル400の関係を示す図である。

[0140] 論理CPU811の終了要因がタイマ割り込みであるため、図20Aのステップ1062では、図21Aで示すように一時保管用論理CPU管理テーブルエントリ500の論理CPU状態503は、スケジューリング機能24によって図21Aで示すようにRUNからREADYを示す値“2”に変更される。

[0141] 論理CPUを実行した期間の当該物理CPUのキャッシュリード回数が“1000”、キャッシュヒット回数が“300”だったことから、スケジューリング機能24は、図20Aのステップ1066にて、図21Bに示す一時計算用テーブル600のキャッシュリード回数602に“1000”を書き込み、キャッシュヒット回数603に“300”を書き込む。また、スケジューリング機能24は、図21Aの一時保管用論理CPU管理テーブルエントリ500の総キャッシュリード回数505に“1000”を加算し、総キャッシュヒット回数506に“300”を加算する。

[0142] 論理CPUの実行終了時刻が“1000”であり、当該物理CPUがノー

ド#0に属していたことから、図20Aのステップ1067にて、スケジューリング機能24は、図21Bの一時計算用テーブル600の実行終了時刻605に“1000”を書き込み、実行終了時刻605と実行開始時刻604との差分800を演算し、図21Aの一時保管用論理CPU管理テーブルエントリ500の総実行時間507と、ノード0の物理CPU4での実行時間508にそれぞれ差分の800ずつ加算する。

[0143] 次に、図20Aのステップ1068にて、図21Aで示した一時保管用論理CPU管理テーブルエントリ500の総キャッシュヒット回数506である“300”を総キャッシュリード回数505である“1000”で除算し、その結果の“0.3”を図21Bに示す一時計算用テーブル600の総キャッシュヒット率601に書き込む。

[0144] 図21Cは、図20B、図20Cの処理を実施した結果、一時保管用論理CPU管理テーブルエントリ500と共通論理CPU管理テーブル400の関係を示す図である。

[0145] LPAR#1(801)の論理CPU#0(811)を示す閾値管理テーブル700のエントリでは、図9で示したようにモニタ有効フラグ705が“0”となっているため、図20Bのステップ1073は実施されず、閾値管理テーブル700は更新されない。

[0146] LPAR#1(801)の論理CPU#0(811)を示す閾値管理テーブル700のエントリでは、閾値 β 704が“2000”と、図21Aの一時保管用論理CPU管理テーブルエントリ500の総実行時間507の“800”よりも大きいため、図20Bのステップ1074から図21Cのステップ1084に進む。

[0147] この時点で、共有論理CPU管理テーブル400のエントリは空となっているため、図20Cのステップ1084にて図21Cのエントリ番号401を“1”として一時保管用論理CPU管理テーブルエントリ500を共有論理CPU管理テーブル400に挿入する。

[0148] 以上がスケジューリング処理である。

[0149] 図 2 2 は、閾値を閾値管理テーブル 7 0 0 に格納する閾値設定処理の一例を示すフローチャートである。図 2 2 に示す閾値設定処理は、ユーザが入出力装置 1 3、入出力機能 2 1 を介して総キャッシュヒット率閾値および総実行時間閾値の設定命令をハイパバイザ 2 0 に入力した際に、L P A R 管理機能 2 2 で行なわれる処理である。

[0150] L P A R 管理機能 2 2 は、入出力機能 2 1 から閾値設定を行なう L P A R 番号、論理 C P U 番号、総キャッシュヒット率閾値、総実行時間閾値を読み込む (1 1 0 2) 。

[0151] L P A R 管理機能 2 2 は、ステップ 1 1 0 2 で読み込んだ当該 L P A R 番号と論理 C P U 番号を有する閾値管理テーブル 7 0 0 のエントリにおいて、読み込んだ総キャッシュヒット率閾値および総実行時間閾値をそれぞれ閾値 α 7 0 3、閾値 β 7 0 4 として書き込む (1 1 0 3) 。

[0152] 以上の処理により、閾値管理テーブル 7 0 0 が設定される。

[0153] 図 2 3 は、論理 C P U のキャッシュヒット率等を測定するためのモニタ有効化処理の一例を示すフローチャートである。図 2 3 に示すモニタ有効化処理は、ユーザが入出力装置 1 3、入出力機能 2 1 を介してモニタリングの有効化命令をハイパバイザ 2 0 へ入力した際に、モニタリング機能 2 5 にて行なわれる処理である。

[0154] モニタリング機能 2 5 は、入出力機能 2 1 からモニタリングを行なう L P A R 番号と論理 C P U 番号を読み込む (1 1 1 2) 。

[0155] モニタリング機能 2 5 は、ステップ 1 1 1 2 で読み込んだ当該 L P A R 番号と論理 C P U 番号をもつ閾値管理テーブル 7 0 0 のエントリにおいて、モニタ有効フラグ 7 0 5 に “ 1 ” を書き込む (1 1 1 3) 。

[0156] 図 2 4 は、論理 C P U のキャッシュヒット率を測定するためのモニタリング処理の一例を示すフローチャートである。図 2 4 に示すモニタリング処理は、ユーザが入出力装置 1 3、入出力機能 2 1 を介してモニタリング実施命令をハイパバイザ 2 0 へ入力した際に、モニタリング機能 2 5 で行なわれる処理である。

- [0157] モニタリング機能25は、入出力機能21からモニタリングを行なうL P A R番号と論理C P U番号を読み込む（1122）。
- [0158] モニタリング機能25は、ステップ1122で読み込んだ当該L P A R番号と論理C P U番号を含む閾値管理テーブル700のエントリにおいて、モニタ有効フラグ705と書き込み完了フラグ706が共に0であるか否かを判定し（1123）、共に“0”であればステップ1124へ進んで入出力機能21を介してエラー出力する。
- [0159] 一方、モニタ有効フラグ705と書き込み完了フラグ706が共に“0”でない場合は、ステップ1125へ進む。
- [0160] モニタリング機能25は、当該L P A R番号と論理C P U番号を含む閾値管理テーブル700のエントリにおいて、書き込み完了フラグ706が“1”か否かを判定し、当該書き込み完了フラグ706が“1”となるまで待機する（1125）。
- [0161] ステップ1125で書き込み完了フラグ706が“1”となった場合、モニタリング機能25は、当該L P A R番号と論理C P U番号を含む閾値管理テーブル700のエントリにおいて、総キャッシュヒット率707および総実行時間708を、入出力機能21を介して出力し、書き込み完了フラグを“0”に設定する（1126）。
- [0162] 上記図23のモニタ有効化処理と図24のモニタリング処理により、ユーザは任意の論理C P Uについて、総キャッシュヒット率および総実行時間を得ることができる。
- [0163] 図25A、図25B、図25Cは、ユーザがL P A R # 1（801）の論理C P U # 0（811）に対してモニタリングの有効化命令とモニタリング実施命令を行い、モニタリング処理が完了するまでの閾値管理テーブル700の遷移を示す。
- [0164] 図25Aは、図23のモニタ有効化処理を実施した後の閾値管理テーブル700を示す図である。図25Bは、図20A～図20Cの物理C P Uディスパッチ終了処理を実施した後の閾値管理テーブル700を示す図である。

図 2 5 C は、図 2 4 のモニタリング処理を実施した後の閾値管理テーブル 7 0 0 を示す図である。

[0165] なお、モニタ有効化が設定されてモニタリング処理が完了するまでの間に、L P A R # 1 (8 0 1) の論理 C P U # 0 (8 1 1) は総キャッシュヒット率が“0. 4”で、総実行時間が“5 0 0”であったものとする。以下、図 2 5 A ~ 図 2 5 C の遷移の理由を説明する。

[0166] 図 2 5 A は、図 2 3 に示したモニタ有効化処理のステップ 1 1 1 3 の処理によって、閾値管理テーブル 7 0 0 の L P A R 番号 7 0 1 が 1、論理 C P U 番号 7 0 2 が“0”のエントリにおいて、モニタ有効フラグ 7 0 5 が“1”に更新された結果である。

[0167] 図 2 5 B は、スケジューリング処理の物理 C P U ディスパッチ終了処理 9 7 4 として図 2 0 B に示したステップ 1 0 7 3 において、閾値管理テーブル 7 0 0 の L P A R 番号 7 0 1 が“1”、論理 C P U 番号 7 0 2 が“0”のエントリについて、総キャッシュヒット率 7 0 7 が“0. 4”、総実行時間 7 0 8 が“5 0 0”、書き込み完了フラグ 7 0 6 が“1”、モニタ有効フラグ 7 0 5 が“0”に更新された結果である。

[0168] 図 2 5 C は、図 2 4 に示したモニタリング処理のステップ 1 1 2 6 によって、閾値管理テーブル 7 0 0 のうち、L P A R 番号 7 0 1 が“1”、論理 C P U 番号 7 0 2 が“0”のエントリについて、モニタリング機能 2 5 が総キャッシュヒット率 7 0 7 および総実行時間 7 0 8 を入出力機能 2 1 へ出力した後、書き込み完了フラグ 7 0 6 を“0”に更新した結果である。

[0169] ユーザは図 2 2 に示した閾値設定処理によって、各々の論理 C P U に対し、期待する所定のキャッシュヒット率となる時間、すなわち閾値 β まで一つのノード内の物理 C P U で連続実行することを許可する。そして、論理 C P U の総実行時間が、閾値 β を経過した後に、キャッシュヒット率が閾値 α に到達しているか否かに応じて論理 C P U 8 1 1 (8 1 2) を割り当てる物理 C P U 4 ~ 7 を制御できる。

[0170] 制御内容の概要は次の通りである。総キャッシュヒット率が閾値 α に到達

していた場合は、論理CPU 811、812はキャッシュメモリ8、9の恩恵を受けているため引き続き同じノードの物理CPU 4～7で当該論理CPUを連続実行してもよい。このため、ハイパバイザ20は別のノードにHALT状態の物理CPUが存在しても起床させず、現在起床している物理CPUのうちのいずれかにて実行させる。

[0171] 一方、総キャッシュヒット率が閾値 α に到達していない場合、ハイパバイザ20は、当該論理CPU 811で実行した処理についてキャッシュの恩恵を受けていないと判断し、別のノードにHALT状態の物理CPU 4～7が存在する場合は、HALT状態の物理CPUを起床させて当該論理CPUを割り当てる実行させる。これにより、閾値 β よりも長い時間にてメモリアクセス性能を測定した際に、論理CPUからみた物理メモリ10、11のアクセスコストをばらつかせ、ひいては物理メモリ10、11へのメモリアクセスコストを最大値と最小値の中間値にすることができる。

[0172] 閾値 α 、閾値 β の設定に貢献する機能として、モニタリング機能25のモニタ有効化処理（図23）とモニタリング処理（図24）がある。これらの機能によりユーザは、各々の論理CPUにおいて所定のキャッシュヒット率となるまでの連続実行時間を知ることができる。

[0173] 一例として、一つの論理CPUに対して、モニタ有効化処理とモニタリング処理を交互に繰り返し実施し、論理CPUを一つのノードの物理CPU上で実行し続けた際の、実行時間とキャッシュヒット率との関係を図26に示す。

[0174] 図26は、論理CPUの実行時間とキャッシュヒット率との関係を示すグラフである。図中グラフG1は、短時間でキャッシュヒット率が上昇している特徴をもつ論理CPUの実行時間とキャッシュヒット率との関係を示すグラフである。また、グラフG2は、緩やかにキャッシュヒット率が上昇している特徴をもつ論理CPUの実行時間とキャッシュヒット率との関係を示すグラフである。さらに、グラフG3は、キャッシュヒット率が上昇することなく低いまま変動している特徴をもつ論理CPUの実行時間とキャッシュ

ヒット率との関係を示すグラフである。

- [0175] グラフG1の場合では、グラフG2やグラフG3と比べて短時間でキャッシュヒット率が上がっている。このような図となるときは、実行時間が閾値 β を経過した後のキャッシュヒット率の高さから、論理CPU上で処理させるプログラムの使用メモリ量がキャッシュメモリ8、9の容量に比べて小さいこと等が考えられる。
- [0176] グラフG2の場合では、グラフG1と比べて緩やかにキャッシュヒット率が上昇している。このような図となるときは、キャッシュメモリ8、9に大量の書き込みを行うことでキャッシュヒット率が徐々に上がっていると考えられるので、プログラムの使用メモリ量がキャッシュ容量と同程度であること等が考えられる。
- [0177] グラフG3の場合では、グラフG1やグラフG2と比べてキャッシュヒット率が上昇することなく低いまま変動している。このような図となるときは、キャッシュメモリ8、9に書き込みをしてもキャッシュヒットが増えないことからプログラムの使用メモリ量がキャッシュ容量と比べて大きいこと等が考えられる。
- [0178] 以上のように、論理CPU811、812で処理させるプログラムによってキャッシュヒット率が所定の閾値 α まで上昇する時間は異なるのだが、モニタリング機能25によってグラフG1～G3のようなキャッシュヒット率と、論理CPU811、812の連続実行時間との関係を得ることができ、ひいては所定のキャッシュ率となるまでの時間を調べることができるのである。
- [0179] この結果を用いて、例えば、グラフG1のようなメモリ使用量の少ないプログラムと図グラフG3のようなメモリ使用量の多いプログラムを一つの論理CPU811（812）で処理させたい場合には、グラフG1の閾値 α をキャッシュメモリ8、9と物理メモリ1、011へのアクセス回数の比がちょうど中間となる“0.5”よりも大きい値に設定する。そして、キャッシュヒット率が閾値 α となるまでの論理CPU811（812）の実行時間を

閾値 β として、上述の閾値設定処理にて設定する。なお、物理メモリ1、011へのアクセス回数は、論理CPU811（812）キャッシュミスの回数となる。

[0180] これにより、閾値 β に達する時間まで同じノードの物理CPU4～7で論理CPU811（812）を実行している際に、論理CPU811（812）がメモリ使用量の少ないプログラムを処理しているときには、閾値 α よりもキャッシュヒット率が高いためにキャッシュメモリ8、9の恩恵が受けやすくなり、メモリ使用量の多いプログラムを処理しているときには閾値 α よりもキャッシュヒット率が低くなるため、論理CPU811（812）から見た物理メモリ10、11へのアクセスコストをばらつかせ、ひいてはメモリアクセスコストを最大値と最小値の中間値にすることができる。

[0181] また、論理CPU811（812）のメモリアクセス性能をより向上させるために、モニタリング機能25を用いて所定のキャッシュヒット率 α となったか否かを判定し、LPAR管理機能22の閾値設定処理にてキャッシュヒット率を入出力装置13にフィードバックして閾値 α と閾値 β をリアルタイムでチューニングする、ということも可能となる。

[0182] 以上の通り、多数の物理CPU（またはコア）4～7を備えた本実施例の仮想計算機システムにより、休止状態の物理CPUが存在する状態において、論理CPU811、812のキャッシュヒット率が閾値 α よりも高いときは、同じノードの物理CPU4～7で論理CPU811、812を再度実行させてキャッシュメモリ8、9の恩恵を受けやすくする。一方、キャッシュヒット率が低い際には物理メモリ10、11へのアクセスを複数のノード上の物理CPU間で移動させ、論理CPU811、812から見た物理メモリ1、011へのアクセスコストを最大値と最小値の中間値にすることができる。

[0183] 次に、図27～図28を用いて従来例によるメモリアクセスの一例を説明する。図27はメモリアクセスの位置によりメモリアクセスコストが異なるNUMA構成のハードウェア（仮想計算機）1の構成を示すブロック図であ

る。この例では、物理CPU 1 (5) が属するノード 0 (2) とは異なるノード (1) 3 の物理メモリ 1 (1 1) に、物理CPU 1 (5) からアクセス (3 1) する際のコストは、ノード (0) 2 とノード (1) 3 との間を接続するインターコネクト 1 4 を経由しなくてはならないため、物理CPU (1) 5 が属するノード (0) 2 の物理メモリ (0) 1 0 に物理CPU (1) 5 からアクセス 3 0 する際のコストより大きなものになってしまう。なお、ハードウェア 1 では、従来のハイパバイザ (図示省略) が稼動する。

[0184] 一方で、従来技術では複数の L P A R 8 0 1、8 0 2 の論理CPU 8 1 1、8 1 2 として物理CPU 4 ~ 7 を時分割共有する際に、物理CPU 4 ~ 7 が十分にある場合は休止状態の物理CPU を起床させず、起床している物理CPU のみの上で動作させようとする。この一例を図 2 8 に示す。

[0185] 図 2 8 では、各ノード 0、1 の物理CPU にて実行させる論理CPU (実線) とハイパバイザ (点線) の時系列を示しており、図中下方向に時間が進むものとしている。また、矢印 4 0 を論理CPU の処理に要する時間、破線の矢印 6 0 をハイパバイザの処理に要する時間、破線 7 0 を休止状態の時間、メモリアクセス性能の平均値を測定する時間を時間 T 8 0 とする。論理CPU の処理を示す矢印 4 0 の長さは最大でも、ハイパバイザにて予め定義されるタイムスライスとよばれる単位時間となる。タイムスライスは一般的には数十ミリ秒程度に設定される。

[0186] 図 2 8 では、論理CPU が一つのみ物理計算機上で稼動しており、ノード 0 に属する物理CPU で論理CPU の処理の実行が開始されている。その後、論理CPU の実行 4 0 が一旦終了してハイパバイザのスケジューリング処理 6 0 に移り、スケジューリング処理が論理CPU を再度実行させる先となる物理CPU を選択しようとしたところ、ノード 1 に属する物理CPU が全て休止状態 7 0 であったため、今まで論理CPU を実行させていたノード 0 に属する物理CPU を再度割り当てて実行させる。このように、ハイパバイザのスケジューリング処理がノード 0 に属する物理CPU を論理CPU に割り当て続ける処理は、ノード 1 に属する物理CPU が起床しない限り繰り返

されてしまう。

[0187] 図28の場合において、LPARの論理CPUとして物理CPU1(5)が割り当てられ、かつ論理CPU上のプログラムにてメモリアクセス31を伴う処理を行なう場合を考える。時間Tの間、例えば一般的なメモリ測定の時間単位である一秒の間に注目したとき、論理CPUはノード0に属する物理CPUでしか実行していないことから、論理CPUはリモートでメモリアクセス31を常に行なっていることになる。このため、時間Tの平均メモリアクセス性能は、リモートメモリへのアクセス性能、すなわち低い性能となってしまう。

[0188] ただし、上記の場合でも論理CPU上のプログラムにて利用するメモリ量が物理キャッシュメモリに収まる場合は、利用するメモリ内容がキャッシュに収まり次第、キャッシュメモリへのアクセスで足りるため、NUMA構成による物理メモリへのアクセス性能の影響は受けなくなる。言い換えると、短時間にて複数の物理CPU間で論理CPUの割り当て先を変更させるだけではキャッシュメモリの恩恵を受けにくくなってしまうため、キャッシュ内にアクセスしたいデータや命令の含有率、すなわちキャッシュヒット率が高い状態で保たれたときには、そのキャッシュの恩恵も受けられるようにするべきである。

[0189] 本発明では、論理CPUが同じノード上の物理CPUで連続実行された際に、ハイパバイザ20にて当該論理CPUの連続実行時間が、所定のキャッシュヒット率となるまでの時間 β を経過しているか否かを判定し、時間 β を経過している場合には論理CPUのキャッシュヒット率が閾値 α に達しているか否かを判定する。

[0190] そして、ハイパバイザ20は、キャッシュヒット率が閾値 α よりも低い場合は、キャッシュメモリ8、9の恩恵が受けられていないと判断し、論理CPUを連続実行したノードとは別のノード上に休止状態の物理CPUが存在すれば当該物理CPUを起床して当該論理CPUを実行させる。

[0191] 一方、キャッシュヒット率が高い(閾値 α に達している)場合は、論理C

PUを連続実行したノードと別のノード上に休止状態の物理CPUが存在しても起床させず、ハイパバイザ20は引き続き現在のノードで起床している物理CPUに当該論理CPUを実行させる。また、キャッシュヒット率は低い、論理CPUを連続実行させたノードとは別のノード上に休止状態の物理CPUが存在しなかった場合に関しても、ハイパバイザ20は現在起床している物理CPUにて当該論理CPUを再度実行させる。

[0192] 実行させる論理CPUはハイパバイザ20の論理CPUのキューにて保持し、キューに入った順序で論理CPUを物理CPU上にて実行させ、所定の周期（時分割の周期など）で論理CPUの実行が完了したら再度論理CPUをキューに入れる。

[0193] 総実行時間閾値 β はメモリ性能の平均値を測定する時間Tよりも小さいものとする。この理由は上述の通りである。

[0194] 図29に本発明にて総キャッシュヒット率の閾値 $\alpha = 0.5$ 、総実行時間の閾値 β が論理CPUのタイムスライス二つ分であった場合に、各ノードの物理CPU4～7で実行させる論理CPUとハイパバイザの時系列の例を示している。こちらの例も上記図28と同様に、図中下方向に時間が進むものとしており、矢印41～44は論理CPU（例えば811）の処理に要する時間、破線の矢印61はハイパバイザ20の処理に要する時間、破線71は休止状態の時間、メモリ性能の平均値を測定する時間は時間T81としている。

[0195] また、図29では論理CPUを一つのみ実行させている。このため、ハイパバイザ20の論理CPUのキューには当該論理CPU811一つしか出し入れされることがなく、当該論理CPU811の実行完了時に、当該論理CPUを論理CPUのキューに再度投入し、次の論理CPUの選択時に当該論理CPUを論理CPUのキューから取り出すことになる。

[0196] 総実行時間の閾値 β を経過していない期間である90～92では、論理CPUのキャッシュヒット率は閾値 α に至っていないとハイパバイザ20が判断し、ハイパバイザ20は別のノード1上に休止状態の物理CPUが存在し

ても起床させずにできるだけ起床している物理CPUにて論理CPUを実行させようとする。

[0197] 最初の総実行時間の閾値 β を経たとき、すなわち期間90の直後に当該論理CPUのキャッシュヒット率が“0.3”だったとする。この場合、ハイパバイザ20は論理CPUのキャッシュヒット率が閾値 α 未満であるので、キャッシュメモリ8、9の恩恵を受けていないと判断し、ノード1に属する物理CPUを起床し、当該論理CPUをノード1に属する物理CPUで実行させている。

[0198] 二回目の総実行時間の閾値 β を経た期間91の直後に当該論理CPUのキャッシュヒット率が“0.2”だったとする。この場合、ハイパバイザ20は一回目の総実行時間の閾値 β を経過した期間90と同様に、論理CPUがキャッシュメモリ8、9の恩恵を受けていないと判断し、ノード0に属する物理CPUを起床し、当該論理CPUをノード0に属する物理CPUで再度実行させている。

[0199] 三回目の総実行時間の閾値 β を経たとき、すなわち期間92の直後に当該論理CPUのキャッシュヒット率が“0.6”と、総キャッシュヒット率閾値 α 以上だったとすると、ハイパバイザ20は論理CPUがキャッシュの恩恵を受けているものと判断し、ノード1に属する物理CPUを起床することなく、引き続き論理CPUをノード0に属する物理CPUで実行させている(44)。

[0200] このとき、時間Tに注目すると、期間90の直後および期間91の直後ではともにキャッシュヒット率が総キャッシュヒット率閾値 α よりも低いため、論理CPUから物理メモリ10、11へのアクセスが頻繁に行なわれると考えることができる。さらに、時間Tの間では、論理CPUの実行時間41の総和と、論理CPUの実行時間42の総和はともに総実行時間の閾値 β 以上の時間となる。このことにより、時間Tの間では論理CPUはノード0に属する物理CPUとノード1に属する物理CPUにて同じ総実行時間の閾値 β 分だけ論理CPUを実行させていることとなる。

[0201] 図27において、図29のノード0に属する物理CPUを物理CPU1（5）、ノード1に属する物理CPUを物理CPU2（6）とし、図29における論理CPUが物理メモリ1（11）へのメモリアクセスを継続していた際のハードウェアの図を図30に示す。この場合、時間Tの間での物理メモリアクセスは、論理CPUがノード0に属する物理CPU1（5）とノード1に属する物理CPU2（6）にて同じ総実行時間分実行していることから、メモリアクセス31とメモリアクセス32を同程度の時間でそれぞれ行なっていることになる。ゆえに、当該論理CPUの時間Tにおける平均の物理メモリアクセスコストは、ローカルメモリのアクセスコスト、すなわち最小値と、リモートメモリのアクセスコスト、すなわち最大値との中間程度となる。

[0202] 本発明により、NUMA構成の仮想計算機のような、メモリのアクセス位置によってメモリアクセス性能（レイテンシ）が異なるハードウェアで、論理CPUの数に対し物理CPUが十分にある場合では、論理CPUに対する物理メモリアクセス性能は、1秒程度の時間Tにて観測した際に、最高性能と最低性能の中間程度の性能が少なくとも得られ、さらにキャッシュヒット率が閾値 α 以上の際にはキャッシュメモリ8、9による高いメモリリード、ライト性能となる効果が得られる。

[0203] また、本発明において、休止状態の物理CPUが存在しないような仮想計算機システムがビジー状態の場合を図31に示す。図31は、図28と同様に、ノード0、1の物理CPUで実行させる論理CPUとハイパバイザの処理の時系列を示しており、図中下方向に時間が進むものとする。

[0204] 図31の場合では、実行させる論理CPUとして論理CPUa、論理CPUb、論理CPUcの三つが存在し、矢印50、53、56は論理CPUaの処理に要する時間、矢印51、54は論理CPUbの処理に要する時間、矢印52、55、57は論理CPUcの処理に要する時間、破線の矢印62～67はハイパバイザ20の処理に要する時間、メモリ性能の平均値を測定する時間は時間T82である。また、論理CPUやハイパバイザ20の処理

が開始される際の論理CPUのキューに入っている論理CPU状態も時系列にて併せて図示する。ハイパバイザ20は、次に実行させる論理CPUを、キューのうち図中左側の要素から順次取り出し、論理CPUの実行が完了したら図中右側のキューの要素として投入される。例えば、論理CPU bの処理51が終わったときのキューではc、a、bとなり、論理CPU bの処理が終わったために図中キューの右側にbが挿入され、次回実行する論理CPUとして左側のc、すなわち論理CPU cがハイパバイザ20によって取り出されることを意味する。

[0205] 論理CPUの上で処理するプログラムによって実行時間は異なるため、各論理CPUの実行時間はまちまちとなり、先に実行を開始した論理CPUよりも後で実行を開始した論理CPUの方が先に実行を完了すると、論理CPUキューに入る順序は逆転する、という現象が発生する。図31においては、論理CPU bの処理54と、論理CPU cの処理55がこの現象に該当する。このため、論理CPUがキューに入る順序に関して、いわばランダム性が働く。

[0206] 論理CPUのキューに入る順序に関してランダム性が働くため、ハイパバイザ20がキューから取り出す論理CPUもランダムとなり、ひいては論理CPUを実行させる物理CPUもランダムとなる。

[0207] このため、仮想計算機システムがビジー状態の場合において、論理CPUから物理メモリへのアクセスは、ローカルメモリのアクセスと、リモートメモリのアクセスをランダムに行なうことになる。このため、結果的に論理CPUに対するメモリアクセスコストが1秒程度の時間Tにて観測すると最大値と最小値の中間程度の性能が得られることになるのである。

[0208] 以上のように、本発明では、論理CPU 811, 812を実行した時間が閾値 β を経過すると、ハイパバイザ20は、論理CPU 811, 812のキャッシュヒット率と閾値 α とを比較した結果に応じて、当該論理CPUを次に実行させる物理CPU 4~7を決定する。そして、ハイパバイザ20は、論理CPU 811, 812の性能情報であるキャッシュヒット率が閾値 α 以

上であれば、同じノード内の物理CPU 4～5を選択することで、論理CPU 811はキャッシュメモリ8に格納されたデータまたは命令を読み込んで、効率よく処理を実行できる。

- [0209] 一方、ハイパバイザ20は、論理CPU 811、812のキャッシュヒット率が閾値 α 未満であれば、他のノードで休止している物理CPU 6～7を選択し、当該物理CPUを起床させてから論理CPU 811、812を実行させる。これにより、キャッシュヒット率が閾値 α 未満であれば、論理CPU 811、812を実行させる物理CPUを他のノードの物理CPUに順次移動させることで、物理メモリへのアクセスコストが高い状態が継続するのを抑制できる。

実施例 2

- [0210] なお、実施例1では、モニタリング機能25が各物理CPU 4～7のキャッシュメモリ8、9のアクセスに関する性能情報を測定する例を示したが、図32で示すように、ノード2、3に接続されたプロセッサ性能監視ユニット12で、物理CPU 4～7のキャッシュリード回数、キャッシュヒット回数などの性能情報を測定し、モニタリング機能25でキャッシュヒット率を演算するようにしてもよい。
- [0211] プロセッサ性能監視ユニット12は、ノード2、3にそれぞれ接続されて、物理CPU 4～7（または論理CPU）の性能に関する情報を測定する。物理CPU 4～7（または論理CPU）のキャッシュメモリ8、9のリード回数（キャッシュリード回数）を測定するカウンタと、キャッシュメモリ8、9のヒット回数（キャッシュヒット回数）を測定するカウンタをそれぞれ備える（図示省略）。なお、図32の例では、物理CPU 4、5でキャッシュメモリ8を共有し、物理CPU 6、7がキャッシュメモリ9を共有し、キャッシュプロセッサ性能監視ユニット12は、各物理CPU 4～7のキャッシュリード回数とキャッシュヒット回数をそれぞれ計数するカウンタを備える。
- [0212] なお、プロセッサ性能監視ユニット12が論理CPUのキャッシュリード

回数、キャッシュヒット回数をカウントする場合、プロセッサ性能監視ユニット12からキャッシュメモリ8、9へのリードを観測すると、物理CPUと論理CPUの区別はつかない。このため、ハイパバイザ20は、プロセッサ性能監視ユニット12のカウンタをリセットした時点以降を、論理CPUのアクセスとみなしてキャッシュリード回数とキャッシュヒット回数を測定する。

[0213] なお、上記各実施例において、タイマ16～19にセットするタイムスライスの時間は、L P A R 8 0 1、8 0 2 のゲストOS（図示省略）のタイマ割り込みの周期としてもよい。

[0214] なお、本発明において説明した計算機等の構成、処理部及び処理手段等は、それらの一部又は全部を、専用のハードウェアによって実現してもよい。

[0215] また、本実施例で例示した種々のソフトウェアは、電磁的、電子的及び光学式等の種々の記録媒体例えば、非一時的な記憶媒体に格納可能であり、インターネット等の通信網を通じて、コンピュータにダウンロード可能である。

[0216] また、本発明は上記した実施例に限定されるものではなく、様々な変形例が含まれる。例えば、上記した実施例は本発明をわかりやすく説明するために詳細に説明したものであり、必ずしも説明した全ての構成を備えるものに限定されるものではない。

請求の範囲

[請求項1]

キャッシュメモリを備えた物理CPUと物理メモリを含むノードを複数有し、前記複数のノードの計算機資源を仮想計算機に割り当てる仮想化部と、を備えた仮想計算機システムであって、

前記複数のノードは、インターコネクトを介して接続され、

前記仮想化部は、

前記仮想計算機に割り当てる計算機資源として論理区画と論理CPUを生成する論理区画管理部と、

前記論理CPUを実行する前記物理CPUを選択するスケジューリング部と、

前記論理CPUから前記物理メモリへのアクセス性能に関する性能情報を測定するモニタリング部と、を備え、

前記スケジューリング部は、

前記性能情報が所定の閾値を満たす場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記性能情報が所定の閾値を満たさない場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択することを特徴とする仮想計算機システム。

[請求項2]

請求項1に記載の仮想計算機システムであって

前記モニタリング部は、

前記論理CPUを実行した際のキャッシュリード回数とキャッシュヒット回数を取得し、一つの論理CPUを連続実行させる間のキャッシュリード回数とキャッシュヒット回数を保持するキャッシュ情報とを有し、

前記スケジューリング部は、

前記論理CPUを実行させるたびに前記モニタリング部により前記キャッシュ情報を更新することを特徴とする仮想計算機システム。

- [請求項3] 請求項2に記載の仮想計算機システムであって
- 前記モニタリング部は、
- 前記キャッシュ情報から前記論理CPUを実行した際のキャッシュリード回数とキャッシュヒット回数を取得してキャッシュヒット率を算出し、
- 前記スケジューリング部は、
- 前記性能情報としてのキャッシュヒット率が所定の閾値以上の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記性能情報としてのキャッシュヒット率が所定の閾値未満の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択することを特徴とする仮想計算機システム。
- [請求項4] 請求項3に記載の仮想計算機システムであって
- 前記スケジューリング部は、
- 前記論理CPUを連続実行させた時間を測定して、前記論理CPU毎に前記連続実行させた時間を論理CPU管理情報に格納し、前記物理CPUで前記論理CPUを前記連続実行させた時間が所定の第2の閾値を超えたときには、
- 前記キャッシュヒット率が所定の閾値以上の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記キャッシュヒット率が所定の閾値未満の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択することを特徴とする仮想計算機システム。
- [請求項5] 請求項4に記載の仮想計算機システムであって
- 前記スケジューリング部は、
- 前回実行した物理CPUのノードとは異なる他のノードから選択する場合には、前記他のノードから休止状態の物理CPUを選択し、当

該物理CPUを起床させて前記論理CPUを実行することを特徴とする仮想計算機システム。

[請求項6]

キャッシュメモリを備えた物理CPUと物理メモリを含むノードを複数有し、前記複数のノードの計算機資源から論理CPUと論理区画を生成して仮想計算機を稼働させる仮想化部が、前記論理CPUを物理CPUに割り当て実行させるスケジューリング方法であって、

前記複数のノードは、インターコネクトを介して接続され、

前記仮想化部が、前記論理CPUを割り当てる前記物理CPUを選択する第1のステップと、

前記仮想化部が、前記論理CPUから前記物理メモリへのアクセス性能に関する性能情報を測定する第2のステップと、

前記仮想化部は、前記性能情報が所定の閾値を満たす場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記性能情報が所定の閾値を満たさない場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択する第3のステップと、

を含むことを特徴とするスケジューリング方法。

[請求項7]

請求項6に記載のスケジューリング方法であって

前記第2のステップは、

前記論理CPUを実行した際のキャッシュリード回数とキャッシュヒット回数を取得し、一つの論理CPUを連続実行させる間のキャッシュリード回数とキャッシュヒット回数をキャッシュ情報に保持し、前記論理CPUを実行させるたびに前記キャッシュ情報を更新することを特徴とするスケジューリング方法。

[請求項8]

請求項7に記載のスケジューリング方法であって

前記第2のステップは、

前記キャッシュ情報から前記論理CPUを実行した際のキャッシュ

リード回数とキャッシュヒット回数を取得してキャッシュヒット率を算出し、

前記第3のステップは、

前記性能情報としてのキャッシュヒット率が所定の閾値以上の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記性能情報としてのキャッシュヒット率が所定の閾値未満の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択することを特徴とするスケジューリング方法。

[請求項9]

請求項8に記載のスケジューリング方法であって

前記第3のステップは、

前記論理CPUを連続実行させた時間を測定して、前記論理CPU毎に前記連続実行させた時間を論理CPU管理情報に格納し、前記物理CPUを割り当てた論理CPUの前記連続実行させた時間を前記論理CPU管理情報から取得し、前記連続実行させた時間が所定の第2の閾値を超えたときには、

前記キャッシュヒット率が所定の閾値以上の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUと同一ノードから選択し、前記キャッシュヒット率が所定の閾値未満の場合には、前記論理CPUを実行する前記物理CPUを、前回実行した物理CPUのノードとは異なる他のノードから選択することを特徴とするスケジューリング方法。

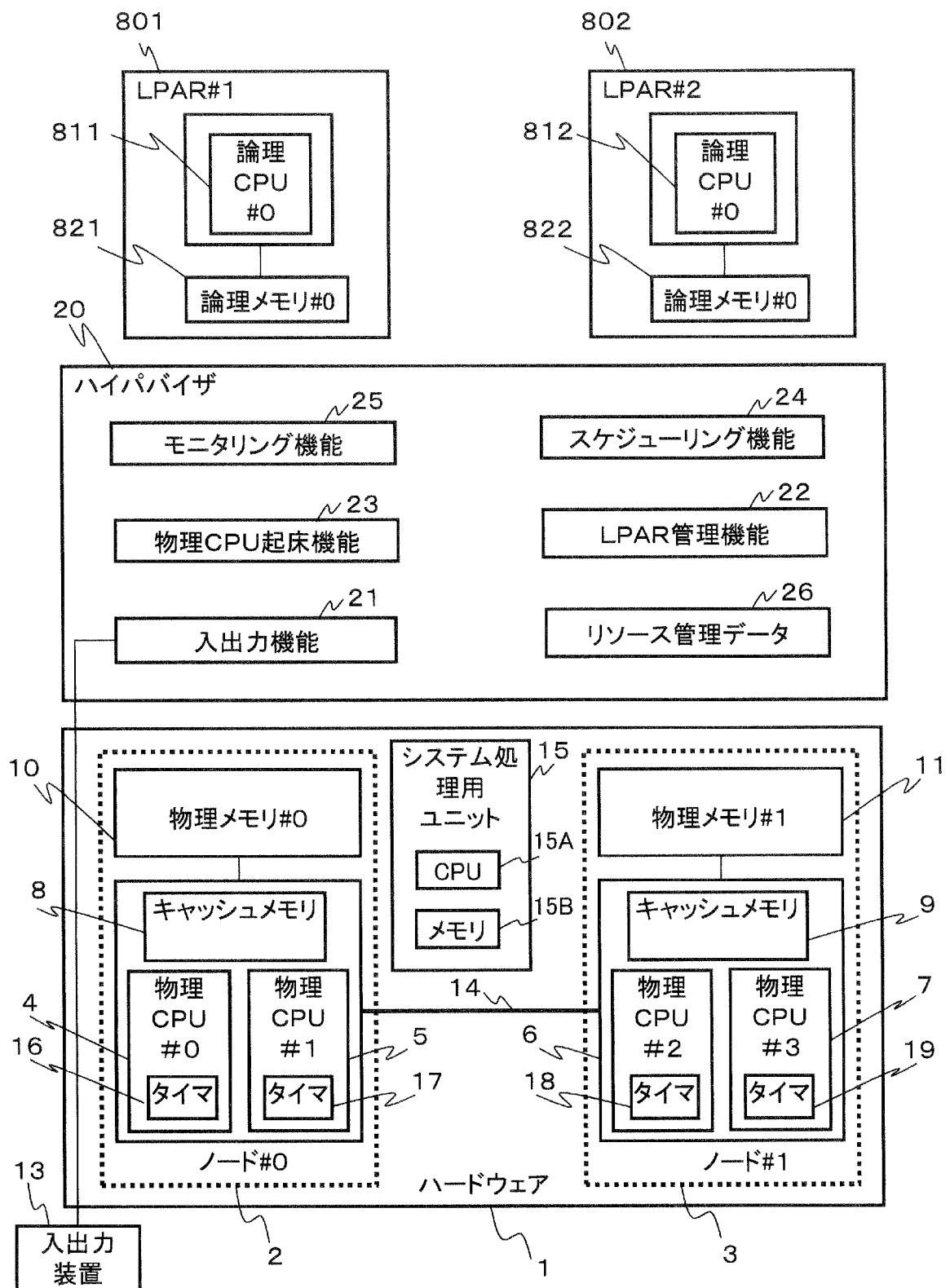
[請求項10]

請求項9に記載のスケジューリング方法であって

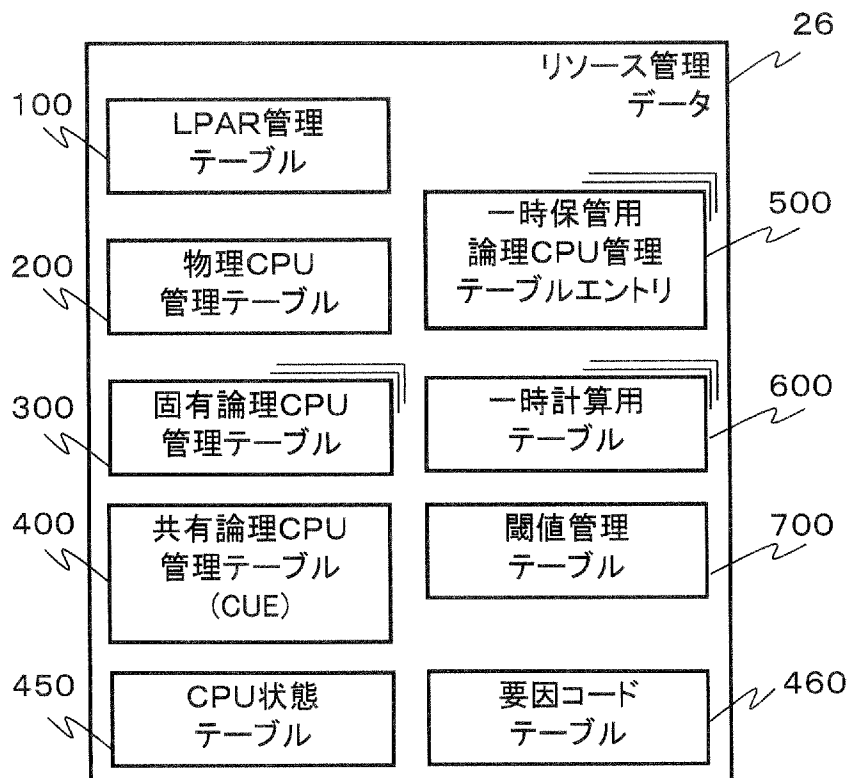
前記第3のステップは、

前回実行した物理CPUのノードとは異なる他のノードから選択する場合には、前記他のノードから休止状態の物理CPUを選択し、当該物理CPUを起床させて前記論理CPUを実行することを特徴とするスケジューリング方法。

[図1A]



[図1B]



[図2]

100 LPAR管理テーブル

101	102	103	104
LPAR番号	論理CPU個数	割り当て 物理メモリ開始位 置	割り当て 物理メモリ サイズ
1	1	0x1000	0x1000
2	1	0x2500	0x1000

[図3]

200 物理CPU管理テーブル

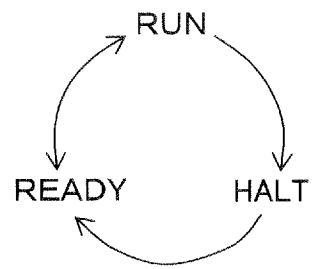
201 物理CPU番号	202 物理CPU状態	203 物理CPUノード番号
0	3	0
1	3	0
2	3	1
3	3	1

[図4A]

450 CPU状態テーブル

451 コード	452 CPU状態
1	RUN
2	READY
3	HALT

[図4B]



[図5]

300 固有論理CPU管理テーブル

301	302	303	304	305	306	307	308	309	310
エントリ番号	LPAR番号	論理CPU番号	論理CPU状態	前回ノード番号	総キャッシュリード回数	総キャッシュヒット回数	総実行時間	ノード0の物理CPU実行時間	ノード1の物理CPU実行時間

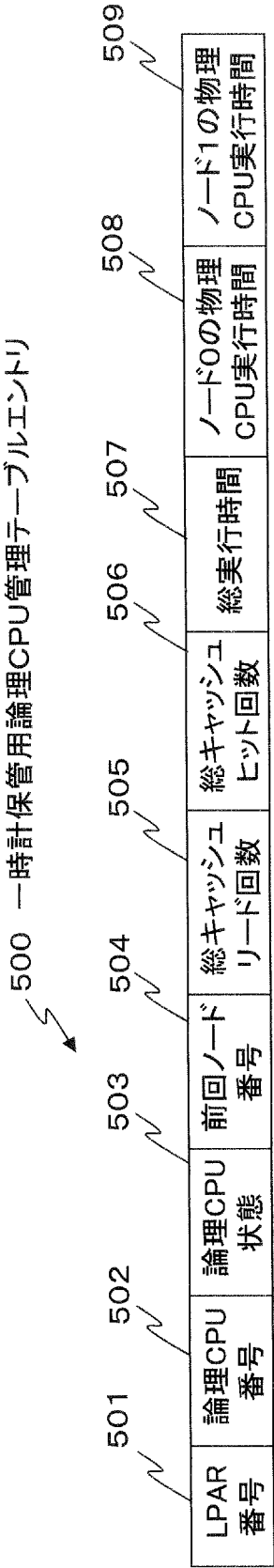
[図6]

400 共有論理CPU管理テーブル

↓

401	402	403	404	405	406	407	408	409	410
エントリ 番号	LPAR 番号	論理CPU 番号	論理CPU 状態	前回ノード 番号	総キャッシュ リード回数	総キャッシュ ヒット回数	総実行時間	ノード0の物理 CPU実行時間	ノード1の物理 CPU実行時間

[図7]



[図8]

600 一時計算用テーブル

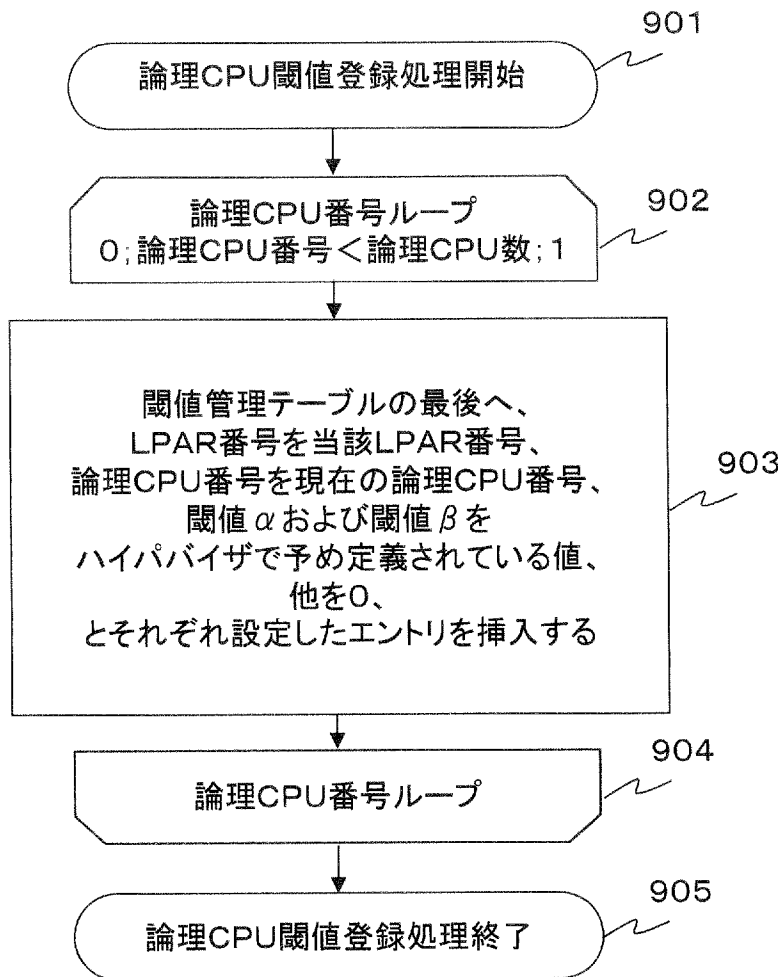
601	602	603	604	605
総キャッシュ ヒット率数	キャッシュ リード回数	キャッシュ ヒット回数	実行開始時刻	実行終了時刻
0	0	0	0	0

[図9]

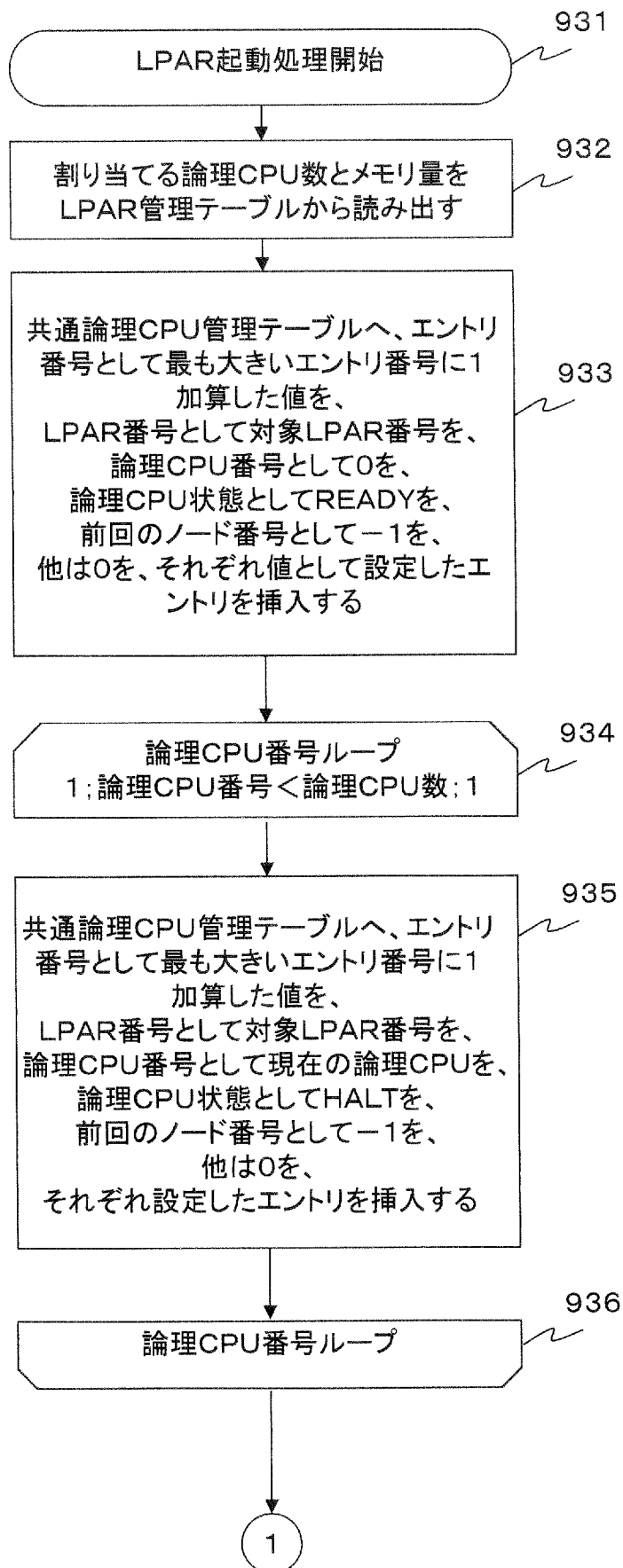
700 閾値管理テーブル

LPAR番号	論理 CPU番号	閾値 α	閾値 β	モニタ有効 フラグ	書き込み完了 フラグ	総キャッシュ ヒット率	総実行 時間
1	0	0.5	2000	0	0	0	0
2	0	0.5	2000	0	0	0	0

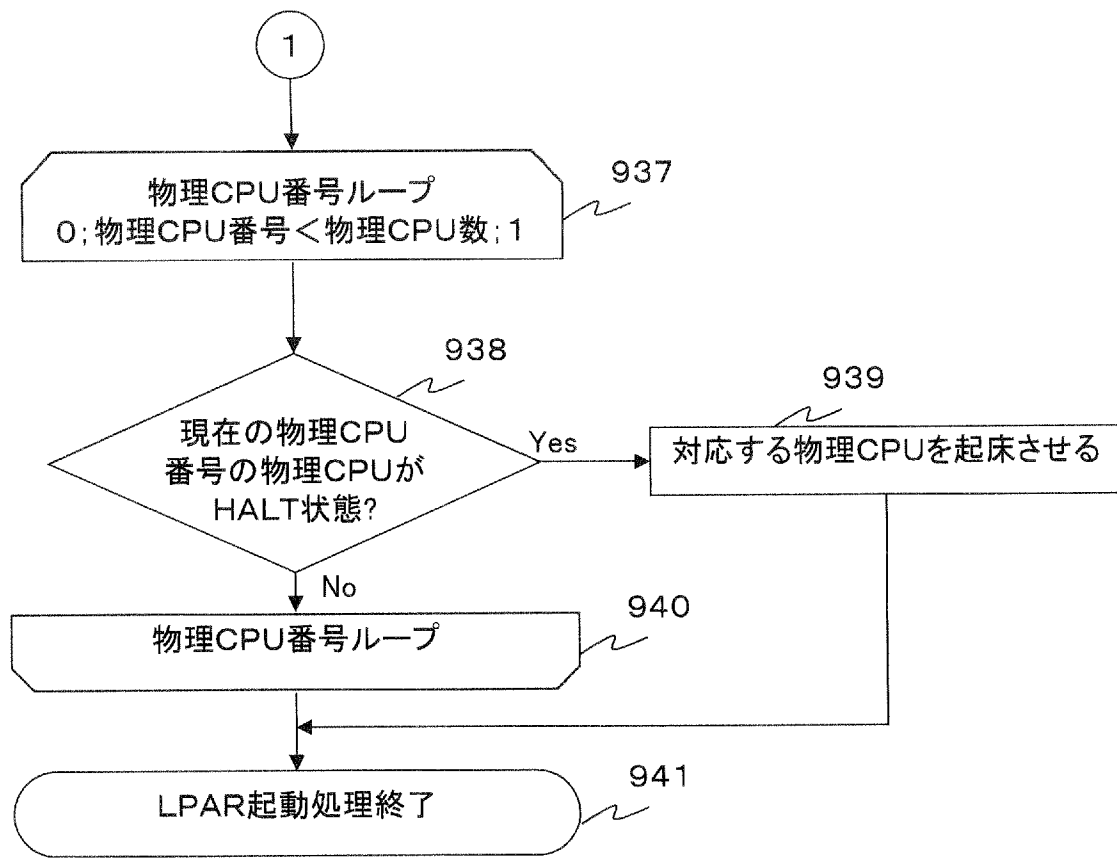
[図10]



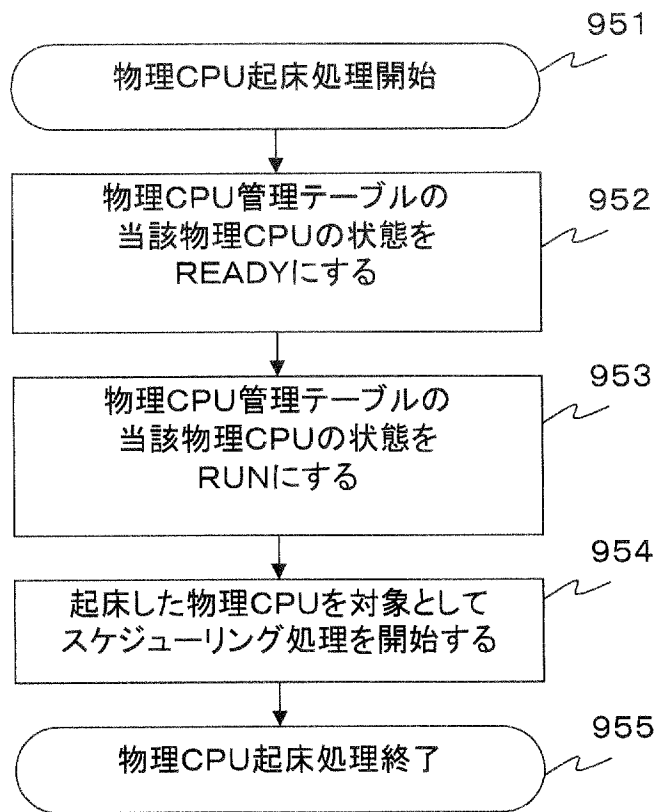
[図11A]



[図11B]



[図12]



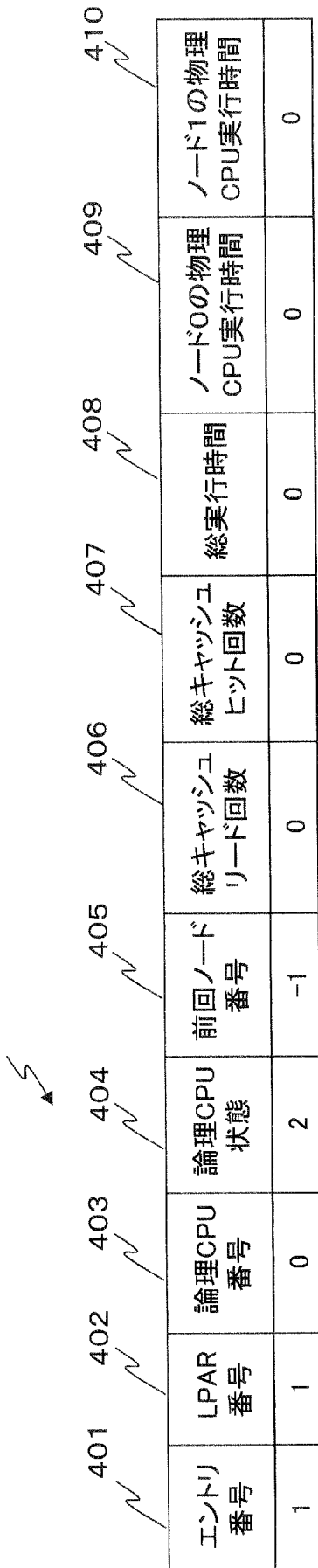
[図13]

物理CPU管理テーブル(200)

物理CPU番号	物理CPU状態	物理CPUノード番号
0	3→2→1	0
1	3	0
2	3	1
3	3	1

[図14]

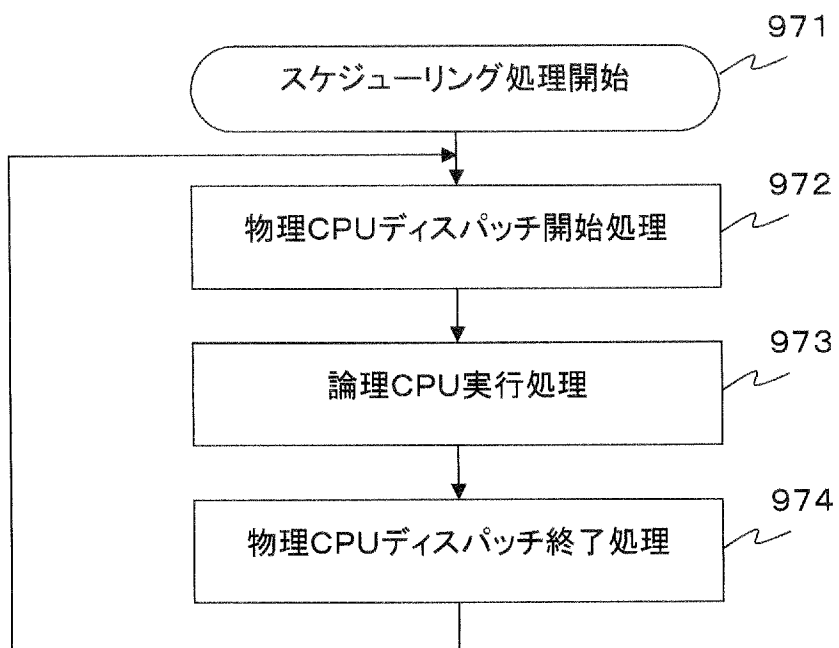
400 共有論理CPU管理テーブル



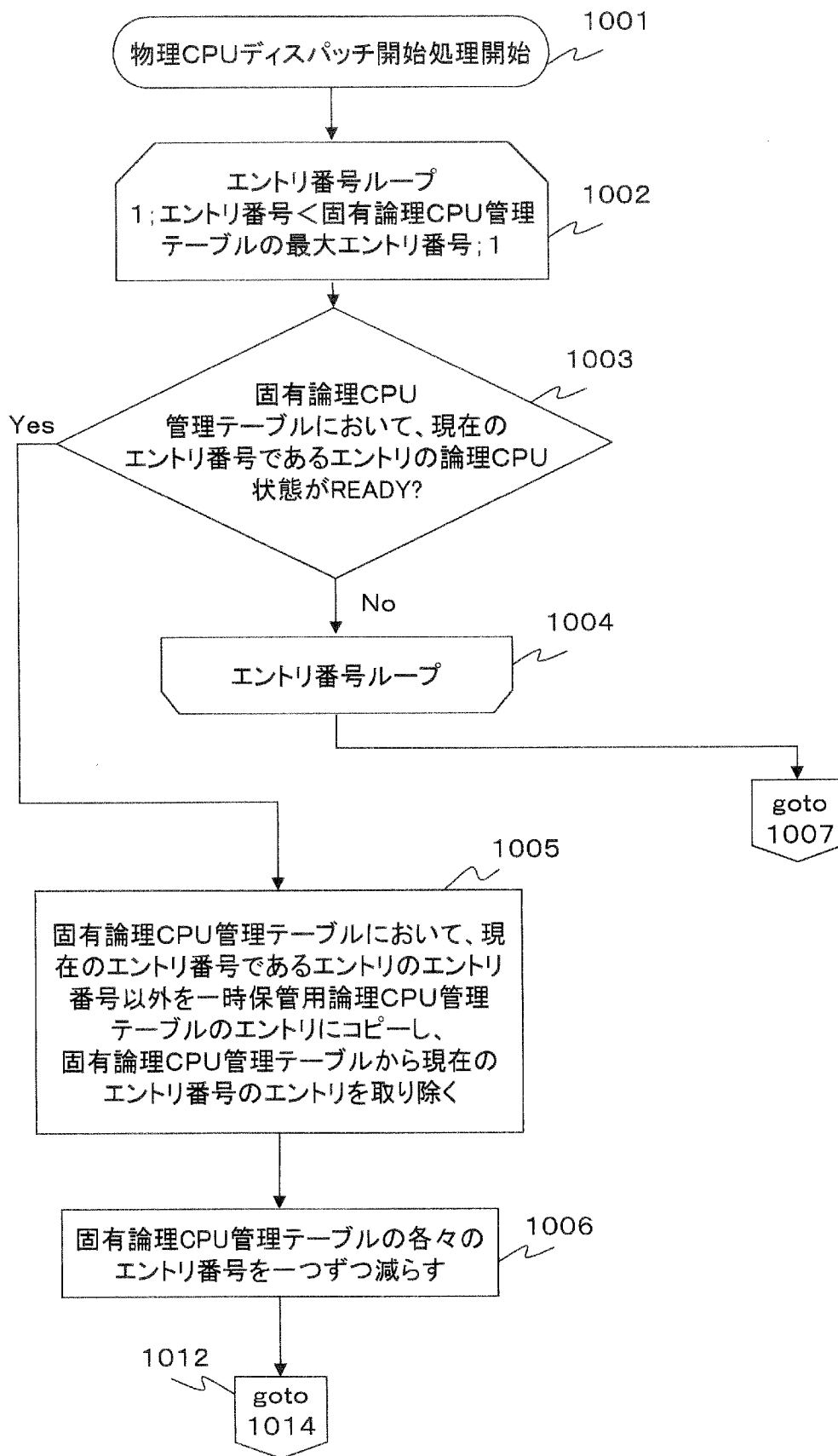
The diagram shows a table with 10 columns and 2 rows. The columns are labeled with numbers 401 through 410. Above the table, there are handwritten squiggly lines under each column number, and a lightning bolt symbol with an arrow pointing to the first column (401). The table contains the following data:

401	402	403	404	405	406	407	408	409	410
エントリー 番号	LPAR 番号	論理CPU 番号	論理CPU 状態	前回ノード 番号	総キャッシュ リード回数	総キャッシュ ヒット回数	総実行時間	ノード0の物理 CPU実行時間	ノード1の物理 CPU実行時間
1	1	0	2	-1	0	0	0	0	0

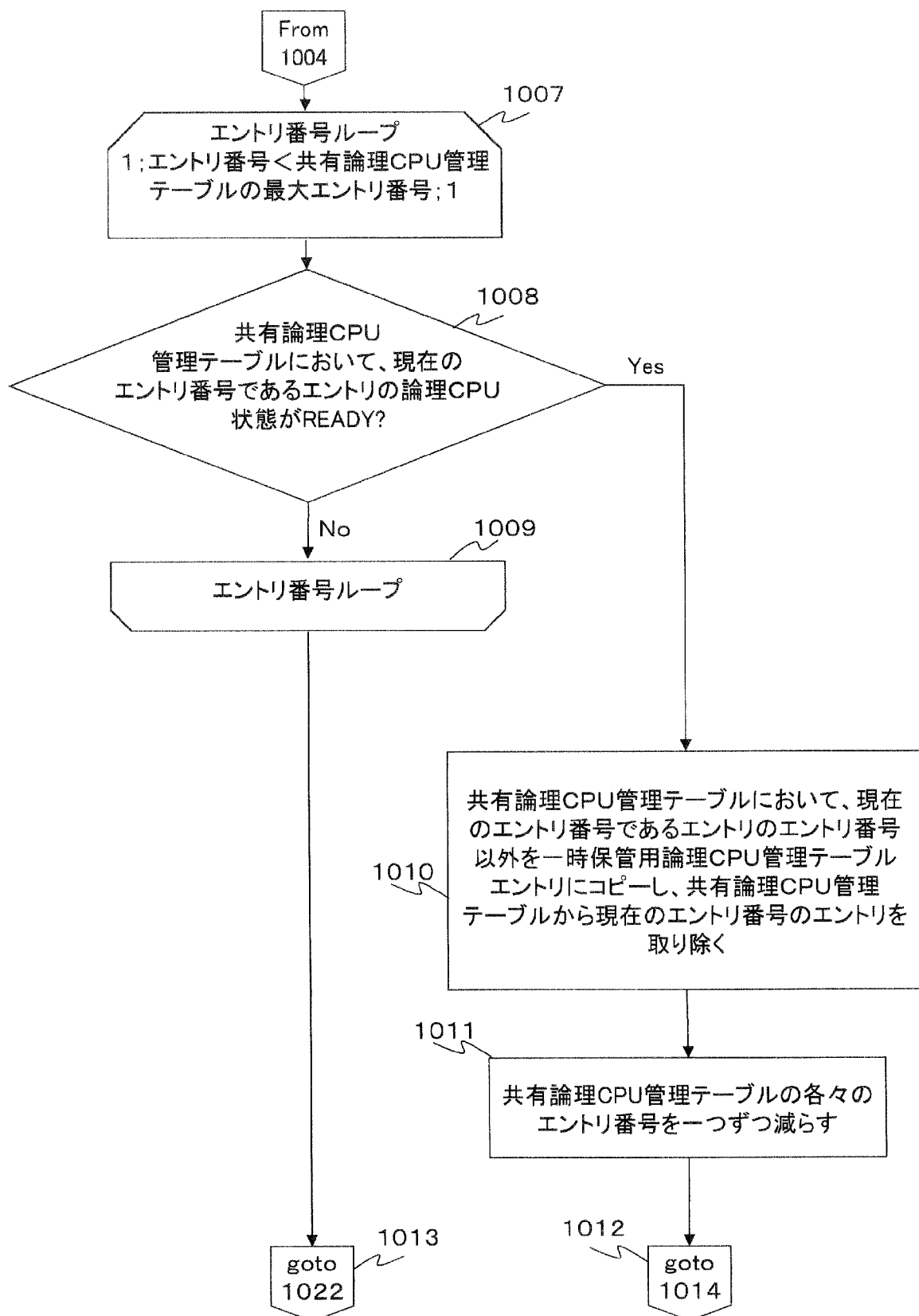
[図15]



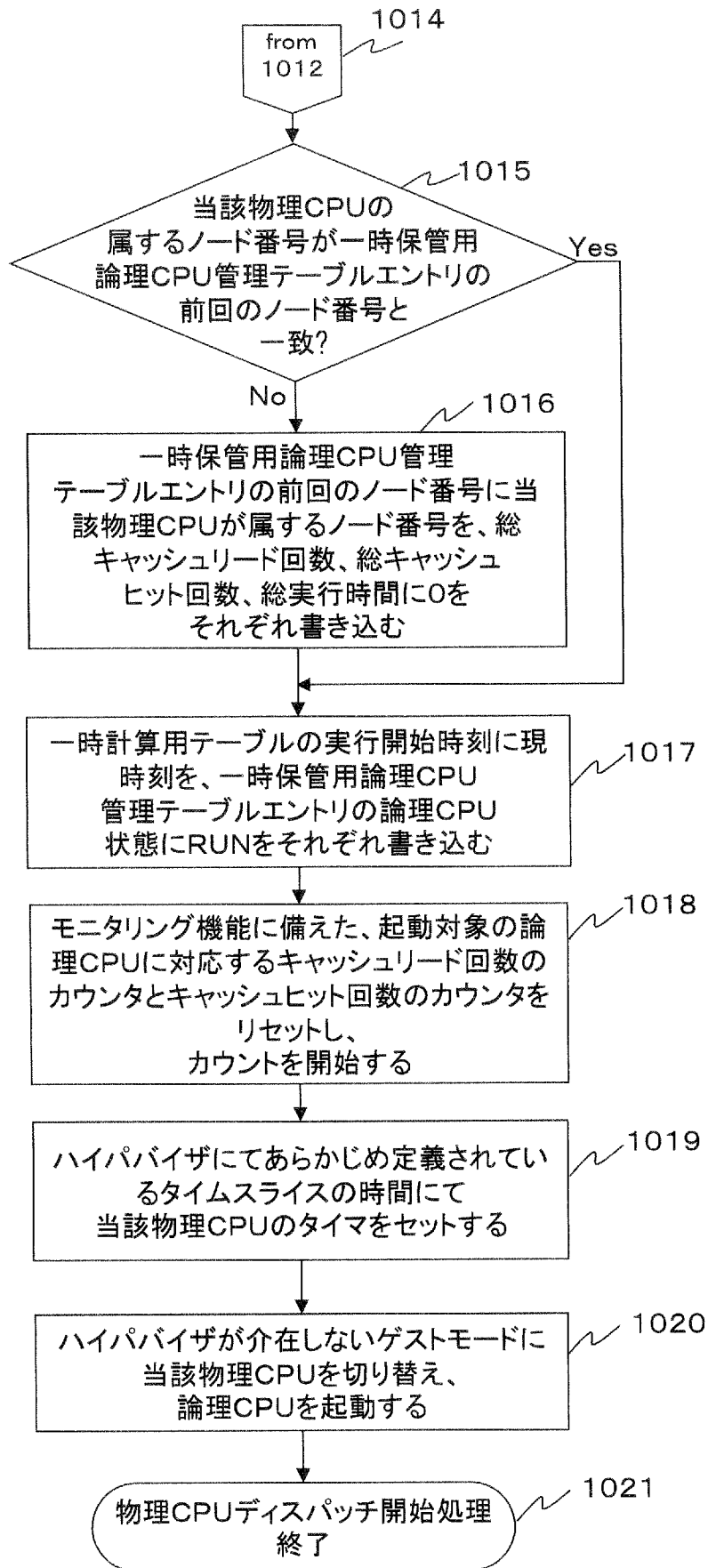
[図16A]



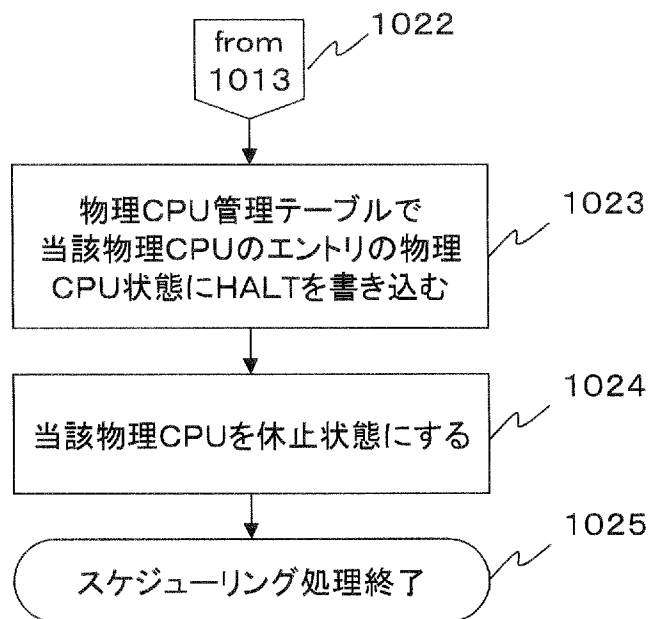
[図16B]



[図16C]



[図16D]



[図17A]

400 共有論理CPU管理テーブル

401	402	403	404	405	406	407	408	409	410
エントリ番号	LPAR番号	論理CPU番号	論理CPU状態	前回ノード番号	総キャッシュリード回数	総キャッシュヒット回数	総実行時間	ノード0の物理CPU実行時間	ノード1の物理CPU実行時間
1	1	0	2	-1	0	0	0	0	0

削除されたエントリ

500 一時保管用論理CPU管理テーブルエントリ

501	502	503	504	505	506	507	508	509
LPAR番号	論理CPU番号	論理CPU状態	前回ノード番号	総キャッシュリード回数	総キャッシュヒット回数	総実行時間	ノード0の物理CPU実行時間	ノード1の物理CPU実行時間
1	0	2	-1	0	0	0	0	0

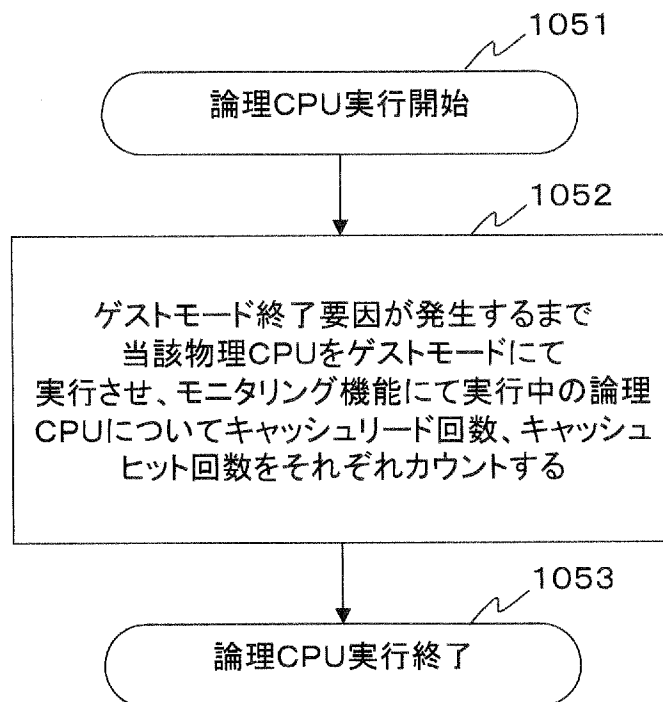
コピーされたエントリ

[図17B]

600 一時計算用テーブル

601	602	603	604	605
総キャッシュ ヒット率数	キャッシュ リード回数	キャッシュ ヒット回数	実行開始時刻	実行終了時刻
0	0	0	0→200	0

[図18]

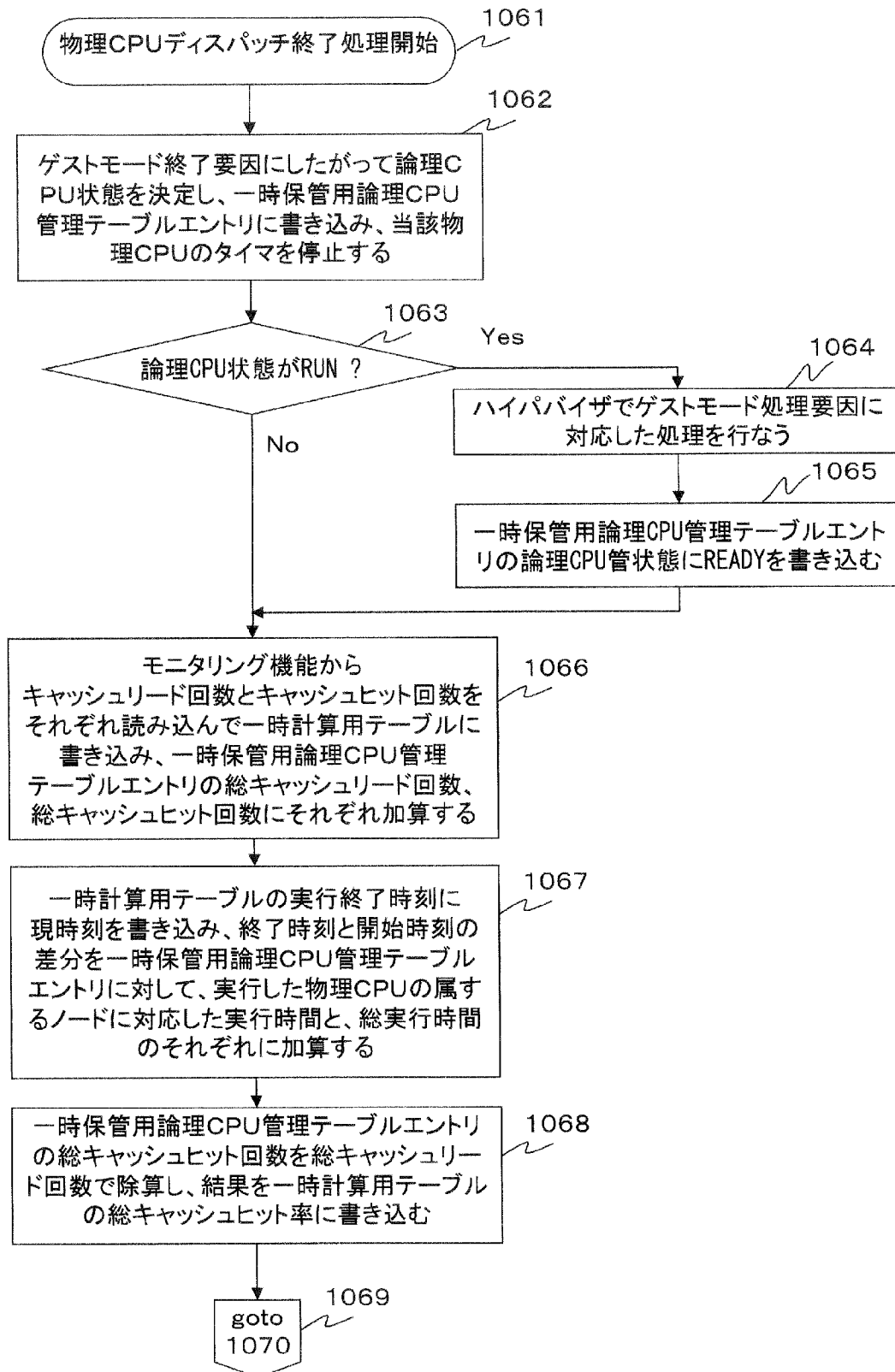


[図19]

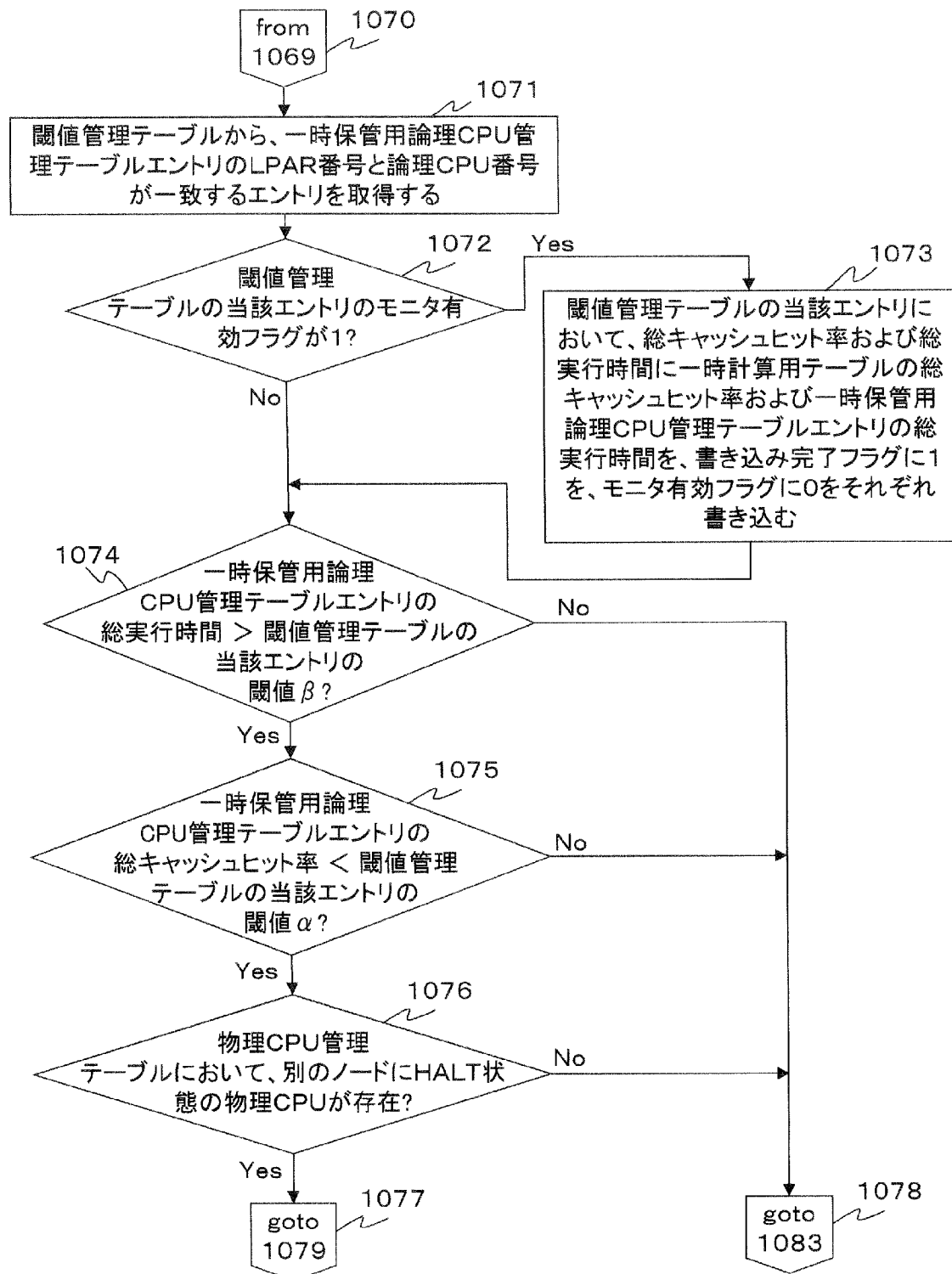
460 要因コードテーブル

要因コード	要因	論理CPU状態
0x000~0x00F	タイマ及びI/O割り込み	READY
0x010~0x01F	ハイパーバイザコール	RUN
0x020~0x02F	プログラム例外	RUN
0x030	HALT命令	HALT

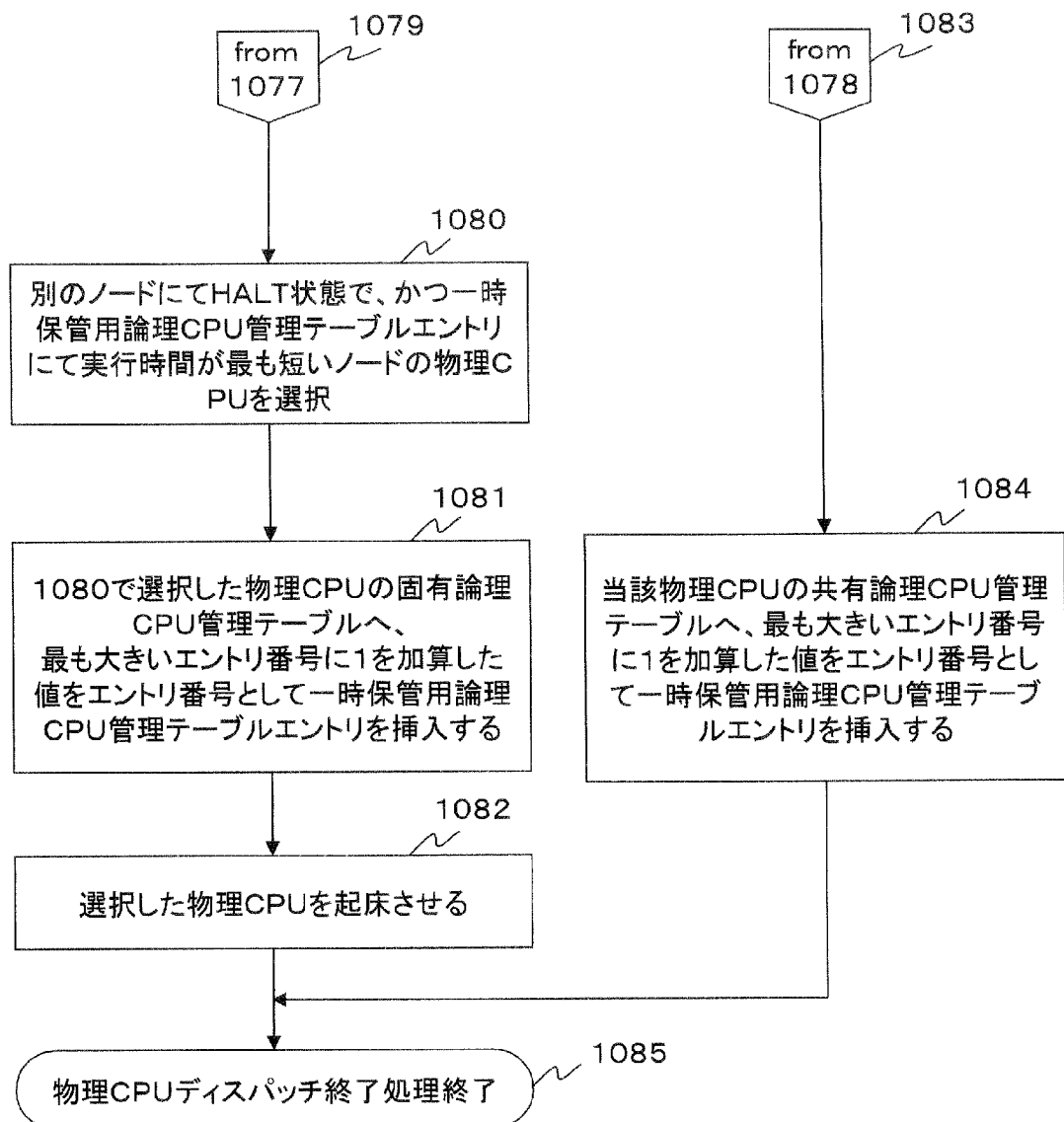
[図20A]



[図20B]



[図20C]



[図21A]

500 一時計保管用論理CPU管理テーブルエントリ

501	502	503	504	505	506	507	508	509
LPAR 番号	論理CPU 番号	論理CPU 状態	前回ノード 番号	総キャッシュ リード回数	総キャッシュ ヒット回数	総実行時間	ノード0の物理 CPU実行時間	ノード1の物理 CPU実行時間
1	0	1→2	0	0→1000	0→300	0→800	0→800	0

[図21B]

600 一時計算テーブル

601	602	603	604	605
総キャッシュ ヒット率数	キャッシュ リード回数	キャッシュ ヒット回数	実行開始時刻	実行終了時刻
0→0.3	0→1000	0→300	200	0→1000

[図21C]

400 共有論理CPU管理テーブル

401 エントリ 番号	402 LPAR 番号	403 論理CPU 番号	404 論理CPU 状態	405 前回ノード 番号	406 総キャッシュ リード回数	407 総キャッシュ ヒット回数	408 総実行時間	409 ノード0の物理 CPU実行時間	410 ノード1の物理 CPU実行時間
1	1	0	2	0	1000	300	800	800	0

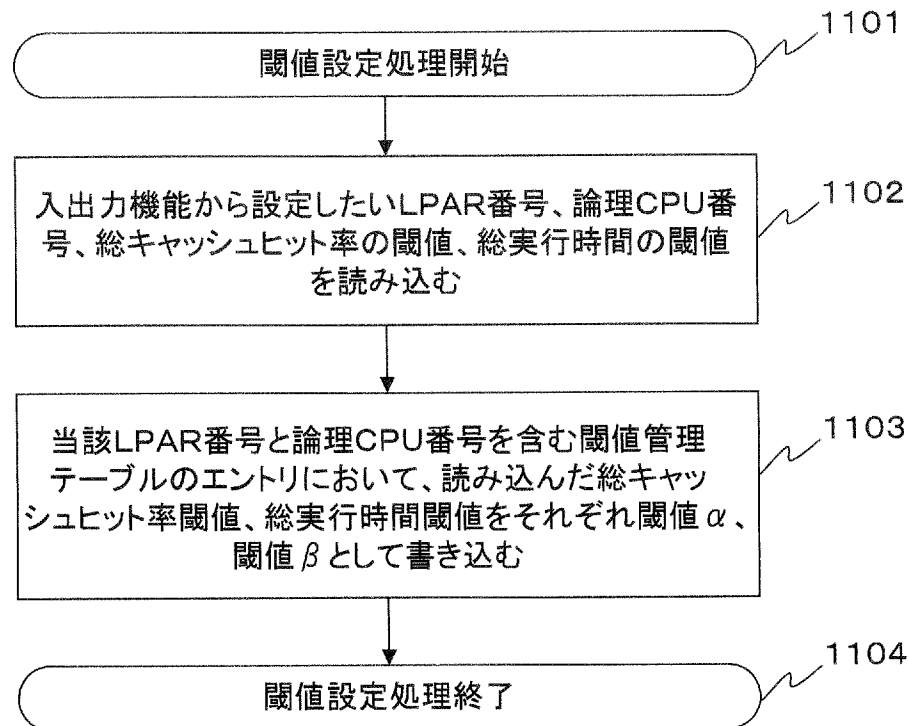
← 挿入されたエントリ

500 一時保管用論理CPU管理テーブルエントリ

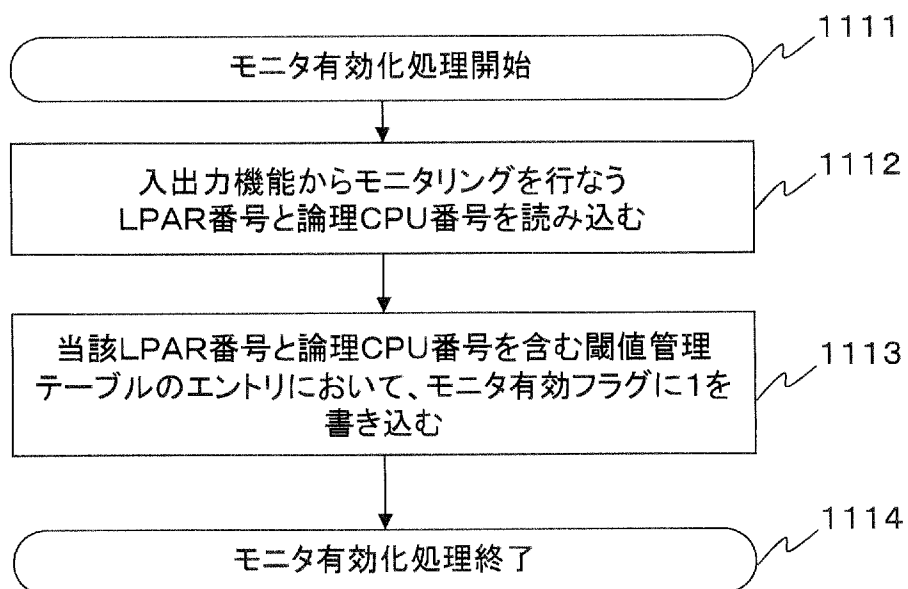
501 LPAR 番号	502 論理CPU 番号	503 論理CPU 状態	504 前回ノード 番号	505 総キャッシュ リード回数	506 総キャッシュ ヒット回数	507 総実行時間	508 ノード0の物理 CPU実行時間	509 ノード1の物理 CPU実行時間
1	0	2	0	1000	300	800	800	0

← 削除されたエントリ

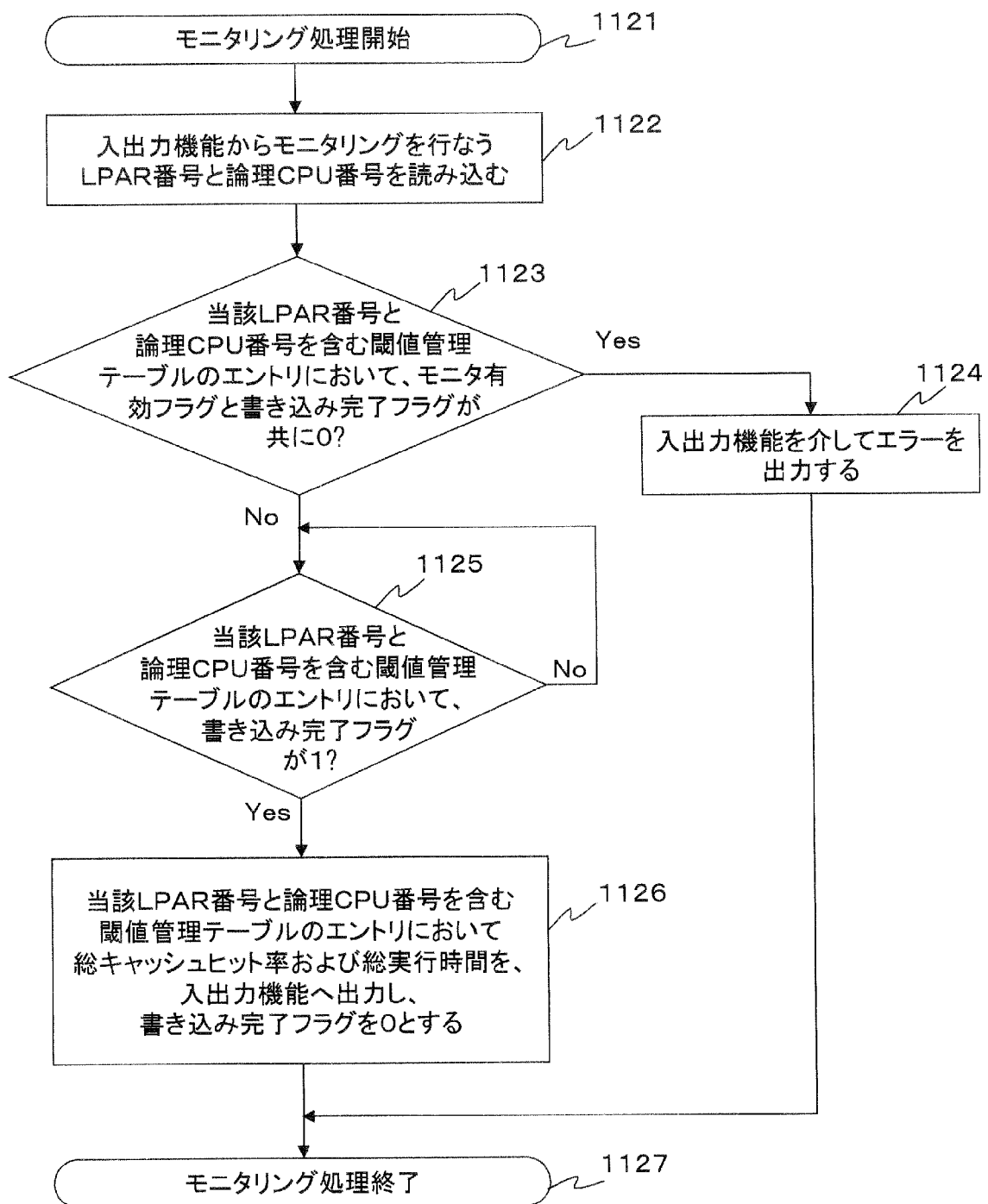
[図22]



[図23]



[図24]



[図25A]

700 閾値管理テーブル

701	702	703	704	705	706	707	708
LPAR番号	論理 CPU番号	閾値 α	閾値 β	モニタ有効 フラグ	書き込み完了 フラグ	総キャッシュ ヒット率	総実行 時間
1	0	0.5	2000	0→1	0	0	0
2	0	0.5	2000	0	0	0	0

[図25B]

700 閾値管理テーブル

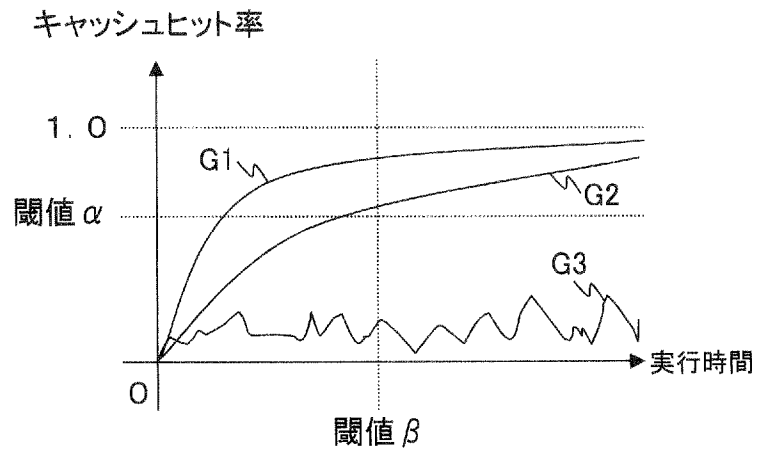
LPAR番号	論理 CPU番号	閾値 α	閾値 β	モニタ有効 フラグ	書き込み完了 フラグ	総キャッシュ ヒット率	総実行時 間
1	0	0.5	2000	1→0	0→1	0→0.4	0→500
2	0	0.5	2000	0	0	0	0

[図25C]

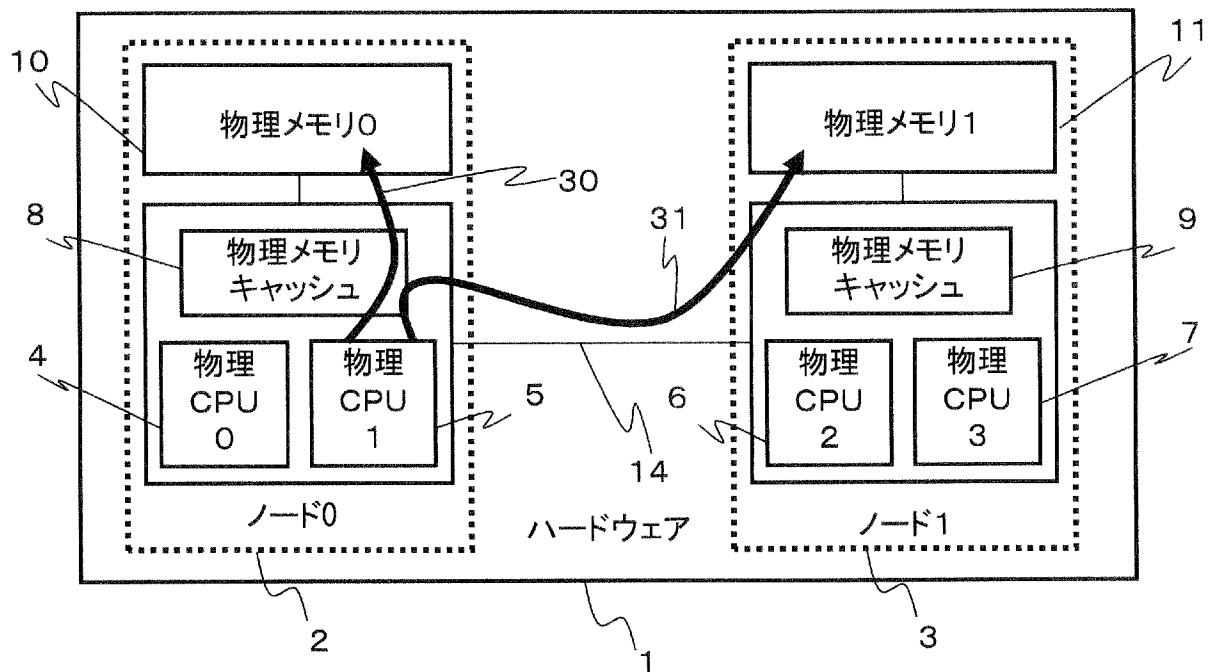
700 閾値管理テーブル

LPAR番号	論理CPU番号	閾値 α	閾値 β	モニタ有効フラグ	書き込み完了フラグ	総キャッシュヒット率	総実行時間
1	0	0.5	2000	0	1→0	0.4	500
2	0	0.5	2000	0	0	0	0

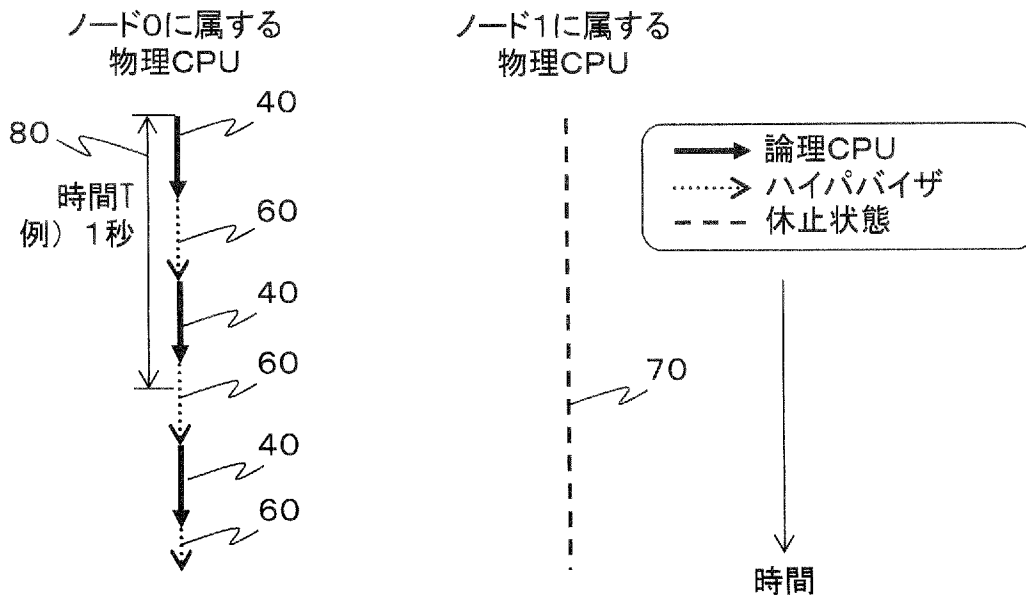
[図26]



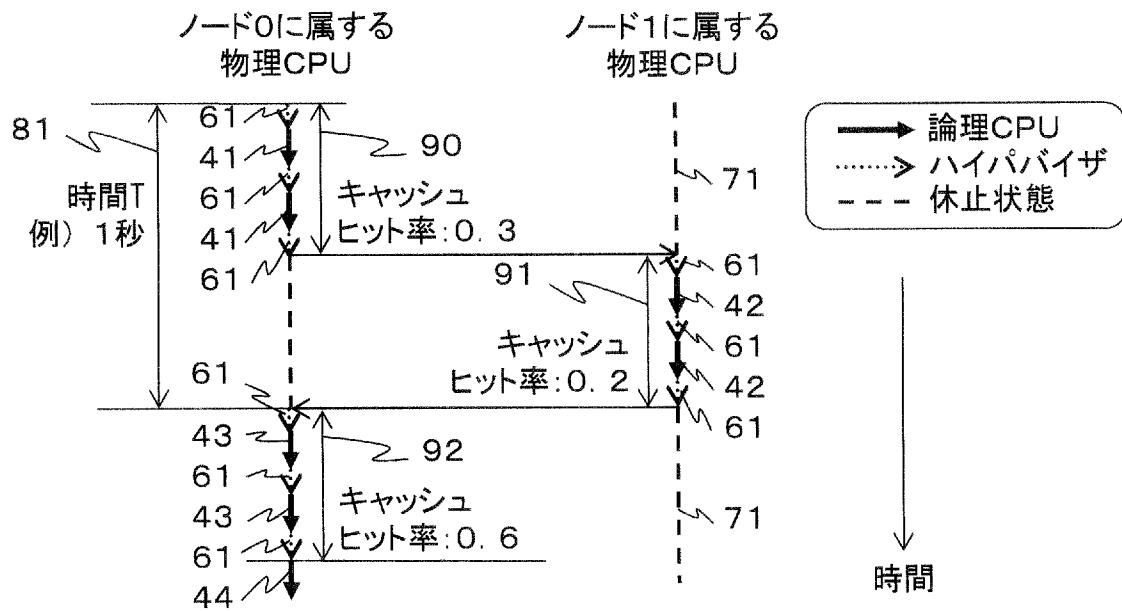
[図27]



[図28]



[図29]



[図30]

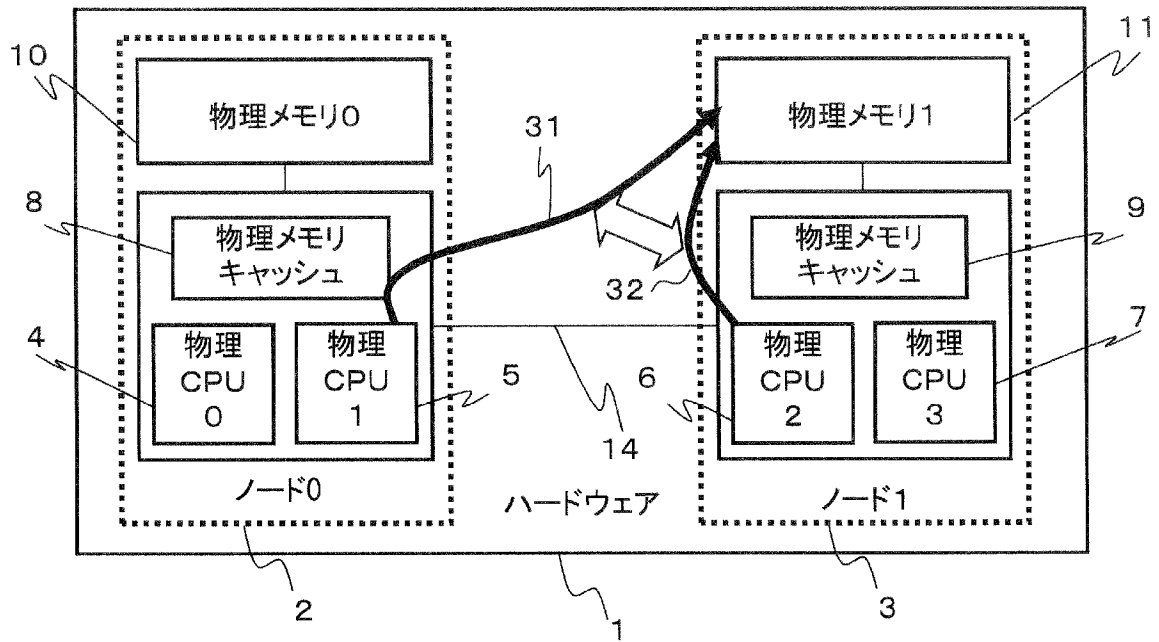
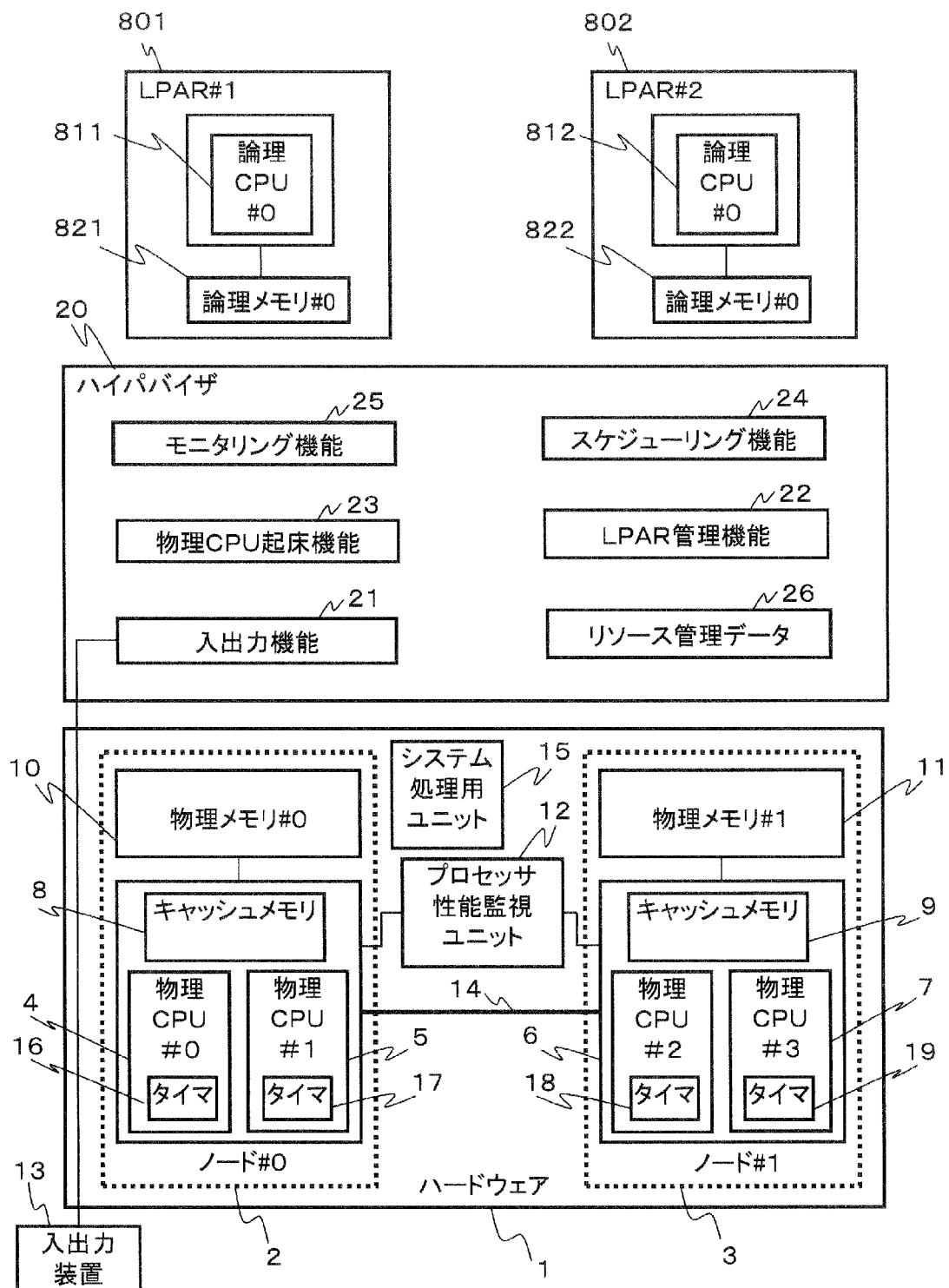


Figure 1 is a scheduling diagram illustrating the execution of logical CPUs (a, b, c) on physical CPUs of Node 0 and Node 1 over time T. The diagram is divided into three main sections: Node 0's physical CPU, Node 1's physical CPU, and the logical CPU queue.

- Node 0's physical CPU:** Shows the execution of tasks a, c, b, and c in sequence. Tasks are labeled with numbers 50, 52, 54, and 57. Dotted lines indicate task boundaries, with labels 62, 64, and 67 marking specific points.
- Node 1's physical CPU:** Shows the execution of tasks b, a, c, and a in sequence. Tasks are labeled with numbers 51, 53, 55, and 56. Dotted lines indicate task boundaries, with labels 63, 65, and 66 marking specific points.
- Logical CPU Queue:** Shows the state of the queue at various time points. The queue contains tasks a, b, and c, and the hypervisor (ハイパバイザ). The queue state is updated at the points marked by the dotted lines in the physical CPU sections.

The diagram also includes a legend for the logical CPUs (a, b, c) and the hypervisor (ハイパバイザ), and a vertical arrow indicating the progression of time (時間).

[図32]



INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2013/057141

A. CLASSIFICATION OF SUBJECT MATTER

G06F9/50(2006.01) i, G06F9/46(2006.01) i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F9/50, G06F9/46

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Jitsuyo Shinan Toroku Koho	1996-2013
Kokai Jitsuyo Shinan Koho	1971-2013	Toroku Jitsuyo Shinan Koho	1994-2013

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 2009-205696 A (Hitachi, Ltd.), 10 September 2009 (10.09.2009), paragraphs [0014] to [0016], [0026]; fig. 1, 2, 12 (Family: none)	1-10
Y	JP 6-12395 A (Canon Inc.), 21 January 1994 (21.01.1994), paragraphs [0007], [0008], [0015]; fig. 6 (Family: none)	1-10
Y	JP 2006-59052 A (Hitachi, Ltd.), 02 March 2006 (02.03.2006), paragraphs [0006], [0007] (Family: none)	4, 5, 9, 10



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

“A” document defining the general state of the art which is not considered to be of particular relevance

“E” earlier application or patent but published on or after the international filing date

“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

“O” document referring to an oral disclosure, use, exhibition or other means

“P” document published prior to the international filing date but later than the priority date claimed

“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&” document member of the same patent family

Date of the actual completion of the international search
29 March, 2013 (29.03.13)

Date of mailing of the international search report
09 April, 2013 (09.04.13)

Name and mailing address of the ISA/
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

A. 発明の属する分野の分類 (国際特許分類 (I P C))

Int.Cl. G06F9/50(2006.01)i, G06F9/46(2006.01)i

B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (I P C))

Int.Cl. G06F9/50, G06F9/46

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	1 9 2 2 - 1 9 9 6 年
日本国公開実用新案公報	1 9 7 1 - 2 0 1 3 年
日本国実用新案登録公報	1 9 9 6 - 2 0 1 3 年
日本国登録実用新案公報	1 9 9 4 - 2 0 1 3 年

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)

C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求項の番号
Y	JP 2009-205696 A (株式会社日立製作所) 2009.09.10, 段落 (0 0 1 4) - (0 0 1 6), (0 0 2 6), 図 1, 図 2, 図 1 2 (ファミリーなし)	1-10
Y	JP 6-12395 A (キヤノン株式会社) 1994.01.21, 段落 (0 0 0 7), (0 0 0 8), (0 0 1 5), 図 6 (ファミリーなし)	1-10
Y	JP 2006-59052 A (株式会社日立製作所) 2006.03.02, 段落 (0 0 0 6), (0 0 0 7) (ファミリーなし)	4, 5, 9, 10

C 欄の続きにも文献が列挙されている。

パテントファミリーに関する別紙を参照。

* 引用文献のカテゴリー

「A」特に関連のある文献ではなく、一般的技術水準を示すもの
「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す)
「O」口頭による開示、使用、展示等に言及する文献
「P」国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Y」特に関連のある文献であって、当該文献と他の 1 以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「&」同一パテントファミリー文献

国際調査を完了した日

2 9 . 0 3 . 2 0 1 3

国際調査報告の発送日

0 9 . 0 4 . 2 0 1 3

国際調査機関の名称及びあて先

日本国特許庁 (I S A / J P)

郵便番号 1 0 0 - 8 9 1 5

東京都千代田区霞が関三丁目 4 番 3 号

特許庁審査官 (権限のある職員)

鈴木 修治

5 B

3 5 6 0

電話番号 0 3 - 3 5 8 1 - 1 1 0 1 内線 3 5 4 5