(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6322968号
(P6322968)
(45)発行日　平成30年5月16日(2018.5.16)　(24)登録日　平成30年4月20日(2018.4.20)

(51)Int.Cl.　F I
G06F 9/50 (2006.01)　G06F 9/46 462A

請求項の数 6 (全 19 頁)

(21)出願番号　特願2013-239098(P2013-239098)
(22)出願日　平成25年11月19日(2013.11.19)
(65)公開番号　特開2015-99494(P2015-99494A)
(43)公開日　平成27年5月28日(2015.5.28)
審査請求日　平成28年10月14日(2016.10.14)

(73)特許権者　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(74)代理人　100109313
弁理士　机　昌彦
(74)代理人　100124154
弁理士　下坂　直樹
(72)発明者　菊池　康祐
東京都港区芝五丁目７番１号
日本電気株式会社内

審査官　原　忠

最終頁に続く

(54)【発明の名称】情報処理装置、情報処理方法およびプログラム

(57)【特許請求の範囲】
【請求項１】
アプリケーションの実行特性に影響を及ぼす、ループ構造を含む少なくとも一部のプログラムコードによる記述である特定記述が当該アプリケーションに含まれる場合、前記特定記述によって呼び出される命令または関数の実行頻度に関する付加情報を生成する特定記述解析手段と、
前記付加情報に基づいて、前記命令または関数の実行に適した計算リソースを判断すると共に、当該計算リソースを前記アプリケーションの実行に割り当てる計算リソース割当手段とを備え、
前記特定記述解析手段は、前記特定記述によって呼び出される命令または関数に関する前記ループ構造における、ループの繰り返し回数とループ構造の深さとを含む付加情報を生成し、
前記計算リソース割当手段は、前記ループの繰り返し回数と前記ループ構造の深さのうち少なくともいずれかがそれぞれの閾値より大きい場合、前記アプリケーションの実行を、前記命令または関数の実行に適した計算リソースに割り当てる
情報処理装置。
【請求項２】
前記特定記述解析手段は、前記アプリケーションの実行プログラム、または当該実行プログラムがコンパイルされて生成される実行バイナリが逆アセンブルされた結果に基づいて、前記特定記述の有無を判断する

請求項１記載の情報処理装置。
【請求項３】
前記特定記述解析手段は、前記アプリケーションにおいて集団通信関数を呼び出す記述を特定記述として抽出し、
前記計算リソース割当手段は、前記集団通信関数の繰り返し回数が閾値より大きい場合、前記アプリケーションの実行を、帯域幅が所定値より高い通信ネットワークを用いて構成された計算リソースに割り当てる
請求項１または請求項２記載の情報処理装置。
【請求項４】
前記特定記述解析手段は、前記アプリケーションにおいてランダムアクセスでのファイルへの読み出し命令または書き込み命令を呼び出す記述を特定記述として抽出し、
前記計算リソース割当手段は、前記読み出し命令または前記書き込み命令の繰り返し回数が閾値より大きい場合、前記アプリケーションの実行を、ランダムアクセスに適した記憶装置を用いて構成された計算リソースに割り当てる
請求項１または請求項２記載の情報処理装置。
【請求項５】
コンピュータによって、
アプリケーションの実行特性に影響を及ぼす、ループ構造を含む少なくとも一部のプログラムコードによる記述である特定記述が当該アプリケーションに含まれる場合、前記特定記述によって呼び出される命令または関数の実行頻度に関する付加情報を生成し、
前記付加情報に基づいて、前記命令または関数の実行に適した計算リソースを判断すると共に、当該計算リソースを前記アプリケーションの実行に割り当て、
前記付加情報は、前記特定記述によって呼び出される命令または関数に関する前記ループ構造における、ループの繰り返し回数とループ構造の深さとを含み、
前記計算リソースは、前記ループの繰り返し回数と前記ループ構造の深さのうち少なくともいずれかがそれぞれの閾値より大きい場合、前記アプリケーションの実行が割り当てられる
情報処理方法。
【請求項６】
アプリケーションの実行特性に影響を及ぼす、ループ構造を含む少なくとも一部のプログラムコードによる記述である特定記述が当該アプリケーションに含まれる場合、前記特定記述によって呼び出される命令または関数の実行頻度に関する付加情報を生成する処理と、
前記付加情報に基づいて、前記命令または関数の実行に適した計算リソースを判断すると共に、当該計算リソースを前記アプリケーションの実行に割り当てる処理とを
コンピュータに実行させ、
前記付加情報は、前記特定記述によって呼び出される命令または関数に関する前記ループ構造における、ループの繰り返し回数とループ構造の深さとを含み、
前記計算リソースは、前記ループの繰り返し回数と前記ループ構造の深さのうち少なくともいずれかがそれぞれの閾値より大きい場合、前記アプリケーションの実行が割り当てられる
プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、アプリケーションの実行に適した計算リソースに、そのアプリケーションの実行を割り当てる技術に関する。
【背景技術】
【０００２】
一般に、計算ノード間のデータ通信は通信ネットワーク（以降、単に「ネットワーク」

とも称する）を介して実施され、ファイルＩ／Ｏ（Ｉｎｐｕｔ／Ｏｕｔｐｕｔ：入出力）はストレージディスクに対して行われる。

【０００３】

データ通信やファイルＩ／Ｏの実行仕様、すなわち特性に応じて、データ通信やファイルＩ／Ｏの実行に適するネットワークやストレージディスクは異なる。これらネットワークやストレージディスクといった要素を用いてシステムを構築する場合、複数の上記特性に適した要素を多く用いると、様々な特性を備えたアプリケーションの実行に適したシステムを構築できる（、すなわちアプリケーションの実行性能を向上できる）という利点がある。

【０００４】

一方、複数の特性に適した要素を利用したシステムは、一部の特性のみに適した要素を利用したシステムと比較して、一般に構築費用が高額になるという問題がある。そこで、構築費用を抑えながら、複数の特性に適したシステムを構築することが考えられている。例えば、システムを構成するすべての要素を、複数の特性に対応可能な要素にて構築するのではなく、システムを構成する要素をいくつかのグループに分割すると共に、グループ間で異なる特性に対応する要素を採用することが考えられる。

【０００５】

しかしながら、上記方法にてシステムを構築するには、エンドユーザが、アプリケーションおよびシステムを構成する各要素の特性を把握すると共に、手動にてシステム内におけるグループを使い分ける作業が必要となる。

【０００６】

上記作業は、計算機の知識を必要とし、また煩雑であるため、単に計算結果のみを必要とする、計算機に精通していないエンドユーザには困難である。一方で、計算ノード間のネットワークにおける通信や、ディスクストレージへのファイルＩ／Ｏは、アプリケーションの実行性能に大きく影響するので、これらの実行に適したシステムでの実行が望まれる。

【０００７】

上記技術に関連して、例えば、特許文献１は、命令セット及び構成の異なるプロセッサエレメントを複数備えたヘテロジニアス・マルチプロセッサシステムを開示する。このマルチプロセッサシステムは、特定のプロセッサエレメントのリソースが不足することを防ぐことにより、マルチプロセッサシステム全体の処理を向上させる手段を備える。

【０００８】

また、特許文献２は、ソースプログラムに基づいてプログラムの特性を判別することにより、スケジューリングに活かす方式を開示する。

【先行技術文献】

【特許文献】

【０００９】

【特許文献１】特許第４９３６５１７号公報

【特許文献２】特開２００３－２７１４０５号公報

【発明の概要】

【発明が解決しようとする課題】

【００１０】

上述のように、システムを構成する要素をいくつかのグループに分割すると共に、グループ間で特性の異なる要素を採用するようにシステムを構築するには、計算機の知識および煩雑な作業が必要となるため、一般のエンドユーザには実施困難であるという課題がある。

【００１１】

また、上記特許文献１に開示される手段は、異種のリソースにおいてタスクを効率よく動作させるものの、プロセッサの処理のみに着目しているため、計算ノード間のネットワークにおける通信や、ディスクストレージへのファイルＩ／Ｏに適用することはできない

。
【００１２】
　また、上記特許文献２には、計算機の知識および煩雑な作業を必要とせずに、アプリケーションの実行性能を向上する技術は開示されていない。
【００１３】
　本願発明は、上記課題を鑑みてなされたものであり、構築費用を低減できると共に、エンドユーザの手間を不要としながらアプリケーションの実行性能を向上することができる情報処理装置等を提供することを主要な目的とする。
【課題を解決するための手段】
【００１４】
　本発明の第１の情報処理装置は、アプリケーションの実行特性に影響を及ぼす特定記述が当該アプリケーションに含まれる場合、前記特定記述によって呼び出される命令または関数の実行頻度に関する付加情報を生成する特定記述解析手段と、前記付加情報に基づいて、前記命令または関数の実行に適した計算リソースを判断すると共に、当該計算リソースを前記アプリケーションの実行に割り当てる計算リソース割当手段とを備える。
【００１５】
　本発明の第１の情報処理方法は、コンピュータによって、アプリケーションの実行特性に影響を及ぼす特定記述が当該アプリケーションに含まれる場合、前記特定記述によって呼び出される命令または関数の実行頻度に関する付加情報を生成し、前記付加情報に基づいて、前記命令または関数の実行に適した計算リソースを判断すると共に、当該計算リソースを前記アプリケーションの実行に割り当てる。
【００１６】
　なお同目的は、上記の各構成を有する情報処理装置または情報処理方法を、コンピュータによって実現するコンピュータ・プログラム、およびそのコンピュータ・プログラムが格納されている、コンピュータ読み取り可能な記憶媒体によっても達成される。
【発明の効果】
【００１７】
　本願発明によれば、構築費用を低減できると共に、エンドユーザの手間をかけることなくアプリケーションの実行性能を向上することができるという効果が得られる。
【図面の簡単な説明】
【００１８】
【図１】本発明の第１の実施形態に係る情報処理システムの構成の概要を示す図である。
【図２】本発明の第１の実施形態に係る情報処理システムの構成を示すブロック図である。
【図３】本発明の第１の実施形態に係る情報処理システムにおいて、同時に全てのプロセス間でデータ交換が行われる例を示す図である。
【図４】本発明の第１の実施形態に係る情報処理システムにおいて、プロセス間でリング状にデータ交換が行われる例を示す図である。
【図５】情報処理システムにおいてバイセクショナルバンド幅が高いネットワークトポロジによって全てのノードを構成した場合の動作を示すフローチャートである。
【図６】情報処理システムにおいてバイセクショナルバンド幅が低いネットワークトポロジによって全てのノードを構成した場合の動作を示すフローチャートである。
【図７】本発明の第１の実施形態に係る情報処理システムにおいてアプリケーションを実行するシステムを振り分ける動作を示すフローチャートである。
【図８】アプリケーションプログラムの記述の例を示す図である。
【図９】アプリケーションプログラムの記述の例を示す図である。
【図１０】アプリケーションプログラムの記述の例を示す図である。
【図１１】情報処理システムにおいてＳＡＳ（Ｓｅｒｉａｌ　Ａｔｔａｃｈｅｄ　ＳＣＳＩ）ディスクによって全てのノードを構成した場合の動作を示すフローチャートである。
【図１２】情報処理システムにおいてＳＡＴＡ（Ｓｅｒｉａｌ　ＡＴＡ）ディスクによっ

て全てのノードを構成した場合の動作を示すフローチャートである。
【図１３】本発明の第１の実施形態に係る情報処理システムにおいてアプリケーションをＳＡＳディスクによるシステムとＳＡＴＡディスクによるシステムとに振り分ける動作を示すフローチャートである。
【図１４】本発明の第２の実施形態に係る情報処理システムの構成を示すブロック図である。
【図１５】アプリケーションプログラムの例の一部を示す図である。
【図１６】プリケーション実行バイナリを逆アセンブルした結果である命令リストを示す図である。
【図１７】本発明の第３の実施形態に係る情報処理システムの構成の概要を示す図である。
【図１８】本発明の各実施形態に係る情報処理システムを実現可能な情報処理装置のハードウエア構成を例示する図である。
【発明を実施するための形態】
【００１９】
以下、本発明の実施形態について図面を参照して詳細に説明する。
【００２０】
第１の実施形態
図１は、本発明の第１の実施形態に係る情報処理システム１００の構成の概要を示す図である。図１に示すように、情報処理システム１００は、ジョブスケジューラ１１０、計算システム120_1、120_2、・・・、120_nを備える。アプリケーションの実行要求は、エンドユーザからジョブスケジューラに送られる。このとき、エンドユーザは、アプリケーションの実行に必要となる各計算システム内のＣＰＵ（Ｃｅｎｔｒａｌ Ｐｒｏｃｅｓｓｉｎｇ Ｕｎｉｔ）コアの数やメモリ容量なども要求する。本実施形態において、アプリケーションとは、情報処理システム１００において実行される特定の機能を実現可能なソフトウエアプログラム（コンピュータプログラム）を表す。
【００２１】
情報処理システム１００は、ＨＰＣ（Ｈｉｇｈ Ｐｅｒｆｏｒｍａｎｃｅ Ｃｏｍｐｕｔｉｎｇ）の分野に属する。
【００２２】
一般的なＨＰＣの分野に属するシステムでは、複数のユーザによりシステムを共有すると共に、アプリケーションの実行量を最大化するためにジョブスケジューラと呼ばれるソフトウェアが用いられる。
【００２３】
ジョブスケジューラは、エンドユーザから要求されたＣＰＵコアの数やメモリ容量を備えた計算システムを確保できた段階で、要求されたアプリケーションを、上記確保した計算システムにおいて実行する。このように、ジョブスケジューラによってアプリケーションを実行する計算システムの選択は自動的に行われる。
【００２４】
ここで、ＨＰＣ分野では、アプリケーションを実現可能なアプリケーションプログラムを記述するのに用いられるプログラミング言語には、一部の種類（言語）が多く使われる。また、複数の計算システムに処理を分割し並列に処理する分散並列処理を行うためのライブラリのＡＰＩ（Ａｐｐｌｉｃａｔｉｏｎ Ｐｒｏｇｒａｍｍｉｎｇ Ｉｎｔｅｒｆａｃｅ）にも、一部の種類が多く使われる。このようなプログラミング言語内の特定記述（詳細は後述する）や、ライブラリのＡＰＩは、それらに使用される種類毎に実行に適するシステムが異なるという特徴がある。
【００２５】
また、ＨＰＣ分野のアプリケーションは、一般的に一部の処理を繰り返し実行するという特徴がある。繰り返し実行はプログラミング言語のループ構造によって記述される。
【００２６】

このように、実行に適する環境が異なるプログラム中の記述と、その記述が存在する場所のループ構造とを調べることにより、そのアプリケーションの実行に適する特性を高い精度で予測することが可能である。また、予測した結果を、ジョブスケジューラに入力することにより、エンドユーザによる手動の操作を必要とすることなく、アプリケーションの実行に適するシステムによりアプリケーションを実行することが自動で可能となる。

【００２７】

以下に、本実施形態に係る情報処理システム１００の詳細を説明する。

【００２８】

図２は、本発明の第１の実施形態に係る情報処理システム１００の構成を示すブロック図である。図２に示すように、情報処理システム１００は、コンパイラ２１０、ジョブスケジューラ２２０、システム２３０，２４０を備える。

【００２９】

コンパイラ２１０は、コンパイラ通常機能部２１１、特定記述解析部２１２および付加情報生成部２１３を備える。ジョブスケジューラ２２０は、アプリケーション実行受付部２２１、反復回数閾値設定部２２２、ループ構造深さ閾値設定部２２３、付加情報解析部２２４および計算リソース割当部２２５を備える。

【００３０】

コンパイラ２１０は、アプリケーション実行要求３１０を出力する。アプリケーション実行要求３１０は、アプリケーション実行バイナリ３１１、反復回数付加情報３１２およびループ構造深さ付加情報３１３を含む。アプリケーション実行要求３１０は、エンドユーザによって、ジョブスケジューラ２２０にアプリケーションの実行を要求するときに渡される。

【００３１】

コンパイラ２１０は、アプリケーションプログラム２００を受け取ると、コンパイラ通常機能部２１１において、通常のコンパイル機能を実施する。コンパイラ通常機能部２１１は、アプリケーションプログラム２００に基づいて、アプリケーション実行バイナリ３１１を生成する。続いて、コンパイラ通常機能部２１１は、アプリケーションプログラム２００を特定記述解析部２１２に送る。

【００３２】

特定記述解析部２１２は、アプリケーションプログラム２００において特定記述が記述されている箇所を調べる。特定記述とは、アプリケーションプログラムを構成する記述のうち、当該アプリケーションプログラムの実行特性に影響を及ぼす少なくとも一部のプログラムコードによる記述である。特定記述解析部２１２は、特定記述が見つかると、それが記述されている箇所のループ構造の深さと、ループの繰り返し回数を調べる。

【００３３】

特定記述解析部２１２は、調べた結果を付加情報生成部２１３に送る。付加情報生成部２１３は、ループの繰り返し回数を反復回数付加情報３１２、ループ構造の深さをループ構造深さ付加情報３１３として生成する。なお、ループの繰り返し回数がアプリケーションプログラム２００において静的に与えられていない場合、付加情報生成部２１３は、それを識別することができる固有の値（例えば、「－１」）を反復回数付加情報３１２として生成する。

【００３４】

ジョブスケジューラ２２０は、計算リソース割当部２２５において、通常のジョブスケジューラと同様の機能を実施する。すなわち、計算リソース割当部２２５は、アプリケーション実行要求３１０にしたがって、アプリケーションを実行する計算システムを確保すると共に、確保した計算システムにおいてアプリケーションの実行を指示する。

【００３５】

また、ジョブスケジューラ２２０は、反復回数閾値設定部２２２、ループ構造深さ閾値設定部２２３において、プログラム中の特定記述が実行される傾向を判断するのに利用する閾値を保持する。閾値には、アプリケーションプログラム２００における、ループ構造

の繰り返し回数との比較に使われる閾値と、ループ構造の深さとの比較に使われる閾値とがある。ループ構造の繰り返し回数との比較に使われる閾値は、反復回数閾値設定部２２２に保持される。ループ構造の深さとの比較に使われる閾値は、ループ構造深さ閾値設定部２２３に保持される。閾値は、システム管理者によってあらかじめジョブスケジューラ２２０に保持されてもよい。

【００３６】

ジョブスケジューラ２２０は、アプリケーション実行受付部２２１において、アプリケーション実行要求３１０を受け取る。アプリケーション実行受付部２２１は、アプリケーション実行要求３１０に含まれる反復回数付加情報３１２とループ構造深さ付加情報３１３を付加情報解析部２２４に渡す。また、アプリケーション実行受付部２２１は、アプリケーション実行要求３１０を、計算リソース割当部２２５に渡す。

【００３７】

付加情報解析部２２４は、まず、受け取った反復回数付加情報３１２と、反復回数閾値設定部２２２に設定された値とを比較する。付加情報解析部２２４は、反復回数閾値設定部２２２に設定された値より、受け取った反復回数付加情報３１２が大きい場合は、アプリケーションプログラム２００は、「特定記述の実行が多い」という特性に当てはまると判断する。

【００３８】

ここで、反復回数付加情報３１２に、アプリケーションプログラム２００においてループの繰り返し回数が静的に与えられていないことを識別する固有の値が設定されていた場合は、付加情報解析部２２４は、上記比較は行わず、ループ構造深さに関する比較を行う。すなわち、付加情報解析部２２４は、受け取ったループ構造深さ付加情報３１３と、ループ構造深さ閾値設定部２２３に設定された値とを比較する。

【００３９】

ループ構造深さ閾値設定部２２３に設定された値よりループ構造深さ付加情報３１３が大きい場合、付加情報解析部２２４は、アプリケーションプログラム２００は、「特定記述の実行が多い」という特性に当てはまると判断する。付加情報解析部２２４は、判断結果を、計算リソース割当部２２５に渡す。

【００４０】

計算リソース割当部２２５は、アプリケーション実行受付部２２１から渡されたアプリケーション実行要求３１０と、付加情報解析部２２４から渡された判断結果に基づいて、アプリケーションを実行するシステムを判断する。計算リソース割当部２２５は、特定記述の実行が多いという特性を持つアプリケーション実行に対して、特定記述の実行に適したシステム２３０を割り当てる。一方、計算リソース割当部２２５は、特定記述の実行が多いという特性を持たないアプリケーションの実行に対しては、特定記述の実行に適さないシステム２４０を割り当てる。

【００４１】

以上のように、情報処理システム１００は、アプリケーションの実行に適したシステムを判断すると共に、当該システムにアプリケーションの実行を振り分ける。

【００４２】

次に、具体的な例を用いて、上記構成の詳細について説明する。

【００４３】

ここで、ＭＰＩ（Ｍｅｓｓａｇｅ　Ｐａｓｓｉｎｇ　Ｉｎｔｅｒｆａｃｅ）を利用したプログラムに着目する。ＨＰＣの分野では、計算量の増加に伴い、複数の計算機に処理を分散する分散並列計算を行うことがよく使われる手法として存在する。分散並列計算を行うにはいくつかの手法が存在し、その中でもＭＰＩと呼ばれるメッセージパッシングＡＰＩの標準規格によって分散並列計算を行うことが今日では一般的となっている。すなわち、本実施形態においてＭＰＩを利用したアプリケーションプログラムを対象にすることは、ＨＰＣ分野のアプリケーションにおいて大きな割合を占めることになる。

【００４４】

そこで、以下に、ＭＰＩを利用したアプリケーションプログラムを本実施形態において実行する場合の情報処理システム１００の動作について説明する。

【００４５】

ＭＰＩは、分散並列計算をメッセージパッシングによって実現するＡＰＩであり、ＭＰＩが備える通信ＡＰＩの種類は、大きく分けて「１対１通信系ＡＰＩ」と、「集団通信系ＡＰＩ」に分類される。例えば、「集団通信系ＡＰＩ」の一関数である”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”は、図３に示すように、同時に全てのプロセス（０～３）間においてデータ交換が行われる場合に使われる。一方、「１対１通信系ＡＰＩ」の一関数である”MPI_Sendrecv()”は、図４に示すように、プロセス（０～３）間でリング状にデータ交換をする場合によく使われる。ネットワークのバイセクショナルバンド幅（バイセクションバンド幅：２分割帯域幅）としては、集団通信系ＡＰＩの実行には、１対１通信系ＡＰＩの実行に比べて、高いバイセクショナルバンド幅が要求される。バイセクショナルバンド幅とは、ネットワークを２つに分割し、その間にある全リンクのバンド幅を合計したものである。ここでは、例えば、バイセクショナルバンド幅が、所定の閾値より高いネットワークトポロジにより構成されるシステムを、「バイセクショナルバンド幅が高いシステム」と称する。また、バイセクショナルバンド幅が、所定の閾値より低いネットワークトポロジにより構成されるシステムを、「バイセクショナルバンド幅が低いシステム」と称する。

【００４６】

例えば、アプリケーションを実行するシステムを１０００台のノードを用いて構成することを考える。図５に示すように、バイセクショナルバンド幅が所定の閾値より高いネットワークトポロジによって１０００ノード全てを構成した場合、どの計算ノードにてアプリケーションを実行したとしても、ＭＰＩが備える集団通信系ＡＰＩは、性能低下することなく実行される。

【００４７】

しかしながら、一般に、バイセクショナルバンド幅が高いシステムは、バイセクショナルバンド幅が低いシステムに比べて、構築費用が高いという特徴がある。

【００４８】

一方、図６に示すように、バイセクショナルバンド幅が低いネットワークトポロジによって１０００ノード全てを構成すると、システムの構築費用を安くすることが可能となる反面、ＭＰＩが備える集団通信系ＡＰＩの実行性能が低下するという問題が生じる。

【００４９】

そこで、本実施形態では、図７に示すように、情報処理システム１００が、ＭＰＩを利用したアプリケーションプログラムを実行するシステムを振り分ける動作について説明する。すなわち、ジョブスケジューラは、ＭＰＩを利用したアプリケーションプログラムの特性を判断し、その判断結果に基づいて、バイセクショナルバンド幅が高いシステムと、バイセクショナルバンド幅が低いシステムに、プログラムの実行を振り分ける。

【００５０】

図８は、ＭＰＩが備える集団通信系ＡＰＩである関数”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”を、ループ構造の深さが「３」の位置（ａ）で呼び出すアプリケーションプログラム（ソースプログラム）の記述（プログラムコード）の一部の例を示す図である。図８に示すアプリケーションプログラムが、図２に示すコンパイラ２１０に供給されると、コンパイラ２１０のコンパイラ通常機能部２１１は、コンパイルを実施することによりアプリケーション実行バイナリ３１１を生成する。

【００５１】

コンパイラ通常機能部２１１はまた、アプリケーションプログラムを特定記述解析部２１２に送る。特定記述解析部２１２は、アプリケーションプログラムにおいて集団通信系ＡＰＩの有無を解析する。図８に示すように、当該アプリケーションプログラムでは、”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”を呼び出しているので、特定記述解析部２１２はそれを検出する。

【００５２】
続いて特定記述解析部２１２は、”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”が呼び出されている箇所におけるループ構造を解析する。図８に示す（ａ）の位置では、ループ構造の深さは「３」と識別できる。一方、ループの繰り返し回数は、「Ｎ１」、「Ｎ２」、「Ｎ３」と示され、静的に与えられていない。よって、特定記述解析部２１２は、ループの繰り返し回数が静的に与えられていないことを示す「－１」を解析結果とする。特定記述解析部２１２は、解析結果を付加情報生成部２１３に送る。ここでは、反復回数付加情報３１２として「－１」、ループ構造深さ付加情報３１３として「３」が、生成される。
【００５３】
上述のように、コンパイラ２１０によって生成されたアプリケーション実行バイナリ３１１、反復回数付加情報３１２、ループ構造深さ付加情報３１３は、アプリケーション実行要求３１０としてジョブスケジューラ２２０に渡される。なお、ここでは、ジョブスケジューラ２２０の反復回数閾値設定部２２２に「１０００」、ループ構造深さ閾値設定部２２３に「２」が、あらかじめシステム管理者によって設定されている。
【００５４】
ジョブスケジューラ２２０は、アプリケーション実行要求３１０をアプリケーション実行受付部２２１において受け取る。アプリケーション実行受付部２２１は、アプリケーション実行要求３１０を計算リソース割当部２２５に送る。また、アプリケーション実行受付部２２１は、アプリケーション実行要求３１０に含まれる反復回数付加情報３１２と、ループ構造深さ付加情報３１３を、付加情報解析部２２４に送る。
【００５５】
付加情報解析部２２４は、受け取った反復回数付加情報３１２と、ループ構造深さ付加情報３１３を、それぞれ、反復回数閾値設定部２２２に設定された値と、ループ構造深さ閾値設定部２２３に設定された値と比較する。そして、付加情報解析部２２４は、その比較の結果に基づいて、アプリケーションプログラムの特性を判断する。
【００５６】
ここでは、反復回数付加情報３１２にループの繰り返し回数が静的に与えられていないことを示す「－１」が設定されるので、付加情報解析部２２４は、ループの反復回数に関する比較結果は採用せず、ループ構造深さに関する比較の結果を採用する。すなわち、ループ構造深さ閾値設定部２２３に設定された値「２」と、ループ構造深さ付加情報３１３の値「３」とを比較すると、ループ構造付加情報３１３の値の方が大きい。よって、付加情報解析部２２４は、「ＭＰＩが備える集団通信系ＡＰＩの実行が多い」と判断する。
【００５７】
付加情報解析部２２４は、上記判断の結果を計算リソース割当部２２５に送る。計算リソース割当部２２５は、「ＭＰＩが備える集団通信系ＡＰＩの実行が多い」との判断に基づいて、図７に示したバイセクショナルバンド幅が高いシステム（図２に示すシステム２３０）を、当該アプリケーションプログラムの実行に割り当てる。システム２３０は、当該アプリケーションプログラムの実行を行う。以上の動作により、本実施形態に係る情報処理システム１００は、ＭＰＩを利用したアプリケーションプログラムを実行するシステムを振り分ける。
【００５８】
次に、上記とは異なるアプリケーションプログラムを実行する例について説明する。
【００５９】
図９は、ＭＰＩを利用したアプリケーションプログラムの他の記述の例を示す図である。図９に示すアプリケーションプログラムでは、ＭＰＩの集団通信系ＡＰＩが記述されている箇所（ｂ）において、ループの繰り返し回数が静的に与えられている。すなわち、ループの繰り返し回数は「１０００」と与えられている。さらに、当該箇所におけるループは２重に記述されているので、繰り返し回数の合計は「１０００００００」である。
【００６０】
図９に示すアプリケーションプログラムを受け取ると、情報処理システム１００は、上

記と同様に動作する。すなわち、コンパイラ２１０は、付加情報生成部２１３において、反復回数付加情報３１２として「１０００００００」を、ループ構造深さ付加情報３１３として「２」を生成する。上述したように、ジョブスケジューラ２２０の反復回数閾値設定部２２２には「１０００」が設定されるので、付加情報解析部２２４は、「ＭＰＩが備える集団通信系ＡＰＩの実行が多い」と判断する。

【００６１】

さらに、上記とは異なるアプリケーションプログラムを実行する例について説明する。

【００６２】

ＨＰＣの分野では、科学技術計算を行うことを目的として作られた言語であるＦＯＲＴＲＡＮ（登録商標）によってアプリケーションプログラムが記述されることが多い。したがって、ここでは、ＦＯＲＴＡＮによって記述されたアプリケーションプログラムに着目し、このようなアプリケーションプログラムを本実施形態に係る情報処理システム１００により実行する場合の動作を説明する。

【００６３】

図１０は、ＦＯＲＴＡＮによって記述されたアプリケーションプログラムの記述の例を示す図である。ＦＯＲＴＲＡＮによるファイルＩ／Ｏは、ＯＰＥＮ文の呼出しにより対象とするファイルとプログラムで使用する識別子とを紐づけると共に、その後にその識別子を指定してＷＲＩＴＥ文またはＲＥＡＤ文を呼び出すことによって実施される。

【００６４】

このとき、本実施形態では、ファイルＩ／Ｏの規則性に応じてＯＰＥＮ文の呼出し時にＡＣＣＥＳＳ指定子に指定する値を変更する。すなわち、ファイルＩ／Ｏの規則性がシーケンシャルファイルアクセスである場合、ＡＣＣＥＳＳ指定子に「ＳＥＱＵＥＮＴＩＡＬ」を指定する。一方、ファイルＩ／Ｏの規則性がランダムアクセスの場合、ＡＣＣＥＳＳ指定子に「ＤＩＲＥＣＴ」を指定する。なお、ＡＣＣＥＳＳ指定子の記述を省略した場合、「ＳＥＱＵＥＮＴＩＡＬ」を指定した場合と同等になるようにしてもよい。

【００６５】

ここで、ファイルの入出力先となるストレージディスク（記憶装置）は、今日ではＳＡＴＡ（Ｓｅｒｉａｌ　ＡＴＡ）ディスクとＳＡＳ（Ｓｅｒｉａｌ　Ａｔｔａｃｈｅｄ　ＳＣＳＩ）ディスクが多く使われている。ＳＡＴＡディスクとＳＡＳディスクを比較した場合、一般的にシーケンシャルアクセス性能に大きな差は出ないものの、ランダムアクセス性能はＳＡＳディスクの方が優れている。また、ＳＡＴＡディスクの方がＳＡＳディスクに比べて安価に入手できるという特徴がある。

【００６６】

例えば、アプリケーションを実行するシステムを１０００台のノードを用いて構成することを考える。図１１に示すように、ＳＡＳディスクによって１０００ノード全てを構成した場合、どの計算ノードによりアプリケーションを実行したとしても、ＦＯＲＴＲＡＮにより記述されたアプリケーションプログラムのファイルＩ／Ｏは、性能低下することなく実行される。しかしながら、この場合、システムの構築費用が高額になるという問題がある。

【００６７】

一方、図１２に示すようにＳＡＴＡディスクによって１０００ノード全てを構成した場合、システムの構築費用が安価になるが、その反面、ランダムアクセスのファイルＩ／Ｏの実行性能が低下するという問題がある。

【００６８】

そこで、以下では、図１３に示すように、ＦＯＲＴＡＮによって記述されたアプリケーションプログラム中のＯＰＥＮ文の記述に基づいてファイルＩ／Ｏの特性を判断し、判断の結果に基づいていずれかのシステムにプログラムの実行を振り分ける動作について説明する。

【００６９】

図１０は、ＦＯＴＲＡＮによって記述されたアプリケーションプログラムにおいて、「

ＡＣＣＥＳＳ＝ＤＩＲＥＣＴ」である指定子によってＯＰＥＮ文が実行されたファイルに対して、ループ構造の深さが「３」の位置でＷＲＩＴＥ文を実行する部分を示す。

【００７０】

このアプリケーションプログラムが図２に示すコンパイラ２１０に入力されると、コンパイラ通常機能部２１１は、アプリケーション実行バイナリ３１１を生成する。次に、コンパイラ通常機能部２１１は、アプリケーションプログラムを特定記述解析部２１２に送る。特定記述解析部２１２は、「ＡＣＣＥＳＳ＝ＤＩＲＥＣＴ」である指定子を含むＯＰＥＮ文の有無を調べる。

【００７１】

図１０に示すアプリケーションプログラムでは、「ＡＣＣＥＳＳ＝ＤＩＲＥＣＴ」である指定子を含むＯＰＥＮ文を呼び出しているため、特定記述解析部２１２は、それが記述されている部分を検出する。次に、特定記述解析部２１２は、「ＡＣＣＥＳＳ＝ＤＩＲＥＣＴ」によりＯＰＥＮ文を実行したファイルに、ＷＲＩＴＥ文またはＲＥＡＤ文を実行する箇所のループ構造を解析する。

【００７２】

図１０に示すプログラムでは、ＷＲＩＴＥ文を実行する箇所のループ構造の深さは「３」である。一方、ループの繰り返し回数は「Ｎ１」、「Ｎ２」、「Ｎ３」と記述され静的に与えられていない。したがって、特定記述解析部２１２は、ループ構造の深さを「３」、ループの繰り返し回数を静的に与えられていないことを示す「－１」として、解析結果を出力する。特定記述解析部２１２は、解析結果を付加情報生成部２１３に送る。ここでは、反復回数付加情報３１２として「－１」、ループ構造深さ付加情報３１３として「３」が、それぞれ生成される。

【００７３】

コンパイラ２１０は、生成したアプリケーション実行バイナリ３１１、反復回数付加情報３１２、ループ構造深さ付加情報３１３を含むアプリケーション実行要求３１０を、ジョブスケジューラ２２０に渡す。なお、ここでは、ジョブスケジューラ２２０の反復回数閾値設定部２２２に「１０００」、ループ構造深さ閾値設定部２２３に「２」があらかじめシステム管理者によって設定されている。

【００７４】

アプリケーション実行要求３１０を受け取ったアプリケーション実行受付部２２１は、アプリケーション実行要求３１０を計算リソース割当部２２５に渡す。また、アプリケーション実行受付部２２１は、アプリケーション実行要求３１０に含まれる反復回数付加情報３１２と、ループ構造深さ付加情報３１３を、付加情報解析部２２４に送る。付加情報解析部２２４は、まず反復回数閾値設定部２２２に設定された値と、反復回数付加情報３１２の値とを比較する。ここでは、反復回数付加情報３１２の値は、ループの繰り返し回数が静的に与えられていないことを示す「－１」であるので、付加情報解析部２２４は、反復回数に関する比較結果は採用しない。続いて、付加情報解析部２２４は、ループ構造深さ閾値設定部２２３に設定された値「２」と、ループ構造深さ付加情報３１３の値「３」とを比較する。その結果、ループ構造付加情報３１３の値の方が大きい。したがって、付加情報解析部２２４は、「ランダムアクセスによりＯＰＥＮ文を実行したファイルへのＩ／Ｏは多い」と判断する。

【００７５】

付加情報解析部２２４は、上記判断結果を計算リソース割当部２２５に渡す。計算リソース割当部２２５は上記判断結果に基づいて、当該アプリケーションプログラムに、図１３に示す、ＳＡＳディスクを備えたシステムを割り当る。ＳＡＳディスクを備えたシステムは、割り当てられたアプリケーションプログラムを実行する。

【００７６】

以上のように、本第１の実施形態によれば、コンパイラ２１０は、特定記述解析部２１２により、アプリケーションプログラムの実行特性に影響を及ぼす特定記述の有無を調べ、特定記述がある場合、その特定記述が記述されている箇所のループ構造の深さとループ

の繰り返し回数を調べる。ジョブスケジューラ２２０における付加情報解析部２２４は、上記ループ構造の深さと、ループの繰り返し回数の少なくともいずれかが閾値より大きい場合、当該アプリケーションプログラムは、「特定記述の実行が多い」という特性に当てはまると判断する。計算リソース割当部２２５は、特定記述の実行が多いと判断されたアプリケーションプログラムに、実行に適したシステムを割り当てる。

【００７７】

上記構成を採用することにより、本第１の実施形態によれば、ＭＰＩが備える集団通信系ＡＰＩの実行が多いアプリケーションプログラムの実行に、バイセクショナルバンド幅が高いシステムを割り当てることができる。また、ＦＯＲＴＲＡＮにより記述され、ランダムアクセスによりＯＰＥＮ文を実行したファイルへのＩ／Ｏが多いアプリケーションプログラムの実行に、ＳＡＳディスクを備えたシステムを割り当てることができる。

【００７８】

つまり、アプリケーションを構成するプログラムの特性をコンパイラによって解析した結果に基づいて、アプリケーションの実行に適した計算リソースを割り当てることができるので、システムの構築コストを抑えながら、エンドユーザに手間をかけずにアプリケーションの実行性能を最大化できるという効果が得られる。

【００７９】

第２の実施形態

図１４は、本発明の第２の実施形態に係る情報処理システム４００の構成を示すブロック図である。図１４に示すように、情報処理システム４００は、ジョブスケジューラ４２０、システム４５０、システム４６０を備える。ジョブスケジューラ４２０は、アプリケーション実行要求４１０を受け取る。

【００８０】

本実施形態に係る情報処理システム４００では、一般的なコンパイラによってアプリケーションプログラムをコンパイルし、エンドユーザによって、アプリケーションの実行がジョブスケジューラ４２０に要求される。

【００８１】

図１４に示すように、ジョブスケジューラ４２０は、アプリケーション実行受付部４２１、ループ構造深さ閾値設定部４２２、特定記述解析部４２３、付加情報生成部４２４、ループ構造深さ付加情報４２５、付加情報解析部４２６、計算リソース割当部４２７を備える。

【００８２】

ジョブスケジューラ４２０は、エンドユーザからのアプリケーション実行要求４１０をアプリケーション実行受付部４２１において受け付ける。アプリケーション実行要求４１０は、一般的なコンパイラによってコンパイルされたアプリケーション実行バイナリ４１１を指定する。

【００８３】

アプリケーション実行受付部４２１は、アプリケーション実行要求４１０において指定されたアプリケーション実行バイナリ４１１を、外部の逆アセンブラ４３０に渡す。逆アセンブラ４３０は、情報処理システム４００と通信可能な情報処理装置で実行可能であり、アプリケーション実行バイナリ４１１に対して逆アセンブルを実行する。

【００８４】

逆アセンブリ４３０による実行の結果、アプリケーションの命令リスト４４０が生成される。命令リスト４４０は、ジョブスケジューラ４２０の特定記述解析部４２３に送られる。特定記述解析部４２３は、命令リスト４４０に基づいてアプリケーションの実行特性に影響を及ぼす特定記述の箇所を調べると共に、当該箇所が見つかった場合は、当該箇所におけるループ構造の深さを調べる。

【００８５】

特定記述解析部４２３は、上記調べた結果を付加情報生成部４２４に送る。付加情報生成部４２４は、ループ構造の深さに関する結果を、ループ構造深さ付加情報４２５として

生成すると共に、生成したループ構造深さ付加情報４２５を付加情報解析部４２６に送る。

【００８６】

付加情報解析部４２６は、ループ構造深さ付加情報４２５とループ構造深さ閾値設定部４２２に設定された値とを比較する。ループ構造深さ閾値設定部４２２に設定された値より、ループ構造深さ付加情報４２５の値が大きい場合、付加情報解析部４２６は、当該アプリケーションプログラムは、「特定記述の実行が多い」という特性に当てはまると判断する。その判断の結果を特性情報と称する。

【００８７】

付加情報解析部４２６は、上記特性情報を計算リソース割当部４２７に渡す。なお、ループ構造深さ閾値設定部４２２には、システム管理者によってあらかじめ閾値が設定される。

【００８８】

計算リソース割当部４２７は、アプリケーション実行受付部４２１から渡されたアプリケーション実行要求４１０と、付加情報解析部４２６から渡された特性情報とに基づいて、アプリケーションを実行するシステムを判断する。この場合、計算リソース割当部４２７は、当該アプリケーションプログラムに、特定記述の実行に適したシステム４５０を割り当てる。システム４５０は、割り当てられたアプリケーションプログラムを実行する。

【００８９】

次に、具体的な例を用いて情報処理システム４００の動作について説明する。ここでは、ＭＰＩを利用したアプリケーションの実行バイナリから、アプリケーションの集団通信系ＡＰＩの呼出し状況を判断すると共に、バイセクショナルバンド幅が高いシステムと、バイセクショナルバンド幅が低いシステムに、アプリケーションの実行を振り分ける動作について説明する。

【００９０】

図１５は、ＭＰＩの集団通信系ＡＰＩである”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”を、３重のループ構造の中（ｃ）で呼び出すアプリケーションプログラムの記述の一部を示す。

【００９１】

図１５を含むソースプログラムをコンパイルした結果として生成されたアプリケーション実行バイナリを、図１４に示すアプリケーション実行バイナリ４１１とする。このアプリケーション実行バイナリ４１１を含むアプリケーション実行要求４１０が、ジョブスケジューラ４２０に送られると、ジョブスケジューラ４２０はそのアプリケーション実行要求４１０をアプリケーション実行受付部４２１で受け付ける。

【００９２】

アプリケーション実行受付部４２１は、アプリケーション実行要求４１０において指定されたアプリケーション実行バイナリ４１１を、逆アセンブラ４３０に渡す。逆アセンブラ４３０は、アプリケーション実行バイナリ４１１に対して逆アセンブルを実行する。

【００９３】

逆アセンブルした結果、アプリケーションの命令リスト４４０が生成される。図１６は、アプリケーション実行バイナリを逆アセンブルした結果から、図１５に示す部分に相当する部分を抜き出した命令リストを示す図である。

【００９４】

命令「ｃａｌｌｑ　４００８５０　＜ＭＰＩ＿Ａｌｌｔｏａｌｌ@ｐｌｔ＞」４３１は、”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”の呼出しを示す。また、ループ４３２－４３４で示される各範囲は、図１５に示すfor文の各範囲に相当する。命令リスト４４０は、特定記述解析部４２３に渡される。

【００９５】

特定記述解析部４２３は、受け取った命令リスト４４０に基づいて、ＭＰＩの集団通信系ＡＰＩの呼出しの有無を調べる。ここでは、”ＭＰＩ＿Ａｌｌｔｏａｌｌ（）”の呼出しが見つかる。また、特定記述解析部４２３は、ＭＰＩの集団通信系ＡＰＩが呼び出され

る箇所におけるループ構造の深さを調べる。ここでは、特定記述解析部４２３は、ループ構造の深さを「３」と検出する。

【００９６】

特定記述解析部４２３は、上記調べた結果を付加情報生成部４２４に送る。付加情報生成部４２４は、ループ構造深さ付加情報４２５として「３」を生成すると共に、それを付加情報解析部４２６に送る。

【００９７】

ここで、ループ構造深さ閾値設定部４２２には、あらかじめシステム管理者によって「２」が設定さる。付加情報解析部４２６は、ループ構造深さ閾値設定部４２２に設定された「２」と、ループ構造深さ付加情報４２５である「３」とを比較する。ループ構造深さ付加情報４２５の方が大きいので、付加情報解析部４２６は、「集団通信系ＡＰＩの呼出しが多い」という特性に当てはまると判断する。

【００９８】

付加情報解析部４２６は、上記調べた特性情報を、計算リソース割当部４２７に渡す。計算リソース割当部４２７は、当該アプリケーションプログラムに、特定記述の実行に適したシステム４５０を割り当てる。システム４５０は、割り当てられたアプリケーションプログラムを実行する。

【００９９】

以上のように、本実施形態によれば、情報処理システム４００におけるジョブスケジューラ４２０は、アプリケーション実行バイナリ４１１を逆アセンブラ４３０に渡すと共に、逆アセンブルの実行結果として命令リスト４４０を受け取る。ジョブスケジューラ４２０は、特定記述解析部４２３において、命令リスト４４０に特定記述があるか否かを調べ、ある場合はその箇所におけるループ構造深さを調べる。ループ構造深さが閾値よりも大きい場合、付加情報解析部４２６は、命令リスト４４０に対応するアプリケーションプログラムは、特定記述の実行が多いと判断する。計算リソース割当部４２７は、そのアプリケーションプログラムに、その実行に適したシステムを割り当てる。

【０１００】

上記構成を採用することにより、本第２の実施形態によれば、ソースプログラムが与えられずコンパイラによるプログラムの特性解析が不可能な場合でも、上記第１の実施形態と同様の効果が得られる。すなわち、アプリケーションの実行バイナリを解析することにより、アプリケーションの実行に適した計算リソースを割り当てることができるので、システムの構築コストを抑えながら、エンドユーザに手間をかけずにアプリケーションの実行性能を最大化できるという効果が得られる。

【０１０１】

第３の実施形態

図１７は、本発明の第３の実施形態に係る情報処理装置５０の構成の概要を示す図である。図１７に示すように、情報処理装置５０は、特定記述解析部５１と計算リソース割当部５２とを備える。

【０１０２】

特定記述解析部５１は、アプリケーションの実行特性に影響を及ぼす特定記述が当該アプリケーションに含まれる場合、前記特定記述によって呼び出される命令または関数の実行頻度に関する付加情報を生成する。計算リソース割当部５２は、前記付加情報に基づいて、前記命令または関数の実行に適した計算リソースを判断すると共に、当該計算リソースを前記アプリケーションの実行に割り当てる。

【０１０３】

上記構成を採用することにより、本第３の実施形態によれば、構築費用を低減できると共に、エンドユーザの手間をかけることなくアプリケーションの実行性能を向上することができるという効果が得られる。

【０１０４】

なお、図２、図１４に示したコンパイラおよびジョブスケジューラの各部は、図１８に

例示するハードウエア資源において実現される。すなわち、図１８に示す構成は、ＣＰＵ１０、ＲＡＭ（Ｒａｎｄｏｍ　Ａｃｃｅｓｓ　Ｍｅｍｏｒｙ）１１、ＲＯＭ（Ｒｅａｄ　Ｏｎｌｙ　Ｍｅｍｏｒｙ）１２、外部接続インタフェース１３および記憶媒体１４を備える。ＣＰＵ１０は、ＲＯＭ１２または記憶媒体１４に記憶された各種ソフトウエア・プログラム（コンピュータ・プログラム）を、ＲＡＭ１１に読み出して実行することにより、コンパイラおよびジョブスケジューラの全体的な動作を司る。すなわち、上記各実施形態において、ＣＰＵ１０は、ＲＯＭ１２または記憶媒体１４を適宜参照しながら、コンパイラおよびジョブスケジューラが備える各機能（各部）を実行するソフトウエア・プログラムを実行する。

【０１０５】

また、上述した各実施形態では、図２、図１４に示したコンパイラおよびジョブスケジューラにおける各ブロックに示す機能を、図１８に示すＣＰＵ１０が実行する一例として、ソフトウエア・プログラムによって実現する場合について説明した。しかしながら、図２、図１４に示した各ブロックに示す機能は、一部または全部を、ハードウエアとして実現してもよい。

【０１０６】

また、各実施形態を例に説明した本発明は、コンパイラおよびジョブスケジューラに対して、上記説明した機能を実現可能なコンピュータ・プログラムを供給した後、そのコンピュータ・プログラムを、ＣＰＵ１０がＲＡＭ１１に読み出して実行することによって達成される。

【０１０７】

また、係る供給されたコンピュータ・プログラムは、読み書き可能なメモリ（一時記憶媒体）またはハードディスク装置等のコンピュータ読み取り可能な記憶デバイスに格納すればよい。そして、このような場合において、本発明は、係るコンピュータ・プログラムを表すコード或いは係るコンピュータ・プログラムを格納した記憶媒体によって構成されると捉えることができる。

【産業上の利用可能性】

【０１０８】

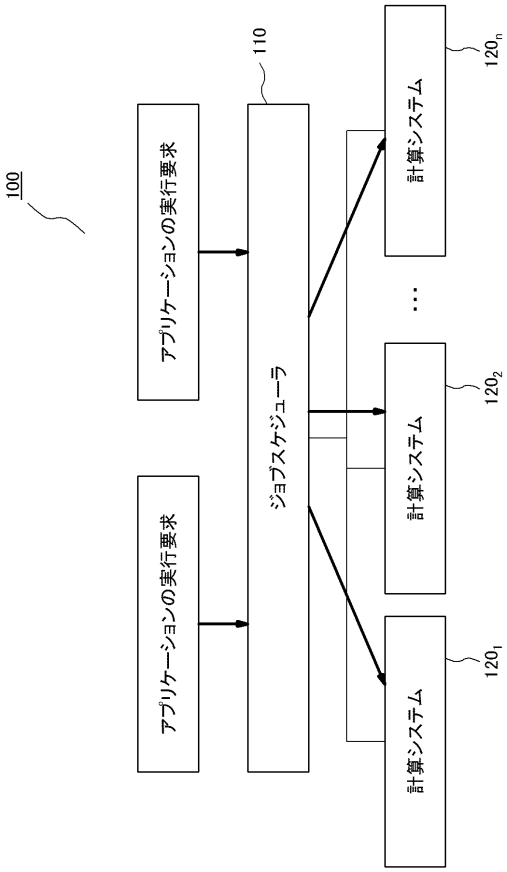
本発明は、例えば、ハイパフォーマンスコンピューティングの分野に適用できる。

【符号の説明】

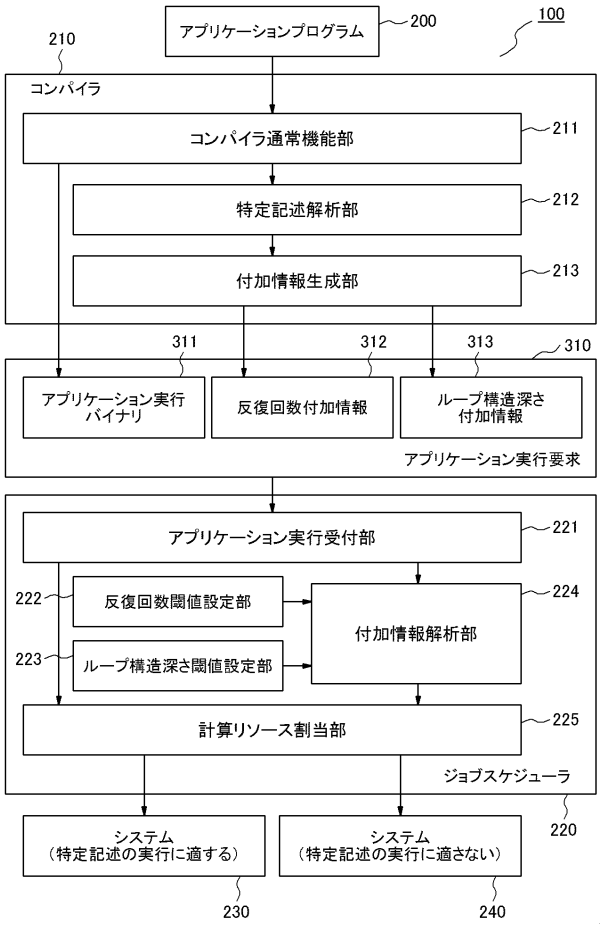
【０１０９】

１０　ＣＰＵ
１１　ＲＡＭ
１２　ＲＯＭ
１３　外部接続インタフェース
１４　記憶媒体
１００、４００　情報処理システム
１１０、２２０、４２０　ジョブスケジューラ
２１０　コンパイラ
２２１、４２１　アプリケーション実行受付部
２２２　反復回数閾値設定部
２２３、４２２　ループ構造深さ閾値設定部
２２４、４２６　付加情報解析部
２２５、４２７　計算リソース割当部
２３０、２４０、４５０、４６０　システム
３１０、４１０　アプリケーション実行要求
３１１、４１１　アプリケーション実行バイナリ
３１２　反復回数付加情報
３１３、４２５　ループ構造深さ付加情報

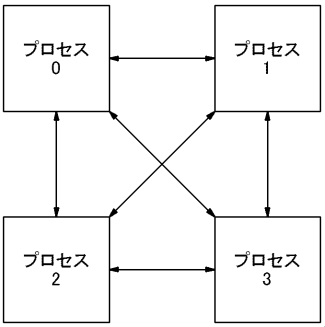
【図１】



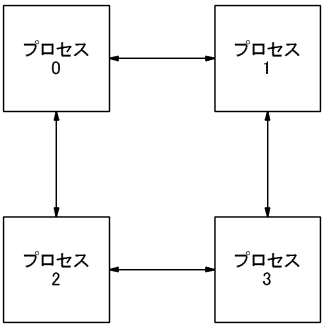
【図２】



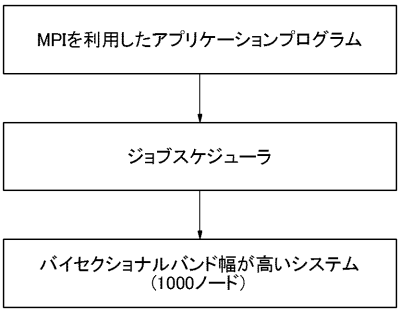
【図３】



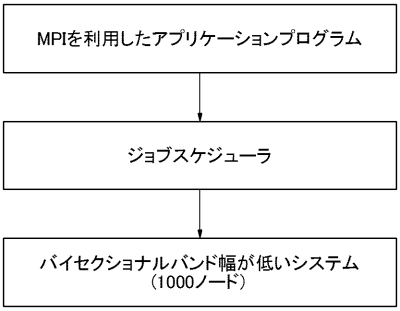
【図４】



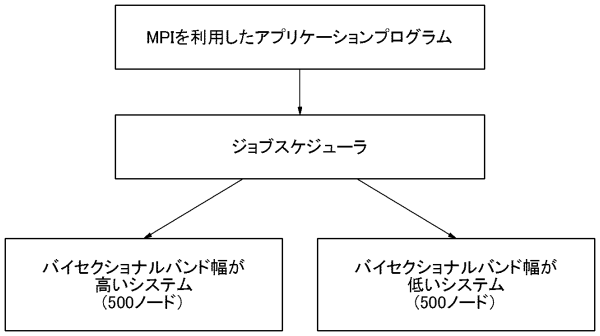
【図５】



【図６】



【図７】



【図８】

```
for (int i = 0 ; i < N1 ; ++i ) {
    for ( int j = 0 ; j < N2 ; ++j ) {
        for ( int k = 0 ; k < N3 ; k++ ) {
            MPI_Alltoall( ) ; .....(a)
        }
    }
}
```

【図９】

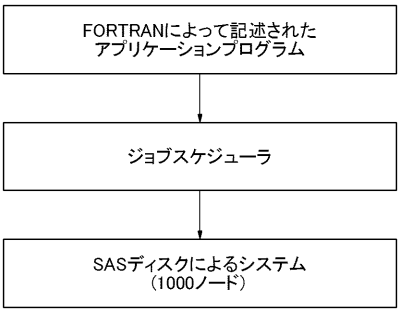
```
for (int i = 0 ; i < 1000 ; ++i ) {
    for ( int j = 0 ; j < 1000 ; ++j ) {
        MPI_Alltoall( ) ; .....(b)
    }
}
```

【図１０】

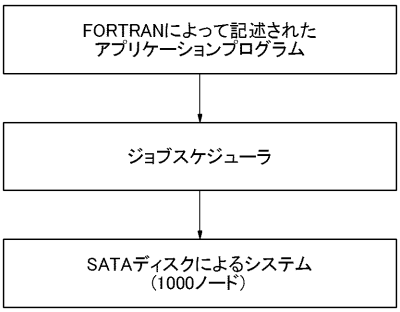
```
open (17,file= 'tmpfile' , access=' direct ', recl=8)

do i = 1, N1
  do j = 1, N2
    do k = 1, N3
      write (17, rec=POSITION) NUM
    end do
  end do
end do
```

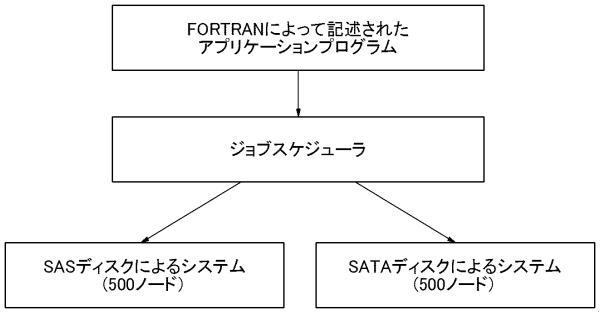
【図１１】



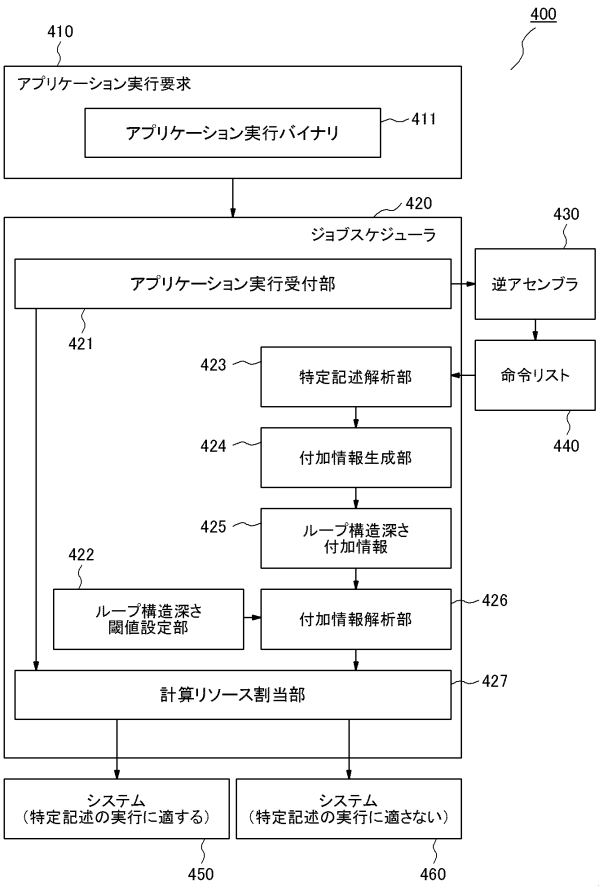
【図１２】



【図１３】



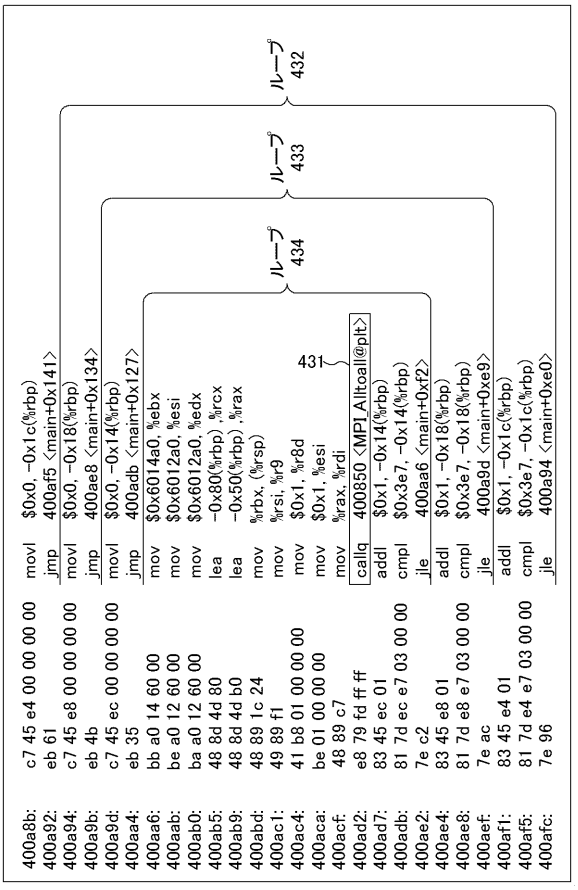
【図１４】



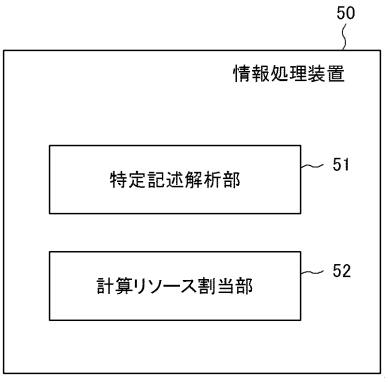
【図１５】

```
for ( i = 0 ; i < 1000 ; i++ ) {
    for ( j = 0 ; j < 1000 ; j++ ) {
        for ( k = 0 ; k < 1000 ; k++ ) {
            MPI_Alltoall(sendbuf , 1 , MPI_INT , recvbuf , 1 ,
                         MPI_INT , MPI_COMM_WORLD ) ; .....(c)
        }
    }
}
```

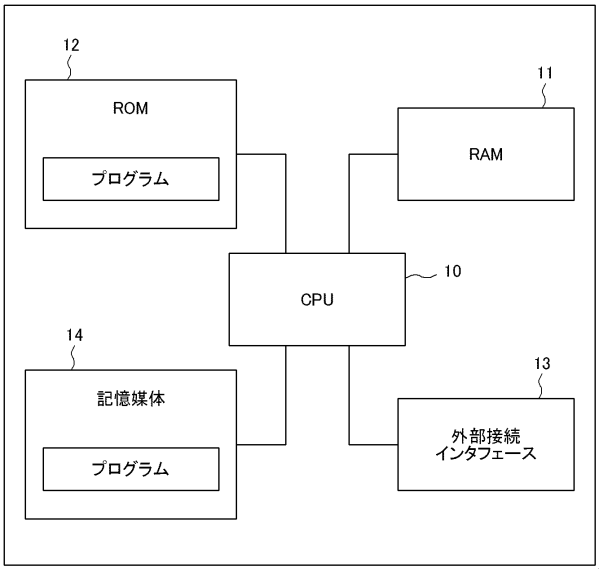
【図１６】



【図１７】



【図１８】



フロントページの続き

(56)参考文献 特表２００９－５２８６４９（ＪＰ，Ａ）
特開２００９－１４０４５１（ＪＰ，Ａ）
特表２０１１－５３０７６８（ＪＰ，Ａ）
特開２００１－１８４２１８（ＪＰ，Ａ）
特開２０１１－１０７９８３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４６－９／５４