(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4696282号
(P4696282)
(45)発行日　平成23年6月8日(2011.6.8)　(24)登録日　平成23年3月11日(2011.3.11)

(51)Int.Cl.　F I
G06F 9/44 (2006.01)　G06F 9/06 620K

請求項の数 3　(全 25 頁)

(21)出願番号　特願2003-22792(P2003-22792)
(22)出願日　平成15年1月30日(2003.1.30)
(65)公開番号　特開2004-234393(P2004-234393A)
(43)公開日　平成16年8月19日(2004.8.19)
審査請求日　平成17年12月2日(2005.12.2)
審判番号　不服2009-9411(P2009-9411/J1)
審判請求日　平成21年4月30日(2009.4.30)

(73)特許権者　503360115
独立行政法人科学技術振興機構
埼玉県川口市本町四丁目１番８号
(74)代理人　110000338
特許業務法人原謙三国際特許事務所
(72)発明者　片桐　孝洋
東京都調布市小島町１－１－１　公務員宿舎ＲＡ１０４

合議体
審判長　赤川　誠一
審判官　宮司　卓佳
審判官　石井　茂和

最終頁に続く

(54)【発明の名称】計算装置、プログラム、及び、記録媒体

(57)【特許請求の範囲】
【請求項１】
演算性能を変化させて演算結果を変化させない性能情報パラメータと演算対象行列のサイズを示す基本情報パラメータとを参照して固有値計算を行う固有値計算手段を備えた計算装置において、上記固有値計算手段が上記固有値計算を実行する前に上記性能情報パラメータの値を最適化する最適化手段を備え、
上記最適化手段は、上記基本情報パラメータの値がユーザにより指定された時点で、予めサンプリングされた複数のサンプリング値の各々を上記性能情報パラメータとし、ユーザにより指定された指定値を上記基本情報パラメータとして上記固有値計算手段に上記固有値計算を試行させ、上記固有値計算手段が上記複数のサンプリング値の各々を上記性能情報パラメータとして上記固有値計算を試行するのに要した演算時間から、上記固有値計算手段が上記指定値を上記基本情報パラメータとして上記固有値計算を実行するのに要する演算時間を最小化する上記性能情報パラメータの値を推定するとともに、推定した上記性能情報パラメータの値をパラメータ情報ファイルに保存し、
上記固有値計算手段は、上記固有値計算の実行がユーザにより指示された時点で、上記パラメータ情報ファイルに保存された上記性能情報パラメータの値であって、上記最適化手段により推定された上記性能情報パラメータの値を参照して上記固有値計算を実行する
、
ことを特徴とする計算装置。
【請求項２】

コンピュータを請求項１に記載の計算装置として動作させるためのプログラムであって、上記コンピュータを上記固有値計算手段及び上記最適化手段として機能させるプログラム。

【請求項３】

請求項２に記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、例えばコンピュータに備えられたライブラリのパラメータの最適化を、コンピュータに実行させるためのプログラム、記録媒体およびコンピュータに関するものである。

【０００２】

【従来の技術】

コンピュータのような計算装置において数値計算ライブラリなどのソフトウェアを用いる際には、ライブラリ中の所望のサブルーチンに対して、ユーザが所望の問題に応じてパラメータを指定する。その後、ユーザの指定したパラメータを用いてサブルーチンが実行され、結果が出力される。

【０００３】

例えば、数値計算ライブラリのサブルーチンとして、行列の固有値を計算する固有値計算サブルーチンを考える。このとき、サブルーチンに対してユーザが指定するパラメータのうちには、所望の行列の実体や、その行列のサイズなどがある。これらのパラメータは、その問題を実際に解く際に必要とされるパラメータである。

【０００４】

一方、パラメータのうちには、どのように設定しても問題の答えとしては同じものが得られるが、適切に設定することによって例えば数値計算に要する時間を短縮できたりするような、いわゆる最適化のためのパラメータがある。

【０００５】

例えば、計算装置が複数のプロセッサを備えた並列計算装置であるとき、行列計算におけるループアンローリング段数（アンローリング段数）は、最適化のためのパラメータである。

【０００６】

ここで、アンローリング段数とは、ループの計算において通常１に設定している、ループごとの増分を意味する。例えば、ベクトルＡ(i)とベクトルＢ(i)の和Ｃ(i)を計算する際に、アンローリング段数を２に設定した場合には、i成分の和Ｃ(i)＝Ａ(i)＋Ｂ(i)とi+1成分の和Ｃ(i+1)＝Ａ(i+1)＋Ｂ(i+1)とがループ内においてそれぞれ計算され、その後iを２だけ増加させる。

【０００７】

問題の性質や計算装置の性能（並列プロセッサの数など）に応じてアンローリング段数を調整することによって、計算装置における計算時間を最も短く（最適化）できる。

【０００８】

このような最適化のためのパラメータを調節して計算時間（性能）などのコストを最適なものとする調節機能を備えた計算装置が知られている。このような調整機能は通常ソフトウェア（自動チューニングソフトウェア）によって実現される。

【０００９】

従来の自動チューニングソフトウェアの構成方式として、ソフトウェアインストール時にパラメータ最適化を行うものがある。例えば、ソフトウェアインストール時にアンローリング段数を最適化する場合には、以下のようにする。

【００１０】

この場合には、解くべき問題の問題サイズなどが決まっていないため、適当にサンプリングした問題サイズごとに最適なアンローリング段数を求める。その後に、サンプリングし

た問題サイズごとの最適なアンローリング段数を、例えば問題サイズについて適当な補間関数によって補間する。なお、ある問題サイズにおいて最適なアンローリング段数を求める際に、例えばアンローリング段数についてもサンプリングを行い、補間関数を用いて最もコストの小さいアンローリング段数を選択してもよい。

【００１１】

このように、例えば適当な補間関数を用いたモデル化によって、後に選択を行う際に、問題サイズに応じた最適なアンローリング段数の推定値を得ることができる。

【００１２】

または、従来の自動チューニングソフトウェアの構成方式の他の一例として、ライブラリ実行時にパラメータ最適化を行うものがある。例えば、コストを変化させる大きな要因として、行列の実体のような実行時でないと確定しない要素が含まれる問題について、このようなパラメータ最適化を行う。

【００１３】

この場合には、ライブラリコールが行われた時点で、所望の問題サイズ、行列の実体などに対して所望のパラメータを幾つか試行して、最適なものを選択する。

【００１４】

なお、このようなライブラリ実行時に最適化を行う場合には、このチューニングを行う時間についてもコストとしての計算時間に含まれることに注意が必要である。すなわち、チューニングを行う時間とその後の計算時間とが、チューニングせずにパラメータを何らかの値に固定しておく場合の計算時間よりも少なくなる必要がある。

【００１５】

これらの従来の自動チューニングソフトウェアについては、以下の非特許文献１、非特許文献２を参照されたい。

【００１６】

なお、例えば日本国の公開特許公報「特開２０００－２７６４５４号公報（公開日：２０００年１０月６日）」には、並列計算機におけるソフトウェアの実行性能を大きく左右し、かつ、ユーザインタフェースには現れないパラメータを調節してインストールを行う機能を有するソフトウェアの構成方法が記載されている。この場合には、ソフトウェアインストール時にパラメータ最適化が行われる。

【００１７】

【特許文献１】

特開２０００－２７６４５４号公報

【００１８】

【非特許文献１】

片桐孝洋，他４名、「自動チューニング機構が並列数値計算ライブラリに及ぼす効果」、情報処理学会論文誌：ハイパフォーマンスコンピューティング、社団法人情報処理学会、２００１年１１月、第４２巻、第１２号（ＨＰＳ４）、ｐ．６０－７６

【００１９】

【非特許文献２】

直野健、山本有作、「単一メモリ型インタフェースを有する自動チューニング並列ライブラリの構成方法」、情報処理学会研究報告、社団法人情報処理学会、２００１年７月２５日、第７７巻、ｐ．２５－３０

【００２０】

【発明が解決しようとする課題】

しかしながら、従来の自動チューニングソフトウェアの構成方式では、ソフトウェアインストール時にパラメータ最適化を行うもの、またはライブラリ実行時にパラメータ最適化を行うもの、のみが存在していたため、パラメータ調整が不十分となる場合があるという問題を生ずる。

【００２１】

すなわち、従来の、ソフトウェアインストール時にパラメータ最適化を行う構成において

は、例えば補間関数を用いたモデルに基づき、最適化パラメータを推定によって決定する。このため、十分な精度が得られない虞れがある。

【００２２】

また、従来の、ライブラリ実行時にパラメータ最適化を行う構成においては、ライブラリ実行時のパラメータチューニングに要する時間もコストとなるため、チューニングに十分な時間を費やすことができずに、精度が不十分なものとなる虞れがある。

【００２３】

また、従来は、汎用的な処理に適用できる自動チューニングソフトウェアがなかったという問題がある。例えば、非特許文献１に記載のＡＴＬＡＳは、数値計算ライブラリの中でも、ＢＬＡＳ(Basic Linear Algebra Subprograms)と呼ばれるライブラリのみ最適化できる。これは、汎用的な処理に適用できるものではない。

【００２４】

すなわち、問題によっては、インストール時にしか最適化できない、または実行時にしか最適化できないものがある。このため、従来のように、ソフトウェアインストール時にパラメータ最適化を行うもの、またはライブラリ実行時にパラメータ最適化を行うもの、のいずれか一方しかない場合には、全ての問題について、それぞれ最適化することはできない。すなわち、汎用的な処理に適用できないという問題がある。

【００２５】

本発明は、上記の問題点に鑑みてなされたものであり、その目的は、精密なパラメータ調整を行うことのできるプログラム、記録媒体およびコンピュータを提供することにある。また、本発明は、汎用的な処理に適用できるプログラム、記録媒体およびコンピュータを提供することも目的とする。

【００２６】

【課題を解決するための手段】

本発明に係るプログラムは、上記課題を解決するために、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの最適化を、上記コンピュータに実行させるためのプログラムにおいて、上記ライブラリの上記パラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する手順と、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う手順とを含んでいることを特徴としている。

【００２７】

このプログラムは、コンピュータにおける例えば数値計算ライブラリのようなライブラリの実行コストを最適化するために用いられるプログラムである。実行コストとは、例えば実行に要する計算資源、計算時間である。このプログラムは、ライブラリのパラメータのうち、実行性能のみを変化させてライブラリの出力を変化させない性能情報パラメータの値を、ライブラリの実行コストが最適なものとなるように調整する。

【００２８】

上記プログラムが実行されたコンピュータは、ライブラリの実際の実行の前に、例えばユーザからの基本情報パラメータの入力を検出することによって、基本情報パラメータが定まった時点を検出する。

【００２９】

ここで、基本情報パラメータとは、実行性能とライブラリの出力とを共に変化させるパラメータである。

【００３０】

例えば数値計算ライブラリのうちの、行列の固有値計算ライブラリにおいては、行列のサイズ、行列の実体などが、基本情報パラメータに相当する。また、例えば並列計算機を用いる場合のループアンローリング段数は、性能情報パラメータに相当する。

【００３１】

すなわち、ライブラリの内容を数式として表したときに、数式中の変数として表現される

パラメータが、基本情報パラメータに相当する。また、数式中に現れず、または数式において単なる媒介変数として現れるパラメータが、性能情報パラメータに相当する。このため、例えば性能情報パラメータを変化させたとしても、数式によって得られる結果（ライブラリの出力）は変わらない。

【００３２】

その後、コンピュータは、ライブラリの実際の実行の前に、基本情報パラメータを用いて性能情報パラメータの最適化を行う。より詳細には、例えば基本情報パラメータを用い、性能情報パラメータのそれぞれの値について試行計算を行って、実行コストを予め実測する。これによって、確実に最適な性能情報パラメータを得ることができる。

【００３３】

ここで、従来の最適化のためのプログラムの一例は、例えばライブラリのインストール時に性能情報パラメータの最適化を行う。この場合、例えば行列のサイズのような基本情報パラメータが定まっていないため、所定の誤差を含んだ、なんらかの推定モデルによって、最適な性能情報パラメータを推測する。

【００３４】

また、従来の最適化のためのプログラムの他の一例は、例えばライブラリの実行時に性能情報パラメータの最適化を行う。この場合には、性能情報パラメータを最適化するための計算時間が、ライブラリの実行コストに計上されてしまう。このため、最適化のために十分な時間を取れずに、最適なパラメータが得られない虞れがある。

【００３５】

そこで、本発明に係る上述のプログラムのように、実際の計算の前に、実行コストを予め実測して、最適な性能情報パラメータを得るようにする。これによって、より精密かつ確実なパラメータ調整が可能となる。また、プログラムの実行前において、計算所要時間を予測できる。

【００３６】

なお、本発明に係るプログラムを、ユーザが知りうる情報が定まった時点でのパラメタ最適化機能を有するソフトウェアである、と表現することもできる。

【００３７】

本発明に係るプログラムは、上記課題を解決するために、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの最適化を、上記コンピュータに実行させるためのプログラムにおいて、上記ライブラリのインストール時に上記性能情報パラメータの最適化を行う初期設定手順と、上記ライブラリの上記パラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記初期設定手順において設定された上記性能情報パラメータを参照して、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う前調整手順とを含んでいることを特徴としている。

【００３８】

このプログラムは、コンピュータにおける例えば数値計算ライブラリのようなライブラリの実行コストを最適化するために用いられるプログラムである。実行コストとは、例えば実行に要する計算資源、計算時間である。このプログラムは、ライブラリのパラメータのうち、実行性能のみを変化させてライブラリの出力を変化させない性能情報パラメータの値を、ライブラリの実行コストが最適なものとなるように調整する。

【００３９】

上記プログラムが実行されたコンピュータは、ライブラリのインストール時に、性能情報パラメータの最適化を行う。この場合、例えば行列のサイズのような基本情報パラメータが定まっていないため、所定の誤差を含んだ、なんらかの推定モデルによって、最適な性能情報パラメータを推測する。

【００４０】

また、コンピュータは、ライブラリの実際の実行の前に、例えばユーザからの基本情報

パラメータの入力を検出することによって、基本情報パラメータが定まった時点を検出する。

【００４１】

ここで、基本情報パラメータとは、実行性能とライブラリの出力とを共に変化させるパラメータである。

【００４２】

例えば数値計算ライブラリのうちの、行列の固有値計算ライブラリにおいては、行列のサイズ、行列の実体などが、基本情報パラメータに相当する。また、例えば並列計算機を用いる場合のループアンローリング段数は、性能情報パラメータに相当する。

【００４３】

その後、コンピュータは、ライブラリの実際の実行の前に、インストール時に設定された性能情報パラメータを参照して、基本情報パラメータを用いて性能情報パラメータの最適化を行う。より詳細には、例えば基本情報パラメータを用い、性能情報パラメータのそれぞれの値について試行計算を行って、実行コストを予め実測する。特に、インストール時に設定された性能情報パラメータの最適値周辺の値のみについて、試行計算を行うようにしてもよい。これによって、試行計算の回数を削減して、最適な性能情報パラメータを得ることができる。このように、より精密かつ確実なパラメータ調整が可能となる。

【００４４】

なお、本発明に係るプログラムを、ソフトウェアのインストール時、およびユーザが知りうる情報が定まった時点でのソフトウェアの実行前、のパラメタ最適化機能を有するソフトウェアである、と表現することもできる。

【００４５】

本発明に係るプログラムは、上記課題を解決するために、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの最適化を、上記コンピュータに実行させるためのプログラムにおいて、上記ライブラリの上記パラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う前調整手順と、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順とを含んでいることを特徴としている。

【００４６】

このプログラムは、コンピュータにおける例えば数値計算ライブラリのようなライブラリの実行コストを最適化するために用いられるプログラムである。実行コストとは、例えば実行に要する計算資源、計算時間である。このプログラムは、ライブラリのパラメータのうち、実行性能のみを変化させてライブラリの出力を変化させない性能情報パラメータの値を、ライブラリの実行コストが最適なものとなるように調整する。

【００４７】

上記プログラムが実行されたコンピュータは、ライブラリの実際の実行の前に、例えばユーザからの基本情報パラメータの入力を検出することによって、基本情報パラメータが定まった時点を検出する。

【００４８】

ここで、基本情報パラメータとは、実行性能とライブラリの出力とを共に変化させるパラメータである。

【００４９】

例えば数値計算ライブラリのうちの、行列の固有値計算ライブラリにおいては、行列のサイズ、行列の実体などが、基本情報パラメータに相当する。また、例えば並列計算機を用いる場合のループアンローリング段数は、性能情報パラメータに相当する。

【００５０】

その後、コンピュータは、ライブラリの実際の実行の前に、基本情報パラメータを用いて性能情報パラメータの最適化を行う。より詳細には、例えば基本情報パラメータを用い、性能情報パラメータのそれぞれの値について試行計算を行って、実行コストを予め実測する。これによって、確実に最適な性能情報パラメータを得ることができる。

【００５１】

また、コンピュータは、ライブラリの実際の実行の際に、既に設定された性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしているか否かを試行により判別する。そして、所望の精度を満たしていないときには、基本情報パラメータを用いて性能情報パラメータの最適化を再度実行する。そして、所望の精度を得ることのできる性能情報パラメータを用いて、ライブラリを実行する。

【００５２】

このように、実際の計算の前に、実行コストを予め実測して、最適な性能情報パラメータを得るようにする。基本情報パラメータの変更がないときには、予め設定した性能情報パラメータを用いてライブラリを実行できる。また、基本情報パラメータの変更があるときでも、所望の精度が得られる場合には、パラメータの最適化のための計算をせずに、ライブラリを実行できる。したがって、実行時におけるパラメータの最適化に要する時間を不要として、ライブラリの実行コスト（計算時間）を増大させない。また、ライブラリの実行の前に精度を確認するので、より精密かつ確実なパラメータ調整が可能となる。

【００５３】

なお、本発明に係るプログラムを、ユーザが知りうる情報が定まった時点でのソフトウェアの実行前、およびソフトウェア実行時、のパラメタ最適化機能を有するソフトウェアである、と表現することもできる。

【００５４】

本発明に係るプログラムは、上記課題を解決するために、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの最適化を、上記コンピュータに実行させるためのプログラムにおいて、上記ライブラリのインストール時に上記性能情報パラメータの最適化を行う初期設定手順と、上記ライブラリの上記パラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順とを含んでいることを特徴としている。

【００５５】

このプログラムは、コンピュータにおける例えば数値計算ライブラリのようなライブラリの実行コストを最適化するために用いられるプログラムである。実行コストとは、例えば実行に要する計算資源、計算時間である。このプログラムは、ライブラリのパラメータのうち、実行性能のみを変化させてライブラリの出力を変化させない性能情報パラメータの値を、ライブラリの実行コストが最適なものとなるように調整する。

【００５６】

上記プログラムが実行されたコンピュータは、ライブラリのインストール時に、性能情報パラメータの最適化を行う。この場合、例えば行列のサイズのような基本情報パラメータが定まっていないため、所定の誤差を含んだ、なんらかの推定モデルによって、最適な性能情報パラメータを推測する。

【００５７】

また、コンピュータは、ライブラリの実際の実行の前に、例えばユーザからの基本情報パラメータの入力を検出することによって、基本情報パラメータが定まった時点を検出する。

【００５８】

ここで、基本情報パラメータとは、実行性能とライブラリの出力とを共に変化させるパラ

メータである。
【００５９】
例えば数値計算ライブラリのうちの、行列の固有値計算ライブラリにおいては、行列のサイズ、行列の実体などが、基本情報パラメータに相当する。また、例えば並列計算機を用いる場合のループアンローリング段数は、性能情報パラメータに相当する。
【００６０】
また、コンピュータは、ライブラリの実際の実行の際に、既に設定された性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしているか否かを試行により判別する。そして、所望の精度を満たしていないときには、基本情報パラメータを用いて性能情報パラメータの最適化を再度実行する。そして、所望の精度が得られる性能情報パラメータを用いて、ライブラリを実行する。
【００６１】
このように、実際の計算の前に、性能情報パラメータを設定しておく。実際の計算の際に、その性能情報パラメータによって所望の精度が得られる場合には、パラメータの最適化のための計算をせずに、ライブラリを実行できる。したがって、実行時におけるパラメータの最適化に要する時間を不要として、ライブラリの実行コスト（計算時間）を増大させない。また、ライブラリの実行の前に精度を確認するので、より精密かつ確実なパラメータ調整が可能となる。
【００６２】
なお、本発明に係るプログラムを、ソフトウェアのインストール時、およびソフトウェア実行時、のパラメタ最適化機能を有するソフトウェアである、と表現することもできる。
【００６３】
本発明に係るプログラムは、上記課題を解決するために、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの最適化を、上記コンピュータに実行させるためのプログラムにおいて、上記ライブラリのインストール時に上記性能情報パラメータの最適化を行う初期設定手順と、上記ライブラリの上記パラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う前調整手順と、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順とを含んでいることを特徴としている。
【００６４】
このプログラムは、コンピュータにおける例えば数値計算ライブラリのようなライブラリの実行コストを最適化するために用いられるプログラムである。実行コストとは、例えば実行に要する計算資源、計算時間である。このプログラムは、ライブラリのパラメータのうち、実行性能のみを変化させてライブラリの出力を変化させない性能情報パラメータの値を、ライブラリの実行コストが最適なものとなるように調整する。
【００６５】
上記プログラムが実行されたコンピュータは、ライブラリのインストール時に、性能情報パラメータの最適化を行う。この場合、例えば行列のサイズのような基本情報パラメータが定まっていないため、所定の誤差を含んだ、なんらかの推定モデルによって、最適な性能情報パラメータを推測する。
【００６６】
また、コンピュータは、ライブラリの実際の実行の前に、例えばユーザからの基本情報パラメータの入力を検出することによって、基本情報パラメータが定まった時点を検出する。
【００６７】
ここで、基本情報パラメータとは、実行性能とライブラリの出力とを共に変化させるパラ

メータである。

【００６８】

例えば数値計算ライブラリのうちの、行列の固有値計算ライブラリにおいては、行列のサイズ、行列の実体などが、基本情報パラメータに相当する。また、例えば並列計算機を用いる場合のループアンローリング段数は、性能情報パラメータに相当する。

【００６９】

その後、コンピュータは、ライブラリの実際の実行の前に、インストール時に設定された性能情報パラメータを参照して、基本情報パラメータを用いて性能情報パラメータの最適化を行う。より詳細には、例えば基本情報パラメータを用い、性能情報パラメータのそれぞれの値について試行計算を行って、実行コストを予め実測する。特に、インストール時に設定された性能情報パラメータの最適値周辺の値のみについて、試行計算を行うようにしてもよい。これによって、試行計算の回数を削減して、最適な性能情報パラメータを得ることができる。このように、より精密かつ確実なパラメータ調整が可能となる。

【００７０】

また、コンピュータは、ライブラリの実際の実行の際に、既に設定された性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしているか否かを試行により判別する。そして、所望の精度を満たしていないときには、基本情報パラメータを用いて性能情報パラメータの最適化を再度実行する。そして、所望の精度が得られる性能情報パラメータを用いて、ライブラリを実行する。

【００７１】

このように、実際の計算の前に、実行コストを予め実測して、最適な性能情報パラメータを得るようにする。基本情報パラメータの変更がないときには、予め設定した性能情報パラメータを用いてライブラリを実行できる。また、基本情報パラメータの変更があるときでも、所望の精度が得られる場合には、パラメータの最適化のための計算をせずに、ライブラリを実行できる。したがって、実行時におけるパラメータの最適化に要する時間を不要として、ライブラリの実行コスト（計算時間）を増大させない。また、ライブラリの実行の前に精度を確認するので、より精密かつ確実なパラメータ調整が可能となる。

【００７２】

なお、本発明に係るプログラムを、ソフトウェアのインストール時、ユーザが知りうる情報が定まった時点でのソフトウェアの実行前、およびソフトウェア実行時、の３階層のパラメタ最適化機能を有するソフトウェアである、と表現することもできる。

【００７３】

本発明に係るプログラムは、上記課題を解決するために、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータについて最適化する機能を上記コンピュータに実現させるためのプログラムにおいて、上記性能情報パラメータの各要素を、上記ライブラリのインストール時に最適化を行うパラメータの第１の集合、上記ライブラリの実行の前に最適化を行うパラメータの第２の集合、または上記ライブラリの実行の際に最適化を行うパラメータの第３の集合のうちの少なくとも一つに含まれるように設定して、第１の集合の要素を最適化する機能と、第２の集合の要素を最適化する機能と、第３の集合の要素を最適化する機能とを上記コンピュータに実現させることを特徴としている。

【００７４】

このプログラムは、コンピュータにおける例えば数値計算ライブラリのようなライブラリの実行コストを最適化するために用いられるプログラムである。実行コストとは、例えば実行に要する計算資源、計算時間である。このプログラムは、ライブラリのパラメータのうち、実行性能のみを変化させてライブラリの出力を変化させない性能情報パラメータの値を、ライブラリの実行コストが最適なものとなるように調整する。例えば数値計算ライブラリのうちの、行列の固有値計算ライブラリにおいては、並列計算機を用いる場合のループアンローリング段数が、性能情報パラメータに相当する。

【００７５】

上記プログラムが実行されたコンピュータにおいては、性能情報パラメータが、ライブラリのインストール時に最適化を行うパラメータの第１の集合、ライブラリの実行の前に最適化を行うパラメータの第２の集合、またはライブラリの実行の際に最適化を行うパラメータの第３の集合のうちの少なくとも一つに含まれるように設定される。

【００７６】

ここで、性能情報パラメータが何らかの意味で最適化可能であるならば、インストール時、ライブラリ実行前、ライブラリ実行の際のいずれかにおいて最適化することは、常に可能である。また、性能情報パラメータを、上述の第１～第３のうちから選択された少なくとも一つ以上の集合に含まれるように設定する具体的な構成には、ある程度任意性があるが、その構成はどのように選択してもよい。

【００７７】

そして、コンピュータは、第１～第３の集合について、それぞれ最適化を行う。したがって、性能情報パラメータの全てが最適化可能となり、汎用な処理に適用できる。すなわち、複数のルーチンを含んだライブラリ全体に対する最適化が可能となる。

【００７８】

一方、従来の最適化法は、ソフトウェアインストール時にパラメータ最適化を行うもの、またはライブラリ実行時にパラメータ最適化を行うもの、のいずれか一方しかなかった。このため、問題によっては、インストール時にしか最適化できない、または実行時にしか最適化できないものがあるので、全ての問題に対して汎用することができなかった。

【００７９】

なお、本発明に係るプログラムを、最適化すべきパラメタに関して、インストール時、実行前、実行時の３種のパラメタに分離し、それぞれのパラメタ最適化を行うソフトウェアである、と表現することもできる。

【００８０】

本発明に係る記録媒体は、上記課題を解決するための、上述のいずれかのプログラムを記録したコンピュータ読み取り可能な記録媒体である。

【００８１】

この記録媒体がコンピュータにて読取られると、上述のいずれかのプログラムがコンピュータにて実行される。したがって、上述のプログラムと同様の効果を得ることができる。

【００８２】

なお、記録媒体の構成としては、ハードディスク、CD ROM(Read Only Memory)などに限るものではなく、どのような記録媒体であってもよい。

【００８３】

また、本発明に係るコンピュータは、上記課題を解決するために、上述の記録媒体を備えている構成である。

【００８４】

このコンピュータにて上述の記録媒体を読み取りすると、上述のいずれかのプログラムがコンピュータにて実行される。したがって、上述のプログラムと同様の効果を得ることができる。

【００８５】

なお、このコンピュータは、コンピュータ内に複数のプロセッサを有する並列計算装置であってもよいし、または、複数のコンピュータがネットワークに接続されて複数のプロセッサを有する計算装置として機能する分散計算装置であってもよい。

【００８６】

また、上述のコンピュータは、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの調整を行う調整方法において、上記ライブラリの上記パラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する手順と、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う手順とを含んでいる調整方法を実行するものである、と表現することもできる。

【００８７】
また、上述のコンピュータは、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの調整を行う調整方法において、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順を含んでいる調整方法を実行するものである、と表現することもできる。
【００８８】
また、上述のコンピュータは、上記調整方法を実行することによって、コンピュータに備えられたライブラリのパラメータに含まれる、実行性能のみを変化させて上記ライブラリの出力を変化させない性能情報パラメータの最適化を行う調整装置として機能する。また、上述のコンピュータは、上述のプログラムとライブラリとを備えた計算装置として機能する。
【００８９】
なお、上述の構成において、性能情報パラメータの最適化とは、性能情報パラメータの全てを最適化するものではなく、最適化が可能なもののうち、適当なものについて最適化を行うことを意味する。
【００９０】
【発明の実施の形態】
本発明の一実施の形態について図１ないし図３に基づいて説明すると以下の通りである。
【００９１】
計算装置（コンピュータ）１は、図２に示すように、プロセッサ２、ユーザライブラリ（ライブラリ）３、パラメータ調整層４、およびパラメータ情報ファイル５を備えている。また、計算装置１は、図示しない記録媒体を備えている。計算装置１は、外部から入力されるパラメータを用いてライブラリ３中のサブルーチンを呼び出して、計算を行う。計算結果は図示しない表示装置に出力される。
【００９２】
プロセッサ２は、計算を行うため計算処理部である。プロセッサ２は、図示しないnprocs個のプロセッサを内部に備えている。計算装置１は、プロセッサ２の複数のプロセッサを用いて、並列計算装置として機能する。
【００９３】
ライブラリ３は、数値計算ライブラリである。ライブラリ３は少なくとも一つ以上のサブルーチンを含んでいる。本実施形態のライブラリ３は、内部に複数のサブルーチン３ａ～３ｋを備えている。
【００９４】
このライブラリ３やサブルーチン３ａ～３ｋには、なんらかの方法（専用記述言語など）を用いて、パラメータを記述してアクセスする。このパラメータのうちの一部は、外部からユーザによってライブラリ３へ直接入力される。また、パラメータの他の一部は、ライブラリ３内で使われる。また、パラメータのさらに他の一部は、パラメータ調整層４を介してライブラリ３に入力される。
【００９５】
このライブラリ３は、ユーザによって開発された数値計算ライブラリであるが、これに限るものではなく、例えばライブラリ開発者によって開発されたシステムライブラリであってもよい。このような、ＭＰＩ(Message Passing Interface)などの計算機環境やＯＳ（Operating System）などであらかじめ用意されているライブラリ等についても、ソフトウェアインタフェースさえ周知であれば、ユーザやライブラリ開発者がパラメータ記述を行うことによって、パラメータ調整層４にパラメータ情報を引き渡すことができる。
【００９６】
なお、ライブラリの備えるサブルーチンの内容、個数などについては、特に限定されない

。また、計算装置１には、ライブラリ以外のプログラムが備えられていてもよく、そのプログラムによって他の機能が実現されてもよい。

【００９７】

パラメータ調整層４は、ライブラリ３の用いるパラメータを調整する調整装置として機能する。パラメータ調整層４は、ライブラリ３に入力するパラメータの一部を調整した上で、ライブラリ３に入力する。パラメータ調整層４は、インストール時最適化層（Installation Optimization Layer:ＩＯＬ）４ａ、実行前最適化層（Before Execution-invocation Optimization Layer:ＢＥＯＬ）４ｂおよび実行時最適化層（Run-time Optimization Layer:ＲＯＬ）４ｃを含んでいる。これらの各層の機能については後述する。

【００９８】

パラメータ情報ファイル５は、パラメータ調整層４において調整されたパラメータを保存するためのファイルである。

【００９９】

本実施形態の計算装置１において、ライブラリ３は、図示しない記録媒体に記録されたプログラムが読み取られ、実行されることによって実現される機能である。また、パラメータ調整層４も、図示しない記録媒体に記録されたプログラムが読み取られ、実行されることによって実現される機能である。

【０１００】

上記構成の計算装置１について、以下でより詳細に説明する。

【０１０１】

計算装置１を用いてユーザがライブラリ３を実行する際には、所望のサブルーチン３ａに対して適当なパラメータを設定した上で実行指示をする。

【０１０２】

ここで、サブルーチン３ａに対して設定されるパラメータには、計算装置１の実行性能のみを変化させて、ライブラリ３のサブルーチン３ａの出力を変化させないパラメータが含まれる。以下では、このようなパラメータを、性能情報パラメータ(Performance Parameters ：ＰＰ)と呼ぶ。

【０１０３】

また、サブルーチン３ａに対して設定されるパラメータのうち、計算装置１の実行性能とライブラリ３のサブルーチン３ａの出力とを共に変化させるようなパラメータを、以下では基本情報(Basic Parameters ：ＢＰ)パラメータと呼ぶ。

【０１０４】

例えば、数値計算ライブラリに含まれるサブルーチン３ａが、行列の固有値を計算する固有値計算サブルーチンであるとする。このとき、所望の行列の実体や、その行列のサイズなどは、基本情報パラメータＢＰに相当する。また、計算装置１の行列計算におけるループアンローリング段数は、性能情報パラメータＰＰに相当する。

【０１０５】

計算装置１においては、与えられた基本情報パラメータＢＰを用いて、性能パラメータＰＰを最適化することによって、所望の結果を最小の時間で得ることができる。性能情報パラメータＰＰ、基本情報パラメータＢＰは、パラメータ調整層４を介してライブラリ３に入力される。性能情報パラメータＰＰおよび基本情報パラメータＢＰ以外のパラメータは、計算装置１の外部からライブラリ３に直接入力されるか、またはライブラリ３の内部で用いられる。

【０１０６】

本実施形態のパラメータ調整層４は、図１に示すように、調整可能なパラメータである性能情報パラメータＰＰを最適化するために、インストール時最適化層４ａ、実行前最適化層４ｂ、実行時最適化層４ｃの各層を備えている。各層４ａ～４ｃはパラメータを自身で保持することはなく、パラメータ情報ファイル５に保存する。

【０１０７】

インストール時最適化層（ＩＯＬ）４ａは、ライブラリ３のインストール時に最適化を行

う。
【０１０８】
インストール時最適化層４ａは、例えば図３（ａ）に示すように、ライブラリ３のインストール時に（Ｓ１）、性能情報パラメータＰＰのうちの一部であるインストール時最適化パラメータ（ＩＯＰ）を最適化し（Ｓ２）、得られたパラメータ（ＩＯＰ）をパラメータ情報ファイル５に出力する。
【０１０９】
なお、ライブラリ３のインストール時には、通常は、基本情報パラメータＢＰが定まっていることはない。このため、インストール時最適化層４ａは、例えば基本情報パラメータＢＰの値を適当にサンプリングして、そのサンプリングした抽出点ごとに、適当に定義したコスト定義関数を最小化するパラメータを決定する。そして、適当なモデル式によって、サンプリングした抽出点と抽出点との間のデータについて補間する。
【０１１０】
実行前最適化層（ＢＥＯＬ）４ｂは、ユーザが指定する特定パラメータ（例えば問題サイズなど）の指定後に最適化を行う。
【０１１１】
実行前最適化層４ｂは、基本情報パラメータＢＰの入力に応じて、これを用いて、性能情報パラメータＰＰのうちの一部である実行前最適化パラメータＢＥＯＰを最適化する。例えば図３（ｂ）に示すように、ユーザ指定パラメータとしての基本情報パラメータＢＰの定義（入力）に応じて（Ｓ４）、パラメータ情報ファイル５のパラメータ（ＩＯＰ）を参照して（Ｓ５）、最適化を行い（Ｓ６）、得られた最適化パラメータ（ＢＥＯＰ）をパラメータ情報ファイル５に出力する。
【０１１２】
なお、実行前最適化層４ｂは、ユーザによって指定された基本情報パラメータＢＰを用いて、最適なパラメータを得るために、実測にて試行をする。
【０１１３】
実行時最適化層（ＲＯＬ）４ｃは、インストール時最適化層４ａまたは実行前最適化層４ｂの少なくとも一方によるパラメータ最適化が終了した後で、かつ対象のライブラリ（やルーチン）の実行時に、最適化を行う。
【０１１４】
実行時最適化層４ｃは、例えば図３（ｃ）に示すように、ライブラリ３（ライブラリ３のサブルーチン３ａ）の実行指示を検出すると（Ｓ８）、既に設定された性能情報パラメータＰＰを参照して（Ｓ９）、この性能情報パラメータＰＰによる計算が所望の精度を満たしていないときには、最適化を再度行う（Ｓ１０）。Ｓ１０においては、計算が所望の精度を満たすような、最適なパラメータＰＰが得られるまで計算を繰り返す。
【０１１５】
このように、実行時最適化層４ｃは、既に設定された性能情報パラメータＰＰを参照して、例えば十分な精度が得られるような所定の場合には、最適化のための計算を行わない。
【０１１６】
以上のように、本実施形態のパラメータ調整層４においては、インストール時最適化層４ａにて最適化したパラメータ情報ＩＯＰは、パラメータ情報ファイル５に保存され、実行前最適化層４ｂと実行時最適化層４ｃとで参照可能となっている。また、実行前最適化層４ｂにて最適化したパラメータ情報ＢＥＯＰは、パラメータ情報ファイル５に保存され、実行時最適化層４ｃで参照可能となっている。
【０１１７】
ここで、性能情報パラメータＰＰの各要素は、パラメータ（ＩＯＰ）、パラメータ（ＢＥＯＰ）、パラメータ（ＲＯＰ）の各集合のうちの少なくとも一つに含まれている。すなわち、性能情報パラメータＰＰの各要素は、パラメータ調整層４の各層４ａ～４ｃのために、重複を許して、３つの部分集合（ＩＯＰ、ＢＥＯＰ、ＲＯＰ）に分解される。これを式で表現すると、以下のようになる。

ＰＰパラメータ＝ＩＯＰ　∪　ＢＥＯＰ　∪　ＲＯＰ　　…（式１）
したがって、本実施形態の計算装置１は、パラメータ調整層４を用いて、性能情報パラメータＰＰに含まれる全ての要素を、上述したタイミングのいずれかにて最適化できる。
【０１１８】
特に、本実施形態の計算装置１は、問題に応じた例えば行列サイズ（ｎ）のような基本情報パラメータＢＰが定まると、実際の計算の実行前の時点で最適化を行う実行前最適化層４ｂを備えている。これによって、従来の計算装置よりも正確な最適化が可能となる。
【０１１９】
〔実施例１〕
以下では、本実施形態の計算装置１について、数値計算ライブラリの具体的な一例を用いて説明する。また、この実施例１では、サブルーチン３ａのインストール時最適化層４ａによる最適化について説明する。
【０１２０】
ここでは一例として、ユーザライブラリ３が固有値計算ライブラリであり、サブルーチン３ａがサブルーチンPEigVecCalである場合について説明する。
【０１２１】
固有値計算ライブラリのサブルーチンPEigVecCalは、実数対称行列において固有値、固有ベクトルを求める際にしばしば用いられる、Householder二分・逆反復法によるサブルーチンを意味する。なお、この実施例では実行時間に関し最適化するので、最適化すべきコスト定義関数は実行時間である。
【０１２２】
サブルーチンPEigVecCalは、以下のような構成である。
call PEigVecCal(A, x, lambda, n, nproc, myid, iDistInd, imv, iud, ihit, icomm, kbi, kort, MAXITER, deps, …)
サブルーチンPEigVecCalにおける各引数は、パラメータ提示によるソフトウェア構成方式により提示された、ソフトウェアパラメータである。
【０１２３】
各パラメータについて説明すると、以下のようになる。
（ｉ）A, x, lambda, nの各パラメータは、行列情報などを表す基本情報パラメータＢＰに相当する。より詳細には、Aは固有値を求める対象となる行列の実体に相当する。xは、固有ベクトルに相当する。lambdaは固有値に相当する。nは行列のサイズに相当する。計算を行う際には、Aとnとを指定してサブルーチンPEigVecCalを呼び出すと、必要に応じてx,lambdaについての結果が得られるようになっている。
（ii）nproc, myid, iDistIndの各パラメータは、並列制御のためのパラメータである。例えばnprocは、プロセッサ２に含まれる、図示しない独立のプロセッサの数を表す。
（iii）imv, iud, ihit, icomm, kbi, kortは、アンローリング段数などの、処理手順に関連して性能に影響するパラメータである。
（iv）MAXITER, deps, …はアルゴリズムに影響するパラメータであり、以下では解法情報パラメータとよぶ。
このうち、(ii)(iii)のうちの一部が、性能情報パラメータ（ＰＰ）に相当する。
【０１２４】
より詳細には、Householder二分・逆反復法では、主要なソフトウェアパラメータは、基本情報パラメータＢＰ＝｛n｝、性能情報パラメータＰＰ＝｛imv, iud, ihit, kbi, kort, nproc]のようになる。
【０１２５】
なお、サブルーチンPEigVecCalは、より詳細には、以下の４種のサブルーチンと性能情報パラメータＰＰで構成されているとする。
・HosehoIder三重対角化ルーチン：　ＰＰ＝[imv, iud],
・二分法ルーチン：　ＰＰ＝｛kbi],
・逆反復法ルーチン：　ＰＰ＝｛kort],

・Householder逆変換ルーチン：　ＰＰ＝｛ihit]。
【０１２６】
ここで、上述した性能情報パラメータＰＰの詳細について説明する。各性能情報パラメータＰＰの定義域についても示す。
・imv＝[1,2,…,16]
imvは、Householder三重対角化で必要な行列・ベクトル積（２重ループ）のうち、最外ループアンローリング段数を指定するものである。
・iud＝[1,2,…,16]
iudは、HouseholIder三重対角化で必要な行列更新処理（２重ループ）のうち、最外ループアンローリング段数を指定するものである。
・kbi＝[vec, non-vec]
kbiは、二分法で必要な処理にっいて、ベクトル向きかそうでないか、の実装方式を指定するものである。
・kort＝[CG-S, MG-S, IRCG-S, NoOrt]
kortは、逆反復法中で密集固有値に対する固有ベクトル計算で必要な、再直交化処理の実装方式を指定するものである。より詳細には、CG-Sは、古典Gram-Schmidt法で固有ベクトルを再直交化することを意味する。また、MG-Sは、修正Gram-Schmidt法で固有ベクトルを再直交化することを意味する。また、IRCG-Sは、反復改良古典Gram-Schmidt法で固有ベクトルを再直交化することを意味する。また、NoOrtは、全く再直交化をしないことを意味する。
・ihit＝[1,2,…,16]
ihitは、Householder逆変換で必要な処理(2重ループ)のうち、最外ループアンローリング段数を指定するものである。
【０１２７】
以下では、固有値計算ライブラリPEigVecCalに対する、インストール時最適化層４ａによる最適化について説明する。
【０１２８】
本実施例では、インストール時最適化層４ａにおいて最適化するインストール時最適化パラメータＩＯＰを、ＩＯＰ＝｛ｉｍｖ，　ｉｕｄ，　ｋｂｉ，　ｉｈｉｔ｝とする。これらのパラメータ｛ｉｍｖ，　ｉｕｄ，　ｋｂｉ，　ｉｈｉｔ｝は、インストール先の計算機アーキテクチャやコンパイラなどの計算機環境が決まった時点で、その情報（レジスタ数、キャッシュサイズ、ベクトル機構など）から決まるパラメータである。このため、インストール時に最適化することが好ましい。
【０１２９】
パラメータＩＯＰの最適化について、パラメータiudを例にして説明する。他のパラメータについての最適化も同様であり、説明は省略する。
【０１３０】
まず、このインストール時においては、基本パラメータであるサイズnは定まっていない。そこで、問題サイズnに関して、適当なサンプリング点｛200，400，800，2000，4000，8000｝を定める。なお、プロセッサ台数nprocは８台と仮定する。
【０１３１】
また、パラメータiudについても、以下のサンプリング点｛1，2，3，4，8，16｝を定める。これは、定義域全体での計算は避けて、計算量を削減するためである。十分な計算時間を取ることができる場合には、定義域全体にわたって計算してもよい。
【０１３２】
そして、各サンプリング点において、適当に試行を行って、実行コストである計算時間を測定する。その後、各サンプリング点における値を補間するような、適当なコスト定義関数を決定する。これによって、定義域全域にわたるコストが推定できる。
【０１３３】
なお、この実施例においては、パラメータiudの最適化実行時間（＝コスト定義関数）を

、パラメータiudについて多項式近似する。なお、補間に用いる近似関数は、多項式近似に限るものではなく、他の関数を用いてもよい。

【０１３４】

並列計算機の一例を用いた、各サンプリング点における実行時間の測定結果（単位：秒）を以下の表１に示す。

【０１３５】

【表１】

n ＼ iud	1	2	3	4	8	16
200	.0628	.0628	.0629	.0623	.0621	.0625
400	.1817	.1784	.1763	.1745	.1723	.1719
800	.7379	.6896	.6638	.6550	.6369	.6309
2000	7.535	6.741	6.333	6.240	6.013	5.846
4000	54.06	48.05	44.85	44.36	42.89	41.19
8000	413.2	366.5	349.2	344.1	327.6	315.5

【０１３６】

この測定結果に対して、基本情報パラメータであるｎを固定して、パラメータiudに関する関数f_n（iud）を推定する。

【０１３７】

ここで、f_n(iud)として、５次の多項式f_n（iud）＝$a1\times iud^5+a2\times iud^4+a3\times iud^3+a4\times iud^2+a5\times iud+a6$ を仮定する。これに対して、適当な手法を用いて係数a1,…,a6を決定できる。ここでは、最小二乗法を用いて各係数を決定した。なお、各係数を決定する手法はこれに限るものではない。

【０１３８】

以下の表２に、表１のデータをサンプル点として最小二乗法を用いて係数を決定した結果を示す。

【０１３９】

【表２】

f_n	a1	a2	a3	a4	a5	a6
f_{200}	-2.2E-6	6.7E-5	-6.5E-4	2.4E-3	-3.8E-3	6.4E-2
f_{400}	-1.3E-6	4.2E-5	-4.6E-4	2.3E-3	-7.8E-3	1.8E-1
f_{800}	9.6E-6	-2.3E-4	7.9E-4	1.1E-2	-8.4E-2	8.1E-1
f_{2000}	2.4E-4	-6.3E-3	3.5E-2	1.1E-1	-1.3E-0	8.6E-0
f_{4000}	3.0E-3	-8.3E-2	6.1E-1	-4.9E-1	-7.7E+0	6.1E+1
f_{8000}	-1.3E-2	5.0E-1	-7.0E+0	4.5E+1	-1.4E+2	5.1E+2

【０１４０】

この表２から、iudの定義域｛1，2，…，16｝で最小となるiudの値を、サンプリングした

各問題サイズにおいて決定できる。

【０１４１】

また、問題サイズｎについては、以下のように補間を行う。

【０１４２】

まず、表２によりiud全ての領域について評価値を得ることができる。したがって定義域{1，2，…，16}全てにおいて、iudを固定し問題サイズｎをサンプル点[200，400，800，2000，4000，8000]だけ変化させた評価値を計算できる。

【０１４３】

そこで、これらの評価値を新たなサンプル点とみなして、関数f_{iud}（n）を最小二乗法により推定する。$f_{iud}(n)$について、５次多項式$f_{iud}(n)=a'1\times n^5+a'2\times n^4+a'3\times n^3+a'4\times n^2+a'5\times n+a'6$を仮定する。ｎに関するサンプル点[200，400，800，2000，4000，8000]だけ変化させた評価値による結果を求める。その結果から、ｎの関数$f_{iud}(n)$がiudに関する定義域{1，2，…，16}で定まるので、実行時に指定されたｎを代入することで、最小となるiudが決定できる。

【０１４４】

このように、サンプルされた問題サイズｎに関して、実行時に全く同じ値が指定される保証はないので、上述のように最適なパラメータを推定する。推定したパラメータをパラメータ情報ファイル５に保存しておく。また、推定するパラメータのための情報、ここでは例えば各係数なども、パラメータ情報ファイル５に保存しておく。これによって、後の最適化において、パラメータ情報ファイル５を参照して、情報を得ることができる。

【０１４５】

〔実施例２〕

次に、実施例１にて説明したサブルーチンPEigVecCalに対する、実行前最適化層４ｂによる最適化について説明する。

【０１４６】

本実施例においては、基本情報パラメータである問題サイズｎが、ｎ=8192と定まったとする。なお、プロセッサ台数nprocsは4であるとする。

【０１４７】

ここで、実行前最適化層４ｂによって最適化する、実行前最適化パラメータ（ＢＥＯＰ）として、ＢＥＯＰ＝{imv，iud，ihit，kbi}とする。

【０１４８】

このように、本実施例においては、ＢＥＯＰは上述の実施例１におけるＩＯＰと同じとなる。しかしながら、本実施例においては、基本情報パラメータｎが定まった後に最適化を行うので、上述のような補間を行う必要がなく、ＢＥＯＰについて実測した確実な最適値を得ることができるという違いがある。

【０１４９】

すなわち、インストール時における最適化においては、サイズｎに関するサンプル標本点以外は、補間などによる推定でコスト定義関数のパラメータ決定をしていた。また、プロセッサ数nprocの値は仮定した値を用いていた。

【０１５０】

このように、例えばインストール時のみに最適化を行う従来の構成では、実行前最適化層がないため、推定値からパラメータを決定するしかない。

【０１５１】

一方、本発明による、実行前における最適化では、所望のサイズｎについて、実測でパラメータ決定をする。このため、実行前最適化によって、インストール時の最適化よりもパラメータの精度を高めることができる。したがって、実行前最適化層４ｂによる最適化は、パラメータ推定に誤差があると致命的になるような場合であっても、用いることができる。また、例えばパラメータ情報ファイル５を参照して、インストール時最適化の結果を利用して、計算時間を削減することもできる。

【０１５２】

なお、この実行前における最適化は、実際に用いる所望のサイズｎについて、例えば上述した実施例１の表１と同様に実測し、表２のように係数を得て、最適なiudを求めることによって行われる。手順の詳細については実施例１と同様であるので、省略する。

【０１５３】

以上の実施例から、本発明を実施することで従来よりも高度なパラメータ調整機構が提供される。

【０１５４】

なお、従来のインストール時最適化と本発明における実行前最適化層４ｂの機能の違いは、以下のようなものである。

【０１５５】

【表３】

	非特許文献１	本願
最適化のタイミング	ソフトウェアインストール時	ソフトウェアインストール時 最適化終了後
処理の概略	実行時に指定される性能情報パラメタＢＰを予想して、パラメタを最適化	確実に指定される性能情報パラメタＢＰを指定して、パラメタを最適化
実行時のＢＰ指定が異なる場合の処理	何らかの方法で、近い値を推定して設定する	ユーザがパラメタＢＰの指定を保証するので、ありえない

【０１５６】

〔実施例３〕

次に、実施例１、２にて説明したサブルーチンPEigVecCalに対する、実行前最適化層４ｂによる最適化の他の例について説明する。

【０１５７】

ここでは、ユーザが係数行列について、ライブラリコールの時点で変化しない、という情報を知っているとする。すなわち、問題サイズｎについては確定しているものとする。

【０１５８】

このとき、実行前最適化層４ｂによって最適化する実行前最適化パラメータＢＥＯＰとして、ＢＥＯＰ＝[imv, iud, ihit, kbi, kort]とする。すなわちこの問題の場合、固有値問題において、逆反復法での直交化処理（パラメータkort）まで最適化できる。また、この実施例においては、プロセッサ数nprocsについても最適化する。

【０１５９】

なお従来法では、実行前最適化層がないため、本実施例ではパラメータ最適化が適用できない。

【０１６０】

ここでは、行列（Frank行列）のサイズがｎ=10,000と与えられたとする。並列計算機の一例を用いて、パラメータkort，プロセッサ数nprocsについて実測を行った実行時間（単位：秒）の結果を、以下の以下の表４に示す。なお、記号>は、所定の制約時間中に収束せず、実行が終了しなかったことを示す。

【０１６１】

【表４】

nprocs	CG-S	MG-S	IRCG-S	NoOrt
８	6,604	38,854	12,883	23
１６	3,646	24,398	6,987	12
３２	3,061	28,050	3,906	7
６４	1,633	27,960	3,059	3
１２８	2,091	>	3,978	1

【０１６２】
また、以下の表５には、逆反復法での各直交化方式による、固有ベクトルの直交精度を示す。単位は、Frobeniusノルムであり、８ＰＥのＭＧ－Ｓにおける最大残差ベクトル$\max_i(|(Ax)_i - \mathrm{lambda}_i\, x_i|^2)$=1.61E-7とする。
【０１６３】
【表５】

nprocs	CG-S	MG-S	IRCG-S	NoOrt
８	6.4E-13	6.6E-13	6.4E-13	1.4
１６	6.6E-13	6.6E-13	6.6E-13	1.4
３２	6.8E-13	6.6E-13	6.8E-13	1.4
６４	9.4E-13	6.6E-13	9.4E-13	1.4
１２８	1.5E-12	時間切れ	1.5E-12	1.4

【０１６４】
表４と表５とから、本実施例では、直交化方式の違いにより実行時間が異なるが、直交精度が1.5E-12以下であるなら、CG-S法が速度と精度の観点からよいことが分かる。したがって、例えばユーザが直交精度の上界をシステムに引き渡せば、ＢＥＯＬによってパラメータkortをCG-Sに固定できる。また、最適なプロセッサ数nprocsについても確定できる。
【０１６５】
以上の実施例から、本発明を適用することで従来よりも高度なパラメータ調整機構が提供される。
【０１６６】
また、本実施例における計算では、パラメータ情報ファイル５に保存された情報を参照して、計算量を削減してもよい。
【０１６７】
〔実施例４〕
次に、実施例１～３にて説明したサブルーチンPEigVecCalに対する、実行時最適化層４ｃによる最適化の例について説明する。
【０１６８】
ここで、この実施例４においては、行列の実体Aや、行列サイズｎが変化しうる状態であるとする。すなわち、行列サイズや行列データは実行時に固定されないとする。このような場合には、最適な直交化方式は、ユーザの与えた条件と実行時にならないと固定されない行列の特性とに、実際には依存する。
【０１６９】

ここで、実行時最適化層４ｃによって最適化する実行時最適化パラメータＲＯＰを、ＲＯＰ＝｛kort｝とする。

【０１７０】

また、実施例３にて説明した実行前最適化層４ｂによる最適化によって、過去の直交化適用例としてユーザの精度要求と合致するパラメータkortが、最適パラメータとしてパラメータ情報ファイル５に保存されているとする。

【０１７１】

そこで、実行時最適化層４ｃにおいては、まずパラメータ情報ファイル５に保存されているパラメータkortを参照して、最もよさそうな直交化方式を選ぶ。

【０１７２】

次に、実行時最適化層４ｃは、計算の精度がユーザ指定の基準を満たすか否かを判別する。そして、指定された精度を満たしていないときには、実施例３にて説明したように、パラメータkortの各値について実測を行って、パラメータの再調整を行う。そして、指定された精度が得られる、最適なパラメータkortが選択できるまで、判別と計算とを繰り返す。これによって、ユーザ指定の精度をシステム側で保証することができる。なお、計算の詳細については上述の実施例３と同様であるのでここでは省略する。

【０１７３】

なお、従来法においては、行列サイズや行列データが実行時に固定されないならば、アルゴリズム上の理由から最適パラメータは決定できない。一般的に従来法では、精度に関して保証するため、コストにかかわらずMG-Sを強制選択する場合が多い。この場合には、上述の表４にて示したように、コストが非常に不利になる。

【０１７４】

一方、本発明では、システムにユーザから与えたれた情報（直交精度など）を引き渡すことで、上述の場合においてもパラメータ調整が適用可能となる。また、コストについても最適化が可能となる

以上の実施例では、本発明特有の実行前最適化層４ｂと実行時最適化層４ｃとが、ソフトウェア構成方式として存在しないとできない。したがって本発明は、従来法に対してパラメータ調整の適用範囲が広い。

【０１７５】

なお、ここでは、このパラメータ情報ファイル５に保存された、実行前に最適化された情報を参照する構成について説明したが、可能であれば、パラメータ情報ファイル５に保存された、インストール時に最適化された情報を参照する構成であってもよい。このように、インストール時最適化層４ａと実行時最適化層４ｃとの組合せによって実現することも可能である。

【０１７６】

以上の各実施例にて説明したように、本実施形態に係るプログラムは、基本情報パラメータＢＰが定まると、それに応じて性能方法パラメータの最適化を、実際のライブラリの実行前に行う構成である。

【０１７７】

したがって、インストール時に最適化したパラメータをより精密に再調整することができ、または実行時に最適化する際の計算時間を削減して十分な最適化時間を確保することができる。これによって、より精密かつ確実なパラメータ調整が可能となる。

【０１７８】

また、性能情報パラメータＰＰの各要素は、インストール時、ライブラリ実行前、ライブラリ実行時のいずれかにおいて最適化されるようになっている。すなわち、また、インストール時、ライブラリ実行時に加えて、ライブラリ実行前においても最適化を行うので、あらゆる問題が最適化できる、汎用性が高いパラメータ調整機能を提供できる。

【０１７９】

なお、上述の実施の形態においては、並列計算装置としての計算装置１について説明をしたが、本発明はこれに限るものではなく、プロセッサ２がネットワークにて接続された複

数の計算装置に備えられたものである分散計算装置であってもよい。
【０１８０】
また、上述の実施の形態においては、ライブラリが固有値計算ライブラリであり、サブルーチンがサブルーチンPEigVecCalである場合についてのみ説明を行ったが、これに限るものではなく、他のライブラリ、サブルーチンについても適用できるのはもちろんである。
【０１８１】
また、以上のように、この発明は、性能やコスト等に関するソフトウェア上のパラメータを自動調整するソフトウェア（自動チューニングソフトウェア）において、性能等の諸コストを考慮しパラメータ調整を行う機構があり、かつその調整機構の適用に関し範囲が広いソフトウェア構成方式に関するものである。また、本発明は、インストール時、実行前、実行時の最適化層を有するソフトウェア構成方式に関するものである。また、本発明では上述の式１のように、パラメータを３種類に分離して用いる。
【０１８２】
ここで、従来の自動チューニングソフトウェアの構成方式では、例えば図４（ａ）に示すようにソフトウェアインストール時にパラメータ最適化を行うもの、または例えば図４（ｂ）に示すようにライブラリ実行時にパラメータ最適化を行うもの、のみ存在していた。これらのソフトウェア構成方式では、汎用的な処理に適用できない、パラメータ調整が不十分となる場合がある、という問題がある。また図４（ａ）（ｂ）から分かるように、従来の自動チューニングではパラメータは１種類であった。
【０１８３】
そこで本発明においては、より汎用的な処理においてパラメータ調整が適用でき、かつ従来よりも高度なパラメータ調整機構を有するソフトウェア構成方式によって課題の解決をねらうものである。
【０１８４】
特に、本実施形態の計算装置１は、問題に応じた例えば行列サイズ（ｎ）のような基本情報パラメータＢＰが定まると、実際の計算の実行前の時点で最適化を行う実行前最適化層４ｂを備えている。これによって、従来の計算装置のような、ＩＯＬ、またはＲＯＬ単独の場合よりもより正確な最適化が可能となる。
【０１８５】
なお、従来の技術における非特許文献１（Ｐ６２）には、自動チューニングとして『(i)実行時自動チューニング(ii)実行前自動チューニング』の二つがある点が記載されている。しかしながら、この非特許文献１における『実行前自動チューニング』は、本発明における上述の『実行前自動チューニング』とは異なるものであり、本発明においては『インストール時最適化』に相当するものである。
【０１８６】
本発明は上述した実施形態、実施例に限定されるものではなく、請求項に示した範囲で種々の変更が可能であり、異なる実施例にそれぞれ開示された技術的手段を適宜組み合わせて得られる実施形態についても、本発明の技術的範囲に含まれる。
【０１８７】
上述の具体的な実施形態または実施例は、あくまでも、本発明の技術内容を明らかにするものであって、本発明はそのような具体例にのみ限定して狭義に解釈されるべきものではなく、特許請求の範囲に示した範囲で種々の変更が可能であり、変更した形態も本発明の技術的範囲に含まれる。
【０１８８】
【発明の効果】
本発明に係るプログラムは、以上のように、ライブラリのパラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する手順と、上記基本情報パラメータを用いて性能情報パラメータの最適化を行う手順とを含んでいる構成である。
【０１８９】

それゆえ、実際の計算の前に実行コストを予め実測して最適な性能情報パラメータを得るようにして、より精密かつ確実なパラメータ調整が可能となるという効果を奏する。

【０１９０】

本発明に係るプログラムは、以上のように、ライブラリのインストール時に性能情報パラメータの最適化を行う初期設定手順と、上記ライブラリのパラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記初期設定手順において設定された上記性能情報パラメータを参照して、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う前調整手順とを含んでいる構成である。

【０１９１】

それゆえ、ライブラリの実際の実行の前に、インストール時に設定された性能情報パラメータを参照して、基本情報パラメータを用いて性能情報パラメータの最適化を行うので、試行計算の回数を削減して最適な性能情報パラメータを得ることができるという効果を奏する。

【０１９２】

本発明に係るプログラムは、以上のように、ライブラリのパラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記基本情報パラメータを用いて性能情報パラメータの最適化を行う前調整手順と、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順とを含んでいる構成である。

【０１９３】

それゆえ、再調整手順にて精度を確認するので、所望の精度が得られる場合には、パラメータの最適化のための計算をせずに、ライブラリを実行できるという効果を奏する。

【０１９４】

本発明に係るプログラムは、以上のように、ライブラリのインストール時に性能情報パラメータの最適化を行う初期設定手順と、上記ライブラリのパラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順とを含んでいる構成である。

【０１９５】

それゆえ、ライブラリの実際の実行の際に、既に設定された性能情報パラメータによって所望の精度が得られる場合には、パラメータの最適化のための計算をせずに、ライブラリを実行できるという効果を奏する。

【０１９６】

本発明に係るプログラムは、以上のように、ライブラリのインストール時に性能情報パラメータの最適化を行う初期設定手順と、上記ライブラリのパラメータに含まれる、実行性能と上記ライブラリの出力とを共に変化させる基本情報パラメータが定まった時点を検出する検出手順と、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を行う前調整手順と、上記ライブラリの実行の際に、既に設定された上記性能情報パラメータを参照して、この性能情報パラメータによる計算が所望の精度を満たしていないときには、上記基本情報パラメータを用いて上記性能情報パラメータの最適化を再度行う再調整手順とを含んでいる構成である。

【０１９７】

それゆえ、ライブラリの実際の実行の前に、最適な性能情報パラメータを得ることができるという効果を奏する。また、ライブラリの実際の実行の際に、既に設定された性能情報パラメータによって所望の精度が得られる場合には、パラメータの最適化のための計算を

せずに、ライブラリを実行できるという効果を奏する。

【０１９８】

本発明に係るプログラムは、以上のように、性能情報パラメータの各要素を、ライブラリのインストール時に最適化を行うパラメータの第１の集合、上記ライブラリの実行の前に最適化を行うパラメータの第２の集合、または上記ライブラリの実行の際に最適化を行うパラメータの第３の集合のうちの少なくとも一つに含まれるように設定して、第１の集合の要素を最適化する機能と、第２の集合の要素を最適化する機能と、第３の集合の要素を最適化する機能とを上記コンピュータに実現させる構成である。

【０１９９】

それゆえ、性能情報パラメータを、インストール時、ライブラリ実行前、ライブラリ実行の際のいずれかにおいて最適化するので、性能情報パラメータの全てが最適化可能となり、汎用な処理に適用できるという効果を奏する。

【０２００】

本発明に係る記録媒体は、以上のように、上述のいずれかのプログラムを記録したコンピュータ読み取り可能な記録媒体である。

【０２０１】

それゆえ、上述のプログラムと同様の効果を奏する。

【０２０２】

また、本発明に係るコンピュータは、以上のように、上述の記録媒体を備えている構成である。

【０２０３】

それゆえ、上述のプログラムと同様の効果を奏する。

【図面の簡単な説明】

【図１】本発明に係るコンピュータの一実施形態の一部を示すブロック図である。

【図２】上記コンピュータを示すブロック図である。

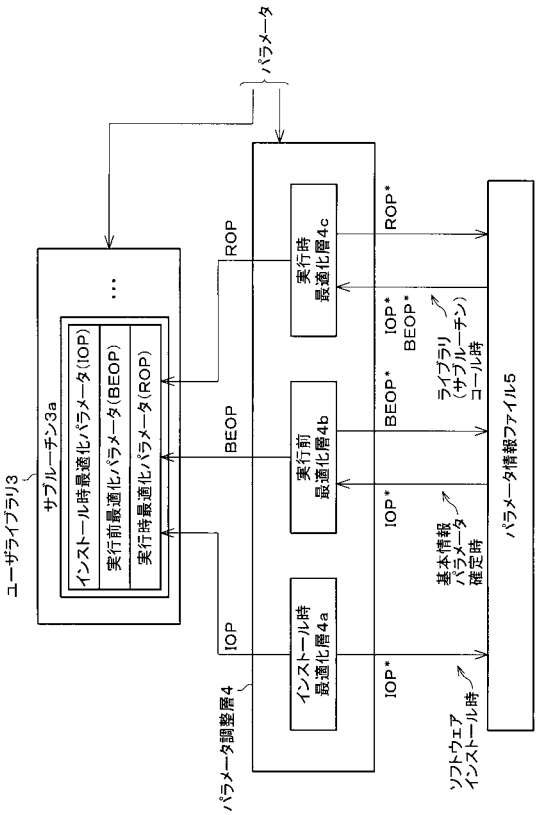
【図３】（ａ）はインストール時最適化の手順を示すフローチャートであり、（ｂ）はライブラリ実行前最適化の手順を示すフローチャートであり、（ｃ）はライブラリ実行時最適化の手順を示すフローチャートである。

【図４】（ａ）は従来のコンピュータの一例の一部を示すブロック図であり、（ｂ）は従来のコンピュータの他の一例の一部を示すブロック図である。

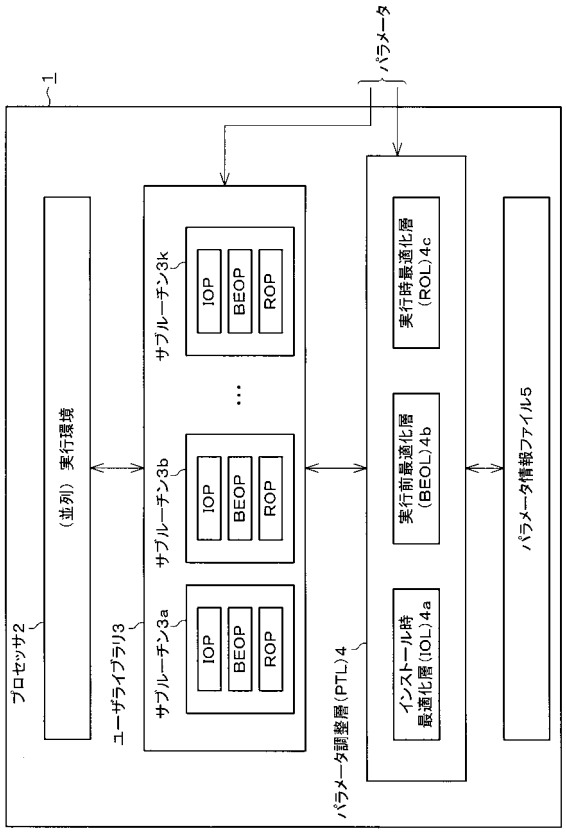
【符号の説明】

１	計算装置（コンピュータ）
２	プロセッサ
３	ユーザライブラリ（ライブラリ）
４	パラメータ調整層
４ａ	インストール時最適化層
４ｂ	実行前最適化層
４ｃ	実行時最適化層
５	パラメータ情報ファイル
ＩＯＰ	インストール時最適化パラメータ（性能情報パラメータ）
ＢＥＯＰ	実行前最適化パラメータ（性能情報パラメータ）
ＲＯＰ	実行時最適化パラメータ（性能情報パラメータ）

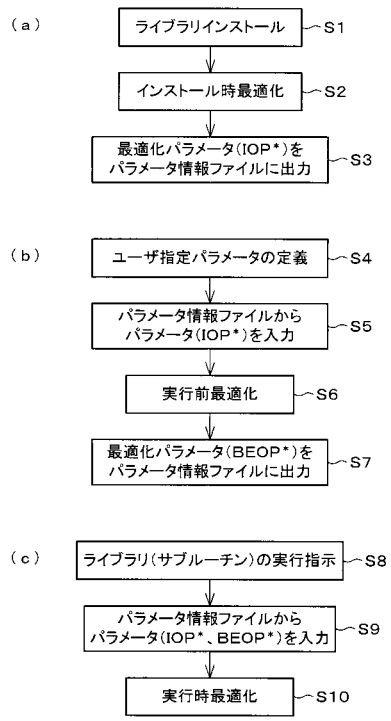
【図１】


パラメータ
ユーザライブラリ3
サブルーチン3a
インストール時最適化パラメータ(IOP)
実行前最適化パラメータ(BEOP)
実行時最適化パラメータ(ROP)
ROP
BEOP
IOP
パラメータ調整層4
実行時最適化層4c
実行前最適化層4b
インストール時最適化層4a
ROP*
IOP*
BEOP*
ライブラリ(サブルーチン)コール時
基本情報パラメータ確定時
ソフトウェアインストール時
パラメータ情報ファイル5

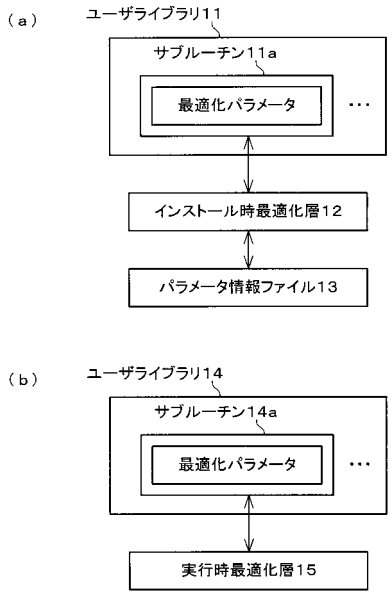
【図２】


1
パラメータ
プロセッサ2
(並列) 実行環境
ユーザライブラリ3
サブルーチン3a
サブルーチン3b
サブルーチン3k
IOP
BEOP
ROP
パラメータ調整層(PTL)4
インストール時最適化層(IOL)4a
実行前最適化層(BEOL)4b
実行時最適化層(ROL)4c
パラメータ情報ファイル5

【図３】


(a)
ライブラリインストール S1
インストール時最適化 S2
最適化パラメータ(IOP*)をパラメータ情報ファイルに出力 S3
(b)
ユーザ指定パラメータの定義 S4
パラメータ情報ファイルからパラメータ(IOP*)を入力 S5
実行前最適化 S6
最適化パラメータ(BEOP*)をパラメータ情報ファイルに出力 S7
(c)
ライブラリ(サブルーチン)の実行指示 S8
パラメータ情報ファイルからパラメータ(IOP*、BEOP*)を入力 S9
実行時最適化 S10

【図４】


(a)
ユーザライブラリ11
サブルーチン11a
最適化パラメータ
インストール時最適化層12
パラメータ情報ファイル13
(b)
ユーザライブラリ14
サブルーチン14a
最適化パラメータ
実行時最適化層15

フロントページの続き

(56)参考文献　特開２０００－２７６４５４号公報（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/06 620K