(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6677262号
(P6677262)
(45)発行日　令和2年4月8日(2020.4.8)　(24)登録日　令和2年3月17日(2020.3.17)

(51)Int.Cl.　F I
G06F 8/41 (2018.01)　G06F 8/41 130

請求項の数 10 (全 30 頁)

(21)出願番号　特願2017-560367(P2017-560367)
(86)(22)出願日　平成28年12月28日(2016.12.28)
(86)国際出願番号　PCT/JP2016/089022
(87)国際公開番号　WO2017/119378
(87)国際公開日　平成29年7月13日(2017.7.13)
審査請求日　令和1年11月15日(2019.11.15)
(31)優先権主張番号　特願2016-46(P2016-46)
(32)優先日　平成28年1月4日(2016.1.4)
(33)優先権主張国・地域又は機関
日本国(JP)

(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　宮本　孝道
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　坂庭　剛史

最終頁に続く

(54)【発明の名称】プログラム変換装置、プログラム変換方法、プログラム変換プログラム

(57)【特許請求の範囲】
【請求項１】
第１処理が順番に処理される第１ループ処理が複数回、繰り返される第２ループ処理が含まれるプログラムであって、前記順番に対して前記第１処理にて連続していない記憶領域にアクセスする処理を前記第１ループ処理が含み、さらに、条件が成り立つか否かに応じて前記第２ループ処理を終了する判定処理を前記第２ループ処理が含む場合に、前記第２ループ処理に関する繰り返し処理が第１回数分に亘って、繰り返される第３ループ処理と、前記第３ループ処理にて第２回数分に亘って、繰り返される第４ループ処理とに変換するプログラム変換手段と、
前記第１ループ処理が前記第２回数分に亘って、繰り返される処理と、前記判定処理が前記第２回数分、繰り返される処理とが前記第１回数分に亘って、繰り返される処理に変換するループ分割手段と、
前記第１処理及び前記判定処理を、前記第４ループ処理ごとに異なる記憶領域であって、前記第４ループ処理における処理順に連続している前記記憶領域にアクセスされる処理に変換する変数並び替え手段と、
前記第４ループ処理と、前記第１ループ処理との処理順序を入れ替える処理入れ替え手段と
を備えるプログラム変換装置。
【請求項２】
前記第２ループ処理において、前記第１処理が実行される場合に経由する前記判定処理

の回数を算出する効率算出手段と、
前記判定処理の回数が所定の条件を満たしている前記第１ループ処理を、前記プログラムから選択するループ選択手段と
をさらに備え、
前記プログラム変換手段は、前記ループ選択手段によって選択された前記第１ループ処理を含む前記第２ループ処理を変換する
請求項１に記載のプログラム変換装置。
【請求項３】
前記効率算出手段は、前記第２ループ処理に含まれている各ループ処理に関して、前記各ループ処理において連続している前記記憶領域にアクセスするか否かを判定し、アクセスしない場合に比べアクセスする場合に小さな値であり、かつ、前記判定処理の回数が多いほど小さな値である評価値を算出し、
前記ループ選択手段は、前記評価値が所定の条件を満たしている前記第１ループ処理を選択する
請求項２に記載のプログラム変換装置。
【請求項４】
前記プログラムが実行される情報処理装置が有しているキャッシュメモリの大きさを、前記第１ループ処理にてアクセスされるデータの大きさの合計値にて割り算された値に基づき、前記第２回数を算出する分配数決定手段
をさらに備え、
前記プログラム変換手段は、前記分配数決定手段が算出した前記第２回数に従い変換する
請求項１乃至請求項３のいずれかに記載のプログラム変換装置。
【請求項５】
前記第２回数として、前記評価値の大小に応じた値を算出する分配数決定手段
をさらに備え、
前記プログラム変換手段は、前記分配数決定手段が算出した前記第２回数に従い変換する
請求項３に記載のプログラム変換装置。
【請求項６】
前記プログラムが実行される情報処理装置が有しているキャッシュメモリの大きさを、前記第１ループ処理にてアクセスされるデータの大きさの合計値にて割り算された値を表す第１分配数と、前記評価値の大小に応じた値を表す第２分配数とを算出する分配数決定手段
をさらに備え、
前記プログラム変換手段は、前記プログラムに含まれている前記第３ループ処理及び前記第４ループ処理に変換される場合の反復回数を含む指示情報に基づき、前記第２ループ処理を、前記反復回数分に亘って、繰り返される前記第３ループ処理及び前記第３ループ処理にて前記第１分配数分、繰り返される前記第４ループ処理と、前記反復回数分に亘って、繰り返された後に、前記プログラムにて指示された回数まで繰り返される前記第３ループ処理、及び、前記第３ループ処理にて前記第２分配数分、繰り返される前記第４ループ処理とに変換する
請求項３に記載のプログラム変換装置。
【請求項７】
情報処理装置によって、第１処理が順番に処理される第１ループ処理が複数回繰り返される第２ループ処理が含まれるプログラムであって、前記順番に対して前記第１処理にて連続していない記憶領域にアクセスする処理を前記第１ループ処理が含み、さらに、条件が成り立つか否かに応じて前記第２ループ処理を終了する判定処理を前記第２ループ処理が含む場合に、前記第２ループ処理に関する繰り返し処理が第１回数分に亘って、繰り返される第３ループ処理と、前記第３ループ処理にて第２回数分に亘って、繰り返される第

４ループ処理とに変換し、
前記第１ループ処理が前記第２回数分に亘って、繰り返される処理と、前記判定処理が前記第２回数分に亘って、繰り返される処理とが前記第１回数分に亘って、繰り返される処理に変換し、
前記第１処理及び前記判定処理を、前記第４ループ処理ごとに異なる記憶領域であって、前記第４ループ処理における処理順に連続している前記記憶領域にアクセスされる処理に変換し、
前記第４ループ処理と、前記第１ループ処理との処理順序を入れ替える
プログラム変換方法。
【請求項８】
第１処理が順番に処理される第１ループ処理が複数回繰り返される第２ループ処理が含まれるプログラムであって、前記順番に対して前記第１処理にて連続していない記憶領域にアクセスする処理を前記第１ループ処理が含み、さらに、条件が成り立つか否かに応じて前記第２ループ処理を終了する判定処理を前記第２ループ処理が含む場合に、前記第２ループ処理に関する繰り返し処理が第１回数分に亘って、繰り返される第３ループ処理と、前記第３ループ処理にて第２回数分に亘って、繰り返される第４ループ処理とに変換するプログラム変換機能と、
前記第１ループ処理が前記第２回数分に亘って、繰り返される処理と、前記判定処理が前記第２回数分に亘って、繰り返される処理とが前記第１回数分に亘って、繰り返される処理に変換するループ分割機能と、
前記第１処理及び前記判定処理を、前記第４ループ処理ごとに異なる記憶領域であって、前記第４ループ処理における処理順に連続している前記記憶領域にアクセスされる処理に変換する変数並び替え機能と、
前記第４ループ処理と、前記第１ループ処理との処理順序を入れ替える処理入れ替え機能と
をコンピュータに実現させるプログラム変換プログラム。
【請求項９】
前記第２ループ処理において、前記第１処理が実行される場合に経由する前記判定処理の回数を算出する効率算出機能と、
前記判定処理の回数が所定の条件を満たしている前記第１ループ処理を、前記プログラムから選択するループ選択機能と
をさらにコンピュータに実現させ、
前記プログラム変換機能においては、前記ループ選択機能によって選択された前記第１ループ処理を含む前記第２ループ処理を変換する
請求項８に記載のプログラム変換プログラム。
【請求項１０】
前記効率算出機能においては、前記第２ループ処理に含まれている各ループ処理に関して、前記各ループ処理において連続している前記記憶領域にアクセスするか否かを判定し、アクセスしない場合に比べアクセスする場合に小さな値であり、かつ、前記判定処理の回数が多いほど小さな値である評価値を算出し、
前記ループ選択機能においては、前記評価値が所定の条件を満たしている前記第１ループ処理を選択する
請求項９に記載のプログラム変換プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、たとえば、変換対象であるコンピュータプログラムを実行効率がよくなるように変換するプログラム変換装置等に関する。
【背景技術】
【０００２】

ベクトル演算器は、ある演算処理を複数の数値に適用する処理を一度に処理することによって、該演算処理を高速に実行する。ある演算処理は、たとえば、コンピュータプログラムにおいて、ループ処理等の繰り返し処理に現れる。以降、コンピュータプログラムを、単に、「プログラム」と表す。

【０００３】

しかし、ベクトル演算器を有する情報処理装置は、必ずしも、入力されたプログラムを高効率に処理することはできない。この理由は、該入力されたプログラムが、ベクトル演算器が処理可能なプログラムに変換されていないからである。たとえば、特許文献１、または、特許文献２には、入力されたプログラムを、ベクトル演算器を有する情報処理装置が高効率に処理することができるプログラムに変換するコンパイラが開示されている。

【０００４】

特許文献１に開示されたコンパイラは、ソースプログラムに含まれている各ループ処理に関して、変数や配列の参照関係（依存関係）を解析する。該コンパイラは、各ループ処理に関して、スカラ逐次処理、ベクトル逐次処理、スカラ並列処理、または、ベクトル並列処理が実行された場合の処理時間（所要時間）を推定し、推定した処理時間のうち最も処理時間が短い組み合わせを選択する。該コンパイラは、選択した組み合わせに従って、該ソースプログラムをオブジェクトプログラムに変換する。

【０００５】

特許文献２に開示されたコンパイラは、多重ループを複数含むソースプログラムを、ベクトル演算器にて高効率に実行可能なオブジェクトプログラムに変換する。該コンパイラは、該多重ループを構成している外側のループを分配する分配先である内側のループを選択する。該コンパイラは、該多重ループを構成している１つのループに関するループ変数にて参照されている配列の次元を多次元に変換し、該多次元の変数配列に関して、内側のループと、分配されたループとを入れ替えることによって、該ソースプログラムを、実行効率が高いオブジェクトプログラムに変換する。

【先行技術文献】

【特許文献】

【０００６】

【特許文献１】特開平０４－２９３１５０号公報

【特許文献２】特開２００３－２５６２１４号公報

【発明の概要】

【発明が解決しようとする課題】

【０００７】

しかし、特許文献１、または、特許文献２に開示されたいずれのコンパイラも、ループにおける反復回数に応じた値を記憶可能なループ変数とは異なる変数（便宜上、「参照変数」と表す）にて記憶領域が参照されているプログラムの場合には、実行効率がよいオブジェクトプログラムに変換することはできない。この理由は、これらのコンパイラが、参照変数の値にて指し示される記憶領域が参照されるループに対して、連続していない記憶領域を参照することによって該ループ処理がベクトル化されたオブジェクトプログラムに変換してしまうからである。この理由について詳細に説明する。

【０００８】

ベクトル演算器は、メインメモリ（以降、単に、「メモリ」と表す）に格納されているデータを読み取り、読み取ったデータを該ベクトル演算器が有するベクトルレジスタに格納し、該ベクトルレジスタに格納されているデータに対して演算処理を実行する。ベクトルレジスタは、たとえば、メモリから読み取られた複数のデータが格納されている。ベクトル演算器は、ベクトルレジスタに格納されている各データに対して一度に演算処理を実行することによって、メモリに格納されているデータに関する該演算処理を高速に実行することができる。

【０００９】

しかし、ループ変数を用いて参照される記憶領域がメモリにて連続していない場合にベ

クトル演算器を用いて高効率に実行する目的のためには、以下に示す２つの処理を実行する必要がある。すなわち、

○記憶領域に格納されているデータをベクトルレジスタに読み取る処理、

○ベクトルレジスタに格納されているデータを、記憶領域に格納する処理にて、該データの順序を並び替える処理。

【００１０】

この結果、ベクトル演算器を有する情報処理装置は、ループ変数を用いて参照される記憶領域がメモリにて連続していない場合に、ループ処理を高速に実行することができない。

【００１１】

特許文献１、または、特許文献２に記載されたコンパイラによって作成されたオブジェクトプログラムは、記憶領域が参照変数によって参照されるループ処理、すなわち、メモリにて記憶領域が連続していないループ処理を含んでいる。したがって、ベクトル演算器を有する情報処理装置は、該ループ処理を高速に処理することができない。たとえば、「発明を実施するための形態」にて後述する打ち切り判定処理を含むプログラムは、メモリにて連続していない記憶領域を参照する処理を含んでいる一例である。

【００１２】

そこで、本発明の目的の１つは、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換するプログラム変換装置等を提供することである。

【課題を解決するための手段】

【００１３】

本発明の一態様において、プログラム変換装置は、

第１処理が順番に処理される第１ループ処理が複数回、繰り返される第２ループ処理が含まれるプログラムであって、前記順番に対して前記第１処理にて連続していない記憶領域にアクセスする処理を前記第１ループ処理が含み、さらに、条件が成り立つか否かに応じて前記第２ループ処理を終了する判定処理を前記第２ループ処理が含む場合に、前記第２ループ処理に関する繰り返し処理が第１回数分に亘って、繰り返される第３ループ処理と、前記第３ループ処理にて第２回数分に亘って、繰り返される第４ループ処理とに変換するプログラム変換手段と、

前記第１ループ処理が前記第２回数分に亘って、繰り返される処理と、前記判定処理が前記第２回数分、繰り返される処理とが前記第１回数分に亘って、繰り返される処理に変換するループ分割手段と、

前記第１処理及び前記判定処理を、前記第４ループ処理ごとに異なる記憶領域であって、前記第４ループ処理における処理順に連続している前記記憶領域にアクセスされる処理に変換する変数並び替え手段と、

前記第４ループ処理と、前記第１ループ処理との処理順序を入れ替える処理入れ替え手段と

を備える。

【００１４】

また、本発明の他の見地として、プログラム変換方法は、

第１処理が順番に処理される第１ループ処理が複数回繰り返される第２ループ処理が含まれるプログラムであって、前記順番に対して前記第１処理にて連続していない記憶領域にアクセスする処理を前記第１ループ処理が含み、さらに、条件が成り立つか否かに応じて前記第２ループ処理を終了する判定処理を前記第２ループ処理が含む場合に、前記第２ループ処理に関する繰り返し処理が第１回数分に亘って、繰り返される第３ループ処理と、前記第３ループ処理にて第２回数分に亘って、繰り返される第４ループ処理とに変換し、

前記第１ループ処理が前記第２回数分に亘って、繰り返される処理と、前記判定処理が前記第２回数分に亘って、繰り返される処理とが前記第１回数分に亘って、繰り返される

処理に変換し、
前記第１処理及び前記判定処理を、前記第４ループ処理ごとに異なる記憶領域であって、前記第４ループ処理における処理順に連続している前記記憶領域にアクセスされる処理に変換し、
前記第４ループ処理と、前記第１ループ処理との処理順序を入れ替える。
【００１５】
また、本発明の他の見地として、プログラム変換プログラムは、
第１処理が順番に処理される第１ループ処理が複数回繰り返される第２ループ処理が含まれるプログラムであって、前記順番に対して前記第１処理にて連続していない記憶領域にアクセスする処理を前記第１ループ処理が含み、さらに、条件が成り立つか否かに応じて前記第２ループ処理を終了する判定処理を前記第２ループ処理が含む場合に、前記第２ループ処理に関する繰り返し処理が第１回数分に亘って、繰り返される第３ループ処理と、前記第３ループ処理にて第２回数分に亘って、繰り返される第４ループ処理とに変換するプログラム変換機能と、
前記第１ループ処理が前記第２回数分に亘って、繰り返される処理と、前記判定処理が前記第２回数分に亘って、繰り返される処理とが前記第１回数分に亘って、繰り返される処理に変換するループ分割機能と、
前記第１処理及び前記判定処理を、前記第４ループ処理ごとに異なる記憶領域であって、前記第４ループ処理における処理順に連続している前記記憶領域にアクセスされる処理に変換する変数並び替え機能と、
前記第４ループ処理と、前記第１ループ処理との処理順序を入れ替える処理入れ替え機能と
をコンピュータに実現させる。
【００１６】
さらに、同目的は、係るプログラムを記録するコンピュータが読み取り可能な記録媒体によっても実現される。
【発明の効果】
【００１７】
本発明に係るプログラム変換装置等によれば、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができる。
【図面の簡単な説明】
【００１８】
【図１】本発明の第１の実施形態に係るプログラム変換装置が有する構成を示すブロック図である。
【図２】第１の実施形態に係るプログラム変換装置における処理の流れの概要を示すフローチャートである。
【図３】第１の実施形態に係るプログラム変換装置における処理の流れを示すフローチャートである。
【図４】第１の実施形態に係るプログラム変換部における処理の流れを示すフローチャートである。
【図５】プログラム変換装置に入力されるプログラムの一例を表す図である。
【図６】図５に例示されたプログラムにおける処理の流れを示すフローチャートである。
【図７】打ち切りプログラムにおける処理の流れを示すフローチャートである。
【図８】ステップＳ１２３における変換処理の結果として出力されたプログラムの一例を表す図である。
【図９】ステップＳ１２３における変換処理の結果として出力されたプログラムにおける処理の流れを示すフローチャートである。
【図１０】ステップＳ１２４における変換処理の結果として出力されたプログラムの一例を表す図である。

【図１１】ステップＳ１２４における変換処理の結果として出力されたプログラムにおける処理の流れを示すフローチャートである。
【図１２】ステップＳ１２６における変換処理の結果として出力されたプログラムの一例を表す図である。
【図１３】ステップＳ１２６における変換処理の結果として出力されたプログラムにおける処理の流れを示すフローチャートである。
【図１４】選択情報の一例を概念的に表す図である。
【図１５】本発明の第２の実施形態に係るプログラム変換装置が有する構成を示すブロック図である。
【図１６】第２の実施形態に係るプログラム変換装置における処理の流れを示すフローチャートである。
【図１７】本発明の第３の実施形態に係るプログラム変換装置が有する構成を示すブロック図である。
【図１８】第３の実施形態に係るプログラム変換装置における処理の流れを示すフローチャートである。
【図１９】本発明の第４の実施形態に係るプログラム変換装置が有する構成を示すブロック図である。
【図２０】第４の実施形態に係るプログラム変換装置における処理の流れを示すフローチャートである。
【図２１】指示情報を含むプログラムの一例を表す図である。
【図２２】本発明の第５の実施形態に係るプログラム変換装置が有する構成を示すブロック図である。
【図２３】打ち切り判定処理を含むプログラムの一例を概念的に表す図である。
【図２４】本発明の各実施形態に係るプログラム変換装置を実現可能な計算処理装置のハードウェア構成例を、概略的に示すブロック図である。
【発明を実施するための形態】
【００１９】
まず、本願発明の理解を容易にする目的のために、図２３を参照しながら打ち切り判定処理（ループを終了する処理）を含むループについて詳細に説明する。図２３は、打ち切り判定処理を含むプログラムの一例を概念的に表す図である。
【００２０】
打ち切りプログラムは、たとえば、第１内側処理、乃至、第４内側処理と、打ち切り判定処理（ステップＳ１６１、及び、ステップＳ１６２）とを含み、これらの処理を所定の回数分、繰り返す処理（説明の便宜上、「外側ループ処理」と表す）を表す。
【００２１】
第１内側処理、乃至、第４内側処理は、それぞれ、ある処理を所定の回数分、繰り返す処理（説明の便宜上、「内側ループ処理」と表す）を表す。第１内側処理、乃至、第４内側処理は、分岐命令（条件判定命令）を含まない命令列によって構成されている。
【００２２】
打ち切り判定処理は、分岐命令にて、ある判定条件が成り立つか否かに応じて外側ループ処理を終了する処理を表す。言い換えると、打ち切りプログラムは、複数の内側ループ処理と、該打ち切り判定処理とを含む外側ループ処理を含む。
【００２３】
尚、打ち切りプログラムが含む内側処理の個数は、必ずしも、４つである必要はなく、複数であればよい。また、打ち切りプログラムが含む打ち切り判定処理の個数は、必ずしも、２つである必要はなく、１つ以上であればよい。打ち切りプログラムは、上述した例に限定されない。
【００２４】
ベクトル演算器を有する情報処理装置にて、プログラムを高効率に実行する場合には、該プログラムに含まれているループ処理をベクトル演算器が高速処理可能なオブジェクト

プログラムに変換する（いわゆる、ベクトル化する）ことが必要である。ベクトル演算器が高速に処理可能なプログラムは、たとえば、ループにおける反復回数に応じた値を記憶可能なループ変数に関して、記憶領域が連続にアクセスされるプログラムである。ベクトル演算器を有する情報処理装置は、ループにおける反復回数に応じた値を記憶可能なループ変数に関して記憶領域が非連続的にアクセス（いわゆる、間接参照等）されるプログラムを高速に処理することができない。

【００２５】

尚、以降に示される各実施形態において、説明の便宜上、変数は、メモリにおける１つの記憶領域を表すとする。変数の値は、該１つの記憶領域に格納されているデータを表すとする。たとえば、「変数の値を読み取る」処理とは、「記憶領域に格納されているデータを読み取る」処理を表すとする。また、配列は、複数の変数から構成されている一連の記憶領域を表しているとする。配列における要素は、複数の変数における各変数を表すとする。

【００２６】

たとえば、配列ａは、Ｎ（ただし、Ｎは、自然数）個の要素（変数）を有する。この場合に、ａ［ｉ］（ただし、ｉは、１≦ｉ≦Ｎを満たす自然数）は、配列ａにおいてｉ番目の変数を表しているとする。配列ａ［Ｍ］［Ｎ］（ただし、Ｍは、自然数）は、（Ｍ×Ｎ）個の要素を有するとする。この場合に、メモリには、２次元目（すなわち、Ｎによって示される次元）の添え字の順に連続している記憶領域に格納され、さらに、該連続している記憶領域が、１次元目（すなわち、Ｍによって示される次元）の順に並んでいるとする。すなわち、配列ａ［Ｍ］［Ｎ］における各要素は、ａ［１］［１］、ａ［１］［２］、・・・、ａ［１］［Ｎ］、ａ［２］［１］、ａ［２］［２］、・・・、ａ［２］［Ｎ］、・・・、ａ［Ｍ］［１］、ａ［Ｍ］［２］、・・・、ａ［Ｍ］［Ｎ］の順に、連続している記憶領域に格納されているとする。

【００２７】

以降の説明においては、説明の便宜上、ループ処理は、該ループ処理を構成している繰り返し命令によって表すとする。

【００２８】

次に、本発明を実施する実施形態について図面を参照しながら詳細に説明する。

【００２９】

＜第１の実施形態＞

図１を参照しながら、本発明の第１の実施形態に係るプログラム変換装置１０１が有する構成について詳細に説明する。図１は、本発明の第１の実施形態に係るプログラム変換装置１０１が有する構成を示すブロック図である。

【００３０】

第１の実施形態に係るプログラム変換装置１０１は、効率算出部１０２と、ループ選択部１０３と、プログラム変換部１０４と、変換判定部１０５と、ループ分割部１０６と、処理入れ替え部１０８と、変数並び替え部１０９とを有する。

【００３１】

図２を参照しながら、本発明の第１の実施形態に係るプログラム変換装置１０１における処理の流れの概要について説明する。図２は、第１の実施形態に係るプログラム変換装置１０１における処理の流れの概要を示すフローチャートである。

【００３２】

プログラム変換装置１０１は、打ち切りプログラム等のプログラム（図２３に例示）を入力し、ベクトル化によって得られる性能向上を見積もる（ステップＳ１０１）。次に、プログラム変換装置１０１は、性能向上が高いループ処理を変換する（ステップＳ１０２）。ステップＳ１０１における詳細な処理の流れについては、図３を参照しながら後述する。ステップＳ１０２における詳細な処理の流れについては、図４を参照しながら後述する。

【００３３】

以降、説明の便宜上、プログラムを、プログラム言語であるＣ言語を用いて表すが、Ｃ言語を用いて表されていなくてもよい。また、本実施形態においては、Ｃ言語に関する説明を省略する。

【００３４】

次に、プログラム変換装置１０１に入力されるプログラムの一例について説明する。図５は、プログラム変換装置１０１に入力されるプログラムの一例を表す図である。

【００３５】

図５に例示されたプログラムは、命令「ｆｏｒ」は、ループ処理を表している。たとえば、図５に例示されたプログラムは、ループ処理「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＋）」と、ループ処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」と、ループ処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」とを含んでいる。

【００３６】

説明の便宜上、プログラムにて、最外におけるループ処理を外側ループ処理と表す。該外側ループ処理に含まれているループ処理を内側処理と表す。図５に例示されたプログラムにおいて、外側ループ処理は、ループ処理「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＋）」である。図５に例示されたプログラムにおいて、内側処理は、ループ処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」と、ループ処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」とである。

【００３７】

図６を参照しながら、図５に例示されたプログラムにおける処理について説明する。図６は、図５に例示されたプログラムにおける処理の流れを示すフローチャートである。尚、図５等の図面に例示されたプログラムにおいて、「＊」は、掛け算を表す。

【００３８】

図６を参照すると、「１００回ループの開始」、及び、「１００回ループの終了」にて区画された処理は、図５に示されたループ処理「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＋）」を表す。第１内側処理は、図５に示されたループ処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」を表す。第２内側処理は、図５に示されたループ処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」を表す。ステップＳ１５１に示された処理は、図５に示された判定処理「ｉｆ（ｖ１＞ＴＨ１）ｇｏｔｏ＿ＬＡＢＥＬ」を表す。図６においては、第３内側処理、及び、第４内側処理が示されているが、図６に示されたプログラムは、さらに、多くの処理を含んでいてもよい。プログラム変換装置１０１に入力されるプログラムは、図５、及び、図６に示された例に限定されない。

【００３９】

説明の便宜上、プログラム変換装置１０１には、情報処理装置が有するベクトル演算器におけるベクトルレジスタの大きさ（長さ、サイズ、すなわち、ベクトル演算器長）に関する情報が入力されているとする。さらに、プログラム変換装置１０１には、各命令を情報処理装置（または、ベクトル演算器）にて処理するのに要するシステムクロック数が入力されているとする。

【００４０】

次に、図３を参照しながら、本発明の第１の実施形態に係るプログラム変換装置１０１における処理について詳細に説明する。図３は、第１の実施形態に係るプログラム変換装置１０１における処理の流れを示すフローチャートである。

【００４１】

効率算出部１０２は、打ち切りプログラムに含まれている内側処理ごとに、該内側処理を構成している演算命令と、該内側処理を構成しているロード命令とを特定する（ステップＥ１１０）。効率算出部１０２は、特定した演算命令に要する処理時間、及び、特定したロード命令に要する処理時間を算出する（ステップＥ１１１、ステップＥ１１２）。効率算出部１０２は、演算命令に要する処理時間（ステップＥ１１１にて算出）と、ロード命令に要する処理時間（ステップＥ１１２にて算出）とが合計された合計値を算出することによって、打ち切りプログラムがベクトル化されていない場合の処理時間（「非ベクト

ル化時間」と表す）を算出する（ステップＥ１１３）。
【００４２】
たとえば、演算命令（または、ロード命令）に要する処理時間は、該演算命令（または、ロード命令）に要するシステムクロック数に、システムクロックあたりの時間を掛け算することにより算出される。演算命令（または、ロード命令）に要するシステムクロック数は、該演算命令が定義されたチップセットに応じて決められている。効率算出部１０２は、たとえば、特定した演算命令（または、ロード命令）ごとに、システムクロック数を求め、求めたシステムクロック数にシステムクロックあたりの時間を掛け算することによって該演算命令（または、ロード命令）に要する処理時間を算出する。
【００４３】
次に、効率算出部１０２は、各内側処理を構成しているループにてアクセスされる変数（または、配列）に関するデータの個数が合計された合計値を求める（ステップＥ１１４）。この場合に、効率算出部１０２は、ループ変数の値にて指し示される可能性がある変数をすべて特定し、特定した各変数に格納されるデータの個数が合計された合計値を求める。効率算出部１０２は、算出した合計値を、ベクトル演算器が有するベクトルレジスタに格納することが可能なデータの個数（すなわち、ベクトル演算器長）にて割り算することによって、該ループが理想的にベクトル化された場合のベクトル化効率（以降、「理想的なベクトル化効率」と表す）を算出する（ステップＥ１１５）。
【００４４】
次に、効率算出部１０２は、ステップＥ１１１にて算出された演算命令に要する処理時間が、ステップＥ１１５にて算出された理想的なベクトル化効率にて割り算された値を算出する（ステップＥ１１６）。すなわち、ステップＥ１１６にて、効率算出部１０２は、内側処理における演算命令がベクトル化されたオブジェクトプログラムを、ベクトル演算器が実行する処理に要する処理時間を算出する。
【００４５】
効率算出部１０２は、ステップＥ１１０にて特定したロード命令にてアクセスされる記憶領域が、アクセスされる順に、メモリにて連続的に配置されているか否かを判定する（ステップＥ１１７）。効率算出部１０２は、アクセスされる記憶領域がメモリにて連続している場合に（ステップＥ１１７にてＹＥＳ）、ステップＥ１１２にて算出された該ロード命令に要する処理時間を理想的なベクトル化効率にて割り算する（ステップＥ１１８）。ステップＥ１１８にて、効率算出部１０２は、内側処理におけるロード命令がベクトル化された場合に、ベクトル演算器が該ロード命令を実行するのに要する処理時間を算出する。
【００４６】
効率算出部１０２は、アクセスされる記憶領域がメモリにて連続していない場合に（ステップＥ１１７にてＮＯ）、内側処理におけるロード命令がベクトル化されたオブジェクトプログラムを実行する処理に要する処理時間として、ステップＥ１１２にて算出されたロード命令に要する処理時間を出力する。この場合に、効率算出部１０２は、ロード命令がベクトル化されない場合において該ロード命令に要する処理時間を、ベクトル演算器にてオブジェクトプログラムを実行する処理に要する処理時間として算出する。すなわち、ステップＥ１１７、及び、ステップＥ１１８に示された処理によって、効率算出部１０２は、アクセスされる記憶領域が連続しているか（すなわち、ベクトル演算器にて高速に処理されるか）否かに応じて、ロード命令がベクトル演算器を有する情報処理装置にて実行される処理に要する処理時間を算出する。
【００４７】
効率算出部１０２は、入力されたプログラムがベクトル化されたオブジェクトプログラムがベクトル演算器によって実行される場合の処理時間（以降、「ベクトル化時間」と表す）として、ステップＥ１１６にて算出された時間、及び、ステップＥ１１２またはステップＥ１１８にて算出された時間が合計された合計値を算出する（ステップＥ１１９）。
【００４８】

効率算出部１０２は、ステップＥ１１３にて算出された非ベクトル化時間を、ステップＥ１１９にて算出されたベクトル化時間にて割り算することによって、内側処理に関するベクトル化効率を算出する（ステップＥ１２０）。すなわち、ベクトル化効率は、一般的なコンパイラによってプログラムが変換された場合に、ベクトル演算器を利用せずにプログラムが実行された場合の処理時間に対する、ベクトル演算器を利用した場合の処理時間の比を表している。したがって、ベクトル化効率が大きな値であるほど、ベクトル演算器における処理時間は短い。以降、ステップＥ１２０にて算出されたベクトル化効率を「通常的なベクトル化効率」とも表す。

【００４９】

効率算出部１０２は、ステップＥ１１０乃至ステップＥ１２０に示された処理を、プログラムに含まれている各内側処理に関して実行することによって、各内側処理に関する通常的なベクトル化効率を算出する。

【００５０】

ここで、ステップＥ１２１、及び、ステップＥ１２２に関する説明にて参照する、打ち切りプログラム（図７に例示）について説明する。図７は、打ち切りプログラムにおける処理の流れを示すフローチャートである。図７に例示された打ち切りプログラムは、外側ループ処理と、該外側ループ処理に含まれている第１内側処理、乃至、第４内側処理と、外側ループ処理における処理を終了するか否かを判定する打ち切り判定処理（ステップＳ１６１、及び、ステップＳ１６２）とを含む。該打ち切りプログラムにおいて、第３内側処理、及び、第４内側処理は、ステップＳ１６１に示された判定条件が満たされなかった場合に（ステップＳ１６１にてＮＯ）実行される。第４内側処理は、さらに、ステップＳ１６２に示された判定条件が満たされなかった場合に（ステップＳ１６２にてＮＯ）実行される。

【００５１】

次に、図３を参照しながら、プログラム変換装置１０１が、図７に例示された打ち切りプログラムを入力した場合の処理（ステップＥ１２１、及び、ステップＥ１２２）について説明する。効率算出部１０２は、プログラムに含まれている各内側処理に関して、該内側処理が実行される可能性を、たとえば、該内側処理が実行される場合に経由する判定処理の回数に基づき算出する（ステップＥ１２１）。効率算出部１０２は、たとえば、所定の値を、判定処理の回数分だけ掛け算することによって、内側処理が実行される可能性（程度）を算出する。

【００５２】

説明の便宜上、所定の値は、５０％であるとする。所定の値は、必ずしも、５０％でなくともよく、たとえば、２５％、７５％等、異なる値であってもよい。所定の値は、上述した例に限定されない。

【００５３】

たとえば、図７に例示された打ち切りプログラムの場合に、第１内側処理、及び、第２内側処理は、打ち切り判定処理（ステップＳ１６１、及び、ステップＳ１６２）を経由することなく実行される。したがって、第１内側処理、または、第２内側処理が実行される場合に経由する判定処理の回数は０回である。この場合に、効率算出部１０２は、第１内側処理、及び、第２内側処理が実行される可能性が１００％であると算出する。

【００５４】

図７に例示された打ち切りプログラムの場合に、第３内側処理は、ステップＳ１６１に示された判定条件が満足されなかった場合に実行され、ステップＳ１６１に示された判定条件が満足された場合に実行されない。したがって、第３内側処理が実行される場合に経由する打ち切り判定処理の回数は１回である。この場合に、効率算出部１０２は、１００％に所定の値を打ち切り判定処理の回数分（この場合、１回）掛け算することによって、第３内側処理が実行される可能性が５０％（＝１００％×５０％）であると算出する。

【００５５】

図７に例示された打ち切りプログラムの場合に、第４内側処理は、ステップＳ１６２に

示された判定条件が満足されなかった場合に実行され、ステップＳ１６２に示された判定条件が満足された場合に実行されない。したがって、第４内側処理が実行される場合に経由する打ち切り判定処理（ステップＳ１６１、及び、ステップＳ１６２）の回数は２回である。この場合に、効率算出部１０２は、１００％に所定の値を打ち切り判定処理の回数分（この場合、２回）掛け算することによって、第４内側処理が実行される可能性が２５％（＝１００％×５０％×５０％）であると算出する。

【００５６】

効率算出部１０２は、打ち切りプログラムに含まれている内側処理が実行される可能性を算出した後に、内側処理に関して算出した可能性と、該内側処理に関してステップＥ１１５にて算出された理想的なベクトル化効率とが掛け算された値（以降、「理想的な期待ベクトル化効率」と表す）を算出する（ステップＥ１２２）。

【００５７】

図４を参照しながら、第１の実施形態に係るプログラム変換部１０４における処理について説明する。図４は、第１の実施形態に係るプログラム変換部１０４における処理の流れを示すフローチャートである。

【００５８】

ループ選択部１０３は、プログラムに含まれている各内側処理に関して、理想的な期待ベクトル化効率が、通常的なベクトル化効率にて割り算された値を表す評価値を算出する（ステップＳ１２１）。この場合に、評価値は、該プログラムに関する性能を改善可能な程度に、可能性が乗算された値を表す。該評価値が大きな値であるほど改善可能な程度は大きく、該評価値が小さいほど改善可能な程度は小さい。たとえば、アクセスされる記憶領域が連続していない内側処理によって構成される外側ループの場合に、異なる内側処理に対して、理想的なベクトル化効率を通常的なベクトル化効率にて割り算された値が同一（または略同一）であるので、評価値は、内側処理が実行される可能性が高いほど大きな値である。上述した処理に従って算出される評価値は、内側処理において連続している記憶領域にアクセスされない場合に比べ、アクセスされる場合に小さな値であり、かつ、該内側処理が実行されるまでに経由する判定処理の回数が多いほど小さな値である。

【００５９】

ループ選択部１０３は、算出した評価値が所定の条件を満たしているか否かを判定する。たとえば、所定の条件は、所定の閾値を超えているか否かを判定する条件である。ループ選択部１０３は、プログラムに含まれている内側処理（すなわち、ループ）のうち、該内側処理に関する評価値が所定の閾値を超えている内側処理を選択する（ステップＳ１２２）。ループ選択部１０３は、プログラムが該内側処理に関する評価値が所定の閾値を超えている内側処理を含んでいない場合に、内側処理を選択しない。ループ選択部１０３は、ステップＳ１２１、及び、ステップＳ１２２に示された処理の結果を表す選択情報（図１４を参照しながら後述する）を作成してもよい。

【００６０】

プログラム変換部１０４は、ループ選択部１０３が選択した内側処理を含んでいる外側ループを、２重のループであって、２重のループにおける内側のループに関する反復回数に応じた値を記憶可能なループ変数の値が連続している整数の値であるループを含むプログラム（図８、及び、図９を参照しながら後述する）に変換する（ステップＳ１２３）。この場合に、プログラム変換部１０４は、外側ループにて、該反復処理を終了するか否かを判定する打ち切り判定処理を、変換後の２重のループ処理をともに終了する打ち切り判定処理に変換する。以降、説明の便宜上、２重のループのうち、内側のループを「第１ループ」、外側のループを「第２ループ」と表す。言い換えれば、プログラム変換部１０４は、外側ループを含むプログラムを、第２ループの内側にて第１ループを実行する処理であって、第１ループの内側にて内側処理を実行するプログラムに変換する。また、プログラム変換部１０４は、第１ループにおける反復回数に応じた値を記憶可能なループ変数として、連続している整数値を設定するプログラムに変換する。

【００６１】

たとえば、プログラム変換部１０４は、外側ループにおける反復回数がＮ（ただし、Ｎは自然数を表す）回である場合に、Ｓ（ただし、Ｓは２以上の自然数を表す）回分処理を繰り返す第２ループにて、Ｔ（ただし、Ｔは、２以上の自然数であって、Ｎ＝Ｓ×Ｔである）回分処理を繰り返す第１ループを実行するプログラム（図８及び図９を参照しながら後述する）に変換する。

【００６２】

本発明の各実施形態においては、説明の便宜上、「Ｎ＝Ｓ×Ｔ」を満たしているとするが、プログラム変換部１０４は、「Ｎ＝Ｓ×Ｔ＋Ｒ」（ただし、Ｒは、正の整数を表す）を満たしているＳ、Ｔ、及び、Ｒを算出してもよい。この場合に、プログラム変換部１０４は、外側ループを、Ｓ回分処理を繰り返す第２ループにてＴ回分処理を繰り返す第１ループと、Ｒ回分処理を繰り返す反復処理とに変換する。

【００６３】

次に、処理入れ替え部１０８は、ステップＳ１２３における変換処理の結果として出力されたプログラム（図８、及び、図９に例示）内の各第２ループにて参照される変数（または、配列）に、依存関係があるか否かを判定する。ここで、依存関係は、ある処理にて算出された値が格納されている記憶領域を、該ある処理以降の処理にて参照する処理が実行されるという関係、すなわち、該ある処理及び該ある処理以降の処理の間の関係を表す。処理入れ替え部１０８は、依存関係があると判定された変数（すなわち、記憶領域）にアクセスする処理を、第２ループにおける反復回数に応じた値を記憶可能なループ変数の値ごとに異なる記憶領域にアクセスする処理を実行するプログラム（図１０及び図１１を参照しながら後述する）に変換する（ステップＳ１２４）。

【００６４】

次に、処理入れ替え部１０８は、ステップＳ１２４における変換処理の結果として出力されたプログラム（図１０、及び、図１１にて例示）を、各内側処理に関して該内側処理が実行される第１ループ及び各打ち切り判定処理に関して該打ち切り判定処理が実行される第１ループが第２ループに含まれているプログラムに変換する。すなわち、ループ分割部１０６は、該内側処理におけるループと、第２ループとに関して、ループの順序を入れ替える（ステップＳ１２５）。

【００６５】

次に、変数並び替え部１０９は、ステップＳ１２５における変換処理の結果として出力されたプログラムにおいて、第２ループに関する反復回数に応じた値を記憶可能なループ変数ごとに異なる記憶領域を、該ループ変数の値（すなわち、第２ループに関する処理順序）に関して連続している記憶領域に並び替えられたプログラム（図１２及び図１３を参照しながら後述する）に変換する（ステップＳ１２６）。たとえば、第２ループにおける反復回数に応じた値を記憶可能なループ変数の値Ｍごとに異なる記憶領域がａ［Ｍ］［Ｎ］（ただし、Ｎは自然数）である場合に、変数並び替え部１０９は、第２ループに関する処理順序に関して連続している記憶領域である記憶領域ａ［Ｎ］［Ｍ］に変換された処理を含むプログラムに変換する。

【００６６】

図１４を参照しながら、選択情報について説明する。図１４は、選択情報の一例を概念的に表す図である。

【００６７】

選択情報は、通常的なベクトル化効率と、理想的な期待ベクトル化効率と、該２つのベクトル化効率に基づき算出される評価値と、該評価値が所定の閾値を超えているか否かを表す情報(「選択の有無」によて示された情報)とが関連付けされている。

【００６８】

図１４に例示された選択情報においては、通常的なベクトル化効率「１．０」と、理想的な期待ベクトル化効率「４．０」と、評価値「４．０」と、選択の有無「選択する」とが関連付けされている。これは、評価値「４．０」が、理想的な期待ベクトル化効率「４．０」及び通常的なベクトル化効率「１．０」に基づき算出され、評価値「４．０」は所

定の閾値（たとえば、「３．０」）を超えているので選択されることを表す。
【００６９】
図１４に例示された選択情報においては、通常的なベクトル化効率「１．０」と、理想的な期待ベクトル化効率「１．２」と、評価値「１．２」と、選択の有無「選択しない」とが関連付けされている。これは、評価値「１．２」が、理想的な期待ベクトル化効率「１．２」、及び、通常的なベクトル化効率「１．０」に基づき算出され、評価値「１．２」は所定の閾値（たとえば、「３．０」）を超えていないので選択されないことを表す。
【００７０】
図８乃至図１３を参照しながら、ステップＳ１２３、ステップＳ１２４、ステップＳ１２６における変換処理の結果として出力されたプログラムについて説明する。
【００７１】
図８に例示されたプログラム、及び、該プログラムにおける処理の流れを示すフローチャート（図９）を参照しながら、ステップＳ１２３(図４)における変換処理の結果として出力されたプログラムについて説明する。図８は、ステップＳ１２３における変換処理の結果として出力されたプログラムの一例を表す図である。図９は、ステップＳ１２３における変換処理の結果として出力されたプログラムにおける処理の流れを示すフローチャートである。
【００７２】
図８において、命令「ｆｏｒ」は、ループ処理を表している。図８に例示されたプログラムにおいて、第２ループは、「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＝２０）」にて示された処理であり、該第２ループにおける反復回数に応じた値を記憶可能なループ変数は、「ｉ１」である。第１ループは、「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」にて示された処理であり、該第１ループにおける反復回数に応じた値を記憶可能なループ変数は、「ｉｉ」である。内側処理は、「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」にて示された処理と、「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」にて示された処理である。内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」を構成しているループ変数は、「ｉ２」である。内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」を構成しているループ変数は、「ｉ３」である。打ち切り判定処理は、「ｉｆ（ｖ１＞ＴＨ１）＿ｇｏｔｏ＿ＬＡＢＥＬ」にて示された処理である。
【００７３】
外側ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＋）」（図５）に関する処理は、第２ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＝２０）」と、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」とに変換されている。
【００７４】
図８に示されたプログラムにおいて、処理「ａ１［ｉ２］＝ａ０［ｉ２］＊ｄｘ［ｉ１＋ｉｉ］」は、記憶領域「ａ１［ｉ２］」に値が格納される処理を表す。また、処理「ａ２［ｉ２］＝ａ０［ｉ２］＊ｄｙ［ｉ１＋ｉｉ］」は、記憶領域「ａ２［ｉ２］」に値が格納される処理を表す。したがって、内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」は、記憶領域「ａ１［ｉ２］」に値が格納される処理と、記憶領域「ａ２［ｉ２］」に値が格納される処理とを含む。また、処理「ｖ１＋＝ａ１［ａ３［ｉ３］］－ａ２［ａ３［ｉ３］］」は、記憶領域「ｖ１」に格納されている値と、記憶領域「ａ１［ｉ３］」に格納されている値とを足し算し、その結果から記憶領域「ａ２［ｉ３］」に格納されている値を引き算した結果を、記憶領域「ｖ１」に格納する処理を表している。したがって、内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」にて算出した結果を、内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」にてアクセスするので、内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」と、内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」との間には依存関係がある。同様に、内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」にて算出した値が格納される記憶領域「ｖ１」は、打ち切り判定処理「ｉｆ（ｖ１＞ＴＨ１）＿ｇｏｔｏ＿ＬＡＢＥＬ」にてアクセスされる。したがって、内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」と、打ち切り判

定処理「ｉｆ（ｖ１＞ＴＨ１）＿ｇｏｔｏ＿ＬＡＢＥＬ」との間には依存関係がある。

【００７５】

図９を参照すると、「５回ループの開始」及び「５回ループの終了」にて区画された処理は、図８に例示された第２ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＝２０）」を表す。「２０回ループの開始」及び「２０回ループの終了」にて区画された処理は、図８に例示された第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を表す。第１内側処理は、図８に例示された内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」を表す。第２内側処理は、図８に例示された内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」を表す。ステップＳ１５１に示された処理は、打ち切り判定処理「ｉｆ（ｖ１＞ＴＨ１）ｇｏｔｏ＿ＬＡＢＥＬ」を表す。図９においては、第３内側処理、及び、第４内側処理が示されているが、図８に示されたプログラムは、さらに、多くの処理を含んでいてもよい。プログラムは、図８、または、図９に示された例に限定されない。

【００７６】

図１０及び図１１を参照しながらステップＳ１２４(図４)における変換処理の結果として出力されたプログラムについて説明する。図１０は、ステップＳ１２４における変換処理の結果として出力されたプログラムの一例を表す図である。図１１は、ステップＳ１２４における変換処理の結果として出力されたプログラムにおける処理の流れを示すフローチャートである。

【００７７】

図８を参照しながら説明したプログラムは、図１０に示すように、第２ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＝２０）」にて後述の３つのループ処理を実行するプログラムに変換されている。すなわち、

○第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」にて内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」が実行される、

○第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」にて内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」が実行される、

○第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」にて打ち切り判定処理「ｉｆ（ｖ１［ｉｉ］＞ＴＨ１）＿ｇｏｔｏ＿ＬＡＢＥＬ」が実行される。

【００７８】

また、図８に示された記憶領域「ａ１［ｉ２］」に値が格納される処理は、図１０に示されているように第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」ごとに異なる記憶領域「ａ１［ｉｉ］［ｉ２］」に値が格納される処理に変換されている。同様に、図８に示された記憶領域「ａ１［ｉ２］」に格納されているデータが参照される処理「ａ１［ａ３［ｉ３］］－ａ２［ａ３［ｉ３］］」は、図１０に示すように、第１ループにおける反復回数に応じた値を記憶可能なループ変数「ｉｉ」ごとに異なる記憶領域「ａ１［ｉｉ］［ａ３［ｉ３］］－ａ２［ｉｉ］［ａ３［ｉ３］］」に変換されている。同様に、図８に示された記憶領域「ｖ１」に値が格納される処理は、図１０に示すように、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」ごとに異なる記憶領域「ｖ１［ｉｉ］」に値が格納される処理に変換されている。

【００７９】

しかし、図１０において、記憶領域「ａ１［ｉｉ］［ｉ２］」は、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」に関して、連続している記憶領域ではない。同様に、記憶領域「ａ２［ｉｉ］［ｉ２］」は、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」に関して、連続している記憶領域ではない。

【００８０】

図１１を参照すると「５回ループの開始」及び「５回ループの終了」にて区画された処理は、図１０に示された第２ループ「ｆｏｒ（ｉｉ＝０；ｉ１＜１００；ｉ１＋＝２０）」を表す。「２０回ループの開始」（ステップＳ１５４乃至ステップＳ１５６のいずれも

）及び「２０回ループの終了」にて区画された処理は、図１０に示された第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を表す。第１内側処理は、図１０に示された内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」を表す。第２内側処理は、図１０に示された内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」を表す。ステップＳ１５１に示された処理は、打ち切り判定処理「ｉｆ（ｖ１［ｉｉ］＞ＴＨ１）ｇｏｔｏ＿ＬＡＢＥＬ」を表す。図１１においては、第３内側処理、及び、第４内側処理が示されているが、図１０に示されたプログラムは、さらに、多くの処理を含んでいてもよい。プログラム変換装置１０１に入力されるプログラムは、図１０、または、図１１に示された例に限定されない。

【００８１】

図１２、及び、図１３を参照しながらステップＳ１２６(図４)における変換処理の結果として出力されたプログラムについて説明する。図１２は、ステップＳ１２６における変換処理の結果として出力されたプログラムの一例を表す図である。図１３は、ステップＳ１２６における変換処理の結果として出力されたプログラムにおける処理の流れを示すフローチャートである。

【００８２】

図１０を参照しながら説明したプログラムは、図１２に示すように、内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」にて第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」が実行されるプログラムに変換されている。同様に、図１０に示されたプログラムは、内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」にて第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」が実行されるプログラムに変換されている。

【００８３】

また、図１２に例示されたプログラムにおいては、さらに、記憶領域「ａ１［ｉｉ］［ｉ２］」に関して、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」に関して連続している記憶領域「ａ１［ｉ２］［ｉｉ］」に変換されている。同様に、図１２に例示されたプログラムにおいては、さらに、記憶領域「ａ２［ｉｉ］［ｉ２］」に関して、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」に関して連続している記憶領域「ａ２［ｉ２］［ｉｉ］」に変換されている。

【００８４】

さらに、内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」にて参照されていた記憶領域「ａ１［ｉｉ］［ａ３［ｉ３］］」（図１０）に関しても同様である。すなわち、記憶領域「ａ１［ｉｉ］［ａ３［ｉ３］］」は、図１２に示されているように、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」に関して連続している記憶領域「ａ１［ａ３［ｉ３］］［ｉｉ］」に変換されている。さらに、記憶領域「ａ２［ｉｉ］［ａ３［ｉ３］］」（図１０）は、図１２に示されているように、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を構成しているループ変数「ｉｉ」に関して連続している記憶領域「ａ２［ａ３［ｉ３］］［ｉｉ］」に変換されている。

【００８５】

図１３を参照すると、「５回ループの開始」及び「５回ループの終了」にて区画された処理は、図１２に例示された第２ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＝２０）」を表す。「第１処理の開始」及び「第１処理の終了」にて区画された処理は、図１２に示された内側処理「ｆｏｒ（ｉ２＝０；ｉ２＜１００；ｉ２＋＋）」を表す。第３処理は、図１２に示された第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を表す。「第２処理の開始」及び「第２処理の終了」にて区画された処理は、図１２に例示された内側処理「ｆｏｒ（ｉ３＝０；ｉ３＜２０；ｉ３＋＋）」を表す。第４処理は、第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を表す。ステップＳ１５１に示された処理は、判定処理「ｉｆ（ｖ１［ｉｉ］＞ＴＨ１）ｇｏｔｏ＿ＬＡＢＥＬ」を表す。

「２０回ループの開始」及び「２０回ループの終了」にて区画された処理は、図１２に示された第１ループ「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」を表す。図１３においては、第３内側処理、及び、第４内側処理が示されているが、図１２に示されたプログラムは、さらに、多くの処理を含んでいてもよい。プログラム変換装置１０１に入力されるプログラムは、図１２、または、図１３に示された例に限定されない。

【００８６】

次に、第１の実施形態に係るプログラム変換装置１０１に関する効果について説明する。

【００８７】

第１の実施形態に係るプログラム変換装置１０１によれば、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができる。この理由は、内側処理、及び、打ち切り判定処理を含むプログラムであっても、変換後の最内のループ（上述した例の場合、「ｆｏｒ（ｉｉ＝０；ｉｉ＜２０；ｉｉ＋＋）」）にて、該ループにおける反復回数に応じた値を記憶可能なループ変数（上述した例の場合に「ｉｉ」）に関して、連続している記憶領域に格納されている値がアクセスされるプログラムに変換されるからである。この理由について詳細に説明する。

【００８８】

上述したようにベクトル演算器を有する情報処理装置は、ループにおける反復回数に応じた値を記憶可能なループ変数の値に関して、連続している記憶領域に格納されている値を参照しながら演算する処理を高速に処理することができる。しかし、ベクトル演算器を有する情報処理装置は、ループにおける反復回数に応じた値を記憶可能なループ変数の値に関して、連続していない記憶領域に格納されている値を参照しながら演算する処理を高速に処理することはできない。

【００８９】

打ち切り判定処理は、ループにおける反復回数に応じた値を記憶可能なループ変数の値に関して、連続していない記憶領域に格納されている値を参照しながら演算する処理を含んでいる。したがって、特許文献１、または、特許文献２に開示されたコンパイラは、打ち切りプログラムを、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができない。

【００９０】

さらに、本実施形態に係るプログラム変換装置１０１によれば、ベクトル演算器にて処理することによって処理時間が長くなるループ処理が含まれていたとしても、情報処理装置にて、処理時間が長くなる状態は生じない。この理由は、各内側処理に関する評価値が所定の閾値以下である場合に、ループ選択部１０３が該内側処理を選択しないからである。この場合には、改善可能な程度が低い内側処理をプログラム変換装置１０１が変換しないので、処理時間が長くなる状態は生じない。

【００９１】

＜第２の実施形態＞

次に、上述した第１の実施形態を基本とする本発明の第２の実施形態について説明する。

【００９２】

以降の説明においては、本実施形態に係る特徴的な部分を中心に説明すると共に、上述した第１の実施形態と同様な構成については、同一の参照番号を付すことにより、重複する説明を省略する。

【００９３】

図１５を参照しながら、本発明の第２の実施形態に係るプログラム変換装置２０１が有する構成について詳細に説明する。図１５は、本発明の第２の実施形態に係るプログラム変換装置２０１が有する構成を示すブロック図である。

【００９４】

第２の実施形態に係るプログラム変換装置２０１は、効率算出部１０２と、ループ選択部１０３と、プログラム変換部２０４と、変換判定部１０５と、ループ分割部１０６と、処理入れ替え部１０８と、変数並び替え部１０９と、分配数決定部２１０とを有する。

【００９５】

プログラム変換装置２０１は、打ち切りプログラム等のプログラムを入力し、該打ち切りプログラムがベクトル化されたオブジェクトプログラムを出力する。

【００９６】

次に、図１６を参照しながら、本発明の第２の実施形態に係るプログラム変換装置２０１における処理について詳細に説明する。図１６は、第２の実施形態に係るプログラム変換装置２０１における処理の流れを示すフローチャートである。

【００９７】

ループ選択部１０３は、図４に示された処理と同様の処理を実行することによって、入力されたプログラムに含まれている内側処理に関する評価値を算出する（ステップＳ１２１）。ループ選択部１０３は、プログラムに含まれている内側処理のうち、該内側処理に関する評価値が所定の閾値を超えている内側処理（すなわち、ループ）を選択する（ステップＳ１２２）。

【００９８】

次に、分配数決定部２１０は、ステップＳ１２２にて選択された内側処理を含む外側ループに含まれている内側処理においてアクセスされるデータの個数を求め、該外側ループに関して求めたデータの個数が合計された合計値を算出する。分配数決定部２１０は、変換後のオブジェクトプログラムを実行する情報処理装置が有しているキャッシュメモリのサイズを、求めた合計値にて割り算することによって、外側ループを２重のループに変換する場合の第２ループにおける反復回数（以降、「分配数」と表す）を算出する（ステップＳ２０１）。分配数決定部２１０は、分配数が整数でない場合に、たとえば、該分配数に小数点以下の値を切り捨てる演算を適用してもよい。

【００９９】

プログラム変換部２０４は、外側ループ処理を構成しているループ変数を用いて設定された反復回数を、分配数決定部２１０が算出した分配数によって割り算することによって第１ループにおける反復回数を算出する。プログラム変換部２０４は、該外側ループ処理を、第１ループにおいて、分配数決定部２１０が算出した分配数分の反復回数を実行する第２ループが、算出した反復回数分繰り返される処理に変換する（ステップＳ２０２）。

【０１００】

次に、第２の実施形態に係るプログラム変換装置２０１に関する効果について説明する。

【０１０１】

本実施形態に係るプログラム変換装置２０１によれば、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができる。この理由は、第１の実施形態にて上述した理由と同様な理由である。

【０１０２】

さらに、本実施形態に係るプログラム変換装置２０１によれば、入力されたプログラムを、ベクトル演算器、及び、キャッシュメモリを有している情報処理装置にて高効率に実行することが可能なプログラムに変換することができる。この理由は、メモリとベクトルレジスタとの間にてデータを中継するキャッシュメモリが、記憶可能なサイズのデータを第２ループにて処理するよう、分配数決定部２１０が分配数を決定するからである。したがって、該情報処理装置は、キャッシュメモリに格納されているデータにアクセスすることによって、連続している内側処理に関してデータを入出力することができる。

【０１０３】

＜第３の実施形態＞

次に、上述した第１の実施形態を基本とする本発明の第３の実施形態について説明する

。
【０１０４】
以降の説明においては、本実施形態に係る特徴的な部分を中心に説明すると共に、上述した第１の実施形態と同様な構成については、同一の参照番号を付すことにより、重複する説明を省略する。
【０１０５】
図１７を参照しながら、本発明の第３の実施形態に係るプログラム変換装置３０１が有する構成について詳細に説明する。図１７は、本発明の第３の実施形態に係るプログラム変換装置３０１が有する構成を示すブロック図である。
【０１０６】
第３の実施形態に係るプログラム変換装置３０１は、効率算出部１０２と、ループ選択部１０３と、プログラム変換部３０４と、変換判定部１０５と、ループ分割部１０６と、処理入れ替え部１０８と、変数並び替え部１０９と、分配数決定部３１０とを有する。
【０１０７】
プログラム変換装置３０１は、打ち切りプログラム等のプログラムを入力し、該打ち切りプログラムがベクトル化されたオブジェクトプログラムを出力する。
【０１０８】
次に、図１８を参照しながら、本発明の第３の実施形態に係るプログラム変換装置３０１における処理について詳細に説明する。図１８は、第３の実施形態に係るプログラム変換装置３０１における処理の流れを示すフローチャートである。
【０１０９】
ループ選択部１０３は、図４に示された処理と同様の処理を実行することによって、入力されたプログラムに含まれている内側処理に関する評価値を算出する（ステップＳ１２１）。ループ選択部１０３は、プログラムに含まれている内側処理のうち、該内側処理に関する評価値が所定の閾値を超えている内側処理（すなわち、ループ）を選択する（ステップＳ１２２）。
【０１１０】
次に、分配数決定部３１０は、ループ選択部１０３が算出した評価値に基づき、外側ループを２重のループに変換する場合の第２ループにおける反復回数である分配数を算出する（ステップＳ３０１）。この場合に、分配数決定部３１０は、所定の算出手順に従い、分配数を算出する。上述したように、評価値が該プログラムに関する性能を改善可能な程度を表しているので、所定の算出手順は、たとえば、ベクトル演算器長よりも十分に大きな値を算出し、かつ、評価値が大きな値であるほど大きな値を算出し、かつ、評価値が小さな値であるほど小さな値を算出する手順である。
【０１１１】
プログラム変換部３０４は、外側ループ処理を構成しているループ変数を用いて設定された反復回数を、分配数決定部３１０が算出した分配数によって割り算することによって第１ループにおける反復回数を算出する。プログラム変換部３０４は、該外側ループ処理を、第１ループにおいて、分配数決定部３１０が算出した分配数分の反復回数を実行する第２ループが、算出された反復回数分繰り返される処理に変換する（ステップＳ３０２）。
【０１１２】
次に、第３の実施形態に係るプログラム変換装置３０１に関する効果について説明する。
【０１１３】
本実施形態に係るプログラム変換装置３０１によれば、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができる。この理由は、第１の実施形態にて上述した理由と同様な理由である。
【０１１４】

さらに、本実施形態に係るプログラム変換装置３０１によれば、さらに効率が良いプログラムに変換することができる。この理由は、評価値が大きな値であるほど分配数として大きな値を分配数決定部３１０が算出するからである。言い換えれば、分配数決定部３１０が、プログラムに関する性能を改善可能な程度が大きいほど、第２ループにおける反復回数を大きな値を算出するので、連続している記憶領域はより大きい。この結果、ベクトル演算器を有する情報処理装置は、プログラム変換装置３０１が作成したオブジェクトプログラムを高速に処理することができる。

【０１１５】

＜第４の実施形態＞

次に、上述した第３の実施形態を基本とする本発明の第４の実施形態について説明する。

【０１１６】

以降の説明においては、本実施形態に係る特徴的な部分を中心に説明すると共に、上述した第３の実施形態と同様な構成については、同一の参照番号を付すことにより、重複する説明を省略する。

【０１１７】

図１９を参照しながら、本発明の第４の実施形態に係るプログラム変換装置４０１が有する構成について詳細に説明する。図１９は、本発明の第４の実施形態に係るプログラム変換装置４０１が有する構成を示すブロック図である。

【０１１８】

第４の実施形態に係るプログラム変換装置４０１は、効率算出部１０２と、ループ選択部１０３と、プログラム変換部４０４と、変換判定部１０５と、ループ分割部１０６と、処理入れ替え部１０８と、変数並び替え部１０９と、分配数決定部４１０とを有する。

【０１１９】

プログラム変換装置４０１は、指示情報を含むプログラム（図２１に例示、打ち切りプログラム）を入力し、該打ち切りプログラムがベクトル化されたオブジェクトプログラムを出力する。図２１は、指示情報を含むプログラムの一例を表す図である。

【０１２０】

図２１に例示されたプログラムは、指示情報「＃ｐｒａｇｍａ＿ｈｉｎｔ＿ａｖｅｒａｇｅｉｔｅｒｓ＝４０」を含んでいる。これは、該指示情報に後続している外側ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＋）」に関する反復回数が、４０回である可能性が高いことを表す。言い換えれば、外側ループ「ｆｏｒ（ｉ１＝０；ｉ１＜１００；ｉ１＋＋）」は、最大１００回分の反復処理であるが、実際には、４０回分の反復処理である可能性が高いことを表す。言い換えれば、指示情報は、プログラムが変換される場合の反復回数を表している。

【０１２１】

次に、図２０を参照しながら、本発明の第４の実施形態に係るプログラム変換装置４０１における処理について詳細に説明する。図２０は、第４の実施形態に係るプログラム変換装置４０１における処理の流れを示すフローチャートである。

【０１２２】

ループ選択部１０３は、図４に示された処理と同様の処理を実行することによって、入力されたプログラムに含まれている内側処理に関する評価値を算出する（ステップＳ１２１）。ループ選択部１０３は、プログラムに含まれている内側処理のうち、該内側処理に関する評価値が所定の閾値を超えている内側処理（すなわち、ループ）を選択する（ステップＳ１２２）。

【０１２３】

次に、分配数決定部４１０は、ステップＳ１２２にて選択された内側処理を含む外側ループに含まれている内側処理においてアクセスされるデータの個数を求め、該外側ループに関して求めたデータの個数が合計された合計値を算出する。分配数決定部４１０は、変換後のオブジェクトプログラムを実行する情報処理装置が有しているキャッシュメモリの

サイズを、求めた合計値にて割り算することによって、外側ループを２重のループに変換する場合の第２ループにおける反復回数（分配数、以降、「第１分配数」と表す）を算出する（ステップＳ２０１）。分配数決定部４１０は、分配数が整数でない場合に、該分配数に小数点以下の値を切り捨てる演算を適用してもよい。

【０１２４】

次に、分配数決定部４１０は、ループ選択部１０３が算出した評価値に基づき、分配数（以降、「第２分配数」と表す）を算出する（ステップＳ３０１）。

【０１２５】

プログラム変換部４０４は、入力プログラムに含まれている指示情報から回数を読み取り（ステップＳ４０１）、外側ループに関する処理（反復回数をＴとする）を、読み取った回数分だけ反復される第１外側ループと、該第１外側ループの後に処理される（Ｔ－（読み取った回数））分だけ反復する第２外側ループとに分ける。プログラム変換部４０４は、第１外側ループ、及び、第２外側ループを、それぞれ、２重のループ処理に変換する（ステップＳ４０２）。プログラム変換部４０４は、第１外側ループに関する第２ループにおける反復回数（分配数）を第１分配数として、該第１外側ループに関する処理を２重のループ処理に変換する。プログラム変換部４０４は、第２外側ループに関する第２ループにおける反復回数（分配数）を第２分配数として、該第２外側ループに関する処理を２重のループ処理に変換する（ステップＳ４０２）。言い換えれば、プログラム変換部４０４は、指示情報に含まれている回数分だけキャッシュメモリのサイズに基づき外側ループが分割された処理を実行し、その後、（Ｔ－（読み取った回数））分だけ、評価値に基づき外側ループが分割された処理を実行する処理に、外側ループに関する処理を変換する。すなわち、プログラム変換部４０４は、指示情報に含まれている回数分だけキャッシュメモリのサイズに基づき外側ループが分割された処理を実行し、その後、プログラムにて指示された回数分まで、評価値に基づき外側ループが分割された処理を実行する処理に、外側ループに関する処理を変換する。

【０１２６】

次に、第４の実施形態に係るプログラム変換装置４０１に関する効果について説明する。

【０１２７】

本実施形態に係るプログラム変換装置４０１によれば、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができる。この理由は、第４の実施形態に係るプログラム変換装置４０１が有する構成は、第３の実施形態にて上述した理由と同様な理由である。

【０１２８】

さらに、本実施形態に係るプログラム変換装置４０１によれば、より実行効率が高いプログラムに変換することができる。この理由は、打ち切りが起こるまでの回転数と予想される回転数の間は連続している複数のループ処理にてアクセスされるデータがキャッシュメモリを介して参照され、さらに、処理が打ち切られる可能性が高い回転数以降においてもループ処理における性能が高いからである。

【０１２９】

＜第５の実施形態＞

次に、上述した本発明の第１の実施形態乃至第４の実施形態に共通している本発明の第５の実施形態について説明する。

【０１３０】

図２２を参照しながら、本発明の第５の実施形態に係るプログラム変換装置５０１が有する構成について詳細に説明する。図２２は、本発明の第５の実施形態に係るプログラム変換装置５０１が有する構成を示すブロック図である。

【０１３１】

第５の実施形態に係るプログラム変換装置５０１は、プログラム変換部５０２と、ルー

プ分割部５０３と、処理入れ替え部５０４と、変数並び替え部５０５とを有する。

【０１３２】

プログラム変換装置５０１は、上述したような打ち切り判定処理が含まれているプログラムを変換する。該プログラムは、第１処理が順番に処理される第１ループ処理が複数回繰り返される第２ループ処理が含まれる。第１ループ処理においては、該処理の順番に対して、連続していない記憶領域にアクセスされる。さらに、該第２ループ処理は、条件が成り立つか否かに応じて該第２ループ処理が終了される判定処理（打ち切り判定処理）を含んでいる。

【０１３３】

プログラム変換部５０２は、該第２ループ処理に関する繰り返し処理を、第１回数分繰り返される第３ループ処理と、該第３ループ処理にて第２回数分繰り返される第４ループ処理とに変換する。

【０１３４】

ループ分割部５０３は、該第１ループ処理が該第２回数分繰り返される処理と、該判定処理が該第２回数分繰り返される処理とが該第１回数分繰り返される処理に変換する。

【０１３５】

変数並び替え部５０５は、該第１処理、及び、該判定処理を、該第４ループ処理ごとに異なる記憶領域であって、該第４ループ処理における処理順に連続している該記憶領域にアクセスされる処理に変換する。

【０１３６】

処理入れ替え部５０４は、該第４ループ処理と、該第１ループ処理との処理順序を入れ替える。

【０１３７】

プログラム変換部５０２は、たとえば、プログラム変換部１０４（図１）、プログラム変換部２０４（図１５）、または、プログラム変換部３０４（図１７）等が有する機能によって実現することができる。ループ分割部５０３は、たとえば、ループ分割部１０６（図１）等が有する機能によって実現することができる。変数並び替え部５０５は、たとえば、変数並び替え部１０９（図１）等が有する機能によって実現することができる。処理入れ替え部５０４は、処理入れ替え部１０８（図１）等が有する機能によって実現することができる。

【０１３８】

次に、第５の実施形態に係るプログラム変換装置５０１に関する効果について説明する。

【０１３９】

第５の実施形態に係るプログラム変換装置５０１によれば、打ち切り判定処理を含むループを有するプログラムであっても、ベクトル演算器を有する情報処理装置が高速に実行することが可能なプログラムに変換することができる。この理由は、内側処理、及び、打ち切り判定処理を含むプログラムであっても、変換後の最内のループ処理（すなわち、処理が入れ替えられた後の第４ループ）にて、該ループにおける反復回数に応じた値を記憶可能なループ変数に関して、連続している記憶領域に格納されている値がアクセスされるプログラムに変換されるからである。

【０１４０】

（ハードウェア構成例）

上述した本発明の各実施形態におけるプログラム変換装置を、１つの計算処理装置（情報処理装置、コンピュータ）を用いて実現するハードウェア資源の構成例について説明する。但し、係るプログラム変換装置は、物理的または機能的に少なくとも２つの計算処理装置を用いて実現してもよい。また、係るプログラム変換装置は、専用の装置として実現してもよい。

【０１４１】

図２４は、第１の実施形態乃至第５の実施形態に係るプログラム変換装置を実現可能な

計算処理装置のハードウェア構成例を概略的に示す図である。計算処理装置２０は、中央処理演算装置（Ｃｅｎｔｒａｌ＿Ｐｒｏｃｅｓｓｉｎｇ＿Ｕｎｉｔ、以降「ＣＰＵ」と表す）２１、メモリ２２、ディスク２３、及び、不揮発性記録媒体２４を有する。計算処理装置２０は、さらに、通信インターフェース（以降、「通信ＩＦ」と表す）２７を有する。計算処理装置２０は、入力装置２５、及び、出力装置２６に接続されていてもよい。計算処理装置２０は、通信ＩＦ２７を介して、他の計算処理装置、及び、通信装置と情報を送受信することができる。

【０１４２】

不揮発性記録媒体２４は、コンピュータが読み取り可能な、たとえば、コンパクトディスク（Ｃｏｍｐａｃｔ＿Ｄｉｓｃ）、デジタルバーサタイルディスク（Ｄｉｇｉｔａｌ＿Ｖｅｒｓａｔｉｌｅ＿Ｄｉｓｃ）である。また、不揮発性記録媒体２４は、ユニバーサルシリアルバスメモリ（ＵＳＢメモリ）、ソリッドステートドライブ（Ｓｏｌｉｄ＿Ｓｔａｔｅ＿Ｄｒｉｖｅ）等であってもよい。不揮発性記録媒体２４は、電源を供給しなくても係るプログラムを保持し、持ち運びを可能にする。不揮発性記録媒体２４は、上述した媒体に限定されない。また、不揮発性記録媒体２４の代わりに、通信ＩＦ２７、及び、通信ネットワークを介して係るプログラムを持ち運びしてもよい。

【０１４３】

すなわち、ＣＰＵ２１は、ディスク２３に記憶されているソフトウェア・プログラム（コンピュータプログラム：以下、単に「プログラム」と称する）を、実行する際にメモリ２２にコピーし、演算処理を実行する。ＣＰＵ２１は、プログラム実行に必要なデータをメモリ２２から読み取る。表示が必要な場合には、ＣＰＵ２１は、出力装置２６に出力結果を表示する。外側からプログラムを入力する場合、ＣＰＵ２１は、入力装置２５からプログラムを読み取る。ＣＰＵ２１は、上述した図１、図１５、図１７、図１９、または、図２２に示す各部が表す機能（処理）に対応するところのメモリ２２にあるプログラム変換プログラム（図２，図３、図４、図１６、図１８、または、図２０）を解釈し実行する。ＣＰＵ２１は、上述した本発明の各実施形態において説明した処理を順次実行する。

【０１４４】

すなわち、このような場合、本発明は、係るプログラム変換プログラムによっても成し得ると捉えることができる。さらに、係るプログラム変換プログラムが記録されたコンピュータが読み取り可能な不揮発性の記録媒体によっても、本発明は成し得ると捉えることができる。

【０１４５】

以上、上述した実施形態を模範的な例として本発明を説明した。しかし、本発明は、上述した実施形態には限定されない。すなわち、本発明は、本発明のスコープ内において、当業者が理解し得る様々な態様を適用することができる。

【０１４６】

この出願は、２０１６年１月４日に出願された日本出願特願２０１６－０００００４６を基礎とする優先権を主張し、その開示の全てをここに取り込む。

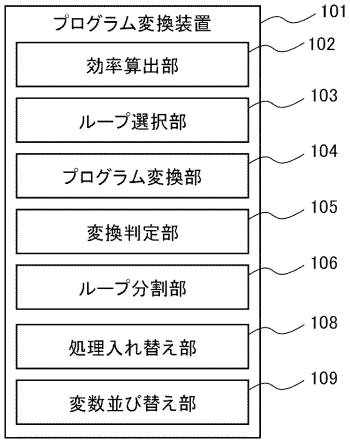
【符号の説明】

【０１４７】

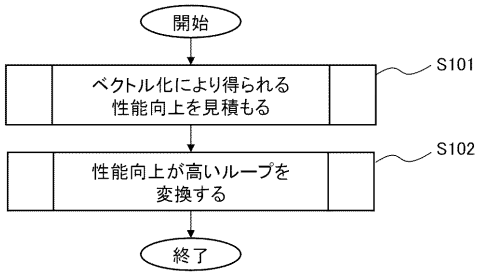
１０１　プログラム変換装置
１０２　効率算出部
１０３　ループ選択部
１０４　プログラム変換部
１０５　変換判定部
１０６　ループ分割部
１０８　処理入れ替え部
１０９　変数並び替え部
２０１　プログラム変換装置
２０４　プログラム変換部

２１０　分配数決定部
３０１　プログラム変換装置
３０４　プログラム変換部
３１０　分配数決定部
４０１　プログラム変換装置
４０４　プログラム変換部
４１０　分配数決定部
５０１　プログラム変換装置
５０２　プログラム変換部
５０３　ループ分割部
５０４　処理入れ替え部
５０５　変数並び替え部
２０　計算処理装置
２１　ＣＰＵ
２２　メモリ
２３　ディスク
２４　不揮発性記録媒体
２５　入力装置
２６　出力装置
２７　通信ＩＦ

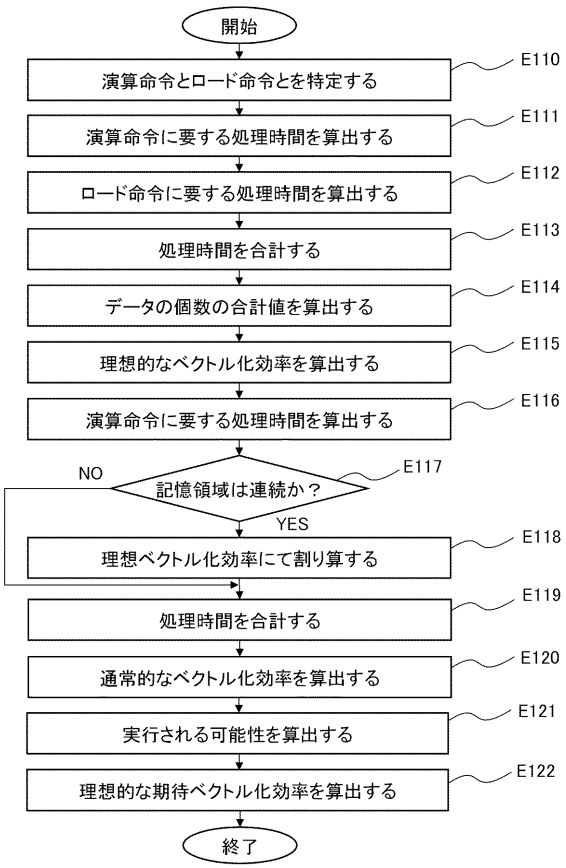
【図１】



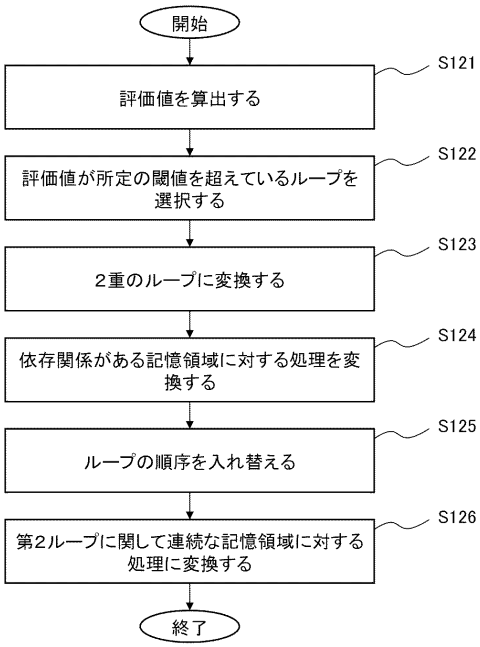
【図２】



【図３】



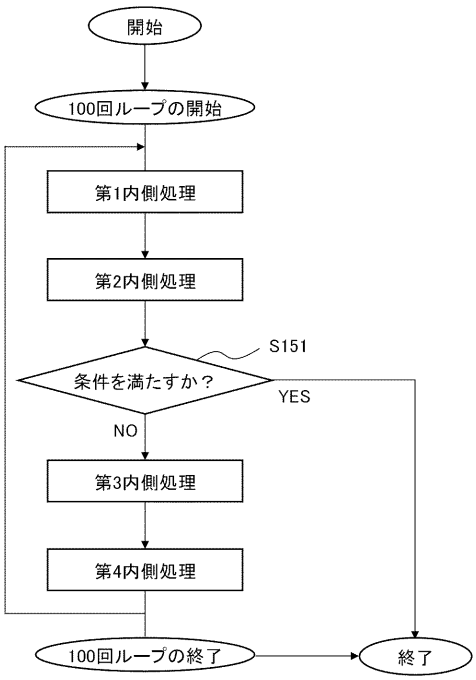
【図４】



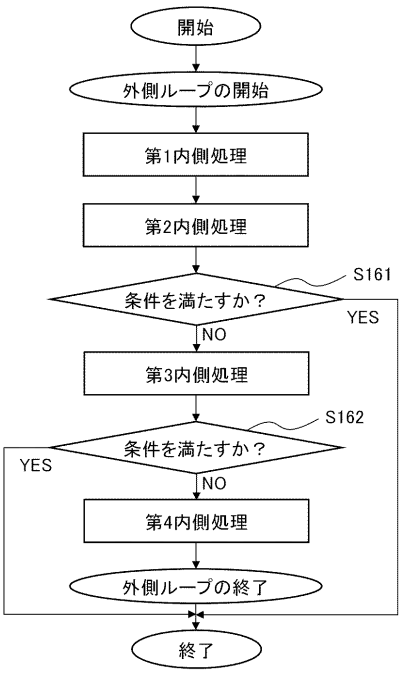
【図５】

```
for (i1=0; i1<100; i1++){
  for (i2=0; i2<100; i2++){
    a1[i2]=a0[i2]*dx[i1];
    a2[i2]=a0[i2]*dy[i1];
  }
  for (i3=0; i3<20; i3++){
    v1+=a1[a3[i3]]-a2[a3[i3]];
  }
  if (v1>TH1) goto LABEL;
}
LABEL:
```

【図６】



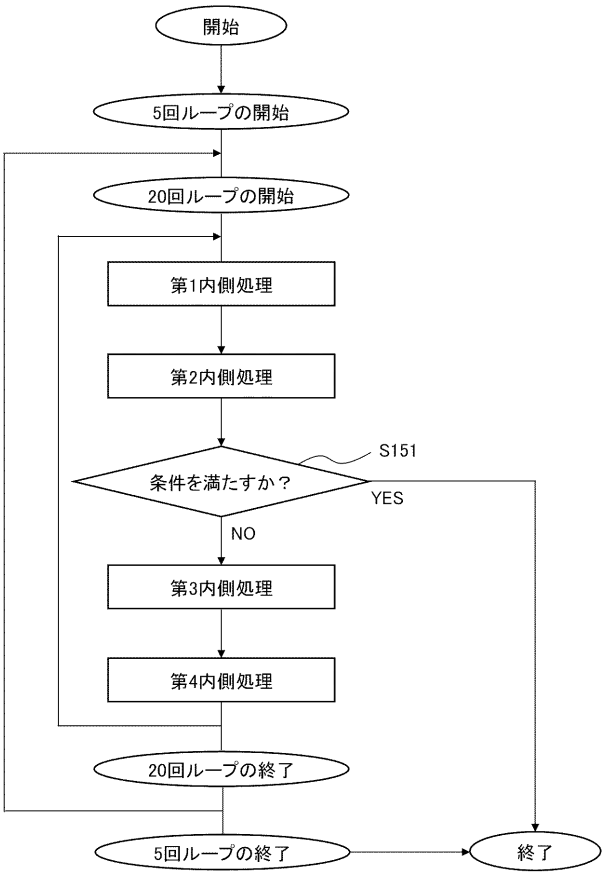
【図７】



【図８】

```
for (i1=0; i1<100; i1+=20){
  for (ii=0; ii<20; ii++){
    for (i2=0; i2<100; i2++){
      a1[i2]=a0[i2]*dx[i1+ii];
      a2[i2]=a0[i2]*dy[i1+ii];
    }
    for (i3=0; i3<20; i3++){
      v1+=a1[a3[i3]]-a2[a3[i3]];
    }
  }
  if (v1>TH1) goto LABEL;
}
LABEL:
```

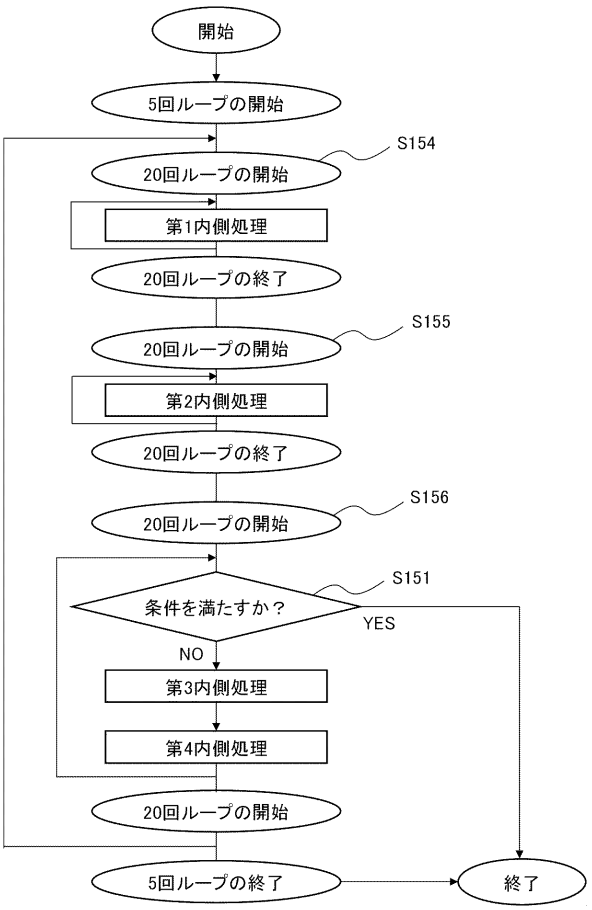
【図９】



【図１０】

```
for (i1=0; i1<100; i1+=20){
  for (ii=0; ii<20; ii++){
    for (i2=0; i2<100; i2++){
      a1[ii][i2]=a0[i2]*dx[i1+ii];
      a2[ii][i2]=a0[i2]*dy[i1+ii];
    }
  }
  for (ii=0; ii<20; ii++){
    for (i3=0; i3<20; i3++){
      v1[ii]+=a1[ii][a3[i3]]-a2[ii][a3[i3]];
    }
  }
  for (ii=0; ii<20; ii++){
    if (v1[ii]>TH1) goto LABEL;
  }
}
LABEL:
```

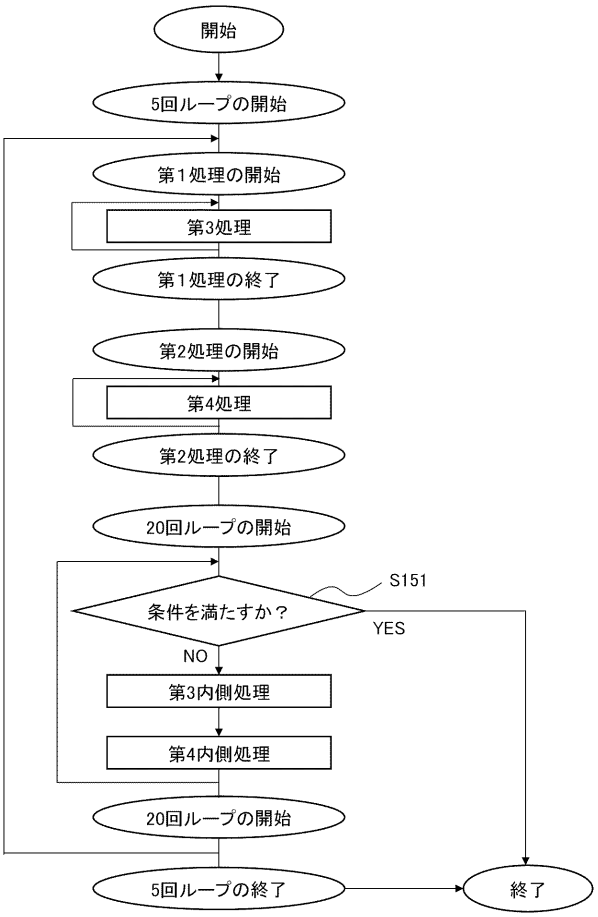
【図１１】



【図１２】

```
for (i1=0; i1<100; i1+=20){
  for (i2=0; i2<100; i2++){
    for (ii=0; ii<20; ii++){
      a1[i2][ii]=a0[i2]*dx[i1+ii];
      a2[i2][ii]=a0[i2]*dy[i1+ii];
    }
  }
  for (i3=0; i3<20; i3++){
    for (ii=0; ii<20; ii++){
      v1[ii]+=a1[a3[i3]][ii]-a2[a3[i3]][ii];
    }
  }
  for (ii=0; ii<20; ii++){
    if (v1[ii]>TH1) goto LABEL;
  }
}
LABEL:
```

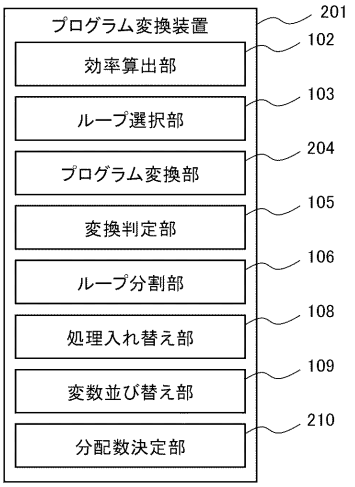
【図１３】



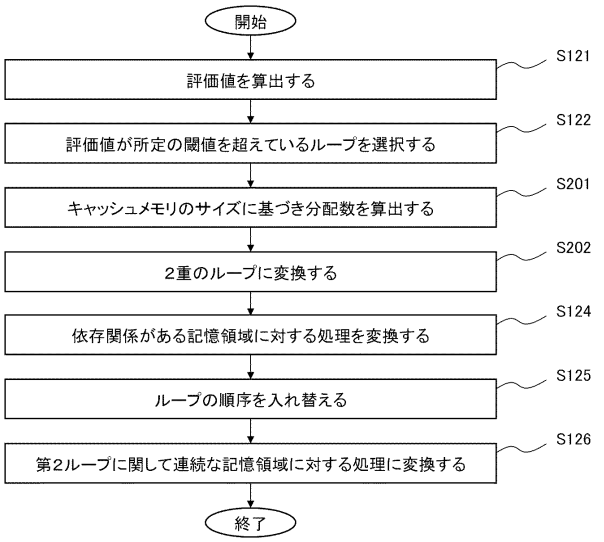
【図１４】

	通常的なベクトル化効率	理想的な期待ベクトル化効率	評価値	選択の有無
第1内側処理	1.0	4.0	4.0	選択する
第2内側処理	1.0	4.0	4.0	選択する
第3内側処理	1.0	1.2	1.2	選択しない
第4内側処理	1.0	1.0	1.0	選択しない

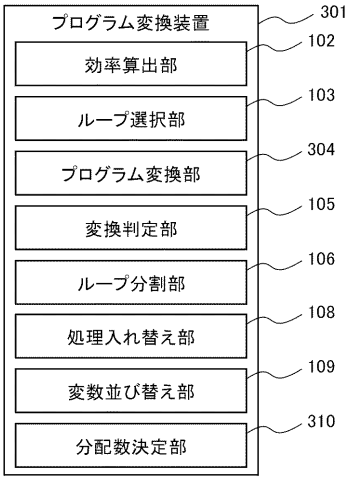
【図１５】



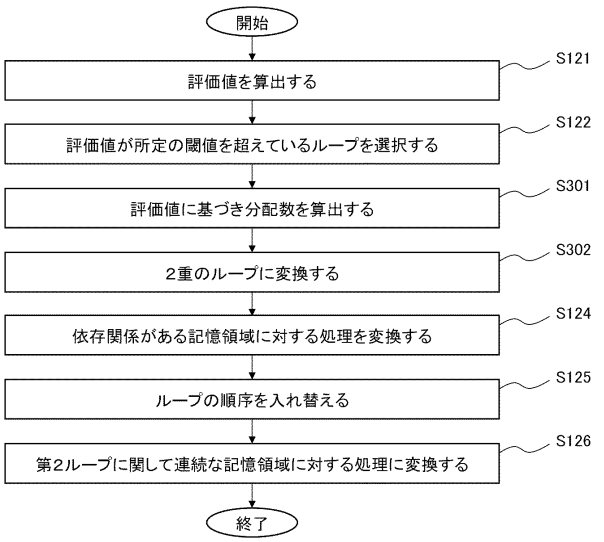
【図１６】



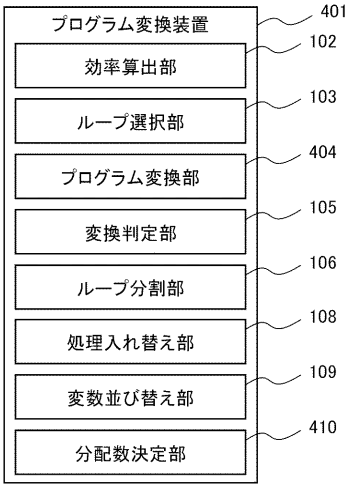
【図１７】



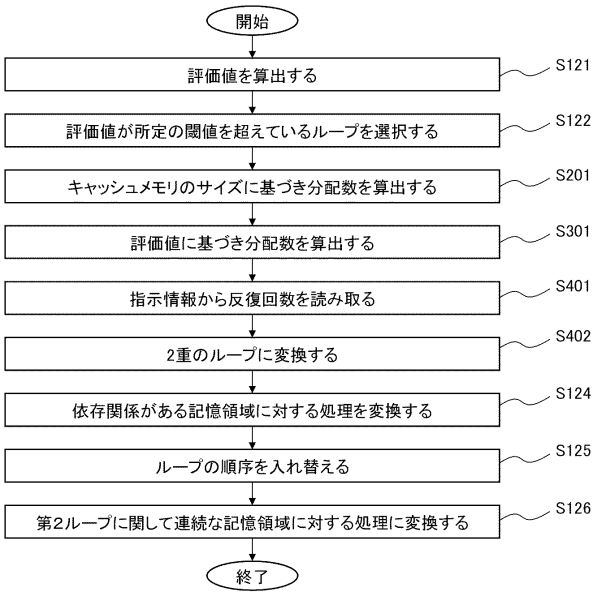
【図１８】



【図１９】



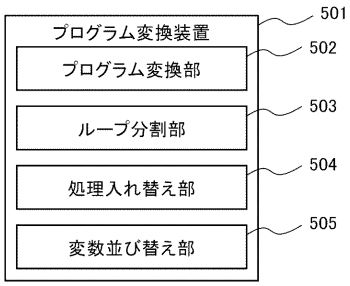
【図２０】



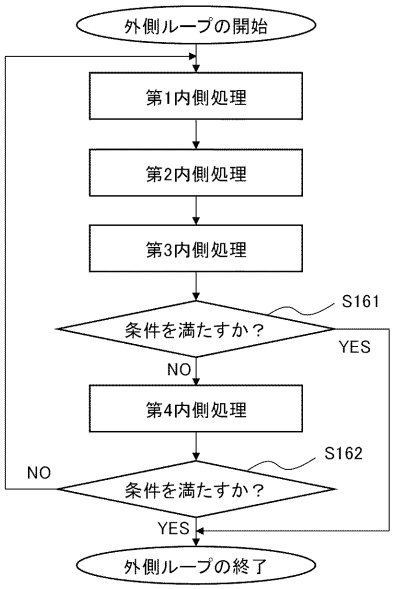
【図２１】

```
#pragma hint averageiters=40
{
for (i1=0; i1<100; i1++){
  for (i2=0; i2<100; i2++){
    a1[i2]=a0[i2]*dx[i1];
    a2[i2]=a0[i2]*dy[i1];
  }
  for (i3=0; i3<20; i3++){
    v1+=a1[a3[i3]]-a2[a3[i3]];
  }
  if (v1>TH1) goto LABEL;
}
LABEL:
}
```

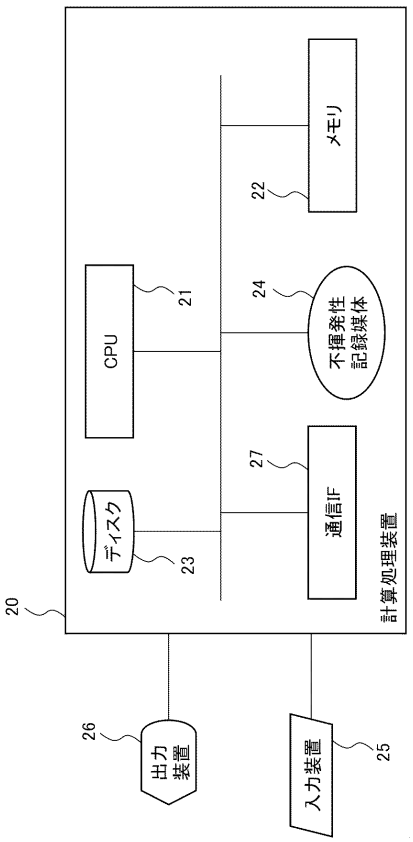
【図２２】



【図２３】



【図２４】



フロントページの続き

(56)参考文献 特開平０２－０５６０６５（ＪＰ，Ａ）
特開昭６３－０５８５７４（ＪＰ，Ａ）
特開平０５－０４０７８０（ＪＰ，Ａ）
特開昭６３－１５５２６４（ＪＰ，Ａ）
特開昭５８－１４９５４４（ＪＰ，Ａ）
高橋良明，速いＣＰＵのしくみ・最新技術 Part3 ＣＰＵを意識したプログラミング，Ｃ ＭＡＧＡＺＩＮＥ，日本，ソフトバンククリエイティブ株式会社，２００５年１１月 1日，第１７巻，第１１号（通巻１９４号），ｐｐ．３８－３９

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ８／４１