

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第5017410号
(P5017410)

(45) 発行日 平成24年9月5日(2012.9.5)

(24) 登録日 平成24年6月15日(2012.6.15)

(51) Int. Cl. F I
G06F 9/45 (2006.01) G O 6 F 9/44 3 2 2 G
G06F 9/50 (2006.01) G O 6 F 9/46 4 6 5 C

請求項の数 5 (全 15 頁)

(21) 出願番号	特願2010-73698 (P2010-73698)	(73) 特許権者	000003078 株式会社東芝
(22) 出願日	平成22年3月26日 (2010.3.26)		東京都港区芝浦一丁目1番1号
(65) 公開番号	特開2011-204209 (P2011-204209A)	(74) 代理人	100108855 弁理士 蔵田 昌俊
(43) 公開日	平成23年10月13日 (2011.10.13)	(74) 代理人	100159651 弁理士 高倉 成男
審査請求日	平成23年9月22日 (2011.9.22)	(74) 代理人	100091351 弁理士 河野 哲
		(74) 代理人	100088683 弁理士 中村 誠
		(74) 代理人	100109830 弁理士 福原 淑弘
		(74) 代理人	100075672 弁理士 峰 隆司

最終頁に続く

(54) 【発明の名称】 ソフトウェア変換プログラム、および、計算機システム

(57) 【特許請求の範囲】

【請求項1】

ホストプロセッサと一以上のアクセラレータプロセッサとを備える計算機システムに実行させるためのソフトウェア変換プログラムであって、

入力ソフトウェアを解析させ、ループ中の算術演算回数をループ中にアクセスするデータのサイズで除した計算密度と、データを参照する領域を合計したデータ参照領域サイズとを求めさせる手段と、

求められた各値と、予め用意された前記ホストプロセッサと前記アクセラレータプロセッサとの実行時間の優劣が定められている勝敗表とに基づいて、各ループを実行させるプロセッサを決定させる手段と、

該決定された各プロセッサで各ループが実行されるように入力ソフトウェアを変換させる手段として前記計算機システムを機能させるためのソフトウェア変換プログラム。

【請求項2】

更に、該ホストプロセッサのメインメモリとアクセラレータメモリとの間のデータ転送レートを示すデータ転送レートを求めさせる手段として前記計算機システムを機能させることを特徴とする、請求項1に記載のソフトウェア変換プログラム。

【請求項3】

更に、テストプログラムのループ処理で参照されるデータの重なり度合いを示すデータ参照領域重なり率を求めさせる手段として前記計算機システムを機能させることを特徴とする、請求項2に記載のソフトウェア変換プログラム。

【請求項 4】

前記勝敗表は、予め定めた複数の前記計算密度と前記データ参照領域サイズと前記データ転送レートと前記データ参照重なり率とをそれぞれ組み合わせた際に、前記マイクロプロセッサ及び前記アクセラレータプロセッサに、テストプログラムを実行させて得た実行時間の優劣をそれぞれ判断し、優劣の結果に従って作成されたものであることを特徴とする、請求項 1 乃至 3 の何れか 1 項 に記載のソフトウェア変換プログラム。

【請求項 5】

ホストプロセッサと、
 ー以上のアクセラレータプロセッサと、
 入力ソフトウェアを解析し、ループ中の算術演算回数をループ中にアクセスするデータのサイズで除した計算密度を求める第 1 取得手段と、
 データを参照する領域を合計したデータ参照領域サイズを求める第 2 取得手段と、
 第 1 取得手段と第 2 取得手段とで求められた各値と、予め用意された前記ホストプロセッサと前記アクセラレータプロセッサとの実行時間の優劣が定められている勝敗表とに基づいて、前記入力ソフトウェア内の各ループを実行させるプロセッサを決定する決定手段と、
 該決定手段で決定された各プロセッサで各ループが実行されるように入力ソフトウェアを変換する変換手段とを備えたことを特徴とする、
 計算機システム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、計算機で実行するためのソフトウェアを高速に処理するために変換するソフトウェア変換プログラムに関する。

【背景技術】

【0002】

近年の計算機システムでは、ホストプロセッサから、GPU (Graphics Processing Unit) をグラフィックス処理のみならず汎用計算に利用する GPGPU (General Purpose GPU) や CELL プロセッサや DSP などのアクセラレータへ、実行するソフトウェア中の高い演算処理能力を要する演算処理を、移行して実行 (以下、オフロードと称す) することで、プログラム全体の実行時間を小さくする技術が注目されている。

【0003】

例えば、非特許文献 1 に明示される C 言語コンパイラを利用すれば、入力されるソフトウェア内に含まれるループ処理をアクセラレータへオフロードすることができる。

【0004】

アクセラレータへ演算処理をオフロードするためには、演算処理に必要なデータを事前にアクセラレータのデバイスメモリへデータ転送する必要がある。

【0005】

従って、アクセラレータへ演算処理をオフロードしたほうが良いか否かを、ソフトウェア作成者がソフトウェア作成時に判断し、オフロードする場合にはその旨をソフトウェアへ予め含めておくことが必要である。一般に、ソフトウェア作成者は、「ループ中の算術演算回数」を、「ループ中にアクセスするデータのサイズで除した値 (= 「演算密度」)」を元に、アクセラレータへ演算処理をオフロードするか否かを判断していた。

【先行技術文献】

【非特許文献】

【0006】

【非特許文献 1】「PGI Fortran & C Accelerator Programming Model v1.0, The Portland Group, June 2009」

【発明の概要】

【発明が解決しようとする課題】

【0007】

しかしながら、計算機システムでソフトウェアを実行すると、データ転送サイズの変更による実際のデータ転送レートの変化や、ホストプロセッサのキャッシュの振る舞いの影響などが発生するが、これを考慮しソフトウェア作成者がソフトウェアを作成することは難しく、或いは考慮して作成したとしても、実際に演算速度の向上に繋がるか不透明であった。

【0008】

本発明は上記に鑑みてなされたものであって、実際のデータ転送レートの変更、ホストプロセッサのキャッシュの振る舞いを考慮し、アクセラレータへオフロードするか否かを判断できるソフトウェア変換プログラムを提供することを目的とする。

10

【課題を解決するための手段】

【0009】

上述した課題を解決し、目的を達成するために、本発明は、ホストプロセッサと一以上のアクセラレータプロセッサとを備える計算機システムに実行させるためのソフトウェア変換プログラムであって、入力ソフトウェアを解析させ、ループ中の算術演算回数をループ中にアクセスするデータのサイズで除した計算密度と、データを参照する領域を合計したデータ参照領域サイズとを求めさせる手段と、求められた各値と、予め用意された前記ホストプロセッサと前記アクセラレータプロセッサとの実行時間の優劣が定められている勝敗表とに基づいて、各ループを実行させるプロセッサを決定させる手段と、該決定された各プロセッサで各ループが実行されるように入力ソフトウェアを変換させる手段とを備えたことを特徴とする。

20

【0010】

また、本発明の計算機システムは、ホストプロセッサと、一以上のアクセラレータプロセッサと、入力ソフトウェアを解析し、ループ中の算術演算回数をループ中にアクセスするデータのサイズで除した計算密度を求める第1取得手段と、データを参照する領域を合計したデータ参照領域サイズを求める第2取得手段と、第1取得手段と第2取得手段とで求められた各値と、予め用意された前記ホストプロセッサと前記アクセラレータプロセッサとの実行時間の優劣が定められている勝敗表とに基づいて、前記入力プロセッサ内の各ループを実行させるプロセッサを決定する決定手段と、該決定手段で決定された各プロセッサで各ループが実行されるように入力ソフトウェアを変換する変換手段とを備えたことを特徴とする。

30

【発明の効果】

【0011】

本発明によれば、転送サイズの変化による実際のデータ転送レートの変化や、ホストプロセッサのキャッシュの振る舞いの影響まで考慮することで、より正確にオフロード判定をすることが可能になる。

【図面の簡単な説明】

【0012】

40

【図1】本実施の形態が適用される計算機システムを例示する図。

【図2】本実施の形態の全体を示すフローチャート。

【図3】生成されるデータ転送時間表の一例を示す図。

【図4】勝敗表生成プログラム112の実行動作フローチャート。

【図5】テストプログラム113の一例。

【図6】<データ参照領域重なり率パラメタ, データ参照領域サイズパラメタ> = <50%, 6000>の組で特定される勝敗表601の例。

【図7】ソフトウェア変換プログラム114の構成を示す図。

【図8】入力ソフトウェアの例。

【図9】データ参照領域情報709の例。

50

【図10】データ転送領域情報の例。

【図11】データ参照領域サイズパラメタを求めるフローチャート。

【図12】まとめられたデータ参照領域情報とデータ参照領域サイズパラメタの一例。

【図13】データ参照領域重なり率パラメタを求めるフローを示す。

【図14】データ転送レートパラメタを求めるフロー。

【図15】勝敗表を補間した勝敗表1501を示す図。

【図16】生成される出力ソフトウェア1601の一例。

【発明を実施するための形態】

【0013】

以下に添付図面を参照して、本発明の一実施の形態について詳細に説明する。

10

【0014】

図1は、本実施の形態が適用される計算機システムを示している。計算機システムは、ホストプロセッサ101、キャッシュ102、メインメモリ103、アクセラレータプロセッサ104、アクセラレータメモリ105、データ転送装置106とを備え、データ転送装置106とメインメモリ103は、バス107を介し接続される。本実施の形態では、アクセラレータプロセッサ104、アクセラレータメモリ105、データ転送装置106を一組だけ備えているが、二組以上あっても良い。また、計算機システムは、特に図示しないが、HDDまたは不揮発性メモリで構成される半導体記憶装置等の二次記憶装置を備えており、更に、キーボードやマウス等の入力装置、表示装置等を備えていても良いことは勿論である。

20

【0015】

本計算システムに、データ転送測定プログラム111、勝敗表生成プログラム112、ソフトウェア変換プログラム114がインストールされた後、実行されることにより、本実施の形態が実現される。

【0016】

本実施の形態の全体のフローを図2に示しつつ、各プログラムについて説明する。

【0017】

計算機システム上でデータ転送測定プログラム111を実行すると、データサイズの異なる複数のデータのそれぞれを、メインメモリ103からアクセラレータメモリ105へ転送させ、各データの転送時間を測定し、各データのデータサイズと測定した転送時間とを対とし記録することによって、データ転送時間表を生成する(ステップ201)。生成されるデータ転送時間表の一例を図3に示す。データ転送時間表301の各エントリ302は、転送サイズと転送時間の組で構成される。なお、測定するデータのデータサイズは離散値で良く、実際に知りたいデータサイズがデータ転送時間表301に無い場合には、線形補間などを行い、補間値を用いればよい。なお、データ転送測定プログラム201の実行は、例えば、データ転送測定プログラム111が計算機システムにインストールされた際に行われる。

30

【0018】

次に、計算機システム上で勝負表生成プログラム112を実行すると、ホストプロセッサ101とアクセラレータプロセッサ104の双方でテストプログラム113を実行させた場合にどちらのプロセッサ101/104での実行が速いかを測定し、測定結果を示した勝敗表を生成する(ステップ202)。もし、アクセラレータプロセッサ104が複数ある場合には、アクセラレータプロセッサ104の個数だけ同様に実行し、該個数分の勝敗表を生成する。勝敗表生成プログラム112の実行動作の詳細は、後述する。なお、勝敗表生成プログラム112の実行は、(前述のデータ転送時間表の生成後で、且つ)例えば、勝敗表生成プログラム112を計算機システムにインストールする時に行われる。

40

次に、計算機システム上でソフトウェア変換プログラム114を実行すると、ユーザが計算機システムで実行しようとする入力ソフトウェアに含まれるループ処理をアクセラレータプロセッサ104へオフロードするか否かを、前記勝敗表を参照して判定し、オフロードすると判定した場合に、入力ソフトウェアの変換処理を行う(ステップ203)。ソ

50

ソフトウェア変換プログラム 114 の実行動作の詳細は、後述する。

【0019】

以上のようなフローにより、データ転送レートやホストプロセッサのキャッシュの振る舞いの影響など計算機システムの実際の動作に基づいた勝敗表を利用するから、より正確なオフロード判定が可能となる。

【0020】

次に、勝敗表生成プログラム 112 の実行動作について、以下に詳細に説明する。勝敗表生成プログラム 112 は、オフロード判定に用いる勝敗表を設定するために、「計算密度パラメタ」、「データ参照領域サイズパラメタ」、「データ参照領域重なり率パラメタ」、「データ転送レートパラメタ」の4つのパラメタの組み合わせを変更しつつ、テストプログラム 113 を実行することによって勝敗表を生成するものである。各パラメタの詳細について後述する。

10

【0021】

図4は、勝敗表生成プログラム 112 の実行動作フローを示している。

【0022】

まず、勝敗表生成プログラム 112 は、各パラメタの組み合わせを全通り生成する（ステップ401）。例えば、4つのパラメタが、「計算密度パラメタ：1, 3, 5の3通り」、「データ参照領域パラメタ：600, 6000の2通り」、「データ転送レートパラメタ：1.0, 1.8, 4.7の3通り」、「重なり率パラメタ：0, 50の2通り」の場合、組み合わせの数（全通り）は、 $3 \times 2 \times 3 \times 2 = 36$ 通りとなる。なお、各パラメタの全通りの組み合わせを事前に求めて、勝敗表生成プログラム 112 内に、予め記録しておいても良い。

20

【0023】

次に、勝敗表生成プログラム 112 は、全てのパラメタの組み合わせでテストプログラム 113 を実行したか否かを確認する（ステップ402）。このステップの結果が Yes (Y) の場合、本実行動作の処理は終了し、勝敗表の生成が完了する。

【0024】

一方、No (N) の場合、即ち、全ての組み合わせの処理が完了していない場合には、勝敗表生成プログラム 112 は、ホストプロセッサ 101 とアクセラレータプロセッサ 104 の双方で、未だ実行していない組み合わせから一組取り出して、その組の各パラメタでテストプログラム 113 を実行し、それぞれの実行時間を測定する（ステップ403）。

30

【0025】

勝敗表生成プログラム 112 は、ステップ403で測定された両実行時間のうち、短かった方を勝者として勝敗表の対応するエントリに記録する（ステップ404）。そして、ステップ402へ戻る。

【0026】

図5に、テストプログラム 113 の一例を示す。このテストプログラム 501 は、C言語で書かれているが、他のプログラミング言語であっても良い。

【0027】

テストプログラムは、多重ループ 504 を含み、多重ループ 504 内では、配列変数 IN、OUT を参照している。

40

【0028】

データ転送指示文箇所 502 は、ホストプロセッサ 101 で実行されるテストプログラムには書かれておらず、アクセラレータプロセッサ 104 で実行されるテストプログラムに書かれているものである。このデータ転送指示文箇所 502 は、アクセラレータプロセッサ 104 で実行するためにアクセラレータメモリ 105 へデータを転送するためのデータ転送指示文である。データ転送指示文は、例えば、`#pragma transfer ()` と表し、引数にデータ転送する範囲を指定する。この範囲ごとにデータ転送が行われる。データ転送指示文で指定する配列の範囲は、部分配列の形で指定する。例えば、配

50

列変数名 [1次元目の開始インデックス番号 : 1次元目の終了インデックス番号] [2次元目の開始インデックス番号 : 2次元目の終了インデックス番号] で表す。図中のデータ転送範囲 $IN[0 : 2 * N - 1][0 : M - 1]$ は、 $IN[0][0]$ から $IN[2 * N - 1][M - 1]$ までを表すとする。

【 0 0 2 9 】

テスト内容箇所 5 0 3 には、テスト内容文が挿入される。

【 0 0 3 0 】

次に、上述の4つのパラメタについて、以下に説明する。

【 0 0 3 1 】

「計算密度パラメタ」は、「ループ中の算術演算回数」を「ループ中にアクセスするデータのサイズ」で除した結果値である。「計算密度パラメタ」は、テスト内容文箇所 5 0 2 に挿入するテスト内容文を変更することで、変更される。例えば、まず、テスト内容文が図 5 中の

$OUT[i][j] = (IN[i * 2][j] * IN[i * 2][j]) * (IN[i * 2 + 1][j] * IN[i * 2 + 1][j]);$

の場合、配列変数 IN の 2 要素を各々 1 回ずつ乗算した結果同士を乗算しその結果を配列変数 OUT の対応する要素へ代入するので、多重ループの算術演算回数 = 3、ループでアクセスするデータサイズ = 3 要素なので、計算密度 = $3 / 3 = 1$ となる。また、配列変数 IN の 2 変数の各々の乗算を 4 回ずつ、7 回ずつ行うようにテスト内容文を変更すると、多重ループの算術演算回数は 9 回、15 回と変更される。その結果、計算密度をそれぞれ $3 (= 9 / 3)$ 、 $5 (= 15 / 3)$ となる。

【 0 0 3 2 】

「データ参照領域サイズパラメタ」は、プログラムを実行するためのデータを参照する領域の合計のサイズを示す値である。「データ参照領域サイズパラメタ」は、2次元配列を表す変数 IN と OUT の 1次元目の長さである N を変更することで、変更させる。 $N = 4$ の場合、データ参照領域サイズは、配列 OUT 分の $200 (= N * M)$ と配列 IN 分の $400 (= OUT$ 分の倍) の合計で、 600 となる。例えば、 $N = 40$ と変更することで、データ参照領域サイズは、配列 OUT 分の $2000 (= N * M)$ と配列 IN 分 ($= OUT$ 分の倍) の 4000 の合計で、 6000 と変更させることができる。

【 0 0 3 3 】

「データ転送レートパラメタ」は、メインメモリからアクセラレータメモリへのデータ転送レートを示す値である。「データ転送レートパラメタ」は、データ転送指示文箇所 5 0 2 に挿入するデータ転送指示文を変更することで、変更させる。図 5 中の

`#pragma transfer (IN[0 : 2 * N - 1][0 : M - 1]), #pragma transfer (OUT[0 : N - 1][0 : M - 1])`

では、配列 IN 全体と配列 OUT 全体が、それぞれ転送される。配列 IN 全体は転送サイズ = $2N * M = 400$ 、配列 OUT 全体は転送サイズ = $N * M = 200$ となり、転送サイズ s での転送時間を $t(s)$ と表記すると、転送時間の合計は $t(400) + t(200)$ となる。平均データ転送レートは、(配列 IN 全体の転送サイズ + 配列 OUT 全体の転送サイズ) / $t(400) + t(200)$ で求めることができる。データ転送時間表 3 0 1 より、 $t(400) = 69$ 、 $t(200) = 59$ と線形補間で求めることができるので、平均データ転送レートは 4.7 と計算できる。例えば、データ転送指示文を、

`#pragma transfer (OUT[0 : 0][0 : M - 1], OUT[1 : 1][0 : M - 1], OUT[2 : 2][0 : M - 1], OUT[3 : 3][0 : M - 1])`

と 4 分割して書くと、各行が個別に転送されるとする。配列 IN と配列 OUT の両方で、転送サイズ = 50 となり、平均データ転送レートは (配列 IN 全体サイズ + 配列 OUT 全体サイズ) / $t(50) * 12$ となる。データ転送時間表 3 0 1 より $t(50) = 52$ と計算できるから、データ転送レートは 1.0 と計算できる。同様に、2 分割して書くと、個々のデータ転送サイズ = 100 となり、 $t(100) = 55$ と計算できるから、データ

10

20

30

40

50

転送レートは (配列 I N 全体サイズ + 配列 O U T 全体サイズ) / t (1 0 0) * 6 = 1 . 8 と計算できる。

【 0 0 3 4 】

「データ参照領域重なり率パラメタ」は、テストプログラムのループ処理で参照されるデータの重なり度合いを示す値である。「データ参照領域重なり率パラメタ」は、テスト内容文箇所 5 0 5 に挿入するテスト内容文を変更することで、変更させる。例えば、テスト内容文箇所 5 0 5 に挿入されているテスト内容文では、変数 i が更新される度に、配列の違う行を参照するので重なりは 0 % である。このテスト内容文を

$OUT[i][j] = (IN[i][j] * IN[i][j]) * (IN[i+2][j] * IN[i+2][j]);$

に変更する。この場合、 $i = k$ のときの $IN[i+2][j]$ と $i = k + 1$ のときの $IN[i][j]$ が重なる (行が重なる) ので、毎回 5 0 % の重なりがあるように変更させることができる。

【 0 0 3 5 】

勝敗表 6 0 1 は、アクセラレータ毎に [データ参照領域重なり率パラメタのサンプル数 × データ参照領域サイズパラメタのサンプル数] 個用意する。例えば、前者のサンプルが 0 % と 5 0 % の 2 つ、後者のサンプルが 6 0 0 、 6 0 0 0 の 2 つの場合、合計 4 つの勝敗表を生成する。なお、ここでは、データ参照領域重なりパラメタとデータ参照領域サイズパラメタとの組み合わせごとに勝敗表を生成しているが、4 つのパラメタの別の 2 つのパラメタの組み合わせごとに勝敗表を生成しても良い。

【 0 0 3 6 】

図 6 に、< データ参照領域重なり率パラメタ , データ参照領域サイズパラメタ > = < 5 0 % , 6 0 0 0 > の組で特定される勝敗表 6 0 1 の例を示す。

【 0 0 3 7 】

勝敗表 6 0 1 は、第 1 軸が「データ転送レート」、第 2 軸が「計算密度」となっている。表の各エントリには、 x を格納する。 x はアクセラレータでの実行時間がホストプロセッサでの実行時間より小さくなった場合 (オフロードした方が速くなった) に格納する。 x は逆にホストプロセッサの実行時間が小さくなった場合に格納する (オフロードするとかえって遅くなった) 。勝敗表を参照する際に、測定値がない値の場合は、簡単な補間を行いその補間値を用いればよい。

【 0 0 3 8 】

次に、ソフトウェア変換プログラム 1 1 4 の実行動作について、以下に詳細に説明する。

【 0 0 3 9 】

図 7 は、ソフトウェア変換プログラム 1 1 4 の構成を示している。

【 0 0 4 0 】

ソフトウェア変換プログラム 1 1 4 は、ユーザがこれから計算機システムで実行する入力ソフトウェア 7 0 2 を解析し、解析結果に基づいて、必要に応じて入力ソフトウェア 7 0 2 を変換して出力ソフトウェア 7 0 3 を生成し出力するものである。

データ参照領域解析部 7 0 4 は、入力ソフトウェア 7 0 2 を解析し、入力ソフトウェア 7 0 2 が参照するデータの領域のそれぞれを抽出し、データ参照領域情報 7 0 9 を生成する。

【 0 0 4 1 】

入力ソフトウェアの例を図 8 に示す。この入力ソフトウェア 8 0 1 は、多重ループ 8 0 2 を含み、多重ループ 8 0 2 内では、配列変数 A および B を参照するものである。なお、入力ソフトウェアは、C 言語で書かれているが、他のプログラミング言語であっても良い。

【 0 0 4 2 】

データ参照領域情報 7 0 9 の例を図 9 に示す。データ参照領域情報 9 0 1 、 9 0 2 の各データ参照領域 9 0 3 には、データ参照領域の先頭アドレスと末尾アドレスが記録される

10

20

30

40

50

。データ参照領域情報 901 は、入力ソフトウェアの配列変数 A の先頭アドレスが 10000 番地の場合、データ参照領域情報 902 は、配列変数 B の先頭アドレスが 20000 番地である場合の例を示している。

【0043】

次に、データ転送領域解析部 705 は、生成されたデータ参照領域情報 709 に基づいて、データ参照領域毎にデータ転送を行う方式（A方式）、所定ルールによって近接するデータ参照領域同士をまとめてデータ転送を行う方式（B方式）、及び、所定ルールによって全てのデータ参照領域をまとめてデータ転送を行う方式（C方式）、のそれぞれの方式について、事前に生成しておいた図3のデータ転送時間表 301 を用いてデータ転送時間を求め、最小のデータ転送時間値となる方式を選択し、その方式でデータ転送する領域を示すデータ転送領域情報 710 を生成する。

10

【0044】

例えば、入力ソフトウェア 701 の配列 B については、該 A 方式での転送時間は「 $4 * t(998) = 4 * 95.8 = 383$ 」となり、該 B 方式および該 C 方式での転送時間は「 $t(3998) = 230$ 」となる。よって、該 B 方式または該 C 方式を採用した方が転送時間を小さくできることが分かる。この結果得られるデータ転送領域情報の例を図10に示す。

【0045】

なお、データ転送領域解析部 705 で行われる処理の詳細は、文献[城田祐介ら，情報処理学会研究報告，ハイパフォーマンスコンピューティング，2006(87)，pp. 293 - 298]で示される。

20

【0046】

次に、パラメタ解析部 706 は、データ参照領域情報 709 からデータ参照領域サイズパラメタを求め、入力プログラムから計算密度パラメタを求め、データ参照領域情報 709 からデータ参照領域重なり率パラメタを求め、データ転送領域情報 710 からデータ転送レートパラメタを求め、パラメタ情報 711 を生成する。

【0047】

図11に、データ参照領域サイズパラメタを求めるフローを示す。

【0048】

始めに、データ参照領域を先頭アドレスの昇順でソートする（ステップ 1101）。

30

【0049】

次に、データ参照領域情報に含まれる全てのデータ参照領域を処理したか確認する（ステップ 1102）。

【0050】

この結果、処理が完了していない場合には、処理対象のデータ参照領域と一つ前のデータ参照領域の2つのデータ参照領域間に重なりの有無を確認する（ステップ 1103）。

【0051】

この結果、重なりがある場合には、2つのデータ参照領域を融合し、融合したデータ参照領域の先頭アドレスには、一つ前のデータ参照領域の先頭アドレスを、末尾アドレスには処理対象のデータ参照領域の末尾アドレスを設定する（ステップ 1104）。一方、重なりが無い場合には、S1102へ戻る。

40

【0052】

S1102において、データ参照領域情報に含まれる全てのデータ参照領域を処理が完了したと確認した場合には、まとめられたデータ参照領域のサイズの総和を求める（ステップ 1105）。以上によって、データ参照領域サイズパラメタを求める。

【0053】

図12には、まとめられたデータ参照領域情報とデータ参照領域サイズパラメタの一例を示す。この場合、データ参照領域サイズパラメタ = $6000 + 998 * 4 = 9992$ となっている。

【0054】

50

次に、計算密度パラメタの求め方について説明する。計算密度パラメタは、対象の多重ループの算術演算回数/ループでアクセスするデータサイズで求める。対象の多重ループでは、算術演算回数はイタレーション回数が $(N - 2) * (M - 2)$ 回、各イタレーションで 8 回の算術演算を行うので多重ループ全体では、

$(N - 2) * (M - 2) * 8 = 4 * 998 * 8 = 31936$ 回となる。一方、ループでアクセスするのは、上記で計算したデータ参照領域サイズパラメタなので、 $31936 / 992 = 3.2$ と容易に求めることができる。

【0055】

次に、図13に、データ参照領域重なり率パラメタを求めるフローを示す。

【0056】

始めに、重なり総サイズとデータ参照領域のデータ参照総サイズを 0 に初期化する（ステップ1301）。次に、データ参照領域情報に含まれる全てのデータ参照領域を処理したか確認する（ステップ1302）。

【0057】

ステップ1302において、処理が完了していない場合には、処理対象のデータ参照領域と一つ前のデータ参照領域との2つのデータ参照領域間の重なりサイズを計算する（ステップ1303）。

【0058】

計算された重なりサイズを該重なり総サイズに加算し、また、データ参照領域のサイズをデータ参照総サイズに加算する（ステップ1304）。

【0059】

ステップ1302に戻り、処理が完了した場合には、重なり率を、重なりサイズ総和/データ参照領域サイズ総和で計算し、データ参照領域重なり率パラメタとする（ステップ1305）。

【0060】

この例では、計算した結果、データ参照領域重なり率パラメタ = 67% となる。

【0061】

次に、図14に、データ転送レートパラメタを求めるフローを示す。

【0062】

始めに、データ総転送時間を 0 に初期化する（ステップ1401）。次に、データ参照領域情報に含まれる全てのデータ転送領域を処理したか確認する（ステップ1402）。

【0063】

ステップ1402において、処理が完了していない場合には、処理対象のデータ転送領域の転送時間を求める（ステップ1403）。そして、データ総転送時間に、求められたデータ転送時間を加算する（ステップ1404）。

【0064】

ステップ1402に戻り、処理が完了した場合にはデータ転送レートを計算し、データ転送レートパラメタとする（ステップ1405）。

【0065】

このフローに従うと、データ転送レートパラメタは、 $((15999 - 10000 + 1) + (24998 - 21001 + 1)) / (t(6000) + t(3998))$ と計算できる。 $t(6000) = 326$ 、 $t(3998) = 234$ と計算できるから、データ転送レートパラメタ = 17.9 と計算できる。

以上のようにして、パラメタ解析部706は、データ参照領域サイズパラメタ、計算密度パラメタ、データ参照領域重なり率パラメタ、及びデータ転送レートパラメタを求め、パラメタ情報711を生成する。

【0066】

図7に戻り、オフロード判定部707は、パラメタ情報711に基づいて予め生成・記憶した勝敗表を選定し、アクセラレータ104へオフロードすべきか否かを判定する。

【0067】

10

20

30

40

50

オフロード判定部707は、パラメタ情報711のデータ参照領域重なりパラメタとデータ参照領域サイズパラメタから、簡単な補間により、最も近い勝敗表を選択する。本例においては、〈データ参照領域重なりパラメタ，データ参照領域サイズパラメタ〉 = 〈67%，9992〉なので、簡単な補間により、4つの表から一番近い〈50%，6000〉の組で特定される勝敗表601を選択する。

【0068】

次に、オフロード判定部707は、選択した勝敗表を補間して、勝敗表を作成する。本例では、勝敗表を補間して、図15のような勝敗表1501を作成する。

【0069】

オフロード判定部707は、(補間した)勝敗表に、パラメタ情報711の計算密度パラメタとデータ転送レートとを照らし、オフロードするか否かを判定する。本例では、補間した勝敗表1501より、計算密度 = 3.2、データ転送レート = 17.9なので、と判定、つまり、オフロードすべきと判定される。なお、この例では、データ参照領域重なりパラメタとデータ参照領域サイズパラメタとの組み合わせごとに、勝敗表を記憶しているから、データ参照領域重なりパラメタとデータ参照領域サイズパラメタとの組みで勝敗表を特定しているが、4つのパラメタから別の2つのパラメタ毎に勝敗表を記憶している場合には、その別の2つのパラメタで勝敗表を特定すればよい。

【0070】

図7に戻り、ソフトウェア変換部708は、オフロード判定部707からオフロードすべきと判定を受けると、入力ソフトウェア702へ、予め用意したオフロード指示文と、データ転送指示文とを挿入する、ソフトウェア変換を行って、出力ソフトウェア709を出力する。図16に、その結果、生成される出力ソフトウェア1601の一例を示す。なお、本例では、ソフトウェア変換は、コンパイラ指示文の挿入という形で行ったが、その限りではない。

【0071】

上記で説明した本実施の形態のソフトウェア変換プログラムは、計算密度、データ参照領域サイズ、データ転送レート、データ参照領域重なり率の4つのパラメタを用いてソフトウェアの変換の要否を判定したが、(性能は劣るが)計算密度、データ参照領域サイズの2つのパラメタを用いてソフトウェアの変換の要否を判定してもよく、これにデータ転送レートを加えた計3つのパラメタを用いてソフトウェアの変換の要否を判定しても

【0072】

以上詳細に説明した本実施の形態によれば、実際のデータ転送レートの変更、ホストプロセッサのキャッシュの振る舞いを考慮し、アクセラレータへオフロードするか否かを判断できるようになった。

【符号の説明】

【0073】

101・・・ホストプロセッサ、102・・・キャッシュ、
 103・・・メインメモリ、104・・・アクセラレータプロセッサ、
 105・・・アクセラレータメモリ、106・・・データ転送装置、
 107・・・バス
 111・・・データ転送測定プログラム、112・・・勝敗表生成プログラム
 113・・・テストプログラム、114・・・ソフトウェア変換プログラム
 704・・・データ参照領域解析部、705・・・データ転送領域解析部
 706・・・パラメタ解析部、707・・・オフロード判定部、
 708・・・ソフトウェア変換部

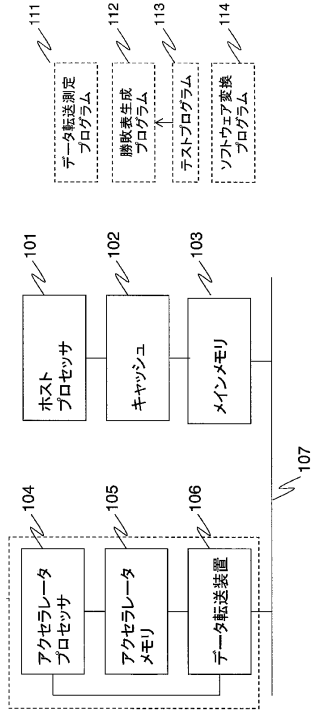
10

20

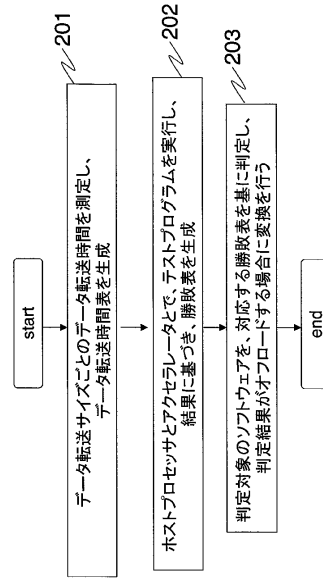
30

40

【図 1】



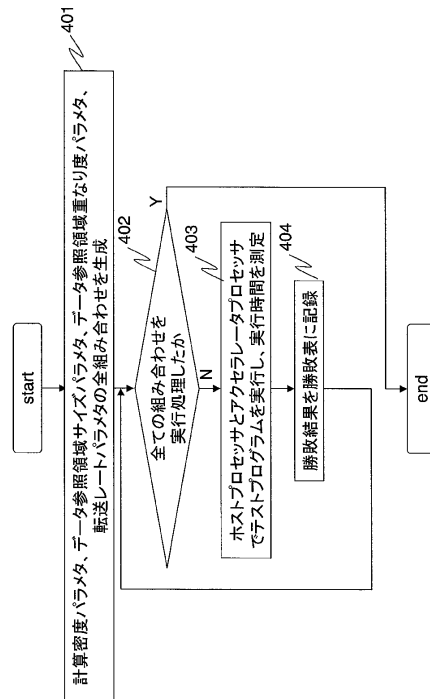
【図 2】



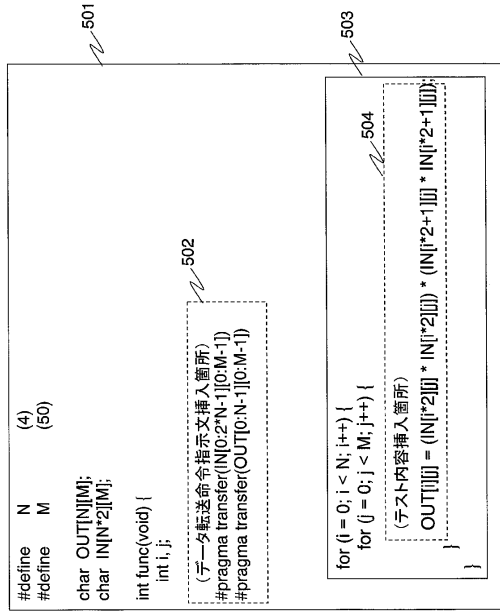
【図 3】

転送サイズ	転送時間
4	50
16	51
64	53
256	62
1024	97
4096	238
16384	804

【図 4】

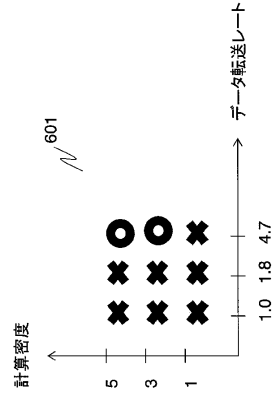


【 図 5 】

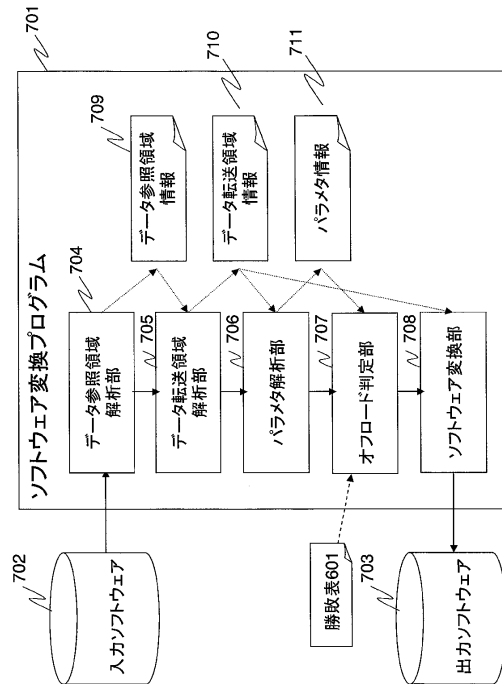


【 図 6 】

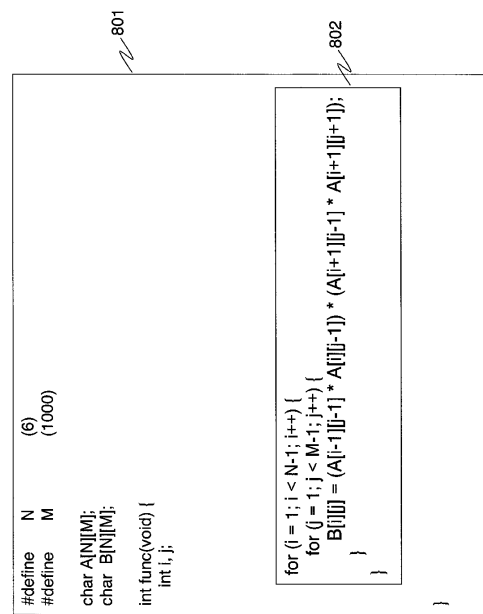
<データ参照領域重なり率パラメタ, データ参照領域サイズパラメタ> = <50%,600>



【 図 7 】



【 図 8 】



【図9】

配列A用 901

先頭アドレス	末尾アドレス
10000	12999
11000	13999
12000	14999
13000	15999

903

配列B用 902

先頭アドレス	末尾アドレス
21001	21998
22001	22998
23001	23998
24001	24998

【図10】

配列A用 1001

先頭アドレス	末尾アドレス
10000	15999

配列B用 1002

先頭アドレス	末尾アドレス
21001	24998

【図12】

配列A用 1201

先頭アドレス	末尾アドレス
10000	15999

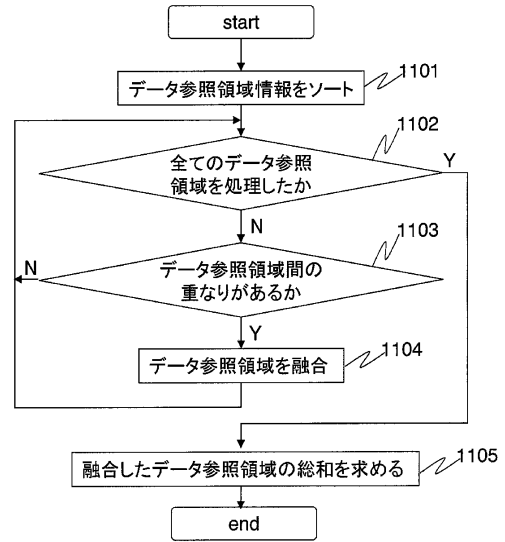
1203

配列B用 1202

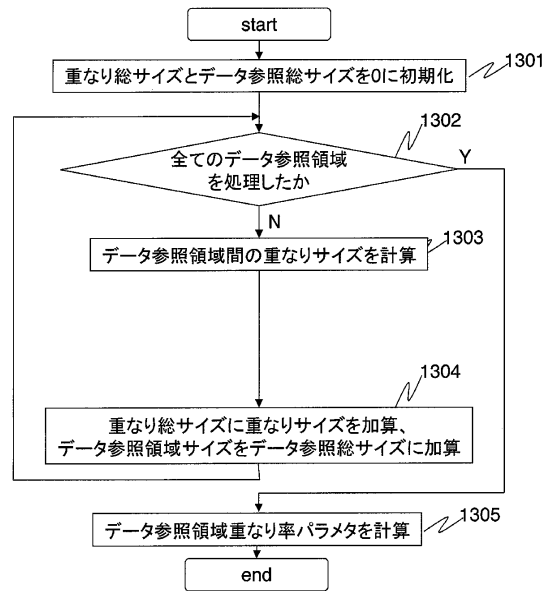
先頭アドレス	末尾アドレス
21001	21998
22001	22998
23001	23998
24001	24998

サイズパラメタ = 9992

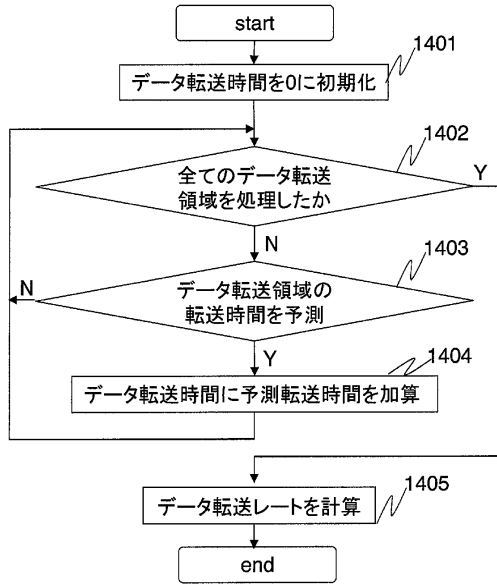
【図11】



【図13】

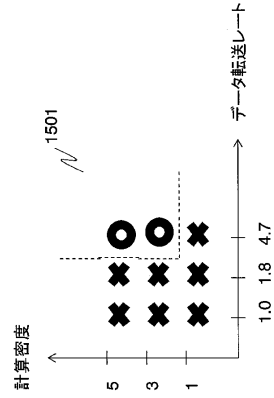


【図14】



【図15】

<データ参照領域重なり率パラメタ、データ参照領域サイズパラメタ> = <50%,600>



【図16】

```

    1601
    #define N (6)
    #define M (1000)
    char A[N][M];
    char B[N][M];
    int func(void) {
      int i, j;
      1602
      #pragma transfer(A[0:N-1][0:M-1])
      #pragma transfer(B[1:N-2][0:M-1]);
      1603
      #pragma offload (p)
      for (i = 1; i < N-1; i++) {
        for (j = 1; j < M-1; j++) {
          B[i][j] = (A[i-1][j-1] * A[i][j-1]) * (A[i+1][j-1] * A[i+1][j+1]);
        }
      }
    }
    1604
  
```

フロントページの続き

- (74)代理人 100095441
弁理士 白根 俊郎
- (74)代理人 100084618
弁理士 村松 貞男
- (74)代理人 100103034
弁理士 野河 信久
- (74)代理人 100119976
弁理士 幸長 保次郎
- (74)代理人 100153051
弁理士 河野 直樹
- (74)代理人 100140176
弁理士 砂川 克
- (74)代理人 100158805
弁理士 井関 守三
- (74)代理人 100124394
弁理士 佐藤 立志
- (74)代理人 100112807
弁理士 岡田 貴志
- (74)代理人 100111073
弁理士 堀内 美保子
- (74)代理人 100134290
弁理士 竹内 将訓
- (72)発明者 城田 祐介
東京都港区芝浦一丁目1番1号 株式会社東芝内
- (72)発明者 鳥井 修
東京都港区芝浦一丁目1番1号 株式会社東芝内

審査官 坂庭 剛史

- (56)参考文献 特開平04-348483(JP,A)
特開2004-046747(JP,A)
特開2008-276395(JP,A)
特開昭63-075935(JP,A)
町田智志、中西 悠、平澤将一、本多弘樹、POSIXスレッドを用いたCellプロセッサ向けAPIの提案、先進的計算基盤システムシンポジウム SAC SIS 2007 論文集、日本、社団法人情報処理学会、2007年 5月23日、Vol.2007, No.5, pp.162~163
檜田和浩、Cell Broadband Engineを用いたフェイストラッキング、東芝レビュー、日本、株式会社東芝、2007年 6月 1日、Vol.62, No.6, p.58
城田祐介、橋内和也、松崎秀則、前田誠司、CellプロセッサにおけるDMA転送の融合方式の提案、情報処理学会研究報告、日本、社団法人情報処理学会、2006年 8月 2日、Vol.2006, No.87, pp.293~298, 2006-HPC-107(50)

(58)調査した分野(Int.Cl., DB名)

G06F 9/45

G06F 9/50