(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第5464146号
(P5464146)

(45)発行日　平成26年4月9日(2014.4.9)　(24)登録日　平成26年1月31日(2014.1.31)

(51)Int.Cl.　　F I
G06F 9/50 (2006.01)　　G06F 9/46 465E
G06F 9/48 (2006.01)　　G06F 9/46 452H

請求項の数 5　(全 25 頁)

(21)出願番号　特願2010-537726 (P2010-537726)
(86)(22)出願日　平成21年8月20日 (2009.8.20)
(86)国際出願番号　PCT/JP2009/064541
(87)国際公開番号　WO2010/055719
(87)国際公開日　平成22年5月20日 (2010.5.20)
審査請求日　平成24年7月12日 (2012.7.12)
(31)優先権主張番号　特願2008-292119 (P2008-292119)
(32)優先日　平成20年11月14日 (2008.11.14)
(33)優先権主張国　日本国(JP)

前置審査

(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　100123788
弁理士　宮崎　昭夫
(74)代理人　100106138
弁理士　石橋　政幸
(74)代理人　100127454
弁理士　緒方　雅昭
(72)発明者　鈴木　紀章
東京都港区芝五丁目７番１号　日本電気株式会社内
(72)発明者　酒井　淳嗣
東京都港区芝五丁目７番１号　日本電気株式会社内

最終頁に続く

(54)【発明の名称】スケジュール決定装置

(57)【特許請求の範囲】
【請求項１】
第１のタスクが分割された複数のサブタスクが満たすべき実行順序の依存関係を取得する依存関係取得手段と、
前記依存関係取得手段により取得された前記依存関係を満たし、かつ、前記複数のサブタスクを複数の処理装置で実行する複数のサブタスク構成候補を作成する候補作成手段と、
前記候補作成手段により作成された前記サブタスク構成候補のそれぞれに、更に少なくとも１つの第２のタスクを配置することにより、複数のスケジュール候補を作成するスケジューリング手段と、
前記スケジューリング手段により作成された前記複数のスケジュール候補のそれぞれについて、前記第１のタスクの有効度と前記第２のタスクの有効度とのそれぞれから導出される、前記第１のタスク及び前記第２のタスクの実行に対する有効性を示す有効度を算出する有効度算出手段と、
前記有効度算出手段により算出された前記有効度に基づいて、前記複数のスケジュール候補の中から、前記第１のタスク及び前記第２のタスクの実行に用いるスケジュール候補を決定する候補決定手段と、
前記候補作成手段により作成された前記複数のサブタスク構成候補を複数のグループに分類し、該グループに属する第１及び第２のサブタスク構成候補を選択し、前記第１のタスク構成候補の実行期間のうち前記第２のタスク構成候補の実行期間と一致しない期間を

、前記第２のタスク構成候補に追加する処理を行う重ね合わせ手段と、
を有するスケジュール決定装置。
【請求項２】
前記有効度算出手段は、前記重ね合わせ手段により前記処理が行われた前記グループに対応するスケジュール候補ごとに、有効度を更に算出し、算出した該有効度に基づいて、いずれか１つのグループを選択し、選択した該グループに属するサブタスク構成候補に対応するスケジュール候補ごとに、有効度を算出する、請求項１に記載のスケジュール決定装置。
【請求項３】
前記重ね合わせ手段は、各グループに属する前記サブタスク構成候補の数の差が所定値以下になるように、前記複数の候補を分類する、請求項２に記載のスケジュール決定装置。
【請求項４】
前記重ね合わせ手段は、それぞれのサブタスク構成候補において所定の処理装置に配置されたサブタスクの実行時間の合計値を算出し、各グループに属する、それぞれのサブタスク構成候補について算出された該合計値がグループごとに予め定められた範囲内になるように、前記複数の候補を分類する、請求項２又は３に記載のスケジュール決定装置。
【請求項５】
前記重ね合わせ手段は、第１の処理装置でサブタスクの実行が開始された時刻から第２の処理装置でサブタスクの実行が開始されるまでの時間であるオフセットが所定値以上異なるサブタスク構成候補が異なるグループに属するように、前記複数の候補を分類する、請求項２乃至４のいずれか１項に記載のスケジュール決定装置。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、複数のタスクを並列して実行するためのスケジューリングを行う技術に関する。
【背景技術】
【０００２】
近年では、デジタル機器の高性能化および低消費電力化の要求が高まっている。これらの要求を解決する手段として、組み込みＬＳＩ（Large Scale Integration）に複数のプロセッサを搭載し、マルチコア構成とする方法が注目を集めている。
【０００３】
このようなマルチコア構成の機器を効率的に利用するには、アプリケーションを並列に実行するために並列プログラミングを行う必要があるが、一般的に並列プログラミングは、従来のシングルコア用のプログラミングよりも考慮すべき点が多く、難易度が高い。
【０００４】
並列プログラミングを容易にするため、特許文献１に記載のマルチプロセッサシステムでは、コンパイラが、入力プログラムから並列性を持つサブタスクを抽出し、各プロセッサユニットの特性に合わせてサブタスクを配置することで、効率的なスケジューリングを行っている。
【０００５】
また、リアルタイムシステムにおいては、並列に実行されるアプリケーションがタスクの実行周期が経過するまでに、そのタスクが完了しない事態（デッドラインミス）とならないように、スケジューリングを行う必要がある。
【０００６】
このデッドラインミスを防止するため、特許文献２に記載のスケジューリング判定方法は、所定の判定式を使用することで、デッドラインミスを起こすことなく、マルチコア用のスケジュールを実行できるかどうかを判定している。
【先行技術文献】

【特許文献】
【０００７】
【特許文献１】特許４０８２７０６号
【特許文献２】特許４０１６０１０号
【発明の概要】
【発明が解決しようとする課題】
【０００８】
しかし、特許文献１に記載のスケジューリングで最適のものとして求めた並列実行形式は、複数のタスクが異なる周期で独立に起動するシステムでは、必ずしも最適のスケジューリングにならないという問題があった。
【０００９】
各タスクの実行周期が異なる場合、それぞれのタスクが必要に応じてプロセッサ（コア）を利用するため、あるタスクの影響により、他のタスクの実行が待たされる場合がある。
【００１０】
特に、複数コアを使用してタスクを並列に実行するようなスケジューリングがなされると、ある１つのコアが他のタスクを実行しているだけで、並列に実行すべきタスク（並列実行タスク）が待たされて、装置全体が、その並列実行タスクを全く実行できない状態が発生しうる。
【００１１】
このため、全てのタスクの影響を考慮すると、特許文献１に記載の自動並列化コンパイラによっても、スケジュールを最適化できないことがある。
【００１２】
特許文献２に記載の方法では、作成したスケジュールにおいてデッドラインミスが生じるか否か、つまり、そのスケジュールが複数のコアで並列実行できるか否かを判断できるが、並列実行できるスケジュールのうち、いずれがタスクの実行に最適であるかについては判断できないという問題があった。
【００１３】
本発明は、マルチコアで、複数のタスクを並列実行するスケジュールを最適化する技術を提供することを目的とする。
【課題を解決するための手段】
【００１４】
上記目的を達成するために、本発明のスケジュール決定装置は、第１のタスクが分割された複数のサブタスクが満たすべき実行順序の依存関係を取得する依存関係取得手段と、前記依存関係取得手段により取得された前記依存関係を満たし、かつ、前記複数のサブタスクを複数の処理装置で実行する複数のサブタスク構成候補を作成する候補作成手段と、前記候補作成手段により作成された前記サブタスク構成候補のそれぞれに、更に少なくとも１つの第２のタスクを配置することにより、複数のスケジュール候補を作成するスケジューリング手段と、前記スケジューリング手段により作成された前記複数のスケジュール候補のそれぞれについて、前記第１のタスク及び前記第２のタスクの実行に対する有効性を示す有効度を算出する有効度算出手段と、前記有効度算出手段により算出された前記有効度に基づいて、前記複数のスケジュール候補の中から、前記第１のタスク及び前記第２のタスクの実行に用いるスケジュール候補を決定する候補決定手段と、を有する。
【００１５】
本発明の並列実行装置は、第１のタスクが分割された複数のサブタスクが満たすべき実行順序の依存関係を取得する依存関係取得手段と、前記依存関係取得手段により取得された前記依存関係を満たし、かつ、前記複数のサブタスクを複数の処理装置で実行する複数のサブタスク構成候補を作成する候補作成手段と、前記候補作成手段により作成された前記サブタスク構成候補のそれぞれに、更に少なくとも１つの第２のタスクを配置することにより、複数のスケジュール候補を作成するスケジューリング手段と、前記スケジューリ

ング手段により作成された前記複数のスケジュール候補のそれぞれについて、前記第１のタスク及び前記第２のタスクの実行に対する有効性を示す有効度を算出する有効度算出手段と、前記有効度算出手段により算出された前記有効度に基づいて、前記複数のスケジュール候補の中から、前記第１のタスク及び前記第２のタスクの実行に用いるスケジュール候補を決定する候補決定手段と、前記候補決定手段により決定された前記スケジュール候補を使用して前記第１のタスク及び前記第２のタスクを実行する並列実行手段と、を有する。

【００１６】

本発明のスケジュール決定方法は、第１のタスクが分割された複数のサブタスクが満たすべき実行順序の依存関係を取得し、前記依存関係を満たし、かつ、前記複数のサブタスクを複数の処理装置で実行する複数のサブタスク構成候補を作成し、前記サブタスク構成候補のそれぞれに、更に少なくとも１つの第２のタスクを配置することにより、複数のスケジュール候補を作成し、前記複数のスケジュール候補のそれぞれについて、前記第１のタスク及び前記第２のタスクの実行に対する有効性を示す有効度を算出し、前記有効度に基づいて、前記複数のスケジュール候補の中から、前記第１のタスク及び前記第２のタスクの実行に用いるスケジュール候補を決定する。

【００１７】

本発明の第１の観点にかかるプログラムは、コンピュータに、第１のタスクが分割された複数のサブタスクが満たすべき実行順序の依存関係を取得する依存関係取得手順、前記依存関係取得手順で取得された前記依存関係を満たし、かつ、前記複数のサブタスクを複数の処理装置で実行する複数のサブタスク構成候補を作成する候補作成手順、前記候補作成手順で作成された前記サブタスク構成候補のそれぞれに、更に少なくとも１つの第２のタスクを配置することにより、複数のスケジュール候補を作成するスケジューリング手順、前記スケジューリング手順で作成された前記複数のスケジュール候補のそれぞれについて、前記第１のタスク及び前記第２のタスクの実行に対する有効性を示す有効度を算出する有効度算出手順、及び前記有効度算出手順で算出された前記有効度に基づいて、前記複数のスケジュール候補の中から、前記第１のタスク及び前記第２のタスクの実行に用いるスケジュール候補を決定する候補決定手順、を実行させるためのプログラムである。

【００１８】

本発明の第２の観点にかかるプログラムは、コンピュータに、第１のタスクが分割された複数のサブタスクが満たすべき実行順序の依存関係を取得する依存関係取得手順、前記依存関係取得手順で取得された前記依存関係を満たし、かつ、前記複数のサブタスクを複数の処理装置で実行する複数のサブタスク構成候補を作成する候補作成手順、前記候補作成手順で作成された前記サブタスク構成候補のそれぞれに、更に少なくとも１つの第２のタスクを配置することにより、複数のスケジュール候補を作成するスケジューリング手順、前記スケジューリング手順で作成された前記複数のスケジュール候補のそれぞれについて、前記第１のタスク及び前記第２のタスクの実行に対する有効性を示す有効度を算出する有効度算出手順、前記有効度算出手順で算出された前記有効度に基づいて、前記複数のスケジュール候補の中から、前記第１のタスク及び前記第２のタスクの実行に用いるスケジュール候補を決定する候補決定手順、及び前記候補決定手順で決定された前記スケジュール候補を使用して前記第１のタスク及び前記第２のタスクを実行する並列実行手順、を実行させるためのプログラムである。

【発明の効果】

【００１９】

本発明によれば、スケジュール決定装置は、第１のタスクから分割されたサブタスクを、依存関係を満たすように複数のコアに配置して複数のサブタスク構成候補を作成し、候補ごとに、第２のタスクを配置した場合におけるスケジューリング候補を作成し、各スケジューリング候補の有効度に基づいてスケジュールを決定する。スケジューリング決定装置は、サブタスクを配置した後に、その合間に第２のタスクを配置するので、並列に実行する第１のタスクが待たされて、スケジュール全体が実行不能となることが少なくなる。

また、スケジュール決定装置は、スケジューリング候補ごとに有効度を算出するので、有効度に基づいて最適のスケジューリング候補を求めることにより、スケジュールを最適化できる。

【図面の簡単な説明】

【００２０】

【図１】本発明の第1の実施形態のスケジュール決定装置の構成を示すブロック図である。

【図２】本発明の第1の実施形態の依存関係取得部の構成を示すブロック図である。

【図３】（ａ）本発明の第1の実施形態の並列実行タスクの形状を示す図である。（ｂ）本発明の第1の実施形態のシングル実行タスクの形状を示す図である。（ｃ）本発明の第1の実施形態のサブタスクの形状を示す図である。（ｄ）本発明の第1の実施形態のサブタスクの依存関係を示す図である。

【図４】本発明の第1の実施形態の候補作成部の構成を示すブロック図である。

【図５】（ａ）本発明の第1の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第1の実施形態のタスク構成候補の形状を示す図である。（ｃ）本発明の第1の実施形態のタスク構成候補の形状を示す図である。

【図６】（ａ）本発明の第1の実施形態の代表化されたタスク構成候補の形状を示す図である。（ｂ）本発明の第1の実施形態の代表化されたタスク構成候補の形状を示す図である。（ｃ）本発明の第1の実施形態の代表化されたタスク構成候補の形状を示す図である。

【図７】本発明の第1の実施形態のタスク構成候補の実行時間およびオフセットを示すテーブルである。

【図８】（ａ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｂ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｃ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｄ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｅ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｆ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｇ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｈ）本発明の第1の実施形態のスケジューリングの結果を示す図である。（ｉ）本発明の第1の実施形態のスケジューリングの結果を示す図である。

【図９】本発明の第1の実施形態の有効度スコアの算出結果を示すテーブルである。

【図１０】本発明の第1の実施形態の候補決定部の構成を示すブロック図である。

【図１１】本発明の第1の実施形態のスケジュール決定装置の動作を示すフローチャートである。

【図１２】本発明の第1の実施形態の依存関係作成処理を示すフローチャートである。

【図１３】本発明の第1の実施形態の候補作成処理を示すフローチャートである。

【図１４】本発明の第1の実施形態の候補決定処理を示すフローチャートである。

【図１５】本発明の第２の実施形態の候補作成処理を示すフローチャートである。

【図１６】（ａ）本発明の第２の実施形態の並列実行タスクの形状を示す図である。（ｂ）本発明の第２の実施形態のサブタスクの形状を示す図である。（ｃ）本発明の第２の実施形態のサブタスクの依存関係を示す図である。

【図１７】（ａ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｃ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｄ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｅ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｆ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｇ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｈ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。

【図１８】（ａ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｃ）本発明の第２

の実施形態のタスク構成候補の形状を示す図である。（ｄ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｅ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｆ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｇ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｈ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。
【図１９】本発明の第２の実施形態のスケジューリングの結果を示す図である。
【図２０】（ａ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第２の実施形態のタスク構成候補の形状を示す図である。
【図２１】本発明の第３の実施形態の候補作成部の構成を示すブロック図である。
【図２２】（ａ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｃ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｄ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｅ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。
【図２３】本発明の第３の実施形態の候補作成処理を示すフローチャートである。
【図２４】本発明の第３の実施形態のスケジュール決定装置の動作を示すフローチャートである。
【図２５】本発明の第３の実施形態の候補の分類結果を示すテーブルである。
【図２６】本発明の第３の実施形態の候補の分類結果を示すテーブルである。
【図２７】本発明の第３の実施形態の候補の分類結果を示すテーブルである。
【図２８】（ａ）本発明の第３の実施形態の並列実行タスクの形状を示す図である。（ｂ）本発明の第３の実施形態のサブタスクの形状を示す図である。（ｃ）本発明の第３の実施形態のサブタスクの依存関係を示す図である。
【図２９】（ａ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｃ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｄ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｅ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｆ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｇ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｈ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。
【図３０】（ａ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｂ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｃ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｄ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｅ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｆ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｇ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。（ｈ）本発明の第３の実施形態のタスク構成候補の形状を示す図である。
【図３１】本発明の第３の実施形態のスケジューリングの結果を示す図である。
【図３２】本発明の第３の実施形態の候補の分類結果を示すテーブルである。
【図３３】本発明の第３の実施形態の候補の分類結果を示すテーブルである。
【図３４】本発明の第４の実施形態の並列実行装置の構成を示すブロック図である。
【発明を実施するための形態】
【００２１】
（第１の実施形態）
本発明を実施するための第１の形態について図面を参照して詳細に説明する。
【００２２】
図１は、本実施形態のスケジュール決定装置１の構成を示すブロック図である。スケジュール決定装置１は、周期の異なる複数のタスクをマルチコアで並列実行する、最適のスケジュールを求める装置である。同図を参照すると、スケジュール決定装置１は、依存関係取得部１１、候補作成部１３、スケジューリング実行部１５、有効度スコア算出部１７

、および候補決定部１９を有する。
【００２３】
依存関係取得部１１は、所定の実行周期（第１の周期）のタスク（第１のタスク）を複数のサブタスクに分割し、これらのサブタスク間の実行順序の依存関係を求める。または、予め第１の周期のタスクを複数のサブタスクに分割したものを入力しサブタスク間の実行順序の依存関係を求めても良い。
【００２４】
候補作成部１３は、依存関係を満たすように、サブタスクを、複数のコアで並列実行するスケジュールの候補（スケジューリング候補）を、複数個作成する。
【００２５】
スケジューリング実行部１５は、候補ごとに、サブタスクの合間に、分割したタスク（第１のタスク）と別のタスク（第２のタスク）を仮想的に実行する。
【００２６】
有効度スコア算出部１７は、実行した結果から、候補ごとに、その候補のタスク実行のスケジュールとしての有効性を示す有効度スコアを算出する。有効度スコアの算出方法については、後述する。
【００２７】
候補決定部１９は、有効度スコアが最も高い候補を最適の候補に決定する。
【００２８】
図２は、依存関係取得部１１の構成を示すブロック図である。同図を参照すると、依存関係取得部１１は、タスク分割部１１１および依存グラフ作成部１１３を有する。
【００２９】
図３（ａ）、（ｂ）、（ｃ）、および（ｄ）は、タスク分割および依存関係について説明するための図である。スケジュール決定装置１は、複数の異なるタスクを、スケジューリングの対象とする。例えば、本実施形態では、図３（ａ）に示す、８０ミリ秒の周期で実行するＴａｓｋ０と、図３（ｂ）に示す、２００ミリ秒の周期で実行するＴａｓｋ１とについてスケジューリングを行う。また、同図（ａ）、（ｂ）におけるバーの長さは、各タスクの最大実行時間を示しており、例えば、Ｔａｓｋ０、Ｔａｓｋ１の最大実行時間は、ともに８０ミリ秒である。
【００３０】
タスク分割部１１１は、スケジューリング対象のタスクのうち、複数のコアで並列実行するタスク（以下「並列実行タスク」という）について、そのタスクを分割できる時点（分割点）を探索する。分割点の決定においては、タスク分割部１１１は、サブタスクに分割しても並列に実行できない性質のものについては一般には分割しないが、サブタスク間のデータの受け渡し量が少ないなど、依存が弱いものについては、分割する。但し、分割したタスク（以下、「サブタスク」という）ごとの最大実行時間などの時間情報を用いて最適化を行うので、タスクを構成するプログラムの分割は要しない。
【００３１】
ここで、タスク（サブタスク）ごとの最大実行時間は、特許文献１のｐ１１３～１１５に記載の方法を使用するなどして求められる。
【００３２】
そして、タスク分割部１１１は、その分割点で並列実行タスクを複数のサブタスクに分割する。１つのコアで実行するタスク（以下「シングル実行タスク」という）については、タスク分割部１１１は分割しない。例えば、本実施形態では、Ｔａｓｋ0が並列実行タスクであり、Ｔａｓｋ１はシングル実行タスクである。このため、タスク分割部１１１は、Ｔａｓｋ0のみをサブタスクに分割する。
【００３３】
図３（ｃ）は、タスク分割部１１１が分割した並列実行タスクを示す図である。例えば、Ｔａｓｋ０は、最大実行時間が２０ミリ秒のＴａｓｋ０[０]と、最大実行時間が３５ミリ秒のＴａｓｋ０[１]と、最大実行時間が２５ミリ秒のＴａｓｋ０[２]とに分割される。

【００３４】
依存グラフ作成部１１３は、サブタスク間の実行順序の依存関係を求める。図３（ｄ）は、依存グラフ作成部１１３により求められた依存関係を、図示したもの（依存グラフ）である。一般に単一のタスクを複数のサブタスクに分割した場合、それらのサブタスク間には実行順序に関する依存関係が生じる場合がある。例えば、本実施形態では、Ｔａｓｋ０[０]で計算した結果を、Ｔａｓｋ０[２]で参照するので、装置は、Ｔａｓｋ０[０]をＴａｓｋ０[２]より先に実行する必要がある。
【００３５】
図４は、候補作成部１３の構成を示すブロック図である。同図を参照すると、候補作成部１３は、タスク構成候補作成部１３１および代表化部１３３を有する。タスク候補作成部１３１は、依存関係を満たすように、全てのサブタスクを複数のコアで並列実行するスケジュールの候補（以下「タスク構成候補」という）を作成する。
【００３６】
例えば、本実施形態では、タスク構成候補作成部１３１は、２つのコア０、コア１で、Ｔａｓｋ０[０]がＴａｓｋ０[２]より先に実行されるように、Ｔａｓｋ０[０]、Ｔａｓｋ０[１]、およびＴａｓｋ０[２]を実行するスケジュール（タスク構成候補）を作成する。
【００３７】
図５（ａ）、（ｂ）、および（ｃ）は、作成されたタスク構成候補の一例である。同図（ａ）に示すタスク構成候補ａ１は、コア０で、Ｔａｓｋ０[１]、Ｔａｓｋ０[２]の順でサブタスクを実行し、コア１でＴａｓｋ０[０]を実行するスケジュールである。同図（ｂ）に示すタスク構成候補ａ２は、コア０で、Ｔａｓｋ０[０]、Ｔａｓｋ０[１]の順でサブタスクを実行し、Ｔａｓｋ０[０]の実行後、コア１でＴａｓｋ０[２]を実行するスケジュールである。同図（ｃ）に示すタスク構成候補ａ３は、コア０で、Ｔａｓｋ０[０]、Ｔａｓｋ０[２]の順でサブタスクを実行し、コア１でＴａｓｋ０[１]を実行するスケジュールである。
【００３８】
代表化部１３３は、図６（ａ）、（ｂ）、および（ｃ）に示すように、各タスク構成候補について代表化を行う。同図（ａ）、（ｂ）、および（ｃ）は、サブタスク構成候補ａ１、ａ２、およびａ３を、代表化した候補の構成を示す図である。代表化とは、タスク構成候補について、これ以降の処理において不要となるサブタスクの構成を隠蔽することである。
【００３９】
具体的には、代表化部１３３は、図７に示すように、タスク構成候補ごとに、各コアの「実行時間」（代表値）およびオフセット（代表値）を記載したテーブルを作成する。「実行時間」は、コアごとの、サブタスクの処理に要する時間の最大値（最大実行時間）である。オフセットは、あるコアでサブタスクの実行が開始されたときから、他のコアでサブタスクの実行が開始されるまでの時間である。
【００４０】
例えば、タスク構成候補ａ１では、コア０において、Ｔａｓｋ０[１]、Ｔａｓｋ０[２]の順でサブタスクが実行され、コア０でＴａｓｋ０[１]が開始されたときと同時にＴａｓｋ０[０]が開始される。Ｔａｓｋ０[０]、Ｔａｓｋ０[１]、およびＴａｓｋ０[２]の最大実行時間は、それぞれ２０ミリ秒、３５ミリ秒、２５ミリ秒なので、コア０の「実行時間」は、６０ミリ秒、コア１の「実行時間」は２０ミリ秒、オフセットは０ミリ秒となる。
【００４１】
タスク構成候補ａ２では、コア０において、Ｔａｓｋ０[０]、Ｔａｓｋ０[１]の順でサブタスクが実行され、コア０でＴａｓｋ０[０]が開始されたときから３０ミリ秒後にＴａｓｋ０[２]が開始される。この場合、コア０の最大実行時間は、５５ミリ秒、コア１の「実行時間」は２５ミリ秒、オフセットは３０ミリ秒となる。また、タスク構成候補ａ３では、コア０の「実行時間」は、４５ミリ秒、コア１の「実行時間」は３５ミリ秒、オフセットは０ミリ秒となる。

【００４２】
スケジューリング実行部１５は、タスク構成候補ごとに、サブタスクと、シングル実行タスクとを、自動最適化システム上などで仮想的に実行し、それぞれのスケジューリング結果をスケジュール候補とする。スケジューリング実行部１５は、実行周期の異なるタスクについては、それぞれの実行周期の最小公倍数の時間だけスケジュールを実行する。最小公倍数の時間より長い時間のスケジューリングを行っても、一般に最小公倍数以内の期間におけるスケジュールが繰り返されることになり、無駄が多い。また、最小公倍数の時間の時間だけ仮想的にスケジューリングを行うことにより、スケジュール決定装置１は、発生しうるスケジューリングの影響全てを網羅することができる。
【００４３】
例えば、本実施形態では、Ｔａｓｋ０の実行周期が８０ミリ秒、Ｔａｓｋ１の実行周期が２００ミリ秒であるから、少なくとも、それらの最小公倍数である４００ミリ秒間のスケジュールを作成する。
【００４４】
図８（ａ）～（ｊ）は、仮想的に実行されたスケジューリング結果を示す図である。これらの図において、横軸方向のバーの長さがタスク（サブタスク）の実行時間を表わしている。
【００４５】
図８（ａ）は、タスク構成候補ａ１における、サブタスクの実行スケジュールを示す図である。コア０の実行時間は６０ミリ秒（斜線部）、コア１の実行時間は２０ミリ秒（斜線部）、オフセットは０ミリ秒であり、このスケジュールが８０ミリ秒ごとに実行される。
【００４６】
図８（ｂ）は、タスク構成候補ａ１実行時の、シングル実行タスク（Ｔａｓｋ１）の実行スケジュールを示す図である。同図（ｂ）に示すように、サブタスクの合間のコア１が空いている期間に、そのコアでシングル実行タスクが実行される。
【００４７】
図８（ｃ）は、タスク構成候補ａ１についての、サブタスクおよびシングル実行タスクのスケジュール結果を示す図である。
【００４８】
図８（ｄ）は、タスク構成候補ａ２における、サブタスクの実行スケジュールを示す図である。同図（ｅ）は、タスク構成候補ａ２実行時の、シングル実行タスク（Ｔａｓｋ１）の実行スケジュールを示す図である。同図（ｆ）は、タスク構成候補ａ２についての、サブタスクおよびシングル実行タスクのスケジュール結果を示す図である。
【００４９】
図８（ｇ）は、タスク構成候補ａ３における、サブタスクの実行スケジュールを示す図である。同図（ｈ）は、タスク構成候補ａ３実行時の、シングル実行タスク（Ｔａｓｋ１）の実行スケジュールを示す図である。同図（ｉ）は、タスク構成候補ａ３についての、サブタスクおよびシングル実行タスクのスケジュール結果を示す図である。
【００５０】
有効度スコア算出部１７は、スケジューリング候補ごとに有効度スコアを算出する。有効度スコアは、開発対象のシステム要件に対して、そのスケジュール（スケジューリング候補）が適合している度合を示す値であり、例えば、下記（１）式により算出される。
【００５１】
（タスク構成候補の有効度スコア）＝ｋ１×（Ｔａｓｋ０の有効度スコア）＋ｋ２×（Ｔａｓｋ１の有効度スコア）・・・（１）
上記（１）において、ｋ１、ｋ２は、重みづけのための係数である。この（１）式における各タスクの有効度スコアは、下記（２）式により算出される。
【００５２】
（各タスクの有効度スコア）＝ｋ３×（余裕度平均）＋ｋ４×（中断時間平均）／（実

行周期）＋ｋ５×（ジッタ平均）／（実行周期）・・・（２）
上記（２）式において、ｋ３、ｋ４、およびｋ５は、重みづけのための係数である。「余裕度平均」は、例えば、下記（３）式により各実行周期について算出された余裕度の平均値である。「中断時間平均」は、各実行周期においてタスクが中断した時間の平均値である。「ジッタ平均」は、各実行周期において、周期の開始時間から、タスクが開始されるまでの遅延時間の平均値である。

【００５３】
（余裕度）＝{（実行周期）－（タスク完了時間）}／（実行周期）・・・（３）
上記（３）式において、「タスク完了時間」は、周期の開始時期からタスクの全ての処理が完了する時期までの時間である。並列実行タスク（Ｔａｓｋ０）については、全てのサブタスクが完了する時期までの時間である。

【００５４】
図９に各タスク構成候補について算出された、余裕度平均、中断時間平均、ジッタ平均、および有効度スコアを示す。有効度スコアは、全ての重みづけ係数（ｋ１～ｋ５）を１としたときの値である。同図を参照すると、タスク構成候補ａ１、ａ２、およびａ３の有効度スコアとして「０．７５」、「０．９２」、および「１．０５」が算出されている。

【００５５】
図１０は、候補決定部１９の構成を示すブロック図である。同図を参照すると、候補決定部１９は、有効度スコア比較部１９１および保持部１９３を有する。有効度スコア比較部１９１は、各タスク構成候補の有効度スコアを比較する。保持部１９３は９、最も高い有効度スコアのタスク構成候補を保持する。スケジュール決定装置１は、保持部１９３に保持されたタスク構成候補を最適の候補として出力する。

【００５６】
図１１～図１５を参照して、スケジュール決定装置の動作について説明する。図１１は、スケジュール決定装置１の動作を示すフローチャートである。同図を参照すると、依存関係取得部１１が、依存関係取得処理を実行し（ステップＳ１）、候補作成部１３が、候補作成処理を実行する（ステップＳ３）。そして、スケジューリング実行部１５が、代表化されたタスク構成候補ごとに、スケジューリングを実行する（ステップＳ４）。有効度スコア算出部１７は、スケジュール候補ごとに、有効度スコアを算出する（ステップＳ５）。候補決定部１９は候補決定処理を実行する（ステップＳ９）。ステップＳ９の後、スケジュール決定装置１は動作を終了する。

【００５７】
図１２は、依存関係取得処理を示すフローチャートである。同図を参照すると、依存関係取得部１１は、並列実行タスクを複数のサブタスクに分割し（ステップＳ１１）、依存グラフを作成することにより、サブタスク間の依存関係を求める（ステップＳ１３）。ステップＳ１３の後、依存関係取得部１１は、依存関係取得処理を終了する。

【００５８】
図１３は、候補作成処理を示すフローチャートである。同図を参照すると、候補作成部１３は、依存関係を満たすように、タスク構成候補を作成し（ステップＳ３１）、作成したタスク構成候補について代表化を行う（ステップＳ３５）。ステップＳ３５の後、候補作成部１３は、候補作成処理を終了する。

【００５９】
図１４は、候補決定処理を示すフローチャートである。同図を参照すると、候補決定部１９は、いずれかのタスク構成候補を選択し、その候補が１回目に選択した候補であるか否かを判断する（ステップＳ９１）。

【００６０】
１回目に選択した候補でなければ（ステップＳ９１：ＮＯ）、候補決定部１９は、選択した候補について、これまでの候補より高い有効度スコアが算出された候補であるか否かを判断する（ステップＳ９３）。

【００６１】

これまでの候補より高い有効度スコアが算出された候補である場合（ステップＳ９３：ＹＥＳ）、または１回目に選択した候補である場合（ステップＳ９１：ＹＥＳ）、候補決定部１９は、そのタスク構成候補のスケジュール構成と、算出された有効度スコアとを保持する（ステップＳ９５）。

【００６２】

これまでの候補より高い有効度スコアが算出された候補でない場合（ステップＳ９３：ＮＯ）、またはステップＳ９５の後、候補決定部１９は、全ての候補を選択して、有効度スコアを比較したか否かを判断する（ステップＳ９７）。全ての候補について比較していなければ（ステップＳ９７：ＮＯ）、候補決定部１９は、ステップＳ９１に戻る。全ての候補について比較したならば（ステップＳ９７：ＹＥＳ）、候補決定部１９は、候補決定処理を終了する。

【００６３】

なお、本実施形態では、スケジュール決定装置１は、１つの並列実行タスク（Ｔａｓｋ０）と、１つのシングル実行タスク（Ｔａｓｋ１）とを、スケジューリングの対象としているが、シングル実行タスクを含まない複数の並列実行タスクをスケジューリングの対象としてもよい。また、スケジュール決定装置１は、１つの並列実行タスクと複数のシングル実行タスクとをスケジューリングの対象とすることもできるし、複数の並列実行タスクと、１以上のシングル実行タスクとをスケジューリングの対象とすることもできる。

【００６４】

本実施形態では、スケジューリング決定装置１は、実行周期の異なるタスクをスケジューリングの対象としているが、実行周期が同一の複数のタスクについてスケジューリングを行ってもよい。

【００６５】

ここで、複数の並列実行タスクについて、スケジューリング候補を求める場合、あるタスクのサブタスクについての形状が同じであっても、他のタスクのサブタスクについての形状が異なるときは、別の候補とする。例えば、あるタスクのサブタスクの形状がＡ、Ｂで、別のタスクのサブタスクの形状がａ、ｂである場合、スケジュール決定装置１は、Ａおよびａの組み合わせと、Ａおよびｂの組み合わせとは別の候補として扱う。

【００６６】

また、本実施形態では、スケジュール決定装置１は、２つのコアで、タスクを並列実行するスケジュールを作成しているが、３つ以上のコアで並列実行するスケジュールを作成してもよいのは勿論である。

【００６７】

本実施形態では、並列実行タスクの分割をスケジュール決定装置自身が行う構成としているが、タスク分割は他の装置が行い、スケジュール決定装置１は、既に分割されたサブタスクをその装置から取得する構成としてもよい。

【００６８】

本実施形態では、スケジュール決定装置１は、図７に示したように、スケジューリングの結果を示すテーブルを作成しているが、タスク構成候補ごとの、各コアでのサブタスクの実行期間を示す情報が含まれるのであれば、テーブル以外の形式で、スケジューリングの結果を出力してもよい。

【００６９】

本実施形態では、テーブルにおいて、各コアの実行時間およびオフセットを記載しているが、それぞれのコアでのサブタスクの実行開始時間（代表値）と、実行終了時間（代表値）とを記載してもよい。

【００７０】

本実施形態では、スケジュール決定装置１は、上記（１）式、（２）式、および（３）式を使用して、有効度スコアを算出しているが、余裕度、中断時間、およびジッタのうち、少なくとも１以上の値に基づいて有効度スコアを算出できるのであれば、他の式を使用してもよい。

【００７１】
本実施形態では、スケジュール決定装置１は、有効度スコアが最大の候補を最適の候補に決定しているが、有効度スコアが所定の閾値以上を最適の候補とするなど、他の方法で、最適の候補を決定してもよい。
【００７２】
以上説明したように、本実施形態によれば、スケジュール決定装置１は、並列実行タスク（第１のタスク）から分割されたサブタスクを、依存関係を満たすように、複数のコア（処理装置）に配置して複数のタスク構成候補（サブタスク構成候補）を作成し、候補ごとに、シングル実行タスク（第２のタスク）を配置した場合におけるスケジューリング候補を作成し、各スケジューリング候補の有効度に基づいてスケジュールを決定する。サブタスクを配置した後に、その合間に第２のタスクを配置するので、第２のタスクにより並列に実行する第１のタスクが待たされて、スケジュール全体が実行不能となることが少なくなる。また、スケジュール決定装置は、スケジューリング候補ごとに有効度を算出するので、有効度に基づいて最適のスケジューリング候補を求めることにより、スケジュールを最適化できる。
【００７３】
候補作成部１３は、コア（処理装置）ごとに、それぞれのサブタスクを実行する期間を定めたテーブルを前記候補として作成するので、作成したスケジュールを出力するとき、スケジュール構成を２次元形状にするなど、可視化しやすい。
【００７４】
スケジューリング実行部１５は、各タスクの実行周期（第１の周期および第２の周期）の最小公倍数の期間だけスケジューリングするので、発生しうる、そのスケジューリングの影響全てを網羅することができ、スケジューリング処理に無駄がなくなる。
【００７５】
スケジュール決定装置１は、余裕度に基づいて有効度スコアを算出するので、デッドラインまでに余裕のあるスケジュールを作成できる。
【００７６】
スケジュール決定装置１は、タスクの中断時間に基づいて有効度スコアを算出するので、タスクがなるべく中断しないようにスケジュールを作成できる。
【００７７】
スケジュール決定装置１は、ジッタ（遅延時間）に基づいて有効度スコアを算出するので、遅延が少ないスケジュールを作成できる。
【００７８】
スケジュール決定装置１は、有効度スコアが最大の候補を最適の候補とするので、簡易な方法で１つの候補に決定できる。
【００７９】
（第２の実施形態）
本実施形態の第２の実施形態について、図１５～図２０を参照して説明する。本実施形態のスケジュール決定装置は、候補作成処理をより効率的に行う以外は、第１の実施形態のスケジュール決定装置の構成と同様である。
【００８０】
図１５は、本実施形態の候補作成処理を示すフローチャートである。同図を参照すると、候補作成部１３は、タスク構成候補を作成した後（ステップＳ３１）、シングルコアで実行するときの最大実行時間以上の最大実行時間となる候補を除外（破棄）する（ステップＳ３２）。本実施形態では、タスクの最大実行時間をシングルコアで実行する場合よりも短くする目的で、並列化を行うためである。
【００８１】
そして、候補作成部１３は、並列実行タスクが中断する候補を破棄する（ステップＳ３３）。本実施形態では、サブタスク間の依存関係の制約で、並列実行タスク（Ｔａｓｋ０）中断中に、他のタスク（Ｔａｓｋ１）を実行することが困難であるためである。

【００８２】
候補作成部１３は、コアを入れ替えると同じ構成となる候補も破棄する（ステップＳ３４）。本実施形態では、コア０とコア１が同じ種類の処理装置であるからである。ステップＳ３４の後は、スケジュール決定装置１はステップＳ３５を実行する。
【００８３】
次に、図１６～図２０を参照して、本実施形態のスケジュール決定装置１の動作例について説明する。図１６（ａ）に示すように、最大実行時間が１０１マイクロ秒のＴａｓｋ２をスケジューリングの対象とする場合について考える。
【００８４】
図１６（ｂ）に示すように、スケジュール決定装置１は、このＴａｓｋ２を、２０マイクロ秒、４１マイクロ秒、２２マイクロ秒、７マイクロ秒、および１１マイクロ秒の複数のサブタスク（Ｔａｓｋ２[０]、Ｔａｓｋ２[１]、Ｔａｓｋ２[３]、およびＴａｓｋ２[４]）に分割する。
【００８５】
スケジュール決定装置１は、図１６（ｃ）に示すように、サブタスク間の依存関係を求める。本実施形態では、Ｔａｓｋ２[０]の次に、Ｔａｓｋ２[１]、Ｔａｓｋ２[２]、およびＴａｓｋ２[３]を実行し、Ｔａｓｋ２[３]の次に、Ｔａｓｋ２[４]を実行する必要がある。
【００８６】
スケジュール決定装置１は、図１７（ａ）～（ｈ）および図１８（ａ）～（ｈ）に示すように、依存関係を満たしたタスク構成候補１１～２６を作成する。そして、スケジュール決定装置１は、図１９に示すように、スケジューリングの結果を示すテーブルを作成する。
【００８７】
図１７（ｅ）に示す候補１５では、コア０でＴａｓｋ２[０]、Ｔａｓｋ２[１]、Ｔａｓｋ２[２]、Ｔａｓｋ２[３]の順でサブタスクが実行され、Ｔａｓｋ２[３]が完了したとき、コア１でＴａｓｋ２[４]が開始される。この結果、並列実行した場合のＴａｓｋ２の最大実行時間（１０１マイクロ秒）が、シングルコアで実行した場合の最大実行時間（１０１マイクロ秒）と変わらないので、スケジュール決定装置１は、この候補１５を破棄する（ステップＳ３２）。同様の理由で図１８（ｈ）に示す候補２６も破棄される。
【００８８】
図１７（ｇ）に示す候補１７では、コア０でＴａｓｋ２[０]が完了したとき、コア０でＴａｓｋ２[２]が、コア１でＴａｓｋ２[１]およびＴａｓｋ２[３]が開始される。そして、Ｔａｓｋ２[３]が完了したとき、コア０でＴａｓｋ２[４]が開始される。Ｔａｓｋ２[２]の最大実行時間（２２マイクロ秒）は、Ｔａｓｋ２[１]の最大実行時間（４１マイクロ秒）より短いので、コア０において、Ｔａｓｋ２が中断する。従って、スケジュール決定装置１は、この候補１７を破棄する（ステップＳ３３）。同様の理由で図１８（ｂ）および（ｄ）に示す候補２０および２２も破棄される。
【００８９】
図２０（ａ）および（ｂ）は、コアを入れ替えると同じ構成になる候補の一例を示す図である。同図（ａ）に示す候補３１では、コア０で、Ｔａｓｋ3［０］およびＴａｓｋ4［０］が、コア１でＴａｓｋ3［１］およびＴａｓｋ4［１］が、コア２でＴａｓｋ4［２］が実行される。同図（ｂ）に示す候補３２では、コア０で、Ｔａｓｋ3［１］およびＴａｓｋ4［１］が、コア１でＴａｓｋ3［０］およびＴａｓｋ4［０］が、コア２でＴａｓｋ4［２］が実行される。そして、これらのコア０、コア１での処理を入れ替えると、同じ構成となるので、スケジュール決定装置１は、候補３１および候補３２のうち、いずれかを破棄する（ステップＳ３４）。
【００９０】
以上説明したように、本実施形態によれば、スケジュール決定装置１は、シングルコアで実行する場合の最大実行時間以上の候補を除外するので、並列化により実行時間が短く

なるスケジュールのみを作成できる。
【００９１】
また、スケジュール決定装置１は、タスクが中断する候補を除外するので、並列実行タスク中断中に他のタスクが実行されることを防止できる。
【００９２】
スケジュール決定装置１は、コアを入れ替えると同じ形状になる候補を除外するので、同じ種類のコアを複数有するシステムについてのスケジュールを作成する場合、無駄な処理を省くことができる。
【００９３】
（第３の実施形態）
本発明の第３の実施形態について、図２１～図３３を参照して説明する。図２１は、本実施形態の候補作成部１３の構成を示すブロック図である。同図を参照すると、本実施形態のスケジュール決定装置１は、候補操作部１３において、更に重ね合わせ代表化部１３５を有する以外は、第１の実施形態のスケジュール決定装置１と同様の構成である。
【００９４】
タスク構成候補作成部１３１は、依存関係を満たすように、タスク構成候補を作成し、タスクが途中で中断する候補は除外する。代表化部１３３は、各タスク構成候補について代表化を行う。重ね合わせの方法の詳細については、後述する。
【００９５】
重ね合わせ代表化部１３５は、タスク構成候補を所定の基準で複数のグループに分類し、各グループに属する候補を重ね合わせる。そして、重ね合わせ代表化部１３５は、重ね合わせた候補について代表化を行う。
【００９６】
グループの分類は、例えば、スケジュール決定装置１は、あるコアの最大実行時間に応じた順番で、タスク構成候補を整列し、各グループに属する候補の数がほぼ等しくなるように分類する方法を使用する。
【００９７】
又は、スケジュール決定装置１は、あるコアの最大実行時間に応じた順番で、タスク構成候補を整列し、グループがカバーする候補の、そのコアについての最大実行時間がほぼ均等になるように、分類する方法を使用する。
【００９８】
或いは、スケジュール決定装置１は、あるコアでの最大実行時間に応じた順番で、タスク構成候補を整列し、隣接する候補間の最大実行時間の差分を算出する。そして、スケジュール決定装置１は、区切りが所定値以上大きくなる箇所で区切ることにより、複数のグループに分類する方法を使用する。
【００９９】
乃至は、スケジュール決定装置１は、グループ内のオフセットが所定範囲内になるように、候補を分類する方法を使用する。スケジュール決定装置１は、候補数、最大実行時間の範囲、差分、およびオフセットを使用する方法のうち、いずれか２つ以上の方法を組み合わせて、候補を分類することもできる。
【０１００】
スケジューリング実行部１５は、重ね合わせた候補について、スケジュールを実行し、有効度スコア算出部１７は、重ね合わせた各スケジュール候補について有効度スコアを算出する。そして、スケジューリング実行部１５は、有効度スコアの最も高い重ね合わせ候補に属するタスク構成候補それぞれについてスケジューリングを実行し、有効度スコア算出部１７は、各スケジューリング候補について有効度スコアを算出する。
【０１０１】
図２２を参照して、重ね合わせの方法について説明する。図２２（ａ）～（ｅ）は、重ね合わせ方法を説明するための図である。
【０１０２】

重ね合わせの方法について詳細に説明する。重ね合わせ代表化部１３５は、グループ内のタスク構成候補のうち、いずれか２つのタスク構成候補を選択する。そして、重ね合わせ代表化部１３５は、それらのタスク構成候補を比較し、一方のタスク構成候補の実行期間のうち、他方の候補の実行期間と一致しない期間を、その他方のタスク構成候補に追加する（重ね合わせる）。重ね合わせ代表化部１３５は、一致しない期間を追加したタスク構成候補と、比較していない他の全てのタスク構成候補とを比較し、一致しない期間があれば、更に追加していく。

【０１０３】

例えば、図２２（ａ）、（ｂ）、（ｄ）に示すサブタスク構成候補１４、１５、および２１を重ね合わせる場合について考える。重ね合わせ代表化部１３５は、まず、同図２２（ａ）に示すサブタスク構成候補１４と、同図（ａ）に示すサブタスク構成候補１５とを比較する。重ね合わせ代表化部１３５は、同図（ｃ）に示すように、サブタスク構成候補１５の実行期間のうち、候補１４と一致していない期間を、サブタスク構成候補１４に追加する（重ね合わせる）。同図（ｃ）において斜線部分は、追加した部分である。

【０１０４】

次いで、重ね合わせ代表化部１３５は、重ね合わせたサブタスク構成候補と、図２２（ｄ）に示すサブタスク構成候補２１とを比較する。重ね合わせ代表化部１３５は、同図（ｅ）に示すように、サブタスク構成１５の実行期間のうち、重ね合わせた候補と一致していない期間を、重ね合わせたサブタスク構成候補に追加する。同図（ｅ）において斜線部分は、追加した部分である。

【０１０５】

図２３は、本実施形態の候補作成処理を示すフローチャートである。同図を参照すると、タスク構成候補作成部１３１は、タスク構成候補を作成し（ステップＳ３１）、並列実行タスクが中断する候補を破棄する（ステップＳ３３）。代表化部１３３は、タスク構成候補について代表化を行う（ステップＳ３５）。重ね合わせ部１３５は、所定の基準でタスク構成候補を複数のグループに分類し、各グループに属する候補を重ね合わせる。そして重ね合わせた候補ついて代表化を行い、重ね合わせ候補とする（ステップＳ３７）。ステップＳ３５の後、候補作成部１３は、候補作成処理を終了する。

【０１０６】

図２４は、本実施形態のスケジュール決定装置の動作を示すフローチャートである。同図を参照すると、ステップＳ３の後、スケジューリング実行部１５は、重ね合わせた候補について、スケジュールを実行し（ステップ４ａ）、有効度スコア算出部１７は、重ね合わせた各スケジュール候補について有効度スコアを算出する（ステップＳ５ａ）。そして、スケジューリング実行部１５は、有効度スコアの最も高い重ね合わせ候補に属するタスク構成候補それぞれについてスケジューリングを実行し（ステップＳ６）、有効度スコア算出部１７は、各スケジューリング候補について有効度スコアを算出する（ステップＳ７）。

【０１０７】

図２５～図２７を参照して、本実施形態のスケジュール決定装置１の動作結果の第１の例について説明する。

【０１０８】

図２５は、グループに属する候補数が均等となるようにタスク構成候補を分類した結果の一例を示す図である。同図は、図１９に示したスケジュール結果から、タスクが途中で中断される候補のみを除外したものである。同図を参照すると、重ね合わせ代表化部１３５は、スケジューリングの結果をコア０での最大実行時間に応じた順番で整列している。そして、重ね合わせ代表化部１３５は、各グループに属する候補数が均等になるように、候補を分類する。例えば、各グループの候補数が３つになるように分類する。この結果、候補１４、１５、および２１は、同じグループ（Ａ）に属することになる。

【０１０９】

図２６は、グループに属する候補の実行時間の範囲が所定値内となるようにタスク構成

候補を分類した結果の一例を示す図である。同図を参照すると、重ね合わせ代表化部１３５は、コア０の最大実行時間が、１００マイクロ秒未満、７５マイクロ秒以上のグループ（Ａ）、７５マイクロ秒未満、６０マイクロ秒以上のグループ（Ｂ）、６０マイクロ秒未満、４０マイクロ秒以上のグループ（Ｃ）、４０マイクロ秒未満のグループ（Ｄ）に各候補を分類している。

【０１１０】

図２７は、整列後、隣接する候補の最大実行時間の差分が所定値以上となる箇所で区切った結果の一例を示す図である。同図を参照すると、重ね合わせ代表化部１３５は、隣接する候補の差分が１０マイクロ秒以上となる箇所で区切って、各候補を分類している。

【０１１１】

図２８～図３３を参照して、本実施形態のスケジュール決定装置１の動作結果の第２の例について説明する。

【０１１２】

図２８（ａ）に示すように、最大実行時間が１０１マイクロ秒のＴａｓｋ５をスケジューリングの対象とする場合について考える。同図（ｂ）に示すように、スケジュール決定装置１は、このＴａｓｋ５を、２０マイクロ秒、４１マイクロ秒、２２マイクロ秒、７マイクロ秒、および１１マイクロ秒の複数のサブタスク（Ｔａｓｋ５[０]、Ｔａｓｋ５[１]、Ｔａｓｋ２[３]、およびＴａｓｋ５[４]）に分割する。

【０１１３】

そして、スケジュール決定装置１は、図２８（ｃ）に示すように、サブタスク間の依存関係を求める。本実施形態では、Ｔａｓｋ５[０]の次に、Ｔａｓｋ２[１]およびＴａｓｋ２[２]を実行し、Ｔａｓｋ２[２]の次にＴａｓｋ２[３]およびＴａｓｋ２[４]を実行する必要がある。

【０１１４】

スケジュール決定装置１は、図２９（ａ）～（ｈ）および図３０（ａ）～（ｈ）に示すように、依存関係を満たしたタスク構成候補４１～５６を作成する。

【０１１５】

そして、スケジュール決定装置１は、図３１に示すように、スケジューリングの結果を示すテーブルを作成する。本実施形態では、途中でタスクが中断する候補（４８、５２、および５３）は除外される。

【０１１６】

図３２および図３３は、スケジュール決定装置1がオフセットおよび実行時間の範囲を使用して候補を分類した結果を示す図である。図３２を参照すると、スケジュール決定装置１は、オフセットが２０の候補（Ａ、Ｂ）と、オフセットが８３の候補（Ｃ）は、異なるグループに分類する。そして、スケジュール決定装置１は、図３３に示すように、オフセットが２０の候補を、コア０の実行時間が５０マイクロ秒以上のグループ（Ａ）と、５０マイクロ秒未満のグループ（Ｂ）とに分ける。

【０１１７】

または、スケジュール決定装置１は、図３３に示すように、コア０での実行時間が６０マイクロ秒以上のグループ（Ａ）と、６０マイクロ秒未満、４０マイクロ秒以上のグループ（Ｂ）と、４０マイクロ秒未満のグループ（Ｃ）とに分類する。

【０１１８】

なお、図１１～図１５、図２２、および図２３に示したフローチャートの全部または一部は、コンピュータプログラムの実行により実現することもできる。

【０１１９】

以上説明したように、本実施形態によれば、タスク構成候補をまとめた重ね合わせ候補を用いて大まかな絞り込みを行い、その中で最適の重ね合わせ候補に属するタスク構成候補ごとに有効度スコアを算出するので、少ない探索量でスケジュールの最適化ができ、高度な自動最適化システムを実現することが可能となる。

【０１２０】

スケジュール決定装置１は、候補数、実行時間、差分、およびオフセットのいずれか1以上を使用して候補を分類するので、スケジュール決定装置１は、簡易な方法で、最適化の効率を上げることができる。

【０１２１】

（第４の実施形態）

本発明の第４の実施形態について、図３４を参照して説明する。同図は、本実施形態の並列実行装置２の構成を示すブロック図である。同図を参照すると、並列実行装置２は、並列実行部２１を更に有する以外は、第１の実施形態のスケジュール決定装置１と同様の構成である。

【０１２２】

並列実行部２１は、候補決定部１９で決定されたスケジュール候補を使用して、実行対象のタスクを複数のコア（不図示）で並列実行する。

【０１２３】

なお、本実施形態では、依存関係取得部１１、候補作成部１３、スケジューリング実行部１５、有効度スコア算出部１７、および候補決定部１９の構成は、第１の実施形態の依存関係取得部１１等と同様の構成としているが、これらは第２または第３の実施形態の依存関係取得部１１等と同様の構成としてもよい。

【０１２４】

以上説明したように、本実施形態によれば、並列実行装置２は、最適のスケジュールでタスクを並列実行できるので、効率的にタスクを処理できる。

【０１２５】

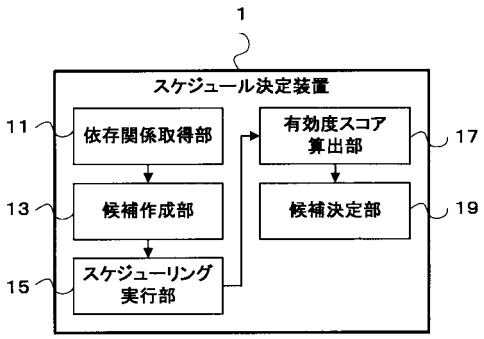
この出願は、２００８年１１月１４日に出願された日本出願特願２００８－２９２１１９を基礎として優先権の利益を主張するものであり、その開示の全てを引用によってここに取り込む。

【符号の説明】

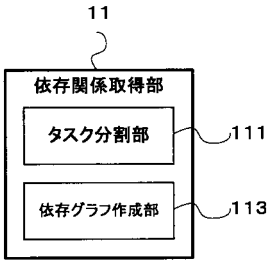
【０１２６】

１　スケジュール決定装置
２　並列実行装置
１１　依存関係取得部
１３　候補作成器
１５　スケジューリング実行部
１７　スコア算出部
１９　候補決定部
２１　並列実行部
１１１　タスク分割部
１１３　依存グラフ作成部
１３１　タスク構成候補作成部
１３３　代表化部
１３５　重ね合わせ部
１９１　有効度スコア比較部
１９３　保持部
ａ１、ａ２、ａ３　タスク構成候補
Ｓ１～Ｓ９、Ｓ１１、Ｓ１３、Ｓ３１～Ｓ３７、Ｓ９１～Ｓ９７、Ｓ４ａ、Ｓ５ａ、ステップ

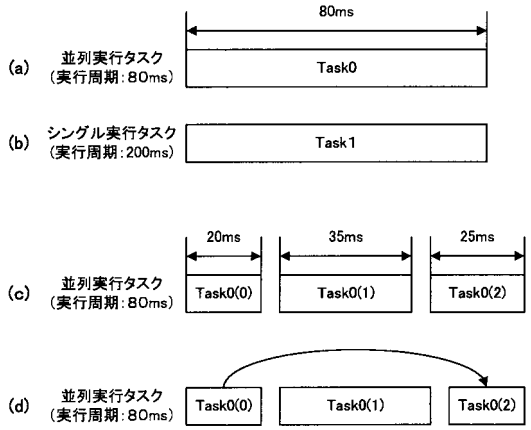
【図１】



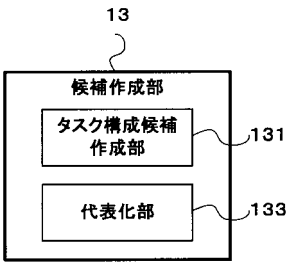
【図２】



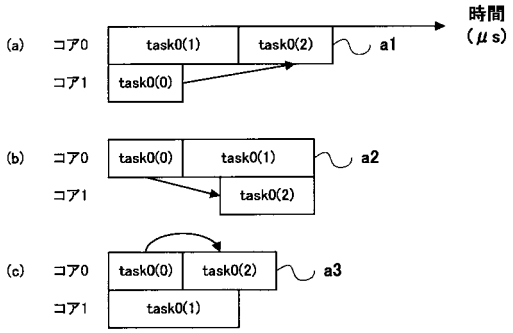
【図３】



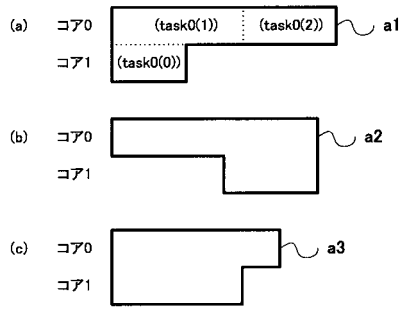
【図４】



【図５】



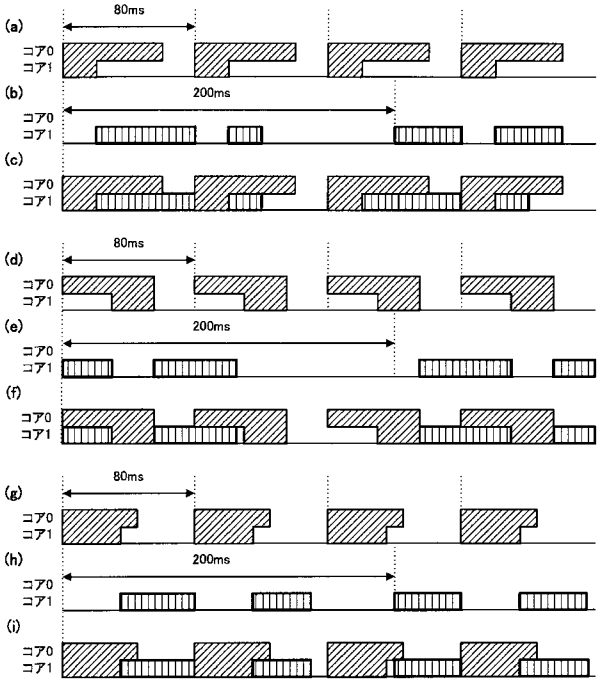
【図６】



【図７】

タスク構成候補			
候補番号	コア0の実行時間(ms)	コア1の実行時間(ms)	オフセット(μs)
a1	60	20	-
a2	55	25	30
a3	45	35	-

【図８】

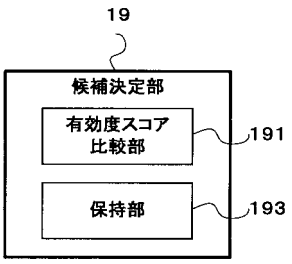


【図９】

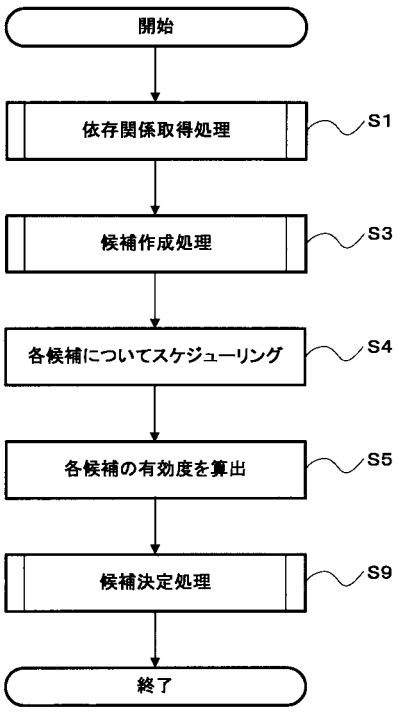
有効度スコアの算出例（算出期間:400ms）

候補番号	タスク種別	余裕度平均	中断時間平均(ms)	ジッタ平均(ms)	・・・	有効度スコア
a1	Task0	0.25	0	0	・・・	0.75
	Task1	0.35	20	10	・・・	
a2	Task0	0.32	0	0	・・・	0.92
	Task1	0.44	25	7.5	・・・	
a3	Task0	0.44	0	0	・・・	1.05
	Task1	0.34	35	17.5	・・・	
・	・	・	・	・	・	・

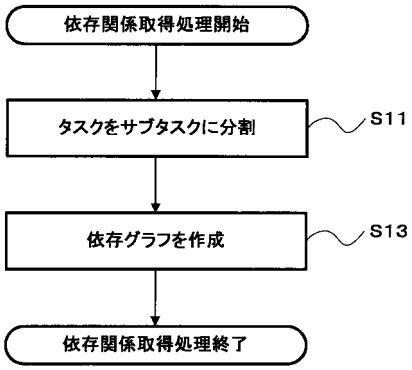
【図１０】



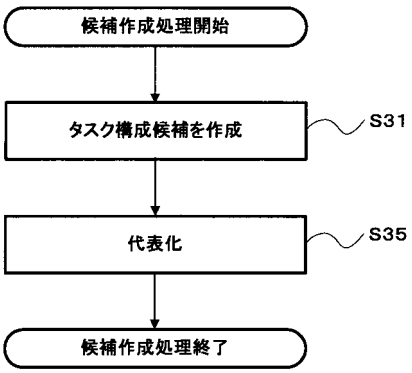
【図１１】



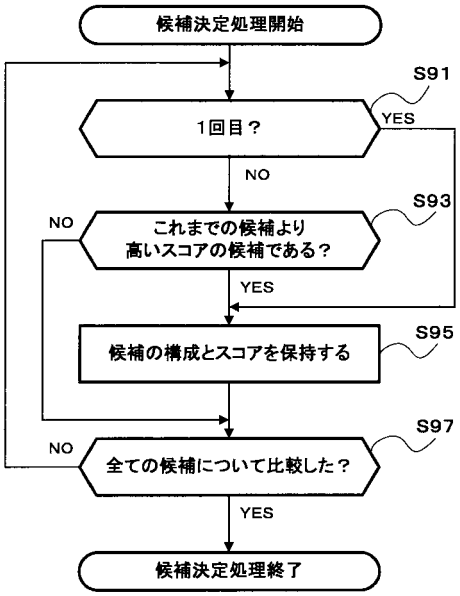
【図１２】



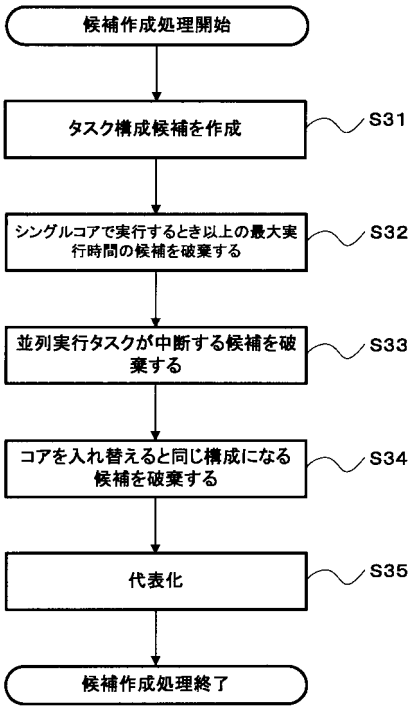
【図１３】



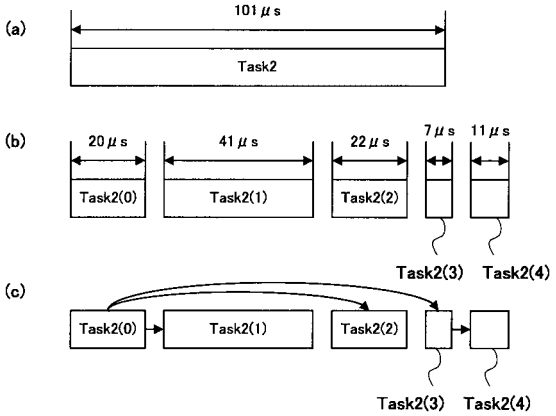
【図１４】



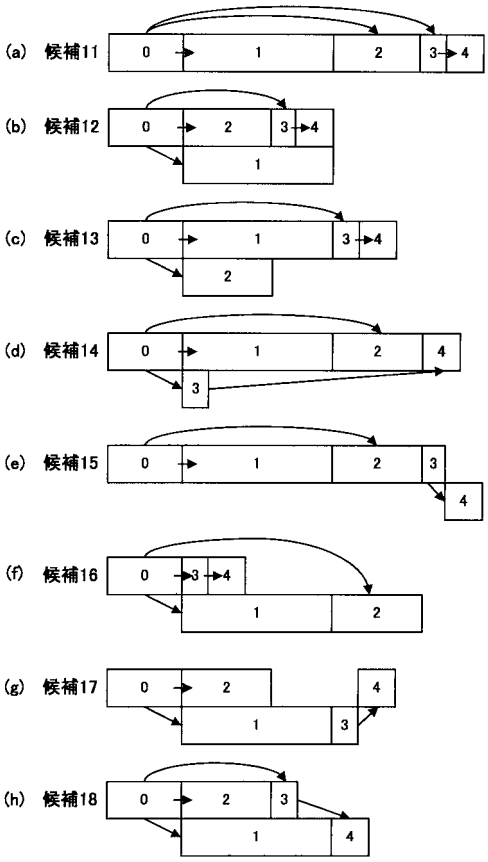
【図１５】



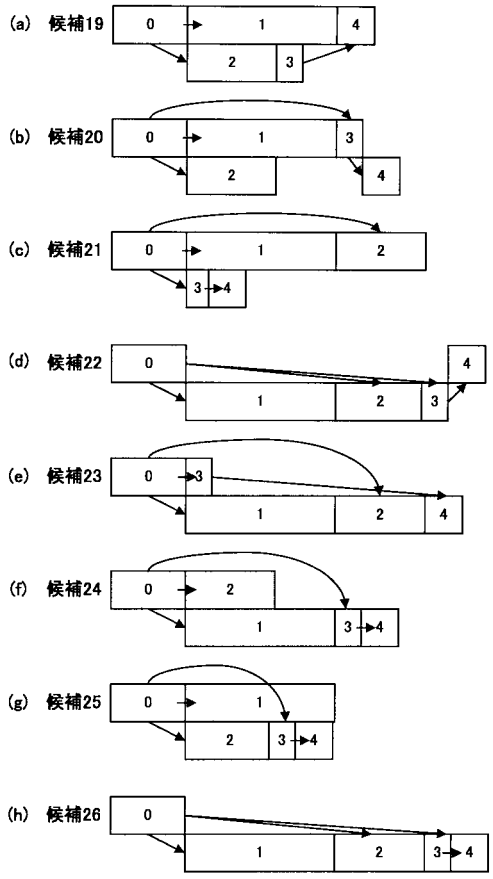
【図１６】



【図１７】



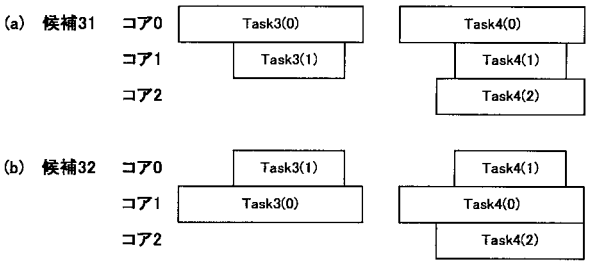
【図１８】



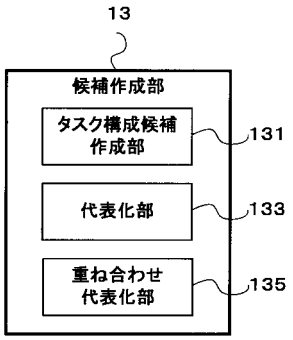
【図１９】

スケジューリングの結果			
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）
11	101	0	-
12	60	41	20
13	79	22	20
14	94	7	20
15	90	11	20
16	38	63	20
17	-	-	-
18	49	52	20
19	72	29	20
20	-	-	-
21	83	18	20
22	-	-	-
23	27	74	20
24	42	59	20
25	61	40	20
26	20	81	20

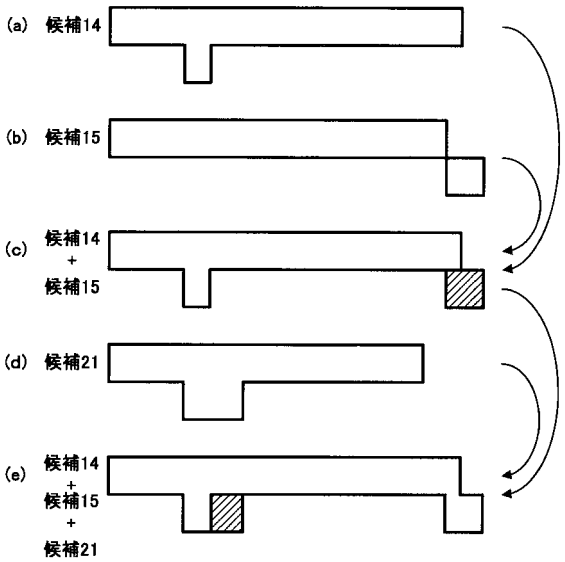
【図２０】



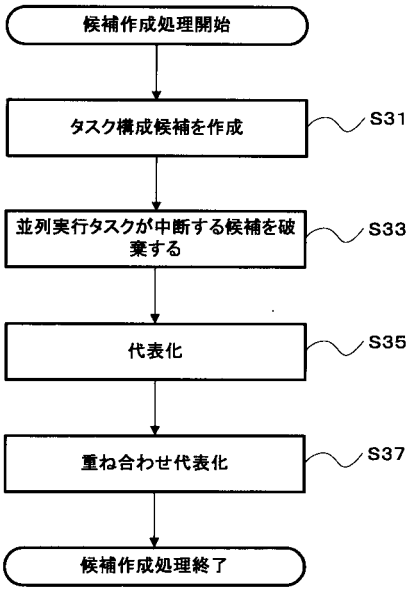
【図２１】



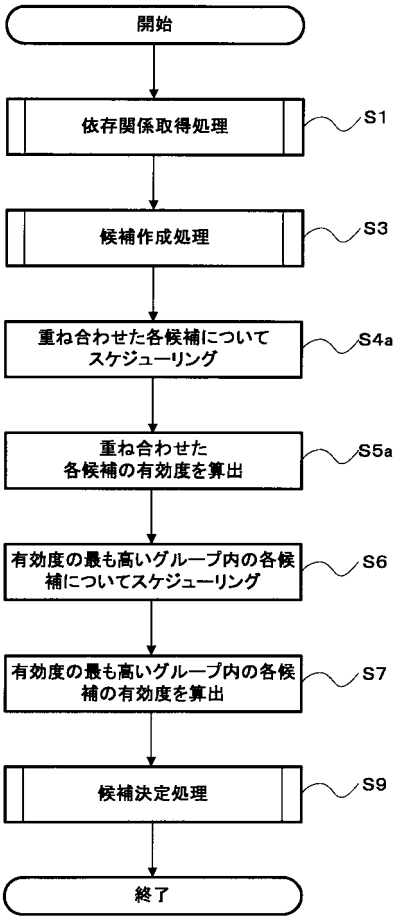
【図２２】



【図２３】



【図２４】



【図２５】

スケジューリングの結果（候補数が均等となるように分類）				
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）	グループ
11	101	0	-	-
14	94	7	20	A
15	90	11	20	A
21	83	18	20	A
13	79	22	20	B
19	72	29	20	B
25	61	40	20	B
12	60	41	20	C
18	49	52	20	C
24	42	59	20	C
16	38	63	20	D
23	27	74	20	D
26	20	81	20	D

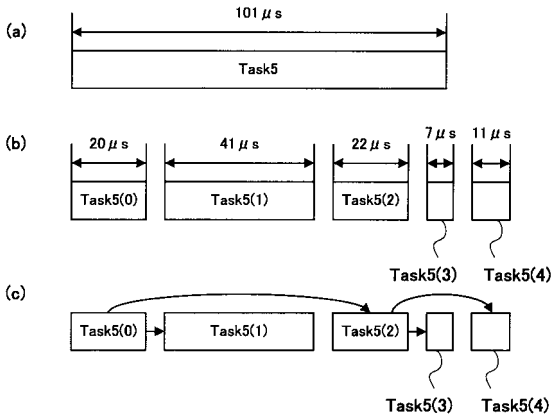
【図２６】

スケジューリングの結果（実行時間の範囲で分類）				
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）	グループ
11	101	0	-	-
14	94	7	20	A
15	90	11	20	A
21	83	18	20	A
13	79	22	20	A
19	72	29	20	B
25	61	40	20	B
12	60	41	20	B
18	49	52	20	C
24	42	59	20	C
16	38	63	20	D
23	27	74	20	D
26	20	81	20	D

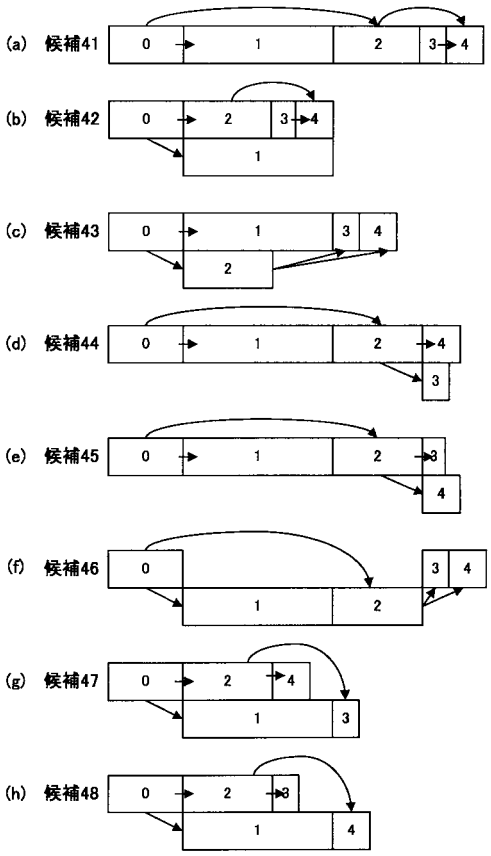
【図２７】

スケジューリングの結果（差分が大きい個所で区切る例）					
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）	差分（μs）	グループ
11	101	0	-	-	-
14	94	7	20	-	A
15	90	11	20	4	A
21	83	18	20	7	A
13	79	22	20	4	A
19	72	29	20	7	A
25	61	40	20	11	B
12	60	41	20	1	B
18	49	52	20	11	C
24	42	59	20	7	C
16	38	63	20	4	C
23	27	74	20	11	D
26	20	81	20	7	D

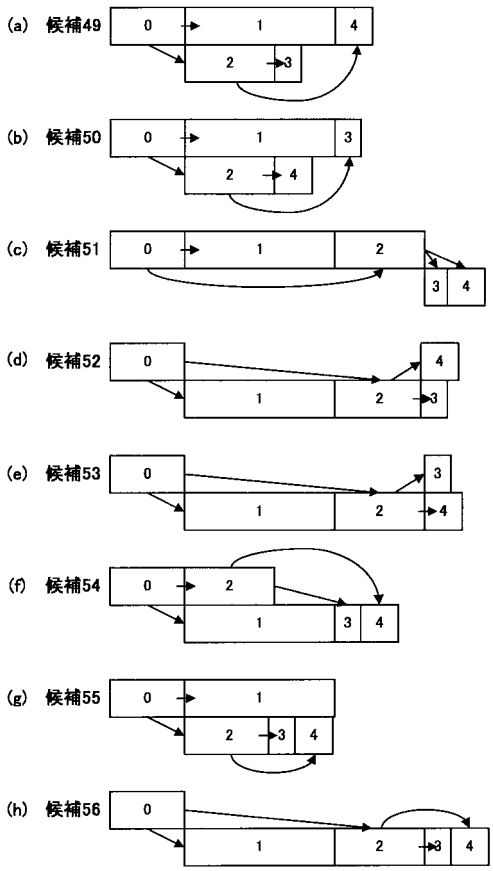
【図２８】



【図２９】



【図３０】



【図３１】

スケジューリングの結果			
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）
41	101	0	–
42	60	41	20
43	79	22	20
44	94	7	83
45	90	11	83
46	–	–	–
47	53	48	20
48	49	52	20
49	72	29	20
50	68	33	20
51	83	18	83
52	–	–	–
53	–	–	–
54	42	59	20
55	61	40	20
56	20	81	20

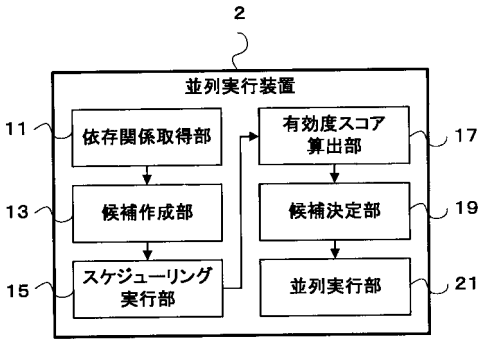
【図３２】

スケジューリングの結果（オフセット及び実行時間の範囲で分類）				
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）	グループ
41	101	0	–	–
43	79	22	20	A
49	72	29	20	A
50	68	33	20	A
55	61	40	20	A
42	60	41	20	A
47	53	48	20	A
48	49	52	20	B
54	42	59	20	B
56	20	81	20	B
44	94	7	83	C
45	90	11	83	C
51	83	18	83	C

【図３３】

スケジューリングの結果（オフセット及び実行時間の範囲で分類）				
候補番号	コア0の実行時間（μs）	コア1の実行時間（μs）	オフセット（μs）	グループ
41	101	0	–	–
43	79	22	20	A
49	72	29	20	A
50	68	33	20	A
55	61	40	20	A
42	60	41	20	A
47	53	48	20	B
48	49	52	20	B
54	42	59	20	B
56	20	81	20	C
44	94	7	83	D
45	90	11	83	D
51	83	18	83	D

【図３４】



フロントページの続き

審査官　田中　幸雄

(56)参考文献　特開平０９－２１８８６１（ＪＰ，Ａ）
特開平０９－０５４６９９（ＪＰ，Ａ）
特開２００１－１９５２６５（ＪＰ，Ａ）
長田徹也ほか，複製タスクを許したマルチプロセッサスケジューリングのための遺伝的アルゴリズムの検討，電子情報通信学会技術研究報告，日本，社団法人電子情報通信学会，１９９６年１１月，Ｖｏｌ．９６　Ｎｏ．３４８，１７－２４頁

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／５０
Ｇ０６Ｆ　　９／４８