(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3730740号
(P3730740)
(45)発行日　平成18年1月5日(2006.1.5)　(24)登録日　平成17年10月14日(2005.10.14)

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

請求項の数 10　(全 42 頁)

(21)出願番号　特願平9-38903
(22)出願日　平成9年2月24日(1997.2.24)
(65)公開番号　特開平10-240549
(43)公開日　平成10年9月11日(1998.9.11)
審査請求日　平成13年1月26日(2001.1.26)

(73)特許権者　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　100077816
弁理士　春日　譲
(72)発明者　中谷　明弘
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社　日立製作所　システム開発研究所内
(72)発明者　西門　隆
神奈川県川崎市麻生区王禅寺１０９９番地
株式会社　日立製作所　システム開発研究所内

最終頁に続く

(54)【発明の名称】並列ジョブ多重スケジューリング方法

(57)【特許請求の範囲】

【請求項１】

同等の機能を有する複数の演算プロセッサを有する計算機において、逐次処理部分と並列処理部分からなる並列ジョブを、複数、多重実行すべく、スケジューリングを行う、並列ジョブ多重スケジューリング方法において、

上記複数の演算プロセッサを、一の並列ジョブの逐次処理部分あるいは並列処理部分を実行する逐次演算プロセッサと、当該並列ジョブの並列処理部分を並列的に実行する複数個の演算プロセッサからなる並列演算プロセッサ群とに、論理的に分割する第１のステップと、

一の上記逐次演算プロセッサが、一の上記並列ジョブの逐次処理部分を実行する第２のステップと、

別の上記逐次演算プロセッサが、別の上記並列ジョブの逐次処理部分を実行する第３のステップと、

上記一の逐次演算プロセッサが、自己を含む上記並列演算プロセッサ群に対し、上記別の逐次演算プロセッサを含む他の逐次演算プロセッサを排して、同期処理を行う第４のステップと、

上記並列ジョブの並列処理部分が実行される第５のステップと、

上記並列演算プロセッサ群が終了通知を発行し、上記一の逐次演算プロセッサが、上記一の並列ジョブの逐次処理部分を実行する第６のステップと、

を有することを特徴とする並列ジョブ多重スケジューリング方法。

【請求項２】
請求項１記載の並列ジョブ多重スケジューリング方法において、
上記第５のステップの並列処理部分の実行中に、上記別の逐次演算プロセッサが、上記並列処理部分を実行する上記並列演算プロセッサ群に対し、同期処理を要求する第７のステップと、
上記逐次演算プロセッサが、複数の並列ジョブを多重に実行するためのスケジューラを起動する第８のステップと、
上記同期処理の要求による割り込みを契機に、上記別の逐次演算プロセッサが、上記並列ジョブの逐次処理部分を実行待ち状態とする第９のステップと、
上記スケジューラが、あらかじめ定められた規則に基づく次のジョブスケジューリングイベントにて、別の実行待ち状態の並列ジョブを取り出す第１０のステップと、
上記スケジューラが、上記一の逐次演算プロセッサが上記第５のステップの並列処理部分を実行する並列演算プロセッサ群を使用することを不可とする第１１のステップと、
上記スケジューラが、上記別の逐次演算プロセッサが上記第１０のステップで取り出した並列ジョブの並列処理部分を実行する並列演算プロセッサ群を使用することを許可する第１２のステップとを有し、
上記取り出した並列ジョブを実行することを特徴とする並列ジョブ多重スケジューリング方法。
【請求項３】
請求項２記載の並列ジョブ多重スケジューリング方法において、
上記ジョブスケジューリングイベントは、上記第５のステップの並列処理部分を実行する上記並列演算プロセッサ群が発行する終了通知であることを特徴とする並列ジョブ多重スケジューリング方法。
【請求項４】
請求項２記載の並列ジョブ多重スケジューリング方法において、
上記ジョブスケジューリングイベントは、１つの並列処理部分に引き続く逐次処理部分の処理時間が一定時間より長いと、コンパイラが設定できたとき、又はユーザがプログラム中で指定したとき、に発行される当該並列処理部分の終了通知であることを特徴とする並列ジョブ多重スケジューリング方法。
【請求項５】
請求項２記載の並列ジョブ多重スケジューリング方法において、
上記ジョブスケジューリングイベントは、同期処理の対象となっている上記逐次演算プロセッサの全てが、シテムコール処理時の待ち状態と設定されていた場合であることを特徴とする並列ジョブ多重スケジューリング方法。
【請求項６】
請求項２記載の並列ジョブ多重スケジューリング方法において、
上記ジョブスケジューリングイベントは、ある並列処理部分と引き続く並列処理部分の間の逐次処理部分の実行に要した計算時間を統計的に記録し、これから実行する逐次処理部分が統計的な時間より長い可能性が高いと判断されたとき行う実行中の並列処理部分の終了通知であることを特徴とする並列ジョブ多重スケジューリング方法。
【請求項７】
請求項２記載の並列ジョブ多重スケジューリング方法において、
上記並列演算プロセッサ群の各並列演算プロセッサが実行していた上記一の並列ジョブの使用していたレジスタのうち、最小限のレジスタの内容のみを退避し、
さらに、次に上記並列演算プロセッサ群を使用する上記別の並列ジョブに対応して退避されたレジスタの内容を回復し、当該別の並列ジョブの実行を再開することを特徴とする並列ジョブ多重スケジューリング方法。
【請求項８】
請求項１記載の並列ジョブ多重スケジューリング方法において、
上記複数の演算プロセッサの、上記逐次演算プロセッサと上記並列演算プロセッサ群へ

の割り当てのプロセッサ構成の比率を動的に変更することを特徴とする並列ジョブ多重スケジューリング方法。

【請求項９】

請求項８記載の並列ジョブ多重スケジューリング方法において、

並列ジョブは、当該ジョブが使用する上記複数の演算プロセッサの構成に従って、予めユーザによって所属する階級が指定され、上記第５のステップにおいて同一の階級の並列ジョブのみが実行されるよう、当該指定に従って、演算プロセッサを切り替えることで上記プロセッサ構成の比率を動的に変更することを特徴とする並列ジョブ多重スケジューリング方法。

【請求項１０】

請求項１記載の並列ジョブ多重スケジューリング方法において、

上記第４のステップの同期処理を行った結果、予め指定された最大待ち時間以内に、同期処理の対象となった全ての演算プロセッサが同期に到達しなかった場合、第三の並列ジョブのスケジューリングを開始することを特徴とする並列ジョブ多重スケジューリング方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、並列ジョブ多重スケジューリング方法に係り、特に、複数の並列化された情報処理プログラムを並行実行するのに好適な並列ジョブ多重スケジューリング方法に関する。

【０００２】

【従来の技術】

従来の並列化された情報処理プログラムを実行する方式としては、例えば、富田眞治著「並列コンピュータ工学」昭晃堂発行（１９９６），第９２頁から第１０４頁に記載されているように、複数の情報処理プログラムを並行して処理する場合には、計算機システム全体を時分割によって利用するようにしている。

【０００３】

【発明が解決しようとする課題】

しかしながら、従来方式では、並列化された複数のジョブを実行するに当たり、計算機システムに備わる並列演算プロセッサを時分割利用することによって並行処理を行うようにしているため、並列演算プロセッサが１並列ジョブに割り当てられた場合、その割り当て時間内に並列ジョブの逐次処理部分が実行されるとき、並列演算プロセッサは遊休状態となり、計算機システム全体の稼働率が低下するという問題がある。

【０００４】

なお、デュアルスカラプロセッサ方式と呼ばれる方式においては、２つの逐次演算機構を用意し、それらの間で単一の並列演算機構を共有するようにしているが、ここでも並列演算機構は時分割利用されるため、逐次演算部分の処理中に並列演算機構が遊休し、計算機システム全体の稼働率が低下するという問題がある。

【０００５】

本発明の目的は、複数の並列化されたジョブの同時実行時に、システム内の全プロセッサの稼働率の向上した並列ジョブ多重スケジューリング方法を提供することにある。

【０００６】

【課題を解決するための手段】

（１）上記目的を達成するために、本発明は、同等の機能を有する複数の演算プロセッサを有する計算機において、逐次処理部分と並列処理部分からなる並列ジョブを、複数、多重実行すべく、スケジューリングを行う、並列ジョブ多重スケジューリング方法において、上記複数の演算プロセッサを、一の並列ジョブの逐次処理部分あるいは並列処理部分を実行する逐次演算プロセッサと、当該並列ジョブの並列処理部分を並列的に実行する複数個の演算プロセッサからなる並列演算プロセッサ群とに、論理的に分割する第１のステッ

プと、一の上記逐次演算プロセッサが、一の上記並列ジョブの逐次処理部分を実行する第２のステップと、別の上記逐次演算プロセッサが、別の上記並列ジョブの逐次処理部分を実行する第３のステップと、上記一の逐次演算プロセッサが、自己を含む上記並列演算プロセッサ群に対し、上記別の逐次演算プロセッサを含む他の逐次演算プロセッサを排して、同期処理を行う第４のステップと、上記並列ジョブの並列処理部分が実行される第５のステップと、上記並列演算プロセッサ群が終了通知を発行し、上記一の逐次演算プロセッサが、上記一の並列ジョブの逐次処理部分を実行する第６のステップと、を有するものである。

かかる方法によって、複数のジョブを並列的に実行できるため、計算機システム内の全プロセッサの稼働率を向上し得るものとなる。

【０００７】

（２）上記（１）において、好ましくは、上記第５のステップの並列処理部分の実行中に、上記別の逐次演算プロセッサが、上記並列処理部分を実行する上記並列演算プロセッサ群に対し、同期処理を要求する第７のステップと、上記逐次演算プロセッサが、複数の並列ジョブを多重に実行するためのスケジューラを起動する第８のステップと、上記同期処理の要求による割り込みを契機に、上記別の逐次演算プロセッサが、上記並列ジョブの逐次処理部分を実行待ち状態とする第９のステップと、上記スケジューラが、あらかじめ定められた規則に基づく次のジョブスケジューリングイベントにて、別の実行待ち状態の並列ジョブを取り出す第１０のステップと、上記スケジューラが、上記一の逐次演算プロセッサが上記第５のステップの並列処理部分を実行する並列演算プロセッサ群を使用することを不可とする第１１のステップと、上記スケジューラが、上記別の逐次演算プロセッサが上記第１０のステップで取り出した並列ジョブの並列処理部分を実行する並列演算プロセッサ群を使用することを許可する第１２のステップとを有し、上記取り出した並列ジョブを実行するようにしたものである。

かかる方法によって、ジョブ切り替えのオーバヘッドを低減し得るものとなる。

【０００８】

（３）上記（２）において、好ましくは、上記ジョブスケジューリングイベントは、上記第５のステップの並列処理部分を実行する上記並列演算プロセッサ群が発行する終了通知としたものである。

かかる方法によって、オーバーヘッドの少ない切り替えを行い得るものとなる。

【０００９】

（４）上記（２）において、好ましくは、上記ジョブスケジューリングイベントは、１つの並列処理部分に引き続く逐次処理部分の処理時間が一定時間より長いと、コンパイラが設定できたとき、又はユーザがプログラム中で指定したとき、に発行される当該並列処理部分の終了通知したものである。

かかる方法によって、スケジューリングのオーバーヘッドを少なくして、計算機システム全体の効率を向上し得るものとなる。

【００１０】

（５）上記（２）において、好ましくは、上記ジョブスケジューリングイベントは、同期処理の対象となっている上記逐次演算プロセッサの全てが、システムコール処理時の待ち状態と設定されていた場合としたものである。

かかる方法によって、オーバーヘッドの少ない切り替えを行い得るものとなる。

【００１１】

（６）上記（２）において、好ましくは、上記ジョブスケジューリングイベントは、ある並列処理部分と引き続く並列処理部分の間の逐次処理部分の実行に要した計算時間を統計的に記録し、これから実行する逐次処理部分が統計的な時間より長い可能性が高いと判断されたとき行う実行中の並列処理部分の終了通知としたものである。

かかる方法によって、動的な終了通知の発行による並列演算プロセッサ群の割り当て変更を行い得るものとなる。

【００１２】

（７）上記（２）において、好ましくは、上記並列演算プロセッサ群の各並列演算プロセッサが実行していた上記一の並列ジョブの使用していたレジスタのうち、最小限のレジスタの内容のみを退避し、さらに、次に上記並列演算プロセッサ群を使用する上記別の並列ジョブに対応して退避されたレジスタの内容を回復し、当該別の並列ジョブの実行を再開するようにしたものである。

かかる方法によって、退避・回復するレジスタの数を大幅に削減することが可能になり、ジョブ切り替えのオーバヘッドを低減し得るものとなる。

【００１３】

（８）上記（１）において、好ましくは、上記複数の演算プロセッサの、上記逐次演算プロセッサと上記並列演算プロセッサ群への割り当てのプロセッサ構成の比率を動的に変更するようにしたものである。

かかる方法によって、計算機システムの稼動率を高める得るものとなる。

【００１４】

（９）上記（８）において、好ましくは、並列ジョブは、当該ジョブが使用する上記複数の演算プロセッサの構成に従って、予めユーザによって所属する階級が指定され、上記第５のステップにおいて同一の階級の並列ジョブのみが実行されるよう、当該指定に従って、演算プロセッサを切り替えることで上記プロセッサ構成の比率を動的に変更するようにしたものである。

かかる方法によって、それぞれの階級に割り付けられた時間内で階級に属するジョブをその階級の実行形式に従って処理し得るものとなる。

【００１５】

（１０）上記（１）において、好ましくは、上記第４のステップの同期処理を行った結果、予め指定された最大待ち時間以内に、同期処理の対象となった全ての演算プロセッサが同期に到達しなかった場合、第三の並列ジョブのスケジューリングを開始するようにしたものである。

かかる方法によって、その同期処理は失敗したものとして、同期失敗を知らせる割り込みが演算プロセッサ上に発生し、これを契機にスケジューラが起動され、エラー処理を行い得るものとなる。

【００２２】

【発明の実施の形態】

以下、図１～図３１を用いて、本発明の一実施形態による並列ジョブ多重スケジューリング方法及び装置について説明する。

最初に、図１を用いて、本発明の一実施形態による計算機システムの構成について説明する。

図１は、本発明の一実施形態による計算機システムの全体構成図である。

【００２３】

演算プロセッサ群１０００は、例えば、８台の演算プロセッサ１００１，…，１００８によって構成されている。ここでは、演算プロセッサ１００１～１００８の台数は、８台としているがこれに限られるものでない。演算プロセッサ１００１～１００８は、それぞれ、共有記憶装置２０００に接続され、データの授受を行う。共有記憶装置２０００は、演算プロセッサ間データ授受領域２１０１，…，２１０８を備えている。演算プロセッサ１００１～１００８間のデータをやり取りは、演算プロセッサ間データ授受領域２１０１，…，２１０８を介して行われる。例えば、送り元である演算プロセッサ１００１から受け手である演算プロセッサ１００２にデータを送るには、送り元の演算プロセッサ１００１が受け手の演算プロセッサのデータ授受領域２１０２にデータを書き込み、このデータを受け手の演算プロセッサ１００２が参照することで行われる。ユーザのプログラムコードは、外部記憶装置３０００に保存され、実行時には共有記憶装置上に配置されて実行される。

【００２４】

ここで、本実施形態においては、演算プロセッサ群１０００を構成する演算プロセッサ１

００１～１００８は、全て同等の演算機能を有するものである。並列ジョブの逐次処理部分及び並列処理部分を実行する演算プロセッサを、逐次演算プロセッサと称する。また、並列処理部分のみを実行する演算プロセッサを、並列演算プロセッサと称する。そして、演算プロセッサ群１０００は、複数の逐次演算プロセッサと、複数の並列演算プロセッサから構成される並列演算プロセッサ群に論理的に分割されている。

【００２５】

並列演算プロセッサ群を構成する複数の並列演算プロセッサは、ジョブの並列処理部分を１群として並列的に実行するため、これらの並列演算プロセッサの集合体を、並列演算プロセッサ群と称している。複数の並列演算プロセッサは、それぞれ、別々の並列ジョブに割り当てられる。一方、複数の逐次演算プロセッサは、それぞれ、独立してジョブの逐次処理部分を実行する。また、複数の逐次演算プロセッサの中の１つの逐次演算プロセッサは、並列演算プロセッサ群を構成する複数の並列演算プロセッサと並列的に、ジョブの並列処理部分の処理を実行する。

【００２６】

ここで、本実施形態においては、例えば、演算プロセッサ１００１，…，１００４が、複数の逐次演算プロセッサを構成し、演算プロセッサ１００５，…，１００８が、並列演算プロセッサ群を構成するように、論理分割されているものとする。なお、複数の逐次演算プロセッサと並列演算プロセッサ群の論理分割は自由であり、例えば、演算プロセッサ１００１，１００２が、複数の逐次演算プロセッサを構成し、演算プロセッサ１００３，…，１００８が、並列演算プロセッサ群を構成するように、論理分割してもよいものである。また、並列演算プロセッサ群は、１群に限らないものであり、例えば、演算プロセッサ１００３，…，１００５によって第１群の並列演算プロセッサ群を構成し、演算プロセッサ１００６，…，１００８によって第２群の並列演算プロセッサ群を構成してもよい。このように複数の群を備える場合には、並列演算プロセッサ群を順番に用いて並列処理部分を実行する。

【００２７】

演算プロセッサ間同期機構４０００は、演算プロセッサ１００１～１００８に接続され、演算プロセッサ１００１～１００８間の同期処理を行うものである。演算プロセッサ間同期機構４０００は、同期範囲指定機構４１００を備えている。同期範囲指定機構４１００は、並列演算プロセッサ群を、複数並列ジョブで共用使用するために、各逐次演算プロセッサに対応して上記並列演算プロセッサ群が使用可能かをプログラム制御可能な並列演算機組合せ機構とするものである。同期範囲指定機構４１００によって指定された計算機システム内の演算プロセッサ１０００間のみで同期処理を行うことが可能である。同期範囲指定機構４１００の詳細については、図２を用いて後述する。

【００２８】

ここで、並列処理部分を実行する機構が、従来の並列専用機構とは異なり、逐次演算プロセッサと同等の複数個の演算プロセッサによって論理的に構成することが重要である。これによって、計算機システム内の演算プロセッサの逐次及び並列演算プロセッサへの割り当て比率を変更することが可能になる。また、必要に応じて演算プロセッサを物理的に増加させることによって、演算性能を向上させることも可能である。

【００２９】

各演算プロセッサ１００１～１００８は、演算プロセッサ間同期機構４０００に対して、同期到達通知信号線ＳＲＬと同期完了通知信号線ＳＣＬによって接続される。同期処理に際しては、同期処理に達した演算プロセッサ１０００は、同期到達通知命令を実行することによって、同期到達通知信号線ＳＲＬに信号を出力し、同期完了待ち命令を実行して、同期完了通知信号線ＳＣＬから信号が来るのを待つ。演算プロセッサ間同期機構４０００は、同期範囲指定機構４１００によって同期の範囲内と指定された全ての演算プロセッサが同期到達信号線ＳＲＬに対して信号を出した時点で同期処理が完了したとして、同期範囲内の演算プロセッサの同期完了信号線ＳＣＬに対して信号を出して同期処理の完了を通知する。同期完了信号を受けた演算プロセッサは、同期完了待ち状態が解除され、同期処

理に引き続く処理の実行に移る。
【００３０】
演算プロセッサ間割り込み制御機構５０００は、演算プロセッサ間同期機構４０００に接続されており、同期範囲指定機構４１００によって同期範囲外と設定された演算プロセッサからの同期要求に対しては、この演算プロセッサに割り込みを発生させて、共有記憶装置２０００内のスケジューラを起動する。また、同期範囲指定機構４１００によって同期範囲内と設定された演算プロセッサが同期要求を行い、その同期の成立前に同期範囲外に設定され直した場合にも、この演算プロセッサに割り込みを発生させてスケジューラを起動する。
【００３１】
次に、図２を用いて、本発明の一実施形態による計算機システムに用いる同期範囲指定機構４１００の構成について説明する。
図２は、本発明の一実施形態による同期範囲指定機構の構成図である。
【００３２】
図１において説明したように、計算機システムが８台の演算プロセッサによって構成される場合においては、同期範囲指定機構４１００は、８ビットの同期範囲指定レジスタ４１１０を備えている。
【００３３】
一般的に、ｎ台の演算プロセッサから成る計算機システムでは、同期処理に参加する演算プロセッサは、同期範囲指定機構が備えるｎビットレジスタ（同期範囲指定レジスタ）によって指定される。このレジスタの第ｋビットが演算プロセッサｋに対応するものとし、このビットに１が設定されていれば、対応する演算プロセッサは同期処理に参加する。同期範囲内と設定された演算プロセッサからの同期要求に対してはそのまま同期処理を続行する。
【００３４】
一方、同期範囲外と設定された演算プロセッサからの同期要求に対してはこの演算プロセッサに割り込みを発生させてスケジューラを起動する。また、同期範囲内と設定された演算プロセッサが同期要求を行い、その同期の成立前に同期範囲外に設定され直した場合にも、同演算プロセッサに割り込みを発生させてスケジューラを起動する。
【００３５】
ここで、同期範囲外の演算プロセッサからの同期要求は無視されるのではなく、同期要求に対して、要求元の演算プロセッサ上に割り込みを発生させること、及び、同期成立前に同期範囲外に設定し直された演算プロセッサ上に割り込みを発生させることが重要である。
【００３６】
図２に示す例では、８ビットからなる同期範囲指定レジスタ４１１０のビット１，５，６，７及び８に”１”を設定し、８台の演算プロセッサの内、図１に示した演算プロセッサ１００１，１００５，１００６，１００７，１００８が同期処理に参加するように指示している。ここで、演算プロセッサ１００１は、逐次演算プロセッサであり、演算プロセッサ１００５，…，１００８は、並列演算プロセッサ群を構成する並列演算プロセッサである。演算プロセッサ１００５，…，１００８が、ジョブの並列処理部分を実行する際には、演算プロセッサ１００１も並列処理部分を実行する。それ以外のときには、演算プロセッサ１００１は、逐次処理部分を実行する。
【００３７】
次に、図３を用いて、共有記憶装置２０００中に用意されるデータについて説明する。
図３は、本発明の一実施形態による共有記憶装置中に用意されるデータの説明図である。
【００３８】
共有記憶装置２０００は、ジョブ実行コード２１００が各ジョブに対応して配置され、各演算プロセッサ１００１～１００８は自らに割り当てられたジョブ実行コード２１００を実行する。また、ＯＳ実行コード２２００は、ユーザジョブのコードとは別に配置される

。これらのジョブ及びハードウェア資源管理のための情報を保持する管理情報２３００が、共有記憶装置２０００上に確保されている。管理情報２３００の詳細については、図４を用いて後述する。

【００３９】

次に、図４を用いて、共有記憶装置２０００上に確保されている管理情報２３００の内容について説明する。

図４は、本発明の一実施形態における共有記憶装置上の管理情報の構成図である。

【００４０】

ジョブ管理情報先頭アドレス設定フィールド２３０１は、ジョブ管理情報が存在するメモリアドレスを保持する領域である。ユーザ実行ジョブは起動されると、ジョブ毎に、図５に示すジョブ管理情報２４００が共有記憶装置２０００上に確保され、ユーザ実行ジョブは、ジョブ管理情報２４００に基づいて管理される。

【００４１】

階級設定フィールド２３０２は、現在実行中の階級を設定するフィールドである。逐次並列間データ授受領域対応付けフィールド２３０３については、図１３を用いて後述する。構成変更情報設定フィールド２３０４は、ここに設定された情報に基づいて階級切り替え及びシステム内の演算プロセッサ構成を変更するのに用いられ、詳細については図２２を用いて後述する。同期範囲指定機構設定フィールド１３００５は、このフィールドの設定によって同期範囲指定レジスタ４１１０（図２）の内容を指定する。

【００４２】

同期タイムアウトレジスタ設定フィールド２３０６は、同期処理が失敗したと判断するまでの時間を設定する特別なレジスタ（同期タイムアウトレジスタ）の設定に用いられるものである。同期範囲指定フィールド２３０７及び同期到達フラグ２３０９は、後述するように、演算プロセッサ間の同期処理をソフトウェアで実現するときにのみ用いるものである。

【００４３】

単一実行フラグ２３０８については、図３０を用いて後述する。待ち状態フラグ２３１０は、演算プロセッサの数だけ用意されているものであり、詳細については後述する。

【００４４】

次に、図５を用いて、共有記憶装置２０００上に確保されているユーザ実行ジョブを管理するジョブ管理情報２４００の構成について説明する。

図５は、本発明の一実施形態において用いるジョブ管理情報の構成図である。

【００４５】

並列ジョブ識別番号フィールド２４０１は、並列ジョブ識別番号を保持するフィールドであり、並列ジョブ識別番号によって、ジョブは一意に指定することができる。所属階級識別フィールド２４０２は、ジョブが所属するジョブ階級を保持するフィールドであり、詳細については後述する。逐次実行時間フィールド２４０３は、ジョブの起動からの逐次実行時間を記録するフィールドである。

【００４６】

並列実行時間フィールド２４０４は、ジョブの起動からの並列実行時間を記録するフィールドである。先行逐次情報フィールド２４０５は、先行逐次情報を記録するフィールドであり、詳細については図２９を用いて後述する。ジョブ状態保存フィールド２４０６は、ジョブ状態を保存するフィールドであり、ジョブ自らの状態を記録する。

【００４７】

レジスタ未退避フラグ２４０７は、レジスタが未退避であることを示すフラグである。レジスタ退避範囲２４０８は、レジスタの退避範囲を示すフラグであり、その詳細については図１２を用いて後述する。レジスタ内容退避エリア２４０９は、ジョブ切り替え時にレジスタの内容を退避するエリアである。

【００４８】

次に、図６を用いて、共有記憶装置２０００上に確保されている階級管理情報２５００に

ついて説明する。
図６は、本発明の一実施形態において用いる階級管理情報の構成図である。
【００４９】
後述するように、ジョブは階級分けされ、これらの階級を切り替えて実行される。各階級は、図６に示すようなフィールドから成る階級管理情報２５００を共有記憶装置２０００上にもち、階級管理情報２５００に従って管理される。
【００５０】
階級毎に、階級管理情報２５００は、階級識別番号フィールド２５０１と、キュー先頭アドレス設定フィールド２５０２と、同期範囲指定レジスタ退避フィールド２５０３と、所属ジョブ識別番号フィールド２５０４とから構成され、共有記憶装置２０００上に保持される。
【００５１】
図１に示す例では、演算プロセッサ１０００を共有記憶装置２０００によって接続しているが、演算プロセッサ間のデータ授受方法は演算プロセッサ間に備えられた通信装置によるメッセージ送達によるものでもよいものである。
【００５２】
本実施形態において重要なことは、逐次演算プロセッサと同等の複数の演算プロセッサによって並列演算機構を論理的に構成することと、同期範囲指定機構を備え、同期範囲外の演算プロセッサからの同期要求に対しては割り込みを発生することである。
【００５３】
次に、図７を用いて、本実施形態による並列ジョブ多重スケジューリング方法が対象とする並列ジョブの形式について説明する。
図７は、本発明の一実施形態によって並列ジョブを複数の演算プロセッサを用いて実行する場合の１方式の概略図である。
【００５４】
図７において、矩形部Ｕ１～Ｕ１３，Ｐ１～Ｐ１３の１つ１つが情報処理プログラムが実行する計算の１単位を表すものとする。一般に、情報処理プログラムは計算機システム上で実行する処理の列を記述したものであり、単一の演算プロセッサから成る計算機システムで情報処理プログラムを実行する場合は、その記述された処理の列を逐次的に実行する。例えば、図７（Ａ）に示すように、１情報処理プログラムが処理Ｕ１～Ｕ１３から成る列を記述していたとすると、単一演算プロセッサから成る計算機システムでは、処理Ｕ１から順に処理Ｕ１３までが順番に行われる。ここでは、時間は図中の左から右に流れているとして表している。
【００５５】
一方、処理Ｕ１～Ｕ１３の内、例えば、処理Ｕ２～Ｕ６の５単位の処理の順序が任意であるとすれば、これらは複数の演算プロセッサ上で並列に実行することができる並列処理可能部分である。同様に処理Ｕ８～Ｕ１２の５単位の処理の順序が任意であるとすれば、これらは複数の演算プロセッサ上で並列に実行することができる並列処理可能部分である。残りの処理Ｕ１，Ｕ７，Ｕ１３は、逐次処理部分である。
【００５６】
そこで、本実施形態においては、処理Ｕ１～Ｕ１３の全体処理は、処理Ｐ１～Ｐ１３のように５台の演算プロセッサ上で実行することができる。ここで、並列処理可能部分である処理Ｕ２～Ｕ６は、並列処理部分として処理Ｐ２～Ｐ６のように時間的に同時に処理することが可能である。並列処理可能部分である処理Ｕ８～Ｕ１２も同様に、並列処理部分として処理Ｐ８～Ｐ１２のように並列に実行される。逐次処理部分Ｕ１，Ｕ７，Ｕ１３は、逐次処理部分として処理Ｐ１，Ｐ７，Ｐ１３のように実行される。ここでは、時間は図中で、左から右に流れているとして表している。また、上下方向に重ねられた矩形は時間的に同時に別々の演算プロセッサ上で実行されることを表している。
【００５７】
図７（Ｂ）において、処理Ｐ３～Ｐ６及び処理Ｐ９～Ｐ１２は、それぞれ、例えば、図１

に示した並列演算プロセッサ群を構成する並列演算プロセッサ１００５～１００８によって実行される。また、逐次処理部分である処理Ｐ１，Ｐ７，Ｐ１３及び並列処理部分の一部である処理Ｐ２，Ｐ８は、例えば、図１に示した複数の逐次演算プロセッサの１つである逐次演算プロセッサ１００１によって実行される。
なお、以降の説明でも、同様の記法を用いて説明するものとする。
【００５８】
次に、図８を用いて、本発明の一実施形態による並列ジョブの多重実行の概略について説明する。
図８は、本発明の一実施形態による並列ジョブの多重実行の概略説明図である。
【００５９】
図８においては、複数の演算プロセッサのジョブに対する割り当ての方法及び演算プロセッサ構成の動的な変更の概略を説明している。
【００６０】
上述したように、演算プロセッサ１００１～１００８は、逐次演算プロセッサと並列演算プロセッサ群とに論理的に分割されている。ここでは、演算プロセッサ１００１～１００４が、逐次演算プロセッサを構成し、演算プロセッサ１００５～１００８が、並列演算プロセッサ群を構成しているものとする。
【００６１】
第１のジョブＪ１０と第２のジョブＪ２０は、並列的に実行されている並列ジョブである。第１のジョブＪ１０は、逐次処理部分である処理Ｐ１０１と、並列処理部分である処理Ｐ１０２～Ｐ１０６と、逐次処理部分である処理Ｐ１０７から構成されている。また、第２のジョブＪ２０は、逐次処理部分である処理Ｐ２０１と、並列処理部分である処理Ｐ２０２～Ｐ２０６と、逐次処理部分である処理Ｐ２０７から構成されている。
【００６２】
並列ジョブＪ１０，Ｊ２０は、時刻Ｔ１において、逐次演算プロセッサ１００３，１００４上でそれぞれ起動される。第１のジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６の方が、第２のジョブＪ２０の並列処理部分の処理Ｐ２０２～Ｐ２０６よりも早い時刻Ｔ２において起動されるとすると、並列処理部分の処理の起動時には、第１のジョブＪ１０の逐次処理部分の処理Ｐ１０１を実行した逐次演算プロセッサ１００４が、並列演算プロセッサ群１００５～１００８を獲得して実行を進める。並列処理部分の処理の起動時には、逐次演算プロセッサ１００４と並列演算プロセッサ群１００５～１００８との間で、同期処理を行う。この同期処理は、ある所定の時期においては、唯一の逐次演算プロセッサからのみ可能なようにすることで並列演算プロセッサ群の割り当て制御を行う。即ち、時刻Ｔ２に逐次演算プロセッサ１００４が同期処理を開始すると、この時刻Ｔ２以降においては、逐次演算プロセッサ１００３は同期処理を行うことができない。逐次演算プロセッサ１００３の同期処理は、逐次演算プロセッサ１００４と並列演算プロセッサ１００５～１００８の並列処理が終了するまで行えないようになっている。なお、この詳細な処理については、図１２を用いて後述する。
【００６３】
逐次演算プロセッサ１００４による同期処理が完了すると、並列処理部分の処理Ｐ１０２～Ｐ１０６が、逐次演算プロセッサ１００４と並列演算プロセッサ群１００５～１００８によって並列的に実行される。
【００６４】
並列処理部分の処理Ｐ１０２～Ｐ１０６の終結時には、並列演算プロセッサ群１００５～１００８は、終了通知を発行し、これを契機にして他の並列実行待ちジョブの並列処理部分の実行に切り替わる。この際、並列処理部分の実行が終了しているため、並列演算プロセッサのレジスタの退避が大幅に削減されることによってオーバーヘッドの少ない切り替えを実現することができる。一般に、並列演算機構は、多くのレジスタを使って高速実行を実現しているため、従来の時分割による実行並列ジョブの切り替え方式では、並列機構使用中にジョブを切り替えると多量のレジスタの退避・回復が必要であり、性能劣化の大

きな原因となっているのに対して、本実施形態では、オーバーヘッドを低減できるものである。

【００６５】

時刻Ｔ３以降において、並列プロセッサ１００５～１００８によるジョブＪ１０の並列処理部分の実行がそれぞれ終了し、終了通知がなされ、そして、時刻Ｔ４において、全てのジョブＪ１０の並列処理部分の実行の終了が確認されたものとする。時刻Ｔ４において、ジョブＪ２０の逐次処理部分の処理Ｊ２０１が実行終了しており、並列処理部分Ｐ２０２～Ｐ２０６の実行待ちになっていたすると、逐次演算プロセッサ１００３は、並列演算プロセッサ１００４～１００８に対して同期処理を行った後、並列処理部分の処理Ｐ２０２～Ｐ２０６が、逐次演算プロセッサ１００３と並列演算プロセッサ群１００５～１００８によって並列的に実行される。なお、ジョブＪ１０の並列処理部分の実行が終了すると、逐次演算プロセッサ１００４は、ジョブＪ１０の残りの逐次処理部分の処理Ｐ１０７を実行する。

【００６６】

時刻Ｔ５における並列処理部分の処理Ｐ２０２～Ｐ２０６の終結時には、並列演算プロセッサ群１００５～１００８は、終了通知を発行し、逐次演算プロセッサ１００３は、ジョブＪ２０の残りの逐次処理部分の処理Ｐ２０７を実行する。なお、このとき、他の並列実行待ちジョブがあれば、そのジョブの並列処理部分の実行に切り替わる。

【００６７】

以上のようにして、本実施形態においては、複数のジョブを並列的に実行できるため、計算機システム内の全プロセッサの稼働率を高めることができる。即ち、図８に示す例では、２つのジョブＪ１０，Ｊ２０が並列的に実行されている。

【００６８】

ここで、従来の方式においては、例えば、時刻Ｔ１からＴ６の間においては、ジョブＪ１０だけしか実行できないものである。その結果、並列プロセッサ１００５～１００８が稼動できる時間は、時刻Ｔ２からＴ３の間に限られ、残りの時間（Ｔ１～Ｔ２，Ｔ３～Ｔ６）は、並列プロセッサ１００５～１００８は稼動していない遊休状態となっているものである。

【００６９】

それに対して、本実施形態の方式においては、並列プロセッサ１００５～１００８が稼動できる時間は、時刻Ｔ２からＴ３と、時刻Ｔ４からＴ５の間となり、並列プロセッサ１００５～１００８は稼動していない遊休状態となっている時間（Ｔ１～Ｔ２，Ｔ３～Ｔ４，Ｔ５～Ｔ６）を短くすることができるため、並列演算プロセッサ１００５～１００８の稼動率を向上することができるものである。

【００７０】

さらに、本実施形態においては、並列演算プロセッサ群は、逐次演算プロセッサと同等の演算プロセッサから構成されているため、逐次演算プロセッサと並列演算プロセッサ群の構成比率を動的に変更することが可能である。

【００７１】

図８において、時刻Ｔ６以降は、演算プロセッサの構成比率を変更した状態を示している。即ち、時刻Ｔ６以降においては、例えば、演算プロセッサ１００１が、逐次演算プロセッサを構成し、演算プロセッサ１００２～１００８が、並列演算プロセッサ群を構成している。従って、時刻Ｔ７において、新しいジョブＪ３０が起動されると、逐次処理部分の処理Ｐ３０１の実行が、逐次プロセッサ１００１上で起動される。時刻Ｔ８において、逐次処理部分の処理Ｐ３０１が終了すると、次の並列処理部分の処理の起動時に、逐次演算プロセッサ１００１と並列演算プロセッサ群１００２～１００８との間で、同期処理を行う。逐次演算プロセッサ１００１による同期処理が完了すると、並列処理部分の処理Ｐ３０２～Ｐ３０９が、逐次演算プロセッサ１００１と並列演算プロセッサ群１００２～１００８によって並列的に実行される。さらに、時刻Ｔ９において、並列処理が終了すると、逐次演算プロセッサ１００１は、逐次処理部分の処理Ｐ３１０を実行する。

【００７２】
以上説明したように、並列演算量が多い場合には、全体の演算プロセッサの内の１台を逐次演算プロセッサとして構成し、残りの７台の演算プロセッサを並列演算プロセッサ群として構成することにより、並列処理部分の処理時には、８台の演算プロセッサをすべて使用して実行することが可能となる。例えば、日中のように、並行ジョブを多数実行したい場合には、図８の左側に示すように、演算プロセッサ１００１～１００４によって逐次演算プロセッサを構成し、演算プロセッサ１００５～１００８によって並列演算プロセッサ群を構成するようにして、複数の逐次演算プロセッサとして構成する演算プロセッサの台数と、並列演算プロセッサ群を構成する演算プロセッサの台数とを等しくする。また、夜間のように、処理するジョブが少なく、しかも、並列演算量が多い場合には、全体の演算プロセッサの内の１台を逐次演算プロセッサとして構成し、残りの７台の演算プロセッサを並列演算プロセッサ群として構成することができる。
【００７３】
従って、図８に示す例では、例えば、時刻Ｔ６までは、日中のシステム運用の形態を示し、時刻Ｔ７以降が夜間のシステム運用とする。時刻Ｔ６～Ｔ７における演算プロセッサの構成比率の変更は、システムオペレータによって行われる。
【００７４】
このようにして、実行対象ジョブの逐次処理と並列処理の比率に応じて計算機システム内の演算プロセッサの構成比率を変更することができる。
【００７５】
従来の並列ジョブ実行を目的とした計算機システムにおいては、逐次処理を行う逐次専用演算プロセッサと、並列処理を行う並列専用演算プロセッサを固定的な形態で備えている。並列ジョブ毎に逐次処理部分と並列処理部分の比率が異なっているにも拘わらず、固定的な構成の計算機システムのみが提供され続けている。しかも、従来方式の計算機システムでは、並列演算の性能向上のために特別な並列演算プロセッサを採用するため、並列演算プロセッサは特別な演算しか行えない上、並列演算プロセッサ全体は１体でしか扱えず、分割使用は不能であった。すなわち、逐次演算量，並列演算量に応じて、逐次演算プロセッサと並列演算プロセッサの構成を替えることができないものである。
【００７６】
それに対して、本実施形態においては、計算機システムを構成する演算プロセッサを全て同等の機能の演算プロセッサにより構成するようにしているので、実行並列ジョブの性質，即ち、逐次処理部分と並列処理部分の比率に応じて、柔軟に逐次演算処理能力と並列演算処理能力の比率を動的に変化させて、計算機システムの稼動率を高めることが可能となる。
【００７７】
次に、図９を用いて、図７（Ｂ）に示したように並列化されたジョブの実行するための計算機システムの構成について説明する。
図９は、本発明の一実施形態による並列ジョブ多重スケジューリング装置のシステム構成図である。
【００７８】
本実施形態においては、計算機システムが８台の演算プロセッサによって構成されているものとして説明する。また、演算プロセッサ１００１～１００４を逐次演算プロセッサとして用い、演算プロセッサ１００５～１００８を並列演算プロセッサ群として用いるものとする。
【００７９】
１並列ジョブは、１つの逐次演算プロセッサ上で起動される。ここでは、ある並列ジョブが、逐次演算プロセッサ１００１上で起動されたとする。ジョブが起動されると、逐次処理部分の処理（図７；処理Ｐ１）の実行を逐次演算プロセッサ１００１上で開始する。この後、並列処理部分の処理（図７；処理Ｐ２～Ｐ６）に達すると、逐次演算プロセッサ１００１と並列演算プロセッサ群１００５～１００８を用いて、これら５単位の処理（図７

；処理Ｐ２～Ｐ６）を５台の演算プロセッサ１００１，１００５～１００８上で並列に行う。並列演算プロセッサ群を構成する演算プロセッサ１００５～１００８は一括して１並列ジョブに割り当てられる。

【００８０】

１並列ジョブの並列処理部分の実行の開始時と終了時にはそのジョブの逐次処理部分を実行する１台の逐次演算プロセッサ１００１と並列演算プロセッサ群を構成する演算プロセッサ１００５～１００８の間で同期処理を行う。この同期処理に参加する演算プロセッサ１００１，１００５～１００８は、同期範囲指定機構４１００（図１）によって指示される。並列処理部分開始時に行われる同期処理を契機として、各演算プロセッサ上では、並列処理部分の内、自分に割り当てられた処理を一斉に開始し、また、並列処理部分の終了時にも同期処理を行い、全ての演算プロセッサの処理が終了したことを確認して次の処理に進む。

【００８１】

各演算プロセッサ１００１～１００８は、演算プロセッサ間データ授受領域２１０１～２１０８を共有記憶装置２０００中に備え、並列処理部分の開始時には逐次演算プロセッサ１００１上で実行されているプログラムが、各並列演算プロセッサに割り当てられた処理の先頭アドレスを演算プロセッサ間データ授受領域２１０１～２１０８に書き込み、そのデータを各並列演算プロセッサ１００５～１００８が読み出すことによって、１並列処理部分の複数の演算プロセッサへの割り当てを実現する。

【００８２】

各演算プロセッサ１００１～１００８が実行するジョブ実行コードは、ユーザコード領域であるジョブ実行コード２１００に保持される。図７に示す例では、処理Ｐ１，Ｐ２，Ｐ７，Ｐ８，Ｐ１３が逐次演算プロセッサ１００１上で実行される。処理Ｐ３，Ｐ９が演算プロセッサ１００５上で、処理Ｐ４，Ｐ１０が演算プロセッサ１００６上で、処理Ｐ５，Ｐ１１が演算プロセッサ１００７上で、処理Ｐ５，Ｐ１２が演算プロセッサ１００８上でそれぞれ実行される。

【００８３】

次に、図１０を用いて、本実施形態における１並列処理部分の起動・終結時の実行の詳細について説明する。

図１０は、本発明の一実施形態による並列ジョブ多重スケジューリング方法における並列処理部分の起動・終結時の実行について説明するフローチャートである。

【００８４】

並列処理部分の起動と終結は、演算プロセッサ間に備えられたバリア線による同期処理を用いて行われる。図１において説明したように、演算プロセッサ１０００は、自らが同期に達したことを他の演算プロセッサに知らせるため同期到達通知信号線ＳＲＬを介して伝達される同期到達通知命令及び、他の演算プロセッサが同期に達するまで待つための同期完了待ち命令を備えている。これらの命令を用いて演算プロセッサ間の同期処理が行なわれる。

【００８５】

また、並列演算プロセッサ群１００５～１００８上で並列処理部分の実行を開始するに当たって必要となる初期データは、逐次演算機構と並列演算機構の間で互いに参照可能な演算プロセッサ間データ授受領域２１０５～２１０８を起動情報格納領域として用いて授受することにより実現する。

【００８６】

図１０は、１並列処理部分の逐次演算プロセッサからの起動、並列演算プロセッサ群上での処理、並列処理部分の終結の概略を示している。図１０の左側は、逐次演算プロセッサ１００１の処理を示しており、図１０の右側は、並列演算プロセッサ群を構成する個々の演算プロセッサ１００５～１００８の処理を示している。

ステップＳ１００１において、並列処理部分の起動を待っている並列演算プロセッサ１００５～１００８は、同期到達通知命令を実行し、さらに、ステップＳ１００２において、

同期完了待ち命令を実行して待ち状態になっている。この時点で、この同期処理に対して、同期に達していないのは逐次演算プロセッサ１００１のみである。

【００８７】

一方、ステップＳ１０１１において、逐次演算プロセッサ１００１は、並列処理部分の処理を並列演算プロセッサ群１００５～１００８上で起動するに当たって、並列演算プロセッサ群上で実行されるコードの先頭番地を、演算プロセッサ間データ授受領域２１０５～２１０８に、起動情報として設定する。

【００８８】

ステップＳ１０１２において、起動情報の設定が完了すると、逐次演算プロセッサ１００１は、同期到達通知命令を実行する。さらに、ステップＳ１０１３において、自らが並列処理部分の起動のための準備が整ったことを並列演算プロセッサ群側に知らせ、同期完了待ち命令を実行する。その後、同期処理の完了を待つ。

【００８９】

ステップ１０２１においては、逐次演算プロセッサ１００１上での同期到達通知命令の実行によって、逐次演算プロセッサ１００１と並列演算プロセッサ群１００５～１００８との間の同期処理が完了する。

【００９０】

ステップＳ１００３において、並列演算プロセッサ群１００５～１００８は、起動のための同期処理の終了後、それぞれ演算プロセッサ間データ授受領域２１０５～２１０８から逐次演算プロセッサ１００１が設定した並列処理部分の起動のための情報であるステップＳ１０１１において設定された実行コード先頭番地）を取り出し、ステップＳ１００４において、それぞれの並列処理部分の処理の実行が開始される。

【００９１】

一方、ステップＳ１０１４において、逐次演算プロセッサ１００１も、自らに割り当てられた並列処理部分の処理の実行を開始する。

【００９２】

ステップＳ１０１５において、逐次演算プロセッサ１００１は、自らに割り当てられた並列処理部分の処理の実行が終了すると、並列演算プロセッサ群上の並列処理部分の終結を待つために、同期到達通知命令を実行し、さらに、ステップＳ１０１６において、同期完了待ち命令を実行する。

【００９３】

他方では、ステップＳ１００５において、並列演算プロセッサ群を構成する演算プロセッサ１００５～１００８は、それぞれ、自らに割り当てられた並列処理部分の処理の実行が終了すると、並列演算プロセッサ群上の他の並列処理部分の終結を待つために、同期到達通知命令を実行し、さらに、ステップＳ１００６において、同期完了待ち命令を実行する。

【００９４】

ステップＳ１０２２において、演算プロセッサ１００１は、全ての演算プロセッサの処理の実行が終了し、同期に達した時点で、１並列処理部分の処理が終了する。並列処理部分の終結に際しては、並列処理部分の起動時に用いた演算プロセッサ間データ授受領域２１０５～２１０８に計算結果等のデータを書き込み、逐次演算プロセッサ１００１がこれらのデータを読み込むことにより、並列演算プロセッサ群１００５～１００８から逐次演算プロセッサ１００１へデータを渡すことができる。

【００９５】

次に、図１１を用いて、本発明の一実施形態による図８の左側に示したような複数の並列ジョブの多重実行方式について説明する。

図１１は、本発明の一実施形態による並列ジョブの多重実行の概略説明図である。

【００９６】

図１１においては、複数の演算プロセッサのジョブに対する割り当ての方法の概略を説明している。

本実施形態における複数の並列ジョブの多重実行方式においては、複数の並列ジョブは、複数の逐次演算プロセッサ上で互いに独立に起動される。それぞれの並列ジョブは、並列処理部分に達すると、並列演算プロセッサ群を用いてその並列処理部分の実行を行う。本実施形態においては、１プログラムの逐次処理部分に専用の逐次演算プロセッサを割り付けるため、１プログラムに一括して割り付けられる並列演算プロセッサ群の使用要求が衝突しない限り、各プログラムは停止することなく、互いに並行に実行することができる。

【００９７】

図１１に示す例は、図９に示した計算機システム上で複数の情報処理プログラムを多重に実行した場合について示している。図９に示した計算機システムにおいて、演算プロセッサ１００１～１００４が逐次演算プロセッサとして、演算プロセッサ１００５～１００８が並列演算プロセッサ群として用いられている。

【００９８】

第１のジョブＪ１０と第２のジョブＪ２０は、並列的に実行されている並列ジョブである。第１のジョブＪ１０は、逐次処理部分である処理Ｐ１０１と、並列処理部分である処理Ｐ１０２～Ｐ１０６と、逐次処理部分である処理Ｐ１０７から構成されている。また、第２のジョブＪ２０は、逐次処理部分である処理Ｐ２０１と、並列処理部分である処理Ｐ２０２～Ｐ２０６と、逐次処理部分である処理Ｐ２０７から構成されている。

【００９９】

並列ジョブＪ１０，Ｊ２０は、時刻Ｔ１において、逐次演算プロセッサ１００１，１００２上でそれぞれ起動される。このように、２つのジョブはそれぞれの逐次処理部分を別々の演算プロセッサで並行に処理する。第１のジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６の方が、第２のジョブＪ２０の並列処理部分の処理Ｐ２０２～Ｐ２０６よりも早い時刻Ｔ２において起動されるとすると、並列処理部分の処理の起動時には、第１のジョブＪ１０の逐次処理部分の処理Ｐ１０１を実行した逐次演算プロセッサ１００２が、並列演算プロセッサ群１００５～１００８を獲得して実行を進める。並列処理部分の処理の起動時には、逐次演算プロセッサ１００２と並列演算プロセッサ群１００５～１００８との間で、同期処理ＳＹＮ１を行う。

【０１００】

逐次演算プロセッサ１００２による同期処理が完了すると、並列処理部分の処理Ｐ１０２～Ｐ１０６が、逐次演算プロセッサ１００２と並列演算プロセッサ群１００５～１００８によって並列的に実行される。

【０１０１】

並列処理部分の処理Ｐ１０２～Ｐ１０６の終結時には、並列演算プロセッサ群１００５～１００８は、終了通知ＥＮＤ１を発行し、これを契機にして他の並列実行待ちジョブの並列処理部分の実行に切り替わる。この際、並列処理部分の実行が終了しているため、並列演算プロセッサのレジスタの退避が大幅に削減されることによってオーバーヘッドの少ない切り替えを実現することができる。

【０１０２】

時刻Ｔ３以降において、並列プロセッサ１００５～１００８によるジョブＪ１０の並列処理部分の実行がそれぞれ終了し、終了通知ＥＮＤ１がなされる。

【０１０３】

一方、ジョブＪ２０は、並列演算プロセッサ群１００５～１００８を使用しなければ、時刻Ｔ１～Ｔ４に示すように、ジョブＪ１０の処理と並行して処理Ｐ２１０を実行することができる。

【０１０４】

時刻Ｔ４において、ジョブＪ２０の逐次処理部分の処理Ｊ２０１が実行終了しており、並列処理部分Ｐ２０２～Ｐ２０６の実行待ちになっていたすると、逐次演算プロセッサ１００１は、並列演算プロセッサ１００５～１００８に対して同期処理ＳＹＮ２を行った後、並列処理部分の処理Ｐ２０２～Ｐ２０６が、逐次演算プロセッサ１００１と並列演算プロセッサ群１００５～１００８によって並列的に実行される。なお、ジョブＪ１０の並列処

理部分の実行が終了すると、逐次演算プロセッサ１００２は、ジョブＪ１０の残りの逐次処理部分の処理Ｐ１０７を実行する。

【０１０５】

時刻Ｔ５における並列処理部分の処理Ｐ２０２～Ｐ２０６の終結時には、並列演算プロセッサ群１００５～１００８は、終了通知ＥＮＤ２を発行し、逐次演算プロセッサ１００１は、ジョブＪ２０の残りの逐次処理部分の処理Ｐ２０７を実行する。

【０１０６】

このように複数のジョブが同時に並列処理部分を実行しなければ、１ジョブの逐次処理部分の実行中に遊休する並列演算プロセッサ群を他のジョブの並列処理部分の実行に割り当てることができ、システム全体の稼動率を高めることが可能になる。

【０１０７】

また、性能測定を行うような特別なジョブを除くと、並列ジョブの並列化率は必ずしも高くないことが知られており、１並列ジョブの処理中には並列演算機構の遊休時間帯が多数存在する。このため、この遊休時間帯を有効活用する本発明の方法によって、計算機システムのスループットの改善が可能になる。

【０１０８】

次に、図１２を用いて、図１１に示したように複数のプログラムが同時に並列実行部分を処理しようとした場合の並列演算プロセッサ群のプログラムへの割り当て制御方式に関して説明する。

図１２は、本発明の一実施形態による並列ジョブ多重スケジューリング方法における並列演算プロセッサ群のプログラムへの割り当て制御方式の説明図である。

【０１０９】

最初に、並列処理部分の起動時の処理について説明する。

計算機システムは、図９に示したように、８台の演算プロセッサから構成されている場合の処理として説明する。

【０１１０】

第１のジョブＪ１０と第２のジョブＪ２０は、図１１において説明したように、並列的に実行されている並列ジョブである。並列ジョブＪ１０，Ｊ２０は、時刻Ｔ１において、逐次演算プロセッサ１００１，１００２上でそれぞれ起動される。

【０１１１】

図中の符号４１１１，４１１２は、それぞれ、図２に示した８ビットからなる同期範囲指定レジスタ４１１０のビット１，２の状態を示している。演算プロセッサ１００１と演算プロセッサ１００２に対応する同期範囲指定レジスタ４１１０のビットの値は、図示する例では、実行開始当初は、逐次演算プロセッサの内、逐次演算プロセッサ１００１に対するビットに”１”が設定されているものとする。逐次演算プロセッサ１００２に対するビットには”０”が設定されている。

【０１１２】

並列処理部分の起動のためには、逐次演算プロセッサ１００１～１００４及び並列演算プロセッサ群１００５～１００８は、同期到達通知命令を用いて同期処理を行う。同期到達通知命令は、同期範囲指定機構４１００中のレジスタ４１１０の内容によって処理の場合分けを行う。同期到達通知命令を実行した演算プロセッサに対応するビットに”１”が設定されていれば、そのまま同期処理を行う。一方、ビットに”０”に設定されていれば、その逐次演算プロセッサには並列演算プロセッサ群の使用が許可されていないとして、その旨を伝える割り込みを発生させる。

【０１１３】

図１２に示した初期状態から、２つのジョブＪ１０，Ｊ２０が実行を開始し、ジョブＪ１０が先に並列処理部分の処理Ｐ１０２～Ｐ１０６に達したとする。この例では、同期範囲指定機構４１００のレジスタ４１１１のビット１のビット状態を見ると、”１”が設定されており、ジョブＪ１０は並列処理が許可されているので、そのまま起動ＳＹＮ１のための同期到達通知命令ＳＲＬ１は実行され、並列処理部分の処理Ｐ１０２～Ｐ１０６が開

始される。

【０１１４】

一方、ジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６の実行中にジョブＪ２０が、並列処理部分の起動ＳＹＮ２のために、同期到達通知命令ＳＲＬ２を実行したとすると、同期範囲指定機構４１００のレジスタ４１１２のビット２に”０”が設定されているため、演算プロセッサ上には割り込みＩＮＴ１が発生する。この割り込みＩＮＴ１を契機として、共有記憶装置２０００内のスケジューラＳＣＨ１が起動され、並列処理要求ＲＥＱ１は並列実行待ちキューＣＵＥに入れられ、並列演算プロセッサ群が割り当てられるのを待つことになる。

【０１１５】

並列実行待ちキューＣＵＥは、図６において説明したように、共有記憶装置２０００上に保持された階級管理情報２５００の中のキュー先頭アドレス設定フィールド２５０２の中に、並列処理要求ＲＥＱが発生した順に順次入れられる。

【０１１６】

このように、高々１つの逐次演算プロセッサのみのビットに”１”を設定し、並列処理部分の起動時に実行される同期到達通知命令が自らを実行した演算プロセッサに対応する同期範囲指定機構４１００の中のレジスタ４１１０に対応するビットを調べ、自分が並列演算プロセッサ群の使用が可能か否かの場合分けに従った処理を行うことによって、排他的に並列演算プロセッサ群を割り付けることが可能である。

【０１１７】

１時期には、同期範囲指定機構４１００内のレジスタ４１１０は、並列演算プロセッサ群に対応するビットと、並列演算プロセッサ群が割り当てられた逐次演算プロセッサに対応するビットに”１”が設定されている。例えば、図２に示した例では、並列演算プロセッサ群１００５～１００８に対応するビット５～８と、並列演算プロセッサ群が割り当てられた逐次演算プロセッサ１００１に対応するビット１に”１”が設定されている。

【０１１８】

従って、並列演算プロセッサ群が割り当てられている逐次演算プロセッサの同期到達通知命令を用いた並列処理要求に対しては、そのまま並列処理の開始を許可し、一方、並列演算プロセッサが割り当てられていない逐次演算プロセッサの並列処理要求に対しては割り込みを発生させてスケジューラを起動して待ち状態にし、起動要求を並列実行待ちキューに入れることが可能になる。

【０１１９】

なお、並列演算プロセッサ群上でのジョブの切替え時には、旧ジョブに割り当てられた逐次演算プロセッサに対応するビットに”０”を設定してその演算プロセッサの並列処理の起動を禁止し、新ジョブに割り当てられた逐次演算プロセッサに対応するビットには”１”を設定して起動を許可にする。

【０１２０】

このように同期範囲指定機構４１００を、逐次演算プロセッサと並列演算プロセッサ群の組み合わせを指定する並列演算機組合わせ機構として用いることが、本実施形態の方法の大きな特徴である。

【０１２１】

次に、図１２に戻って、並列処理部分の実行完了時の計算機システムの動作について説明する。図１２の右側の部分は、並列演算プロセッサ群上での実行プログラムの変更の概略を示している。

【０１２２】

本実施形態においては、並列演算プロセッサ群を構成する演算プロセッサは、１つのプログラムの並列処理部分に一括して割り付けられ、１つの並列処理部分の起動時と終結時に並列演算プロセッサ群と逐次演算プロセッサ間で同期処理を行う。

【０１２３】

上述したように、ジョブＪ１０が並列処理部分の処理Ｐ１０２～Ｐ１０６を並列演算プロ

セッサ群１００５～１００８上で処理している間に、ジョブＪ２０が並列演算の起動に失敗し、並列実行待ちキューに入れられているものとする。

【０１２４】

並列処理部分の終結のための同期処理の完了後に、▲１▼引き続く逐次処理部分の実行時間が一定時間より長いとコンパイラが判断できたとき、或いは、▲２▼プログラムの記述中にユーザが並列処理部分の最終部分で指示を行ったときに終結時の同期完了後に、並列演算プロセッサ群を即座に他のジョブの並列処理部分の実行に割り当て替える特別な終了通知（並列部分終了通知）を行う。この並列部分終了通知が行われると、並列演算プロセッサ群を構成する演算プロセッサ上に割り込みが発生する。

【０１２５】

即ち、ジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６が終結するに当たって、並列演算プロセッサ群１００５～１００８は、並列演算プロセッサ群を並列実行待ちキューに入れられているジョブに譲渡することを指示する並列部分終了通知ＥＮＤ１が発行される。この終了通知ＥＮＤ１によって、並列演算プロセッサ群１００５～１００８上では割り込みＩＮＴ２が発生する。また、この割り込みＩＮＴ２を契機として共有記憶装置２０００内のスケジューラＳＣＨ２が起動され、終結した先行情報処理プログラムの状態の内の規定の最小限のレジスタのみを退避する。

【０１２６】

この時、ジョブＪ１０の並列演算プロセッサ群１００５～１００８上での処理は完了しているため、演算プロセッサの全ての状態を退避する必要はなく、規定の最小限のレジスタの内容のみを退避する。即ち、並列処理部分の処理の終結時には、並列演算プロセッサ群を構成する演算プロセッサ１００５～１００８のレジスタ上の非常に少ない量のデータの退避のみを行えばよいため、並列演算プロセッサ群の割り当てられたジョブの変更を高速に行うことができる。

【０１２７】

ここで、必要最小限のレジスタのみが退避されていることをレジスタ退避範囲フラグ２４０８（図５）を設定することによって記録し、並列実行待ちの他のジョブの処理を開始する。

【０１２８】

なお、終了通知が行われなかったときは、切り替えを行わない。ここで、後続する逐次処理部分の実行時間が一定時間より長い場合にのみ終了通知を発行することが重要である。これによって、１並列ジョブの逐次処理時間が、他の並列ジョブの並列処理部分の実行に十分な長さをもっている時のみ、並列演算プロセッサの切り替えが行われ、並列演算プロセッサ群の遊休時間に、他の並列ジョブの並列処理部分を割り当てることが可能になる。もちろん、後続の逐次処理部分の実行時間が短い場合に終了通知を発行して、並列演算プロセッサ群の割り当て先を変更しても計算結果には影響を与えない。

【０１２９】

ところが、例えば、１並列ジョブが実行時間が短い並列処理部分と逐次処理部分を頻繁に繰り返す場合に、並列処理部分の終結毎に終了通知を発行して並列演算プロセッサ群の割り当て先を変更すると、スケジューリングのオーバーヘッドが実質的な計算に費やされる計算時間に対して大きくなり、計算機システム全体の効率を低下させる。

【０１３０】

以上のように、本実施形態においては、単純に並列処理部分の終了を契機として並列演算プロセッサの割り当て先を切り替えるのではなく、実行ジョブの逐次処理及び並列処理の実行時間の長さと分布を解析し、その結果に基づいて切り替えを行うという機能をもつことによって、１ジョブの処理中に遊休する並列演算プロセッサ群を低オーバヘッドで他のジョブに割り当てることが可能となる。

【０１３１】

割り込みＩＮＴ２の発生後、スケジューラＳＣＨ２は、並列実行待ちキューＣＵＥに入っている先頭の並列処理要求ＲＥＱ１を読みだして、並列実行待ち状態であったジョブＪ２

０に並列演算プロセッサ群１００５～１００８が割り付けられ、ジョブＪ２０の並列処理部分の処理Ｐ２０２～Ｐ２０６の実行が開始される。

【０１３２】

並列演算プロセッサ群上でのジョブの切替え時には、スケジューラＳＣＨ２は、同期範囲指定機構４１００内のレジスタ４１１０の設定を変更し、旧ジョブであるジョブＪ１０に割り当てられた逐次演算プロセッサ１００１に対応するビット４１１１に”０”を設定して、その演算プロセッサの並列処理の起動を禁止し、新ジョブであるジョブＪ２０に割り当てられた逐次演算プロセッサ１００２に対応するビット４１１２には”１”を設定して起動を許可にする。なお、スケジューラＳＣＨ２の動作の詳細については、図１４～図２１を用いて後述する。

【０１３３】

ここで、スケジューラが並列プロセッサ群上のみで起動されることが重要である。ジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６が終了すると、演算プロセッサ１００１は、ジョブＪ１０の引き続く逐次処理部分の処理Ｐ１０７の実行を即座に開始する。

【０１３４】

一方、これと並行して、並列演算プロセッサ群１００５～１００８上では、ジョブ切り替えのための処理が行われるが、この処理はジョブＪ１０の逐次処理部分の処理Ｐ１０７の実行に影響を及ぼすことなく行われるため、ジョブ切り替えのオーバヘッドを隠蔽することが可能になる。

【０１３５】

以上説明したように、本実施形態による並列ジョブ多重スケジューリング方法によれば、並列処理部分は並列演算プロセッサ群と逐次演算プロセッサが一体となって実行するため、逐次演算プロセッサが遊休することはない。これは、並列演算プロセッサ群を逐次演算プロセッサと同等の演算プロセッサで構成し、並列演算プロセッサ群を複数の逐次演算プロセッサで共有して使用するという本実施形態の方法を用いることで初めて可能になるものである。

【０１３６】

また、従来の方法では、切り替えのタイミングはタイマ割り込みによるものであるため、切り替えに際しては、一般に多量のレジスタの内容の退避・回復が必要になる。それに対して、本実施形態の方法では、並列処理部分の終了を並列演算プロセッサ群の割り当て替えの契機として用いることによって、切りのよいタイミングでジョブを切り替え、退避・回復するレジスタの数を大幅に削減することが可能になり、ジョブ切り替えのオーバヘッドを取り除いている。

【０１３７】

ここで、本実施形態による同期処理の失敗検出方法について説明する。

１つの同期処理に参加する複数の並列演算プロセッサの内、予め指定した時間（所定時間）内に同期ポイントに達しない並列演算プロセッサが存在したとき、その同期処理は失敗したと定義する。本実施形態では、同期処理が失敗したと判断するまでの時間を設定する同期タイムアウトレジスタが用意されている。

【０１３８】

同期処理が開始され、この指定された時間が経過する前にすべての演算プロセッサが同期ポイントに達しない場合、同期失敗を知らせる割り込みが演算プロセッサ上に発生し、これを契機にスケジューラが起動され、エラー処理を行う。同期タイムアウトレジスタの設定は、共有記憶装置２０００上に備えられた同期タイムアウトレジスタ設定フィールド２３０７（図４）によって行う。

【０１３９】

次に、図１３を用いて、本実施形態によるジョブ切り替え時のデータ授受領域の管理方法について説明する。

図１３は、本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるデータ授受領域の管理方法の説明図である。

【０１４０】
逐次演算プロセッサと並列演算プロセッサとの間のデータの授受方法については既に説明したとおりである。本実施形態においては、複数のジョブを並行に動作させるため、逐次演算プロセッサと並列演算プロセッサとの間のデータの授受に用いる演算プロセッサ間データ授受領域２１０１～２１０８の内容は、ジョブの切替え毎に、退避・回復する必要がある。そこで、並列に実行させようとするジョブの数だけ演算プロセッサ間データ授受領域を実記憶空間上に用意し、これらを同一の仮想記憶空間上の番地に対応づけるようにしている。演算プロセッサ間データ授受領域は、演算プロセッサ間データ授受領域対応つけフィールド２３０３（図４）に設定される。ジョブの切り替えに際して、両者の対応づけを変更することによって、演算プロセッサ間データ授受領域の状態を退避・回復する必要がなくなり、並列演算機構の割り当て変更を高速に行うことができる。
【０１４１】
ここで、図１３を用いて、例えば、２つのジョブＡ，Ｂを並行に実行する場合について説明する。例えば、ジョブＡは、図１１に示したジョブＪ１０であり、ジョブＢは、図１１に示したジョブＪ２０とする。演算プロセッサ１００５に対する演算プロセッサ間データ授受領域２１０５を例にとって説明する。
【０１４２】
２つのジョブＡ，Ｂに対応して、それぞれ専用の２つの演算プロセッサ間データ授受領域が、実記憶空間上の異なった記憶ページＰＡ１，ＰＡ２上に用意される。それぞれの記憶ページＰＡ１，ＰＡ２の先頭番地を、実記憶番地ａと実記憶番地ｂとする。
【０１４３】
一方、仮想記憶空間上では、仮想記憶番地ｘに、演算プロセッサ間データ授受領域２１０５として用意される。逐次演算プロセッサ１００１及び並列演算プロセッサ群の中の並列演算プロセッサ１００５は、見かけ上、この仮想記憶空間上の領域２１０５を介してデータのやり取りを行う。一方、仮想記憶番地ｘには、現在実行されているジョブ専用の領域の番地が対応づけられている。この対応付けは、逐次並列間データ授受領域対応付けフィールドＣＦによって指定される。
【０１４４】
図示の例は、ジョブＡの専用領域の先頭番地ａが仮想記憶番地ｘに対応付けられている状況を示している。これによって、ジョブＡの実行中はこの領域に対する逐次演算プロセッサ１００１からのデータの書き込み及び並列演算プロセッサ１００５からのデータの読み出しによって、逐次演算プロセッサと並列演算プロセッサ群との間のデータの授受を実現する。
【０１４５】
実行ジョブの切替え時、例えば、ジョブＡからジョブＢに実行ジョブが切り替わるときは、仮想記憶番地ｘに対応づけられた実記憶番地を”ａ”から”ｂ”に変更することによって、ジョブＢの専用領域への書き込みと読み出しができるようになる。
【０１４６】
次に、図１４～図２１を用いて、本実施形態による並列演算プロセッサ割り当て切り替えのためのスケジューラの処理手順について説明する。
図１４は、本発明の一実施形態による図９で示した並列演算プロセッサ群上で起動されるスケジューラＳＣＨ２（図１２）の全体の動作の概略を示すフローチャートであり、図１５～図２１は、図１４に示した処理の細部を説明するフローチャートである。
【０１４７】
図１５は、図１４のステップＳ１４５１の処理の詳細を説明したものである。図１６は、図１４のステップ１４１２の処理の詳細を説明したものである。図１７は、図１４のステップＳ１４１３の処理の詳細を説明したものである。図１８は、図１４のステップＳ１４０２，Ｓ１４０３の処理の詳細を説明したものである。図１９は、図１４のステップＳ１４０４の詳細を説明したものである。図２０は、図１４のステップＳ１４１４の処理の詳細を説明したものである。図２１は、図１４のステップＳ１４３１，Ｓ１４３２の処理の

詳細を説明したものである。

【０１４８】

図１４において、ステップＳ１４０１において、並列演算プロセッサ１００５～１００７における並列処理部分が終了し、ステップＳ１４１１において、並列演算プロセッサ１００８における並列処理部分が終了し、ステップＳ１４２１において、逐次演算プロセッサ１００１における並列処理部分が終了すると、ステップＳ１４４１において、並列終結時の同期処理が行われる。

【０１４９】

並列終結時の同期処理に引き続いて、ステップＳ１４５１において、終了通知とスケジューラの起動処理が実行される。即ち、図１５に詳細に示すように、ステップＳ１４５１１において、並列演算プロセッサ１００５～１００７は、並列演算プロセッサ群の譲渡を指示する終了通知を発行し、また、ステップＳ１４５１６において、並列演算プロセッサ１００８は、並列演算プロセッサ群の譲渡を指示する終了通知を発行する。

【０１５０】

次に、ステップＳ１４５１２において、並列演算プロセッサ１００５～１００７では、割り込みが発生し、また、ステップＳ１４５１７において、並列演算プロセッサ１００８では、割り込みが発生する。これらの割り込みを契機として、ステップＳ１４５１３において、並列演算プロセッサ１００５～１００７では、スケジューラが起動し、また、ステップＳ１４５１８において、並列演算プロセッサ１００８では、スケジューラが起動する。

【０１５１】

スケジューラが起動すると、ステップＳ１４５１４において、並列演算プロセッサ１００５～１００７では、以降で必要となるワークレジスタを退避し、また、ステップＳ１４５１９において、並列演算プロセッサ１００８では、以降で必要となるワークレジスタを退避する。

【０１５２】

次に、図１４に戻って、ステップＳ１４４２において、演算プロセッサ１００１，１００５～１００８は、全てスケジューリング可能な状態に達したことを保証するために同期処理を行う。

【０１５３】

次に、演算プロセッサ１００８が親プロセッサとして、スケジューラの中で振る舞う。ステップＳ１４１２において、演算プロセッサ１００８は、図６のキュー先頭アドレス設定フィールド２５０２を参照して、並列実行待ちキューＣＵＥ（図１２）が空か否かを調べる。即ち、図１６に詳細に示すように、ステップＳ１４１２１において、並列演算プロセッサ１００８は、並列実行待ちキューＣＵＥが空か否かを調べる。

【０１５４】

その結果、キューが空の場合には、ステップＳ１４１２２において、演算プロセッサ１００８は、現ジョブの並列起動を再許可した後、ステップＳ１４１２３において、変数ｎｏ＿ｓｗｉｔｃｈに”＃ｆ”（ｆａｉｌ）を書き込む。

【０１５５】

キューが空でない場合には、ステップＳ１４１２４において、演算プロセッサ１００８は、変数ｎｏ＿ｓｗｉｔｃｈに”＃ｔ”’（ｔｒｕｅ）を書き込む。これらの変数ｎｏ＿ｓｗｉｔｃｈへの書き込みは、各演算プロセッサに伝達される。

【０１５６】

さらに、ステップＳ１４１２５において、演算プロセッサ１００８は、キューから切り替え先新ジョブの並列起動要求ＲＥＱ１（図１２）を取り出し、演算プロセッサ１００５～１００７が参照可能な位置に格納する。さらに、ステップＳ１４１２６において、新ジョブの逐次演算プロセッサに対応する同期範囲指定レジスタのビットに”１”を設定しての並列演算プロセッサ群の使用を許可する。即ち、例えば、図１２に示したように、ジョブＪ２０の逐次演算プロセッサ１００２に対する同期範囲指定レジスタのビット４１１２に”１”を設定する。

【０１５７】
図１４に戻って、ステップＳ１４１２の処理が終了すると、ステップＳ１４４３において、同期命令を実行する。この同期処理は、キューが空であった場合には、変数ｎｏ＿ｓｗｉｔｃｈへの変更と現ジョブの並列起動再許可の終了の保証を行うため、空でなかった場合は、変数ｎｏ＿ｓｗｉｔｃｈへの変更及び、新ジョブの設定、新ジョブの逐次演算プロセッサに対する並列演算プロセッサ群の使用の許可の設定の完了を保証するために行うものである。
【０１５８】
続けて、演算プロセッサ１００８の処理について説明すると、ステップＳ１４１２における並列待ちキューが空の場合には、ステップＳ１４１３において、現ジョブ続行する。即ち、図１７に詳細に示すように、ステップＳ１４１３１において、演算プロセッサ１００８は、ワークレジスタを回復し、その後、ステップＳ１４１３２において、現ジョブを続行する。元のジョブでは、並列処理部分は終了しているため、並列演算プロセッサ群１００８上では、次の並列演算実行要求が到着するまで、待ち状態に入ることになる。
【０１５９】
一方、演算プロセッサ１００５～１００７は、ステップＳ１４４３における同期処理の完了後、ステップＳ１４０２において、並列待ちキューが空か否かを判断し、ステップＳ１４０３における現ジョブの続行か、若しくは、ステップＳ１４０４における新ジョブへの切替を実行する。ここで、ステップＳ１４０２における判断は、図１８のステップＳ１４０２に示すように、演算プロセッサ１００８）が、ステップＳ１４１２３，Ｓ１４１２４（図１６）において設定した変数ｎｏ＿ｓｗｉｔｃｈの値に従って場合分けして、処理を進める。
【０１６０】
図１８において、変数ｎｏ＿ｓｗｉｔｃｈが＃ｆであった場合は、ステップＳ１４０３１において、演算プロセッサ１００５～１００７は、実行ジョブの変更は行わず、先に退避したワークレジスタを回復し、ステップＳ１４０３２において、元のジョブの実行を再開する。元のジョブでは、並列処理部分は終了しているため、並列演算プロセッサ群１００５～１００７上では、次の並列演算実行要求が到着するまで、待ち状態に入ることになる。
【０１６１】
一方、演算プロセッサ８がｎｏ＿ｓｗｉｔｃｈに＃ｔを設定している場合、ステップＳ１４０４において、新ジョブへの切替えを実行する。即ち、図１９に詳細に示したように、演算プロセッサ１００５～１００７は、ステップＳ１４０４１において、旧ジョブの実行の状態を規定された最小限のレジスタのみを退避する。そして、レジスタ退避範囲フラグ２４０８（図５）を設定して、最小限のレジスタのみが退避されたことを記録した後、新しいジョブの実行を開始する。
【０１６２】
即ち、ステップＳ１４０４２において、図１３において説明したように、新ジョブの仮想空間の切り替え設定を行い、ステップＳ１４０４３において、データ授受領域の対応付けの変更を行う。さらに、ステップ１４０４４において、これから回復するジョブのレジスタ退避範囲フラグを参照する。レジスタ退避範囲フラグが設定されていれば、ステップＳ１４０４５において、必要最低限のレジスタのみを回復し、設定されていなければ、ステップＳ１４０４６において、全レジスタを回復して、ステップＳ１４０４７において、新ジョブを再開する。
【０１６３】
一方、図１４のステップＳ１４１４における演算プロセッサ１００８の新ジョブへの切替えは、図２０に詳細を示す処理フローに従って行われる。即ち、図２０のステップＳ１４１４１において、最低限のレジスタを退避し、ステップＳ１４１４２において、最低限のレジスタのみが退避されたことをレジスタ退避範囲フラグを設定し、ステップＳ１４１４３において、図１３において説明したように、新ジョブの仮想空間の切り替え設定を行い

、ステップＳ１４１４４において、データ授受領域の対応付けの変更を行う。さらに、ステップ１４１４５において、これから回復するジョブのレジスタ退避範囲フラグを参照する。レジスタ退避範囲フラグが設定されていれば、ステップＳ１４１４６において、必要最低限のレジスタのみを回復し、ステップＳ１４１４７において、レジスタ退避範囲フラグをリセットする。一方、レジスタ退避範囲フラグが設定されていなければ、ステップＳ１４１４８において、全レジスタを回復して、ステップＳ１４１４９において、新ジョブを再開する。

【０１６４】

次に、図１４における演算プロセッサ１００２の処理について説明する。

ステップＳ１４３１において、並列起動要求がなされた場合、図１２において説明したように、既に他の並列ジョブが実行中であるため、並列起動に失敗し、ステップＳ１４３２において、並列演算プロセッサの割当待ちとなる。即ち、図２１に詳細に説明するように、ステップＳ１４３１において、並列起動に失敗すると、ステップＳ１４３２１において、起動禁止割り込みが発生する。これを契機に、ステップＳ１４３２２において、スケジューラが起動され、ステップＳ１４３２３において、スケジューラは、起動要求を並列実行待ちキューへ登録し、ステップＳ１４３２４において、並列演算プロセッサ群の割り当て待ちとなっている。以上が、並列演算プロセッサ上で起動されたスケジューラの動作の概略である。

【０１６５】

さらに、図１４において、ステップＳ１４０４，Ｓ１４１４における新ジョブへの切替え処理が行われると、ステップＳ１４４４において、演算プロセッサ１００２，１００５～１００８は、並列部分起動時の同期処理を行う。同期処理が終了すると、ステップＳ１４０５において、演算プロセッサ１００５～１００７は、並列処理部分の処理を開始し、ステップＳ１４１５において、演算プロセッサ１００８は、並列処理部分の処理を開始し、ステップＳ１４３３において、演算プロセッサ１００２は、並列処理部分の処理を開始する。以上のようにして、並列演算プロセッサ群上で起動されるスケジューラを用いた並列演算プロセッサ割り当て切り替えが実行される。

【０１６６】

次に、図２２を用いて、本実施形態による計算機システム内の演算プロセッサの逐次演算プロセッサ及び並列演算プロセッサ群への割り付け個数の動的な変更について説明する。図２２は、本発明の一実施形態による並列ジョブ多重スケジューリング方法により演算プロセッサの割り付けを動的変更する例の説明図である。

【０１６７】

上述したように、本実施形態においては、複数の演算プロセッサを論理的に逐次演算プロセッサと並列演算プロセッサに分割し、１時期には１つの逐次演算プロセッサが複数の演算プロセッサから構成される並列演算プロセッサ群を一括して獲得して、この並列演算プロセッサ群を用いて並列処理部分の実行を行っている。ここで、逐次演算プロセッサに用いられる演算プロセッサと並列演算プロセッサ群に用いられる演算プロセッサは同等のものであり、且つ同等の装置によって接続されるため、計算機システム内の演算プロセッサを、逐次演算プロセッサに割り当てる個数と並列演算プロセッサ群に割り当てる個数の比率を動的に変更することが可能であり、実行ジョブが要求する計算機資源の大小に応じた運用を行えるものである。この割り付け個数の動的変更は、共有記憶装置２０００の管理情報２３００の中の構成変更情報フィールド２３０４（図４）に保持されたデータに基づいて、構成比率、頻度、契機等が指定される。

【０１６８】

図２２に示す例は、時間Ｔの経過毎に演算プロセッサにタイマ割り込みを発生させ、このタイマ割り込みを契機として、演算プロセッサ上でスケジューラを起動する。起動されたスケジューラは、直前まで実行されていたプログラムの状態を退避し、計算機システム内の逐次演算プロセッサに割り当てる演算プロセッサ数と並列演算プロセッサ群に割り当てる演算プロセッサ数の構成比率を変更するように運用される。なお、スケジューラの処理

動作の詳細については、図２３を用いて後述する。

【０１６９】

例えば、図９に示した８台の演算プロセッサから構成される計算機システムにおいて、８台の演算プロセッサの内の４台の演算プロセッサを逐次演算プロセッサとし、残りの４台の演算プロセッサを並列演算プロセッサ群として用いることによって複数のジョブを並行実行し、システム全体のスループットを向上させるシステム構成や、８台の演算プロセッサの内の１台を逐次演算プロセッサとして用い、残り７台を並列演算プロセッサ群として用いる１ジョブの最高性能を実現するシステム構成など、異なった演算プロセッサの使用方法を同一の枠組内で実現することが可能であり、且つこれらのシステム構成を時間Ｔの経過毎に切り替えることでさまざまな要求を満たすことが可能になる。

【０１７０】

ここで、図２２のシステム構成の動的変更について具体的に説明する。

図９に示した８台の演算プロセッサから構成される計算機システムにおいて、演算プロセッサ１００１～１００８は、逐次演算プロセッサと並列演算プロセッサ群とに論理的に分割されている。階級B^Tの実行時には、演算プロセッサ１００１～１００４が、逐次演算プロセッサを構成し、演算プロセッサ１００５～１００８が、並列演算プロセッサ群を構成しているものとする。また、階級A^Tの実行時には、演算プロセッサ１００１が、逐次演算プロセッサを構成し、演算プロセッサ１００２～１００８が、並列演算プロセッサ群を構成しているものとする。

【０１７１】

第１のジョブＪ１０，第２のジョブＪ２０，第４のジョブＪ４０，第５のジョブＪ５０は、並列的に実行されている並列ジョブである。第１のジョブＪ１０は、逐次処理部分である処理Ｐ１０１と、並列処理部分である処理Ｐ１０２～Ｐ１０６と、逐次処理部分である処理Ｐ１０７から構成されている。また、第２のジョブＪ２０は、逐次処理部分である処理Ｐ２０１と、並列処理部分である処理Ｐ２０２～Ｐ２０６と、逐次処理部分である処理Ｐ２０７から構成されている。第４のジョブＪ４０は、逐次処理部分である処理Ｐ４０１のみから構成されており、第５のジョブＪ５０は、逐次処理部分である処理Ｐ５０１のみから構成されている。

【０１７２】

最初に、並列ジョブＪ１０，Ｊ２０、Ｊ４０，Ｊ５０が、逐次演算プロセッサ１００４，１００３，１００２，１００１上でそれぞれ起動される。第１のジョブＪ１０の逐次処理部分の処理Ｐ１０１が終了すると、第１のジョブＪ１０の逐次処理部分の処理Ｐ１０１を実行した逐次演算プロセッサ１００４が、並列演算プロセッサ群１００５～１００８を獲得して、ジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６が実行を進める。

【０１７３】

並列処理部分の処理Ｐ１０２～Ｐ１０６が終結すると、逐次演算プロセッサ１００４は、続く逐次処理部分Ｐ１０７の処理を開始する。また、並列プロセッサ１００５～１００８によるジョブＪ１０の並列処理部分の実行の終了時に、ジョブＪ２０の並列処理部分Ｐ２０２～Ｐ２０６の実行待ちになっていたすると、逐次演算プロセッサ１００３は、並列演算プロセッサ１００４～１００８に対して同期処理を行った後、並列処理部分の処理Ｐ２０２～Ｐ２０６が、逐次演算プロセッサ１００３と並列演算プロセッサ群１００５～１００８によって並列的に実行される。並列処理部分の処理Ｐ２０２～Ｐ２０６の終結時には、並列演算プロセッサ群１００５～１００８は、終了通知を発行し、逐次演算プロセッサ１００３は、ジョブＪ２０の残りの逐次処理部分の処理Ｐ２０７を実行する。

【０１７４】

共有記憶装置２０００の管理情報２３００の中の構成変更情報フィールド２３０４（図４）に保持されたデータに基づいて、例えば、時間Ｔの経過毎に演算プロセッサ１００１～１００８上に割り込みを発生する。この割り込みを契機として、演算プロセッサ１００１～１００８上ではスケジューラＳＣＨ１０，ＳＣＨ１１が起動され、システムの構成と実行対象となるプログラムを変更して、逐次演算プロセッサと並列演算プロセッサ群の構成

比率，即ち、割り付け個数の動的変更を行う。

【０１７５】

即ち、階級B^Tの実行時が終了すると、割り込みが発生し、スケジューラＳＣＨ１０が起動して、演算プロセッサ１００１が、逐次演算プロセッサを構成し、演算プロセッサ１００２～１００８が、並列演算プロセッサ群を構成するシステム構成に変更して、階級A^Tを実行可能とする。

【０１７６】

階級A^Tを実行時には、例えば、第３のジョブＪ３０が実行される。新しいジョブＪ３０が起動されると、逐次処理部分の処理Ｐ３０１の実行が、逐次プロセッサ１００１上で起動される。逐次処理部分の処理Ｐ３０１が終了すると、次の並列処理部分の処理の起動時に、逐次演算プロセッサ１００１と並列演算プロセッサ群１００２～１００８との間の同期処理が完了すると、並列処理部分の処理Ｐ３０２～Ｐ３０９が、逐次演算プロセッサ１００１と並列演算プロセッサ群１００２～１００８によって並列的に実行される。さらに、並列処理が終了すると、逐次演算プロセッサ１００１は、逐次処理部分の処理Ｐ３１０を実行する。

【０１７７】

逐次処理部分の処理Ｐ３１０が終了すると、次の並列処理部分の処理の起動時に、逐次演算プロセッサ１００１と並列演算プロセッサ群１００２～１００８との間の同期処理が完了すると、並列処理部分の処理Ｐ３１１～Ｐ３１８が、逐次演算プロセッサ１００１と並列演算プロセッサ群１００２～１００８によって並列的に実行される。さらに、並列処理が終了すると、逐次演算プロセッサ１００１は、逐次処理部分の処理Ｐ３１９を実行する。

【０１７８】

以上のようにして、実行ジョブが要求する計算機資源の大小に応じて、逐次演算プロセッサと並列演算プロセッサ群の構成比率を変更した運用が可能となる。従って、実行並列ジョブの性質，即ち、逐次処理部分と並列処理部分の比率に応じて、柔軟に逐次演算処理能力と並列演算処理能力の比率を動的に変化させて、計算機システムの稼動率を高めることが可能となる。

【０１７９】

次に、図２３を用いて、本実施形態によるジョブ階級切り替え時のスケジューラの動作の概略について説明する。

図２３は、本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるジョブ階級切り替え時のスケジューラの動作を示すフローチャートである。

【０１８０】

ステップＳ２３０１において、共有記憶装置２０００の中の管理情報２３００の構成変更情報設定フィールド２３０４（図４）に設定されている構成変更情報に従って、逐次演算プロセッサ１００１に対してタイマ割り込みが発生する。この割り込みを契機として、ステップＳ２３０２において、演算プロセッサ１００１は、スケジューラＳＣＨを起動する。

【０１８１】

スケジューラＳＣＨが起動されると、ステップＳ２３０３において、演算プロセッサ１００１は、以降の処理で用いるワークレジスタの内容を退避する。次に、ステップＳ２３０４において、構成変更情報を参照して切り替え先の階級を取得する。切り替え先の階級とは、図２２において説明したような階級B^Tや階級A^Tのことである。ステップＳ２３０５において、切り替え先階級があったか否かを判断し、ない場合には、ステップＳ２３０６において、退避しておいたワークレジスタを回復し、ステップＳ２３０７において、現行階級を続行する。

【０１８２】

一方、切り替え先階級が存在した場合、ステップＳ２３０８において、演算プロセッサ１００１は、演算プロセッサ１００２～１００８に対して割り込みを発生させる。

【０１８３】
他方、演算プロセッサ１００２～１００８では、ステップＳ２３２１において、演算プロセッサ１００１からの割り込みが発生する。割り込みが発生した演算プロセッサ１００２～１００８は、ステップＳ２３２２において、スケジューラを起動し、ステップＳ２３２３において、以降の処理で用いるワークレジスタの内容を退避する。
【０１８４】
その後、ステップＳ２３３１において、演算プロセッサ１００１，１００２～１００８は、同期処理を行う。
【０１８５】
この同期処理が完了すると、ステップＳ２３０９において、演算プロセッサ１００１は、レジスタを退避し、ステップＳ２３１０において、現階級での同期範囲指定機構の状態を退避する。その後、ステップＳ２３１１において、切り替え先の階級に所属するジョブのレジスタを回復する。
【０１８６】
次に、ステップＳ２３１２において、新階級の同期範囲指定機構の状態を回復し、ステップＳ２３１３において、新階級でのジョブの仮想空間を設定し（１４０１７）、さらに、ステップＳ２３１４において、データ授受領域の内容を階級切り替えの反映したものに変更する。
【０１８７】
一方、演算プロセッサ１００２～１００８は、ステップＳ２３２４において、レジスタの退避を行った後、ステップＳ２３２５において、切り替え先階級のジョブのレジスタ内容を回復する。さらに、ステップＳ２３２６において、新ジョブの仮想空間の設定し、ステップＳ２３２７において、データ授受領域の設定の変更を行う。
【０１８８】
その後、ステップＳ２３３２において、演算プロセッサ１００１，１００２～１００８は、同期処理を行う。
【０１８９】
この同期処理が完了すると、ステップＳ２３１５において、演算プロセッサ１００１は、全ての演算プロセッサが新階級の実行の準備ができたことを確認した後、新階級の実行に移り、また、ステップＳ２３２８において、演算プロセッサ１００２～１００８は、新階級の実行に移る。
【０１９０】
本実施形態におけるジョブ階級切り替えの運用に当たっては次にように行っている。即ち、前述したように、計算機システムの最高速度を実現しようとすれば、全ての演算プロセッサをそのジョブに割り付ける必要が生じる。一方、最終的には最高速度を目指すジョブでも、デバッグの段階では計算機システム全てを占有する必要はなく、複数のデバッグ作業を並行して進行させることが望ましい場合や、１つのジョブの性能をある程度犠牲にしても、複数のジョブを並行に動作させ、計算機システム全体の稼働率を高めることが要求される場合もある。
【０１９１】
そこで、本実施形態においては、これらの要求に従って、並列ジョブは階級分けされる。計算機システム上では、構成変更情報設定フィールド２３０４（図４）に設定された構成変更情報に基づいて階級を切り替え、それぞれの階級に割り付けられた時間内で階級に属するジョブをその階級の実行方式に従って処理する。
【０１９２】
さらに、計算機システムは、各階級毎にスケジューリングのためのジョブスケジューリングキュー（図６）を備える。計算機上で起動されたジョブは自分の属する階級のジョブスケジューリングキューに入れられ、その階級の処理の順番が到来し次第、処理される。
【０１９３】
同一階級に属するジョブは、並列実行待ちキュー（図６）を用いて同一階級のプログラム

間で共有される並列演算プロセッサ群を切り替えて処理を進める。階級内でこれらのジョブがどのように処理されるかは、その階級の定義に従うものである。

【０１９４】

ここで、図２２を参照して、以上述べたジョブ階級切り替えの運用例について説明する。図２２に示した例では、階級Ａは全システムを占有する並列ジョブの階級とし、階級Ｂは４台の逐次演算プロセッサと４台の演算プロセッサから成る並列演算プロセッサ群を備えるシステム構成で実行するジョブの階級と定めるとする。図２２は、階級Ｂの実行から階級Ａの実行への切替えの様子を示している。

【０１９５】

図２２において、階級Ｂに属する４つの並列ジョブＪ１０，Ｊ２０，Ｊ４０，Ｊ５０を階級Ｂに割り付けられた計算時間Ｔの中で処理している。階級Ｂでは、４台の演算プロセッサを逐次演算プロセッサとして用い、同時に４つの並列ジョブを実行する。５つ以上のジョブが同時に起動されている場合は、５つ目以降のジョブは、階級Ｂのジョブスケジューリングキューの中で先行する４つのジョブの実行の終了を待つ。

【０１９６】

階級Ｂに割り当てられた時間Ｔの経過後に起動されたスケジューラＳＣＨ１０は、階級Ｂの実行の状態を退避し、他方の階級Ａの実行のためにシステム内のプロセッサ構成の変更と階級Ａに属するジョブの処理を開始する。

【０１９７】

階級Ａでは、同時には１つのプログラムしか処理しないので、同時に起動されている２つ目以降のジョブは、階級Ａのジョブスケジューリングキューの中で実行中のジョブの終了を待つ。

【０１９８】

同様にして、階級Ａに割り当てられた時間Ｔが経過すると、再びスケジューラＳＣＨ１１が起動され、階級Ａの実行状態を退避し、システム内の演算プロセッサの構成の変更を行い、前回退避した階級Ｂの実行の状態を回復し、再開する。

以降、同様にして交互に異なった階級を時分割で実行することになる。

【０１９９】

なお、以上の説明では、階級Ａと階級Ｂに割り付ける計算時間Ｔを等しくしたが、階級毎に異なった時間を割り付けることも可能である。また、階級Ａ，階級Ｂの２つの階級のみではなく、他の演算プロセッサの構成をもつ階級を加えることも可能である。

【０２００】

また、同一の演算プロセッサの構成で複数の階級を作成することも可能である。さらに、同一の演算プロセッサの構成で、異なる計算時間の割り当てられた階級や、時分割された時間の割り当ての優先度を異ならせたものを作成することも可能である。

【０２０１】

また、１階級内では、同時に処理され得るジョブ数の上限をその階級の逐次演算機構の数としたが、各階級が備えるジョブスケジューリングキューを用いて、それ以上の数のジョブを階級に割り振られた計算時間内で切り替えて実行することも可能である。

【０２０２】

なお、構成変更情報内に保持された時間Ｔと切り替えの契機の設定によって、例えば、昼間は前述の階級Ｂで運用し、夜間は階級Ａで運用するといった方式を実現することも可能である。

【０２０３】

次に、図２４を用いて、本実施形態による待ち状態の発生を契機とした計算機システム内の演算プロセッサの逐次演算プロセッサ及び並列演算プロセッサ群への割り付け個数の動的な変更，即ちスケジューリングについて説明する。

図２４は、本発明の一実施形態による並列ジョブ多重スケジューリング方法による待ち状態の発生を契機としたスケジューリングの説明図である。

【０２０４】

計算機システム内の演算プロセッサ１００１～１００８は、それぞれ、共有記憶装置２０００の中の管理情報２３００（図４）に、自らが待ち状態になっているか否かの情報を保持する待ち状態フラグ２３１０（図４）を有している。並列演算プロセッサ群を構成する複数の演算プロセッサの内の１台が待ち状態になると、待ち状態フラグ２３１０によって、並列演算プロセッサ群に参加する他の演算プロセッサ及び、逐次演算プロセッサが待ち状態になったか否かを調べる。ここで、全ての並列演算プロセッサ群中の全ての演算プロセッサ及び逐次演算プロセッサが待ち状態になったことが判明した場合、スケジューラは並列演算機構の割り当てを待ち状態となった現在のジョブから、並列実行待ちキューに入れられた他のジョブに切り替える。

【０２０５】

図２４に示す例において、計算機システムは、図９に示したように、８台の演算プロセッサから構成されているものとする。第１のジョブＪ６０と第２のジョブＪ７０は、並列的に実行されている並列ジョブである。並列ジョブＪ６０，Ｊ７０は、逐次演算プロセッサ１００１，１００２上でそれぞれ起動される。

【０２０６】

２つのジョブＪ６０，Ｊ７０が実行を開始し、ジョブＪ６０の処理Ｐ６０１及びジョブＪ７０の処理Ｐ７０１が実行される。ジョブＪ６０の処理Ｐ６０１が終了し、並列処理部分の処理Ｐ６０２～Ｐ６０６に達したとすると同期処理ＳＹＮ６を行って、演算プロセッサ１００１及び演算プロセッサ１００５～１００８を用いて、並列処理部分の処理Ｐ６０２～Ｐ６０６が実行される。

【０２０７】

一方、ジョブＪ６０の並列処理部分の処理Ｐ６０２～Ｐ６０６の実行中にジョブＪ７０が、並列処理部分の起動ＳＹＮ６を行うと、他の並行ジョブが実行中であるため、割り込みＩＮＴ６が発生する。この割り込みＩＮＴ６を契機として、共有記憶装置２０００内のスケジューラＳＣＨ６が起動され、並列処理要求ＲＥＱ６は並列実行待ちキューＣＵＥに入れられ、並列演算プロセッサ群が割り当てられるのを待つことになる。

【０２０８】

この状態から、並列演算プロセッサ群を構成する全ての演算プロセッサ１００５～１００８及び演算プロセッサ１００１が、ディスクの読み書き等と行うことにより、待ち状態ＷＡＩＴ１になったとすると、スケジューラＳＣＨ６が並列実行待ちキューＣＵＥの中から、ジョブＪ７０の並列実行要求れＱ６を取り出す。スケジューラＳＣＨ６の詳細な処理については、図２５を用いて後述する。

【０２０９】

並列実行待ち状態であったジョブＪ７０に並列演算プロセッサ群１００５～１００８が割り付けられ、ジョブＪ７０の並列処理部分の処理Ｐ７０２～Ｐ７０６の実行が開始される。

【０２１０】

待ち状態を生じた事象ＷＡＴ－ＲＥＬが終了し、ジョブＪ６０の待ち状態が解除されると、スケジューラＳＣＨ６は、実行中の並列ジョブＪ７０の終了を待って、再び、ジョブＪ６０に並列演算プロセッサ群を割り付け、待ち状態に引き続く処理Ｐ６０２～Ｐ６０６を再開する。ジョブＪ６０の並列処理部分の処理Ｐ６０２～Ｐ６０６が終了すると、演算プロセッサ１００１は、ジョブＪ６０の引き続く逐次処理部分の処理Ｐ６０７の実行を即座に開始する。

【０２１１】

また、ジョブＪ７０の並列処理部分の処理Ｐ７０２～Ｐ７０６が終了すると、演算プロセッサ１００２は、ジョブＪ７０の引き続く逐次処理部分の処理Ｐ７０７の実行を即座に開始する。

【０２１２】

以上が並列演算プロセッサ群上での待ち状態の発生を契機としたスケジューリングの方法である。

【０２１３】
次に、同じく、図２４を用いて、逐次演算プロセッサ上での待ち状態の発生を契機としたスケジューリングの方法について説明する。
逐次演算プロセッサ上で待ち状態になった場合は、その待ち状態になったジョブの並列処理部分が並列演算プロセッサ群上で処理されていなければ、他のジョブの逐次処理部分に逐次演算プロセッサを割り当て替える。
【０２１４】
即ち、図２４において、演算プロセッサ１００２上のジョブＪ７０がその逐次処理部分の処理Ｐ７０７の実行中に待ち状態ＷＡＩＴ２となったとし、且つこの時、並列演算プロセッサ群１００５～１００８上では、ジョブＪ７０の並列処理部分は実行されていないものとする。すると、演算プロセッサ１００２上では、スケジューラＳＣＨ７が、他のジョブの逐次処理部分への切り替えＣＨＡＮ１を行う。
【０２１５】
以上のようにして、逐次演算プロセッサ上での待ち状態発生を契機としてスケジューリングを行うことができる。
【０２１６】
ここで、図２５を用いて、本実施形態による待ち状態の発生を契機としたスケジューリング時の演算プロセッサの処理動作について説明する。
図２５は、本発明の一実施形態による並列ジョブ多重スケジューリング方法における待ち状態の発生によるスケジューリングの処理を説明するフローチャートである。
【０２１７】
図２５のステップＳ２５０１において、例えば、システムコールが発行され、ステップＳ２５０２において、待ち状態が発生する。次に、ステップＳ２５０３において、スケジューラＳＣＨが起動され、待ち状態フラグが設定される。
【０２１８】
ここで、ステップＳ２５０４において、待ち状態フラグが設定された演算プロセッサが同期範囲内であるか否かが判断され、同期範囲内であれば、ステップＳ２５０５において、同期範囲内の全ての演算プロセッサが待ち状態か否かが判断され、他の演算プロセッサが待ち状態にならなければ、ステップＳ２５０６において、そのまま続行する。並列実行待ちの他の同期範囲内の全ての演算プロセッサが待ち状態になったら、ステップＳ２５０７において、並列実行待ちの他のジョブに切り替わる。即ち、図２４に示す例では、ジョブＪ６０が待ち状態となり、ジョブＪ７０に切り替わる。
【０２１９】
また、ステップＳ２５０４における判断で、待ち状態になった演算プロセッサが同期範囲内でないとき、すなわち、並列処理部分を実行中でないジョブに割り当てられた逐次演算プロセッサならば、ステップＳ２５０８において、スケジューリングキュー内の他のジョブの逐次処理部分に切り替える。即ち、図２４に示す例では、ジョブＪ７０が待ち状態となり、他のジョブに切り替わる。
【０２２０】
次に、図２６を用いて、本実施形態による複数の並列演算プロセッサ群を備えた計算機システムの動作について説明する。
図２６は、本発明の一実施形態による並列ジョブ多重スケジューリング装置における複数の並列演算プロセッサ群を備えた計算機システムの動作の説明図である。
【０２２１】
本実施形態においては、計算機システムが、合計１２台の演算プロセッサ１００１～１０１２によって構成されているものとして説明する。演算プロセッサ１００１～１００４の４台は逐次演算プロセッサとして構成され、演算プロセッサ１００５～１００８の４台が第１の並列演算プロセッサ群として構成され、演算プロセッサ１００９～１０１２の４台が第２の並列演算プロセッサ群として構成されているものとする。
【０２２２】

４台の逐次演算プロセッサ１００１～１００４では、４つのジョブＪ１０，Ｊ２０，Ｊ８０，Ｊ９０が起動され、それぞれの処理Ｐ１０１，Ｐ２０１，Ｐ８０１，Ｐ９０１が実行される。最初に並列実行部分に達したジョブＪ８０が、最初に第１の並列演算プロセッサ群１００５～１００８を獲得し、並列処理部分の処理Ｐ２０２～Ｐ２０６の実行を開始する。次に並列処理部分に達したジョブＪ１０は、第２の並列演算プロセッサ群１００９～１０１２を獲得して、ジョブＪ８０の並列処理部分の処理Ｐ８０２～Ｐ８０６と並行して、自らの並列処理部分の処理Ｐ１０２～Ｐ１０６の実行を開始する。

【０２２３】

ここで、並列演算プロセッサ群が１つのみの場合には、同期範囲指定機構４１００中の同期範囲指定レジスタ４１１０のレジスタのビットを、図２において示したように、演算プロセッサに対応付けて同期範囲の指定及び逐次演算プロセッサの並列演算プロセッサ群の使用の許可・禁止を制御した。しかしながら、並列演算プロセッサ群が複数個用意された環境では、逐次演算プロセッサと並列演算プロセッサ群との組み合わせを指示・制御する同期範囲指定レジスタを全演算プロセッサ個数分のビット数だけ用意し、並列演算プロセッサが割り当てられている逐次演算プロセッサは、自らの同期範囲指定レジスタに自らの演算プロセッサ番号を設定し、並列演算プロセッサは自分が割り当てられている先の逐次演算プロセッサの演算プロセッサ番号を、同期範囲指定レジスタに設定する。並列演算プロセッサ群が割り付けられていない逐次演算プロセッサに対応するビットは”０”を設定する。自らの演算プロセッサ番号が設定されている演算プロセッサに並列演算プロセッサ群の使用が許可される。並列処理部分の起動時の同期処理では、このビットの値によって、同期処理を続行するか、割り込みを発生してスケジューラを起動し、並列実行待ちキューに登録するか否かが判断される。

【０２２４】

ここで、図２８を用いて、本実施形態による同期範囲指定レジスタの構成について説明する。

図２８は、本発明の一実施形態による複数の並列演算プロセッサ群を備えた計算機システムにおける同期範囲指定レジスタの構成図である。

【０２２５】

１２台の演算プロセッサから構成される計算機システムでは、１２ビットの同期範囲指定レジスタ３１１０Ａから構成されている。逐次演算プロセッサ１００２上のジョブＪ２０には、第２の並列演算プロセッサ群を構成する演算プロセッサ１００９～１０１２が割り当てられることを示すため、同期範囲指定レジスタ３１１０Ａのビット２，９～１２には、”２”が設定されている。また、逐次演算プロセッサ１００４上のジョブＪ９０には、第１の並列演算プロセッサ群を構成する演算プロセッサ１００５～１００８が割り当てられることを示すため、同期範囲指定レジスタ３１１０Ａのビット４，５～８には、”４”が設定されている。並列演算プロセッサ群の使用が禁止された演算プロセッサ１００１と演算プロセッサ１００３に対応する同期範囲指定レジスタ３１１０Ａのビット１，３には、”０”が設定されている。

【０２２６】

並列演算プロセッサ群１００１上のジョブＪ１０の並列処理部分の処理Ｐ１０２～Ｐ１０６の実行が完了すると、引き続いて起動されたジョブＪ２０の並列処理部分の処理Ｐ２０２～Ｐ２０６が、第２の並列演算プロセッサ群１００９～１０１２で処理される。一方、第１の並列演算プロセッサ群では、ジョブＪ８０の処理の終了後に起動されたジョブＪ９０の並列処理部分の処理Ｐ９０２～Ｐ９０６の実行が開始される。

【０２２７】

この処理の終了後、第２の並列演算プロセッサ群では、ジョブＪ８０の並列処理部分の処理Ｐ８０８～Ｐ８１２の実行が開始される。ジョブＪ８０の１つ前の並列処理部分の処理Ｐ８０２～Ｐ８０６は、第１の並列演算プロセッサ群で実行されたが、同一のジョブ中の並列処理部分に前回の並列処理部分が実行された並列演算プロセッサ群とは異なる並列演算プロセッサ群が割り当てられることも有り得る。

【０２２８】
次に、図２７を用いて、演算プロセッサ間の同期処理をソフトウェア的に実現する方法について説明する。
図２７は、本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるソフトウエア的同期処理を説明するフローチャートである。
【０２２９】
上述した例では、演算プロセッサ間の同期処理を専用のハードウェア機構を用いていたが、本実施形態においては、その機能をソフトウェア的に実現するようにしている。
【０２３０】
演算プロセッサは、バリア同期機構を保持する代わりに、共有記憶装置２０００上に、各演算プロセッサに対応した同期到達フラグ２３０９（図４）を備える。同期ポイントに達した演算プロセッサは、同期到達フラグ２３０９を設定することで同期要求を行う。全ての演算プロセッサの同期到達フラグ２３０９が設定された時点で同期処理が完了したことをソフトウェアによって検知する。
【０２３１】
同期範囲指定機構４１００の代わりに、共有記憶装置２０００上に、同期範囲指定フィールド２３０７（図４）を備え、同期範囲内の演算プロセッサを同期範囲指定フィールド２３０７に登録することで、同期範囲の指定をソフトウェアで行う。各演算プロセッサは同期要求に際して、同期範囲指定フィールド２３０７の内容によって自らが同期の範囲内か否かを判定し、同期範囲内であれば、そのまま同期処理を続行し、同期範囲外であれば自らを実行する演算プロセッサ上に割り込みを発生させる。さらに、同期範囲内と設定されていた演算プロセッサが同期要求を行い、要求後で同期成立前に、プログラムにて同期範囲指定機構に対して該演算プロセッサを同期範囲外と設定した場合は、同設定を行った演算プロセッサが、新たに同期範囲外に指定された演算プロセッサに対して割り込みを発生させることによって、ハードウェアによって実現していた同期範囲指定機構４１００をソフトウェアによって実現することができる。
【０２３２】
また、同期タイムアウト機構もソフトウェア的に実現可能である。同期処理に到達した演算プロセッサは、同期範囲内の他の演算プロセッサが同期に達するまで無限に待つのではなく、一定の時間が経過しても同期に達しない演算プロセッサが存在した場合、同期処理の失敗として、同期範囲内の演算プロセッサに割り込みを発生させスケジューラを起動してエラー処理を行う。
【０２３３】
図２７のステップＳ２８０１において、同期範囲内の演算プロセッサを同期範囲外に設定し直すときは、演算プロセッサが同期フラグを設定しているか否かを調べる。設定していなければ、ステップＳ２８０２において、そのまま同期範囲外に設定する。一方、同期フラグが設定されていれば、ステップＳ２８０３において、演算プロセッサ間割り込みを同演算プロセッサに発生させる。割り込みを受けた演算プロセッサは、ステップＳ２８０４において、スケジューラを起動し、同演算プロセッサ上で実行されているジョブを並列実行待ちキューに登録すると同時に同演算プロセッサを同期範囲外に設定する。
【０２３４】
次に、図２９を用いて、終了通知を動的な判断によって発行する方法について説明する。
図２９は、本発明の一実施形態による並列ジョブ多重スケジューリング方法における終了通知の動的判断処理を示すフローチャートである。
【０２３５】
静的な終了通知の発行は、コンパイラによって自動的に引き続く逐次処理部分の実行時間の解析を行い、終了通知を発行することにより行われる。
【０２３６】
本方式の適用の可否は、１並列処理の終了時にどの程度の頻度で終了通知を発行し、退避・回復のオーバヘッドが低い並列演算プロセッサ群上の実行ジョブの切り替えが行えるか

に依存するものである。少しでも多くの場合に終了通知が発行されるように、ユーザは並列処理の終了時に引き続く逐次処理部分が長いと予めプログラムの構造として明らかな時は、並列演算プロセッサ群の割り当て変更を促進するために、動的に終了通知を発行する命令を実行することができる。

【０２３７】

本実施形態においては、実行時に並列処理部分と逐次処理部分の実行時間を、共有記憶装置上のジョブ管理情報２４００中の先行逐次情報フィールド２４０５（図５）に記録し、そのデータに基づいて統計的な解析を加えて、確率的に１並列処理部分に引き続く逐次処理部分の実行時間が一定時間よりも長いと判断されたときは、終了通知をスケジューラが自動的に発行するといった、実行時の動的な解析に基づいた処理を行う。

【０２３８】

図２９のステップＳ２９０１において、先行逐次処理部分の実行時間を共有記憶装置２０００中のジョブ管理情報２４００中に新たに備える先行逐次情報フィールド２４０５（図５）に保存する。次に、ステップＳ２９０２において、演算プロセッサは、並列処理部分の処理を実行する。

【０２３９】

ステップＳ２９０３において、スケジューラは、並列処理部分の終了時には、先行逐次情報フィールド２４０５の内容に基づいて後続逐次処理部分の実行時間が一定時間より長いか否かを判定する。長いと判定された場合には、ステップＳ２９０４において、終了通知を行い、短いと判定された場合には、ステップＳ２９０５において、終了通知は行わないようにする。

【０２４０】

なお、１並列ジョブの実行開始から終了までの全ての逐次処理部分の実行時間を記録しなくてもよい。ある定数Ａを定め、過去Ａ回分の逐次処理部分の実行時間の履歴を保持し、これに基づいて判定を行うことも可能である。例えば、並列処理部分が繰り返し構造の中に存在する場合、実行時には同並列処理部分が繰り返し回数分周期的に出現する。ここで、仮に、同並列処理部分間の逐次処理部分の実行時間が一定時間よりも長いにも拘わらず、終結時に終了通知を発するコードが生成されていなかったとする。この時、繰り返しで実行される並列処理部分は過去Ａ回の履歴を参照することにより、終結時に終了通知を発行することが可能になる。

【０２４１】

本実施形態によれば、コンパイラによる静的な解析による終了通知を発行が難しい場合にも、この動的な終了通知の発行によって、並列演算プロセッサ群の割り当て変更が可能となる。

【０２４２】

次に、図３０及び図３１を用いて、本実施形態における実行ジョブが、唯一つのときの実行方法について説明する。

図３０は、本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるジョブが単一で実行されているときのレジスタ退避時の動作を示すフローチャートであり、図３１は、本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるジョブが単一で実行されているときのレジスタ回復時の動作を示すフローチャートである。

【０２４３】

１個のジョブのみが実行されている状態では、すべての並列ジョブの多重実行のための処理を省くことが可能になる。これまでの説明では、並列処理部分の起動と終結においては、レジスタ退避範囲フラグ２４０８（図５）を設定することによって、最小限のレジスタの退避・回復を行うものとして説明している。しかし、計算機システム内で実行されるジョブが１つである場合は、この最小限のレジスタの退避・回復も不要である。

【０２４４】

実行されるジョブが１つである状態を検出するために、計算機システム内にジョブ単一実行フラグ２３０８（図４）を備え、ジョブの起動及び終結に伴って、実行中のジョブの数

が１となった場合は、スケジューラがジョブ単一実行フラグを設定する。ジョブ単一実行フラグを参照し、起動中のジョブ数が１であるときは、並列演算プロセッサ群のレジスタの内容の退避・回復を行わず、並列ジョブの多重実行のための一切のオーバヘッドを生じさせずに、並列演算プロセッサ群のスケジューリングを可能にする。

【０２４５】

最初に、図３０に基づいて、レジスタの退避処理について説明する。

終了通知に引き続いて、ステップＳ３００１において、スケジューラは、並列演算プロセッサ群上のジョブを切り替えるときに、ジョブ単一実行フラグ２３０８（図４）が設定されているかを調べる。設定されていれば、ステップＳ３００２において、レジスタの退避は行わなず、ステップＳ３００３において、レジスタが未退避であることをジョブ管理情報中のレジスタ未退避フラグ２４０７（図５）を設定することで記録する。

【０２４６】

単一実行フラグが設定されていないときは、ステップＳ３００４において、規定の最小限のレジスタのみを退避し、さらに、ステップＳ３００５において、最小限のレジスタのみが退避されたことをレジスタ退避範囲フラグ２４０８（図５）を設定して記録する。

【０２４７】

次に、図３１をに基づいて、レジスタの回復処理について説明する。

ステップＳ３１０１において、スケジューラは、ジョブ単一実行フラグ２３０８（図４）が設定されているかを調べる。設定されていれば、ステップＳ３１０２において、レジスタの回復は行わなず、ステップＳ３１０３において、ジョブ管理情報中のレジスタ未退避フラグ２４０７（図５）を解除する。

【０２４８】

単一実行フラグが設定されていないときは、ステップＳ３１０４において、切り替え先の新ジョブのレジスタ退避範囲フラグ２４０８（図５）が設定されているかを調べる。設定されていれば、ステップＳ３１０５において、必要なレジスタのみ回復し、さらに、ステップＳ３１０６において、新ジョブのレジスタ退避範囲フラグ２４０８を解除する。設定されていなければ、ステップＳ３１０７において、全てのレジスタを回復する。

【０２４９】

以上説明したように、本実施形態によれば、複数のジョブが同時に並列処理部分を実行しなければ、１ジョブの逐次処理部分の実行中に遊休する並列演算プロセッサ群を他のジョブの並列処理部分の実行に割り当てることができ、複数のジョブを並列的に実行できるため、計算機システム内の全プロセッサの稼働率を高めることができる。

【０２５０】

また、並列処理部分を実行する並列演算プロセッサ群は、従来の並列専用機構とは異なり、逐次演算プロセッサと同等の複数個の演算プロセッサによって論理的に構成しているため、計算機システム内の演算プロセッサの逐次及び並列演算プロセッサへの割り当て比率を動的に変更することができる。従って、実行対象ジョブの逐次処理と並列処理の比率に応じて計算機システム内の演算プロセッサの構成比率を変更することができるので、計算機システムの稼動率を高めることが可能となる。

【０２５１】

また、同期範囲指定機構を用いて、同期範囲内の演算プロセッサによる並列処理を実行し、同期範囲外の演算プロセッサからの同期要求に対して、要求元の演算プロセッサ上に割り込みを発生させ、また、同期成立前に同期範囲外に設定し直された演算プロセッサ上に割り込みを発生させてスケジューラを起動して待ち状態にし、起動要求を並列実行待ちキューに入れるようにしているため、１時期には、１つの逐次演算プロセッサに対して並列演算プロセッサ群を容易に割り当てることができる。

【０２５２】

また、並列処理部分の起動時に実行される同期到達通知命令が自らを実行した演算プロセッサに対応する同期範囲指定機構の中のレジスタに対応するビットを調べることで、容易に、自分が並列演算プロセッサ群の使用が可能か否かの場合分けに従った処理を行うこと

によって、排他的に並列演算プロセッサ群を割り付けることが可能である。

【０２５３】

また、並列処理部分の処理の終結時には、並列演算プロセッサ群は終了通知を発行し、これを契機にして他の並列実行待ちジョブの並列処理部分の実行に切り替える。この際、並列処理部分の実行が終了しているため、並列演算プロセッサのレジスタの退避が大幅に削減されることによってオーバーヘッドの少ない切り替えを実現することができる。従って、ジョブの変更を高速に行うことができる。

【０２５４】

また、後続する逐次処理部分の実行時間が一定時間より長い場合にのみ終了通知を発行するようにしているため、１並列ジョブの逐次処理時間が、他の並列ジョブの並列処理部分の実行に十分な長さをもっている時のみ、並列演算プロセッサの切り替えが行われ、並列演算プロセッサ群の遊休時間に、他の並列ジョブの並列処理部分を割り当てることが可能になるため、スケジューリングのオーバーヘッドを少なくして、計算機システム全体の効率を向上できる。

【０２５５】

また、スケジューラは並列プロセッサ群上のみで起動されて、ジョブ切り替えの処理を行うようにしているため、この処理は、他のジョブの逐次処理部分の処理の実行に影響を及ぼすことなく行われるため、ジョブ切り替えのオーバヘッドを隠蔽できる。

【０２５６】

また、並列処理部分の終了を並列演算プロセッサ群の割り当て替えの契機として用いることによって、切りのよいタイミングでジョブを切り替え、退避・回復するレジスタの数を大幅に削減することが可能になり、ジョブ切り替えのオーバヘッドを低減できる。

【０２５７】

また、同期タイムアウトレジスタを用いて、予め指定した時間内に同期ポイントに達しない並列演算プロセッサが存在したとき、その同期処理は失敗したものとして、同期失敗を知らせる割り込みが演算プロセッサ上に発生し、これを契機にスケジューラが起動され、エラー処理を行える。

【０２５８】

また、ジョブの切り替えに際しては、演算プロセッサ間データ授受領域を実記憶空間上に用意し、これらを演算プロセッサ間データ授受領域対応つけフィールドを用いて同一の仮想記憶空間上の番地に対応づけ、ジョブの切り替えに際して、両者の対応づけを変更することによって、演算プロセッサ間データ授受領域の状態を退避・回復する必要がなくなり、並列演算機構の割り当て変更を高速に行うことができる。

【０２５９】

また、並列ジョブは階級分けされ、構成変更情報設定フィールドに設定された構成変更情報に基づいて階級を切り替え、それぞれの階級に割り付けられた時間内で階級に属するジョブをその階級の実行方式に従って処理することができる。

【０２６０】

また、各階級毎にスケジューリングのためのジョブスケジューリングキューを備えており、ジョブは自分の属する階級のジョブスケジューリングキューに入れるようにしているため、各階級毎に、処理の順番に応じた処理が行える。

【０２６１】

また、同一階級に属するジョブは、並列実行待ちキューを用いて同一階級のプログラム間で共有される並列演算プロセッサ群を切り替えて処理を進めることができる。

【０２６２】

また、並列演算プロセッサ群上での待ち状態の発生を契機として、ジョブ切り替えのためのスケジューリングを実行できる。

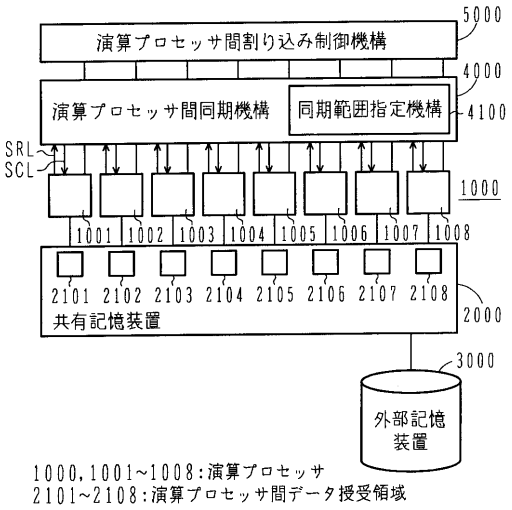
【０２６３】

また、複数の並列演算プロセッサ群を備えることにより、複数の並列処理を並行して実行できる。

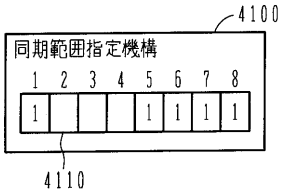
【０２６４】
また、演算プロセッサ間の同期処理は、ソフトウェア的に実現することも可能である。
【０２６５】
また、並列処理の終了時に引き続く逐次処理部分が長いと予めプログラムの構造として明らかな時には、動的な終了通知の発行によって、並列演算プロセッサ群の割り当て変更ができる。
【０２６６】
【発明の効果】
本発明によれば、並列ジョブ多重スケジューリング方法及び装置における複数の並列化されたジョブの同時実行時に、システム内の全プロセッサの稼働率を向上することができる。
【図面の簡単な説明】
【図１】　本発明の一実施形態による計算機システムの全体構成図である。
【図２】　本発明の一実施形態による同期範囲指定機構の構成図である。
【図３】　本発明の一実施形態による共有記憶装置中に用意されるデータの説明図である。
【図４】　本発明の一実施形態における共有記憶装置上の管理情報の構成図である。
【図５】　本発明の一実施形態において用いるジョブ管理情報の構成図である。
【図６】　本発明の一実施形態において用いる階級管理情報の構成図である。
【図７】　本発明の一実施形態によって並列ジョブを複数の演算プロセッサを用いて実行する場合の１方式の概略図である。
【図８】　本発明の一実施形態による並列ジョブの多重実行の概略説明図である。
【図９】　本発明の一実施形態による並列ジョブ多重スケジューリング装置のシステム構成図である。
【図１０】　本発明の一実施形態による並列ジョブ多重スケジューリング方法における並列処理部分の起動・終結時の実行について説明するフローチャートである。
【図１１】　本発明の一実施形態による並列ジョブの多重実行の概略説明図である。
【図１２】　本発明の一実施形態による並列ジョブ多重スケジューリング方法における並列演算プロセッサ群のプログラムへの割り当て制御方式の説明図である。
【図１３】　本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるデータ授受領域の管理方法の説明図である。
【図１４】　本発明の一実施形態による図９で示した並列演算プロセッサ群上で起動されるスケジューラＳＣＨ２（図１２）の全体の動作の概略を示すフローチャートである。
【図１５】　図１４に示した処理の細部を説明するフローチャートである。
【図１６】　図１４に示した処理の細部を説明するフローチャートである。
【図１７】　図１４に示した処理の細部を説明するフローチャートである。
【図１８】　図１４に示した処理の細部を説明するフローチャートである。
【図１９】　図１４に示した処理の細部を説明するフローチャートである。
【図２０】　図１４に示した処理の細部を説明するフローチャートである。
【図２１】　図１４に示した処理の細部を説明するフローチャートである。
【図２２】　本発明の一実施形態による並列ジョブ多重スケジューリング方法により演算プロセッサの割り付けを動的変更する例の説明図である。
【図２３】　本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるジョブ階級切り替え時のスケジューラの動作を示すフローチャートである。
【図２４】　本発明の一実施形態による並列ジョブ多重スケジューリング方法による待ち状態の発生を契機としたスケジューリングの説明図である。
【図２５】　本発明の一実施形態による並列ジョブ多重スケジューリング方法における待ち状態の発生によるスケジューリングの処理を説明するフローチャートである。
【図２６】　本発明の一実施形態による並列ジョブ多重スケジューリング装置における複数の並列演算プロセッサ群を備えた計算機システムの動作の説明図である。

【図２７】　本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるソフトウエア的同期処理を説明するフローチャートである。
【図２８】　本発明の一実施形態による複数の並列演算プロセッサ群を備えた計算機システムにおける同期範囲指定レジスタの構成図である。
【図２９】　本発明の一実施形態による並列ジョブ多重スケジューリング方法における終了通知の動的判断処理を示すフローチャートである。
【図３０】　本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるジョブが単一で実行されているときのレジスタ退避時の動作を示すフローチャートである。
【図３１】　本発明の一実施形態による並列ジョブ多重スケジューリング方法におけるジョブが単一で実行されているときのレジスタ回復時の動作を示すフローチャートである。

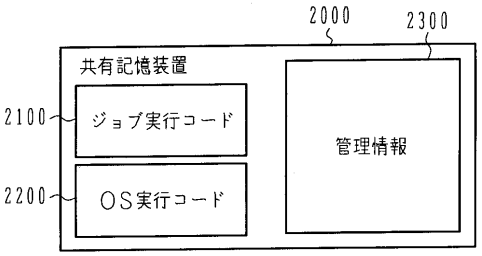
【図１】



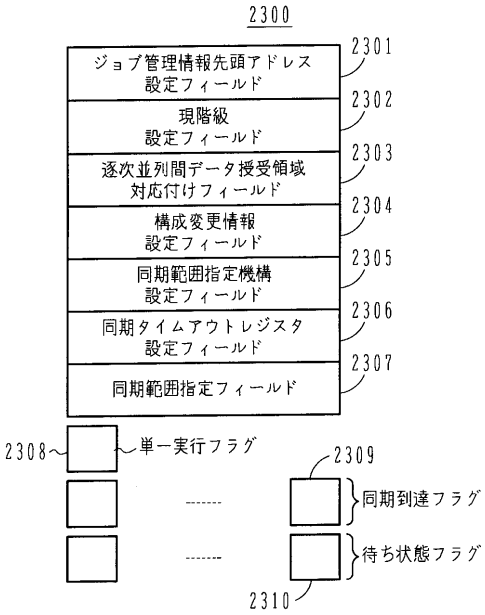
【図２】



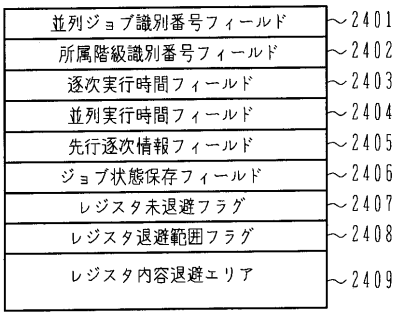
【図３】



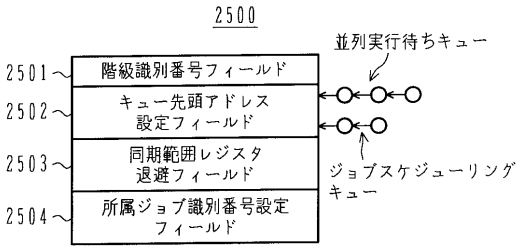
【図４】


2300
ジョブ管理情報先頭アドレス設定フィールド
2301
現階級設定フィールド
2302
逐次並列間データ授受領域対応付けフィールド
2303
構成変更情報設定フィールド
2304
同期範囲指定機構設定フィールド
2305
同期タイムアウトレジスタ設定フィールド
2306
同期範囲指定フィールド
2307
2308
単一実行フラグ
2309
同期到達フラグ
待ち状態フラグ
2310

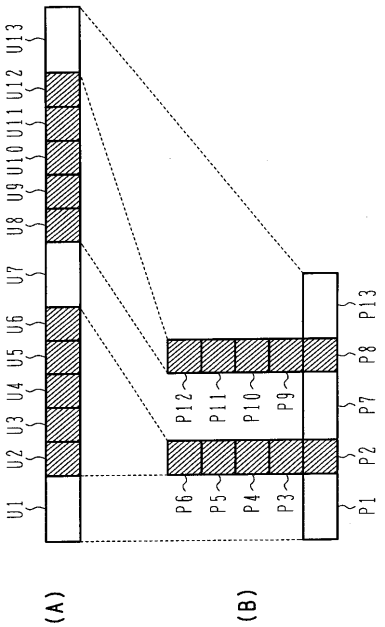
【図５】


並列ジョブ識別番号フィールド
2401
所属階級識別番号フィールド
2402
逐次実行時間フィールド
2403
並列実行時間フィールド
2404
先行逐次情報フィールド
2405
ジョブ状態保存フィールド
2406
レジスタ未退避フラグ
2407
レジスタ退避範囲フラグ
2408
レジスタ内容退避エリア
2409

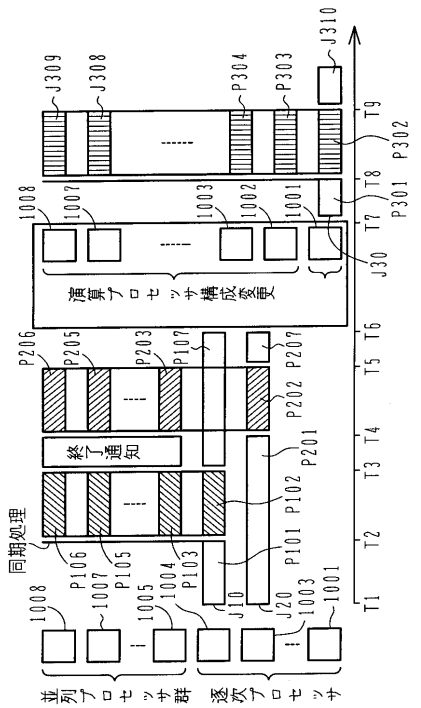
【図６】


2500
並列実行待ちキュー
2501
階級識別番号フィールド
2502
キュー先頭アドレス設定フィールド
2503
同期範囲レジスタ退避フィールド
2504
所属ジョブ識別番号設定フィールド
ジョブスケジューリングキュー

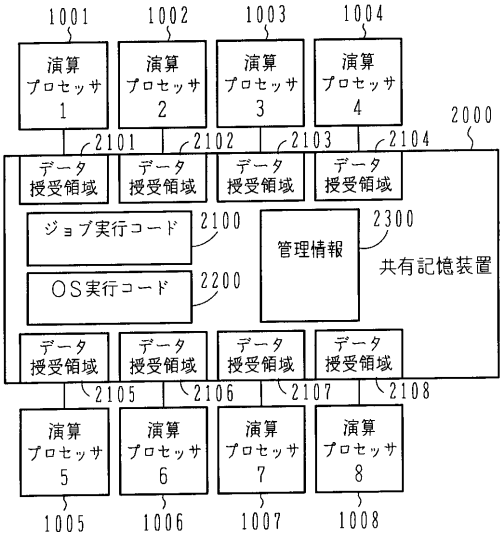
【図７】


U1
U2
U3
U4
U5
U6
U7
U8
U9
U10
U11
U12
U13
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
(A)
(B)

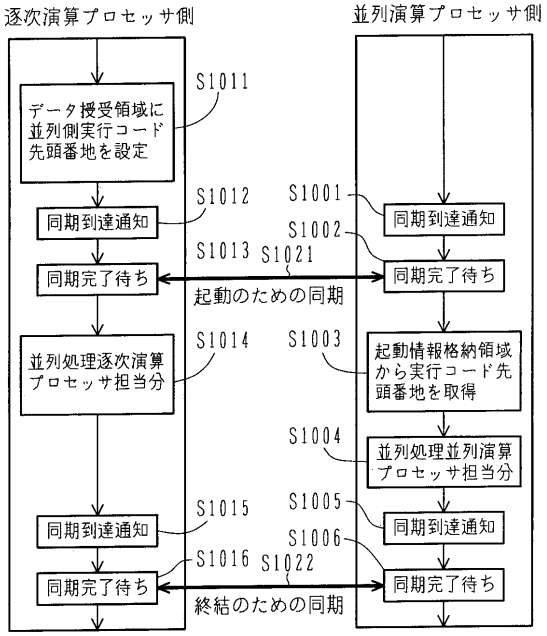
【図８】


J310
J309
J308
P304
P303
T9
T8
T7
T6
T5
T4
T3
T2
T1
P302
P301
1008
1007
1003
1002
1001
J30
演算プロセッサ構成変更
P206
P205
P203
P107
P207
P202
P201
P102
P101
終了通知
同期処理
P106
P105
P104
P103
J10
J20
1005
1004
並列プロセッサ群
逐次プロセッサ

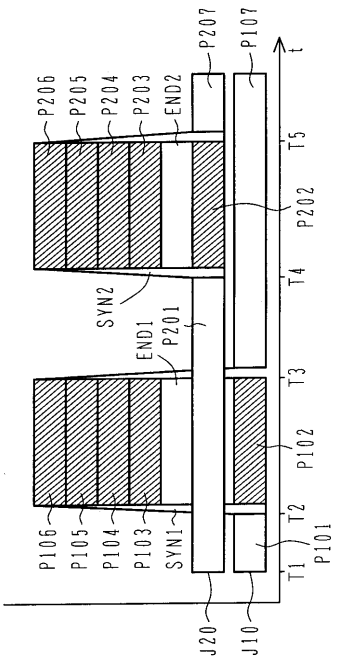
【図９】



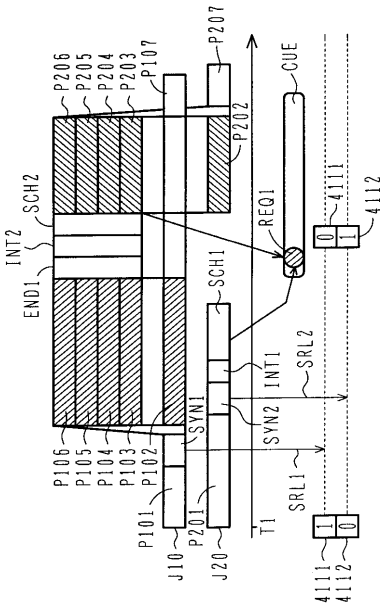
【図１０】



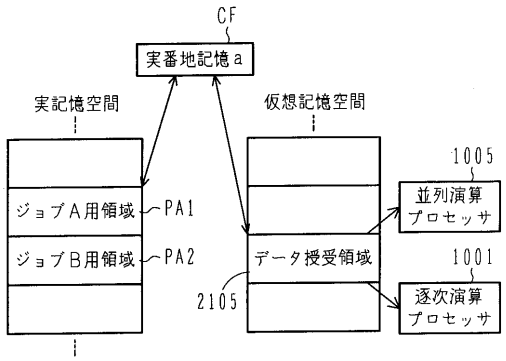
【図１１】



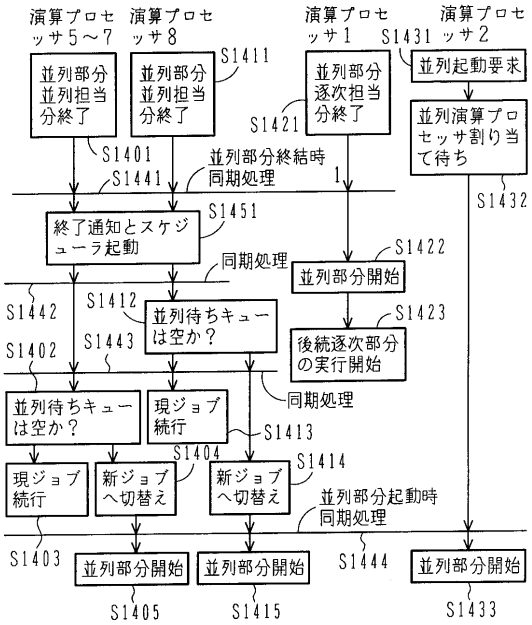
【図１２】



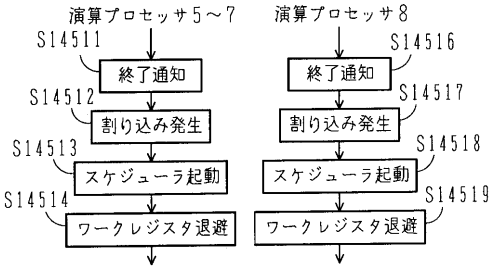
【図１３】



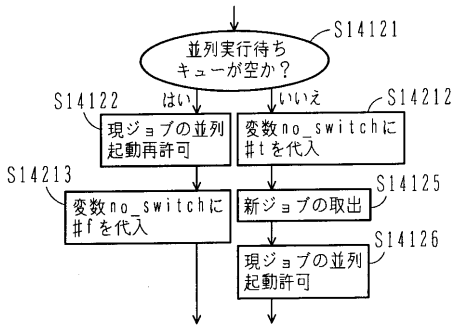
【図１４】



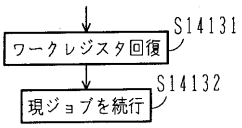
【図１５】



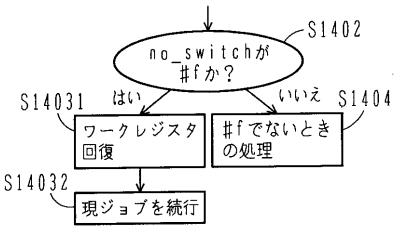
【図１６】



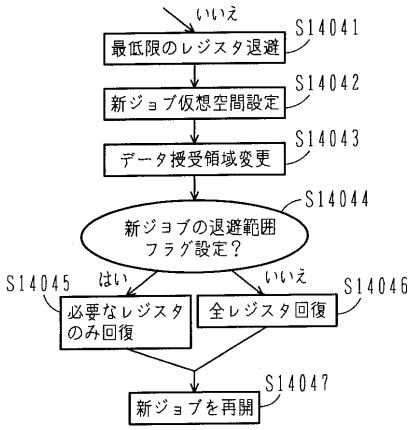
【図１７】



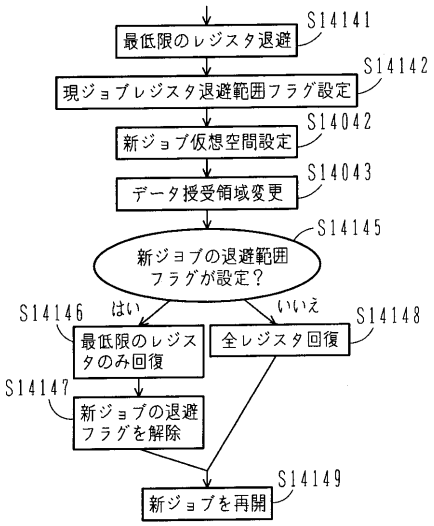
【図１８】



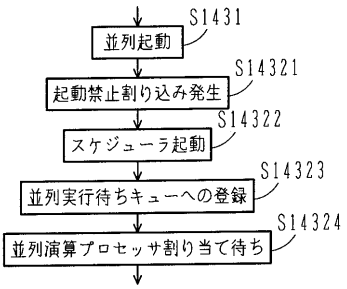
【図１９】



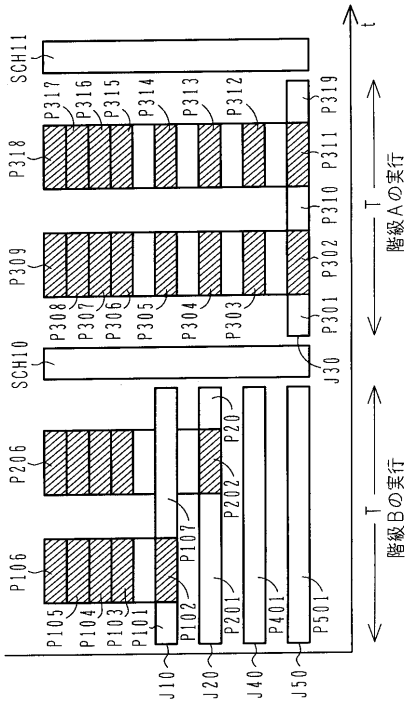
【図２０】



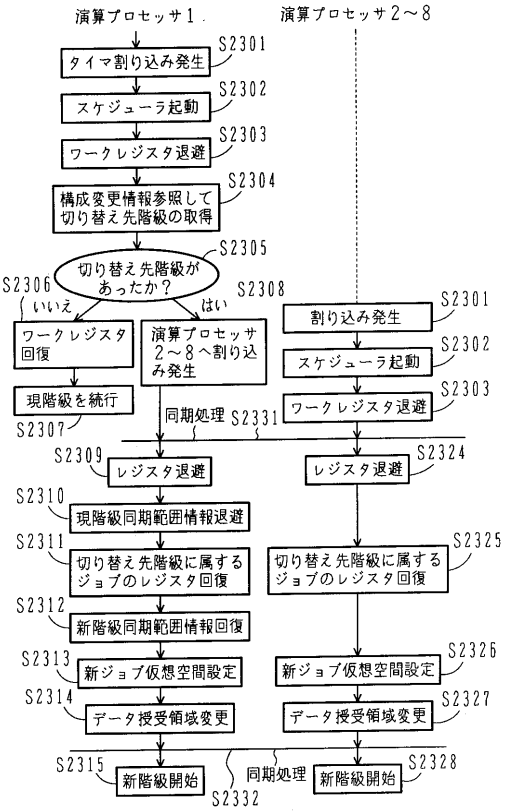
【図２１】



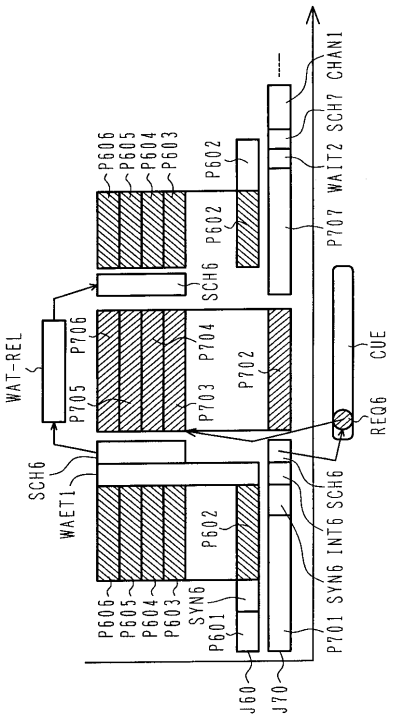
【図２２】



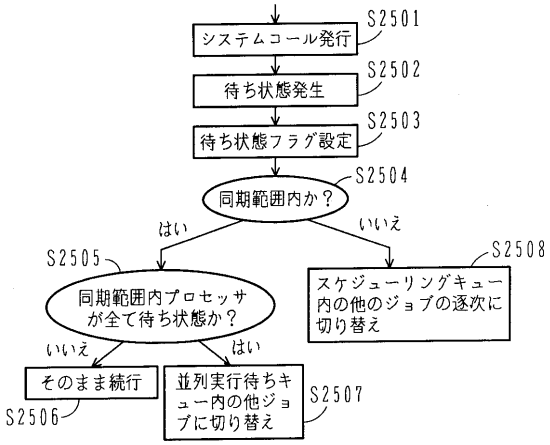
【図２３】



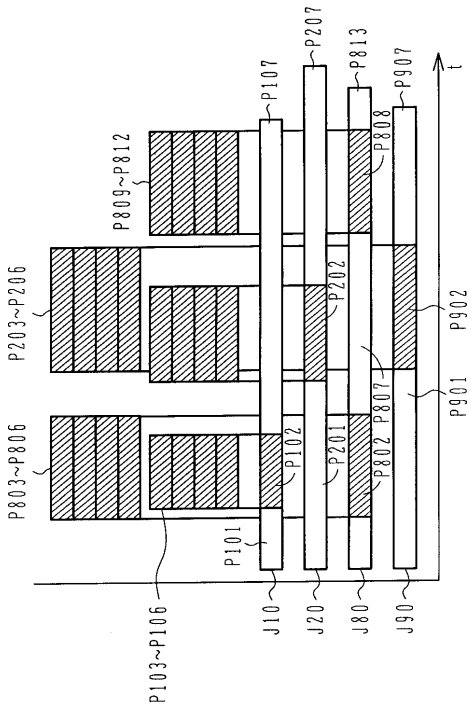
【図２４】



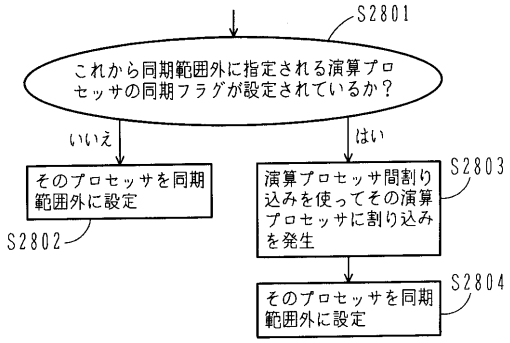
【図２５】



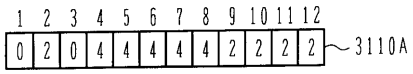
【図２６】



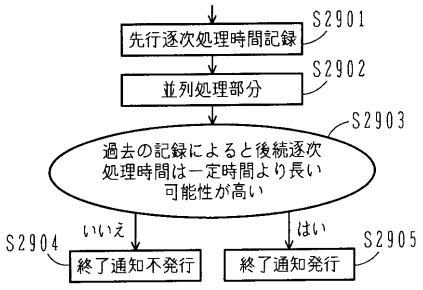
【図２７】



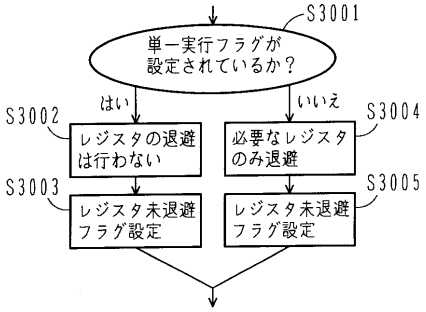
【図２８】



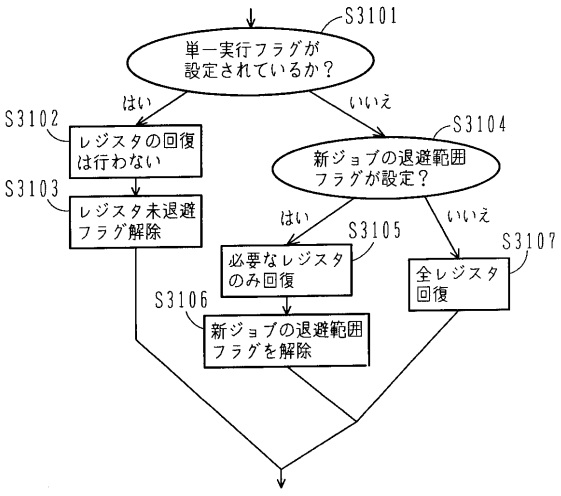
【図２９】



【図３０】



【図３１】



フロントページの続き

(72)発明者 熊崎 裕之
神奈川県横浜市戸塚区戸塚町５０３０番地 株式会社 日立製作所 ソフトウェア開発本部内
(72)発明者 助川 直伸
東京都国分寺市東恋ケ窪一丁目２８０番地 株式会社 日立製作所 中央研究所内
(72)発明者 中島 恵
神奈川県横浜市戸塚区戸塚町５０３０番地 株式会社 日立製作所 ソフトウェア開発本部内
(72)発明者 深川 正一
神奈川県秦野市堀山下１番地 株式会社 日立製作所 汎用コンピュータ事業部内

審査官 殿川 雅也

(56)参考文献 特開平０２－１４４６５７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/46- 9/54
G06F 15/16- 15/177