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

特許第6581727号
(P6581727)

(45)発行日　令和1年9月25日(2019.9.25)　(24)登録日　令和1年9月6日(2019.9.6)

(51)Int.Cl.　F I
G06F 9/54 (2006.01)　G06F 9/54 B
G06F 9/50 (2006.01)　G06F 9/50 １５０Ｅ

請求項の数 30　(全 34 頁)

(21)出願番号　特願2018-529031(P2018-529031)
(86)(22)出願日　平成28年12月5日(2016.12.5)
(65)公表番号　特表2019-502989(P2019-502989A)
(43)公表日　平成31年1月31日(2019.1.31)
(86)国際出願番号　PCT/US2016/064898
(87)国際公開番号　WO2017/105888
(87)国際公開日　平成29年6月22日(2017.6.22)
審査請求日　平成30年7月25日(2018.7.25)
(31)優先権主張番号　62/268,762
(32)優先日　平成27年12月17日(2015.12.17)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　509123208
アビニシオ　テクノロジー　エルエルシー
アメリカ合衆国　０２４２１　マサチューセッツ州　レキシントン　スプリング　ストリート　２０１
(74)代理人　100107984
弁理士　廣田　雅紀
(74)代理人　100102255
弁理士　小澤　誠次
(74)代理人　100096482
弁理士　東海　裕作
(74)代理人　100188352
弁理士　松田　一弘
(74)代理人　100113860
弁理士　松橋　泰典

最終頁に続く

(54)【発明の名称】動的なパーティショニングを使用してデータを処理すること

(57)【特許請求の範囲】
【請求項１】
保留キュー、再配置キュー、及び準備完了キューを含むワークユニットを記憶するキューを使用して、コンピューティングシステムにおいてデータを処理するための方法であって、
前記コンピューティングシステムの１又は２以上のノードのストレージに記憶された複数の保留キューにワークユニットを受信するステップと、
前記１又は２以上のノードの各々における少なくとも１つのプロセッサを使用して、前記複数の保留キューから前記ワークユニットを処理するステップとを含み、前記処理するステップが、
前記保留キューごとに、その保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定するステップであって、各パーティションが一意の保留キューに関連付けられる、前記決定するステップと、
各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離するステップであって、前記２つの他のキューが、ワークユニットが記憶されている前記保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている前記保留キューに関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離するステップと、

各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられた前記パーティションに関連付けられる前記保留キューに再配置するステップと、
前記複数の保留キューが空になる前に、前記パーティショニングルールの新しいパーティショニングルールへの変更に応答して、前記ワークユニットを再パーティショニングするステップと
を含む、前記方法。
【請求項２】
パーティショニングルールの変更が、複数のパーティション内のパーティションの総数の変更を含む、請求項１に記載の方法。
【請求項３】
コンピューティングシステムの１又は２以上のノードが複数のノードを含み、前記複数のノードのうちの少なくとも第１のノードが、第１のパーティションに関連付けられる第１の保留キューを記憶し、前記第１の保留キューに対応する第１の再配置キュー及び第１の準備完了キューを記憶するストレージを有し、前記複数のノードのうちの少なくとも第２のノードが、第２のパーティションに関連付けられる第２の保留キューを記憶し、前記第２の保留キューに対応する第２の再配置キュー及び第２の準備完了キューを記憶するストレージを有する、請求項１又は２に記載の方法。
【請求項４】
パーティショニングルールの変更が、保留キューのうちのいずれかを記憶するために使用されるストレージを有する複数のノード内のノードの総数の変更を含む、請求項３に記載の方法。
【請求項５】
第１の再配置キュー内に１又は２以上のワークユニットを再配置するステップが、第１のノードから第２のノードに前記１又は２以上のワークユニットを含むメッセージを送信するステップを含み、第２のパーティションが、前記メッセージに含まれる前記１又は２以上のワークユニットの各々に割り当てられる、請求項３又は４に記載の方法。
【請求項６】
第１のノードのストレージがまた、第３のパーティションに関連付けられる第３の保留キューを記憶し、前記第３の保留キューに対応する第３の再配置キュー及び第３の準備完了キューを記憶する、請求項３～５のいずれかに記載の方法。
【請求項７】
再パーティショニングするステップが、特定の保留キューに対応する再配置キュー及び準備完了キューに記憶された任意のワークユニットを、その特定の保留キューに戻すステップを含む、請求項１に記載の方法。
【請求項８】
パーティショニングルールが、各ワークユニットに対して複数のパーティションのうちの１つのコンテンツに依存する割当てを使用し、特定のワークユニットに割り当てられた前記パーティションが、前記ワークユニットに含まれる少なくとも１つの値に基づいて決定される、請求項７に記載の方法。
【請求項９】
各保留キューが、初期パーティション決定がまだ行われていないワークユニットを記憶するための第１の部分と、それらのワークユニットをその保留キューに再配置させる初期パーティション決定が行われたワークユニットを記憶するための第２の部分とを含む、請求項７又は８に記載の方法。
【請求項１０】
ワークユニットを各保留キューから対応する再配置キュー又は準備完了キューに分離するステップが、第２の部分が空である場合にのみ第１の部分に記憶されたワークユニットを分離するステップを含む、請求項９に記載の方法。
【請求項１１】
少なくとも第１の保留キューが、パーティショニングルールの変更後にその保留キュー

に戻された任意のワークユニットを記憶するための第３の部分を含む、請求項９又は１０に記載の方法。

【請求項１２】

再パーティショニングするステップが、

第１の保留キューに対して、新しいパーティショニングルールに従って前記第１の保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを決定するステップと、

ワークユニットを前記第１の保留キューから対応する再配置キュー又は準備完了キューに分離するステップと

をさらに含む、請求項１１に記載の方法。

【請求項１３】

ワークユニットを第１の保留キューから対応する再配置キュー又は準備完了キューに分離するステップが、第３の部分が空である場合にのみ第２の部分に記憶されたワークユニットを分離するステップと、前記第２の部分と前記第３の部分の両方が空である場合にのみ第１の部分に記憶されたワークユニットを分離するステップとを含む、請求項１２に記載の方法。

【請求項１４】

パーティショニングルールが、複数のパーティションのうちの１つのコンテンツに依存しない割当てを各ワークユニットに使用し、特定のワークユニットに割り当てられた前記パーティションが、前記ワークユニットに含まれる任意の値から独立して決定される、請求項１に記載の方法。

【請求項１５】

パーティショニングルールが、複数のパーティションのうちの１つの負荷バランシング割当てを各ワークユニットに使用し、特定のワークユニットに割り当てられた前記パーティションが、異なるパーティションに対する負荷の定量的尺度に基づいて決定される、請求項１４に記載の方法。

【請求項１６】

異なるパーティションに対する負荷の定量的尺度が、異なるパーティションに関連付けられる保留キューに再配置されるべき記憶されたワークユニットの異なるセットのバックログの定量的尺度である、請求項１５に記載の方法。

【請求項１７】

第１のノード上で実行する１又は２以上のワーカーを使用して、第１の準備完了キューからワークユニットを処理するステップをさらに含む、請求項１に記載の方法。

【請求項１８】

再パーティショニングするステップが、新しいパーティショニングルールが、（１）各ワークユニットに対する複数のパーティションのうちの１つのコンテンツに依存する割当てであって、特定のワークユニットに割り当てられた前記パーティションが、前記ワークユニットに含まれる少なくとも１つの値に基づいて決定される割当てを使用するか、又は、（２）各ワークユニットに対する前記複数のパーティションのうちの１つのコンテンツに依存しない割当てであって、特定のワークユニットに割り当てられた前記パーティションが、前記ワークユニットに含まれる任意の値から独立して決定される割当てを使用するかを決定するステップを含む、請求項１７に記載の方法。

【請求項１９】

再パーティショニングするステップが、第１の準備完了キューからワークユニットを処理するステップが、前記ワークユニットが第１の保留キューに受信されたのと同じ順序で前記第１の保留キューから発するワークユニットを処理するステップを含むかどうかを決定するステップを含む、請求項１８に記載の方法。

【請求項２０】

再パーティショニングするステップが、古いパーティショニングルールから新しいパーティショニングルールにパーティショニングルールを変更する前に、及び前記新しいパー

ティショニングルールがコンテンツに依存する割当てを使用すると決定し、第１の準備完了キューからワークユニットを処理するステップが、前記ワークユニットが第１の保留キューに受信されたのと同じ順序で前記第１の保留キューから発するワークユニットを処理するステップを含むと決定した後に、すべての再配置キューを排出させるステップを含む、請求項１９に記載の方法。

【請求項２１】

排出させるステップが、（１）各再配置キュー内の各ワークユニットを、古いパーティショニングルールに従ってそのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置するステップと、（２）前記排出中に、第１の準備完了キューからのワークユニットの処理を中断するステップとを含む、請求項２０に記載の方法。

【請求項２２】

パーティショニングルールが、各ワークユニットに対して複数のパーティションのうちの１つのキーベースの割当てを使用し、特定のワークユニットに割り当てられた前記パーティションが、前記ワークユニットに含まれるキー値の関数に基づいて決定される、請求項１に記載の方法。

【請求項２３】

コンピューティングシステムの１又は２以上のノードが複数のノードを含み、前記複数のノードのストレージが、複数の保留キューと、キー値に基づいて前記複数のノードの各々にアクセス可能な共有データとを含む、請求項２２に記載の方法。

【請求項２４】

共有データが、第１のノードのストレージに記憶された第１の準備完了キューから１又は２以上のワークユニットの処理中に、複数のノードのうちの前記第１のノード内の少なくとも第１のプロセッサによってアクセスされる、請求項２３に記載の方法。

【請求項２５】

共有データが、ワークユニットに割り当てられた複数のパーティションに従ってパーティショニングされ、前記共有データにアクセスするために使用される各キー値が、前記複数のパーティションのうちの一意のパーティションに関連付けられる、請求項２３又は２４に記載の方法。

【請求項２６】

パーティショニングルールの変更に応じてワークユニットを再パーティショニングするステップが、新しいパーティショニングルールに従って共有データを再パーティショニングし始めた後に発生する、請求項２５に記載の方法。

【請求項２７】

新しいパーティショニングルールに従って共有データを再パーティショニングするステップが、

前記共有データに記憶された複数の参照ユニットの参照ユニットごとに、前記新しいパーティショニングルールに従ってその参照ユニットに割り当てられた複数のパーティションのうちの１つを決定するステップと、

以前のノードから新しいノードに、参照ユニットが記憶されている前記ノードに関連付けられていないパーティションが割り当てられたその任意の参照ユニットを転送するステップと、

前記以前のノードと前記新しいノードの両方において前記共有データを再パーティショニングする前記ステップの間に、新しいノードに転送された前記参照ユニットのうちの少なくとも１つにアクセスしようと試みるステップと

を含む、請求項２６に記載の方法。

【請求項２８】

処理するステップがさらに、

各準備完了キューから少なくとも１つの結果を提供するステップであって、前記各準備完了キューからの前記結果が、その準備完了キュー内に記憶された１又は２以上のワークユニットに基づく、前記提供するステップと、

前記結果が提供された前記ワークユニットを前記各準備完了キューから除去するステップと
を含む、請求項１～２７のいずれかに記載の方法。
【請求項２９】
保留キュー、再配置キュー、及び準備完了キューを含むワークユニットを記憶するキューを使用して、データを処理するために、コンピュータ可読媒体上に非一時的な形式で記憶されるソフトウェアであって、前記ソフトウェアが、コンピューティングシステムに、
前記コンピューティングシステムの１又は２以上のノードのストレージに記憶された複数の保留キューにワークユニットを受信することと、
前記１又は２以上のノードの各々における少なくとも１つのプロセッサを使用して、前記複数の保留キューから前記ワークユニットを処理することと
を行わせるための命令を含み、前記処理することが、
前記保留キューごとに、その保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定することであって、各パーティションが一意の保留キューに関連付けられる、前記決定することと、
各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離することであって、前記２つの他のキューが、ワークユニットが記憶されている前記保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている前記保留キューに関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離することと、
各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられた前記パーティションに関連付けられる前記保留キューに再配置することと、
前記複数の保留キューが空になる前に、前記パーティショニングルールの新しいパーティショニングルールへの変更に応答して、前記ワークユニットを再パーティショニングすることと
を含む、前記ソフトウェア。
【請求項３０】
保留キュー、再配置キュー、準備完了キューを含むワークユニットを記憶するキューを使用してデータを処理するためのコンピューティングシステムであって、
１又は２以上のノードを含み、それぞれが、
複数の保留キューのうちのストレージに記憶された少なくとも１つの保留キューにワークユニットを受信するように構成された入力デバイス又はポートと、
前記保留キューから前記ワークユニットを処理するように構成された少なくとも１つのプロセッサとを含み、前記複数の保留キューのために処理するステップが、
前記保留キューごとに、その保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定するステップであって、各パーティションが一意の保留キューに関連付けられる、前記決定するステップと、
各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離するステップであって、前記２つの他のキューが、ワークユニットが記憶されている前記保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている前記保留キューに関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離するステップと、
各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられた前記パーティションに関連付けられる前記保留キューに再配置するステップと、
前記複数の保留キューが空になる前に、前記パーティショニングルールの新しいパーティショニングルールへの変更に応答して、前記ワークユニットを再パーティショニングす

るステップと
を含む、前記コンピューティングシステム。

【発明の詳細な説明】
【技術分野】
【０００１】
関連出願の相互参照
本出願は、２０１５年１２月１７日に出願された米国特許出願第６２／２６８，７６２号に対する優先権を主張する。
【０００２】
本明細書は、動的なパーティショニングを使用してデータを処理することに関する。
【背景技術】
【０００３】
複数のコンピューティングノード（例えば、サーバクラスタのサーバ、及び／又はマルチコアプロセッサのコア）を使用して並列にデータを処理するコンピューティングシステムは、パーティショナを使用してそれらのコンピューティングノード間でデータを分散することができる。データは、ワークの個々のユニットを表す「ワークユニット」において処理され得、ワークユニットは、任意の数のそれぞれのフィールドの値を有する個々のレコードの形態であり得る。或いは、ワークユニットは、レコードに関連付けられるメタデータ、処理されるべきイベントを記述する情報、又は実施されるべきワークの個々の部分を表す任意の他の情報の形態であり得る。各コンピューティングノードには異なるパーティションが割り当てられ得、そのコンピューティングノード上で動作する１又は２以上のワーカー（例えば、プロセス又はスレッド）は、その割り当てられたパーティション内のワークユニットを処理する。レコードが処理されると、フィールド内の値が変更され得る。データ処理のいくつかの段階において、ワークユニットは、そのレコードに適用されるべき識別されたタスクとともに、個々のレコードの形態であり得る。パーティショナは、特定のワークユニットを、そのワークユニットが属する特定のパーティションに分散するために、パーティショニングルールを使用する。いくつかのコンピューティングシステムでは、処理は段階的に実施され、ワークユニットは、同じパーティション又は異なる数のパーティションの間で後続の段階のために再パーティショニングされる。例えば、各コンピューティングノードは、入力ワークユニットを受信するための入力バッファと、出力ワークユニットを提供するための出力バッファとを有し得る。パーティショナは、出力バッファごとに、前の段階からの出力ワークユニットを、次の段階の入力ワークユニットとして処理されるべき入力バッファ間で分散する。そのような再パーティショニングは、通常、前の段階の計算がすべて完了した後に実施される。そのような再パーティショニングに使用されるパーティショニングルールは変更される可能性があるが、変更は異なるワークユニットのセット間で発生する必要がある。言い換えれば、所与のワークユニットのセットについてパーティショニングルールは静的であり、したがって、次のパーティショニングルールを使用して次のセット内の任意のワークユニットがパーティショニングされ得る前に、所与のセット内のすべてのワークユニットが現在のパーティショニングルールを使用して完全にパーティショニングされる。
【発明の概要】
【課題を解決するための手段】
【０００４】
一態様では、一般に、保留キュー、再配置キュー、及び準備完了キューを含むワークユニットを記憶するキューを使用して、コンピューティングシステムにおいてデータを処理するための方法は、コンピューティングシステムの１又は２以上のノードのストレージに記憶された複数の保留キューにワークユニットを受信するステップと、１又は２以上のノードの各々における少なくとも１つのプロセッサを使用して、複数の保留キューからワークユニットを処理するステップとを含む。処理するステップは、保留キューごとに、その

保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定するステップであって、各パーティションが一意の保留キューに関連付けられる、前記決定するステップと、各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離するステップであって、２つの他のキューが、ワークユニットが記憶されている保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている保留キューに関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離するステップと、各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置するステップと、複数の保留キューが空になる前に、パーティショニングルールの新しいパーティショニングルールへの変更に応答して、ワークユニットを再パーティショニングするステップとを含む。

【０００５】

態様は、以下の特徴のうちの１又は２以上を含んでもよい。疑念を避けるために、以下の特徴のうちのいずれかは、特徴が相互に排他的でない限り、他の以下の特徴のうちのいずれかと組み合わせて使用され得る（例えば、コンテンツに依存する割当てに関連する特徴と、コンテンツに依存しない割当てに関連する特徴は、代替である）。

【０００６】

パーティショニングルールの変更は、複数のパーティション内のパーティションの総数の変更を含む。

【０００７】

コンピューティングシステムの１又は２以上のノードは複数のノードを含み、複数のノードのうちの少なくとも第１のノードは、第１のパーティションに関連付けられる第１の保留キューを記憶し、第１の保留キューに対応する第１の再配置キュー及び第１の準備完了キューを記憶するストレージを有し、複数のノードのうちの少なくとも第２のノードが、第２のパーティションに関連付けられる第２の保留キューを記憶し、第２の保留キューに対応する第２の再配置キュー及び第２の準備完了キューを記憶するストレージを有する。

【０００８】

パーティショニングルールの変更は、保留キューのうちのいずれかを記憶するために使用されるストレージを有する複数のノード内のノードの総数の変更を含む。

【０００９】

第１の再配置キュー内に１又は２以上のワークユニットを再配置するステップは、第１のノードから第２のノードに１又は２以上のワークユニットを含むメッセージを送信するステップを含み、第２のパーティションは、メッセージに含まれる１又は２以上のワークユニットの各々に割り当てられる。

【００１０】

第１のノードのストレージはまた、第３のパーティションに関連付けられる第３の保留キューを記憶し、第３の保留キューに対応する第３の再配置キュー及び第３の準備完了キューを記憶する。

【００１１】

再パーティショニングするステップは、特定の保留キューに対応する再配置キュー及び準備完了キューに記憶された任意のワークユニットを、その特定の保留キューに戻すステップを含む。

【００１２】

パーティショニングルールは、各ワークユニットに対して複数のパーティションのうちの１つのコンテンツに依存する割当てを使用し、特定のワークユニットに割り当てられたパーティションは、ワークユニットに含まれる少なくとも１つの値に基づいて決定される。

【００１３】
各保留キューは、初期パーティション決定がまだ行われていないワークユニットを記憶するための第１の部分と、それらのワークユニットをその保留キューに再配置させる初期パーティション決定が行われたワークユニットを記憶するための第２の部分とを含む。
【００１４】
ワークユニットを各保留キューから対応する再配置キュー又は準備完了キューに分離するステップは、第２の部分が空である場合にのみ第１の部分に記憶されたワークユニットを分離するステップを含む。
【００１５】
少なくとも第１の保留キューは、パーティショニングルールの変更後にその保留キューに戻された任意のワークユニットを記憶するための第３の部分を含む。
【００１６】
再パーティショニングするステップは、第１の保留キューに対して、新しいパーティショニングルールに従って第１の保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを決定するステップと、ワークユニットを第１の保留キューから対応する再配置キュー又は準備完了キューに分離するステップとをさらに含む。
【００１７】
ワークユニットを第１の保留キューから対応する再配置キュー又は準備完了キューに分離するステップは、第３の部分が空である場合にのみ第２の部分に記憶されたワークユニットを分離するステップと、第２の部分と第３の部分の両方が空である場合にのみ第１の部分に記憶されたワークユニットを分離するステップとを含む。
【００１８】
パーティショニングルールは、複数のパーティションのうちの１つのコンテンツに依存しない割当てを各ワークユニットに使用し、特定のワークユニットに割り当てられたパーティションは、ワークユニットに含まれる任意の値から独立して決定される。
【００１９】
パーティショニングルールは、複数のパーティションのうちの１つの負荷バランシング割当てを各ワークユニットに使用し、特定のワークユニットに割り当てられたパーティションは、異なるパーティションに対する負荷の定量的尺度に基づいて決定される。
【００２０】
異なるパーティションに対する負荷の定量的尺度は、異なるパーティションに関連付けられる保留キューに再配置されるべき記憶されたワークユニットの異なるセットのバックログの定量的尺度である。
【００２１】
本方法は、第１のノード上で実行する１又は２以上のワーカーを使用して、第１の準備完了キューからワークユニットを処理するステップをさらに含む。
【００２２】
再パーティショニングするステップは、新しいパーティショニングルールが、（１）各ワークユニットに対する複数のパーティションのうちの１つのコンテンツに依存する割当てであって、特定のワークユニットに割り当てられたパーティションが、ワークユニットに含まれる少なくとも１つの値に基づいて決定される割当てを使用するか、又は、（２）各ワークユニットに対する複数のパーティションのうちの１つのコンテンツに依存しない割当てであって、特定のワークユニットに割り当てられたパーティションが、ワークユニットに含まれる任意の値から独立して決定される割当てを使用するかを決定するステップを含む。
【００２３】
再パーティショニングするステップは、第１の準備完了キューからワークユニットを処理するステップが、ワークユニットが第１の保留キューに受信されたのと同じ順序で第１の保留キューから発するワークユニットを処理するステップを含むかどうかを決定するス

テップを含む。
【００２４】
再パーティショニングするステップは、古いパーティショニングルールから新しいパーティショニングルールにパーティショニングルールを変更する前に、及び新しいパーティショニングルールがコンテンツに依存する割当てを使用すると決定し、第１の準備完了キューからワークユニットを処理するステップが、ワークユニットが第１の保留キューに受信されたのと同じ順序で第１の保留キューから発するワークユニットを処理するステップを含むと決定した後に、すべての再配置キューを排出させるステップを含む。
【００２５】
排出させるステップは、（１）各再配置キュー内の各ワークユニットを、古いパーティショニングルールに従ってそのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置するステップと、（２）排出中に、第１の準備完了キューからのワークユニットの処理を中断するステップとを含む。
【００２６】
パーティショニングルールは、各ワークユニットに対して複数のパーティションのうちの１つのキーベースの割当てを使用し、特定のワークユニットに割り当てられたパーティションは、ワークユニットに含まれるキー値の関数に基づいて決定される。
【００２７】
コンピューティングシステムの１又は２以上のノードは複数のノードを含み、複数のノードのストレージは、複数の保留キューと、キー値に基づいて複数のノードの各々にアクセス可能な共有データとを含む。
【００２８】
共有データは、第１のノードのストレージに記憶された第１の準備完了キューから１又は２以上のワークユニットの処理中に、複数のノードのうちの第１のノード内の少なくとも第１のプロセッサによってアクセスされる。
【００２９】
共有データは、ワークユニットに割り当てられた複数のパーティションに従ってパーティショニングされ、共有データにアクセスするために使用される各キー値は、複数のパーティションのうちの一意のパーティションに関連付けられる。
【００３０】
パーティショニングルールの変更に応じてワークユニットを再パーティショニングするステップは、新しいパーティショニングルールに従って共有データを再パーティショニングし始めた後に発生する。
【００３１】
新しいパーティショニングルールに従って共有データを再パーティショニングするステップは、共有データに記憶された複数の参照ユニットの参照ユニットごとに、新しいパーティショニングルールに従ってその参照ユニットに割り当てられた複数のパーティションのうちの１つを決定するステップと、以前のノードから新しいノードに、参照ユニットが記憶されているノードに関連付けられていないパーティションが割り当てられたその任意の参照ユニットを転送するステップと、以前のノードと新しいノードの両方において共有データを再パーティショニングするステップの間に、新しいノードに転送された参照ユニットのうちの少なくとも１つにアクセスしようと試みるステップとを含む。
【００３２】
処理するステップはさらに、各準備完了キューから少なくとも１つの結果を提供するステップであって、各準備完了キューからの結果が、その準備完了キュー内に記憶された１又は２以上のワークユニットに基づく、ステップと、結果が提供されたワークユニットを各準備完了キューから除去するステップとを含む。
【００３３】
別の態様では、一般に、ソフトウェアは、保留キュー、再配置キュー、及び準備完了キューを含むワークユニットを記憶するキューを使用して、データを処理するために、コン

ピュータ可読媒体上に非一時的な形式で記憶される。ソフトウェアは、コンピューティングシステムに、コンピューティングシステムの１又は２以上のノードのストレージに記憶された複数の保留キューにワークユニットを受信することと、１又は２以上のノードの各々における少なくとも１つのプロセッサを使用して、複数の保留キューからワークユニットを処理することとを行わせるための命令を含む。処理することは、保留キューごとに、その保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定することであって、各パーティションが一意の保留キューに関連付けられる、前記決定することと、各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離することであって、２つの他のキューが、ワークユニットが記憶されている保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている保留キューに関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離することと、各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置することと、複数の保留キューが空になる前に、パーティショニングルールの新しいパーティショニングルールへの変更に応答して、ワークユニットを再パーティショニングすることとを含む。

【００３４】

別の態様では、一般に、保留キュー、再配置キュー、準備完了キューを含むワークユニットを記憶するキューを使用して、データを処理するためのコンピューティングシステムは、１又は２以上のノードを含む。各ノードは、複数の保留キューのうちのストレージに記憶された少なくとも１つの保留キューにワークユニットを受信するように構成された入力デバイス又はポートと、保留キューからワークユニットを処理するように構成された少なくとも１つのプロセッサとを含む。複数の保留キューのために処理するステップは、保留キューごとに、その保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定するステップであって、各パーティションが一意の保留キューに関連付けられる、前記決定するステップと、各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離するステップであって、２つの他のキューが、ワークユニットが記憶されている保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている保留キューに関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離するステップと、各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置するステップと、複数の保留キューが空になる前に、パーティショニングルールの新しいパーティショニングルールへの変更に応答して、ワークユニットを再パーティショニングするステップとを含む。

【００３５】

別の態様では、一般に、保留キュー、再配置キュー、及び準備完了キューを含むワークユニットを記憶するキューを使用して、データを処理するためのコンピューティングシステムは、コンピューティングシステムの１又は２以上のノードのストレージに記憶された複数の保留キューにワークユニットを受信するための手段と、複数の保留キューからワークユニットを処理するための手段とを含む。処理するステップは、保留キューごとに、その保留キューに記憶された各ワークユニットに割り当てられた複数のパーティションのうちの１つを、パーティショニングルールに従って決定するステップであって、各パーティションが一意の保留キューに関連付けられる、前記決定するステップと、各保留キューから、ワークユニットを、その保留キューと同じストレージに記憶された少なくとも２つの他のキューの対応する１つに分離するステップであって、２つの他のキューが、ワークユニットが記憶されている保留キューに関連付けられていないパーティションが割り当てられたワークユニットごとの再配置キューと、ワークユニットが記憶されている保留キュー

に関連付けられているパーティションが割り当てられたワークユニットごとの準備完了キューとを含む、前記分離するステップと、各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置するステップと、複数の保留キューが空になる前に、パーティショニングルールの新しいパーティショニングルールへの変更に応答して、ワークユニットを再パーティショニングするステップとを含む。

【００３６】

異なるキューの名前（例えば、「保留キュー」、「再配置キュー」、「準備完了キュー」）、又は異なるサブキュー（例えば、「保留中のサブキュー」、「再循環サブキュー」）は、純粋にラベル付け目的にすぎず、それらの名前には技術輸入が全くない点に留意されたい。異なる名前を有する異なるキュー又はサブキュー（本明細書では、異なる「キューのタイプ」とも呼ばれる）は、同じ技術を使用して実装されてもよく、本明細書に明示的に記載されているように、それらが使用される方法に関してのみ異なっていてもよい。異なる方法で使用されるキュー又はサブキューは、「第１の」、「第２の」、「第３の」（又は、同様のもの）を使用してラベル付けされてもよいが、純粋に読者の便宜のために、より記述的な言葉を使用してラベル付けされていてもよい。

【００３７】

態様は、以下の利点のうちの１又は２以上を含むことができる。

【００３８】

いくつかのデータ処理システムは、「ワークスティーリング」などの技法を使用して、サーバクラスタの異なるサーバ間で計算のワークユニットを分散する。例えば、各サーバ上のワーカーは、対応するワークキューからワークユニットを消費し得、空のワークキューを有するアイドル状態のサーバは、そのワークキュー内のワークユニットのバックログが大きい別のサーバからワークユニットを取得することができる。これは、負荷バランシングを容易にする一種の動的な並列処理の形式を提供するが、部分的にはワークユニットを要求するメッセージを通信するためのコストのために、及び部分的にはデータのローカル性を管理することの複雑さ（すなわち、一部のワークユニットは別のサーバによって盗まれるべきではなく、ローカルサーバに残すべきである）のために、ワークスティーリング(work stealing)のスケーラビリティ(scalability)が制限されることがある。

【００３９】

ローカル性を実現する１つの方法は、特定のワークユニットを処理する必要があるタスクを、そのタスクが生成されたサーバから、その特定のワークユニットがローカルに記憶されたサーバに移行させることである。しかし、そのような移行は、動的ではあるが、タスクに関連付けられる状態をシリアル化して転送する必要があるため、コストが高くなる可能性がある。場合によっては、特定のワークユニットが適切なサーバに送信され、ワーカーがその特定のワークユニットで実施されるべきタスクを生成する方が効率的な場合もある。さらに、短命のタスクの場合、アクティブなタスクの移行は、そのタスクの全寿命のかなりの部分を占める可能性がある。

【００４０】

異なるサーバ（又は、他のタイプのコンピューティングノード）間でのワークユニットの明示的なパーティショニングは、適切なパーティショニングルールを使用してローカル性の管理を容易にすることができる。場合によっては、ローカル性が重要ではない場合もあり、その場合、サーバ間でワークユニットを均等に分散するラウンドロビン(round-robin)パーティショニングルールなどの、コンテンツに依存しないパーティショニングが使用され得る。しかし、他の場合、ワークユニットは、レコードのキーフィールド内のキー値に基づいてパーティションを割り当てるパーティショニングルールなどのコンテンツに依存する方法でサーバに分散される必要がある場合がある。

【００４１】

本明細書に記載される技法は、パーティショニングルールを動的に（すなわち、ワークユニットのグループのパーティショニング中に）変更することを可能にする。したがって

、データ処理システム内の複雑さ又は効率の損失に過度の負担をかけることなしに、ワークユニットの動的なパーティショニングの利点が達成され得る。例えば、以下でより詳細に説明するように、パーティショニングの異なる態様に複数のキューがあるため、新しいパーティショニングルールに従って再度処理する必要があり得る任意のワークユニットを決定する方がより簡単である。また、必要に応じて特定のワークユニット間の相対的順序付けの保存を容易にするパーティショニング手順の態様もある。また、データ処理システムは、様々なコンピューティングノードにおいてパーティショニングを追加又は除去するためにパーティショニングルールを変更することによって、必要に応じてコンピューティングリソースを増減させることもできる。ワークスティーリングに関連付けられる負荷バランシングの利点の一部は、パーティショニングルールの適切な管理によっても達成され得る。

【００４２】

本発明の他の特徴及び利点は、以下の説明及び特許請求の範囲から明らかになるであろう。

【図面の簡単な説明】

【００４３】

【図１】データ処理システムのブロック図である。

【図２】データ処理グラフの図である。

【図３Ａ】複数のキュー構造のセットの図である。

【図３Ｂ】複数のキュー構造の代替セットの図である。

【図４Ａ】ワークユニットの接続とパスの例を示す、複数のキュー構造のセットの図である。

【図４Ｂ】ワークユニットの接続とパスの例を示す、複数のキュー構造のセットの図である。

【図５Ａ】データ処理グラフの図である。

【図５Ｂ】データ処理グラフの図である。

【発明を実施するための形態】

【００４４】

図１は、動的なパーティショニング技法が使用され得るデータ処理システム１００の例を示す。システム１００は、ネットワーク１０４を介して通信する複数のコンピューティングノード１０２（例えば、ローカルエリアネットワーク（ＬＡＮ，local area network）及び／又はワイドエリアネットワーク（ＷＡＮ，wide-area network）を介して通信するサーバ、或いはマルチコアプロセッサ内のネットワークを介して通信するコア、或いはそれらの組合せ）を含む。システム１００はまた、コンピューティングノード１０２に加えて、データベース又は他のストレージシステムなどのデータのソース又はリポジトリ、或いはオンラインデータストリームへの接続を表す１又は２以上のデータストレージノード１０６を含む、他のタイプのノードを含むことができる。コンピューティングノード１０２は、例えば、プロセッサ１１０（例えば、中央処理装置（ＣＰＵ，central processing unit）又はマルチコアプロセッサ）、ワーキングメモリ１１２（例えば、揮発性ストレージ媒体）、及び永続ストレージ１１４（例えば、不揮発性ストレージ媒体）を有するサーバコンピュータとして実装され得る。各コンピューティングノード１０２は、ノード１０２に記憶されたワークユニットを処理するためにワーカーを実行するために、ＵＮＩＸオペレーティングシステムのバージョンなどの適切なオペレーティングシステムの制御下で実行環境をホストするように構成され得る。システム１００内の別のノードは、プログラミングのための、或いはコンピューティングノード１０２の一部又は全部及びネットワーク１０４の少なくとも一部を備えるサーバクラスタと相互作用するためのユーザインターフェースを提供するアクセスノード１１８であり得る。いくつかのコンピューティングノード１０２は、サーバクラスタの一部でなくてもよいが、他のコンピューティングノード１０２の機能のサブセット（例えば、データを記憶するが、ワーカーを実行しないノード）を提供してもよい。

【００４５】
アクセスノード１１８は、開発者が、データ処理システム１００によって実行されるべきデータ処理プログラムを特定することができる開発環境を提供し得る。いくつかの実施形態では、データ処理プログラムは、頂点間の有向リンク（ワークエレメント、すなわちデータの流れを表す）によって接続された頂点（データ処理コンポーネント又はデータセットを表す）を含むデータフローグラフである。例えば、そのような環境は、参照により本明細書に組み込まれる、「Managing Parameters for Graph-Based Data processing graphs」（グラフに基づくデータ処理グラフのためのパラメーター管理）と題された米国特許出願公開第２００７／００１１６６８号明細書により詳細に記載されている。そのようなグラフベースの計算を実行するためのシステムは、参照により本明細書に組み込まれる、「EXECUTING COMPUTATIONS EXPRESSED AS GRAPHS」と題された米国特許第５９６６０７２号明細書に記載されている。このシステムに従って作成されたデータフローグラフは、グラフコンポーネントによって表される個々のプロセスに出入りする情報を取得することと、プロセス間で情報を移動することと、プロセスの実行順序を定義することとを行うための方法を提供する。このシステムは、あらゆる使用可能な方法からプロセス間通信方法を選択するアルゴリズムを含む（例えば、グラフのリンクによる通信パスが、ＴＣＰ／ＩＰ又はＵＮＩＸドメインソケットを使用することもでき、プロセス間でデータを渡すために共有メモリを使用することもできる）。

【００４６】
いくつかの実施形態では、データ処理プログラムは、データフローに加えて、参照により本明細書に組み込まれる、「COMPILING GRAPH-BASED PROGRAM SPECIFICATIONS」（グラフに基づくプログラム仕様のコンパイリング）と題された米国特許出願公開第２０１６／００７０７２９号明細書に記載される特徴を含む、データ処理グラフである。例えば、この公開は、データ処理グラフが、データの転送又は制御の転送のいずれか、或いはその両方を表すノード間のリンクを含み得ることを説明している。リンクの特性を示す１つの方法は、コンポーネントに異なるタイプのポートを提供することである。リンクは、上流コンポーネントの出力ポートから下流コンポーネントの入力ポートに結合された有向リンクである。ポートには、データエレメントがリンクからどのように書き込まれ、及び読み取られるか、並びに／又は、データを処理するためにコンポーネントがどのように制御されるかの特性を表すインジケータがある。

【００４７】
これらのポートは、いくつかの異なる特性を有し得、そのうちのいくつかは、コンパイラ及び／又は実行時(runtime)コンピューティングシステムが、並列化を達成するために、どのようにワークユニットをパーティショニングするかに影響を及ぼす。ポートの１つの特徴は、入力ポート又は出力ポートとしてのその方向性である。有向リンクは、上流コンポーネントの出力ポートから下流コンポーネントの入力ポートに伝達されるデータ及び／又は制御を表す。ポートのもう１つの特徴は、そのタイプである。開発者は、異なるタイプのポートをリンクすることが許容されている。データ処理グラフのデータ処理特性のいくつかは、異なるタイプのポートがどのようにリンクされているかに依存する。例えば、異なるタイプのポート間のリンクは、各実行が、その実行セット内のコンポーネントの複数のインスタンスにパーティショニングされ得るワークユニットの上流集合によって供給される階層型の並列性を提供する、異なる「実行セット」内のコンポーネントのネストされたサブセットにつながることができる。コンポーネントが有し得る異なるタイプのポートは、次のものを含む。

・コンポーネントのインスタンスが、それぞれ、ポートに接続されたリンクを通過する収集のすべてのワークユニットを読み書きすることを意味する、収集の入力ポート又は出力ポート。それらの収集ポート間に単一のリンクを有するコンポーネントのペアの場合、下流コンポーネントは、上流コンポーネントによって書き込まれるときにワークユニットを読み取ることが一般に許容され、上流コンポーネントと下流コンポーネントとの間のパイプラインの並列性が可能になる。また、ワークユニットは並べ替えが行われてもよく、

これにより並列化の効率化が可能になる。いくつかのグラフ表示では、例えば図２、図５Ａ、及び図５Ｂに示されるデータ処理グラフにおいて、そのような収集ポートは、一般に、コンポーネントにおいて四角形のコネクタ記号で示されている。

・コンポーネントのインスタンスが、ポートに接続されたリンクから、又はそこへ、最大で１つのワークユニットにおいてそれぞれ読み書きを行うことを意味する、スカラ入力ポート又は出力ポート。それらのスカラポート間に単一のリンクを有するコンポーネントのペアの場合、上流コンポーネントの実行が終了した後の下流コンポーネントのシリアル実行は、単一のワークユニットの転送を制御の転送として使用して強制される。いくつかのグラフ表示では、例えば図２、図５Ａ、及び図５Ｂに示されるデータ処理グラフにおいて、そのようなスカラポートは、一般に、コンポーネントにおいて三角形のコネクタ記号で示されている。

・スカラ入力又は出力と類似しているが、ワークユニットは送信される必要がなく、コンポーネント間で制御の転送を通信するために使用される、制御入力又は出力ポート。それらの制御ポート間のリンクを有するコンポーネントのペアの場合、上流コンポーネントの実行が終了した後の下流コンポーネントのシリアル実行が強制される（たとえ、それらのコンポーネントが収集ポート間のリンクも有する場合であっても）。いくつかの図形表現では、そのような制御ポートは、一般に、コンポーネントにおいて円形のコネクタ記号で示されている。

【００４８】

これらの異なるタイプのポートは、データ処理グラフの柔軟な設計を可能にし、ポートタイプの重複するプロパティを使用して、データと制御フローとの強力な組合せを可能にする。特に、データを何らかの形で伝達する収集ポートとスカラポートの２つのタイプのポートがあり（「データポート」と呼ばれる）、シリアル実行を強制するスカラポートと制御ポートの２つのタイプのポートがある（「シリアルポート」と呼ばれる）。データ処理グラフは、一般に、「ソースコンポーネント」（接続された入力データポートを有さない場合がある）である１又は２以上のコンポーネント、及び「シンクコンポーネント」（接続された出力データポートを有さない場合がある）である１又は２以上のコンポーネントを有する。

【００４９】

異なるタイプのポート間の接続も、異なる実行セットを示す。特に、収集タイプ出力ポートからスカラタイプ入力ポートへのリンクは、実行セットへの入口点を意味し、スカラタイプ出力ポートから収集タイプ入力ポートへのリンクは、実行セットの出口点を意味する。コンパイラは、コンポーネントを実行セットに自動的に割り当てるために、発見アルゴリズムを実施することができ、コンピューティングシステムは、本明細書に記載される動的なパーティショニング技法を使用してワークユニットの集合からワークユニットを処理するために、実行セットを動的に並列化することができる。

【００５０】

図２は、プログラミンググラフィカルインターフェース内のデータ処理グラフ２００の例を示す。データ処理グラフ２００は、第１のデータセット（Ｄ１）２０２、第１のコンポーネント（Ｃ１）２０４、第２のコンポーネント（Ｃ２）２０６、第３のコンポーネント（Ｃ３）２０８、第４のコンポーネント（Ｃ４）２１０、第５のコンポーネント（Ｃ５）２１２、第６のコンポーネント（Ｃ６）２１４、及び第２のデータセット（Ｄ２）２１６を含む。コンパイラは、（ルート、「レベル０」実行セット２２２に加えて）「レベル１」実行セット２１８及び「レベル２」実行セット２２０の発見をもたらす実行セットディスカバリ手順を使用して、データ処理グラフ２００の個々のコンポーネント及びデータセットを実行セットに割り当てることができる。ルート実行セット２２２以外の各実行セットは、ワークユニットの上流集合によって供給され、各入来ワークユニット上で動作する実行セットの異なるインスタンスに対応するワークユニットは、実行時にコンピューティングノード１０２の間でパーティショニングされ得る。そのようなパーティショニングを実施する能力は、コンパイラが実行可能コードに挿入する「ｆｏｒ－ａｌｌ」プリミテ

ィブを使用してコンパイラによって示されてよく、これは実行時に複数のキュー構造（ＭＱＳ，multiple queue structure）と呼ばれる複数のキューを有するバッファデータ構造の作成をトリガし、以下でより詳細に説明される。

【００５１】

実行時に、ｆｏｒ－ａｌｌプリミティブは、「サーバグループ」と呼ばれるコンピューティングノード１０２のグループに分散されるワーカーのプールによって実施されるｆｏｒ－ａｌｌ操作をトリガする（各コンピューティングノード１０２は、ワーカーのプールを管理する「サーバ」である）。実行セットによって処理されるべきワークユニットの集合内のすべてのワークユニットの場合、その実行セット内のコンポーネントのグループのインスタンスが、サーバグループ内のサーバのうちの１つの上で実行される。サーバグループ内の各サーバには、対応するパーティショナによって管理されるＭＱＳがあり、そのサーバグループ用のＭＱＳのグループをともに形成する。サーバグループ内のサーバは、ｆｏｒ－ａｌｌ操作の期間にわたって必ずしも一定ではない（例えば、サーバはサーバグループに動的に追加、又はそこから除去され得る）。ｆｏｒ－ａｌｌ操作は、サーバグループ内の変更を認識する。サーバが追加されると、割り振られたＭＱＳからワークを消費し、ＭＱＳを管理するすべてのパーティショナによって使用されるパーティショニング機能を変更するために、新しいサーバ上のワーカープールが始動する。サーバが除去された場合、本明細書に記載される方法に従って、そのワーカープールは削除され、そのワークは再配置されるか排出される。

【００５２】

ＭＱＳは、ｆｏｒ－ａｌｌ操作以外の他のタイプの操作のバッファとしても使用され得る。例えば、「ロールアップ」や「結合」などのデータベース操作は、操作の結果を累積するために追加のステップを含むことによって、ｆｏｒ－ａｌｌ操作として実装され得る。また、上流とは異なる数のコンピューティングノード１０２の下流上で実行するデータ処理プログラムの上流部分と下流部分との間でＭＱＳが使用され得る。例えば、コンピューティングノード１０２がマルチコアプロセッサ上のコアであり、下流プロセッサが上流プロセッサの２倍のコアを有する場合、１つの上流スレッドから２つの下流スレッドにワークユニットを渡すためにＭＱＳのグループが使用され得る。メモリ階層に従ってインテリジェントにパーティショニングされている場合、１つのコアで処理されるすべてのデータは、同じローカルキャッシュを共有する他のコアに移動することができる。

【００５３】

各ＭＱＳは、（実行前にコンパイラによって）静的に決定され、（実行時に）動的に決定され、及び／又はワークユニットが処理されるときに動的に変更され得るパーティショニングルールで構成されたパーティショナによって管理される。例えば、コンパイラは、ほとんどのパーティショニングを使用しないデフォルトのパーティショニングルール（又は、１つのローカルパーティショニングのみを使用する、この場合、パーティショニングが実施される必要はない）を示す場合があり、次いで、実行時に実際のワークロードがあるしきい値より大きくなると、パーティションの数を増やすためにある時点で動的な決定が行われ得る。そのようなパーティショニングルールの変更後、以下でより詳細に説明されるように、ＭＱＳからワークユニットを消費するワーカーによって処理されるまで、任意のワークユニットが再パーティショニングされ得る（例えば、パーティショニングを追加又は除去するために、或いは別の理由のために）。

【００５４】

所与のワークユニットのパーティションを計算するために、様々なパーティショニングルールのいずれかが使用され得る。場合によっては、パーティションはワークユニットのキー値に基づいており、ハッシュ関数などの関数がキー値に適用される。パーティションの数を法とするハッシュ関数の結果が使用され得る。他のオプションは範囲パーティショニングを含み、キー値の範囲が異なるパーティションに割り当てられる。場合によっては、並列化の効率を高めるために、固有ではないキーに対して統計的な範囲パーティショニングが使用され得、より多数のワークユニットに属する可能性が高いことが知られている

「重い」キーは、単独のパーティションに置かれるか、又はより少ない数のワークユニットに属する可能性が高いと知られている「軽い」キーのみに配置される。例えば、キーが市外局番に対応する場合、大都市の市外局番が異なるパーティションに割り当てられてもよく、最も大きな都市の市外局番が、最も小さい町の市外局番と同じパーティションに意図的に割り当てられてもよい。

【００５５】

データ処理プログラムによって指定されたタスクを実行するワーカーによって消費又は生産されるワークユニット（おそらくはワークユニットの一部）に加えて、場合によっては、データ処理プログラムの実行時間を超えて持続する可能性のある他のより長命のデータもある。そのようなより長命のデータの「参照ユニット」は、例えば、永続ストレージ１１４からコンピューティングノード１０２上で実行するワーカー、又はデータストレージノード１０６からアクセスされ得る。いくつかの実施形態では、参照ユニットは、共有変数ごとに一意のキー値を使用してアクセス（すなわち、読取り又は書込み）され得るキー付き共有変数の形式で、すべてのコンピューティングノード１０２にアクセス可能な共有データとして提供される。これらのキー付き共有変数にアクセスする際のデータ競合を回避するために、ロックなどの適切なアクセス制御が使用され得る。参照ユニットはまた、コンピューティングノード１０２にアクセス可能なデータセット（例えば、ファイル又はデータベーステーブル）内に記憶され得る。場合によっては、参照ユニットは、システム１００の異なるノード上に記憶された複数のパーティションにわたってパーティショニングされ得る。

【００５６】

データ処理システム１００は、パーティションのうちの１つ及び対応するＭＱＳをそれぞれ管理する複数のパーティショナを使用してワークユニットを（例えば、ｆｏｒ－ａｌｌプリミティブを実装するために）パーティショニングするように構成され得、ワークユニットは、コンピューティングノード１０２のワーキングメモリ１１２又は永続ストレージ１１４、或いは両方の組合せに記憶され得る（例えば、キャッシングスキームを使用して）。図３Ａを参照すると、ＭＱＳ３００Ａ、３００Ｂ、３００Ｃ、３００Ｄのセットの例は、各ＭＱＳを構成する複数のキューが互いにどのように関係しているか、及び他のＭＱＳのキューにどのように関係しているかを示す。ＭＱＳ内の各キューは「先入れ先出し（ＦＩＦＯ，first in, first out）」プロパティを有し、したがって、ワークユニットは、一方の端（バックエンド又は「末尾」エンド）に追加され、また他方の端（フロントエンド又は「先頭」エンド）から除去される。パーティションの数Ｎはパーティショニングルールによって決定され、パーティショナのそれぞれについて同じである。この例ではＮ＝４であり、Ａ、Ｂ、Ｃ、及びＤとラベル付けされた異なるパーティションを有する。典型的には、コンピューティングノード１０２ごとに１つのパーティションがあり、その場合、各ＭＱＳは異なるコンピューティングノード１０２に記憶される。しかしながら、単一のコンピューティングノード１０２に、例えばそのコンピューティングノード１０２上の異なるプロセス及び／又は異なるコアに関連付けられ得る複数のパーティションを割り当てることも可能である。その場合、コンピューティングノード１０２は、複数のＭＱＳを記憶することができる。

【００５７】

各ＭＱＳは、パーティショニングされるべき入来ワークユニットを受信する保留キューを含む。したがって、この例では、ＭＱＳ３００Ａ、３００Ｂ、３００Ｃ、３００Ｄごとに、それぞれ１つの保留キュー３０２Ａ、３０２Ｂ、３０２Ｃ、３０２Ｄ（したがって、各パーティションＡ、Ｂ、Ｃ、Ｄに関連付けられる一意の保留キュー）が存在する。これらのワークユニットは、コンピューティングノード１０２のサブセット上にある１又は２以上のソースから到着している可能性があり、したがって、ワークユニットの保留キューへの最初の到着は、保留キューのすべてより少ない数のデータを取り込み、及び／又は保留キューを不均等に満たす可能性がある。パーティショナによって使用されるパーティショニングルールの目標は、保留キューに到着するワークユニットを特定の方法で再分散す

ることである。

【００５８】

プロセッサ１１０上で実行されるパーティショナは、所与のＭＱＳのワークユニットを処理する。具体的には、ＭＱＳごとのパーティショナは、パーティショニングルールに従って、Ｎ個のパーティショニングのうちのいずれがそのＭＱＳの保留キューに記憶された各ワークユニットに割り当てられているかを決定する。パーティショニングルールは、「コンテンツに依存する」方法であって、特定のワークユニットに割り当てられたパーティションが、ワークユニットに含まれる少なくとも１つの値に基づいて決定される方法で、又は、「コンテンツに依存しない」方法であって、特定のワークユニットに割り当てられたパーティションが、ワークユニットに含まれる任意の値から独立して決定される方法で、ワークユニットにパーティションを割り当ててよい。コンテンツに依存するパーティショニングルールは、例えば、ワークユニット内のキー値（又は、キー値に適用されるハッシュ関数）を使用し得る。これらのキー値は、ワークユニットのプライマリキー、ワークユニットに割り当てられた明示的なパーティショニングキー、同じシリアル化キーが割り当てられたワークユニットをそれらの到着順にシリアルに処理されなければならないシリアル化キー、特定の共有変数にアクセスするために使用される共有変数キー、或いはロールアップキー又は結合キーなどのデータベース操作のためのキーなどの、様々なタイプのキー値のいずれかであり得る。コンテンツに依存しないパーティショニングルールは、例えば、ラウンドロビン分布、擬似ランダム分布、又は、ＭＱＳ内のワークユニットのバックログがより小さいパーティショニングを優先する負荷バランシング分布、又はローカルコンピューティングノード１０２上のパーティショニングにワークユニットを維持することを優先する分布などの、ワークユニットコンテンツ以外の要因に依存する分布を使用し得る。

【００５９】

特定のワークユニットに割り当てられたパーティションの決定に基づいて、パーティショナは、各保留キューからワークユニットを、それぞれのＭＱＳ内の２つの他のタイプのキューのうちの対応する１つに分離する。キューの１つのタイプは、その対応する保留キューと同じストレージにそれぞれ記憶された再配置キュー３０４Ａ、３０４Ｂ、３０４Ｃ、３０４Ｄのセット（それぞれ保留キュー３０２Ａ、３０２Ｂ、３０２Ｃ、３０２Ｄに対応する）である。ワークユニットが記憶されている保留キューに関連付けられていないパーティションが割り当てられたワークユニットについては、パーティショナは、ワークユニットをその保留キューから対応する再配置キューに移動する。別のタイプのキューは、対応する保留キューと同じストレージにそれぞれ記憶された準備完了キューのセット３０６Ａ、３０６Ｂ、３０６Ｃ、３０６Ｄ（それぞれ保留キュー３０２Ａ、３０２Ｂ、３０２Ｃ、３０２Ｄに対応する）である。ワークユニットが記憶されている保留キューに関連付けられているパーティションが割り当てられたワークユニットについては、パーティショナは、ワークユニットをその保留キューから対応する準備完了キューに移動する。準備完了キュー３０６Ａ、３０６Ｂ、３０６Ｃ、３０６Ｄごとに、その準備完了キューからワークユニットを消費する、対応するワーカープール３０８Ａ、３０８Ｂ、３０８Ｃ、３０８Ｄが存在する。

【００６０】

所与のＭＱＳの再配置キューも、そのＭＱＳのパーティショナによって処理される。再配置キューがいっぱいになると、パーティショナは、各再配置キュー内の各ワークユニットを、そのワークユニットに割り当てられたパーティションに関連付けられる保留キューに再配置する。したがって、図３Ａに示されるように、再配置キュー３０４Ａ内のワークユニットは保留キュー３０２Ｂ、３０２Ｃ、３０２Ｄのいずれかに送られ、再配置キュー３０４Ｂ内のワークユニットは保留キュー３０２Ａ、３０２Ｃ、３０２Ｄのいずれかに送られ、再配置キュー３０４Ｃ内のワークユニットは、保留キュー３０２Ａ、３０２Ｂ、３０２Ｄのいずれかに送られ、再配置キュー３０４Ｄ内のワークユニットは、保留キュー３０２Ａ、３０２Ｂ、３０２Ｃのいずれかに送られる。ＭＱＳが異なるコンピューティング

ノード１０２に記憶されている場合、再配置キュー（例えば、再配置キューＡ）から特定のパーティション（例えば、パーティションＢ）に割り当てられた１又は２以上のワークユニットを再配置することは、適切なコンピューティングノード１０２の間のネットワーク１０４を介して、１又は２以上のワークユニットを含むメッセージを送信すること含む。ワークユニットは個別に送信されてよく、又は効率のために、パーティショナ（又はパーティショナとは別のバックグラウンドプロセス）が、同じパーティション／ＭＱＳに移動するワークユニットのバッチで、同じメッセージにバンドルされたワークユニットを定期的に（例えば、１００ミリ秒ごとに）送信する（つまり、メッセージの総数を減らす）ことができる。或いは、ＭＱＳが同じコンピューティングノード１０２に記憶されている場合、１又は２以上のワークユニットを再配置することは、それらのワークユニット（又は、それらのワークユニットへのポインタ）をワーキングメモリ１１２のある部分から別の部分に単に移動することを伴い得る。また、いくつかの実施形態では、各ＭＱＳ内の単一の再配置キューの代わりに、潜在的な宛先パーティション（すなわち、Ｎ－１再配置キュー）ごとに１つずつ、複数の別個の再配置キューが存在し得る。

【００６１】

準備完了キューは、ＭＱＳを記憶しているコンピューティングノードのワーカーによって処理される。準備完了キューがいっぱいになると、その準備完了キューが記憶されている同じコンピューティングノード１０２上の１又は２以上のワーカーは、その準備完了キューからワークユニットを消費する。準備完了キューからワークユニットを除去した後にワークユニットを「消費」するためにワーカーが実施するタスクは、ＭＱＳの割振りを開始したデータ処理プログラムに依存し、場合によってはワークユニット自体にも依存する。タスクの１つの例は、データ処理グラフから発見された実行セットによって表される部分グラフである。タスクの別の例は、消費されたワークユニットを他のワークユニットと結合する結合操作などのデータベース操作、又は各ワークユニットが消費されると累積された結果を更新することによってワークユニットを集計するロールアップ操作である。ロールアップ操作の場合、累積された結果は部分集合の配列などの参照ユニットとして記憶され得、ワークユニットが消費されると、配列内の新しい項目として追加されるか（キーが以前に表示されていない場合）、部分集計のうちの１つを更新するために使用される。

【００６２】

いくつかの実施形態では、各保留キューは２つの部分に分割されるか、又は均等に、２つの別個のキューとして維持される。「保留中のサブキュー」と呼ばれる保留キューの第１の部分は、初期パーティション決定がまだ行われていない（すなわち、異なるＭＱＳから再配置されていないワークユニットに対して）ワークユニットを記憶する。「再循環サブキュー」と呼ばれる保留キューの第２の部分は、それらのワークユニットをその保留キューに再配置させる初期パーティション決定が行われたワークユニットを記憶する。ワークユニットが各保留キューから対応する再配置キュー又は準備完了キューに分離されると、ワークユニットは最初に再循環サブキューに記憶されたものから取り出され、再循環サブキューが空である場合にのみ、保留中のサブキューに記憶されたサブキューから取り出される。

【００６３】

いくつかの実施形態では、各保留キューは３つの部分に分割されるか、又は均等に、３つの別個のキューとして維持される。保留中のサブキュー及び再循環サブキューに加えて、「戻りサブキュー」と呼ばれる保留キューの第３の部分は、パーティショニングルールの変更後にその保留キューに戻された任意のワークユニットを記憶する。パーティショニングルールが変更されると、ワークユニットは、次の順序で、新しいパーティショニングルールに従って保留キューのこれらの部分から（準備完了キュー又は再配置キューに）除去される：最初に戻りサブキュー、次いで再循環サブキュー、次いで保留中のサブキュー。したがって、ワークユニットは、戻りサブキューが空の場合にのみ再循環サブキューから取り出され、再循環サブキューと戻りサブキューの両方が空の場合にのみ、ワークユニットは保留中のサブキューから取り出される。

【００６４】
上述のように、保留キューを２つ又は３つの部分に分割することは、ワークユニットがそれらのコンテンツに従って「正しく」再パーティショニングされることを保証するために役立つ。しかしながら、コンテンツに依存しないパーティショニングの場合、再パーティショニングのためにワークユニットを戻りサブキューに戻す必要はない可能性がある。
【００６５】
ワークユニット間の相対的順序付けが維持されなければならない場合もあり、そのような場合には、パーティショニングルールを変更する際に部分的順序付けを維持するための追加の技法がある。例えば、維持され得る１つの部分的順序付けプロパティは、常に同じＭＱＳに別のワークユニットとして存在していた任意のワークユニットが、そのワークユニットに対して相対的順序で存在することである。ワークユニット間の相対的順序付けが重要な場合のために、「パーティショニングエポック」に基づく例示的な手順を以下でより詳細に説明する。
【００６６】
ＭＱＳのキューはまた、複数の部分に分割されてもよく、又は他の理由により別々のキューに分割されてもよい。例えば、各キューは、ＭＱＳの準備完了キューからワークユニットを消費するワーカーごとに異なるサブキューに分割され得る。複数のサブキューが存在する可能性があるもう１つの理由は、動的な負荷バランシングを容易にするためである。パーティショナは、ワークユニットが送信され得る他のＭＱＳごとに異なるバージョンの再配置キューである別々のサブキューを維持することができる。次いで、それらの再配置サブキューの長さは、各ＭＱＳがどのくらいビジーであるかを示す背圧を感知するために使用され得る。また、パーティショナは、所与のワークユニットが、異なるＭＱＳのパーティションに再配置するために、それ自体のパーティションに残るか、又は再配置サブキューのうちの１つに移動する必要があるかどうかを決定するために、それ自体の準備完了キューから背圧を感知することができる。
【００６７】
動的なパーティショニングの場合、パーティショニングルールが変更され得、前のパーティショニングルールの下でのパーティショニングの完了前に（すなわち、すべてのＭＱＳの保留キューが空になる前に）ワークユニットが再パーティショニングされる。パーティショニングルールが変更される際には、パーティショニングの性質とＭＱＳの下流の任意の順序要件の両方に応じて、パーティショナが使用できる様々な手順がそれらのＭＱＳを管理する。一般に、パーティショニングルールの変更に応じてパーティショナがワークユニットを再パーティショニングする方法は、上記で画定されたように、ワークユニットへのパーティショニングの割当てが、コンテンツに依存するかコンテンツに依存しないかによって異なる。コンテンツに依存する場合、再パーティショニングするステップは、再配置キューに記憶された任意のワークユニットと特定の保留キューに対応する準備完了キューをその特定の保留キューに戻すステップを含む。戻されるワークユニットは、上記で画定された戻りサブキューに戻される場合がある。コンテンツに依存しない場合、再パーティショニングするステップはそのステップを実施する必要はない。コンテンツに依存する場合に対して特定の順序要件がある場合、再配置キューを排出する追加のステップもある。コンテンツに依存する、又はコンテンツに依存しないいずれのパーティショニングでも、パーティショニングルールを変更すると、パーティションの数が変更され得る。新しいパーティショニングごとに新しいＭＱＳとパーティショナが追加されるが、除去されている古いパーティションは、そのＭＱＳが任意のワークユニットを記憶しない限りＭＱＳとパーティショナを維持し得る。再パーティショニングを実施するための具体的な手順の詳細な例については、以下でより詳細に説明する。
【００６８】
これらの手順を説明する前に、代替のＭＱＳ実施形態の例が図３Ｂに示されている。この例では、ＭＱＳ４００Ａ、４００Ｂ、４００Ｃ、４００Ｄごとに、それぞれ１つの保留キュー４０２Ａ、４０２Ｂ、４０２Ｃ、４０２Ｄが存在する。図３Ａの例では、図３Ｂの

例はまた、対応する再配置キュー４０４Ａ、４０４Ｂ、４０４Ｃ、４０４Ｄのセット、対応する準備完了キュー４０６Ａ、４０６Ｂ、４０６Ｃ、４０６Ｄのセット、及びそれらの準備完了キューからワークユニットを消費するワーカープール４０８Ａ、４０８Ｂ、４０８Ｃ、４０８Ｄを含む。しかしながら、この例では、所与のシリアライゼーションキーを有する単一のワークユニットだけがいつでもワーカーによって消費され得るというシリアライゼーション制約を強制するために、各準備完了キューは、所与のシリアライゼーションキーを有する１つのワークユニットしか１度に記憶できない。したがって、ビジーキュー４１０Ａ、４１０Ｂ、４１０Ｃ、４１０Ｄ（それぞれ保留キュー４０２Ａ、４０２Ｂ、４０２Ｃ、４０２Ｄに対応する）の追加のセットが存在する。シリアルに実施されるように制約されているすべてのワークユニットがシリアライゼーションキーの同じ値を有するように、各ワークユニットには、シリアライゼーションキーが割り当てられているものとする（同じ「キーグループ」にあると言われる）。特定のワークユニットに割り当てられたパーティションを決定した後、パーティショナは、各保留キューからワークユニットを、それぞれのＭＱＳ内の３つの他のタイプのキューのうちの対応する１つに分離する。再配置される必要があるワークユニットは、依然として対応する再配置キューに移動される。しかし、再配置される必要がないワークユニットは、その準備完了キューが現在、同じキーグループ内に別のワークユニット（すなわち、同じシリアライゼーションキーを有する）を記憶していない場合にのみ、対応する準備完了キューに移動され、その準備完了キューのワーカープールは、現在、同じキーグループ内の別のワークユニットを処理していない。ワーカープールが複数のワークユニットを同時に処理できるとすると、ワークユニットが準備完了キューに入ることが許容されていれば、ワーカープールは準備完了キューから取り出し、両方のワークユニットを同じキーグループから同時に処理することができるため、ワーカープールが現在同じキーグループからのワークユニットを処理している場合、ワークユニットは準備完了キューに入れられない。パーティショナは、例えば、準備完了キュー内の異なるシリアライゼーションキーを有する、又はワーカープールによって処理されているワークユニットの存在を追跡する各シリアライゼーションキー（例えば、ビットマップ内）ごとに１又は２以上のブールフラグを使用して、これらの条件のいずれかが真であるかどうかを追跡することができる。準備完了キューが同じキーグループ内に別のワークユニットを記憶している場合、又はワーカープールが同じキーグループ内の別のワークユニットを処理している場合、ワークユニットは対応するビジーキューに移動される。パーティショナ（又は、パーティショナとは別のバックグラウンドプロセス）は、同じキーグループからのワークユニットがワーカープールによって準備完了キューから除去され、ワーカープールが同じキーグループからのワークユニットの処理を終了すると、ワークユニットをビジーキューから準備完了キューに移動する。パーティショニングルールに変更がある場合、ビジーキュー内のワークユニットは、それらの相対的順序を変更せずに、準備完了キュー内のワークユニットとともに戻りサブキューに移動される。

【００６９】

場合によっては、パーティショナによって行われた決定が単純化され得る。例えば、ラウンドロビン又はランダムパーティショニングルールを使用するコンテンツに依存しないケースでは、パーティショナがそのパーティショニングを決定（又は確認）せずに、保留キューの再循環サブキューから移動されているワークユニットが、準備完了キューに直接移動され得る。これは、最初のラウンドロビン又はランダムな決定が、ワークユニットが保留キューの保留中のサブキューから移動されたときに１度だけ行われる必要があるためである。次いで、ワークユニットが再循環サブキューに入っているとき、パーティショニングルールが変更されていない限り、その決定は依然として正しいと見なされ得る。

【００７０】

いくつかの実施形態では、データ処理システム１００のノード間でより長命の参照ユニットをパーティショニングするための技法は、より短命のワークユニットをパーティショニングするための技法とは異なる。ＭＱＳを管理するために、システム１００はワークユニットパーティショナを生成し、参照ユニットを管理するために、システム１００は、パ

ーティショニングルールに基づいて参照ユニットをパーティショニングするために適切なノードにおいて参照ユニットパーティショナを生成する（例えば、変数が、そのキーに対するタスクが生成されるコンピューティングノード１０２上に配置されることを確実にするためにコンパイラによって提供されるキー付き共有変数のキーに基づいて）。個々の参照ユニットは、一般に個々のワークユニットよりも大きい可能性があるため、ノード間で参照ユニットを転送することはより時間がかかる場合がある。しかしながら、ワーカーは、ワークユニットを消費するためにタスクを実施している間に参照ユニットにアクセスする必要があり得る。したがって、参照ユニットの動的な再パーティショニングのための技法は、参照ユニットを非同期的に移行するパーティショニング手順を含み、任意の特定の参照ユニットが古いノード、新しいノード、又は古いノードと新しいノードとの間のトランジットに配置され得る可能性を可能にする、移行中に参照ユニットにアクセスする方法を提供する。

【００７１】

例えば、パーティショニング手順中にコンピューティングノード１０２上でパーティショニングされたキー付き共有変数にアクセスするために、データ処理システム１００は、どのコンピューティングノード１０２が特定のキーを有する変数を記憶しているかを決定し、次いで、そのコンピューティングノード１０２上のその変数にアクセスするためのタスクを生成する。どのコンピューティングノード１０２が変数を記憶しているかを決定するために、パーティショニング手順は、古いパーティションと新しいパーティションの両方を計算するためにキーに適用される関数（例えば、ハッシュ関数に基づく）を提供する。共有変数にアクセスする前に、古いパーティションを決定し、その変数の古いパーティションの古いストレージ位置をチェックするために、そのキーが使用される。古いストレージ位置に変数が見つからない場合、新しいパーティションを決定し、変数の新しいパーティションの新しいストレージ位置をチェックするために、そのキーが使用される。新しいストレージ位置に変数が見つからない場合、（チェックするための手順がストレージ位置間の通過時間を考慮するために十分に長く待機する限り）存在しないと決定される。パーティショニング手順中に変数がアクセスされると、その変数はアクセスが完了するまで一時的に移行が遅れる場合がある。パーティショニング手順が完了した後、古いパーティションを計算するための関数はもはや必要なくなる。いくつかの実施形態は、複数のパーティション手順が同時に進行されるようにし、その場合、変数の存在をチェックするためのストレージ位置の数は、それが見つかるまで最も古いものから最も新しいものまで増加し、見つからない場合は存在しないと決定される。

【００７２】

参照ユニットのパーティショニング手順の変形は、参照ユニットがすべて同時に移行されない増分パーティショニング手順である。例えば、キー付き共有変数の場合、キーを所定の数のグループ（パーティショニングから独立している）に分割するために使用されるハッシュ関数によって決定されるように、参照ユニットの複数のグループが存在する。次に、グループのうちの１つにあるキーを有する変数だけが１度に移行され、そのグループにない他の任意の変数は、特定のストレージ位置（そのグループがすでに移行されたかどうかに応じて、古いストレージ位置又は新しいストレージ位置のいずれか）にあることが保証される。この方法では、移行されるグループ内のキーを有する変数だけが、変数を見つける前に両方のストレージ位置をチェックする必要がある可能性がある。したがって、増分パーティショニング手順は、移行中に参照ユニットの多くがアクセスされる場合に必要な全体的な通信を削減する。

【００７３】

図４Ａは、明示的に示されたＭＱＳ内の異なるタイプのキュー間の接続を有する２つのＭＱＳのセットの例を示す。ＭＱＳ４２０Ａ、４２０Ｂごとに、それぞれ１つの保留キュー４２２Ａ、４２２Ｂが存在する。対応する再配置キュー４２４Ａ、４２４Ｂのセット、対応する準備完了キュー４２６Ａ、４２６Ｂのセットも存在する。この例では、再配置キュー４２４Ａと保留キュー４２２Ｂとの間の接続、及び再配置キュー４２４Ｂと保留キュ

ー４２２Ａとの間の接続が明示的に示されている。準備完了キュー４２６Ａ、４２６Ｂからワークユニットをそれぞれ消費するワーカープール４０８Ａ、４０８Ｂも存在する。この例では、各ＭＱＳの保留キューにワークエレメントを提供する入来パーティショナ４３２とともに、ワークエレメントの上流ソース４３０も示されている。或いは、他の例では、各保留キューは異なるソースからワークエレメントを受信することができ、パーティショニングは、入来パーティショナ４３２を必要とせずに、各ＭＱＳに関連付けられる個々のパーティショナによって完全に達成され得る。図４Ｂは、ＭＱＳ４２０Ａ、４２０Ｂのセットの様々なキューを横断するときにワークユニット４４０が取るパスの例を示す。ワークユニット４４０は、ＭＱＳのセットを通って進むにつれて、相対時間Ｔ１、Ｔ２、Ｔ３、Ｔ４、Ｔ５、Ｔ６によってラベル付けされる。この例では、ワークユニット４４０は、（時間Ｔ２において）ＭＱＳ４２０Ｂのパーティション Ｂに最初に割り当てられるが、最終的にワーカープール４２８Ａによって処理（又は「消費」）される前に（時間Ｔ６において）、ＭＱＳ４２０Ａのパーティション Ａに動的に再パーティショニングされる。

【００７４】

データ処理システム１００はまた、データ処理プログラム内で必要とされるバッファリングの種類に基づいてＭＱＳを管理する際にパーティショナによって使用されるべきパーティショニングルールの種類を選択するように構成される。図５Ａは、内側ループ５０２及び外側ループ５０４を含む、ネストされた実行セットを有するデータ処理グラフ５００の一例を示し、その各々は、ワークユニットの入来ストリームにわたって反復するループにおいて計算を実施する。コンパイラは、システム１００の複数のコンピューティングノード１０２にわたってループをパーティショニングするためのＭＱＳのセットにそれぞれ対応するバッファコンポーネントを挿入することができる。図５Ｂは、コンパイラが、ネストされた実行セットのためのパーティショニングを扱うためにバッファコンポーネントを追加した後に、対応するデータ処理グラフ５５０が（少なくとも概念的に）どのように現れるかを示す。具体的には、バッファコンポーネント５５２Ａ、５５２Ｂ、５５２Ｃ、及び５５２Ｄの各々は、実行時に割り振られ、コンピューティングノード１０２を介して分散されるＭＱＳのそれぞれのセットを表す。いくつかの実施形態では、コンパイラによって追加されたこれらのバッファコンポーネントはユーザに表示されない。

【００７５】

この例では、ＭＱＳが提供する動的なパーティショニング機能が、コンパイラと実行時システムに、内側ループ及び外側ループを並列化するための効率的な選択を行うための柔軟性を与える。システム１００は、内側ループ及び外側ループのパーティショニングが、互換性があり効率的になるように、ＭＱＳを処理するパーティショナを構成することができる。システム１００は、パーティショニングルールがコンテンツに依存する必要があるか、又はコンテンツに依存しない必要があるかを考慮に入れることもできる。例えば、コンテンツに依存しないパーティショニングルールの場合、内側ループのオプションは、（１）外側ループパーティションのすべてのワークエレメントを、関連付けられる「ローカル」ノードに維持する（すなわち、再パーティショニングしない）こと、又は（２）すべてのノードにわたる外側ループパーティションのローカルノードからのワークエレメントをパーティショニングすることを含むことができる。オプション１は潜在的によりスケーラブルであるが、オプション２は、処理されているワークエレメントの数に応じて、潜在的により高速である。外側ループが大きい（すなわち、反復回数が多い）場合、及び内側ループが小さい（すなわち、反復回数が少ない）場合、オプション１はオプション２より高速である可能性がある。これについての１つの説明は、オプション２の場合、（小さい）内側ループのワークエレメントをパーティショニングするオーバーヘッドコストを回収するのに十分な時間がないということである。一方、外側ループが小さく、内側ループが大きい場合、オプション１はオプション２よりも高速ではない可能性がある。これについての１つの説明は、（小さい）外側ループがコンピューティングノード１０２のうちのいくつかのみパーティショニングすることができ、オプション２のパーティショニングは、（大きい）内側ループをより高速に実施することを助けるために、残りのコンピューティ

ングノード１０２が使用されることを可能にする。
【００７６】
動的なパーティショニング機能は、システム１００が実行時まで内側ループ及び外側ループの相対的なサイズに関する十分な情報を有していない可能性があるため、適切なオプションを選択するために有用であり得る。コンパイラは、特定の前提条件に基づいて、パーティショナにデフォルトのパーティショニングルールを使用し得る。例えば、内側ループの方が、それらの外側ループに比べて小さくなるという前提条件があり得る。さらに、多数のワークエレメントを生成する可能性のある特定のソース（例えば、過去の実行からの統計に基づいて）、及び／又は生成される可能性が高いワークエレメントの数を推定するために分析され得るソース（例えば、ファイルの場合、それが含むワークユニットの数を推定するためにファイルサイズが使用され得る）が存在する場合がある。次いで、実行時に、システム１００は、いつデフォルトのパーティショニングルールが変更されるべきかを動的に決定することができる。例えば、内側ループによって処理されるワークエレメントの数がしきい値（あらかじめ定められたしきい値、又は外側ループに対するしきい値）を超える場合、パーティショニングルールは、本明細書に記載の技法を使用して変更され得る。
【００７７】
上述のように、パーティショニングルールの変更への対応方法を決定する際には、複数のオプションもある。最良のオプションは、パーティショニングの性質と、ＭＱＳのセットで実装されているバッファの下流で発生する処理の順序要件の両方に依存する。例えば、以下は可能な３つのオプションである。
【００７８】
オプション１：パーティショニングルールがコンテンツに依存しない場合－つまり、すべてのワークユニットにあらかじめ定められた正しいパーティションがない場合－準備完了キュー又は再配置キューにすでに作成されたワークユニットを再パーティショニングする必要さえない。この例としては、すでにパーティショニングされているワークユニットを再パーティショニングする必要がないラウンドロビン又は負荷バランシングのパーティショニングルールがある。
【００７９】
オプション２：上記で説明したこのオプションでは、ＭＱＳ内のワークユニットが回収され、保留キューに戻される（保留キューの戻りサブキュー部分を使用していてもよい）。このオプションは、パーティショニングルールがコンテンツに依存し、下流処理が順序付けられていない場合、最も効率的なオプションである。これは、例えば、ワークユニットがキーによってパーティショニングされているが、処理は同じキーを共有するワークユニットに対して特定の順序付けを必要としなかった場合に発生する。
【００８０】
オプション３：このオプションは、パーティショニングルールがコンテンツに依存し、下流処理が順序付けされている場合に適用される。これは、例えば、ワークユニットが、キーによってパーティショニングされており、同じキーを有し、同じソースからのワークユニットが順番に処理されなければならない場合に発生する。このオプションでは、ワークユニットを再パーティショニングすることは、ワークユニットが正しいパーティショニングで処理されることを保証するために十分であるが、順序が維持されることを保証するためには不十分である。特定の部分的順序付けを保証するために、「エポック移行手順」と呼ばれる複数ステップの手順が使用され得る。この手順は、特定のＭＱＳのグループの一部であるＭＱＳを管理しているすべてのパーティショナについてパーティショニングルールが同じである期間に対応する「パーティショニングエポック」間の移行のために使用される。隣接するパーティショニングエポックの間には、エポック移行手順が発生する「移行期間」がある。場合によっては、１又は２以上のパーティショナが新しいパーティショニングルールを使用し始めるか、依然として古いパーティショニングルールを使用している移行期間中に、いくつかのパーティショニングが依然として発生している場合がある

。１つのパーティショニングエポックから次のパーティショニングエポックへの進展を調整することによって、システム１００は、以下でより詳細に説明するように、パーティショニングルールの変更にわたって部分的順序付けを保存することができる。

【００８１】

パーティショナがＭＱＳを管理する方法は、移行期間ではなくパーティショニングエポックの間に異なる場合がある。パーティショニングエポックの間、再循環サブキューに入るワークユニットのパーティションは、以前のコンピューティングノードで計算されたパーティションが正しいため、新しいコンピューティングノードにおいて再計算される必要はないが、新しいコンピューティングノードにおいて計算されたパーティションと必ずしも同じではない。一方、移行期間中に、再循環サブキューに入るワークユニットのパーティションが再計算される。例えば、ランダムパーティショニングの場合、同じパーティショニングエポックの間の第２のパーティション決定は、最初にランダムに選択されるパーティションに再配置されたばかりのワークユニットには不適切である。その代わりに、再循環サブキューに入る任意のワークユニットは、単に準備完了キューに直接移動され得る（１つずつ、又は複数のワークユニットのグループにおいて）。

【００８２】

ＭＱＳは、ネットワーク１０４を介してコンピューティングノード１０２間にワークユニットを分散する。データ処理プログラムは、ワークユニットをＭＱＳに注入する１又は２以上のソース（例えば、データ処理グラフの処理コンポーネント又はソースコンポーネントに対応する）と、ＭＱＳからワークユニットを消費する１又は２以上のワーカープール（例えば、データ処理グラフの処理コンポーネント又はシンクコンポーネントに対応する）を定義する。パーティショナが同じパーティショニングルールを使用するＭＱＳの特定のセットは、コンピューティングノード１０２の特定のサブネットワーク上でのパーティショニングを管理すると考えられる。このＭＱＳのセットは、例えば、上述したように、データ処理グラフ内のバッファコンポーネントとして表され得る。ＭＱＳのセットによって使用されるサブネットワークを介したルーティングは、それらの共通のパーティショニングルール、及び順序制約などのデータ処理プログラムの他の制約によって決定される。ＭＱＳの特定のセットは、それらのルーティング制約に基づいて、３つのタイプのうちの１つを有するものとして分類され得る。

・タイプ１：最小限の制約付きルーティングにより、ワークユニットが任意のワーカープールによって処理されることが可能になる。データ処理プログラムは、ワークユニットを特定のワーカープール（又は、特定のコンピューティングノード１０２）にルーティングするためにＭＱＳのセットを必要とせず、同じソースからのワークユニットが互いに順番に処理される必要はない。したがって、ＭＱＳを管理するパーティショナは、ラウンドロビンパーティショニング又は負荷バランシングパーティショニングなどのパーティショニングルールを使用することができる。

・タイプ２：いくつかのデータ処理プログラムは、ワークユニットから導出されたキー値に基づいて、ワークユニットが特定のワーカープール（又は、特定の計算ノード１０２）によって処理されることを必要とするが、同じソースからのワークユニットがワーカープールにおいて順番に処理されることを必要としない。ＭＱＳを管理するパーティショナは、キーベースのパーティショニングルールを使用する。

・タイプ３：最も制約の厳しいルーティングは、ワークユニットが、ワークユニットから導出されたキー値に基づいて、特定のワーカープール（又は、特定のコンピューティングノード１０２）によって処理されることを必要とし、また、同じキーを有する同じソースからのワークユニットが、それらがソースにおいて生成されたのと同じ順序でワーカープールにおいて処理されることを必要とする。ＭＱＳを管理するパーティショナは、キーベースのパーティショニングルールを使用し、順序が保存されることを保証するために追加のステップを実行する。

【００８３】

場合によっては、サブネットワークが動作している間にパーティショニングルールを変

更する必要がある。これは、ワーカープールをあるコンピューティングノード１０２から別のコンピューティングノード１０２に移動させるために、又はワークユニットのキーとそれを処理するワーカープールとの間のマッピングを変更するために、ワーカープールを追加又は除去するために行われ得る。パーティショニングルールが変更されると、特定のパーティションに関連付けられるＭＱＳに記憶されたワークユニットが、新しいパーティショニングルールに従って誤ったパーティションに存在する可能性がある。それらのワークユニットは、それらの新しくて正しいパーティションに移動される必要があり、順序を保存するために特定のステップが実行される。

【００８４】

パーティショニングエポック内では、どのワークユニットもそのキーを１度だけ調べることによってその宛先にルーティングされ得る。パーティショナは、ワークユニットを（保留キューの）保留中のサブキューからそのＭＱＳの準備完了キュー又は再配置キューのいずれかに移動させるときに、ワークユニットのキーを調べるだけでよい。ワークユニットが、現在のパーティショニングエポックの全体にわたって有効であった同じパーティショニングルールに基づいてこのＭＱＳにルーティングされたことがわかっているため、（保留キューの）再循環サブキューに到着するワークユニットは、それらのキーを再検査せずに、準備完了キューに直接移動され得る。

【００８５】

１つのワークユニットがソースＡからワーカープールＰまでのサブネットワークを通るパスをたどる場合、ソースＡにおいて発生し、同じパーティショニングエポックの間にワーカープールＰによって消費されるすべてのワークユニットは、ＡからＰまでの同じパスに従う。ＡからＰに送信されるすべてのワークユニットは同じパスに従い、パーティショナは移行中のワークユニットの相対的順序を保存するように構成されているため、ソースにおいてワークユニットが生産された順序は、ソースとワーカープールのすべてのペアについて、ワーカープールによって消費された順序で保存される。異なるソースにおいて発生するか、又は異なるワーカープールによって消費される２つのワークユニットは、必然的にサブネットワークを通る異なるパスを取るため、それらのワークユニット間には順序は画定されていない。

【００８６】

セット内のＭＱＳを管理しているパーティショナのうちの少なくとも１つに対してパーティショニングルールが変更されると、現在のパーティショニングエポックが終了し、新しいパーティショニングエポックへの移行期間が開始される。移行期間中、順序が保存されていることを保証する不変量が侵害されることが可能になる。データ処理プログラムの順序要件及びパーティショニング要件に応じて、新しいパーティショニングエポックに安全に移行するために、異なる手法が必要である。

【００８７】

単純で低コストの移行戦略は、再循環サブキュー、再配置キュー、又は準備完了キュー内にすでにある任意のワークユニットを再処理するために時間を費やすことなしに、新しいパーティショニングルールを使用するようにパーティショナを単純に更新することである。この戦略は、タイプ１のＭＱＳセットに対してのみ許容される。パーティショニングエポックＮの間にキーが検査された、いくつかのワークユニットは、エポックＮ＋１の間に検査された場合にルーティングされたワーカープールとは異なるワーカープールによって、パーティショニングエポックＮ＋１の間に消費されるが、これは、特定のプールへのワークユニットのルーティングは、そのタイプのＭＱＳセットの要件ではないため、タイプ１のＭＱＳセットについて許容される。

【００８８】

よりコストの高い移行戦略は、移行期間中にワークユニットを再処理することを含む。タイプ２とタイプ３のＭＱＳセットは、現在のパーティショニングルールに従ってワークユニットのキーから決定される特定のワーカープールによってワークユニットが消費されることを必要とする。パーティショニングルールが変更されると（移行期間中に）、ワー

クユニットの正しい宛先ワーカープールも変更されることがある。したがって、パーティショナはワークユニットを再処理し、これは、新しいパーティショニングルールに従って、ワークユニットキーを再検査することと、必要に応じてワークユニットをそれらの正しい宛先に潜在的に再ルーティングすることとを含む。順序が保存されるか（タイプ３）、否か（タイプ２）によって、移行期間中の再処理を処理するための２つの方法がある。

【００８９】

タイプ２のＭＱＳセットの場合、ワークユニットの再検査と潜在的な再ルーティングが必要であるが、順序を保存することは必要ない。この場合、セット内のすべてのＭＱＳのすべての準備完了キュー、再循環サブキュー、及び再配置キュー内のワークユニットを回収し、それらを適切な保留中のサブキューに挿入することによって、ワークユニットが再処理され得る。この再処理の後、次のパーティショニングエポックにおいて、新しいパーティショニングルールを使用してワークユニットがパーティショニングされ、正しい宛先ワーカープールにおいて消費される。

【００９０】

以下は、それぞれのＭＱＳ上で各パーティショナによって実施される、タイプ２のＭＱＳセットのタイプ２エポック移行手順（ルーティングエポックＮからＮ＋１まで）の例である。

・あらゆるキュー（又はサブキュー）からワークユニットの処理を中断する。

・再循環サブキュー、再配置キュー、及び準備完了キューからワークユニットを保留中のサブキューに回収する。

・エポックＮ＋１をパーティショニングするためのパーティショニングルールに従ってパーティショニングを切り替える。

・すべてのキュー（及びサブキュー）からワークユニットの処理を再開する。

【００９１】

パーティショニングエポックＮの間に送信されたが、パーティショニングエポックＮ＋１までに配信されなかったワークユニットは、パーティショニングエポック移行の前にそれらがすでに再配置されたか（再配置キューから再循環サブキューに）どうかに応じて、サブネットワークを通る２つの異なるパスに従い得るので、順序付けは、このエポック移行手順において保存されることが保証されていない。ワークユニットが両方のパスに沿って宛先ワーカープールに同時に収束すると、それらがインターリーブされ得、この場合、相対的順序が失われることになる。このため、この手順はタイプ２のＭＱＳセットでは受け入れられるが、タイプ３のＭＱＳセットでは受け入れられない。

【００９２】

２つのＭＱＳ－ＭＱＳ１とＭＱＳ２－のＭＱＳセットの場合、ワークユニットの状態は、すべて同じ特定のキーＫを有し、パーティショニングエポックＮの間は、ＭＱＳ１を供給するソースＳから同じパスをたどるように、パーティション２に割り当てられ、次いでＭＱＳ２に再配置され、次いでＭＱＳ２においてワーカープールによって消費されると見なす。この例では、パーティショニングエポックＮの終了時のキューのコンテンツは以下の通りであり、ソースＳから提供された順序に従ってレコード番号Ｒ＃によってラベル付けされたワークユニットを有し、ワークユニットは左側の先頭から除去されて、右側の末尾に追加され、空白のリストは空のキュー又はサブキューを示す。

ＭＱＳ　１　　保留中のサブキュー：Ｒ７、Ｒ８
　　再循環サブキュー：
　　再配置キュー：Ｒ６
　　準備完了キュー
ＭＱＳ　２　　保留中のサブキュー：
　　再循環サブキュー：Ｒ４、Ｒ５
　　再配置キュー：
　　準備完了キュー：Ｒ３

【００９３】

次いで、エポック移行手順の後、各ＭＱＳ内のすべてのワークユニットが回収され、保留中のサブキューに移動され、以下のようになる。

ＭＱＳ　１　　保留中のサブキュー：Ｒ６、Ｒ７、Ｒ８
　　再循環サブキュー：
　　再配置キュー：
　　準備完了キュー
ＭＱＳ　２　　保留中のサブキュー：Ｒ３、Ｒ４、Ｒ５
　　再循環サブキュー：
　　再配置キュー：
　　準備完了キュー

【００９４】

エポックＮ＋１をパーティショニングするための新しいパーティショニングルールに基づいて、キーＫを有するワークユニットは、ＭＱＳ２におけるワーカープールではなく、ＭＱＳ１においてワーカープールによって消費されるべきパーティション１に割り当てられ、したがって、パーティショニングエポックＮ＋１の間、ワークユニットはＭＱＳ２の再配置キューに移動され、次いで、ＭＱＳ１の再循環サブキューに送られ、最終的にそのワーカープールによって消費されるためにＭＱＳ１の準備完了キューに移動される。それらのワークユニットはすべてＭＱＳ１準備完了キューにおいて終了するが、ワークユニットの相対的順序は保証されない。以下は、ＭＱＳ１及びＭＱＳ２のパーティショナがいくつかのワークユニットを処理した後の有効な状態である。

ＭＱＳ　１　　保留中のサブキュー：Ｒ８
　　再循環サブキュー：Ｒ４
　　再配置キュー：
　　準備完了キュー：Ｒ６、Ｒ３、Ｒ７（ワークユニットが順番通りでない）
ＭＱＳ　２　　保留中のサブキュー：
　　再循環サブキュー：
　　再配置キュー：Ｒ５
　　準備完了キュー：

【００９５】

タイプ３のＭＱＳセットは、パーティショニングエポック移行にわたって順序付けが保存されることを必要とする。これを保証するために、エポックＮの間に処理されたが消費されなかったすべてのワークユニットが、任意のワークユニットがエポックＮ＋１の間に処理され得る前に、新しい最終宛先において再循環キュー又は準備完了キューに順番に配信される必要がある。

【００９６】

以下は、それぞれのＭＱＳ上の各パーティショナによって実施されるタイプ３のＭＱＳセットのタイプ３のエポック移行手順（ルーティングエポックＮからＮ＋１まで）の例であり、必要な順序保証を実現している。

・準備完了キューからワークユニットを消費することを中断する（依然として新しいワークユニットを受信することができる）。

・保留中のサブキューを２つのキューに分割する。エポックＮ＋１の保留中のサブキューは、保留中のサブキューにあったすべてのワークユニットを含み、新しく到着した作業ユニットが追加され得るが、そこからワークユニットは除去されない。エポックＮの保留中のサブキューは空になり始めるが、１度回収されたワークユニットがそこに置かれると、再検査され、潜在的に再ルーティングされ得る。

・すべての再配置キューが完全に排出されるまで、エポックＮをパーティショニングするためのパーティショニングルールに従って処理を続行する。

【００９７】

この時点で、同じキーを有するすべてのワークユニットは、同じＭＱＳの再循環サブキ

ュー又は準備完了キューのいずれかにまとめられる。それらは、異なるキーを有するワークユニット、又は異なるソースからの同じキーを有する他のワークユニットと混在され得るが、同じソースからの同じキーを有するワークユニットの相対的順序は保存される。

・エポックＮ＋１をパーティショニングするためのパーティショニングルールに従ってパーティショニングに切り替える。

・ワークユニットを再循環サブキュー及び準備完了キューからエポックＮの保留中のサブキューに回収し、それらの元の順序付けを保存する（同じソースにおいて発生したワークユニットに対して）。

・準備完了キューからワークユニットを消費することを再開する。

・すべてのエポックＮの保留中のサブキューが完全に排出され、再配置キューが空になるまで待機する。

【００９８】

この時点では、すべてのワークユニットが新しいパーティショニング機能のためにそれらの正しい宛先にあり、順序はパーティショナによって保存される。パーティショナは、順序を保存するために、回収プロセス中に特定のルールに従うことができる。例えば、各パーティショナが、先頭における準備完了キューからのワークユニット、次に再循環サブキューからのワークユニット、及び末尾における保留中のサブキューにおける元のワークユニット（回収される際に並べ替えられていないワークユニット）、を有するワークユニットをエポックＮの保留中のサブキューに戻すと、順序制約が保存され、したがって、回収プロセスが完了した後、同じソースからの、及び同じキーグループ内のワークユニットが依然として順番に並ぶようになる。相互に並べ替えられた可能性のある唯一のワークユニットは、異なる保留中のサブキュー又は再循環サブキューから来たワークユニットであり、それらが元々異なるソースから来たものであるか、異なるキーを有するため異なるＭＱＳに送られた場合にのみ起こりうる。いずれにしても、ワークユニットは、画定された順序制約を有するために同じソースと同じキーの両方を有していなければならないので、その並べ替えが許容される。

・空のエポックＮの保留中のサブキューを破棄し、パーティショナがパーティショニングエポックＮ＋１に完全に移行したので、現在は通常の保留中のサブキューであるエポックＮ＋１の保留中のサブキューからの読取りを再開する。

【００９９】

以下は、タイプ３のエポック移行手順が、タイプ２のエポック移行手順の例において上記で使用されたものと同じ開始ワークユニット設定を扱う方法であり、ここで、パーティショニングエポックＮの終了時のキューのコンテンツは以下の通りである。

ＭＱＳ　１　　保留中のサブキュー：Ｒ７、Ｒ８
　　再循環サブキュー：
　　再配置キュー：Ｒ６
　　準備完了キュー：
ＭＱＳ　２　　保留中のサブキュー：
　　再循環サブキュー：Ｒ４、Ｒ５
　　再配置キュー：
　　準備完了キュー：Ｒ３

【０１００】

保留中のサブキューを分割した後、コンテンツは次の通りである。

ＭＱＳ　１　　保留中のサブキューＮ：
　　保留中のサブキューＮ＋１：Ｒ７、Ｒ８
　　再循環サブキュー：
　　再配置キュー：Ｒ６
　　準備完了キュー：
ＭＱＳ　２　　保留中のサブキューＮ：
　　保留中のサブキューＮ＋１：

再循環サブキュー：Ｒ４、Ｒ５
再配置キュー：
準備完了キュー：Ｒ３

【０１０１】
次いで、パーティショナは、再配置キューが排出されるまで（ワークユニットＲ４が再循環サブキューの先頭から準備完了キューの末尾に移動することを許容された状態で）、エポックＮをパーティショニングするためのパーティショニングルールで処理を続行する。

ＭＱＳ　１　保留中のサブキューＮ：
保留中のサブキューＮ＋１：Ｒ７、Ｒ８
再循環サブキュー：
再配置キュー：
準備完了キュー：
ＭＱＳ　２　保留中のサブキューＮ：
保留中のサブキューＮ＋１：
再循環サブキュー：Ｒ５、Ｒ６
再配置キュー：
準備完了キュー：Ｒ３、Ｒ４

【０１０２】
再配置キューが空になった後、再循環サブキュー及び準備完了キューは、エポックＮの保留中のサブキューに回収され得る。

ＭＱＳ　１　保留中のサブキューＮ：
保留中のサブキューＮ＋１：Ｒ７、Ｒ８
再循環サブキュー：
再配置キュー：
準備完了キュー：
ＭＱＳ　２　保留中のサブキューＮ：Ｒ３、Ｒ４、Ｒ５、Ｒ６
保留中のサブキューＮ＋１：
再循環サブキュー：
再配置キュー：
準備完了キュー：

【０１０３】
次いで、パーティショナは、エポックＮ＋１パーティショニングルールをパーティショニングするためのパーティショニングルールを使用して正常に処理を再開し、完全に排出され、再配置キューが空になるまで、エポックＮの保留中のサブキューからワークユニットを除去し、これは最終的に以下の状態になる（ワークユニットＲ３～Ｒ５は、再循環サブキューから準備完了キューに移動することができる）。

ＭＱＳ　１　保留中のサブキューＮ：
保留中のサブキューＮ＋１：Ｒ７、Ｒ８
再循環サブキュー：Ｒ６
再配置キュー：
準備完了キュー：Ｒ３、Ｒ４、Ｒ５
ＭＱＳ　２　保留中のサブキューＮ：
保留中のサブキューＮ＋１：
再循環サブキュー：
再配置キュー：
準備完了キュー：

【０１０４】
すべてのワークユニットが正しい宛先ＭＱＳに再配置され、エポックＮの保留中のサブキューと再配置キューとが空になった後、パーティショナは空のエポックＮの保留中のサ

ブキューを破棄して、エポックＮ＋１の保留中のサブキューを唯一の保留中のサブキューとして使用して正常に処理を続行することができる。

ＭＱＳ　１　　保留中のサブキュー：Ｒ７、Ｒ８
　　再循環サブキュー：Ｒ６
　　再配置キュー：
　　準備完了キュー：Ｒ３、Ｒ４、Ｒ５
ＭＱＳ　２　　保留中のサブキュー：
　　再循環サブキュー：
　　再配置キュー：
　　準備完了キュー：

【０１０５】

この時点で、パーティショナは新しいパーティショニングエポックに正常に移行し、ワークユニットを正しい宛先と正しい順序で有している。

【０１０６】

上述の動的なパーティショニング手法は、例えば、適切なソフトウェア命令を実行するプログラム可能なコンピューティングシステムを使用して実装されてもよく、フィールドプログラマブルゲートアレイ（ＦＰＧＡ，field-programmable gate array）などの適切なハードウェア又はある種のハイブリッド形式で実装されてもよい。例えば、プログラムされた手法では、ソフトウェアは、１又は２以上のプログラムされた又はプログラム可能なコンピューティングシステム（分散型、クライアント／サーバ型、又はグリッドなどの様々なアーキテクチャのものであり得る）で実行される１又は２以上のコンピュータプログラムにおける手順を含み得、それぞれが、少なくとも１つのプロセッサと、少なくとも１つのデータストレージシステム（揮発性及び／又は不揮発性メモリ並びに／或いはストレージエレメントを含む）と、少なくとも１つのユーザインターフェース（少なくとも１つの入力デバイス又はポートを使用して入力を受信するための、及び少なくとも１つの出力デバイス又はポートを使用して出力を提供するための）とを含む。ソフトウェアは、例えば、データフローグラフの設計、構成、及び実行に関連するサービスを提供する、より大きなプログラムの１又は２以上のモジュールを含み得る。プログラムのモジュール（例えば、データフローグラフのエレメント）は、データ構造又はデータリポジトリに記憶されたデータモデルに適合する他の組織化データとして実装され得る。

【０１０７】

ソフトウェアは、媒体の物理的特性（例えば、表面ピット及びランド、磁区、又は電荷）を使用して、一定期間（例えば、ダイナミックＲＡＭなどのダイナミックメモリデバイスのリフレッシュ期間の間の時間）、揮発性又は不揮発性ストレージ媒体、或いは他の任意の非一時的媒体に具体化されるような非一時的形態で記憶され得る。命令のロードの準備において、ソフトウェアは、ＣＤ－ＲＯＭ又は他のコンピュータ可読媒体（例えば、汎用又は専用コンピューティングシステム或いはデバイスによって読取り可能）などの有形の非一時的媒体上に提供されてもよく、ネットワークの通信媒体を介して、それが実行されるコンピューティングシステムの有形の非一時的媒体に配信（例えば、伝搬された信号で符号化）されてもよい。処理の一部又は全部は、専用コンピュータ上で、或いはコプロセッサ又はフィールドプログラマブルゲートアレイ（ＦＰＧＡ）或いは専用のデータ処理グラフ専用集積回路（ＡＳＩＣｓ）などの専用ハードウェアを使用して実施され得る。処理は、ソフトウェアによって指定された計算の異なる部分が異なるコンピューティングエレメントによって実施される分散方式で実装され得る。そのようなコンピュータプログラムの各々は、本明細書に記載される処理を実施するためにコンピュータによってストレージデバイス媒体が読み取られたときに、コンピュータを構成し、動作させるために、汎用又は専用のプログラム可能コンピュータによってアクセス可能なストレージデバイスのコンピュータ可読ストレージ媒体（例えば、固体メモリ又は媒体、或いは磁気又は光学媒体）に記憶される、又はダウンロードされることが好ましい。本発明のシステムは、コンピュータプログラムで構成された有形の非一時的媒体として実装されると考えられてもよく

、そのように構成された媒体は、コンピュータに本明細書に記載される処理ステップのうちの１又は２以上を実施するために特定のあらかじめ画定された方法で動作させる。

【０１０８】

本発明の多くの実施形態について説明した。それにもかかわらず、前述の説明は、本発明を説明することが意図されるものであり、本発明の範囲を限定するものではないことが理解されるべきであり、本発明の範囲は、特許請求の範囲によって画定される。したがって、他の実施形態も以下の特許請求の範囲内にある。例えば、本発明の範囲から逸脱することなしに、様々な変更が行われてよい。さらに、上述のステップのうちのいくつかは、順序に依存しないものであってよく、したがって、記載された順序とは異なる順序で実施され得る。

【図１】

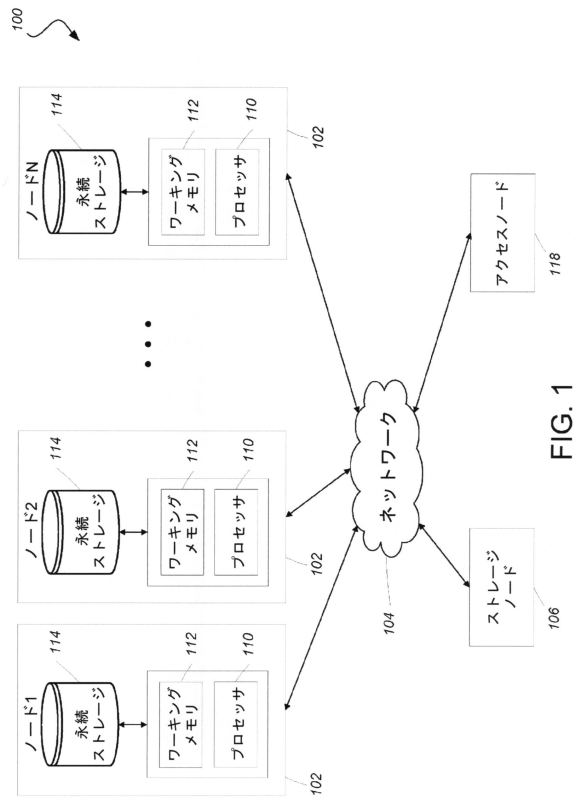

FIG. 1

【図２】

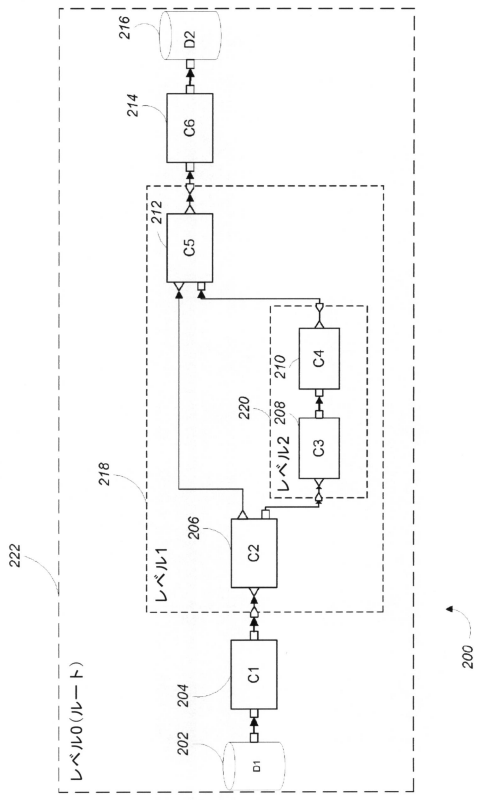

FIG. 2

【図３Ａ】

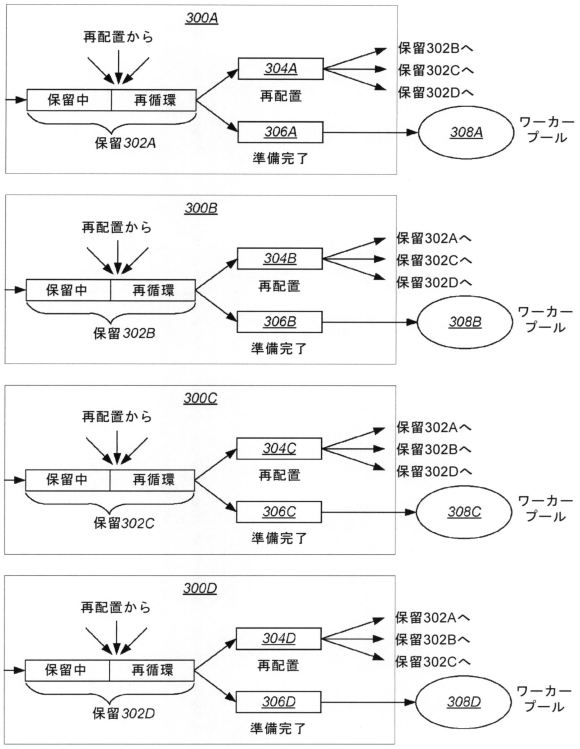

FIG. 3A

【図３Ｂ】

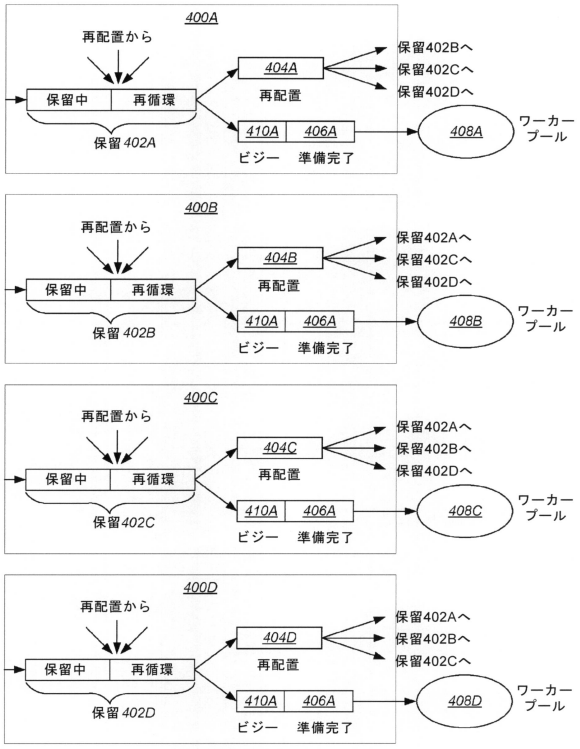

FIG. 3B

【図４Ａ】

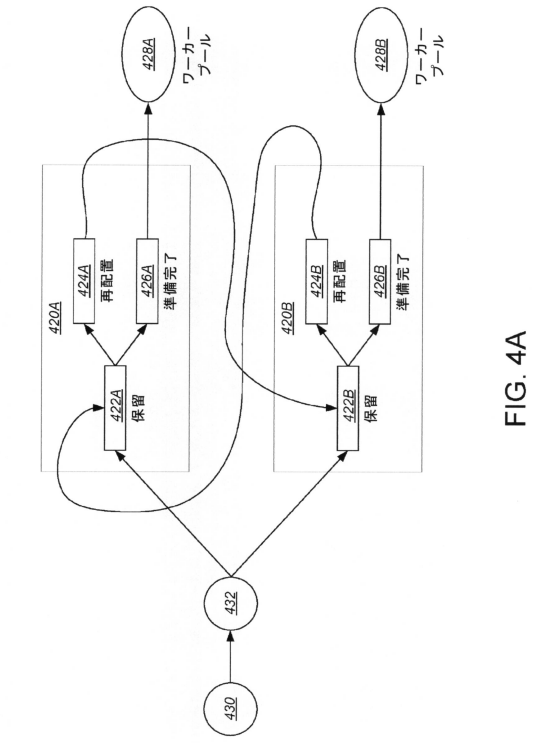

FIG. 4A

【図４Ｂ】

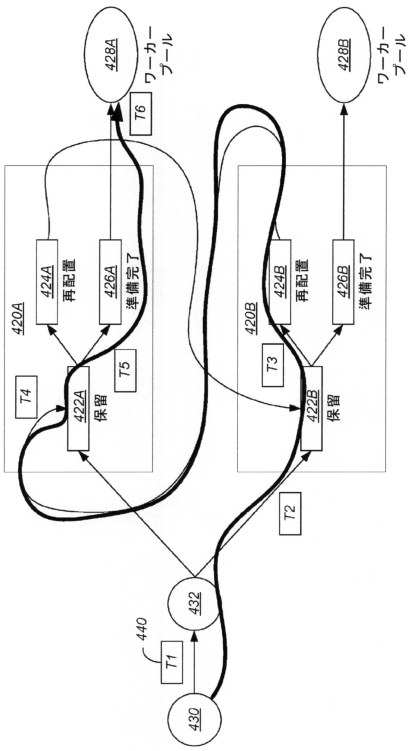

FIG. 4B

【図５Ａ】

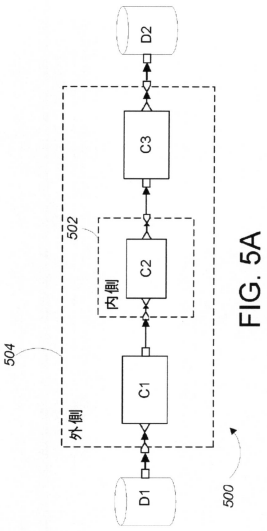

FIG. 5A

【図５Ｂ】

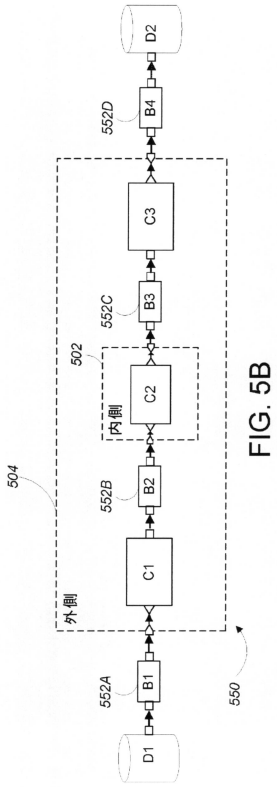

FIG. 5B

フロントページの続き

(74)代理人 100131093
弁理士 堀内 真
(74)代理人 100150902
弁理士 山内 正子
(74)代理人 100141391
弁理士 園元 修一
(74)代理人 100198074
弁理士 山村 昭裕
(74)代理人 100145920
弁理士 森川 聡
(74)代理人 100096013
弁理士 富田 博行
(72)発明者 ニューバーン ジェフェリー
アメリカ国 ０１７７６ マサチューセッツ サドベリー ケンドールロード２０
(72)発明者 スタンフィル クレイグ ダブリュー.
アメリカ国 ０１７７３ マサチューセッツ リンカン ハックルベリーロード４３

審査官 漆原 孝治

(56)参考文献 特開２００６－２５９８１２（ＪＰ，Ａ）
特開２００８－１５９０８１（ＪＰ，Ａ）
特開２０１３－２３９１９９（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／５４
Ｇ０６Ｆ ９／５０