

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第6199812号  
(P6199812)

(45) 発行日 平成29年9月20日 (2017.9.20)

(24) 登録日 平成29年9月1日 (2017.9.1)

(51) Int.Cl.

F I

G O 6 F 17/30 (2006.01)

G O 6 F 17/30 4 1 9 B

G O 6 F 9/50 (2006.01)

G O 6 F 9/46 4 6 5 E

請求項の数 15 (全 19 頁)

(21) 出願番号 特願2014-125823 (P2014-125823)  
 (22) 出願日 平成26年6月18日 (2014.6.18)  
 (65) 公開番号 特開2015-11709 (P2015-11709A)  
 (43) 公開日 平成27年1月19日 (2015.1.19)  
 審査請求日 平成29年6月19日 (2017.6.19)  
 (31) 優先権主張番号 13/932, 377  
 (32) 優先日 平成25年7月1日 (2013.7.1)  
 (33) 優先権主張国 米国 (US)

早期審査対象出願

(73) 特許権者 502096543  
 パロ・アルト・リサーチ・センター・イン  
 コーポレーテッド  
 Palo Alto Research  
 Center Incorporated  
 アメリカ合衆国、カリフォルニア州 94  
 304、パロ・アルト、コヨーテ・ヒル・  
 ロード 3333  
 (74) 代理人 100079049  
 弁理士 中島 淳  
 (74) 代理人 100084995  
 弁理士 加藤 和詳

最終頁に続く

(54) 【発明の名称】 明示的に表されたグラフで並列探索を行うシステムおよび方法

(57) 【特許請求の範囲】

【請求項 1】

消費者に製品を推薦するコンピュータで実行可能な方法であって、

1 つ以上のプロセッサにより、源点、終点及びグラフの辺を表すデータを 1 つ以上のコン  
 ピューティングデバイスのメモリにコピーするステップであって、各消費者を表す源点  
 から各製品を表す終点までを指す各辺が、前記各消費者が前記各製品を消費する旨を示す  
 ステップと、

終点に基づいて前記グラフの辺を分割して複数の後方分割グラフを生成するステップと

、

源点に基づいて前記グラフの辺を分割して複数の前方分割グラフを生成するステップと

10

、

並列計算を行う複数のプロセッサにより、各プロセッサによって前記消費者を表す源点  
 から 1 つ以上の終点まで後方分割グラフを検討することを含み、前記消費者が消費する 1  
 つ以上の製品を決定するステップであって、各終点が、前記消費者が消費する製品を表す  
 ステップと、

並列計算を行う前記複数のプロセッサにより、各プロセッサによって各終点から各源点  
 まで前方分割グラフを検討することを含み、前記消費者がまた消費する少なくとも 1 つの  
 製品を消費する 1 以上の他の消費者を決定するステップであって、各源点が、前記消費者  
 がまた消費する製品を消費する他の消費者を表すステップと、

前記 1 以上の他の消費者が消費する 1 つ以上の他の製品を決定するステップと、

20

前記 1 つ以上の他の製品のうちのいずれか 1 つを消費するように前記消費者に推薦を提供するステップと

を備える、方法。

【請求項 2】

前記並列計算が、 $n$  ビットの並列幅優先探索を実行することを備え、 $n$  ビットが隔たりの次数又は頂点を表し、 $n < 8$  である、請求項 1 に記載の方法。

【請求項 3】

第 1 の範囲において頂点を有する第 1 の後方分割グラフを第 1 のプロセッサに割り当てるステップと、

第 2 の範囲において頂点を有する第 2 の後方分割グラフを第 2 のプロセッサに割り当てるステップと

10

をさらに備え、前記第 1 の範囲及び前記第 2 の範囲が異なる範囲である、請求項 1 に記載の方法。

【請求項 4】

並列計算を行う前記複数のプロセッサのうちの 2 つのプロセッサが、同じ頂点について動作しない、請求項 1 に記載の方法。

【請求項 5】

終点に基づいて前記グラフの辺を分割するステップが、  
前記終点を分割するための複数の分割を決定すること、  
前記複数の分割の中から前記終点の範囲を分割すること  
をさらに備える、請求項 1 に記載の方法。

20

【請求項 6】

コンピュータによって実行されたときに消費者に製品を推薦する方法を前記コンピュータに実行させる命令を格納する持続性コンピュータ可読媒体であって、

1 つ以上のプロセッサにより、源点、終点及びグラフの辺を表すデータを 1 つ以上のコンピューティングデバイスのメモリにコピーすることであって、各消費者を表す源点から各製品を表す終点までを指す各辺が、前記各消費者が前記各製品を消費する旨を示すこと

、  
終点に基づいて前記グラフの辺を分割して複数の後方分割グラフを生成すること、  
源点に基づいて前記グラフの辺を分割して複数の前方分割グラフを生成すること、

30

並列計算を行う複数のプロセッサにより、各プロセッサによって前記消費者を表す源点から 1 つ以上の終点まで後方分割グラフを検討することを含み、前記消費者が消費する 1 つ以上の製品を決定することであって、各終点が、前記消費者が消費する製品を表すこと

、  
並列計算を行う前記複数のプロセッサにより、各プロセッサによって各終点から各源点まで前方分割グラフを検討することを含み、前記消費者がまた消費する少なくとも 1 つの製品を消費する 1 以上の他の消費者を決定することであって、各源点が、前記消費者がまた消費する製品を消費する他の消費者を表すこと、

前記 1 以上の他の消費者が消費する 1 つ以上の他の製品を決定すること、

前記 1 つ以上の他の製品のうちのいずれか 1 つを消費するように前記消費者に推薦を提供すること

40

を備える、持続性コンピュータ可読媒体。

【請求項 7】

前記並列計算が、 $n$  ビットの並列幅優先探索を実行することを備え、 $n$  ビットが隔たりの次数又は頂点を表し、 $n < 8$  である、請求項 6 に記載の持続性コンピュータ可読媒体。

【請求項 8】

前記方法が、

第 1 の範囲において頂点を有する第 1 の後方分割グラフを第 1 のプロセッサに割り当てること、

第 2 の範囲において頂点を有する第 2 の後方分割グラフを第 2 のプロセッサに割り当て

50

ること

をさらに備え、前記第 1 の範囲及び前記第 2 の範囲が異なる範囲である、請求項 6 に記載の持続性コンピュータ可読媒体。

【請求項 9】

並列計算を行う前記複数のプロセッサのうちの 2 つのプロセッサが、同じ頂点について動作しない、請求項 6 に記載の持続性コンピュータ可読媒体。

【請求項 10】

終点に基づいて前記グラフの辺を分割することが、  
前記終点を分割するための複数の分割を決定すること、  
前記複数の分割の中から前記終点の範囲を分割すること  
をさらに備える、請求項 6 に記載の持続性コンピュータ可読媒体。

10

【請求項 11】

消費者に製品を推薦するコンピューティングシステムであって、  
1 つ以上のプロセッサと、  
前記 1 つ以上のプロセッサによって実行されたときに方法を前記 1 つ以上のプロセッサに実行させる命令をその中に格納した、前記 1 つ以上のプロセッサに接続された持続性コンピュータ可読媒体であって、前記方法が、  
1 つ以上のプロセッサにより、源点、終点及びグラフの辺を表すデータを 1 つ以上のコンピューティングデバイスのメモリにコピーすることであって、各消費者を表す源点から各製品を表す終点までを指す各辺が、前記各消費者が前記各製品を消費する旨を示すこと

20

、  
終点に基づいて前記グラフの辺を分割して複数の後方分割グラフを生成すること、  
源点に基づいて前記グラフの辺を分割して複数の前方分割グラフを生成すること、  
並列計算を行う複数のプロセッサにより、各プロセッサによって前記消費者を表す源点から 1 つ以上の終点まで後方分割グラフを検討することを含み、前記消費者が消費する 1 つ以上の製品を決定することであって、各終点が、前記消費者が消費する製品を表すこと  
、  
並列計算を行う前記複数のプロセッサにより、各プロセッサによって各終点から各源点まで前方分割グラフを検討することを含み、前記消費者がまた消費する少なくとも 1 つの製品を消費する 1 以上の他の消費者を決定することであって、各源点が、前記消費者がまた消費する製品を消費する他の消費者を表すこと、  
前記 1 以上の他の消費者が消費する 1 つ以上の他の製品を決定すること、  
前記 1 つ以上の他の製品のうちのいずれか 1 つを消費するように前記消費者に推薦を提供すること  
を備える、持続性コンピュータ可読媒体と、を含むコンピューティングシステム。

30

【請求項 12】

前記並列計算が、 $n$  ビットの並列幅優先探索を実行することを備え、 $n$  ビットが隔たりの次数又は頂点を表し、 $n < 8$  である、請求項 11 に記載のコンピューティングシステム。

【請求項 13】

前記方法が、  
第 1 の範囲において頂点を有する第 1 の後方分割グラフを第 1 のプロセッサに割り当てること、  
第 2 の範囲において頂点を有する第 2 の後方分割グラフを第 2 のプロセッサに割り当てること、  
をさらに備え、  
前記第 1 の範囲及び前記第 2 の範囲が異なる範囲である、  
請求項 11 に記載のコンピューティングシステム。

40

【請求項 14】

並列計算を行う前記複数のプロセッサのうちの 2 つのプロセッサが、同じ頂点について

50

動作しない、請求項 11 に記載のコンピューティングシステム。

【請求項 15】

終点に基づいて前記グラフの辺を分割することが、  
前記終点を分割するための複数の分割を決定すること、  
前記複数の分割の中から前記終点の範囲を分割すること  
をさらに備える、請求項 11 に記載の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】

本開示は、グラフ検索に関し、より具体的には、明示的に表されたグラフで並列処理を行う方法およびシステムに関する。 10

【発明の概要】

【課題を解決するための手段】

【0002】

本発明の一実施形態では、顧客の購買行動を表して並列計算を容易にするグラフを分割するシステムが提供される。動作中、このシステムは最初に、グラフの頂点および辺を示すグラフデータを受信する。このデータでは、頂点が顧客および製品を表し、辺が購買行動を表す。次に、このシステムは、グラフの辺を分割して、グラフの各辺が各区画の節となるように分割グラフ生成する。次いで、このシステムは、1つ以上の区画内のグラフデータ上で並列計算を行って推薦する製品を決定することができる。 20

【0003】

本実施形態の一変更例では、分割することには、辺に関連する終点に基づいてグラフの辺を分割することが含まれる。

【0004】

本実施形態の一変更例では、分割することには、辺に関連する源点に基づいてグラフの辺を分割することが含まれる。

【0005】

本実施形態の一変更例では、分割することには、源点に基づいてグラフの辺を分割して、前方分割グラフを生成することが含まれる。このシステムはまた、終点に基づいて、グラフの辺を分割して後方分割グラフも生成する。次いで、このシステムは、前方分割グラフと後方分割グラフを組み合わせて、推薦する製品を決定する際の両方向のグラフ探査を容易にする。 30

【0006】

本実施形態の一変更例では、並列計算には、顧客の購買行動を分析する際にグラフの各区画を探索して辺の源点および/または終点を決定することが含まれる。

【0007】

本実施形態の一変更例では、並列計算とは  $n$  ビットの並列の幅優先探索を実行することであり、この  $n$  ビットの並列の幅優先探索では、隔たりの次数および/または頂点が  $n$  ビットで表され、 $n < 8$  である。

【図面の簡単な説明】 40

【0008】

【図 1】図 1 は、実施形態による、グラフ分割技術を用いたグラフ計算システムの例示的なアーキテクチャを示すブロック図である。

【図 2】図 2 は、実施形態による、グラフ分割技術を示すブロック図である。

【図 3】図 3 は、実施形態による、異なる分岐因子の値に関してグラフの区画の数が増えてしまった、最悪の場合の分割グラフの相対サイズを示すグラフである。

【図 4】図 4 は、追加の空間オーバーヘッドがないグラフの区画の最良の例を示すブロック図である。

【図 5】図 5 は、実施形態による、グラフを分割して並列計算を容易にするための例示的な処理を示すフローチャートである。 50

【図6】図6は、実施形態による、並列計算のためにグラフの分割を容易にする例示的な装置を示すブロック図である。

【図7】図7は、本発明の一実施形態による、並列計算のためにグラフの分割を容易にする例示的なコンピュータシステムを示すブロック図である。 図中では、同様の参照符号は同じ番号の要素を示す。

【発明を実施するための形態】

【0009】

本発明の実施形態により、各辺の終点または源点に基づいて、グラフの辺を分割しグラフの辺の重ならない集合を示す分散データを格納することにより、サイズが大きく明示的なグラフに関する並列計算を容易にするという課題を解決する。このようなグラフはサイズが大きいため、1台のマシンのメモリまたは記憶装置では、そのグラフを示すデータが収まらない可能性があり、グラフ計算システムでは、このようなグラフを示すデータを分割し、多くの記憶装置および/またはマシンに分散させることができる。そして、1つのサーバ（または記憶装置）だけが、各辺およびそれに関連する終点を書き込むことができるため、グラフ探査には同期するためのオーバーヘッドがなく効率が向上する。

【0010】

明示的なグラフでは、源点から終点までの各辺の点と頂点は、ある辺に関する源点になると同時に、別の辺に関する終点にもなり得ることに留意されたい。明示的なグラフの辺または頂点を推測する必要はない。さらに、方向を持たない辺は、一对の頂点の間の両方向を示す2つの方向を持つ辺として表すことが可能である。

【0011】

終点（例えば、後方頂点）に基づいて分割されたグラフは、後方分割グラフと呼ぶことができ、源点に基づいて分割されたグラフは、前方分割グラフと呼ぶことができる。システムの中には、後方分割グラフおよび前方分割グラフの両方を用いるために、グラフを分割することができるものもある。

【0012】

このシステムは、開示されたグラフ分割技術を用いて、各区画の終点の集合が他の区画の終点の集合と重ならないように辺を分割する。すなわち、2つの区画が同じ終点を共有することはない。これにより、2つのスレッド（または、処理/コア/コンピュータ）が同じ終点で動作していることで互いに通信・同期して時間を無駄してしまう状況を防ぐことができる。さらに、この技術では潜在的な辺が全てグラフに現れてしまう最悪の場合でも、辺で分割することにより分割性が保証される。

【0013】

方向を有する明示的なグラフで計算を行うために、グラフ計算システムは、最初にグラフが記述されたデータを受信する。ある実施形態では、このシステムは、辺の終点に基づいてグラフの辺を分割して、グラフの辺を別々に格納できるようにする（例えば、別々の記憶装置またはマシンに）。グラフの中の方向を有する辺が、2つの異なる区画に同時に存在することはない。但し、2つの頂点をつなぐ両方向の方向を持った2つの辺が存在する可能性はある。次いで、このシステムはグラフ上で並列計算を行うことができる。例えば、このシステムは、協調フィルタリングの並列計算を行って、ユーザに製品に薦めることができる。協調フィルタリングは、多くのユーザから嗜好情報を集めることにより、ユーザの興味を予測する技術である。この協調フィルタリングの計算では、システムは別々の記憶装置に格納されたグラフデータに対して並列で処理を実行する。分割されたグラフデータ上で同時に動作する、プロセッサどうし、および/または処理どうし、および/またはスレッドどうしが、グラフデータを探査するとき、互いに通信する必要はない。これにより、時間および記憶空間に関する計算コストを著しく削減し、並列グラフ処理に関する、広く応用可能で非常に効率の良い技術の分類を容易にする。

【0014】

本明細書で開示された新規なグラフ分割技術では、グラフ固有の構造の有無にかかわらずそのグラフの分割を保証する。グラフが完全につながっている最悪の場合でも分割性が

10

20

30

40

50

保証される。さらに、区間固有の情報を格納するために必要な空間オーバヘッドを、最小限にすることもでき、それでも並列処理に基づいた処理速度を著しく向上させることが可能である。このことは実験結果から確認できる。

【 0 0 1 5 】

なお、終点から源点までグラフを探索する、この分割技術の双対形式により、終点の代わりに源点に基づいて辺を分割することもできる。前述の通り、源点（すなわち、前方頂点）に基づいて分割されたグラフを前方分割グラフと呼ぶことができる。辺の源点と終点の両方を検査しなければならない従来の分割技術とは異なり、本明細書に開示された技術では、源点または終点のどちらか一方に基づいて分割を行うことができる。

【 0 0 1 6 】

図 1 は、実施形態によるグラフ分割技術を用いたグラフ計算システム 1 0 0 の例示的なアーキテクチャを示すブロック図である。グラフ計算システム 1 0 0 は、グラフを分割し分割されたグラフを別々の記憶装置など複数のロケーションに格納することができる。これらの記憶装置を 1 つ以上のマシンの中に設けることができる。記憶装置を複数のマシンの中に設ける場合、グラフの分割部分の書き込み、および読み出しを管理するために、ローカルのグラフ管理モジュールを個々のマシンに取り付けることができる。グラフを分割した後、グラフ計算システム 1 0 0 は、分割されたグラフを探索して協調フィルタリングなど種々のアプリケーションの実行を容易にすることができる。なお、図 1 には、分散メモリでの並列グラフ探索を用いる実施形態が示されているが、共有メモリでの並列グラフ探索など用いるその他の実施形態も実施可能である。

【 0 0 1 7 】

グラフ計算システム 1 0 0 は、グラフ管理モジュール 1 0 2 A を含むことができ、このグラフ管理モジュール 1 0 2 A は、サーバ 1 0 6 A に接続する記憶装置 1 0 4 A 内にインストールされている。サーバ 1 0 6 A は、サーバ 1 0 6 B、1 0 6 C、1 0 6 D と通信可能である。なお、本発明の種々の実装形態では、あらゆる数のサーバおよび記憶装置を含むことができる。グラフの書き込み、および探索に関連するグラフの動作、およびその他のグラフの動作を行うために、システム 1 0 0 は、種々のグラフ管理モジュール 1 0 2 A、1 0 2 B、1 0 2 C、1 0 2 D を含むことができる。システム 1 0 0 は、グラフを複数の区画に分割し、これらのグラフの区画を、区画 1 0 8 A、1 0 8 B、1 0 8 C、1 0 8 D として、それぞれ記憶装置 1 0 4 A、1 0 4 B、1 0 4 C、1 0 4 D に格納することができる。ある実施形態では、区画の数がネットワーククラスタ内のノードの数と等しくなるように、システム 1 0 0 がグラフを分割することができる。システムは、ユーザにより規定された区画の数に応じて、グラフを分割することができる。

【 0 0 1 8 】

いくつかの実施形態では、効率的なグラフ探索を容易にするために、システムが効率的な索引を付けて辺を格納することができる。このシステムは、辺に索引を付けるための索引の表を生成し用いることができ、このような索引の表を記憶装置内に格納することができる。

【 0 0 1 9 】

図 2 は、実施形態に従った、グラフ分割技術を示すブロック図である。図 2 には、システム 1 0 0 がグラフ 2 0 2 の辺の区画を、終点に基づいて、2 つのグラフの区画 2 0 4、2 0 6 に分割する様子が示されている。

【 0 0 2 0 】

グラフ 2 0 2 には、源点 2 0 8、2 1 0、2 1 2 と、終点 2 1 4、2 1 6 と、および辺 2 1 8、2 2 0、2 2 2、2 2 4、2 2 6、2 2 8 とが含まれる。辺 2 1 8、2 2 0、2 2 2 は、源点 2 0 8、2 1 0、2 1 2 と終点 2 1 4 を接続する。辺 2 2 4、2 2 6、2 2 8 は、源点 2 0 8、2 1 0、2 1 2 と終点 2 1 6 を接続する。

【 0 0 2 1 】

例示的な実装形態では、源点は人を表すことができ、終点は製品を表すことができる。例えば、源点 2 0 8 で表した人が、終点 2 1 4 で表された製品を購入する。また、源点 2

10

20

30

40

50

10で表した人は、終点214で表された製品、および終点216で表された製品を購入する。グラフを探索することにより、特定の製品（例えば、終点214で表された製品）を購入した人が同時に購入した別の製品を、システム100は、協調フィルタリングを通して、特定することができる。したがって、このシステムは、頂点210で表した人が頂点214で表された製品、および頂点216で表された製品を購入したことを、グラフから、特定することができる。頂点208および210で表した人達は二人とも頂点214で示された製品を購入する。

#### 【0022】

このシステムは、終点214、216に基づいて、グラフ202の辺を2つのグラフの区画204、206に分割してグラフの並列探索を容易にすることができる。グラフの区画204には、頂点214への方向を有する、3つの辺218、220、222を含む全ての辺が含まれる。辺218、220、222は、源点208、210、212から終点214への方向付を有する。グラフの区画206には、頂点216への方向を有する、3つの辺224、226、228を含む全ての辺が含まれる。

#### 【0023】

このシステムは、別々の記憶装置および/または別々のマシンにグラフの区画204および206を格納し、そこで処理を実行することができる。例えば、サーバ106Bはグラフの区画204を探索することができ、サーバ106Cはグラフの区画206を探索することができる。なお、サーバ106Bおよびサーバ106Cは、別々の頂点に向かって探索するため、同期する必要はない。あるアプリケーションの例では、このシステムは、プロセッサ間および/またはコンピュータ間で干渉することなく、異なる区画の隔たりの次数の情報を、それぞれ別々に並列で伝えることができる。この隔たりの次数とは、他の誰かと出会うために必要となる、連鎖的な紹介の数を示す数字でよい。例えば、このシステムが頂点の間の隔たりの次数をカウントするならば、グラフの区画204内の頂点208、210、212に関する隔たりの次数は1となり、したがってサーバ106Bは、頂点214に関する隔たりの次数を2と書き込むことができる。同時に、グラフの区画206内の頂点208、210、212に関する隔たりの次数が2ならば、サーバ106Cは、頂点216に関する隔たりの次数を3と書き込むことができる。同じ頂点に書き込むことによるサーバ106Bとサーバ106Cの間に干渉は発生しない。なお、いくつかの例では、終点216が終点214への方向を有する源点となることもあり得る。

#### 【0024】

図3は、実施形態に従った、異なる分岐因子の値に関してグラフの区画の数が増えてしまった、最悪な場合の後方分割グラフの相対的なサイズを示すグラフである。図3に示す通り、分岐因子「b」が増加すると、同じ数の区画に対する分割グラフの相対サイズは小さくなってしまふ。例えば、 $b = 10$ の場合、 $P = 8$ のとき分割グラフのサイズは元のグラフのサイズより64%大きくなるが、 $b = 40$ の場合は、たった17%しか大きくなりません。ソーシャルネットワークなどの大規模なグラフの場合、頂点（例えば、人）は約40のリンク（例えば、友達）を有し得る。したがって、ソーシャルネットワークに関する後方分割グラフの相対的な空間オーバーヘッドは、かなり小さくなる。さらに、区画の数を少なくなると、追加の空間オーバーヘッドも小さくなる。例えば、 $P = 4$ のとき、 $b = 40$ に関して分割グラフは分割されていないものよりもたった7%しか大きくなりません。なお、最良の場合の空間の複雑度は、分割グラフと非分割グラフの両方に関して同じであり、したがって分割グラフの実際の空間オーバーヘッド図3に示される最悪の場合の結果より小さい。

#### 【0025】

図4は、付加的な空間オーバーヘッドのないグラフの区画の最良の例を示すブロック図である。図4には、追加の空間オーバーヘッド、およびその非分割の同等物を必要としない2つの区画402、404を有するグラフの例が示されている。実在のグラフは、図4に示したもの全く同じとは言えないが、地理的な近さや社会地域などの制約により、これらの実在のグラフが、図4の2つに分割された（またはn個分割された）グラフと非常に似て

10

20

30

40

50

いるネットワーク構造に近づく可能性がある。したがって、分割グラフの空間オーバーヘッドは、非分割グラフの空間オーバーヘッドに近づく可能性がある。

【0026】

図5は、実施形態に従った、グラフを分割して並列計算を容易にする例示的な処理のフローチャートである。動作中、このシステムはまず、グラフの辺と頂点を規定する入力を受信する(動作502)。このシステムは、ユーザの入力を通して、または前もって格納されているグラフデータであるグラフデータを取得することができる。次に、このシステムは区画の数を決定する(動作504)。例えば、このシステムは50個の区画に対して $P = 50$ と決定することができる。このシステムはまた、区画の数を規定するユーザの入力を受信することもできる。

10

【0027】

次いで、このシステムは、グラフの辺を異なる区画に分割する(動作506)。このシステムは、最初にどの頂点がグラフの辺の源点および終点になるかを決定する。次いで、このシステムは、終点ノードの範囲を決定し(グラフの頂点を1から $n$ の範囲で表示することができる)、 $P$ 個の区画で終点の範囲を分割することができる。例えば、第1の区画には範囲 $[1, (n/P)]$ の頂点が含まれ得、第2の区画には範囲 $[(n/P) + 1, 2n/P]$ の頂点が含まれ得る。なお、区画の平均サイズは「 $n/P$ 」であり、最悪の場合の終点の範囲は1から $n$ である。

【0028】

このシステムは、各区画に関連する辺を格納することができる。例えば、このシステムは、種々の記憶装置、種々のファイル、または種々のディレクトリ、あるいは別々のマシンに辺を格納することができる(動作508)。グラフの分割後、このシステムはグラフの区画を用いて並列計算を行うことができる。例えば、グラフを探索することができる(動作510)。

20

【0029】

図6は、実施形態に従った、並列計算のためにグラフの分割を容易にする例示的な装置600を示すブロック図である。装置600は複数のモジュールを含むことができ、これらのモジュールは有線または無線の通信チャネルを介して互いに通信可能である。装置600は、1つ以上の集積回路用いて実現可能であり、図6で示されているものよりも多いモジュール、または少ないモジュールを含むことができる。さらに、装置600をコンピュータシステムの中に組み込むことも、あるいは他のコンピュータシステムおよび/または装置と通信可能な独立の装置として実現することも可能である。具体的には、装置600は、グラフデータ受信モジュール602、グラフ分割モジュール604、区画記憶・読出しモジュール606、グラフ探索モジュール608を含むことができる。

30

【0030】

いくつかの実施形態では、グラフデータ受信モジュール602は、グラフの頂点と辺を記述したデータを受信することができる。グラフ分割モジュール604は、本明細書に記載される技術を用いて、グラフを分割することができる。区画記憶・読出しモジュール606は、記憶装置への書き込みおよび読み出し、グラフの分割および探索に関連する全ての機能、およびグラフ検索に関連するその他の全てのグラフデータの操作機能を行う。グラフ探索モジュール608は、メモリまたは記憶装置内のグラフの探索を容易にする。なお、図1に示すグラフ管理モジュール102A~102Dは、図6で説明した種々のモジュールのありとあらゆる機能を実行可能である。

40

【0031】

このセクションでは、本明細書に記載された技術の複雑度について分析する。まず、いくつかの正式な定義を行い、記法の紹介から始め、分析、追加の例および説明と続く。

【0032】

グラフ $G$ が、頂点の集合(すなわち頂点 $v \in V$ )、および辺の集合(すなわち $e \in E$ )から成り、頂点「 $u$ 」から頂点「 $v$ 」まで方向を有する辺がグラフ $G$ に存在する場合、およびその場合に限って辺「 $e$ 」は $(u, v)$ の形態をとる。この場合、頂点「 $u$ 」は頂点

50



「 $v$ 」の前方頂点となり、頂点「 $v$ 」は頂点「 $u$ 」の後方頂点となる。グラフ $G$ が方向を持たない場合は、「 $(u, v) \in E$ 」となる。

【0033】

ある実装形態では、頂点は、その頂点に割り当てられた $[0, |V|]$ の範囲の一意の整数識別子を有し得る。符号「 $u$ 」および「 $v$ 」は、それぞれ辺に関連する源点および終点の整数識別子を示す。「 $v_{\min - dest}$ 」および「 $v_{\max - dest}$ 」を、それぞれ、終点（すなわち、少なくとも1つの（1）前方頂点を有する頂点）の最小整数識別子および最大整数識別子とする。システムがグラフ $G$ から作成するグラフの区画の数を「 $P$ 」とし、「 $v_{\min - dest}$ 」を $(P + 1)$ 個の要素の整数のアレイとし、下記の関係が成り立つようにする。

10

【数1】

$$\begin{aligned} v_{\min - dest}[0] &= v_{\min - dest} \\ v_{\min - dest}[p] &< v_{\min - dest}[p + 1] \quad (\text{但し } p = 0, 1, \dots, P - 1) \\ v_{\min - dest}[P] &= v_{\max - dest} + 1 \end{aligned}$$

【0034】

システム100は、辺の集合「 $E$ 」を $P$ 個の区画に分割して下記の関係が成り立つようにすることができる。

【数2】

$$E_p = \{ (u, v) \mid (u, v) \in E \wedge v_{\min - dest}[p] \leq v < v_{\min - dest}[p + 1] \} \quad (\text{但し、 } p = 0, 1, \dots, P - 1)$$

20

【0035】

これらの辺の区画「 $E_p$ 」は、以下の特性を有することが分かる。

【数3】

$$\begin{aligned} \bigcup_{p \in [0, P]} E_p &= E \\ \forall i \neq j \rightarrow E_i \cap E_j &= \emptyset, \\ \forall (u_i, v_i) \in E_i \wedge (u_j, v_j) \in E_j &\rightarrow v_i \neq v_j \end{aligned}$$

【0036】

これに応じて、システム100はまた、頂点の集合「 $V$ 」も $P$ 個の区画に分割して下記の関係が成り立つようにすることもできる。

【数4】

$$V_p = \{ v \mid \exists (u, v) \in E_p \vee (v, u) \in E_p \} \quad (\text{但し、 } p = 0, 1, \dots, P - 1)$$

【0037】

グラフに独立した頂点が無い場合、 $\bigcup_{p \in [0, P]} V_p = V$ となることが分かる。 $DV_p = \{ v \mid (u, v) \in E_p \}$ を区画「 $p$ 」内の終点の集合（但し、 $p = 0, 1, \dots, P - 1$ ）とする。

【0038】

これらの終点の集合「 $DV_p$ 」は下記の特性を有することが分かる。

40

【数5】

$$\forall i \neq j \rightarrow DV_i \cap DV_j = \emptyset$$

【0039】

すなわち、区画内の終点の集合が、他の区画内の終点の集合と干渉しないことが保証される。別々の区画で動作する別々のスレッド（またはコンピュータ）が、同じ終点には到達しないことが保証されるため、このことが、これらの区画を作成する主な理由である。これにより、異なる区画からの並列の書き込みが同じ頂点で起こらないことが保証される。単一の区画が、単一のスレッドだけにより処理されている場合、並列の書き込みが同じ

50

頂点に起こらないことをさらに保証することができる。並列の書き込みがされない場合、グラフ探索のステージでデータ破壊を防ぐための異なるスレッドどうし同期は必要ない。

【 0 0 4 0 】

探索ステージは、最も時間がかかるステージであり得るが、普通は、これがグラフ検索処理の唯一のステージではない。グラフ検索処理には、検索状態を精査し余分なものを切り詰め、その先を拡張するための頂点を先端に配置するなどのステージも含まれ得る。各探索ステージの終了時（すなわち、次の探索ステージの開始時）に、更新された検索状態が、影響を受ける得るスレッドに伝わるならば、本発明の実施形態では、グラフ検索処理で行われる探索以外の動作のタイプについて限定はしない。良好に構成されたグラフでは、スレッドの一部分だけしか、この更新情報を受信する必要がある可能性がある。一方、ありとあらゆるスレッド（またはコンピュータ）が、その更新された検索状態を、残りのスレッド（またはコンピュータ）伝えなければならない可能性もある。更新があった場合に、次の動作に進むことができるように、全ての更新が行われ、その処理が通知されれば、本発明の実施形態では、検索状態の更新を伝えるために用いる、通信プロトコル（複数可）またはメモリのアーキテクチャ（例えば、共有メモリ対分散メモリ）のタイプを限定はしない。共有メモリを有するコンピュータの場合、情報伝達ステージがない可能性がある、というのも、全ての局所的な更新はすぐに全体から見られ得るからである。このような場合、処理が延滞することなくすぐに次の動作に移る。

10

【 0 0 4 1 】

元のグラフ  $G$  が複数の区画に分割される場合、もとのグラフ  $G$  の符号化された情報には含まれていない、その区画固有の情報を格納する追加の空間が必要となる。1つの区画にだけ辺がマッピングするため、元のグラフの全ての辺は分割されたグラフ  $G$  の符号化された情報内に正確にたった1度だけ示される。

20

【数 6】

$$\sum p \in [0, P) \quad |E_p| = |E|$$

【 0 0 4 2 】

したがって、分割グラフは、元のグラフの辺の集合を示すための付加的な空間を必要としないが、システム 100 は、付加的な空間を用いて、これらの区画の辺のために効率的な検索用索引を作成することができる。

30

【 0 0 4 3 】

辺を符号化するための空間と同じサイズの空間を索引項目として使用すると仮定すると、 $|V|$  個の索引項目、つまりグラフの頂点ごとに1つの索引項目が存在し得る。元のグラフに関して必要とされる全空間は、「 $O(|E| + |V|)$ 」となり、これは辺のサイズと検索索引のサイズの合計となる。分割グラフに関して、次の不等式の関係が保たれている。

【数 7】

$$|V| \leq \sum_{p \in [0, P)} |V_p| \leq P |V|$$

【 0 0 4 4 】

40

【数 8】

$$\forall i \neq j \rightarrow V_i \cap V_j = \emptyset$$

の場合、およびその場合に限って、 $|V| = \sum_{p \in [0, P)} |V_p|$  ということが分かる。その一方で、 $V_p = V$  の場合、およびその場合に限って、 $\sum_{p \in [0, P)} |V_p| = P |V|$  である。両方とも起こり得るが、その可能性は高くない。明確な構造を持たないグラフでは、 $\sum_{p \in [0, P)} |V_p|$  は  $|V|$  よりも  $P |V|$  に近い。いずれの場合でも、 $\sum_{p \in [0, P)} |V_p|$  は、 $O(P |V|)$  よりも小さい。

【 0 0 4 5 】

$$\sum_{p \in [0, P)} |E_p| = |E| \text{ および } \sum_{p \in [0, P)} |V_p| \leq P |V|$$

50

に基づいて、 $P$  個の区画を有する後方分割グラフに関する空間の複雑度を次の通り表すことができる。

【数 9】

$$\begin{aligned} \sum_{p \in [0, P)} O(|E_p| + |V_p|) &= O\left(\sum_{p \in [0, P)} |E_p| + \sum_{p \in [0, P)} |V_p|\right) \\ &= O\left(\sum_{p \in [0, P)} |E_p|\right) + O\left(\sum_{p \in [0, P)} |V_p|\right) \\ &= O(|E|) + O(P|V|) \\ &= O(|E| + P|V|) \end{aligned}$$

【0046】

したがって、 $O(|E| + P|V|)$  は、 $P$  個の区画を有する後方分割グラフの最悪の場合の空間複雑度である。前の分析  $\sum_{p \in [0, P)} |V_p| = |V|$  から、最良の空間の複雑度は  $O(|E| + |V|)$  であり、これは元の非分割グラフの空間複雑度と同じである。

10

【0047】

後方分割グラフに関する追加の空間オーバーヘッドが重要かどうかは、 $|V|$  に対する  $|E|$  の割合に依存する。 $b = |E| / |V|$  (平均の分岐因子) とすると、非分割のグラフと比較した、最悪の場合の後方分割グラフの相対サイズは下記のように表すことができる。

【数 10】

$$\frac{O(|E| + P|V|)}{O(|E| + |V|)} = \frac{O(b|V| + P|V|)}{O(b|V| + |V|)} = O\left(\frac{b+P}{b+1}\right)$$

20

【0048】

後方分割グラフのコンセプトは、頂点からその後方頂点までの並列グラフ探索に対して有効であるということである。しかし、システム 100 では、所与の処理またはアプリケーションに関して両方の前方および逆方向に探索する必要がある。頂点から、その後方頂点までの非同期の並列グラフ探索を可能にするために、システム 100 は、前方分割グラフを生成することができ、この前方分割グラフは、以下に規定する通り、後方分割グラフと双対である。

【0049】

30

「 $v_{min-src}$ 」および「 $v_{max-src}$ 」を、それぞれ源点(すなわち、少なくとも 1 つの (1) 後方頂点を有する頂点)の最小整数識別子および最大整数識別子とする。下記の関係が成り立つように、「 $v_{min-src}$ 」を  $(P+1)$  個の要素の整数の配列とする(「 $P$ 」はグラフの区画の数)。

【数 11】

$$\begin{aligned} v_{min-src}[0] &= v_{min-src} \\ v_{min-src}[P] &< v_{min-src}[p+1], \quad (\text{但し、} p=0, 1, \dots, P-1) \\ v_{min-src}[P] &= v_{max-src} + 1 \end{aligned}$$

【0050】

40

下記の関係が成り立つように、辺の集合  $E$  を  $P$  個の区画に分割することができる。

【数 12】

$$E'_p = \{(u, v) \mid (u, v) \in E \wedge v_{min-src}[p] \leq u < v_{min-src}[p+1]\}, \quad (\text{但し、} p=0, 1, \dots, P-1)$$

【0051】

これらの辺の区画「 $E'_p$ 」が以下の特性を有することが分かる。

## 【数 1 3】

$$\begin{aligned} U_{p \in [0, P]} E'_p &= E \\ \forall i \neq j \rightarrow E'_i \cap E'_j &= \emptyset, \\ \forall (u_i, v_i) \in E'_i \wedge (u_j, v_j) \in E'_j &\rightarrow v_i \neq u_j \end{aligned}$$

## 【0 0 5 2】

これに応じて、システム 1 0 0 はまた、下記の式のように、頂点の集合「V」も P 個の区画に分割することができる。

## 【数 1 4】

$$V'_p = \{u \mid \exists (u, v) \in E'_p \vee (v, u) \in E'_p\} \quad (\text{但し、} p = 0, 1, \dots, P-1)$$

10

## 【0 0 5 3】

グラフに独立した頂点が無い場合、 $p \in [0, P]$   $V'_p = V$  となることが分かる。「 $SV_p = \{v \mid (u, v) \in E'_p\}$ 」を区画「p」内の源点の集合とする（但し、 $p = 0, 1, \dots, P-1$ ）。これらの源点の集合「 $SV_p$ 」は下記の特性を有することが分かる。

## 【数 1 5】

$$\forall i \neq j \rightarrow SV_i \cap SV_j = \emptyset$$

20

## 【0 0 5 4】

すなわち、ある区画内の源点の集合が別の区画内の源点の集合と重ならないことが保証される。ある頂点からその後方頂点までの並列グラフ探索を行う際、別々の前方グラフの区画で動作する別々のスレッド（またはコンピュータ）が、同じ源点に書き込むことができないことが保証される。単一の前方グラフの区画が、単一のスレッドだけで処理される場合、同じ源点への並列書き込みが無いことをさらに保証することができる。並列書き込みがされない場合は、データ破壊を防ぐために異なるスレッド（またはコンピュータ）どうしを、前方分割グラフの探索中に、同期（または通信）させる必要はない。終点は、このような探索では、読み出されるだけなので、この終点も影響を受けることはない。

## 【0 0 5 5】

30

グラフ計算システムは、前方分割グラフと後方分割グラフと組み合わせることにより、両方向（すなわち、ある頂点からその前方頂点および後方頂点まで）のグラフ探索を行うことが可能である。さらに、これらのどちらの方向の探索を行っている間にも同期する必要がないことが保証される。

## 【0 0 5 6】

（ $P+1$ ）個の要素の整数アレイである  $V_{min}$  を必要とする分割グラフの両方のタイプを呼び出すと、下記の関係が成り立つ。

## 【数 1 6】

$$\begin{aligned} V_{min}[0] &= v_{min} \\ V_{min}[p] &< V_{min}[p+1], \quad (\text{但し、} p = 0, 1, \dots, P-1) \\ V_{min}[P] &= v_{max} + 1 \end{aligned}$$

40

「 $v_{min}$ 」および「 $v_{max}$ 」は、それぞれ前方分割グラフおよび後方分割グラフに関する、源点および終点の最小整数識別子および最大整数識別子である。「 $v_{min}$ 」と「 $v_{max}$ 」の間の範囲を包括的に、P 個の部分範囲に分割するための最も簡単な方法は、P 個の（ほぼ）等しいサイズの区画に分割することである。最初の（ $P-1$ ）個の区画は、全く同じサイズ（ $(v_{max} - v_{min} + 1) / P$ ）を有し、最後の区画が残った頂点を含む。つまり下記ようになる。

## 【数 17】

$$\begin{aligned}
 V_{min}[0] &= v_{min} \\
 V_{min}[p] &= v_{min} + p^L (v_{max} - v_{min} + 1) / P^L \quad (\text{但し、} p = 0, 1, \dots, P-1) \\
 V_{min}[P] &= v_{max} + 1
 \end{aligned}$$

## 【0057】

なお、 $(v_{max} - v_{min} + 1) / P$  は、これらの各区画の頂点の数だけを合計しているため、最初の  $(P - 1)$  個の区画の頂点のサイズだけを有する。そして、区画内の辺の数を合計する辺のサイズは、その頂点のサイズとはかなり異なる。したがって、全く同じ数の頂点を有する区画どうしであっても、お互いの区画のサイズはかなり異なる。というのも、区画のサイズは頂点のサイズと辺のサイズの合計であるからである。

10

## 【0058】

非均一な頂点サイズの区画を含む「 $v_{min}$ 」と「 $v_{max}$ 」の間の範囲を分割するやり方は他にもある。例えば、その一つとして、区画サイズを全ての区画で（大まかに）同じにするために、非均一な頂点のサイズでも可能にすることが挙げられる。辺から区画へのマッピングが一意的（すなわち、必ず単一の辺がきっちりと1つの区画にマッピングされる）であるならば、その他の変形例も可能である。

## 【0059】

前述した通り、本開示の一樣態では、全てのグラフの探索ステージ中、前方頂点ベースまたは後方頂点ベースであれば、そのスレッド（またはコンピュータ）が別のスレッド（またはコンピュータ）と同期する（または通信する）必要がないことを保証している。本開示では、グラフ検索の最もコストのかかる部分、すなわち探索ステージで、同期または通信を必要と減らす。さらに、スレッド（またはコンピュータ）が、探索ステージ後、検索情報を他のスレッド（またはコンピュータ）に送信する際、ストリーミング方式で送信している、というのも複数の頂点に関する更新された検索情報を一緒にまとめることで通信オーバーヘッドを減らすことができるからである。

20

## 【0060】

共有メモリの実施形態、および分散メモリの実施形態に関して、

- ・共有メモリの並列グラフ探索の場合、下記に記載する分散メモリとは異なり、同期または通信する必要なしに、更新された全ての検索状態を全てのスレッドですぐに見ることができるので、同期の数をグラフ探索のステージごとに  $O(|E|)$  から  $O(1)$  に削減することができる。

30

- ・分散メモリの並列グラフ探索の場合、本開示により、辺がカウントされている全ての探索ステージで通信する必要がないため、本発明の実施形態では、通信の数を  $O(|E|)$  から  $O(|V|)$  に減らすことができる。辺の数  $|E|$  は通常、頂点の数  $|V|$  より（非常に）多いため、削減される数は多くなり得る。

## 【0061】

但し、より正確に分析するためには、分散メモリ環境での辺の格納のやり方、および検索のやり方、ならびにコンピュータ間で通信するときの辺のグループ化のやり方などの要因が必要となるため、上記の通信の複雑度の分析は概算の値となることに留意されたい。同様に、頂点の検索状態のグループ化のやり方も、本発明の種々の実施形態の通信の複雑度に影響する。より具体的には、 $O(|E| / b_e)$  が、辺に関する通信の複雑度であり、 $O(|V| / b_v)$  が、頂点に関する通信の複雑度である（「 $b_e$ 」および「 $b_v$ 」は、それぞれグラフ探索のステージで用いられる辺および頂点の通信のブロックサイズ）。一般には、「 $b_e \ll b_v$ 」であり、したがって、本開示での相対的な通信の複雑度の利点としては、「 $b_e$ 」と「 $b_v$ 」の間の差を考慮した場合、「 $b_e$ 」が「 $O(|E|) / O(|V|) = O(|E| / |V|) = O(b)$ 」よりも著しく大きくなり得ることである。すなわち、「 $O(b \cdot b_v / b_e)$ 」を用いて、開示されている実施形態の複雑度の利点をより良く推定することが可能である。

40

50

## 【0062】

実証的な調査を行わないと、「 $b_v$ 」と「 $b_e$ 」の間の正確な割合を算出するのは困難であるが、実際には「 $b_v$ 」>「 $b_e$ 」と仮定しておけば、通常は間違いない。というのも、「 $b_v$ 」はその通信プロトコルで可能な最大値となり、一方「 $b_e$ 」は平均の分岐因子「 $b$ 」およびグラフの局所性の影響を受け得るためである。例えば、「 $b$ 」が小さい場合、またはある頂点の後方頂点（または前方頂点）を生成するコンピュータがローカルメモリ内の辺を全て含まなければ、「 $b_e$ 」の値は小さくなる傾向がある。その一方で、本開示の技術を適用するシステムは、線形走査を用いて単一のメッセージ内のできるだけ多くの頂点をグループ化することができるため、「 $b$ 」またはグラフの局所性の影響を受けにくい。

10

## 【0063】

上記の複雑度の分析は単一のグラフ探索ステージに対するものであり、複数の（探索または非探索）ステージが存在する場合、その結果をその都度調整しなければならない。例えば、共有メモリでは、このシステムが「 $d$ 」回の探索ステップを行わなければならない場合、同期の複雑度は、本開示の技術を用いた「 $O(1)$ 」ではなく「 $O(d)$ 」となり得る。

## 【0064】

本明細書で開示される実施形態のその他の利点として、グラフの区画の数に制限されることなく、特別な事例のアプローチ（カテゴリ # 2 で）を行うことができる。このことは重要である。というのも、これによりグラフの区画より多い数のスレッド（またはコンピュータ）を用いて、特別な事例のアプローチを実行できるためである。P 個のグラフの区画の事例を考察するが、幅優先探索を行うために、「 $m \times P$ 」個のコアが利用可能である。各区画でも  $m$  個のコアの同時に使用することができ、したがって、これらの P 個の区画内で同期することなく「 $m \times P$ 」個のコアを使用可能である。この考えを少し変えてみると、不統一の数のコアを用いて、各区画を処理することも可能である。例えば、単一の区画を処理するために割り当てられるコアの数は、区画内の辺の数に比例し得る。単一の区画を処理するために用いるコアの数を制限することなしに、これらの変形例がカテゴリ # 2 内の特別な事例のアプローチと完全に用いることができることは大きな魅力である。

20

## 【0065】

さらに別の利点は、 $n$  ビットの幅優先探索（ $n < 8$ ）などの特定の特別な事例のアプローチの空間効率および時間効率を向上させることができるということである。ソーシャルネットワークで周知の六次の隔たりの規則を考慮すると、スモールワールドグラフに関する隔たりの次数を符号化するために数ビットの容量しか必要としないことが分かる。例えば、7 次の隔たりまでを符号化するためには 3 ビットで十分であり得る。しかし、並列の幅優先探索のアプローチでは、最大の隔たりの次数がどの程度になるかにかかわらず、この隔たりの次数を符号化するために、少なくとも 1 バイト（8 ビット）を用いなければならない。その理由は、複数のスレッドが同じ値をメモリセルに書き込む際、1 バイトが一貫性を保証できる最小のメモリ単位であるためである。本開示の実施形態では、1 つのスレッドで各区画が検索されている限り、このような最小 1 バイトの符号化の制限を取り除くことができる。

30

40

## 【0066】

これにより、並列の 2 ビットの幅優先探索などの幅優先探索の高度な変形例をその固有の形態で適用して、メモリおよび実行時間を著しく節約することが可能となる。なお、4 次より小さい隔たりの次数しか計算できないものとは対照的に、高度な 2 ビットの幅優先探索では、任意のグラフに関する隔たりの次数を計算することができる。2 ビットの幅優先探索の詳細な説明は文献で見ることができ、最良の並列の 2 ビットの幅優先探索は、2 ビットよりむしろ 8 ビットを要して頂点を格納する。本開示では、最初に、隔たりの次数および / または頂点を符号化するために 2 ビットしか用いない、本当の 2 ビットの並列幅優先探索を実行可能にする。さらに広く言えば、本開示では、隔たりの次数および / または頂点を符号化するために  $n$  ビット（ $n < 8$ ）用いる、 $n$  ビットの並列幅優先探索を実行

50

可能にする。

【実施例】

【0067】

開示された技術の効果を調査するために実験が行われた。使用されたグラフは以下の形の顧客の購買行動データに基づく。〈顧客\_\_id〉、〈製品\_\_id〉、〈顧客\_\_id〉を有する顧客が、〈製品\_\_id〉を有する製品購入する。この顧客 - 製品グラフは、2部グラフであり、辺は顧客の頂点と製品の頂点の間にだけ存在し得る。この辺により、顧客が製品を購入したことが示される。ベンチマークのクエリーは以下のステージを有する。

1. 顧客の頂点からスタートし、顧客により購入された製品の集合を検索する。
2. これらの製品のうちの少なくとも1つを購入した別の顧客を検索する。
3. これらの顧客に購入された製品の中で最も人気のある製品を検索する。

10

【0068】

上記のクエリーは、ステップ1で特定された顧客に製品を推薦することができる協調フィルタである。以下の理由から、本開示の実施形態の並列効率に対してベンチマークを行うことを選択した。

【0069】

第1に、後方分割グラフおよび前方分割グラフの両方が必要である。ステップ1では、探査方向が顧客から製品になるため（すなわち、ある頂点からその後方頂点）後方分割グラフが必要となる。ステップ2では、その反対の探査方向（すなわち、ある頂点から前方頂点）のため、前方分割グラフが必要となる。第2に、このクエリーには、非グラフベースのステップ、すなわち、仕分けを必要とするステップ3、が含まれる。多くの実在のアプリケーションには、グラフベースの計算、および非グラフベースの計算の両方が含まれるため、ベンチマークのクエリーで仕分けステップを有することにより、この実験におけるアプリケーションの現実感を増加させることができる。

20

【0070】

この実験で使用された顧客 - 製品グラフでは、1探査方向に約24,400,000個の辺を有する。したがって、格納される辺の総数は、両方向の探査に対して約24,400,000 × 2 = 48,800,000個の辺となる。約3,000,000人の一意の顧客および400,000個を超える一意の製品が存在する。100人の顧客の集合をステップ1で使用するクエリーの異なる「種」としてランダムに選び、壁時計の平均秒数を記録する。基準値として、キャッシュに優しいグラフ探査の実装形態を用い、この実装形態により頂点の集合を、数多くの重ならない範囲に分割し、これらの頂点の値を後方の頂点から並列にて戻す。基準値の実装形態では、各スレッドをその他のスレッドと重ならない「書き込み」領域に割り当てることによりスレッド間の干渉が避けられる。基準値の実装形態内の全てのスレッドは、源点へ戻す必要のある値を求めて終点の全範囲を走査しなければならない。使用されたテストマシンは、インテルのXeon E3-1225、および4つコア有する3.1GHzのプロセッサと、8GBのRAMとを備える。本開示で使用可能な新しい並列技術、および基準値に対して、最良の構成が使用された。

30

【0071】

壁時計の平均時間は、基準値に関しては172,000,000秒であり、新しい技術に関しては24,000,000秒であり、こちらの方が基準値よりも7.2倍速い。なお、ベンチマークのクエリーの、メモリの初期化および製品のフィルタリングを含む複数のステップは、基準値と新しい技術の両方で共通であるが、これらはグラフ検索には関係ない。したがって、グラフ探査に関する新しい技術の相対速度の利点は、ここで見られたものよりも大きい。

40

【0072】

さらに、基準値の実装形態では、最も多く購入された製品を検索するだけでよい場合、ステップ3での仕分けを行わないが、新しい技術では、普通のシーケンスソートのアルゴリズム（qsort）を用いて、ステップ2で特定された顧客の中の少なくとも1人により購入された全ての製品を順位付けする。新しい技術の24,000,000秒には、（

50

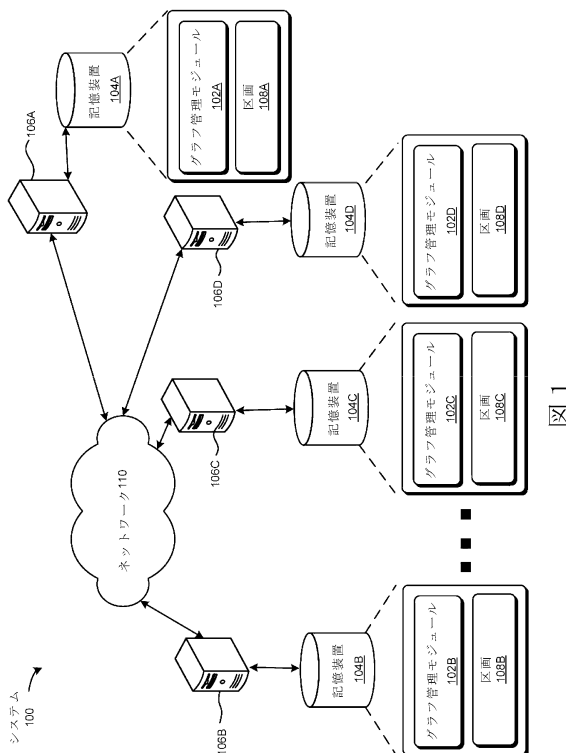
上位から  $k$  番目までの、最も人気のある製品を検索するために必要な) 仕分けステップを含む全ての計算結果が含まれる。この 24,000,000 秒に含まれないものは、前方分割グラフおよび後方分割グラフを作成するための 1 度きりの計算結果の投資であり、これには 3,000,000 件の全ての顧客に関して 430,000,000 秒かかる。したがって、分割グラフの償却原価は非常に小さい。

### 【0073】

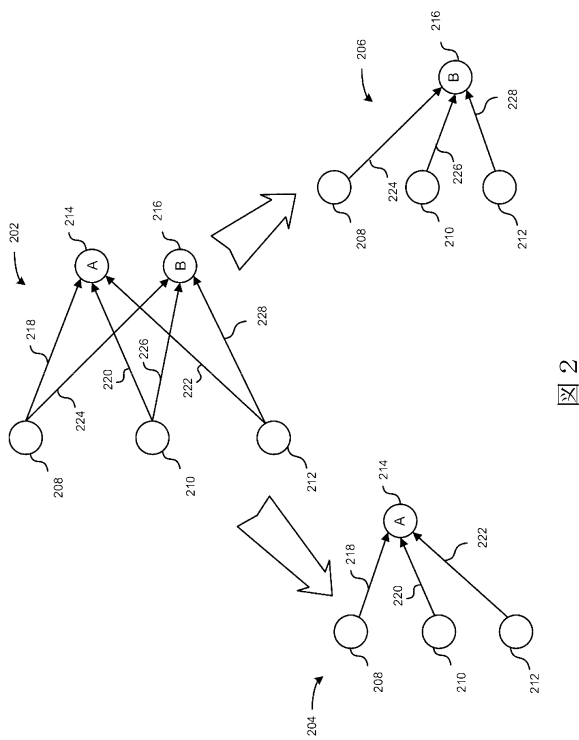
(例示的なシステム)

図 7 には、本発明の一実施形態に従った、並列計算を行うための分割グラフを容易にする例示的なコンピュータシステム 700 が示されている。ある実施形態では、コンピュータシステム 700 は、プロセッサ 702、メモリ 704、記憶装置 706 を含む。記憶装置 706 は、アプリケーション 710 および 712、ならびにオペレーティングシステム 716 などの複数のアプリケーションを格納する。記憶装置 706 はまた、グラフ計算システム 100 を格納し、このグラフ計算システム 100 には、グラフデータ受信モジュール 602、グラフ分割モジュール 604、区画記憶・読出しモジュール 606、グラフ探索モジュール 608 が含まれ得る。動作中、グラフ計算システム 100 など、1 つ以上のアプリケーションが、記憶装置 706 からメモリ 704 に読み出され、プロセッサ 702 が、読み出されたアプリケーションを実行する。プロセッサ 702 は、プログラムを実行すると同時に、上述した機能も実行する。コンピュータおよび通信システム 700 は、随意的に、ディスプレイ 717、キーボード 718、ポインティングデバイス 720 に接続可能である。

【図 1】



【図 2】





【図 3】

元のグラフに  
対する大きさ

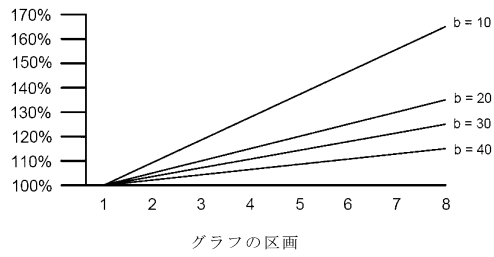


図 3

【図 4】

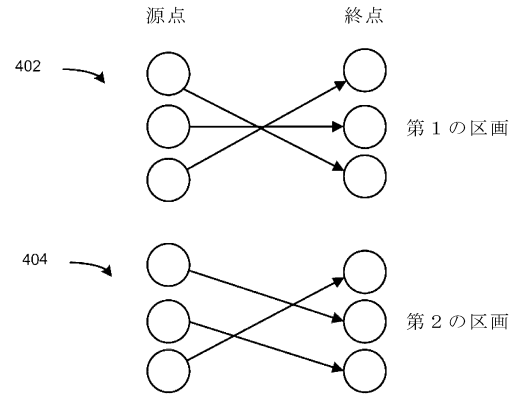


図 4

【図 5】

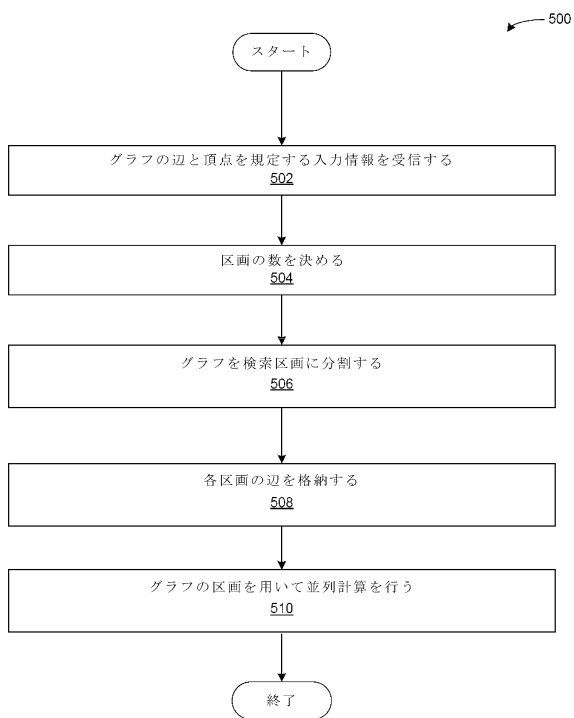


図 5

【図 6】



図 6

【図 7】

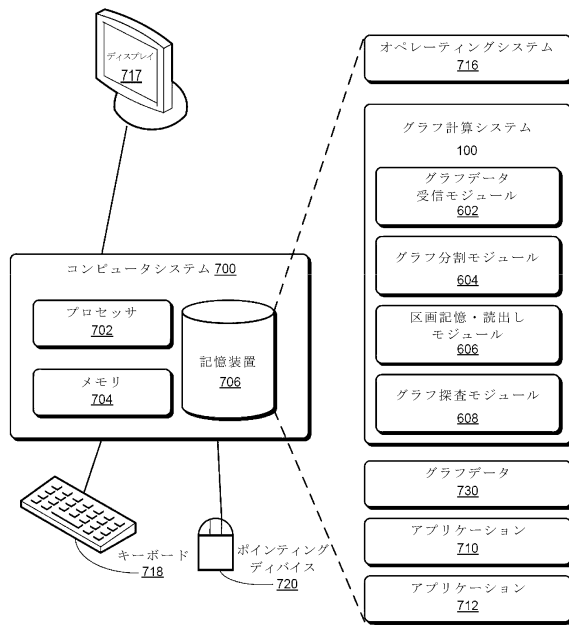


図 7

---

フロントページの続き

(72)発明者 ロン・チョウ

アメリカ合衆国 カリフォルニア州 95129 サンノゼ ホワイトブリック・ドライブ 10  
34

審査官 石田 信行

(56)参考文献 特開2013-89236(JP,A)

特開2012-198873(JP,A)

特開2012-98845(JP,A)

国際公開第2012/067949(WO,A1)

(58)調査した分野(Int.Cl., DB名)

G06F 17/30

G06F 9/50

G06Q 10/00 - 99/00