

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7629464号
(P7629464)

(45)発行日 令和7年2月13日(2025.2.13)

(24)登録日 令和7年2月4日(2025.2.4)

(51)国際特許分類

G 0 6 F 8/34 (2018.01)

F I G 0 6 F 8/34

請求項の数 40 (全37頁)

(21)出願番号	特願2022-545888(P2022-545888)	(73)特許権者	509123208
(86)(22)出願日	令和2年4月30日(2020.4.30)		アビニシオ テクノロジー エルエルシー
(65)公表番号	特表2023-511631(P2023-511631 A)		アメリカ合衆国 0 2 4 2 1 マサチュー
			セッツ州 レキシントン スプリング ス
(43)公表日	令和5年3月20日(2023.3.20)		トリート 2 0 1
(86)国際出願番号	PCT/US2020/030612	(74)代理人	100108453
(87)国際公開番号	WO2021/154319		弁理士 村山 靖彦
(87)国際公開日	令和3年8月5日(2021.8.5)	(74)代理人	100110364
審査請求日	令和5年4月27日(2023.4.27)		弁理士 実広 信哉
(31)優先権主張番号	62/966,768	(74)代理人	100133400
(32)優先日	令和2年1月28日(2020.1.28)		弁理士 阿部 達彦
(33)優先権主張国・地域又は機関	米国(US)	(72)発明者	イアン・シェクター
			アメリカ合衆国・マサチューセッツ・0
			2 0 6 7・シャロン・ブルック・ロード
			・ 9 4

最終頁に続く

(54)【発明の名称】 計算グラフを生成するためのエディタ

(57)【特許請求の範囲】

【請求項1】

第1のデータフローグラフを第2のデータフローグラフに変換するためのコンピュータ実施方法であって、前記第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、前記第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、前記方法は、

データ処理における前記複数の第1のコンピュータ動作を表す前記複数の第1のノードを含む前記第1のデータフローグラフを生成することであって、前記複数の第1のコンピュータ動作のうちの少なくとも1つは、データ処理の1つ以上の結果の1つ以上の特性を指定する宣言動作である、ことと、

前記第1のデータフローグラフにおいて表される少なくともいくつかの宣言動作の中からパターンを識別することと、

前記少なくともいくつかの宣言動作の中から識別された前記パターンに基づいて、前記複数の第1のコンピュータ動作に従ってデータを処理するために前記第1のデータフローグラフを前記第2のデータフローグラフに変換することであって、前記第2のデータフローグラフは、前記複数の第2のコンピュータ動作を表す前記複数の第2のノードを含み、前記第2のノードのうちの少なくとも1つは、前記宣言動作によって指定されるロジックを実施する1つ以上の命令動作を表す、ことと、

前記第2のデータフローグラフをデータストアに記憶することと、を含む、コンピュータ実施方法。

【請求項 2】

前記第 1 のデータフローグラフを前記第 2 のデータフローグラフに変換することは、
前記命令動作を作成することと、
前記命令動作を表す所与の第 2 のノードを作成することであって、所与の前記第 2 のノードは、前記第 1 のデータフローグラフに表されない、ことと、を含む、請求項 1 に記載のコンピュータ実施方法。

【請求項 3】

前記第 2 のデータフローグラフに表され、前記第 1 のデータフローグラフに表されない前記複数の第 2 のコンピュータ動作のうちの 1 つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択される、請求項 1 に記載のコンピュータ実施方法。

10

【請求項 4】

前記複数の第 2 のコンピュータ動作のうちの 1 つ以上は、少なくとも、

(i) 前記第 1 のデータフローグラフで指定された前記複数の第 1 のコンピュータ動作のうちの 1 つ以上に従ってデータを処理するために必要であるか、又は

(i i) 前記 1 つ以上の追加動作を伴わないデータ処理と比較して、前記第 1 のデータフローグラフで指定された前記複数の第 1 のコンピュータ動作のうちの 1 つ以上に従ったデータ処理を改善する、請求項 1 に記載のコンピュータ実施方法。

【請求項 5】

1 つ以上のデータフローグラフ最適化ルールを前記第 2 のデータフローグラフに適用して、前記適用前の前記第 2 のデータフローグラフの計算効率と比較して、前記第 2 のデータフローグラフの計算効率を改善することによって、前記第 2 のデータフローグラフを最適化されたデータフローグラフに変換することを更に含む、請求項 1 に記載のコンピュータ実施方法。

20

【請求項 6】

前記 1 つ以上のデータフローグラフ最適化ルールは、前記第 2 のデータフローグラフから冗長ノードを削除すること、前記第 2 のデータフローグラフからデッドノードを削除すること、前記第 2 のデータフローグラフ内でノードの順序を変更すること、前記第 2 のデータフローグラフ内でノードの強度を低減すること、前記第 2 のデータフローグラフ内で 2 つ以上のノードを統合すること、前記第 2 のデータフローグラフ内のノードを直列動作から並列動作に変換すること、又は前記第 2 のデータフローグラフ内で分割動作を挿入することのうちの少なくとも 1 つを含む、請求項 5 に記載のコンピュータ実施方法。

30

【請求項 7】

前記複数の第 2 のコンピュータ動作のうちの少なくとも 1 つは、自動並列動作又は自動分割動作を含む、請求項 1 に記載のコンピュータ実施方法。

【請求項 8】

前記複数の第 2 のコンピュータ動作のうちの少なくとも 1 つは、並べ替え動作を含む、請求項 1 に記載のコンピュータ実施方法。

【請求項 9】

前記複数の第 2 のコンピュータ動作のうちの少なくとも 1 つは、前記第 2 のノードのうちの 1 つ以上の間でメタデータを指定する動作を含む、請求項 1 に記載のコンピュータ実施方法。

40

【請求項 10】

データを提供して、キャンバス部分と、カタログ部分と、を含むグラフィカルエディタインターフェースを生成することであって、前記カタログ部分は、前記キャンバス部分において、計算のロジックを視覚的に描写するための 1 つ以上の選択可能なアイコンを含む、ことと、

前記キャンバス部分において描写された計算のロジックを示す、アイコン選択データを受信することであって、前記アイコン選択データは、前記カタログ部分から選択され、前記キャンバス部分に含まれる、前記 1 つ以上の選択可能なアイコンのうちの少なくとも 1

50

つを指定する、ことと、

受信した前記アイコン選択データから、前記キャンバス部分で指定された前記ロジックを表す前記複数の第1のノードを含む前記第1のデータフローグラフを生成することであって、前記第1のノードのうちの少なくとも1つは、前記カタログ部分から選択された前記1つ以上の選択可能なアイコンのうちの前記少なくとも1つを表す、ことと、を更に含む、請求項1に記載のコンピュータ実施方法。

【請求項11】

選択された各アイコンは、データをプリフォーマットするデータカタログからデータにアクセスする命令を表すか、又は前記データカタログを介してアクセスされるデータの形式を指定する、請求項10に記載のコンピュータ実施方法。

【請求項12】

前記第1のデータフローグラフは、ユーザ定義のデータフローグラフである、請求項1に記載のコンピュータ実施方法。

【請求項13】

データを提供して、キャンバス部分と、カタログ部分と、を含むグラフィカルエディタインターフェースを生成することであって、前記カタログ部分は、複数のデータセット選択アイコンと、複数の変換選択アイコンと、を含む、ことと、

選択されたデータセット選択アイコン及び選択された変換選択アイコンによって表される記憶ユニットに記憶された要素に従って、前記第1のデータフローグラフ内に初期ノードを生成することと、

前記初期ノードにラベルを付けて、ラベル付きノードを提供することと、

前記キャンバス部分で、前記ラベル付きノードの視覚的表現をレンダリングすることと、を更に含む、請求項1に記載のコンピュータ実施方法。

【請求項14】

前記初期ノードは、動作を保持するための動作プレースホルダフィールドと、データのソース又はシンクを保持するためのデータプレースホルダフィールドと、を有する、請求項13に記載のコンピュータ実施方法。

【請求項15】

修正することは、

記憶システムから前記動作プレースホルダフィールドに保持された前記動作の要素を取得することと、

前記記憶システムから前記データプレースホルダフィールドに保持されたデータソース又はデータシンクの要素を取得して、前記データのの前記ソース又は前記シンクを指すリンクを前記データプレースホルダフィールドに追加することと、を更に含む、請求項14に記載のコンピュータ実施方法。

【請求項16】

データを提供して、前記グラフィカルエディタインターフェースの前記キャンバス部分に前記第1のデータフローグラフをレンダリングすることを更に含む、請求項13に記載のコンピュータ実施方法。

【請求項17】

生成した前記初期ノードの全てにラベルを付けると、前記方法は、

前記第1のデータフローグラフの全てのラベル付きノードを、計算データフローグラフである前記第2のデータフローグラフにコンパイルすることを更に含む、請求項13に記載のコンピュータ実施方法。

【請求項18】

修正した前記初期ノードの全てのラベルを付けると、前記方法は、

前記第1のデータフローグラフの全てのラベル付きノードを最適化することを更に含み、前記第1のデータフローグラフの前記ラベル付きノードを最適化することは、前記ラベル付きノードのうちの少なくとも1つに記憶された前記要素を最適化することを更に含む、請求項13に記載のコンピュータ実施方法。

10

20

30

40

50

【請求項 19】

プロトタイプノードにアクセスすることと、

アクセスした前記プロトタイプノードからパラメータをコピーして、前記初期ノードのうちの少なくとも1つを修正するアルゴリズムを適用することと、を更に含む、請求項13に記載のコンピュータ実施方法。

【請求項 20】

前記初期ノードの少なくとも1つのパラメータは、前記プロトタイプノードによって上書きされない設定パラメータである、請求項19に記載のコンピュータ実施方法。

【請求項 21】

前記プロトタイプノードは、前記初期ノード、前記初期ノード上のポート、又は前記グラフィカルエディタインターフェースの前記キャンバス部分に提示されたコンポーネントのパラメータのうちの少なくとも1つを宣言する、請求項19に記載のコンピュータ実施方法。

10

【請求項 22】

プロトタイプを適用することは、前記プロトタイプからの記述子で既存のパラメータの記述子を置き換えるが、パラメータの既存の値は置き換えない、請求項19に記載のコンピュータ実施方法。

【請求項 23】

メタデータ及び前記第1のデータフローグラフに記述された値を計算する変換を適用することを更に含む、請求項1に記載のコンピュータ実施方法。

20

【請求項 24】

前記ラベルは、キー、値、名前、及びソースのうちの1つ以上を参照する、請求項13に記載のコンピュータ実施方法。

【請求項 25】

選択されたデータセット選択アイコンによって表される記憶ユニット及び選択された変換選択アイコンによって表される記憶ユニットに記憶された1つ以上の要素を記憶する前記複数の初期ノードのうちの少なくともいくつかは、少なくとも部分的に、前記複数の初期ノードのうちの前記少なくともいくつかに対して対応する記憶ユニット機能を指定する、請求項13に記載のコンピュータ実施方法。

【請求項 26】

30

第1のデータフローグラフを第2のデータフローグラフに変換するためのシステムであって、前記第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、前記第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、前記システムは、

1つ以上のプロセッサと、命令を記憶する1つ以上の記憶デバイスであって、前記命令は、前記1つ以上のプロセッサによって実行されると、

データ処理における前記複数の第1のコンピュータ動作を表す前記複数の第1のノードを含む前記第1のデータフローグラフを生成することであって、前記複数の第1のコンピュータ動作のうちの少なくとも1つは、データ処理の1つ以上の結果の1つ以上の特性を指定する宣言動作である、ことと、

40

前記第1のデータフローグラフにおいて表される少なくともいくつかの宣言動作の中からパターンを識別することと、

前記少なくともいくつかの宣言動作の中から識別された前記パターンに基づいて、前記複数の第1のコンピュータ動作に従ってデータを処理するために前記第1のデータフローグラフを前記第2のデータフローグラフに変換することであって、前記第2のデータフローグラフは、前記複数の第2のコンピュータ動作を表す前記複数の第2のノードを含み、前記第2のノードのうちの少なくとも1つは、前記宣言動作によって指定されるロジックを実施する1つ以上の命令動作を表す、ことと、

前記第2のデータフローグラフをデータストアに記憶することと、を含む動作を前記1つ以上のプロセッサに実行させる、1つ以上の記憶デバイスと、を含む、システム。

50

【請求項 27】

命令を記憶する非一時的コンピュータ可読媒体であって、前記命令は、

データ処理における第1のコンピュータ動作を表す複数の第1のノードを含む第1のデータフローグラフを生成することであって、前記第1のコンピュータ動作のうちの少なくとも1つは、データ処理の1つ以上の結果の1つ以上の特性を指定する宣言動作である、ことと、

前記第1のデータフローグラフにおいて表される少なくともいくつかの宣言動作の中からパターンを識別することと、

前記少なくともいくつかの宣言動作の中から識別された前記パターンに基づいて、前記第1のコンピュータ動作に従ってデータを処理するために前記第1のデータフローグラフを第2のデータフローグラフに変換することであって、前記第2のデータフローグラフは、第2のコンピュータ動作を表す複数の第2のノードを含み、前記第2のノードのうちの少なくとも1つは、前記宣言動作によって指定されるロジックを実施する1つ以上の命令動作を表す、ことと、

前記第2のデータフローグラフをデータストアに記憶することと、を計算システムに実行させる、非一時的コンピュータ可読媒体。

【請求項 28】

第1のデータフローグラフを第2のデータフローグラフに変換するためのコンピュータ実施方法であって、前記第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、前記第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、前記方法は、

データ処理における前記複数の第1のコンピュータ動作を表す前記複数の第1のノードを含む前記第1のデータフローグラフを生成することと、

前記第1のデータフローグラフにおいて表される少なくともいくつかの宣言動作の中からパターンを識別することと、

前記少なくともいくつかの宣言動作の中から識別された前記パターンに基づいて、前記複数の第1のコンピュータ動作に従ってデータを処理するために前記第1のデータフローグラフを前記第2のデータフローグラフに変換することであって、前記第2のデータフローグラフは、前記複数の第2のコンピュータ動作を表す前記複数の第2のノードを含み、前記複数の第2のコンピュータ動作のうちの少なくとも所与の1つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択される、ことと、

前記第2のデータフローグラフをデータストアに記憶することと、を含む、コンピュータ実施方法。

【請求項 29】

第1のデータフローグラフを第2のデータフローグラフに変換するためのシステムであって、前記第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、前記第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、前記システムは、

1つ以上のプロセッサと、命令を記憶する1つ以上の記憶デバイスであって、前記命令は、前記1つ以上のプロセッサによって実行されると、

データ処理における前記複数の第1のコンピュータ動作を表す前記複数の第1のノードを含む前記第1のデータフローグラフを生成することと、

前記第1のデータフローグラフにおいて表される少なくともいくつかの宣言動作の中からパターンを識別することと、

前記少なくともいくつかの宣言動作の中から識別された前記パターンに基づいて、前記複数の第1のコンピュータ動作に従ってデータを処理するために前記第1のデータフローグラフを前記第2のデータフローグラフに変換することであって、前記第2のデータフローグラフは、前記複数の第2のコンピュータ動作を表す前記複数の第2のノードを含み、前記複数の第2のコンピュータ動作のうちの少なくとも所与の1つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択される、ことと、

10

20

30

40

50

前記第 2 のデータフローグラフをデータストアに記憶することと、を含む動作を前記 1 つ以上のプロセッサに実行させる、1 つ以上の記憶デバイスと、を含む、システム。

【請求項 30】

命令を記憶する非一時的コンピュータ可読媒体であって、前記命令は、

データ処理における第 1 のコンピュータ動作を表す複数の第 1 のノードを含む第 1 のデータフローグラフを生成することと、

前記第 1 のデータフローグラフにおいて表される少なくともいくつかの宣言動作の中からパターンを識別することと、

前記少なくともいくつかの宣言動作の中から識別された前記パターンに基づいて、前記第 1 のコンピュータ動作に従ってデータを処理するために前記第 1 のデータフローグラフを第 2 のデータフローグラフに変換することであって、前記第 2 のデータフローグラフは、第 2 のコンピュータ動作を表す複数の第 2 のノードを含み、前記第 2 のコンピュータ動作のうちの少なくとも所与の 1 つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択される、ことと、

前記第 2 のデータフローグラフをデータストアに記憶することと、を計算システムに実行させる、非一時的コンピュータ可読媒体。

【請求項 31】

前記複数の第 2 のコンピュータ動作のうちの少なくともいくつかは、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 1 に記載のコンピュータ実施方法。

【請求項 32】

前記 1 つ以上の命令動作は、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 1 に記載のコンピュータ実施方法。

【請求項 33】

前記複数の第 2 のコンピュータ動作のうちの少なくともいくつかは、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 26 に記載のシステム。

【請求項 34】

前記 1 つ以上の命令動作は、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 26 に記載のシステム。

【請求項 35】

前記 1 つ以上の命令動作は、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 27 に記載の非一時的コンピュータ可読媒体。

【請求項 36】

前記複数の第 2 のコンピュータ動作のうちの少なくともいくつかは、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 28 に記載のコンピュータ実施方法。

【請求項 37】

前記複数の第 2 のコンピュータ動作のうちの前記少なくとも所与の 1 つは、前記第 1 のデータフローグラフ内の第 1 のノードによって表されない、請求項 28 に記載のコンピュータ実施方法。

【請求項 38】

前記複数の第 2 のコンピュータ動作のうちの少なくともいくつかは、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 29 に記載のシステム。

【請求項 39】

前記複数の第 2 のコンピュータ動作のうちの前記少なくとも所与の 1 つは、前記第 1 のデータフローグラフ内の第 1 のノードによって表されない、請求項 29 に記載のシステム。

【請求項 40】

前記第 2 のコンピュータ動作のうちの前記少なくとも所与の 1 つは、前記第 1 のデータフローグラフ内の前記第 1 のノードによって表されない、請求項 30 に記載の非一時的コンピュータ可読媒体。

10

20

30

40

50

【発明の詳細な説明】

【技術分野】

【0001】

(優先権の主張)

本出願は、2020年1月28日に出願された米国特許仮出願第62／966,768号の優先権を主張し、その全内容は、参照により本明細書に組み込まれる。

【背景技術】

【0002】

本開示は、計算グラフを生成することに関する。

【0003】

複雑な計算は、多くの場合、有向グラフを使用したデータフローとして表すことができ、計算のコンポーネントは、グラフの頂点及びグラフのリンク（弧、辺）に対応するコンポーネント間のデータフローと関連付けられている。かかるグラフベースの計算を実施するシステムは、米国特許第5,966,072号、名称「Executing Computations Expressed as Graphs」に記載されており、その全内容は、参照により本明細書に組み込まれる。場合によっては、頂点と関連付けられた計算は人間可読形式で記述され、「ビジネスルール」と呼ばれる。

【0004】

データフローグラフを生成するための1つの技術は、ビジネスルールエディタを使用する。ビジネスルールエディタの例は、米国特許第8,069,129号、名称「Editing and Compiling Business Rules」に開示されており、その全内容は、参照により本明細書に組み込まれる。

【発明の概要】

【課題を解決するための手段】

【0005】

一般的態様1において、第1のデータフローグラフを第2のデータフローグラフに変換するための方法であって、第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、第2のコンピュータ動作のうちの少なくともいくつかは、第1のデータフローグラフ内の第1のノードによって表されず、この方法は、データ処理における第1のコンピュータ動作を表す複数の第1のノードを含む第1のデータフローグラフを生成することであって、第1のコンピュータ動作のうちの少なくとも1つは、データ処理の1つ以上の結果の1つ以上の特性を指定する宣言動作である、ことと、第1のコンピュータ動作に従ってデータを処理するために第1のデータフローグラフを第2のデータフローグラフに変換することであって、第2のデータフローグラフは、第2のコンピュータ動作を表す複数の第2のノードを含み、第2のノードのうちの少なくとも1つは、宣言動作によって指定されるロジックを実施する1つ以上の命令動作を表し、1つ以上の命令動作は、第1のデータフローグラフ内の第1のノードによって表されない、ことと、第2のデータフローグラフをデータストアに記憶することと、を含む。

【0006】

態様1に記載の態様2において、第2のデータフローグラフへの第1のデータフローグラフは、命令動作を作成することと、命令動作を表す所与の第2のノードを作成することであって、所与の第2のノードは、第1のデータグラフに表されない、ことと、を含む。

【0007】

態様1又は2のいずれかに記載の態様3において、第2のデータフローグラフに表され、第1のデータフローグラフには表されない第2の動作のうちの1つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択される。

【0008】

態様1～3のいずれかに記載の態様4において、第2の動作のうちの1つ以上は少なくとも、(i)第1のデータフローグラフで指定された第1の動作のうちの1つ以上に従っ

10

20

30

40

50

てデータを処理するために必要であるか、又は (i i) 1 つ以上の追加動作を伴わないデータ処理と比較して、第 1 のデータフローグラフで指定された第 1 の動作のうちの 1 つ以上に従ったデータ処理を改善する。

【 0 0 0 9 】

態様 1 ～ 4 のいずれかに記載の態様 5 において、方法は、1 つ以上のデータフローグラフ最適化ルールを第 2 のデータフローグラフに適用することによって第 2 のデータフローグラフを最適化されたデータフローグラフに変換して、適用前の第 2 のデータフローグラフの計算効率と比較して、第 2 のデータフローグラフの計算効率を改善することを更に含む。

【 0 0 1 0 】

態様 1 ～ 5 のいずれかに記載の態様 6 において、1 つ以上のデータフローグラフ最適化ルールは、第 2 のデータフローグラフから冗長ノードを削除すること、第 2 のデータフローグラフからデッドノードを削除すること、第 2 のデータフローグラフでノードの順序を変更すること、第 2 のデータフローグラフでノードの強度を低減すること、第 2 のデータフローグラフで 2 つ以上のノードを統合すること、第 2 のデータフローグラフ内のノードを直列動作から並列動作に変換すること、又は第 2 のデータフローグラフで分割動作を挿入することのうちの少なくとも 1 つを含む。

【 0 0 1 1 】

態様 1 ～ 6 のいずれかに記載の態様 7 において、第 2 の動作のうちの少なくとも 1 つは、自動並列動作又は自動分割動作を含む。

【 0 0 1 2 】

態様 1 ～ 7 のいずれかに記載の態様 8 において、第 2 の動作のうちの少なくとも 1 つは、並べ替え動作を含む。

【 0 0 1 3 】

態様 1 ～ 8 のいずれかに記載の態様 9 において、第 2 の動作のうちの少なくとも 1 つは、第 2 のノードのうちの 1 つ以上の間でメタデータを指定する動作を含む。

【 0 0 1 4 】

態様 1 ～ 9 のいずれかに記載の態様 1 0 において、方法は、データを提供して、キャンバス部分及びカタログ部分を含むグラフィカルエディタインターフェースを生成することであって、カタログ部分は、キャンバス部分において、計算のロジックを視覚的に描写するために 1 つ以上の選択可能なアイコンを含む、ことと、キャンバス部分において描写された計算のロジックを表す、アイコン選択データを受信することであって、アイコン選択データは、カタログ部分から選択され、キャンバス部分に含まれる、1 つ以上の選択可能なアイコンのうちの少なくとも 1 つを指定する、ことと、受信したアイコン選択データから、キャンバス部分で指定されたロジックを表す複数の第 1 のノードを含む第 1 のデータフローグラフを生成することであって、第 1 のノードのうちの少なくとも 1 つは、カタログ部分から選択された 1 つ以上の選択可能なアイコンのうちの少なくとも 1 つを表す、ことと、を更に含む。

【 0 0 1 5 】

態様 1 ～ 1 0 のいずれかに記載の態様 1 1 において、選択された各アイコンは、データをプリフォーマットするデータカタログからデータにアクセスする命令を示すか、又はデータカタログを介してアクセスされるデータの形式を指定する。

【 0 0 1 6 】

態様 1 ～ 1 1 のいずれかに記載の態様 1 2 において、第 1 のデータフローグラフは、ユーザ定義のデータフローグラフである。

【 0 0 1 7 】

態様 1 ～ 1 2 のいずれかに記載の態様 1 3 において、方法は、データを提供して、キャンバス部分及びカタログ部分を含むグラフィカルエディタインターフェースを生成することであって、カタログ部分は、複数のデータセット選択アイコンと、複数の変換選択アイコンと、を含む、ことと、選択されたデータセット選択アイコン及び選択された変換選択

10

20

30

40

50

アイコンによって表される記憶ユニットに記憶された要素に従って、第1のデータフローグラフで初期ノードを生成することと、初期ノードにラベルを付けて、ラベル付きノードを提供することと、キャンバス部分にラベル付きノードの視覚的表現をレンダリングすることと、を更に含む。

【0018】

態様1～13のいずれかに記載の態様14において、初期ノードは、動作を保持する動作プレースホルダフィールドと、データのソース又はシンクを保持するデータプレースホルダフィールドと、を有する。

【0019】

態様1～14のいずれかに記載の態様15において、修正することは、動作プレースホルダフィールドに保持された動作の要素を記憶システムから取得することと、データプレースホルダフィールドに保持されたデータソース又はデータシンクの要素を記憶システムから取得して、データのソース又はシンクを指すリンクをデータプレースホルダフィールドに追加することと、を更に含む。

10

【0020】

態様1～15のいずれかに記載の態様16において、方法は、データを提供して、グラフィカルエディタインターフェースのキャンバス部分に第1のデータフローグラフをレンダリングすることを更に含む。

【0021】

態様1～16のいずれかに記載の態様17において、生成された初期ノードの全てにラベルを付けると、方法は、第1のデータフローグラフの全てのラベル付きノードを、計算データフローグラフである第2のデータフローグラフにコンパイルすることを更に含む。

20

【0022】

態様1～17のいずれかに記載の態様18において、修正した初期ノードの全てにラベルを付けると、方法は、第1のデータフローグラフの全てのラベル付きノードを最適化することを更に含み、第1のデータフローグラフのラベル付きノードを最適化することは、ラベル付きノードのうちの少なくとも1つに記憶された要素を最適化することを更に含む。

【0023】

態様1～18のいずれかに記載の態様19において、方法は、プロトタイプノードにアクセスすることと、アクセスされたプロトタイプノードからパラメータをコピーして初期ノードのうちの少なくとも1つを修正するアルゴリズムを適用することと、を更に含む。

30

【0024】

態様1～19のいずれかに記載の態様20において、初期ノードの少なくとも1つのパラメータは、プロトタイプノードによって上書きされない設定パラメータである。

【0025】

態様1～20のいずれかに記載の態様21において、プロトタイプノードは、初期ノード、初期ノード上のポート、又はエディタインターフェースのキャンバスに提示されたコンポーネントのパラメータのうちの少なくとも1つを宣言する。

【0026】

態様1～21のいずれかに記載の態様22において、プロトタイプを適用することは、プロトタイプからの記述子で既存のパラメータの記述子を置き換えるが、パラメータの既存の値は置き換えない。

40

【0027】

態様1～22のいずれかに記載の態様23において、方法は、メタデータ及び第1のデータフローグラフに記述された値を計算する変換を適用することを更に含む。

【0028】

態様1～23のいずれかに記載の態様24において、ラベルは、キー、値、名前、及びソースのうちの1つ以上を参照する。

【0029】

態様1～24のいずれかに記載の態様25において、選択されたデータセット選択アイ

50

コンによって表される記憶ユニット及び選択された変換選択アイコンによって表される記憶ユニットに記憶された1つ以上の要素を記憶する複数の初期ノードのうちの少なくともいくつかは、少なくとも部分的に、複数の初期ノードのうちの少なくともいくつかに対して対応する記憶ユニット機能を指定する。

【0030】

一般的態様26において、第1のデータフローグラフを第2のデータフローグラフに変換するためのシステムであって、第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、第2のコンピュータ動作のうちの少なくともいくつかは、第1のデータフローグラフ内の第1のノードによって表されず、このシステムは、1つ以上のプロセッサと、命令を記憶する1つ以上の記憶デバイスであって、命令は、1つ以上のプロセッサによって実行されると、データ処理における第1のコンピュータ動作を表す複数の第1のノードを含む第1のデータフローグラフを生成することであって、第1のコンピュータ動作のうちの少なくとも1つは、データ処理の1つ以上の結果の1つ以上の特性を指定する宣言動作である、ことと、第1のコンピュータ動作に従ってデータを処理するために第1のデータフローグラフを第2のデータフローグラフに変換することであって、第2のデータフローグラフは、第2のコンピュータ動作を表す複数の第2のノードを含み、第2のノードのうちの少なくとも1つは、宣言動作によって指定されるロジックを実施する1つ以上の命令動作を表し、1つ以上の命令動作は、第1のデータフローグラフ内の第1のノードによって表されない、ことと、第2のデータフローグラフをデータストアに記憶することと、を含む動作を1つ以上のプロセッサに実行させる、1つ以上の記憶デバイスと、を含む。

【0031】

一般的態様27において、非一時的コンピュータ可読媒体であって、この非一時的コンピュータ可読媒体は、データ処理における第1のコンピュータ動作を表す複数の第1のノードを含む第1のデータフローグラフを生成することであって、第1のコンピュータ動作のうちの少なくとも1つは、データ処理の1つ以上の結果の1つ以上の特性を指定する宣言動作である、ことと、第1のコンピュータ動作に従ってデータを処理するために第1のデータフローグラフを第2のデータフローグラフに変換することであって、第2のデータフローグラフは、第2のコンピュータ動作を表す複数の第2のノードを含み、第2のノードのうちの少なくとも1つは、宣言動作によって指定されるロジックを実施する1つ以上の命令動作を表し、1つ以上の命令動作は、第1のデータフローグラフ内の第1のノードによって表されない、ことと、第2のデータフローグラフをデータストアに記憶することと、をコンピューティングシステムに実行させるための命令を記憶する。

【0032】

一般的態様28において、第1のデータフローグラフを第2のデータフローグラフに変換するための方法であって、第1のデータフローグラフは、複数の第1のコンピュータ動作を表す複数の第1のノードを含み、第2のデータフローグラフは、複数の第2のコンピュータ動作を表す複数の第2のノードを含み、第2のコンピュータ動作のうちの少なくともいくつかは、第1のデータフローグラフ内の第1のノードによって表されず、この方法は、データ処理における第1のコンピュータ動作を表す複数の第1のノードを含む第1のデータフローグラフを生成することと、第1のコンピュータ動作に従ってデータを処理するために第1のデータフローグラフを第2のデータフローグラフに変換することであって、第2のデータフローグラフは、第2のコンピュータ動作を表す複数の第2のノードを含み、第2のコンピュータ動作のうちの少なくとも所与の1つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択され、第2のコンピュータ動作のうちの少なくとも所与の1つは、第1のデータフローグラフ内の第1のノードによって表されない、ことと、第2のデータフローグラフをデータストアに記憶することと、を含む。

【0033】

一般的態様 29 において、第 1 のデータフローグラフを第 2 のデータフローグラフに変換するためのシステムであって、第 1 のデータフローグラフは、複数の第 1 のコンピュータ動作を表す複数の第 1 のノードを含み、第 2 のデータフローグラフは、複数の第 2 のコンピュータ動作を表す複数の第 2 のノードを含み、第 2 のコンピュータ動作のうちの少なくともいくつかは、第 1 のデータフローグラフ内の第 1 のノードによって表されず、このシステムは、1 つ以上のプロセッサと、命令を記憶する 1 つ以上の記憶デバイスであって、1 つ以上のプロセッサによって実行されると、データ処理における第 1 のコンピュータ動作を表す複数の第 1 のノードを含む第 1 のデータフローグラフを生成することと、第 1 のコンピュータ動作に従ってデータを処理するために第 1 のデータフローグラフを第 2 のデータフローグラフに変換することであって、第 2 のデータフローグラフは、第 2 のコンピュータ動作を表す複数の第 2 のノードを含み、第 2 のコンピュータ動作のうちの少なくとも所与の 1 つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択され、第 2 のコンピュータ動作のうちの少なくとも所与の 1 つは、第 1 のデータフローグラフ内の第 1 のノードによって表されない、ことと、第 2 のデータフローグラフをデータストアに記憶することを含む動作を 1 つ以上のプロセッサに実行させる、1 つ以上の記憶デバイスと、を含む。

【0034】

一般的態様 30 において、非一時的コンピュータ可読媒体であって、この非一時的コンピュータ可読媒体は、データ処理における第 1 のコンピュータ動作を表す複数の第 1 のノードを含む第 1 のデータフローグラフを生成することと、第 1 のコンピュータ動作に従ってデータを処理するために第 1 のデータフローグラフを第 2 のデータフローグラフに変換することであって、第 2 のデータフローグラフは、第 2 のコンピュータ動作を表す複数の第 2 のノードを含み、第 2 のコンピュータ動作のうちの少なくとも所与の 1 つは、並べ替え動作、データ型動作、指定キーとの結合動作、及び分割動作からなる群から選択され、第 2 のコンピュータ動作のうちの少なくとも所与の 1 つは、第 1 のデータフローグラフ内の第 1 のノードによって表されない、ことと、第 2 のデータフローグラフをデータストアに記憶することと、をコンピューティングシステムに実行させるための命令を記憶する。

【0035】

前述の全て又は一部（態様 1 ～ 30 及びそれらの任意の組み合わせなど）は、1 つ以上の非一時的機械可読記憶媒体に記憶され、1 つ以上の処理デバイスで実行可能である命令を含むコンピュータプログラム製品として実施され得る。前述の全て又は一部は、実行可能命令を記憶する 1 つ以上の処理デバイス及びメモリを含み、記述された機能を実施し得る装置、方法、又は電子システムとし実施され得る。

【0036】

本明細書で使用される「表されない」という用語は、第 2 のコンピュータ動作のうちの少なくともいくつかは、第 1 のデータフローグラフにおいて直接的又は間接的に生じないか、又は複数の第 1 のノードのいずれも、第 2 のコンピュータ動作のうちの少なくともいくつかを表さないことを意味し得る。

【0037】

上記の実施例のうちの 1 つ以上は、以下の利点のうちの 1 つ以上を提供し得る。本明細書に記載の技術は、最低限の技術的背景を有するユーザが、使いやすいグラフィックユーザインターフェースを用いてデータ処理機能を指定することを可能にする。データフローグラフシステムは、キャンバスと、複数の変換アイコン及びデータセットアイコンを有するデータセットカタログと、を含むグラフィカルユーザインターフェースを提供する。所望のアイコンを選択し、それらをキャンバス上に配置することにより、ユーザは、データセットアクセスの詳細又は並べ替え及び分割動作など他の低レベルの実施詳細を指定する必要なく、データフローグラフ、したがってその関連する計算を容易に作成し、修正することができる。このようにして、データフローグラフシステムは、ユーザによって開発され、スキーマ駆動型開発を可能にするため、データフローグラフの視覚的表現を提供する。更に、システムは、ユーザによって配置されたアイコンを自動的に接続することによ

て開発を促進する自動レイアウト機能を含み得る。

【0038】

データフローグラフが完成すると、システムは、ユーザによって作成されたデータフローグラフを、コンパイルされ、実行され得る、変換されたデータフローグラフに変換する。このプロセスの一環として、システムは、データフローグラフで指定された宣言動作を実行するために必要な命令動作を追加することによって、例えば、並列動作及び分割動作を自動的に追加すること、結合及びロールアップのために並べ替え動作を追加すること、並びに中間メタデータを指定することによって、データフローグラフを自動的に最適化する。システムはまた、冗長コンポーネントの削除又はレコードの縮小などによって、データフローグラフ自体のデータ処理を改善する。

10

【0039】

1つ以上の実施例の詳細を、添付の図面及び以下の説明に記載する。本明細書に記載の技術の他の特徴、目的、及び利点は、説明及び図面、並びに特許請求の範囲から明らかとなるであろう。

【図面の簡単な説明】

【0040】

【図1】 計算グラフを指定するためのシステムのブロック図である。

【図2A】 データフローグラフのブロック図である。

【図2B】 データフローグラフのブロック図である。

【図3A】 異なる構築段階におけるデータフローグラフの視覚的表現の図である。

20

【図3B】 異なる構築段階におけるデータフローグラフの視覚的表現の図である。

【図3C】 異なる構築段階におけるデータフローグラフの視覚的表現の図である。

【図4A】 異なる動作段階における図1のシステムのブロック図である。

【図4B】 異なる動作段階における図1のシステムのブロック図である。

【図4C】 異なる動作段階における図1のシステムのブロック図である。

【図4D】 異なる動作段階における図1のシステムのブロック図である。

【図5】 計算データフローグラフを生成するための例示的なプロセスのフローチャートである。

【発明を実施するための形態】

【0041】

30

本明細書に記載の技術は、コンパイルのために最適化され、変換され得る、容易に修正可能な計算グラフを生成するための技術に関する。いくつかの実施例では、データフローグラフシステムは、キャンバスと、複数の変換選択アイコン及びデータセット選択アイコンを有するデータセットカタログと、を含むグラフィカルユーザインターフェースを提供する。所望のアイコンを選択し、それらをキャンバス上に配置することにより、ユーザは、データセットアクセスの詳細又は並べ替え及び分割動作など他の低レベルの実施詳細を指定（又は再指定）する必要なく、データフローグラフを容易に作成し、視覚化し、修正することができる。一般に、本明細書で使用される場合、動作は、コンピュータ動作、例えば、機械、コンピュータシステムなどによって実行又は実施される1つ以上の動作又は命令を指す。

40

【0042】

システムは、ユーザによって選択されたアイコンを示すデータを受信し、宣言動作及び選択されたアイコンによって参照されるデータセットを含むようにデータフローグラフのノードを修正することによって、第1のデータフローグラフ（本明細書では「中間データフローグラフ」と呼ばれることもある）を生成する。第1のデータフローグラフが完成すると、システムは、このデータフローグラフを、コンパイルされ、実行され得る、第2のデータフローグラフ（本明細書では「変換されたデータフローグラフ」と呼ばれることもある）に変換する。このプロセスの一環として、システムは、第1のデータフローグラフで指定された宣言動作を実行するために必要な命令動作を第2のデータフローグラフに追加することによって、例えば、並列動作及び分割動作を自動的に追加すること、結合及び

50

ロールアップのために並べ替え動作を追加すること、並びに中間メタデータを指定することによって、第1のデータフローグラフを自動的に最適化して第2のデータフローグラフを生成する。システムはまた、図4に関連して以下に記載するように、冗長コンポーネントの削除又はレコードの縮小などによって、第2のデータフローグラフ自体によるデータ処理を改善する。

【0043】

図1は、計算グラフを指定するためのシステム10の概略図を示す。システム10（本明細書では「実行環境」とも呼ばれる）は、エディタユーザインターフェースジェネレータ14と、データフローグラフ（dataflow graph、DFG）エンジン16と、変換エンジン18と、を有するデータフローグラフ生成システム12を含む。グラフ生成システム12はクライアントデバイス20に動作可能に連結されて、グラフ生成システム12へのアクセスを提供し、それによって、実行環境へのアクセスも提供し、また、コンパイラシステム22へのアクセスを提供して、グラフ生成システム12によって生成されたデータフローグラフをコンパイルする。コンパイルされたデータフローグラフは、実行及び記憶のためにデータ処理システム24及び記憶システム26に提供される。記憶システム26は、記憶デバイスなど1つ以上のデータソース又はオンラインデータソースへの接続を含み、これらのそれぞれは、様々な形式（例えば、とりわけデータベーステーブル、スプレッドシートファイル、フラットテキストファイル、又はメインフレームコンピュータによって使用されるネイティブ形式）のいずれかでデータを記憶し得るか、又は提供し得る。特に、記憶システム26は、データ処理システム24によって取得されて、コンパイルされたデータフローグラフ、並びにコンパイルされたデータフローグラフの実行を通じて生成された出力データを実行することができる、メタデータ及びレコード形式など入力データを記憶し得る。

【0044】

一般に、実行環境は、好適なシステム、例えば、とりわけUNIX（登録商標）ベースのオペレーティングシステム、Windowsベースのオペレーティングシステムなどの制御下にある1つ以上の汎用コンピュータ上でホストされ得る。例えば、実行環境は、ローカル（例えば、対称マルチ処理（symmetric multi-processing、SMP）コンピュータなどマルチプロセッサシステム）、又はローカル分散（例えば、クラスタ若しくは超並列処理（massively parallel processing、MPP）システムとして連結された複数のプロセッサ、又はリモート、又はリモート分散（例えば、ローカルエリアネットワーク（local area network、LAN）若しくはワイドエリアネットワーク（wide-area network、WAN）を介して連結された複数のプロセッサ）、あるいはそれらの任意の組み合わせのいずれかの、複数の処理ユニット（CPUなど）又はプロセッサコアを使用したコンピュータシステムの構成を含む、マルチノード並列コンピューティング環境を含むことができる。

【0045】

データフローグラフの生成された表現にアクセスし、グラフを定義するコンピュータ命令を生成する他のシステムとは異なり、システム10は中間グラフ（例えば、データフローグラフ17）にアクセスし、最適化及び他の動作によって中間グラフを変換して、コンパイルのために変換されたグラフ（例えば、変換されたデータフローグラフ19）を生成する。例えば、データフローグラフエンジン16は、所望の計算グラフのデータソース、データシンク、及びデータ処理機能を示すクライアントデバイス20から選択データを受信する。これらの詳細はデータフローグラフ生成システム12によって得られ得るため、クライアントデバイス20のユーザは、データアクセスの詳細又は他の低レベルの実施詳細を指定する必要はない。選択データに基づいて、データフローグラフエンジン16は、データフローグラフ17を生成するか、又は以前に作成されたデータフローグラフ17を修正する。変換エンジン18は、完成したデータフローグラフ17を受信し、例えば、最適化及び変換の中でもとりわけ、図4Dに関連して以下に記載するように、データフローグラフ17における冗長性を除去すること、データフローグラフ17に並べ替え又はパーティションを追加すること、及び中間メタデータ（例えば、データフローグラフ17を変

換されたデータフローグラフ19に変換する、あるいは別の方法で変換するためのメタデータ)を指定することによって、データフローグラフを変換されたデータフローグラフ19に変換する。その後、変換されたデータフローグラフ19はコンパイラシステム22に提供されて、変換されたデータフローグラフ19をコンパイルされた計算グラフ(例えば、実行可能プログラム23)にコンパイルする。

【0046】

一般に、データフローグラフ17(「修正されたデータフローグラフ」と呼ばれることもある)は、ノード(又はコンポーネント)を有する、変換されたデータフローグラフ19などコンパイルされたグラフのコア構造を表す。データフローグラフ17は、任意選択的に、パラメータ(例えば、名前、値、場所、解釈)を含む。いくつかの実施例では、データフローグラフ17は、サブグラフとして使用されることを意図したグラフのように、グラフ自体に入力ポート及び出力ポートを含む。

【0047】

いくつかの実施例では、ノード(又はコンポーネント)は、ノードの挙動又は機能を示すノードの「種類」を有するか、又はかかるノードの「種類」である。ノードの種類は、ノードのプロトタイプを選択するため、パターンマッチングを容易にするため(例えば、並べ替えノードに続く並べ替えノードを見出すため)、変換されたデータフローグラフ19内でインスタンス化されるコンポーネントを判定するために使用される。例えば、データフローグラフ17におけるゴミノードは、変換されたデータフローグラフ19においてゴミノードとしてインスタンス化され得る。ノード(又はコンポーネント)は、以下に記載するように、入力ポート、出力ポート、及びパラメータを含み得る。

【0048】

ノードは、任意選択的に、ノードを識別するラベルを有する。いくつかの実施例では、ノードがラベルを有しない場合、システムが、ノードにラベルを割り当てる。ノードラベルは、英数字、空白文字、及び句読点の任意の集合体を含み得、一意である必要はない(ただし、グラフへの変換中に一意になり得る)。システムは、ノードラベルを使用してノード(又はノードの入力ポート、出力ポート、又はパラメータ)を参照して、例えば、ノードの入出力又はノード間のデータフローを定義することができる。

【0049】

図2Aを参照すると、データフローグラフ17の一実施例が、ノード34a~34nを含むものとして示されている。ノード34のそれぞれは、少なくとも1つの動作プレースホルダフィールドと、少なくとも1つのデータプレースホルダフィールドと、を含む。例えば、「初期」ノード34aは、1つ以上の動作要素35a'を保持する動作プレースホルダフィールド35aと、1つ以上のデータソース又はデータシンク要素35b'を保持するデータプレースホルダフィールド35bと、を有する。動作要素35a'は、初期ノード34aに対するデータの入出力で機能を実行するコード又はコードの位置を指定し得る。データソース又はデータシンク要素35b'は、初期ノード34aの(初期ノード34aの機能の)データソース若しくはデータシンク、又はデータソース若しくはデータシンクの位置を指定し得る。いくつかの実施例では、要素35a'若しくは要素35b'、又はその両方は、データベースへのリンク又は記憶システム26に含まれるコードへのポインタなど記憶システム26へのリンク又はアドレスを含む。いくつかの実施例では、要素35a'若しくは要素35b'、又はその両方は、スクリプトを含む。

【0050】

データフローグラフ17の構築中、ノード34のそれぞれは、動作プレースホルダフィールドに配置される動作要素及びデータプレースホルダフィールドに配置されるデータソース又はデータシンク要素を取得してそれぞれのフィールドに追加することによって修正され得る。例えば、初期ノード34aは、動作要素35a'を(例えば、記憶システム26から)取得して、指定された機能又は機能を指すリンクを動作プレースホルダフィールド35aに追加することによって、またデータソース又はデータシンク要素35b'を取得して、データのソース又はシンクを指すリンクをデータプレースホルダフィールド35bに

10

20

30

40

50

追加することによって、構築中に修正される。特定のノード 34 の修正が完了すると、ノードにはラベルが付けられて、ラベル付きノードを提供し得る。ノード 34 のそれぞれが修正（及びラベル付け）された後、完成したデータフローグラフ 17 が（例えば、記憶システム 26 に）記憶され、以下に記載するように、他のデータフローグラフを生成するために使用される。

【0051】

いくつかの実施例では、データフローグラフ 17 のノード 34 のそれぞれは、最初は未修正である。例えば、ノード 34 のそれぞれは、上述のように、指定された動作要素 35 a' 及びデータソース又はデータシンク要素 35 b' を含むように後に修正される、空の動作プレースホルダフィールド 35 a 及びデータプレースホルダフィールド 35 b を有し得る。いくつかの実施例では、データフローグラフ 17 は、以前に完成したデータフローグラフであり、ノード 34 の一部又は全ては、動作要素 35 a' を保持する、対応する動作プレースホルダフィールド 35 a と、データソース又はデータシンク要素 35 b' を保持するデータプレースホルダフィールド 35 b と、を有する。完成した、かかるデータフローグラフ 17 は、（例えば、それぞれのフィールド 35 a、35 b に配置される追加の又は代替の要素 35 a'、35 b' を取得することによって）更に修正され、新しい又は修正されたデータフローグラフとして記憶され得る。

【0052】

いくつかの実施例では、初期ノード 34 a など特定のノードは「再利用」されて、以前のノード 34 a に関連付けられた、任意選択的にラベル付けされた新しいノードを生成する。初期ノード 34 a から新しいノードを生成するこの反復プロセスは、ユーザが所望の計算グラフのために指定機能を有するまで継続する。反復プロセスが完了すると、完成したデータフローグラフ 17 が提供される。完成したデータフローグラフ 17 は、例えば、初期ノード 34 a からインスタンス化された複数のノード 34 a ~ 34 n を含む。完成したデータフローグラフ 17 は、以下に記載するように、（例えば、記憶システム 26 に）記憶され、他のデータフローグラフを生成するために使用され得る。

【0053】

図 2 B は、完成した（例えば、修正された）データフローグラフ 17 のある実施例を示す。修正されたデータフローグラフ 17 は、動作要素を保持する、対応する動作プレースホルダフィールド 35 a と、データソース又はデータシンク要素を保持するデータプレースホルダフィールド 35 b とを有する、OP-0 ~ OP-6 とラベル付けされた 7 つのノードを含むものとして示されている。例えば、OP-0 とラベル付けされたノード 34 a は、「データセット I」のデータソース要素 37 b' が読み取られることを示す、読み取り動作要素 37 a' を含む。データフローグラフ 17 は、図 1 に示され、図 2 ~ 図 4 を参照して以下に詳述するデータフローグラフに対応する。修正されたデータフローグラフ 17 は、例えば、データ構造として記憶システム 26 に記憶される。

【0054】

ここで図 3 A ~ 図 3 C を参照すると、データフローグラフ 17 及び変換されたデータフローグラフ 19 の視覚化が、構築の様々な段階で示されている。例えば、図 3 A は、完成した（例えば、修正された）データフローグラフ 17 の視覚化 70 を描写する。図 3 B は、変換エンジン 18 による最適化後の完成したデータフローグラフ 17 の視覚化 72 を示す。視覚化 72 に示されるように、完成したデータフローグラフ 17 の最適化により、図 4 D に関連して以下に記載するように、変換エンジン 18 が冗長である、ないしはグラフの機能に不要であると判定した再フォーマット及びフィルタ動作が削除された。図 3 C は、最適化され、完成したデータフローグラフ 17（図 3 B）と同じ機能を実行するために、変換エンジン 18 によって生成された、変換されたグラフ 19 の視覚化 74 を示す。視覚化 74 に示されるように、完成したデータフローグラフ 17 に不要であった追加の並べ替え及び分割コンポーネントが変換されたデータフローグラフ 19 に追加されており、コンパイル及び実行を容易にする。

【0055】

一般に、変換エンジン 18 は、データフローグラフ 17 で指定された動作のうちの 1 つ以上に従ってデータを処理するために、又は最適化若しくは変換を行わないデータ処理と比較して、データフローグラフ 17 で指定された動作のうちの 1 つ以上に従ったデータ処理を改善するために必要とされ得る、最適化若しくは他の変換、又はその両方を実行する。例えば、変換エンジン 18 は、とりわけ、1 つ以上の並べ替え動作、データ型動作、データフローグラフ 17 で指定されたキーに基づいた結合動作など結合動作、分割動作、自動並列動作、又はメタデータを指定する動作を追加して、データフローグラフ 17 の所望の機能を有する、変換されたデータフローグラフ 19 を生成する。いくつかの実施例では、変換されたデータフローグラフ 19 は、1 つ以上のデータフローグラフ最適化ルールを変換されたデータフローグラフに適用して、最適化適用前の変換されたデータフローグラフの計算効率と比較して、変換されたデータフローグラフの計算効率を改善することによって最適化されたデータフローグラフである（又は、最適化されたデータフローグラフに変換される）。データフローグラフ最適化ルールとしては、例えば、図 4 D に関連して以下に記載するように、例えば、とりわけ、デッド若しくは冗長コンポーネントの削除、早期フィルタリング、又はレコードの縮小が挙げられ得る。

【0056】

ここで図 4 A を参照すると、計算グラフ（例えば、データフローグラフ 17）を編集するための例示的なグラフィックエディタインターフェース 50 が示されている。エディタ UI ジェネレータ 14 は、クライアントデバイス 20 に UI データ 30 を提供して、クライアントデバイスにグラフィカルエディタインターフェース 50 を生成させる。一般に、グラフィックエディタインターフェースは、キャンバス部分 52 と、カタログ部分 54 と、を含む。カタログ部分 54 は、データセット選択アイコン 55 a ~ 55 n を含むデータセット選択部分 55 と、変換選択アイコン 57 a ~ 57 n を含む変換選択部分 57 と、を含む。データセット選択アイコン 55 a ~ 55 n は、記憶システム 26 に記憶され、データセット選択アイコンによって表される、対応する要素（例えば、データソース又はデータシンク要素 35 b'）を参照する。いくつかの実施例では、これらの要素は、動作が実行され得るか、又はデータが記憶され得るデータセット（又はデータセットへのポインタ）を含み得る。変換選択アイコン 57 a ~ 57 n は、記憶システム 26 に記憶され、変換選択アイコンによって表される、対応する要素（例えば、動作要素 35 a'）を参照する。いくつかの実施例では、これらの要素は、実行される動作のタイプ（例えば、書き込み動作、結合動作など）を指定するデータ（又はデータへのポインタ）を含む。

【0057】

動作中、クライアントデバイス 20 のユーザは、変換選択部分 57 からアイコン（例えば、変換選択アイコン 57 a ~ 57 n のうちの 1 つ）を選択し、例えば、アイコンをキャンバス 52 にドラッグする。ユーザはまた、カタログ選択部分 55 からアイコン（例えば、データセット選択アイコン 55 a ~ 55 n のうちの 1 つ）を選択し、例えば、アイコンをキャンバス 52 にドラッグする（キャンバス 52 上の所望の変換選択アイコン上にデータセット選択アイコンをドラッグすることを含み得る）。この例では、記憶されたデータ構造又は他の記憶データは、図 4 C に関連して以下に記載するように、データフローグラフ 17 の修正に使用される要素（例えば、動作要素 35 a' 及びデータソース又はデータソース要素 35 b'）にアイコンを関連付ける。ユーザの選択により、データフローグラフエンジン 16 は、選択されたデータセット選択アイコン及び選択された変換選択アイコンを示すデータなどアイコン選択データ 32 を、生成されたエディタインターフェースから受信する。キャンバス 52 内のアイコンは、例えば、グラフ生成システム 12 によって自動的に接続され得る。いくつかの実施例では、キャンバス 52 内で、あるアイコンを別のアイコンの下に配置すると、グラフ生成システム 12 にこれらのアイコン間を自動的に接続させるように、アイコンが自動的に接続される。いくつかの実施例では、ユーザは、アイコンを接続する選択権を有する。

【0058】

図 4 B は、キャンバス部分 52 に完成したデータフローグラフの視覚表現 60 を有する

エディタインターフェース 50 を示す。完成したデータフローグラフの視覚表現 60（便宜上本明細書では、完成したデータフローグラフ 60 とも呼ばれる）は、ユーザがカタログ部分 54 からアイコンを選択し（例えば、データセット選択部分 55 及び変換選択部分 57 から選択し）、所望の配置でキャンバス 52 上にアイコンを配置することによって生成された。図 4 A から図 4 B へと進むために、ユーザは、アイコン（例えば、アイコン 55 a ~ n、57 a ~ n）をキャンバス 52 にドラッグして、所望の視覚データフローグラフ 60 を生成することができる。完成したデータフローグラフ 60 は、データセット I に適用される読み取り動作を表す第 1 のコンポーネントアイコン 60 a と、データセット I に適用される読み取り動作を表す第 2 のコンポーネントアイコン 60 b と、キー値に基づいて、データセット I 及びデータセット I に適用される結合動作を表す第 3 のコンポーネントアイコン 60 c と、結合動作 60 c の出力に適用される再フォーマット動作を表す第 4 のコンポーネントアイコン 60 d と、再フォーマット動作 60 d の結果に適用されるフィルタ動作を表す第 5 のコンポーネントアイコン 60 e と、フィルタ動作 60 e の結果に適用されるロールアップ動作を表す第 6 のコンポーネントアイコン 60 f と、データシンク I に適用される書き込み動作を表す第 7 のコンポーネントアイコン 60 g と、を含む。

【0059】

完成したデータフローグラフ 60 を構成する選択されたデータセットアイコン及び選択された変換アイコン（及びいくつかの実施例では、それらの配置）を示すアイコン選択データ 32 は、データフローグラフエンジン 16 によって受信される。図 4 C に示されるように、データフローグラフエンジン 16 は、選択データ（例えば、アイコン選択データ 32）に基づいてデータフローグラフ 17 を修正して選択された要素を含め、指定されたデータフローグラフ（例えば、視覚化 60 で視覚化された、指定されたデータフローグラフ）の機能を有する完成されたデータフローグラフ 17 を生成する。そのようにするために、データフローグラフエンジン 16 は、アイコン選択データ 32 を処理して、アイコン選択データに対応する動作要素 35 a'、又はデータ要素 35 b'、又はその両方を識別する。次いで、データフローグラフエンジン 16 は、識別した要素 35 a'、35 b' を使用して、データフローグラフ 17 の各ノード 34 a ~ 34 n のフィールド 35 a、35 b を追加（又は修正）することによってデータフローグラフ 17 を生成する。例えば、データフローグラフエンジン 16 によって受信されたアイコン選択データ 32 は、データフローグラフ 60 が、データセット I を表すデータ選択アイコン 55 a に適用される読み取り動作を表す変換選択アイコン 57 a を含むように指定し得る。データフローグラフエンジン 16 は、選択されたアイコン 57 a を、読み取り動作を表す動作要素 35 a' に関連付ける、1 つ以上の記憶データ構造又は他の記憶データを用いてこのアイコン選択データ 32 を処理し得る。次いで、データフローグラフエンジン 16 は、データフローグラフ 17 のノード（例えば、ノード 34 a）の動作フィールド 35 a に読み取り動作に対応する動作要素 35 a'（又は動作要素 35 a' を指すリンク）を追加（又は修正）し得る。同様に、この例では、データフローグラフエンジン 16 は、データセット選択アイコン 55 a を、データセット I を表すデータソース要素 35 b' に関連付ける、1 つ以上の記憶データ構造又は他の記憶データを用いて、受信したアイコン選択データ 32 を処理し得る。次いで、データフローグラフエンジン 16 は、データフローグラフ 16 のノード 34 a のデータフィールド 35 b に、データセット I に対応するデータソース要素 35 b'（又はデータソース要素 35 b' を指すリンク）を追加（又は修正）し得る。

【0060】

ここで図 4 D を参照すると、完成した（例えば、修正した）データフローグラフ 17 が変換エンジン 18 に提供される。一般に、完成したデータフローグラフ 17 は、（1）冗長データ処理動作を表すノードを含み得る；（2）結果がその後に使用されないデータ処理動作を実行する必要がある；（3）並列処理が可能である場合、順次処理を不必要に実行する必要がある；（4）所望の結果を得るために必要とされるよりも多くのデータにデータ処理動作が適用され得る；（5）複数ノードにわたる計算が生じ、各データ

10

20

30

40

50

フローグラフノードのデータ処理が、コンピュータプログラムの専用スレッド、専用コンピュータプログラム（例えば、オペレーティングシステム内のプロセス）、又は専用計算デバイスによって実行される場合には、計算を実行する計算コストが著しく増加し得る；

（６）より少量の計算を必要とするより弱いタイプのデータ処理動作（例えば、グループ内並べ替え動作、グループ内ロールアップ動作など）で十分である場合に、より大量の計算を必要とするより強いタイプのデータ処理動作（例えば、並べ替え動作、ロールアップ動作など）を実行する必要がある；（７）処理作業の重複を必要とし得る；又は（８）とりわけ、データ処理に有用な、若しくは必要な動作又は他の変換、又はそれらの組み合わせを含まないことがあり得る。

【００６１】

したがって、変換エンジン１８は、データフローグラフ１７で指定された動作に従ってデータを処理するために、又は最適化若しくは変換を行わないデータの処理と比較して、データフローグラフ１７で指定された動作に従ってデータの処理を改善するために有用である、又は必要とされ得る、以下の最適化若しくは他の変換のうちの１つ以上、又はその両方をデータフローグラフ１７に適用する。例えば、上述のように、ユーザは、所望のアイコンを選択し、並べ替え及び分割動作など低レベルの実施詳細を指定する必要なく、それらのアイコンをキャンバスに配置することによって、データフローグラフ１７を作成し得る。しかしながら、これらの動作は、データフローグラフをコンパイルし、実行して、データフローグラフ１７で指定された動作に従ってデータを処理するために、変換されたデータフローグラフ１９において有用であり得る、若しくは必要とされ得るか、又はデータフローグラフ１７で指定された動作に従ったデータの処理を改善し得るか（例えば、処理速度を上げること、計算リソースの消費を低減することなどによって）、又はその両方であり得る。したがって、変換エンジン１８は、変換されたデータフローグラフ１９に、とりわけ並べ替え動作、データ型動作、指定キーとの結合動作、分割動作、自動並列動作、又はメタデータを指定する動作など１つ以上の動作を追加して、データフローグラフ１７で指定された動作を最適化し得るか、又は実施し得る。変換されたデータフローグラフ１９に追加された動作のうちの少なくともいくつかは、データフローグラフ１７に存在しないか、ないしは別の方法で表されていないことがある。

【００６２】

いくつかの例では、これらの動作を追加するために、変換エンジン１８は、追加される動作を表す１つ以上のノード３４を、変換されたデータフローグラフ１９を生成するために使用されるデータフローグラフ１７に挿入し得る。いくつかの例では、変換エンジン１８は、データフローグラフ１７のノードを修正することなく、変換されたデータフローグラフ１９内で追加される動作を直接挿入することができる。変換エンジン１８は、これらの動作を、それらの対応する変換されたデータフローグラフ１９の生成時に全てのデータフローグラフ１７に追加し得る、これらの動作を、データフローグラフ１７に含まれる動作（以下に記載するようにパターンマッチング技術を使用して識別され得る）に基づいて追加し得る、又はいくつかの他の最適化ルールに基づいてこれらの動作を追加し得る。

【００６３】

変換エンジン１８はまた、１つ以上のデータフローグラフ最適化ルールをデータフローグラフに適用して、変換されたデータフローグラフの計算効率を改善することによって、とりわけ、デッドコンポーネント若しくは冗長コンポーネントを削除することによって（例えば、デッドコンポーネント若しくは冗長コンポーネントに対応する１つ以上のノード３４を削除することによって）、データフロー内の以前のフィルタリング工程を移動することによって（例えば、フィルタリングコンポーネントに対応する１つ以上のノード３４を移動することによって）、又はレコードを縮小することによって、データフローグラフ１７（又は変換されたデータフローグラフ１９自体）を最適化し得る。このようにして、変換エンジン１８は、容易に修正可能なデータフローグラフ１７を最適化し、コンパイルに好適である、最適化された、変換されたデータフローグラフ１９に変換する。

【００６４】

本明細書に記載のデータフローグラフの最適化は、以下の最適化手段のうちの1つ以上を含み得る。データフローグラフの追加の最適化は、米国特許出願第15／993,284号、名称「Systems and Methods for Dataflow Graph Optimization」に記載されている最適化のうちの1つ以上を含み得、その全内容は、参照により本明細書に組み込まれる。

【0065】

いくつかの例では、変換エンジン18は、それぞれの動作を表す、データフローグラフ17内の2つの隣接するノード（例えば、ノード34）を識別し得、第2の動作は、動作のうちの1つが冗長であるように、第1の動作の効果を複製するか、又は無効化する。したがって、変換エンジン18は、変換されたデータフローグラフ19の生成時に、冗長動作を表すノード34（例えば、複製された動作又は無効化された動作を表すノード）を削除することによってデータフローグラフ17を最適化し得る。例えば、変換エンジン18は、同一動作を有する2つの隣接するノード34を識別し得る。同一動作を実行する2つの隣接するノードは、典型的には冗長であるため、両方の動作を実行する必要はなく、2つの隣接するノードのうちの1つを削除して、データフローグラフを最適化することができる。別の例として、変換エンジン18は、再分割動作（異なる計算デバイスでの並列処理のためにデータを分割する）を表す第1のノード、続いて、シリアル化動作（単一の計算デバイスによる順次処理のために全データを統合するように動作する）を表すノードを有する2つの隣接するノード34を識別し得る。再分割の効果は、後続のシリアル化動作によって無効化されるため、再分割動作を実行する必要はなく（例えば、再分割動作は冗長である）、再分割動作は、最適化プロセス中に変換エンジン18によって削除され得る。

【0066】

いくつかの例では、変換エンジン18は、他の動作を表す1つ以上の他のノードと交換可能である、第1の動作を表す第1のノードを識別し得る。第1のノードが1つ以上の他のノードに交換可能である場合、変換エンジン18は、1つ以上の他のノードのうちの少なくとも1つを用いて第1のノードの順序を変更することによって（例えば、ノード34の順序を組み替えることによって）、データフローグラフを最適化することができる。このようにして、変換エンジン18は、結果を変更せずに、データフローグラフによる処理効率、速度を改善するか、ないしは別の方法で最適化する方法で、ノード及び対応する動作を順序付けることによって、データフローグラフを最適化することができる。更に、このようにノードを交換することにより、変換エンジン18は、他の最適化を適用することが可能であり得る。前の例を再び参照すると、変換エンジン18は、第1の並べ替え動作を表す第1のノードが第2の並べ替え動作を表す第2のノードに隣接して配置されるように、1つ以上の他のノードのうちの少なくとも1つを用いて第1のノードの順序を変更し得る。結果として、第1の並べ替え動作及び第2の並べ替え動作は冗長になり、変換エンジン18は、第1の並べ替え動作又は第2の並べ替え動作のうちの1つを削除することによって（例えば、変換されたデータフローグラフ19の生成時に、データフローグラフ17から対応するノード34を削除することによって）、データフローグラフを最適化することができる。

【0067】

いくつかの例では、変換エンジン18は、未使用である、ないしは別の方法で不要である動作を表す「デッド」ノードを識別し、削除し得る。例えば、変換エンジン18は、結果が参照されないか、又は使用されない動作（例えば、並べ替え動作から生じた順序が不要であるか、又はその後の処理で依拠されていないために参照されない、並べ替え動作）を表す1つ以上のノードを識別し得る。したがって、変換エンジン18は、デッド動作又は未使用動作を削除することによって（例えば、変換されたデータフローグラフ19の生成時に、対応するノード34を削除することによって）データフローグラフを最適化し得る。

【0068】

いくつかの例では、変換エンジン18は、1つ以上のノードで強度低減最適化を実行す

ることができる。例えば、変換エンジン 18 は、第 1 の種類の第 1 の動作（例えば、主要キーでの第 1 の並べ替え動作、主要キーでの第 1 のロールアップ動作など）を表す第 1 のノードを識別し、続いて、第 2 のより弱い種類の第 2 の動作（例えば、非主要キーでの第 2 の動作、グループ内並べ替え動作、非主要キーでの第 2 のロールアップ動作、グループ別ロールアップ動作など）を表す第 2 のノードを識別し得る。第 1 の動作によるデータ処理には、第 2 のより弱い動作によるデータ処理データよりも多くの計算リソースを必要とし得るため、変換エンジン 18 は、第 1 の動作を第 2 の動作と置き換える強度低減最適化を実行し得る（例えば、第 1 のノードの動作要素 35 a ' 若しくはデータ要素 35 b '、又はその両方を第 2 のノードのそれらと置き換えることによって）。

【0069】

更に別の例として、変換エンジン 18 は、2 つ以上のノードを統合することによって、データフローグラフを最適化し得る。例えば、変換エンジン 18 は、1 つ又は複数の計算デバイスで実行される異なるプロセスによって実行され得る動作を表す別個のノードを識別し得、別個のノード及びそれぞれの動作を単一ノードに統合することによって、データフローグラフを最適化し得、その結果、全ての動作が単一の計算デバイスで実行される単一プロセスによって実行されるようになり、プロセス間（及び潜在的にデバイス間）通信のオーバーヘッドを低減することができる。変換エンジン 18 はまた、多くの他の組み合わせの中でも、組み合わせられ得る 2 つ以上の別個の結合動作又はロールアップ動作と組み合わせられ得るフィルタリング動作など、組み合わせられ得る他のノードを識別することができる。

【0070】

いくつかの例では、変換エンジン 18 は、別々に実行されたときにより効率的であり得るいくつかの動作を実行するように構成されたノードを識別し得、並列処理用にいくつかの動作（例えば、自動並列動作）のうちの 1 つ以上を別個のノードに分割する、データフローグラフの直列ー並列最適化を実行し得る。次いで、動作は、1 つ又は複数の計算デバイスで実行される異なるプロセスを使用して並列に実行され得る。次いで、変換エンジン 18 は、マージ動作を追加して、並列動作の結果をマージすることができる。同様に、いくつかの例では、変換エンジン 18 は、大量のデータチャンク（例えば、大型のテーブル及びインデックスに対応するデータ）を含むデータフローグラフ内でポイントを識別し得、データフローグラフの分割最適化を実行して（例えば、自動分割動作）、データをより小さなパーティションに分割し得る。次いで、パーティションは、直列又は並列に（例えば、自動分割動作を自動並列動作と統合することによって）処理され得る。処理されるデータのサイズを低減することにより、若しくは並列処理用の動作を分けることにより、又はその両方により、変換エンジン 18 は、得られたデータフローグラフ 19 の効率を大幅に改善することができる。

【0071】

いくつかの例では、変換エンジン 18 は、変換されたデータフローグラフ 19 の生成時に、幅低減最適化を実行することができる。例えば、変換エンジン 18 は、データ（例えば、削除されるデータ）が後続の動作において使用されず、処理の一環として伝搬される必要はないために後続の動作の実行前にデータフローグラフの特定の時点で削除されるデータ（例えば、1 つ以上のデータ列）を識別し得る。別の例として、データフローグラフ内のノードは、いくつかの動作を実行するように構成され得、これらの動作の一部の結果は、未使用であり得る。したがって、変換エンジン 18 は、（例えば、ノードを挿入して識別された時点でデータを削除することによって、複数の動作を実行するように構成されたノードを結果が使用される動作のみを実行するように構成された別のノードに置き換えることによって）、未使用データ、ないしは不要データを削除する幅低減最適化を実行し得る。このようにして、変換エンジン 18 は、後続の動作を通じてデータを持ち運ぶためにデータフローグラフによって必要とされる計算リソースの量を低減することによって（例えば、利用されるネットワーク、メモリ、及び処理リソースを削減することによって）データフローグラフを最適化する。

【0072】

1つ以上の最適化を適用するデータフローグラフ（例えば、データフローグラフ17）の部分を識別するために、変換エンジン18（又はグラフ生成システム12の別のコンポーネント）は、データフローグラフパターンマッチング言語を用いることができる。データフローサブグラフパターンマッチング言語は、以下で詳述するように、最適化のためにデータフローグラフ内の特定のノード又は動作を識別するために1つ以上の表現を含み得る。例えば、パターンマッチング言語は、統合動作最適化ルールを使用して、データフローグラフ内の単一ノードによって統合され、表され得る、それぞれの一連の計算を表す、少なくとも閾値長さ（例えば、少なくとも2、3、4、5など）の一連のノードを識別するための表現を含み得る。かかるパターンを識別することは、上記の統合動作最適化ルールの適用を容易にし得る。1つのかかる表現の好ましいが非限定的な例は、「 $A \rightarrow B \rightarrow C \rightarrow D$ 」であり、統合され得る一連の4つの連続するデータ処理動作を識別するのに役立ち得る。

10

【0073】

別の例として、パターンマッチング言語は、特定の種類のノードが他のノードと交換可能であってデータフローグラフを最適化する、データフローグラフの部分を識別するための表現を含み得る。これは、複数の異なる種類の最適化ルールをデータフローグラフに適用することを促進し得る。データ処理システムが、処理結果を変更することなくデータフローグラフ内の1つ以上のノードの順序を変更できると判定すると、データ処理システムは、最適化ルールを適用可能な部分を識別するために、（交換動作を通じて利用可能な自由度によって許可されるように）データフローグラフの構造に対する変更を検討することが可能になる。交換ベースの変更を検討した結果として、1つ以上の最適化ルールが、そうでなければルールが適用可能ではないグラフの一部に適用可能になり得る。

20

【0074】

例えば、上述のように、最適化ルールは、それぞれの並べ替え動作を表す初期データフローグラフ内の2つの隣接するノードを識別することを含み得、第2の並べ替え動作は、第1の動作が冗長であるように第1の動作の効果を無効化する。定義により、かかる最適化ルールは、並べ替え動作を表す隣接するノードを有しないデータフローグラフには適用されない。しかしながら、第1の並べ替え動作を表す第1のノードが1つ以上の他のノードと交換可能である場合、第1の並べ替え動作を表す第1のノードが第2の並べ替え動作を表す第2のノードに隣接して配置されるように、1つ以上の他のノードのうちの少なくとも1つを用いて第1のノードの順序を変更することが可能であり得る。このようにノードを交換する結果として、冗長な第1の並べ替え動作を削除する最適化ルールがデータフローグラフに適用され得る。

30

【0075】

したがって、いくつかの例では、パターンマッチング言語は、データフローグラフ内でのノードの順序が変更され得る状況において、データフローグラフのサブグラフを識別するために1つ以上の表現を含み得る。一例として、表現「 $A \rightarrow (\dots) \rightarrow B$ 」（A及びBのそれぞれは、並べ替え、マージなど任意の好適なデータ処理動作であり得る）は、ノード「A」（例えば、動作「A」を表すノード）及びノードB（動作Bを表す）、並びにノードAと交換可能である、ノードAとノードBとの間にある1つ以上のノード（例えば、ノードの順序が変更される場合、これらのノードによって実行される処理の結果は変化しない）を有するデータフローグラフの部分を見出すために使用され得る。かかる部分が識別されると、データフローグラフは、ノードAをノードBに隣接するように移動させて、部分「AB」を得ることによって変更され得るか、又は最適化され得る。特定の例として、データフローグラフがノードACDBを有し、動作Aが動作C及びDと交換可能である場合、データフローグラフは、「CDAB」になるように変更され得る。次に、変換エンジン18は、最適化ルールが部分「AB」に適用されるかどうかを検討し得る。例えば、動作Aが並べ替えであり、動作Bが並べ替えであった場合、変換エンジン18は、これら2つの並べ替えが単一の並べ替えで置き換えられて、データフローグラフを最適化し得

40

50

るかどうかを判定しようと試み得る。

【0076】

別の例として、表現「 $A \rightarrow (\dots) \rightarrow B^*$ 」は、ノードA、第2のノードB、及びノードBと交換可能である、これらのノード間にある1つ以上のノードを有するデータフローグラフの部分を見出すために使用され得る。特定の例として、データフローグラフがノードACDBを有し、動作Bが動作C及びDと交換可能である場合、データフローグラフは、「ABCD」となるように変更され得るか、又は最適化され得る。次に、変換エンジン18は、最適化ルールが部分「AB」に適用されるかどうかを考慮し得る。

【0077】

別の例として、表現「 $A \rightarrow (\dots) \rightarrow B^{**}$ 」は、ノードA、ノードB、及びノードBと交換不能である、ノードAとノードBとの間にある1つ以上のノード（例えば、C及びD）を有するデータフローグラフの部分を見出すために使用され得る。その場合、システムは「プッシュ」交換を試み得る（可能であれば、ノードC及びDがノードAの左側へとプッシュされる）。特定の例として、データフローグラフがノードACEDBを有し、動作Bが動作Eと交換可能であるが、動作C及びDとは交換不能であるとする、データフローグラフは、「CDAEB」となるように変更され得る。つまり、BはEと交換されるが、C及びDはAの左側へとプッシュされる。

【0078】

更に別の例として、表現「 $A^{**} \rightarrow (\dots) \rightarrow B$ 」は、ノードA、ノードB、及びノードAと交換不能である、ノードAとノードBとの間にある1つ以上のノード（例えば、C及びD）を有するデータフローグラフの部分を見出すために使用され得る。その場合、システムは「プッシュ」交換を試み得る（可能であれば、ノードC及びDがノードBの右側へとプッシュされる）。特定の例として、データフローグラフがノードACEDBを有し、動作Aが動作Eと交換可能であるが、動作C及びDとは交換不能であるとする、データフローグラフは、「EABCD」となるように変更され得る。つまり、ノードAはEと交換されるが、C及びDはBの右側へとプッシュされる。

【0079】

一般に、最適化及び変換プロセスは反復であり、最適化又は変換（18a）の各反復により、試験（18b）が、更なる最適化又は変換は不可能である、不要である、又は所望されないことを示すまで、データフローグラフ17は変換される。例えば、変換エンジン18は、（1）第1の最適化ルールを選択すること、（2）第1の最適化ルールを適用するデータフローグラフ17の第1の部分を識別すること、及び（3）第1の最適化ルールをデータフローグラフ17の第1の部分に適用すること、によって、データフローグラフ17を変換し得る。その後、データ処理システムは、本明細書に記載の別の1つ以上の追加の最適化がデータフローグラフ17に適用可能であるかどうか、又はコンパイルされ、実行され得る変換されたデータフローグラフ19を生成するために必要であるかどうかを判定し得る。追加の最適化が適用可能である場合、変換エンジン18は、（1）第1の最適化ルールとは異なる第2の最適化ルールを選択すること、（2）第2の最適化ルールを適用するデータフローグラフ17の第2の部分を識別すること、及び（3）第2の最適化ルールをデータフローグラフ17の第2の部分に適用すること、によって、データフローグラフ17の更新を続行し得る。

【0080】

更なる最適化又は変換が存在しない時点で、変換エンジン18は、変換されたデータフローグラフ19をコンパイラシステム22に出力し、コンパイラシステム22は、変換されたデータフローグラフ19を実行可能な計算グラフ（例えば、実行可能なプログラム23）にコンパイルし、この実行可能な計算グラフは、実行及び記憶のためにデータ処理システム24及び記憶システム26に提供される。

【0081】

図5は、計算データフローグラフを生成するための例示的なプロセス500のフローチャートを示す。プロセス500は、例えば、システム10の1つ以上のコンポーネントに

10

20

30

40

50

よって実行され得る。

【0082】

プロセス500の動作は、データを提供して、キャンバス部分と、カタログ部分と、を含むグラフィカルエディタインターフェースを生成することを含み、カタログ部分は、キャンバス部分において、計算（502）のロジックを視覚的に描写するための1つ以上の選択可能なアイコンを含む。例えば、カタログ部分は、データセット選択アイコン及び変換選択アイコンを含み得、選択された各アイコンは、動作又はデータソース若しくはシンクを指定するデータカタログからデータにアクセスする命令を表し得る。

【0083】

キャンバス部分に描写された計算のロジックを表すアイコン選択データを受信する（504）。アイコン選択データは、カタログ部分から選択され、キャンバス部分に含まれる1つ以上の選択可能なアイコンのうちの少なくとも1つを指定することができる。

【0084】

受信したアイコン選択データに基づいて、第1のデータフローグラフを生成する（506）。第1のデータフローグラフは、データ処理における第1のコンピュータ動作を表す複数の第1のノードを含み、第1のコンピュータ動作のうちの少なくとも1つは、宣言動作である。一般に、宣言動作は、結果の達成方法を必ずしも指定せずに、データの処理の1つ以上の結果の1つ以上の特性を指定するものである。第1のノードは、キャンバス部分で指定されたロジックを表し得、第1のノードのうちの少なくとも1つは、カタログ部分から選択された選択可能なアイコンを表す。いくつかの実施例では、第1のノードの一部又は全ては、動作を保持するための動作プレースホルダフィールドと、データのソース又はシンクを保持するためのデータプレースホルダフィールドと、を含む。第1のデータフローグラフを生成することは、動作プレースホルダフィールドに保持された動作の要素を記憶システムから取得して、動作プレースホルダフィールドに動作（又は動作へのリンク）を追加（又は修正）することと、データプレースホルダフィールドに保持されたデータソース又はデータシンクの要素を記憶システムから取得して、データプレースホルダフィールドにデータのソース又はシンクを指すリンクを追加（又は修正）することと、を含み得る。

【0085】

いくつかの実施例では、第1のデータフローグラフの視覚化は、例えば、グラフィカルエディタインターフェースのキャンバス部分にレンダリングされる。いくつかの実施例では、第1のノードのうちの1つ以上がラベルを付けられて、ラベル付きノードを提供する。ラベルは、キー、値、名前、及びソースのうちの1つ以上を参照し得る。いくつかの実施例では、ラベル付きノードの視覚化は、キャンバス部分にレンダリングされる。

【0086】

第1のデータフローグラフは、第1のコンピュータ動作に従ってデータを処理するために第2のデータフローグラフに変換される（508）。第2のデータフローグラフは、第2のコンピュータ動作を表す複数の第2のノードを含み、第2のノードのうちの1つ以上は、1つ以上の命令動作を表す。一般に、命令動作は、宣言動作によって指定されたロジックを実施する方法を（例えば、プログラミング言語又は他の機械可読コードで）指定するものである。1つ以上の命令動作は、第1のデータフローグラフ内の第1のノードによって表されない。いくつかの実施例では、第2のデータフローグラフに表され、第1のデータフローグラフには表されない第2の動作のうちの少なくとも1つは、並べ替え動作、データ型動作、指定キーとの結合動作、又は分割動作を含む。いくつかの実施例では、第2の動作のうちの少なくとも1つは、自動並列動作又は自動分割動作を含み、自動並列ノード又は自動分割ノードを第2のデータフローグラフに挿入することを含み得る。いくつかの実施例では、第2の動作のうちの少なくとも1つは、第2のノードのうちの1つ以上の間でメタデータを指定する動作を含む。いくつかの実施例では、ラベル付きノードであり得る第1のデータフローグラフの第1のノードは、計算データフローグラフである第2のデータフローグラフにコンパイルされ得る。ノード又はノードに記憶された要素は、最

10

20

30

40

50

適化され得る。

【0087】

いくつかの実施例では、第2の動作のうちの1つ以上は、(i)第1のデータフローグラフで指定された第1の動作のうちの1つ以上に従ってデータを処理するために必要であるか、又は(ii)1つ以上の追加動作を伴わないデータ処理と比較して、第1のデータフローグラフで指定された第1の動作のうちの1つ以上に従ったデータ処理を改善する。

【0088】

いくつかの実施例では、第2のデータフローグラフは、1つ以上のデータフローグラフ最適化ルールを第2のデータフローグラフに適用して、適用前の第2のデータフローグラフの計算効率と比較して、第2のデータフローグラフの計算効率を改善することによって、最適化されたデータフローグラフに変換される。データフローグラフ最適化ルールとしては、例えば、図4A～図4Dに関連して以下に記載するように、例えば、とりわけ、デッドコンポーネントの削除、早期フィルタリング、又はレコードの縮小が挙げられ得る。

【0089】

第2のデータフローグラフは、データストア(510)に記憶される。いくつかの実施例では、第2のデータフローグラフは、コンパイラシステムによってコンパイルされて、コンパイルされたデータフローグラフ(例えば、実行可能プログラム)を生成する。コンパイルされたデータフローグラフは、実行及び記憶のためにデータ処理システム及び記憶システムに提供され得る。

【0090】

いくつかの実施例では、プロトタイプノードがアクセスされ、アクセスされたプロトタイプノードからパラメータをコピーして第1のノードのうちの少なくとも1つを修正するアルゴリズムが適用される。少なくとも1つのノードの少なくとも1つのパラメータは、プロトタイプノードによって上書きされない設定パラメータであり得る。プロトタイプノードは、少なくとも1つのノード上のポート又はノード自体を宣言することができる。いくつかの実施例では、プロトタイプノードは、エディタインターフェースのキャンバスに示されるコンポーネントのパラメータを宣言する。プロトタイプを適用することは、プロトタイプからの記述子で既存のパラメータの記述子を置き換え得るが、パラメータの既存の値は置き換え得ない。

【0091】

字句構造：

データフローグラフ17などデータフローグラフは、字句構造を有する。いくつかの実施例では、字句構造は、記号、キーワード、数字、文字列、コード様文字列、及び他の字句要素を含む。

【0092】

記号は、文字、数字、及び句読点(例えば、アンダースコア又はピリオド)を含み得、少なくとも1つの文字は、句読点文字ではない。いくつかの実施例では、記号の最初の文字は数字であることができないというルールなど、他のルールが採用され得る。

【0093】

キーワードは、小文字のみを含む短い記号を含み得る。列挙される全キーワードに対して構造(例えば、データ構造)が開発され得る。

【0094】

数字は、任意の長さを有する符号付き整数であり得る。パラメータ値である数字については、大きすぎて表せない数字(例えば、int8として)は、数値の代わりに文字列値を用いてパラメータに変換され得る。

【0095】

文字列は、一重引用符又は二重引用符を用いて引用され得る。文字列内では、閉じ引用文字は、バックスラッシュでエスケープされ得る。

【0096】

コード様文字列は括弧で引用され得る。括弧内では、一重及び二重引用文字列のように

10

20

30

40

50

、均衡した入れ子状の括弧が許可される。この表現は、キー指定子、変換、メタデータ、及び複数行のテキストパラメータ値に使用され得る。また、括弧内では、解析中に一定のインデントが削除され、シリアル化中に再挿入される。したがって、グラフ構造と一致するインデント埋め込みデータ操作言語（data manipulation language、DML）である。
【0097】

ポート、フロー、及び分岐：

ポートは、入力ポート及び出力ポートであり得る。各ポートは、ポートがそれらのコンポーネントのデフォルトの入力又は出力ポートではない限り、名前（例えば、「in」、「out」、「reject0」）を有する（デフォルトのポートである場合、名前は省略され得る）。ポートはフローを有する。フローは、1つの出力ポートを1つの入力ポートに接続することができる。ポートはパラメータを有する。ポートは、グラフポートに結合され得る。例えば、入力ポートはグラフ入力ポートに結合され得、出力ポートはグラフ出力ポートに結合され得る。

10

【0098】

フローは、ノードとは別に、グラフの要素として指定され得る。フローは、ノードの上位レベルで指定され、デフォルト入力ポートへのフローを示し得るか、又は入力ポート内で指定され得る。

【0099】

フローは、一連のノードを分岐に入れることによって暗黙的に指定され得る。分岐内では、新しいノードを導入するためにノードキーワードは不要である。分岐はIDを有し、フローがIDによってノードを指定することができる場合、分岐を指定することができる。分岐からのフローは、分岐の最後のノードからのフローであり得る。分岐へのフローは、分岐の最初のノードへのフローであり得る。

20

【0100】

ノード又は分岐から1つのフローのみが存在する場合、当該ノード又は分岐は、その出力を消費するノードの内側で入れ子状になった、フローの一部として指定され得る。

【0101】

グラフ要素は、グラフの他の要素を参照し得る。例えば、フローはノードを参照し得、パラメータは他のパラメータを参照し得、キーパラメータは、とりわけ、値を参照し得る。これらの参照を可能にするために、要素が導入されると、任意選択的に、IDが与えられ得る。異なる要素の種類は、別個のID空間を有し得る。いくつかの実施例では、IDは任意選択的であるが、IDがなければ、要素を参照することができない。

30

【0102】

パラメータ：

パラメータは、名前、ロケータであり得る値、及び解釈を有し得る。パラメータは、記号であり得る。句読文字（例えば、ピリオド）が先行するパラメータ名は、データフローグラフに対してプライベートであるとみなされ、他のグラフモデル（変換されたデータフローグラフなど）でのパラメータの生成には使用されない。他の文字パターン（例えば、先行する2つのピリオド）で開始するパラメータ名は遷移状態とみなされ、デフォルトでは、グラフと共に保存されない。これは、コマンドで上書きされ得る。

40

【0103】

パラメータは、それらに値を与えることによって定義される。パラメータ値は、文字列としてグラフパラメータに翻訳され得るが、他の形態は許可され得る。これらの他の形態は、内部モデルに、またシリアル化時に保存される。重複排除などグラフアルゴリズムでは、2つのパラメータの構造型が同一であり、それらの値が同一である場合、2つのパラメータは同一とみなされ得る。パターンマッチングでは、パターンパラメータの型が制御する（例えば、実際のパラメータは、当該型に変換されるパターンと比較される）。これにより、パターンは、例えば、パラメータ「sorted」が整数値1又は文字列値「true」（又は典型的には「true」として解釈されることになる他のパラメータ値のうちのいずれか）を有するノードに一致する「param sorted=true」を

50

使用して、true の値のパラメータをアサートすることができる。

【0104】

パラメータは、指定の位置に位置し得る。他のグラフモデルの生成時には、ロケータを保存することができる。パラメータの値を利用するアルゴリズムは、ロケータパスを解決し、ファイルの内容を読み取ることができる。

【0105】

グラフレベルのパラメータはデータフローグラフ 17 を変換されたデータフローグラフ 19 など他のグラフモデルに翻訳するときに、宣言を使用することができる。宣言は、多数の部分（例えば、入力又は出力パラメータ、パラメータの型、必須パラメータ、エクスポートされたパラメータ、パラメータコマンドラインインターフェース、環境から読み取られたパラメータ）を有し、それぞれデフォルト状態を有し得る。宣言のデフォルトは、特別なパラメータに値を提供することによって上書きすることができる。

【0106】

ノードプロトタイプ又は置換ノード内では、パラメータは、記述子を有し得る。記述子を使用して、パラメータの編集に使用されるユーザインターフェース及びパラメータ値に適用される検証を決定することができる。一例では、記述子は、パラメータキーワードの直後のプロパティのセットである。

【0107】

パラメータ記述子は、言語においてオープンエンドであり得る。例えば、新しいフィールドはプロトタイプファイルで使用され、実行時にコードを全く変更することなく、ユーザインターフェースで認識され得る。パラメータ値は、以下に記載する内蔵「apply-prototypes」パス内の記述子に対して、置換が生じるたびに検証される。

【0108】

読み書き可変グラフ：

データフローグラフは、コマンドラインユーティリティを使用することなどによって、ファイル又はコマンドライン引数から読み取られ得る。

【0109】

以下は、本明細書に記載の字句構造を使用した、完全なデータフローグラフの例である。

【0110】

【表 1】

```
node 0{
    statement 1[input]
    statement 2[processing instructions]
    statement 3[output]
}

node 1{
    label="discard trash";
}

flow from node 0 to node 1;
```

【0111】

データフローグラフのプロトタイプ：

パラメータ値は、パラメータとしてではなく引数としてスクリプトに現れることがあり、他のグラフモデルではパラメータを生成しない。代わりに、このパラメータは、データフローグラフ変換コードに対する命令であり、生成され、翻訳されたコンポーネントのコンポーネントパスを設定する。プロトタイプは、かかる種類の各ノードが、かかる同一の

パラメータ値を有することを可能にする。

【0112】

いくつかの実施例では、プロトタイプはデータフローグラフの一部として指定される。プロトタイプは直ちに適用されないことがあるため、グラフの解析及びシリアル化による変更は生じない。内蔵アルゴリズムは、プロトタイプからのパラメータをコピーするために提供され得る。設定済みのパラメータは、プロトタイプによって上書きされない。トランスレータによって使用される共通コンポーネントの種類は、列挙されており、名前を有する。可変ユーティリティは、コンポーネントのプロトタイプのデフォルトセットを生成する。これにより、明示的な指定の必要なパラメータが少なくなるため、グラフの生成が容易になる。

10

【0113】

プロトタイプは、プロトタイプアルゴリズムを適用すること、新しいノードを導入するアルゴリズム内のコードでプロトタイプルーチンを明示的に呼び出すことなどによって、いくつかの状況で、又はマッピングルールの適用によるノードの生成時に常に適用される。

【0114】

プロトタイプはまた、編集インターフェースを提示する際に使用するために、ノード、ポート、及びコンポーネントのパラメータを宣言するために使用され得る。この使用法は、以下の「PROTOファイル」に記載する。

【0115】

プロトタイプの適用により、パラメータの既存の値が置き換えられることはない。しかしながら、プロトタイプの適用により、既存パラメータの記述子がプロトタイプからの記述子で置き換えられる。これにより、パラメータの型に関する情報を記述子に配置することが可能になり、パラメータ値に適用される。

20

【0116】

データフローグラフのラベル：

ノードは、ゼロ以上のラベルを有し得る。コンポーネントがノードから生成されるとき、ノードのラベルは先頭文字が大文字になり、組み合わせてコンポーネントラベルを形成する。ノードが統合されると、それらのラベルセットも統合され得る。いくつかの実施例では、ラベルに優先権が割り当てられて、ラベルの増殖を低減し、有意な最終ラベルを生成することができる。ラベルは、名付きラベルであり得、とりわけ、ソース、キー、又は値を指し得る。

30

【0117】

データフローグラフのアルゴリズム：

いくつかの実施例では、1つ以上のアルゴリズムを適用してデータフローグラフを変換又は最適化することができ、アルゴリズムとしては、とりわけ、パターンマッチングパス、(上記の)プロトタイプを適用する内蔵パス、重複排除を行う内蔵パス、明示的複製を行う内蔵パス、及びフローをアंक罗斯する内蔵パスが挙げられる。

【0118】

いくつかの実施例では、データフローグラフのアルゴリズムは、パターンマッチングパスを含む。パターンマッチングパスは、ルールを含み、各ルールは、パターンと、置換と、を有する。パターンは、パターンが一致するために必要なパラメータ値を指定し得る。パターンは、2つ以上のノードを含み得る。ノードの種類はワイルドカード化され得るが、そうすることにより、計算コストが高くなり得る。置換は、ゼロノード(グラフの一致セクションを完全に除去するため)、単一ノード、又はフローによって接続された複数のノードからなり得る。ワイルドカードのノード種類がパターンで使用された場合、当該ワイルドカードが一致したノード種に結合され、置換で使用され得る。

40

【0119】

いくつかの実施例では、データフローグラフのアルゴリズムは、重複排除パスを含む。重複排除パスは、機能的に同一であるサブグラフを見出し、それらを統合する。2つのノードは、入力フローを有しない場合、又はそれらの入力フローが同一ノードからのフロー

50

であり、それらのパラメータが一致する場合、同一であるとみなされる。いくつかの実施例では、ノードが同等であるかをチェックするときに、出力フローは重視されない。ラベルはまた、ノードの同等性を試験するために重視されないことがある。

【0120】

重複排除で組み合わせられる2つのノードは、入力ノードから全てのラベルを収集することができる。重複排除は、単一ポートからの複数の出力フローを有するノードをもたらし得る。重複排除はまた、フローが単一ポートから分岐し、次いで再収束するバブルをグラフ内にもたらし得る。他のアルゴリズムは、以下に記載するように、明示的複製内蔵パス及びアंकロスフロー内蔵パスを含む。

【0121】

いくつかの実施例では、データフローグラフのアルゴリズムは、明示的複製パスを含む。データフローグラフでは、ごく少ないコンポーネントの種類が、出力ポートからの複数のフローをサポートする。しかし、データフローグラフモデルは、任意のポートに対する複数のフローの出入りを可能にし得る。明示的複製パスは、「複製」種類を有し、複数の出力フローを有する任意のポートの下流の他のプロパティを有しないノードを生成し、フローを当該ノードに移動させる。その後のパターンマッチングパスは、これらの一部をマルチ出力に再フォーマットし得る。

【0122】

いくつかの実施例では、データフローグラフのアルゴリズムは、アंकロスフローパスを含む。アंकロスフローパスは、ギャザーポートへとフローを並べ替え、結合時に入力ポートの番号を付け直すことによってグラフを平坦化しようと試みる。多くのグラフが平面ではないため、これは、ヒューリスティックアルゴリズムである。

【0123】

データセットカタログとのデータフローグラフの統合：

データソースカタログは、データソースに関するメタデータ、とりわけデータへのアクセスに使用されるURL、データのレコード形式、存在する場合は並べ替えキー及び分割キーなどを記憶する。データフローグラフ17は、クエリインスタンスのカタログ若しくはCatalog Interchange Format (CIF) ファイルにエクスポートされたカタログを使用すること、又はデータソース情報をグラフに直接埋め込むことができる。

【0124】

例えば、システムが可変ユーティリティをカタログファイルに割り当てる場合、グラフは、当該カタログにおいて名前データソースを参照することができる。データソースを導入したため、そのプロパティは、データフローグラフの他の場所で参照することができる。ソースパラメータは、データフローグラフ17を処理する後の段階において解決することができ、それにより、特定のパラメータ値のソースが明確なままになる。これは、パラメータが大きい場合に特に有用である。

【0125】

データフローグラフの拡張：

データフローグラフは、値指向処理及びグラフセマンティクスのために拡張することができる。いくつかの実施例では、データフローグラフは、とりわけ、アスペクト、式値、内蔵パス、ピア、ギャザー、マージ、フローの統合、及びアスペクトの伝播を使用することによって拡張することができる。

【0126】

アスペクトは、グラフのフローに沿って伝播するいくつかの情報又はメタ情報である。主な例は、値である。値は、グラフ内の1つのノードで計算され、更に下流の任意のノードで使用することができる。他のアスペクトとしては、とりわけ、フローに沿って存在する行のセット（行セット）、それらの行の順序（順序）、及び並列データのための分割キー（パーティション）が挙げられる。ノードのレイアウトも同じ機構によって伝搬されるため、レイアウトは別のアスペクトとして処理され得る。

10

20

30

40

50

【0127】

新しいアスペクトは、（任意選択の）「新しい」キーワードを用いて導入される。値は、通常、後に参照され得るように、IDを与えられる。他のアスペクトは、シングルトンであり、最新のアスペクトのみが参照可能であるため、IDなしで参照され得る。アスペクトは、明示的に削除されるまで、又は（非値の場合に）置換されるまで導入されるノード又はポートの下流のノードにおける参照に利用可能である。

【0128】

アスペクトは、ノードの効果をモデル化するために使用され、その結果、システムは、効果を有するノード、及び並べ替え可能であり得るノードを示すことができる。値は、このように使用され、また、ポート上のメタデータを、変換機能の生成時に決定する。

10

【0129】

新しい式値は、DML式から構築され得る。式は一定であり得るか、又はIDによって他の値を参照することができる。式値はまた、正に出力に適した型のみで値を構築するために使用される。

【0130】

いくつかの実施例では、内蔵パスを使用して、共通アスペクトを組み合わせ、組み合わせを折り畳むことができる。例えば、値が1回計算され、依然として有効である場合、同じ式を用いた新しい値は冗長であり、代わりに既存の値が使用され得る。

【0131】

いくつかの実施例では、空のノードを削除するために、内蔵パスが使用され得る。例えば、ノードがデータソースに書き込むか、新しい値若しくは他のアスペクトを導入するか、又は「del」で値又は他のアスペクトの寿命を終了させる場合、ノードは「副作用」を有する。副作用を有しないノードは、空とみなされ、パス「remove-empty-nodes」は、空のノードを削除する。一部のノードは除外され得る。例えば、「keep-alive」フラグセットを有する任意のノード、又はグラフレベルのポートに結合されたポートを有する任意のノード、又は単一の統一ポートに対する複数の入力フローを有する任意のノードである。

20

【0132】

（自己結合でのように）分岐し、次いで再収束するパスを含むグラフでは、同一の値が複数のパスに沿ってノードを到達することが可能である。結合において、異なる経路に沿った値は、異なる値になる。これは、値が到着したポートを示す「ビア」値を導入することによって行われる。ビア値は、参照する値の寿命を終了させ、いずれの下流参照も、ビア値を使用する。

30

【0133】

いくつかの実施例では、曖昧さがある場合にのみビア値が必要であり、ツリー形状を有するグラフにはビア値は不要であるべきである。重複排除パスは、再収束フローを導入するため、ビア値を導入する。

【0134】

複数のフローが単一の入力ポートに到着すると、複数の行セットが単一の行セットに統合され得る。入力フローの値は、互いに一致して、出力用に統一形式を生成する。値が全フローに沿って入力ポートに到着する場合、明示的な統一は不要であり、値は、入力ポートの下流に単一の値として見える。値がフローの一部のみに沿って入力ポートに到着する場合、入力ポートの下流に見ることはできない。

40

【0135】

新しいアスペクト（特定の値）は、ノード又はポートにおいて導入され得る。次いで、このアスペクトは、伝播ルールに従って、当該ポートの下流に見ることができる。一般に、アスペクトはフローに沿って、結合を超えてグラフレベルのポートに伝播する。出力ポートにおいて導入されたアスペクトは、当該出力ポートからフローの外に伝搬する。ノードにおいて導入されたアスペクトは、当該ノードの出力ポートの全てに、またそこから伝播する。入力ポートにおいて導入されたアスペクトは、全体としてノードに伝搬し、そこ

50

からその出力ポートに、またそれらのフローを下って伝搬する。

【0136】

「統一」について上述したように、複数の入力フローを有する入力ポートは、全ての入力フローに値が現れる場合、値の伝播を可能にする。更に、エラー及びログポートは、上流からデータを伝搬しないように構成可能であり得、これらは「`pass through none`」とタグ付けされる。他のポートは、単一の入力ポートからのみ伝播し、これらは、当該ポートの名前でタグ付けされる。

【0137】

名前、メタデータ、及び変換：

データフローグラフが最適化されて最適化されたデータフローグラフが提供されると、当該グラフは変換されたデータフローグラフ（例えば、変換されたデータフローグラフ19）に変換される。そうするために、メタデータ及びデータフローグラフに記載されている値を計算する変換を識別することができる。

【0138】

値は、導入時に名前を与えられ得る。かかる名前は、最終値名を割り当てるために使用するヒントではあるが、必須ではない。同一の所与の名前を有する2つの値がフローを超えて有効である場合、システムは、値のうちの1つに対して異なる名前を選択することができる。値名はロックされ得、その値がレコード形式で現れるときに、値が当該名を有することを示す。

【0139】

値は、構造化コメントを有し得る。構造化コメントを有する値がレコード形式でフィールドとして含まれる場合、値の構造化コメントが当該フィールドに適用される。値に直接構造化されたコメントがない場合、値は構造化コメントが存在するレコードからフィールドを抽出することによって計算され、当該構造化コメントは順方向に伝播される。

【0140】

内蔵パス「`assign-names`」は、フロー全体において有効である全ての値が名前を有し、これらの名前が、フロー全体において有効である値のセットは、当該フローにとってレコード形式で必要とされるフィールドを決定する、値の名前は、当該レコード形式のフィールド名を決定するために使用される、同一フローにわたって機能する2つの値は、異なる名前を有する、単一ポート（ギャザー）に入る複数のフローでのレコード形式は同一である、ポートにおいて単一値に統一される特定の値について、それらの値は同一名を有する、ロックされた名前を有する値は、それらの名前を保持する、など特定のルールに従うことを確実にする。

【0141】

いくつかの実施例では、内蔵の「`assign-names`」パスは、ロックされた名前があらゆる解決を阻止する場合、エラーを生成し得る。そうでない場合、名前は、潜在的に、所望の名前を有する新しい値を導入する新しいノードを追加することによって割り当てられる。このパスは、順序が比較的後ろの方であるため、それらのフローにわたって有効であるフローは少なく、幅は狭い。

【0142】

内蔵パス「`set-metadata`」は、全ての値が名前を有すると仮定し、グラフ内の全ポートに「メタデータ」パラメータを割り当てる。メタデータの類似性を強化する試みが行われる。ポート上のメタデータが入力メタデータと同じ構造及びフィールド名を有する場合、入力メタデータはそのまま再利用される。唯一の相違点が、末端の改行フィールドのドロップである場合、当該フィールドは、出力に保存される。メタデータが同一ではないが、入力からの一部のフィールド名が出力に現れる場合、それらのフィールドは最初に、かつ元の順序で現れる。メタデータがデータフローグラフに存在する任意のデータソースのレコード形式について厳密に構造的に一致する場合、メタデータのレコード形式は、当該ソースに厳密に一致するメタデータ文字列又はロケータに設定される。内蔵パスの「`set-metadata`」パスは、統一して、ギャザーポートの上流のレコード

形式を同一にする他のノードを導入し得る。

【0143】

別の内蔵パスは、全てのポートがメタデータを有し、各ノードの値から変換、パッケージ、及び式値パラメータを構築すると仮定する。

【0144】

PROTOファイル：

データフローグラフエンジの言語は、2つの部分を有する。すなわち、許可されるノード種類及びノードの編集方法を記述するプレゼンテーション層、並びに低レベルノードに関して各ノードの実施を記述するコンパイル層である。プロトタイプファイルは、言語のプレゼンテーション層を記述する。単一のプロトタイプファイルは、複数のプロトタイプ区分を含むことができる。

【0145】

プロトタイプは、統計関数を呼び出すことによって統計動作を実行するノードなどノードの種類の命名のように単純であり得る。

【0146】

ノード記述子は、例えば、表示名、ノードの説明（例えば、「データセットからの読み取り」）、ノードが属するカテゴリ（例えば、「データセット」）及びノードの形状を指定することによって、ノードの表示方法及びオーガナイザでの表示位置をエディタに指示する。

【0147】

ノードは、入力ポート及び出力ポートを有し得る。プロトタイプでポートに言及することにより、ポートをエディタに表示させることができる。入力及び出力ポートのそれぞれが命名され得る。ノードと同様に、ポートは、結合などポートの様々な機能を記述する記述子を有し得る。

【0148】

本明細書に記載の主題及び動作の実施は、デジタル電子回路、本明細書に開示された構造及びそれらの構造的均等物を含むコンピュータソフトウェア、ファームウェア、若しくはハードウェア、又はそれらの1つ以上の組み合わせで実施されることができる。本明細書に記載の主題の実施例は、1つ以上のコンピュータプログラム（データ処理プログラムとも呼ばれる）、すなわち、データ処理装置によって実行するために、又はデータ処理装置の動作を制御するためにコンピュータ記憶媒体で符号化されたコンピュータプログラム命令の1つ以上のモジュールとして実施されることができる。コンピュータ記憶媒体は、コンピュータ可読記憶デバイス、コンピュータ可読記憶基板、ランダムアクセス若しくはシリアルアクセスメモリアレイ若しくはデバイス、又はそれらの1つ以上の組み合わせであり得るか、又はそれらに含まれ得る。コンピュータ記憶媒体はまた、1つ以上の別個の物理的コンポーネント又は媒体（例えば、複数のCD、ディスク、又は他の記憶デバイス）であってもよく、又はそれらに含まれてもよい。主題は、非一時的コンピュータ記憶媒体に記憶されたコンピュータプログラム命令で実施され得る。

【0149】

本明細書に記載の動作は、1つ以上のコンピュータ可読記憶デバイスに記憶されたデータ、又は他のソースから受信したデータに対してデータ処理装置によって実行される動作として実施されることができる。

【0150】

「データ処理装置」という用語は、例として、プログラマブルプロセッサ、コンピュータ、システムオンチップ、又は上記の複数のもの、又は組み合わせを含む、データを処理するためのあらゆる種類の装置、デバイス、及び機械を包含する。装置は、専用ロジック回路（例えばFPGA（フィールドプログラマブルゲートアレイ）又はASIC（特定用途向け集積回路）など）を含むことができる。装置はまた、ハードウェアに加えて、問題のコンピュータプログラムに実行環境を提供するコード（例えば、プロセッサファームウェア、プロトコルスタック、データベース管理システム、オペレーティングシステム、ク

10

20

30

40

50

ロスプラットフォームランタイム環境、仮想マシン、又はそれらの1つ以上の組み合わせを構成するコード)を含むことができる。装置及び実行環境は、ウェブサービス、分散コンピューティング、及びグリッドコンピューティングインフラストラクチャなどの様々な異なるコンピューティングモデルインフラストラクチャを実現することができる。

【0151】

コンピュータプログラム(プログラム、ソフトウェア、ソフトウェアアプリケーション、スクリプト、又はコードとしても知られる)は、コンパイルされた又は解釈された言語、宣言型言語、手続き型言語など任意の形態のプログラミング言語で書き込まれ得、スタンダードプログラムとして、又はモジュール、コンポーネント、サブルーチン、若しくはコンピューティング環境での使用に好適な他のユニットとしてなど任意の形態で展開され得る。コンピュータプログラムは、ファイルシステム内のファイルに対応してもよいが、対応しなくてもよい。プログラムは、他のプログラム又はデータ(例えば、マークアップ言語文書に記憶された1つ以上のスクリプト)を保持するファイルの一部、問題のプログラム専用の単一のファイル、又は複数の協調ファイル(例えば、1つ以上のモジュール、サブプログラム、又はコードの一部を記憶するファイル)に記憶することができる。コンピュータプログラムは、1つのコンピュータ上で、又は1つのサイトに配置された、若しくは複数のサイトに分散され、通信ネットワークによって相互接続された複数のコンピュータ上で実行するように展開され得る。

【0152】

本明細書に記載のプロセス及びロジックフローは、1つ以上のコンピュータプログラムを実行する1つ以上のプログラマブルプロセッサによって実行され、入力データに基づいて動作して出力を生成することによって動作を実行することができる。プロセス及びロジックフローはまた、専用ロジック回路(例えばFPGA(フィールドプログラマブルゲートアレイ)又はASIC(特定用途向け集積回路))によって実行することができ、装置はまた、専用ロジック回路として実施されることができる。

【0153】

コンピュータプログラムの実行に適したプロセッサは、例として、汎用マイクロプロセッサ及び専用マイクロプロセッサの両方、並びに任意の種類のデジタルコンピュータの任意の1つ以上のプロセッサが挙げられる。概して、プロセッサは、リードオンリーメモリ若しくはランダムアクセスメモリ又はその両方から命令及びデータを受信するであろう。コンピュータの必須要素は、命令に従って動作を実行するためのプロセッサ、並びに命令及びデータを記憶するための1つ以上のメモリデバイスである。一般に、コンピュータはまた、データを記憶するための1つ以上の大容量記憶デバイス(例えば、磁気、光磁気ディスク、又は光ディスク)を含むか、それらからデータを受信するか、それらにデータを転送するか、それらの両方を行うように動作可能に連結されるが、コンピュータがかかるデバイスを有する必要はない。更に、コンピュータは、別のデバイス(例えば、携帯電話、携帯情報端末(personal digital assistant、PDA)、モバイルオーディオ若しくはビデオプレーヤ、ゲーム機、全地球測位システム(Global Positioning System、GPS)受信機、又は携帯型記憶デバイス(例えば、ユニバーサルシリアルバス(universal serial bus、USB)フラッシュドライブ))に埋め込まれ得る。コンピュータプログラム命令及びデータを記憶するのに適したデバイスは、例として、半導体メモリデバイス(例えば、EPROM、EEPROM、フラッシュメモリデバイス)、磁気ディスク(例えば、内蔵ハードディスク、又はリムーバブルディスク)、光磁気ディスク、並びにCD ROM及びDVD ROMディスクなど、あらゆる形態の不揮発性メモリ、媒体及びメモリデバイスを含む。プロセッサ及びメモリは、専用ロジック回路によって補完され得るか、又は専用ロジック回路に組み込まれ得る。

【0154】

本明細書に記載の主題の実施例は、(例えば、データサーバとして)バックエンドコンポーネントを含むか、又はミドルウェアコンポーネント(例えば、アプリケーションサーバ)を含むか、又はフロントエンドコンポーネント(例えば、ユーザが本明細書に記載の

10

20

30

40

50

主題の実施例と対話することができる、グラフィカルユーザインターフェース又はWebブラウザを有するユーザコンピュータ)を含むか、又は1つ以上のかかるバックエンド、ミドルウェア、若しくはフロントエンドコンポーネントの任意の組み合わせを含むコンピューティングシステムで実施することができる。システムのコンポーネントは、デジタルデータ通信(例えば、通信ネットワーク)の任意の形態又は媒体によって相互接続することができる。通信ネットワークの例としては、ローカルエリアネットワーク(LAN)及びワイドエリアネットワーク(WAN)、インターネットワーク(例えば、インターネット)、並びにピアツーピアネットワーク(例えば、アドホックピアツーピアネットワーク)が挙げられる。

【0155】

コンピューティングシステムは、ユーザ及びサーバを含み得る。ユーザ及びサーバは、一般に、互いに遠隔であり、通常、通信ネットワークを介して対話する。ユーザとサーバとの関係は、それぞれのコンピュータ上で実行され、互いにユーザーサーバ関係を有するコンピュータプログラムによって生じる。いくつかの実施例では、サーバは、データ(例えば、HTMLページ)をユーザデバイスに送信する(例えば、データを表示し、ユーザデバイスと対話するユーザからユーザ入力を受信する目的で)。ユーザデバイスにおいて生成されたデータ(例えば、ユーザ対話の結果)は、サーバにおいてユーザデバイスから受信することができる。

【0156】

本明細書は多くの特定の実施例の詳細を含むが、これらはいずれかの実施例又は特許請求され得るものの範囲に対する限定として解釈されるべきではなく、特定の実施例に特有の特徴の説明として解釈されるべきである。別個の実施例の文脈で本明細書に記載されている特定の特征是、単一の実施例において組み合わせで実施することもできる。逆に、単一の実施態様の文脈で記載されている様々な特征是、複数の実施例において別々に、又は任意の適切な副組み合わせで実施することもできる。更に、特征是、特定の組み合わせで作用するように記載され、かつ更に当初はそのように特許請求され得るが、特許請求された組み合わせからの1つ以上の特征是、場合によってはその組み合わせから削除され得、特許請求された組み合わせは、部分組み合わせ又は部分組み合わせの変形例を対象とし得る。

【0157】

同様に、動作は特定の順序で図面に示されているが、望ましい結果を達成するためには、かかる動作が、示される特定の順序で、若しくは順番に実行されること、又は全ての図示された動作が実行されることを必要とすると理解されるべきではない。特定の状況では、マルチタスキング及び並列処理が有利であり得る。更に、上記の実施例における様々なシステムコンポーネントを分離させることは、全ての実施例でかかる分離を必要とするものとして理解されるべきではなく、記載のプログラコンポーネント及びシステムは、一般に単一のソフトウェア製品で一体化され得るか、又は複数のソフトウェア製品にパッケージされ得ることを理解されたい。

【0158】

他の実施例は、以下の「特許請求の範囲」の範囲内である。

10

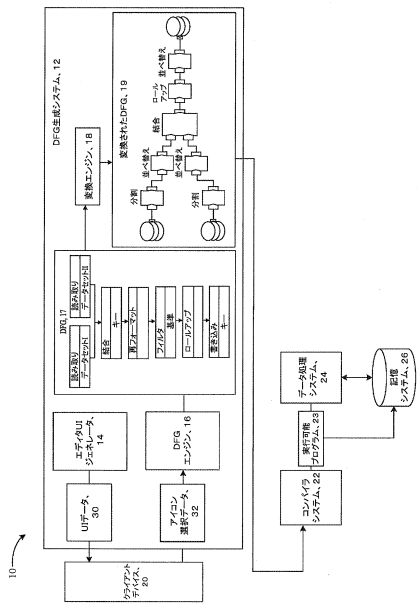
20

30

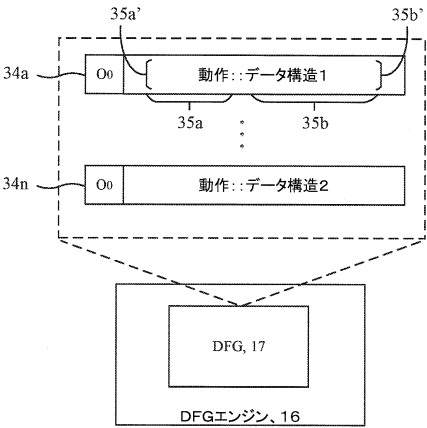
40

50

【図面】
【図 1】



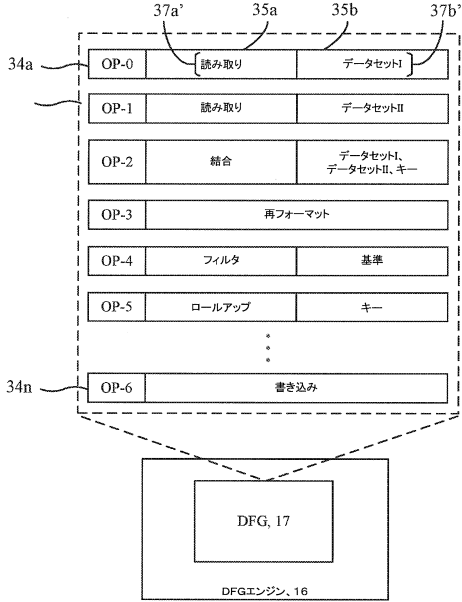
【図 2 A】



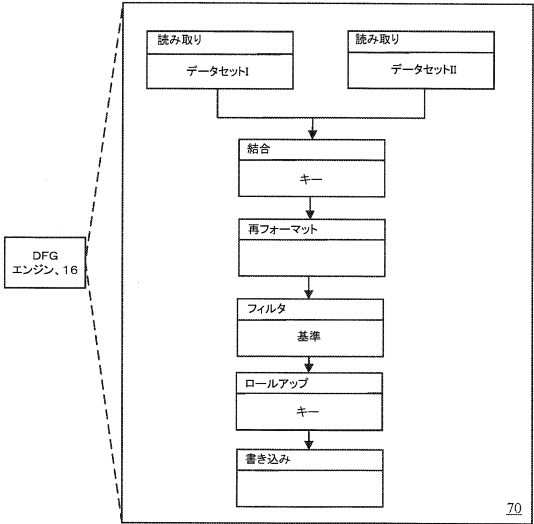
10

20

【図 2 B】



【図 3 A】

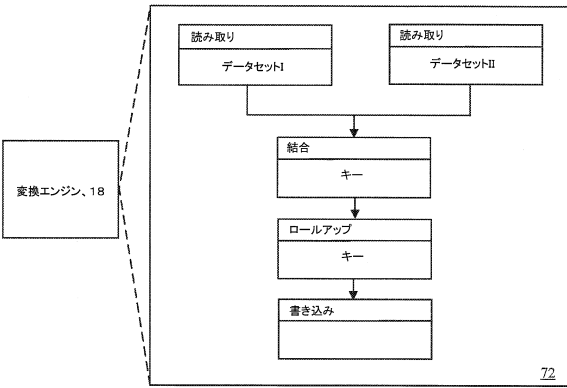


30

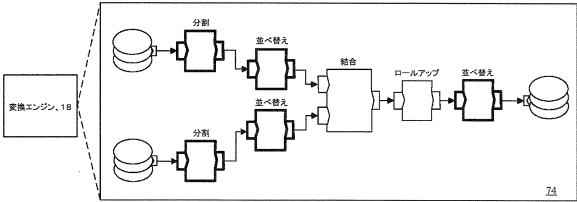
40

50

【図 3 B】

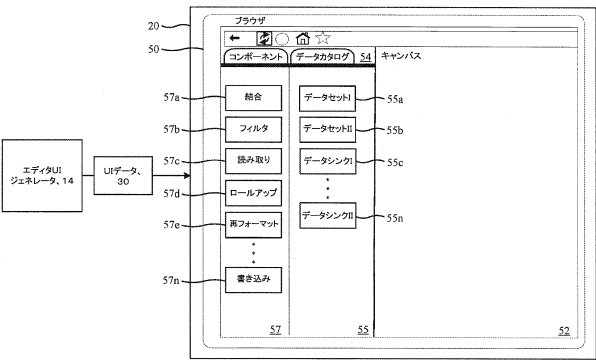


【図 3 C】

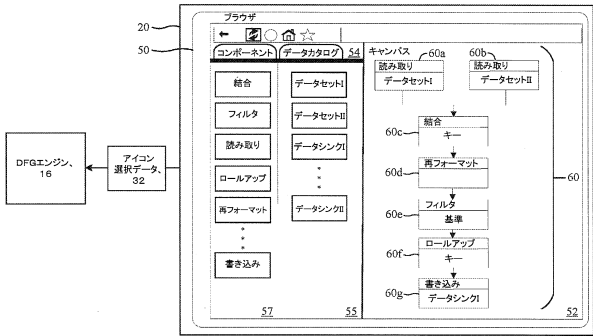


10

【図 4 A】



【図 4 B】



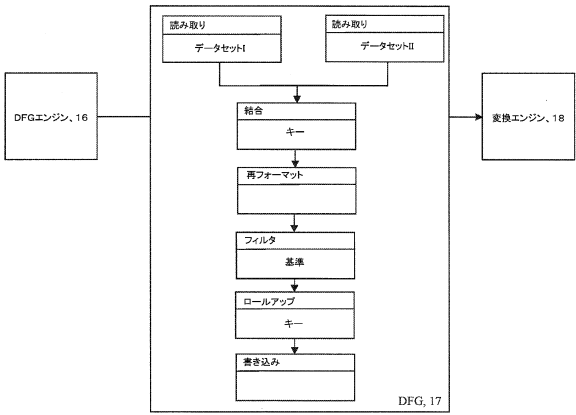
20

30

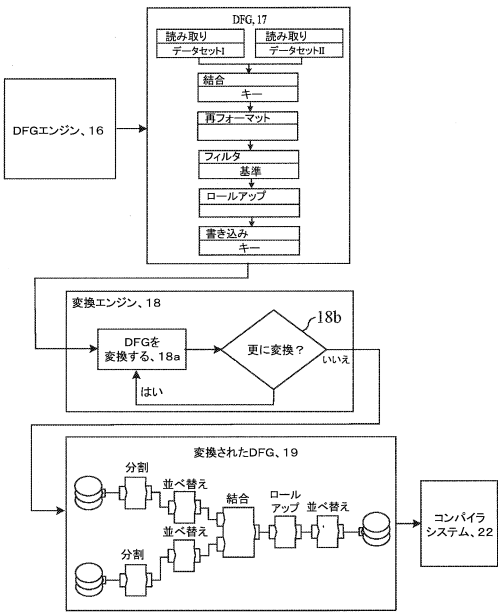
40

50

【図 4 C】



【図 4 D】

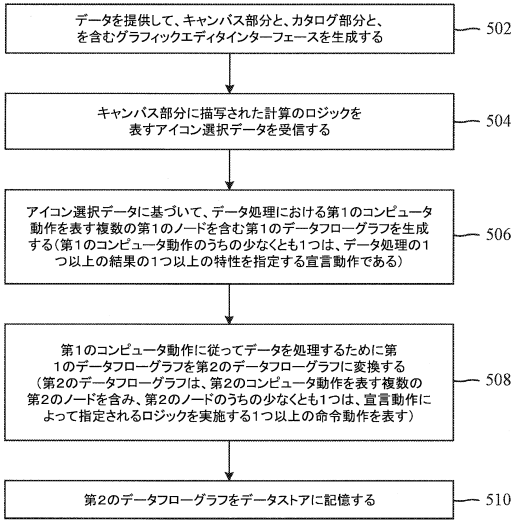


10

20

【図 5】

500



30

40

50

フロントページの続き

(72)発明者 ガース・ディッキー

 アメリカ合衆国・マサチューセッツ・０１７０２・フラミンガム・パーカー・ロード・１６１

審査官 児玉 崇晶

(56)参考文献 米国特許出願公開第２０１９／０３７０４０７（ＵＳ，Ａ１）

 特表２０１３－５４１７８４（ＪＰ，Ａ）

 米国特許出願公開第２０１５／０１０６８１８（ＵＳ，Ａ１）

(58)調査した分野 (Int.Cl., D B名)

 G 0 6 F 8 / 3 4