(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6123339号
(P6123339)
(45)発行日　平成29年5月10日(2017.5.10)　(24)登録日　平成29年4月14日(2017.4.14)

(51)Int.Cl.　F I
G06F 17/30 (2006.01)　G06F 17/30 419B
G06F 17/30 110C

請求項の数 15　外国語出願　(全 22 頁)

(21)出願番号　特願2013-29535(P2013-29535)
(22)出願日　平成25年2月18日(2013.2.18)
(65)公開番号　特開2013-175181(P2013-175181A)
(43)公開日　平成25年9月5日(2013.9.5)
審査請求日　平成27年10月7日(2015.10.7)
(31)優先権主張番号　12156707.7
(32)優先日　平成24年2月23日(2012.2.23)
(33)優先権主張国　欧州特許庁(EP)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100107766
弁理士　伊東　忠重
(74)代理人　100070150
弁理士　伊東　忠彦
(74)代理人　100146776
弁理士　山口　昭則
(72)発明者　カルヴァーリョ・ヌノ
イギリス国，ダブリュ５　４ディーダブリュ，ロンドン，ウィンミル　ロード，ピカリング　ハウス　８番

最終頁に続く

(54)【発明の名称】エンコードされたトリプルを格納するデータベース、装置及び方法

(57)【特許請求の範囲】
【請求項１】
トリプルとしてエンコードされたグラフデータのデータベースであって、前記データベースは、前記グラフデータにアクセスするアプリケーションにより用いられ、各トリプルは、主語、述語及び目的語の３つのトリプル要素を有し、各トリプルは該トリプルのデータに従って順序付けられたデータアイテムのセットの中のデータアイテム内に格納され、分散型ノードネットワークの複数のノードに渡って分散され、
各トリプルは、
データアイテム内で主語が述語及び目的語より前にある第１の構成、
データアイテム内で述語が主語及び目的語より前にある第２の構成、
データアイテム内で目的語が主語及び述語より前にある第３の構成、
のうちのそれぞれ異なる構成を有する２以上のデータアイテム内に格納され、
前記データアイテムの各々がマッピングされるノードは、前記データアイテム内で最初に現れるトリプル要素に依存し、更に前記順序付けられたセット内のデータアイテムの位置に依存し、前記マッピングを用いて、前記アプリケーションは、アクセスするグラフデータに対応するデータアイテムの範囲を検索するためのクエリを作成する、
データベース。
【請求項２】
各データアイテムは、対応するトリプルの主語、述語及び目的語を有するストリングオブジェクトを含む、請求項１に記載のデータベース。

【請求項３】
前記データアイテムは、前記ストリングオブジェクトのアルファベット順の比較に従って順序付けられる、請求項２に記載のデータベース。
【請求項４】
各前記データアイテムは、キー値体系内のキー値ペアのキーである、請求項１乃至３のいずれか一項に記載のデータベース。
【請求項５】
複数の前記キー値ペアの各々の値は、キーに格納されたトリプルに関連する追加情報を有する、請求項４に記載のデータベース。
【請求項６】
前記追加情報は、アプリケーションのアイデンティティに依存して前記データベースにアクセスする前記アプリケーションに利用可能にされるデータである、請求項５に記載のデータベース。
【請求項７】
前記追加情報は、リード及び／又はライトアクセスが前記キー値ペアに格納されたトリプルに対して行われるとき通知を受信するために登録されるアプリケーションのリストを含む、請求項５又は６に記載のデータベース。
【請求項８】
各キー値ペアの値は、アプリケーションに前記通知を登録及び／又は登録解除させるアプリケーションプログラミングインタフェース（ＡＰＩ）を含む、請求項７に記載のデータベース。
【請求項９】
各キー値ペアの値は、前記追加情報へのアクセスを管理するアプリケーションプログラミングインタフェース（ＡＰＩ）を含む、請求項５乃至８のいずれか一項に記載のデータベース。
【請求項１０】
前記追加情報は、値が格納されているノードにより実行されるためのソフトウェアコードを含む、請求項５乃至９のいずれか一項に記載のデータベース。
【請求項１１】
各前記データアイテムは、順序付けたハッシュテーブルを用いてノードにマッピングされる、請求項１乃至１０のいずれか一項に記載のデータベース。
【請求項１２】
前記順序付けたハッシュテーブルは、コンシステントハッシュ関数を用いる、請求項１１に記載のデータベース。
【請求項１３】
前記データベースは１又は複数の他のデータベースによりアクセス可能であり、
前記追加情報は、
前記キー値ペアに格納されたトリプルにより表される情報を含む１又は複数の他のデータベースの間のデータベースのリスト、
リストされた各データベースに対して、前記キー値ペアに格納されたトリプルに含まれる情報が、該リストされたデータベースに対して動作するアプリケーションにより更新されるようにするコネクタ、
を含む、請求項５乃至１０のいずれか一項に記載のデータベース。
【請求項１４】
分散型ノードネットワークのノードとして動作するよう構成されるコンピューティング装置であって、前記コンピューティング装置は、トリプルとしてエンコードされたグラフデータのデータベースのサブセットを格納し、前記コンピューティング装置は、前記グラフデータにアクセスするアプリケーションにより用いられ、各トリプルは、主語、述語及び目的語の３つのトリプル要素を有し、各トリプルは該トリプルのデータに従って順序付けられたデータアイテムのセットの中のデータアイテム内に格納され、前記分散型ノード

ネットワークの複数のノードに渡って分散され、
各トリプルは、
データアイテム内で主語が述語及び目的語より前にある第１の構成、
データアイテム内で述語が主語及び目的語より前にある第２の構成、
データアイテム内で目的語が主語及び述語より前にある第３の構成、
のうちのそれぞれ異なる構成を有する２以上のデータアイテム内に格納され、
前記データアイテムの各々がマッピングされるノードは、前記データアイテム内で最初に現れるトリプル要素に依存し、更に前記順序付けられたセット内のデータアイテムの位置に依存し、前記マッピングを用いて、前記アプリケーションは、アクセスするグラフデータに対応するデータアイテムの範囲を検索するためのクエリを作成する、
コンピューティング装置。
【請求項１５】
コンピューティング装置により実行されると、前記コンピューティング装置に請求項１４に記載のコンピューティング装置として動作させる、コンピュータプログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、データ記憶の分野に関する。特に、本発明の実施形態は、分散記憶環境でグラフデータを記述するトリプルの記憶に関連する。
【背景技術】
【０００２】
関係型データベースは、データを行と列で格納する。行及び列は、データを格納する前に定める必要のあるテーブルを構成する。テーブルの定義及びこれらのテーブルに含まれるデータ間の関係は、スキーマと称される。関係型データベースは、固定スキーマを用いる。グラフデータベースは、データをノード及びアークの形式で格納することにより、関係型データベースの重要な拡張を表す。ここで、ノードはエンティティ又はインスタンスを表し、アークは任意の２つのノード間の特定種類の関係を表す。無向グラフでは、ノードＡからノードＢへのアークは、ノードＢからノードＡへのアークと同じであると考えられる。有向グラフでは、２つの方向は別のアークとして扱われる。
【０００３】
グラフデータベースは、概して２つの主な種類に分類できる広範な種類の異なるアプリケーションで用いられる。第１の種類は、知的意思決定支援及び自己学習のようなクラス記述子の大規模な集合体（「知識ベースアプリケーション」と称される）を有する複雑な知識ベースシステムを有する。第２の種類は、社会的データ及びビジネスインテリジェンスのようなトランザクションデータに対するグラフ検索の実行を含むアプリケーション（「トランザクションデータアプリケーション」と称される）を有する。多くのアプリケーションは、両方の種類を表し得る。しかしながら、大部分のアプリケーションは、主に知識ベース又はトランザクションデータアプリケーションのいずれかで特徴付けられ得る。グラフデータベースは、種々の分野の膨大な構造化又は非構造化データを格納できる大規模な「意味ネットワーク」を維持するために用いることができる。意味ネットワークは、知識表現の形式として用いられ、コンセプトを表すノード及びコンセプト間の意味関係を表すアークを有する有向グラフである。
【０００４】
幾つかの種類のグラフ表現がある。グラフデータは、多次元アレイとして又は他のシンボルにリンク付けされたシンボルとしてメモリに格納されても良い。別の形式のグラフ表現は、各々指定された種類のオブジェクトの有限シーケンス又は順序付きリストである「タプル」の使用である。ｎ個のオブジェクトを含むタプルは、「ｎタプル」として知られる。ここで、ｎは零より大きい任意の非負整数である。長さ２のタプル（２タプル）は、通常、ペアと呼ばれる。３タプルはトリプルと呼ばれ、４タプルはクワドラプルと呼ばれ

、以降同様である。
【０００５】
ＲＤＦ（Resource Description Framework）は、概念記述又は意味ネットワークの標準である情報のモデル化のための一般的方法である。今日利用可能なＲＤＦデータの量は、増大しており、既に単一のサーバに格納することが不可能である。膨大な量のデータを格納し検索可能にするために、データは複数のサーバに保持されなければならない。データの追加、削除及び検索は、分散システムのために特注されたアルゴリズム及びデータ構造を用いて協調的方法で行われなければならない。コンピュータ的に効率的にデータの検索、保守及び操作を可能にするような方法でグラフデータを格納することが望ましい。
【発明の概要】
【発明が解決しようとする課題】
【０００６】
本発明は、エンコードされたトリプルを格納するデータベース、装置及び方法を提供する。
【課題を解決するための手段】
【０００７】
本発明の実施形態は、トリプルとしてエンコードされたグラフデータのデータベースであって、各トリプルは、主語、述語及び目的語を有し、各トリプルは該トリプルのデータに従って順序付けられたデータアイテムのセットの中のデータアイテム内に格納され、分散型ノードネットワークの複数のノードに渡って分散され、前記データアイテムの各々がマッピングされるノードは、前記順序付けられたセット内のデータアイテムの位置に依存し、各トリプルは、データアイテム内で主語が述語及び目的語より前にある第１の構成、データアイテム内で述語が主語及び目的語より前にある第２の構成、データアイテム内で目的語が主語及び述語より前にある第３の構成、のうちのそれぞれ異なる構成を有する２以上のデータアイテム内に格納される、データベースを提供する。
【０００８】
このようなトリプルを分散キー値ストア（key-value store：KVS）のような従来の（非ソート）システムに格納するためには、各トリプルはキーに関連付けられなければならない。トリプルを定まった（非ソート）ＫＶＳに格納することは、システムが格納されたデータに対して長い詳細な検討を実行する必要がある場合に、コンピュータ的に集中的なプロシジャをもたらし得る。非ソートＫＶＳ内の長く詳細な検討を実行することは、ＫＶＳに格納された各々１つのトリプルに対してget（キー）演算を実行することにより実施される。大文字Ｏ表記を用いると、この演算はＯ（Ｋ）である。ここで、Ｋは、クライアントアプリケーションが検索に必要なキーの数（長く詳細な検討を行うときは大きな数になる可能性がある）である。get()演算の度に、システムは以下を行う必要がある。（１）ＫＶＳの一部として動作しているプロセスＰの１つへ要求を送信する。（２）Ｐは、データを保持するネットワークノードを発見する関数を実行する。この関数は例えばＱを返しても良い。（３）Ｐは要求をＱへ転送する。（４）Ｑはクライアントアプリケーションに直接応答でき、又はＰを媒介として用いることができる。（５）上述のタスクはＫ回繰り返される。
【０００９】
このプロシジャは、通常３通信ステップで、データがＰ内にＫ回ある場合に時には２通信ステップだけで実行される。このプロシジャは、特に、個々のキーを検索する度にデータストアを詳細に検索するために用いられるコンピュータリソースの使用の観点から集約的である。
【００１０】
本発明の実施形態では、（ＲＤＦトリプルのような）各トリプルは、データアイテムとして、例えば単純なストリングオブジェクトとして格納され、グラフＧ内で一意である。よって、Ｇに関する全ての情報は、データアイテム内に保持される。トリプルは、（ピアツーピアネットワーク環境のような）分散型ネットワーク環境のネットワークノード内の

プロセスを通じて分散された順序付けられたデータセット内のデータアイテムとして格納される（データアイテムにエンコードされる）。各ノードで動作するプロセスは、データアイテムの順序付けられたサブセットの記憶を実現する。プロセス及びプロセスを動作させるネットワークノードは、本願明細書では事実上同義的に表され、前者又は後者が排他的に意図されるときは文脈から明らかである。本発明の実施形態は、データに対して効率的な問い合わせプロシジャを行うことを可能にする。例えば、（発行されると）ｄｍｉｎとｄｍａｘとの間のデータアイテムの範囲を検索するプロシジャは、以下のステップを実行する。（１）クライアントアプリケーションがプロセスのうちの１つＡに要求を送信する。（２）Ａはマッピング関数を実行し、要求されたデータ範囲の１つのサブセットを含むプロセスのノードＩＤ（又はラベル）を得る（例示的な場合には、データアイテムは均等に分散され、各プロセスは多数のデータアイテムの格納を担う）。例として、クライアントアプリケーションにより要求されたキーがプロセスＡ及びＢに格納されている場合を検討する。（３）Ａは、要求されたデータアイテムの各サブセットを検索するサブ範囲要求を自身へ及びＢへブロードキャストする。（４）各プロセスは、要求されたデータアイテムと共にＡに応答する。（５）Ａは、（結果の順序を維持しながら）結果を集め、クライアントアプリケーションに応答する。

【００１１】

上述の例から、本発明の実施形態が更に効率的なデータの問い合わせを可能にすることが分かる。さらに、ステップは、従来技術の場合のように要求内のデータアイテム毎に繰り返される必要はない。

【００１２】

本発明の実施形態におけるグラフデータは有向グラフデータなので、第１のグラフノードから第２のグラフノードへのアークは、第２のグラフノードから第１のグラフノードへのアークと同じであるとは考えられない。意味ネットワークは、知識又は情報の表現として形成され、エンティティ又はインスタンスのようなコンセプトを表すグラフノード、及びコンセプト間の意味関係を表すアークを有する。

【００１３】

本発明の実施形態では、グラフデータは、トリプルとしてエンコードされる。トリプルは、それぞれ特定の種類である３つのオブジェクトの無限シーケンス又は順序付きリストである。

【００１４】

任意で、トリプルは、ＲＤＦ（Resource Description Framework）トリプルであっても良い。本願明細書を通じて、「ＲＤＦトリプル」への特定の参照が行われるとき、それはＲＤＦ標準に準拠するトリプルの例示的形式であることが理解されるべきである。さらに、「トリプル」への参照は、問題のトリプルがＲＤＦトリプルである可能性を有する。同様に、本願明細書のいずれかの箇所で議論されるＲＤＦプロセッサは、ＡＰＩラッパと格納されたデータアイテムとの間の相互作用のために用いられるプロセッサの例である。

【００１５】

ＲＤＦ（Resource Description Framework）は、概念記述又は意味ネットワークの標準である情報のモデル化のための一般的方法である。意味ネットワークにおける情報のモデル化の標準化は、共通の意味ネットワークで動作するアプリケーション間の相互接続性を可能にする。ＲＤＦは、ＲＤＦスキーマ（ＲＤＦＳ）をＲＤＦ内の語彙を記述するための言語として提供することにより、一義的な形式意味論と共に語彙を保持する。

【００１６】

トリプルは、グラフデータを複数の主語－述語－目的語の表現として特徴付けることにより、グラフデータのエンコードを提供する。この文脈では、主語及び述語は、グラフデータのグラフノードであり、オブジェクト、インスタンス又はコンセプトのようなエンティティであり、述語は、主語と目的語の間の関係の表現である。述語は、目的語への特定の種類のリンクを提供することにより、主語に関する何かを断言する。例えば、主語は、（例えば、ＵＲＩを介して）ウェブリソースを示しても良く、述語はリソースの個々の特

性、特徴又は状況を示し、目的語は、該特性、特徴又は状況のインスタンスを示す。言い換えると、トリプルステートメントの集合は、元来、方向性グラフデータを表す。ＲＤＦ標準は、このようなトリプルの形式化された構造を提供する。

【００１７】

分散型ノードネットワークは、互いに通信する１より多い異なる記憶ユニットを有し得る。例示的な通信パラダイムはピアツーピア（Ｐ２Ｐ）である。したがって、分散型ノードネットワークはピアツーピアノードネットワークであっても良い。Ｐ２Ｐは、タスク又は負荷をピア間に区分する分散アーキテクチャである。ピア（個々のノード又はプロセス）は、等価な特権を有し、アプリケーション内で等しい力を有する参加者である。各ピアは、処理能力、ディスクストレージ又はネットワーク帯域幅のようなそれ自体のリソースの一部を、サーバ又は安定したホストによる集中的強調の必要無しに、他のネットワーク参加者に直接利用可能にするよう構成される。ピアは、リソースの供給者及び消費者の両方であると考えられ、サーバが供給しクライアントが消費するという従来のクライアントーサーバモデルとは対照的である。有利なことに、Ｐ２Ｐは、対数的な通信コストでメッセージを交換するノードの大規模なグループを維持できる。

【００１８】

実施形態では、トリプルの目的語が複雑な例では、順序付けられたデータに格納される目的語は、該目的語のために生成されるＵＵＩＤ（universal unique ID）であっても良い。この選択肢は、オブジェクトが大きく、そのサイズが範囲クエリプロシジャの効率を低下させてしまう実装シナリオで用いられる実施形態で有用である。

【００１９】

本発明の実施形態では、トリプルは、順序付けられたデータアイテムに格納されるので、データアイテムを返すクエリは、トリプルの表現が検索されるのを可能にする。

【００２０】

本発明の実施形態では、データアイテムは、トリプルの要素を表すストリングオブジェクトの辞書順により順序付けられても良い。ストリングオブジェクトは、データアイテムであるか又はデータアイテムに含まれている。例えば、辞書順はアルファベット順であっても良い。本発明の実施形態の実装では、データアイテムの主語、述語又は目的語（又は目的語のＵＵＩＤ）のいずれかを固定することによりデータアイテムのセットをクエリすることが望ましい。このようなくえりの結果を効率的に返すために、トリプルの２以上の要素がデータアイテム内の最初の要素として格納されることが有利である。

【００２１】

順序付けられているデータアイテムは、データアイテム間の比較を行うこと及び範囲クエリを実行することを可能にする。Ｋ１とＫ２（Ｋ１＜Ｋ２）の間の範囲クエリは、キーの特定の所定の順序メトリックに従って、Ｋ１より大きくＫ２より小さいデータアイテムの順序付けされたセットに含まれるデータアイテムを返す。勿論、データアイテムのセットは、分散型ノードネットワークのノードに渡って分割される（ここで、ノードは、サーバのようなリソース、又は該サーバで動作するプロセスである）。例示的な実施形態では、範囲クエリは、クエリをノードのうちの１つに送信することにより、アプリケーションの代わりにＲＤＦプロセッサのようなプロセッサにより開始される。ノードは、どの他のノードがクエリにより探し出されたデータアイテムを有するかを、データアイテムの論理表現をＫ１及びＫ２に適用し、サブ範囲クエリをこれらのノードに対して実行することにより計算するよう構成される。ノードは、次に、（データアイテムの順序を維持したまま）結果を集め、それらをクエリを行使したプロセッサに返すよう構成される。

【００２２】

任意で、トリプルの１又は複数の要素のうちの各々は（要素は、述語、目的語又は主語である）、ＵＲＩ（Uniform Resource Identifier）である。ＲＤＦ及び他のトリプルの形式は、識別するものの概念（つまり、オブジェクト、リソース又はインスタンス）を前提として、ＵＲＩのようなウェブ識別子を用い、それら識別される「もの」を簡易な特性及び特性値の観点で記述する。トリプルの観点では、そのトリプルのウェブリソースの

具体化において、主語はエンティティを記述するウェブリソースを特定するＵＲＩであっても良く、述語は特性の種類（例えば、色）を特定するＵＲＩであっても良く、目的語は問題のエンティティに起因する特性の種類の特定のインスタンスを指定するＵＲＩであっても良い。ＵＲＩの使用は、トリプルに、個々の特性及び値と同様に、リソースを表すノード及びアークのグラフのようなリソースに関する簡易なステートメントを表すことを可能にする。ＲＤＦグラフは、SPARQLプロトコル及びＲＤＦクエリ言語（SPARQL）を用いて問い合わせることができる。SPARQLは、World Wide Web ConsortiumのRDF Data Access Working Group (DAWG)により標準化され、主要なセマンティックウェブ技術と考えられている。SPARQLは、クエリがトリプルのパターン、連結、分離、任意のパターンを有することを許容する。

【００２３】

任意で、各データアイテムは、対応するトリプルの主語、述語及び目的語を有するストリングオブジェクトを含んでも良い。

【００２４】

有利なことに、ストリングオブジェクトは、一般に読み取り可能であり、その比較及び存在する他の処理要求では確立されたルーチンである。データベース自体及びデータベースにアクセスするアプリケーションは、ストリングオブジェクトを処理する確立したルーチンを有しても良い。さらに、ストリングオブジェクトは、検索及び比較（オーダ）するのが速い。

【００２５】

データアイテムは、ストリングオブジェクトのアルファベット順の比較に従って順序付けられても良い。

【００２６】

有利なことに、検索、範囲、他の比較関数のようなデータベース関数が利用可能である。データベース関数は、ストリングデータオブジェクトのアルファベットの内容を比較するために、コンピュータ的に効率的な観点で最適化される。したがって、このようにデータアイテムを順序付けする実施形態は、コンピュータ的効率の観点で特に有効である。上述の実施形態では、前記データアイテムは、該データアイテムのストリングオブジェクトのアルファベット順に従って順序付けられても良い。データアイテムは、単にストリングオブジェクトであっても良く、又は他のオブジェクト若しくはデータを有しても良い。ストリングは、英数字シンボルのシーケンスである。

【００２７】

本願明細書で議論するデータアイテムは、独立した情報片であっても良い。しかしながら、本発明の実施形態は、各前記データアイテムがキー値体系（ＫＶＳ）内のキー値ペアのキーである実施形態を含む。

【００２８】

有利なことに、キー値体系のキーに完全なトリプルを含むキーの格納は、関数が、トリプルが見付かるかも知れない場所への単なるリンク又は識別子ではなく、完全なトリプルを返すキーのセットに対して実行されるのを可能にする。

【００２９】

キー値ペアのキーに格納されることは、キー値ペアのキーであるストリングオブジェクトとしてトリプルの要素を表すことを含む。

【００３０】

キー値システム（ＫＶＳ）又はキー値ストアは、複数の格納されたキー及び値である。各キーは、関連付けられた値を有し、論理関数又は論理木、例えばハッシュテーブル又はハッシュマップを介して該関連付けられた値にマッピングされる。ハッシュテーブル又はハッシュマップは、ハッシュ関数を用いて（値を特定する）キーをそれらの関連付けられた値にマッピングするデータ構造である。本発明の実施形態では、ハッシュ関数は、キーを、ノードのピアツーピアネットワークを形成する複数のノードのうちのノード（記憶リソース）の識別表示に変換するために用いられても良い。

【００３１】
実施形態は、複数の前記キー値ペアの各々の値は、キーに格納されたトリプルに関連する追加情報を有しても良い。
【００３２】
上述の技術では、トリプル全体、又は目的語のＵＵＩＤを有するトリプルの述語及び主語は、キー値ペアのキーに格納され、有利なことに、キー値ペアの値がデータベースの機能及び使い勝手を向上させるために用いられることを可能にする。値の特性又は詳細な内容は、特定の実施形態及び実装の詳細に依存する。値は、キーを介して参照されるコンテナであっても良い（コンテナは、文字列、ソフトウェア及び他のオブジェクトを一緒に格納するよう適応されたコンテナ（Container）と称されるオブジェクトのクラスを有する）。このようなコンテナの幾つかの例示的な内容を以下に説明する。
【００３３】
単純な例として、目的語のＵＵＩＤが目的語自体を表すストリングではなく、キーのストリングに含まれるキー値ペアでは、目的語自体を表すストリングは、コンテナに格納されても良い。
【００３４】
更なる例として、トリプルに関するメタデータは、値に格納される。この文脈におけるメタデータは、トリプルの中の情報アイテムに関する説明データである。キーはトリプルを格納し、値は該トリプルに関する情報を格納する。
【００３５】
任意で、上述の実施形態では、前記追加情報は、アプリケーションのアイデンティティに依存して前記データベースにアクセスする前記アプリケーションに利用可能にされるデータである。
【００３６】
有利なことに、このような実施形態は、データベース内のデータを読み出し及び／又は書き込むアプリケーションに依存する特徴を有効にすることにより、データベースの機能を拡張する。データは、アプリケーション依存データであり、ＫＶＳにおいて不可解オブジェクトとして見なされ取り扱われても良い。値コンテナオは、アプリケーション依存データを設定及び読み出すＡＰＩを有しても良い
任意的に、上述のＫＶＳ実装では、前記追加情報は、リード及び／又はライトアクセスが前記キー値ペアに格納されたＲＤＦトリプルに対して行われるとき通知を受信するために登録されるアプリケーションのリストを含む。さらに、追加情報は、コンテナを格納するノードによる実行のためにソフトウェアコードを有しても良い。例えば、ソフトウェアコードは、特定のイベントに応答して格納され呼び出されても良い。また、ソフトウェアコードの引数もコンテナに格納される。例えば、値は、ソフトウェアコードのブロックを更新関数として格納しても良い。関連付けられたトリプルが更新されるとき、更新関数が呼び出され、他のトリプルのリスト及びそれらの場所が引数である。他のトリプルのリストは、クライアントアプリケーションがトリプルを、特定のイベントと関連付けられたコンテナ内のリストに追加することにより、生成されても良い。
【００３７】
有利なことに、トリプルデータへの読み出し又は書き込みアクセスが行われるときを通知するアプリケーションは、格納されたデータを用いてデータベースとアプリケーションとの間の相互作用のレベルを向上させる。通知を受信するために登録されたアプリケーションのリストは、通知が管理され得るメカニズムを提供する。
【００３８】
各キー値ペアの値は、アプリケーションに前記通知を登録及び／又は登録解除させるアプリケーションプログラミングインタフェース（ＡＰＩ）を含んでも良い。また、通知を実行するソフトウェアコードは、上述のようにコンテナに格納されても良い。
【００３９】
有利なことに、通知を登録及び登録解除するアプリケーションの形式化されたメカニズ

ムを提供することは、特性が動的である可能性の高いアプリケーションの現在のプロファイル及び該アプリケーションの要件を反映するよう、通知システムを適応し変更することを可能にする。さらに、データベースは、データベース管理者又は他のユーザがＡＰＩ自体を更新又は変更できるように、構成されても良い。

【００４０】

任意的に、上述のＫＶＳ実装では、各キー値ペアの値は、前記追加情報へのアクセスを管理するアプリケーションプログラミングインタフェース（ＡＰＩ）を含む。

【００４１】

ＡＰＩは、ソフトウェアコンポーネントが互いに相互作用する仕様である。本例では、ＡＰＩは、特定の情報片の場所が決定できるようにコンテナ内のデータ構造の定義、データに関連する書き込みコマンド又は読み出しコマンドを発行する枠組み、及び追加情報内のデータ種類の定義を含んでも良い。

【００４２】

例示的な実施形態では、各前記データアイテムは、順序付けたハッシュテーブルを用いてノードにマッピングされる。

【００４３】

有利なことに、ハッシュテーブルは、データアイテムをノードにマッピングするコンピュータ的に効率的な方法を提供する。順序付けたハッシュテーブルは、ハッシュ関数を用いて、データアイテムの内容に基づきデータアイテムをノードに割り当てる。

【００４４】

キー値体系では、ハッシュテーブル又はハッシュマップは、ハッシュ関数を用いてキーとして知られる特定する値をそれらの関連付けられた値にマッピングするデータ構造である。したがって、ハッシュテーブルは、連想配列を実施すると言われている。ハッシュ関数は、対応する値が検索されるアレイ要素（記憶場所、スロット又はバケットとも称される）のインデックス（ハッシュ）にキーを変換するために用いられる。

【００４５】

任意的に、上述のデータベースでは、前記順序付けたハッシュテーブルは、コンシステントハッシュ関数を用いる。

【００４６】

一貫したハッシングは、関連付けられた値が格納されているスロットの数の変化に応答して再マッピングされる必要のあるキーの数を調整する。例えば、キー及びＫ個のキーに関連付けられた値がピアツーピアシステム内のｎ－１個の「ピア」の間に分散されているシステムを考える。新しいピアがシステムに参加した場合、Ｋ／ｎ個のキーのみが再マッピングされる必要がある。一貫したハッシングでは、新しいピアがシステムに追加されるとき、他のピアから格納されたキーのほぼ等しい分担を取り、ピアが削除されるとき、そのキーは残りのピアの間で分担される。

【００４７】

任意的に、上述のＫＶＳ実装では、前記データベースは１又は複数の他のデータベースによりアクセス可能であり、前記追加情報は、前記キー値ペアに格納されたトリプルにより表される情報を含む１又は複数の他のデータベースの間のデータベースのリスト、　ストされた各データベースに対して、前記キー値ペアに格納されたトリプルに含まれる情報が、該リスとされたデータベースに対して動作するアプリケーションにより更新されるようにするコネクタ、を含む。

【００４８】

有利なことに、このような実施形態では、他のデータベースは、アプリケーションとして効率的に動作する関係型データベースであっても良く、又はアプリケーションを介してグラフデータベースにリンク付けされ、グラフデータベース内のデータをアクセスし変更しても良い。このような実施形態は、共通の、相互運用可能な、マシンアクセス可能なデータ記憶を提供する。このようなデータ記憶のスキーマは、データベース全体の再コーディングを必要とすることなく簡単に変更できる。

【００４９】
コネクタは、外部データソースと通信可能なソフトウェアライブラリを有しても良い。一例として、ＪＤＢＣドライバは、関係型データベースへのコネクタである。この文脈では、コネクタは、外部ソースフォーマットからトリプルへ（及びその逆に）データを変換する。
【００５０】
本発明の別の態様の実施形態では、（ノードのピアツーピアネットワークのような）分散型ノードネットワークのノードとして動作するよう構成されるコンピューティング装置であって、前記コンピューティング装置は、（ＲＤＦトリプルのような）トリプルとしてエンコードされたグラフデータのデータベースのサブセットを格納し、各トリプルは、主語、述語及び目的語を有し、各トリプルは該トリプルのデータに従って順序付けられたデータアイテムのセットの中のデータアイテム内に格納され、前記分散型ノードネットワークの複数のノードに渡って分散され、前記データアイテムの各々がマッピングされるノードは、前記順序付けられたセット内のデータアイテムの位置に依存し、各トリプルは、データアイテム内で主語が述語及び目的語より前にある第１の構成、データアイテム内で述語が主語及び目的語より前にある第２の構成、データアイテム内で目的語が主語及び述語より前にある第３の構成、のうちのそれぞれ異なる構成を有する２以上のデータアイテム内に格納される、コンピューティング装置が提供される。
【００５１】
本発明の別の態様の実施形態では、コンピューティング装置により実行されると、前記コンピューティング装置に上述のコンピューティング装置として動作させる、コンピュータプログラムが提供される。
【００５２】
本発明の好適な特徴は、単なる例として添付の図面を参照して以下に説明される。
【図面の簡単な説明】
【００５３】
【図１】本発明の実施形態の概略図である。
【図２】本発明の実施形態を実装する例示的なシステムアーキテクチャである。
【図３】本発明の実施形態を実装する別のシステムアーキテクチャである。
【図４】本発明を実装するグラフデータベースがデータベース連合実装でどのように使用され得るかの説明である。
【図５】本発明を実装するグラフデータベースがデータベース連合実装においてデータベース更新を実行するためにどのように用いられ得るかの第１の例である。
【図６】本発明を実装するグラフデータベースがデータベース連合実装においてデータベース更新を実行するためにどのように用いられ得るかの第２の例である。
【発明を実施するための形態】
【００５４】
図１は、本発明の実施形態を現す概略図である。本実施形態では、分散ネットワークを形成する４個のネットワークノード１０がある。この特定の実施形態では、使用される通信パラダイムはＰ２Ｐであるが、実施形態は他の通信パラダイムと共に機能し得る。４個のネットワークノードは、それぞれａ－ｄでラベル付けされている。ラベルは、ネットワークノードのアドレスの例であり、勿論、ネットワークノードを互いに識別させる任意のデータであっても良い。本実施形態では、ネットワークノードは、それぞれ、記憶サーバのような別個のコンピュータであると考えられる。各ネットワークノード１０は、自身の記憶ユニット１０１を有する。しかしながら、単一のコンピューティング装置が１より多い記憶ユニットを有し、各記憶ユニットがそれ自体のアドレスを有し他から別個に問い合わせられるように構成されることも可能である。このような例では、単一のコンピューティング装置が１より多いネットワークノードとして取り扱われても良い。ネットワークノード１０は、プロセスであるとも考えられる。ここで、プロセスは、単に、特定の機能を提供するために特定のソフトウェアを実行するマシンである。

【００５５】
ネットワークノード１０間の線は、データコネクションを現す。データコネクションは、あるネットワークノード１０から別のネットワークノード１０へ敷設されたハードワイヤードの専用ケーブルであっても良く、必要に応じて確立されるチャネル又は無線搬送波を有する無線コネクションであっても良い。或いは、データコネクションは、ケーブル、スイッチ、無線リンク及び他の通信手段を含むネットワークを介して実現されても良い。各ネットワークノード１０は、互いに他のネットワークノ―ド１０とのデータコネクションを有しなくても良い。しかしながら、各ネットワークノード１０は、少なくとも１つの他のネットワークノード１０と通信可能に構成され、他のネットワークノード１０又は他のネットワーク機器を介したとしても各ネットワークノード１０が互いに通信できるようにすべきである。図１に示した実施形態では、各ネットワークノード１０は、それぞれの他のネットワークノード１０とのデータコネクションを設けられている。
【００５６】
本実施形態のピアツーピアネットワークにおける各ネットワークノード１０は、２つのデータアイテムを格納する。勿論、本発明の実施形態は、ピアツーピアネットワークにおけるネットワークノードの数（最小数２より多い）又は各ネットワークノード１０に格納されるデータアイテムの数により限定されない。
【００５７】
図中、データアイテムは、括弧＜＞で囲まれたシンボルのグループにより表現される。各シンボルは、３種類の要素（Ｐ＝述語（predicate）、Ｓ＝主語（subject）、Ｏ＝目的語（object））を示す文字とデータアイテムとして格納されているＲＤＦトリプルを識別する数を含む英数字対である。これらのシンボルは、本発明の実施形態が格納するデータの実際の内容ではなく、データが格納され得る方法を示すために含まれる。本実施形態ではトリプルはＲＤＦトリプルであるが、本発明の実施形態は他のトリプルフォーマットのグラフデータをエンコード可能である。よって、４つのＲＤＦトリプル１、２、３、４がある。各ＲＤＦトリプルは、２つのデータアイテムとして格納される。各データアイテムは、異なる構成を有する。一方の構成では、主語が述語及び目的語より前にある。他方の構成では、述語が主語及び目的語より前にある。
【００５８】
データアイテムは、データアイテム内で最初に現れるＲＤＦトリプル要素の種類に従って、及び所定のメトリックによってデータアイテム同士を比較することにより確立されたデータアイテムセットの中のデータアイテムの位置に従って、ネットワークノード１０にマッピングされる。例えば、メトリックはアルファベット順であり、ＲＤＦトリプル１の主語は「Adam」であり、ＲＤＦトリプル２の主語は「Acorn」であり、ＲＤＦトリプル３の主語は「Abdul」であり、ＲＤＦトリプル４の主語は「Aaron」であっても良い。したがって、最初に主語を有するデータアイテムがｂ及びｄとラベル付けされたネットワークノードに渡って（且つ、ｂ、ｄの順に）格納され、データアイテムが図１に示したようにネットワークノードにマッピングされることが、マッピング関数により確立されている。データアイテムは、特定のネットワークノード１０内に順序付けられて格納されても良く、順序付けられないが例えばクエリに応答して順序が確立できるように格納されても良い。
【００５９】
因みに、マッピングは、ネットワークノード１０により、互いに協調した１より多いネットワークノードにより、リモートパーティにより、中央マッピングモジュールにより、又は任意の他の方法で実行されても良い。各ネットワークノード１０がマッピングを実行するよう構成されても良い。各ＲＤＦトリプル要素を比較するために用いられる順序付けメトリックは、互いに同一であっても良く、異なるメトリックが異なる要素のために用いられても良い。図１の実施形態では、述語を比較するために用いられる順序付けメトリックは長さであっても良い。よって、マッピング関数が、順序付けられるときデータアイテムの１番目のサブセットをｃとラベル付けされたネットワークノードに、２番目のサブセットをａとラベル付けされたネットワークノードに置き、Ｐ３は最も長い述語を有し、次

がＰ１、次がＰ４、最後にＰ２が続く。

【００６０】

図１に示した実施形態では、ｂ及びｄとラベル付けされた各ネットワークノードには偶数のデータアイテムがある。好適な実施形態では、各ネットワークノード１０は、データアイテムの等しい分担又はほぼ等しい分担を格納する。しかしながら、本発明の実施形態は、これに関して限定されない。マッピングメカニズム（又はマッピング関数）は、ネットワークノードに渡るデータアイテムの一様でない分配を生じても良い。

【００６１】

アプリケーションがデータベースからのグラフデータにアクセスしたい場合、アプリケーションは、データベースへのクエリの形式をフォーマットするＡＰＩラッパに従ってクエリを作成しても良い。例えば、アプリケーションは、データベースに問い合わせて、アルファベット順に「Acorn」から「Aaron」の間の主語を有する範囲のデータアイテムを検索しても良い。

【００６２】

アプリケーションは、ネットワークノード１０のうちの１つに要求を送信する。要求を受信したネットワークノードは、マッピング関数を用いて、どのネットワークノードにおいて主語「Acorn」を有する最高順位のデータアイテムが見付かるか、どのネットワークノードにおいて主語「Aaron」を有する最低順位のデータアイテムが見付かるか、を決定する。上述の範囲のうちの２つの極限の位置に基づき、データアイテムの範囲全体の位置が決定できる。したがって、要求を受信したネットワークノードは、ｂとラベル付けされたネットワークノードへ、主語「Acorn」を有するデータアイテム及びより低い順位の任意のデータアイテムに対する下位範囲要求をブロードキャストする。本例では、図１で＜Ｓ２，Ｐ２，Ｏ２＞と示したデータアイテムのみが、返されるだろう。また、主語「Aaron」を有するデータアイテム及びより高い順位の任意のデータアイテムに対するサブ範囲要求が、ｄとラベル付けされたネットワークノードへ発行される。本例では、＜Ｓ３，Ｐ３，Ｏ３＞及び＜Ｓ４，Ｐ４，Ｏ４＞と示したデータアイテムが、返されるだろう。要求を受信したネットワークノードは、アプリケーションに対してデータアイテム＜Ｓ２，Ｐ２，Ｏ２＞、＜Ｓ３，Ｐ３，Ｏ３＞及び＜Ｓ４，Ｐ４，Ｏ４＞で応答するよう構成される。

【００６３】

上述のクエリのコンピュータ的な効率は、従来のグラフデータ記憶技術により達成されるものよりも良好である。さらに、効率の節約は、より大きな且つより広範な分散データ環境においてより範囲を大きくする。

【００６４】

記憶システム、つまり、本発明の実施形態で用いられる、データベースが格納される記憶ユニット１０１の相互接続されたネットワークは、ピアツーピア（Ｐ２Ｐ）インタフェースを介して通信するプロセスのセットを有する。用語「プロセス」は、コンピュータ上で動くプログラムを表し、したがって、１又は複数のプログラム又はプロセスを動作させることによりネットワークノード１０がその機能を実現するよう構成されるという意味でネットワークノード１０と等価である。プロセスは、それ自体のメモリアドレス空間を有し、「ソケット」を通じて他のプロセスと通信するよう構成される。

【００６５】

トリプルは、グラフ中の情報を表す単純なデータユニットである。実装に依存して、トリプルは小さいサイズであっても良い。トリプルは、例えばウェブソースに関する情報を表しても良い。DBPediaは、現実世界のエンティティに関する情報を表すオンラインデータ記憶である。以下のＲＤＦトリプルは、本発明の実施形態においてトリプルがデータアイテムに格納され得る形式の例である。以下のＲＤＦトリプルｉ）－ｉｖ）は、http://dbpedia.org/resource/Aristotleにウェブリソースとして格納されたAristotle（現実世界のエンティティ）に関する情報を表す。

【数１】

i) "http://dbpedia.org/resource/Aristotle
http://dbpedia.org/ontology/birthPlace
http://dbpedia.org/resource/Stageira"
ii) "http://dbpedia.org/resource/Aristotle
http://purl.org/dc/elements/1.1/description ¥"Greek
philosopher¥"@en"
iii) "http://dbpedia.org/resource/Aristotle
http://www.w3.org/1999/02/22-rdf-syntax-ns#type
http://xmlns.com/foaf/0.1/Person"
iv) "http://dbpedia.org/resource/Aristotle
http://xmlns.com/foaf/0.1/name ¥"Aristotle¥"@en"

【００６６】

各トリプルは、Aristotleに関する特定の情報を表す。例えば、ｉ）は、ＵＲＩ http://dbpedia.org/resource/Aristotle（主語）により表されるリソースが別のリソースへのリンク、つまりhttp://dbpedia.org/ontology/birthPlaceに定められた関係種類である出生地を定めているリンク（述語）を有し、「出生地（birthplace）」関係により主語にリンク付けられた目的語はＵＲＩ http://dbpedia.org/resource/Stageiraにより表されるリソースであるという記述を表す。言い換えると、トリプルｉ）は、Aristotleが彼の出生地としてStageiraにリンク付けられるという情報を表す。同様に、トリプルｉｉ）は、ウェブリソース…Aristotle（主語）が解説（述語）として…“Greek Philosopher”（目的語）にリンク付けられるという情報を表す。同様に、トリプルｉｉｉ）は、ウェブリソース…Aristotle（主語）がPerson（目的語）タイプのリソース（述語）であるという情報を表す。トリプルｉｖ）は、ＵＲＩ…Aristotleに見付かったウェブリソース（主語）が関係「名前（name）」（述語）により文字列の目的語「Aristotle」（目的語）にリンク付けられるという情報を表す。

【００６７】

本発明の実施形態は、各トリプルを少なくとも２つの構成で格納し、各構成は他者に先行する異なるトリプル要素を有する。したがって、データベースは、主語、述語及び目的語のうちの少なくとも２つに基づきクエリにより効率的方法で問い合わせることができる。検索が実施されるメカニズムは、トリプルを格納するデータアイテムのセットに対して範囲クエリを実行することによる。トリプルは＜Ｓ，Ｐ，Ｏ＞の汎用形式なので、検索は、トリプルの１つ（又は複数）の要素を固定することにより行うことができる。しかしながら、これは、トリプルがトリプルの１より多い要素により順序付けられる場合、更に効率的になる。例えば、クライアントアプリケーションが次のＵＲＬで特定されるリソースに関する全ての情報を知りたい場合、
http://dbpedia.org/resource/Aristotle
本実施形態のネットワークノード１０により受信される範囲クエリは、この主語を１番目の要素として有する全てのデータアイテムを検索するためのクエリである。より一般的には、クエリを実行するために、range_query()プロシジャは、辞書順に比較可能な２つのキー、つまり下限と上限を受信する。この特定の例のクエリでは、＜Ｓ，Ｐ，Ｏ＞エンコーディングが効率的である。プロシジャは、下端のキーと上端のキーとの間の全てのキーを検索する。しかしながら、クエリは、自身にリンク付けられた次のＵＲＬで特定されるリソースを有するデータベース内の全てのリソースのリストを検索しても良い。
http://dbpedia.org/resource/Stageira
したがって、特定されるＵＲＬは目的語として問い合わせられ、同じ目的語を有するデ

ータアイテムは＜Ｓ，Ｐ，Ｏ＞エンコーディングで連続的に順序付けられないので、＜Ｓ，Ｐ，Ｏ＞エンコーディングは効率的ではない。しかしながら、例えば＜Ｏ，Ｐ，Ｓ＞エンコーディングにより、効率的になり得る。したがって、本発明の実施形態は、コンピュータ的に効率的な方法で実行できるクエリのプールを広げる。

【００６８】

トリプルは、順序がデータアイテム間に定められ、該順序が２つの端点の間のデータの線形範囲を定めるために用いられるように、データアイテムとして（又はデータアイテムにエンコードされて）格納される。線形範囲はセグメント又はサブセットに分けられ、各々の又は１つより多いサーバ又はネットワークノードがデータのサブセットの記憶を担う。データアイテムのネットワークノードへのマッピングは、マッピングメカニズム、例えばTrie（順序木）により論理的に整理される。マッピングメカニズムは、どのサーバ又はネットワークノードが範囲クエリにより指定された範囲に含まれるデータアイテムを格納しているかを効率的に検索するのを助ける。範囲クエリのために要求されたサブセットのリストを確立した後に、これらのサブセットを格納しているサーバのアドレスは、データアイテムをネットワークノードにマッピングするマッピング関数を用いて見付けられる。

【００６９】

データアイテム間の順序は、辞書順にキーを比較することにより確立されても良い。例えば、文字列「Ka」は文字列「Kc」より低い。クエリの主語＜Ｓ，＊，＊＞、クエリの述語＜＊，Ｐ，＊＞又はクエリの目的語＜＊，＊，Ｏ＞を固定することによりクエリから結果を効率的に検索できるためには、トリプルの３つの部分の全てがデータアイテム内の文字列の先行する要素として格納されなければならない。これを可能にするために、トリプルは、次のキーを用いてＫＶＳに３回挿入されなければならない。

＜Ｓ，Ｐ，Ｏ＞、＜Ｐ，Ｏ，Ｓ＞、＜Ｏ，Ｓ，Ｐ＞

このようにトリプルを３回格納することは、クエリの種類に拘わらず、データベースが同じ効率で任意のクエリを実行できるようにし、トリプルが必然的に複製され障害の場合に情報を再構成できるという更なる利点を有する。勿論、これらの利点は、格納されている各トリプルの３つのバージョンに依存しない。各トリプルの２つのバージョンを格納することも幾つかの利点をもたらし得る。ＫＶＳの実施形態では、トリプルの各バージョンは、同じ値にリンク付けされても良い。

【００７０】

図２は、本発明の実施形態の例示的なシステムアーキテクチャ、及びクライアントアプリケーションを特徴とする環境での該実施形態の実装を示す。図２は、１又は複数のコンピュータで動作するプログラムにより実現され得る、プロセスの階層型システムアーキテクチャを示す。例えば、クライアントアプリケーション２０はラッパＡＰＩ１２、ＲＤＦプロセッサ１４及び分散型キー値ストア１６と同じコンピュータで動作しないことも可能である。しかしながら、勿論、それらは１又は複数のマシンに渡って分散され、プロセスはシステムアーキテクチャ内の少なくとも隣接するプロセスへのデータコネクションを有する。

【００７１】

第１の層はクライアントアプリケーションプロセス２０である。クライアントアプリケーションプロセス２０は、本発明を実現するデータベースへのアクセスを望み得るマシンの例である。クライアントアプリケーション２０は、クラウド環境内に、クライアント－サーバアーキテクチャのサーバ内に設けられても良く、エンドユーザのマシンで動作されても良い。クライアントアプリケーション２０は、データベースにアクセスするための、例えば該データベースに格納されたデータを問い合わせるための特定の要件を有する。

【００７２】

ラッパＡＰＩ１２及びＲＤＦプロセッサ１４は、別個のプロセスとして設けられても良く、又は単一のプロセスで一緒に動作しても良い。ラッパＡＰＩ１２は、例えばシステムアーキテクチャ内の他のＡＰＩにより提供される基本機能の異なる特徴を結合する機能を提供することにより、クライアントアプリケーション２０がＲＤＦプロセッサ１４及び分

散型キー値ストア１６にアクセスするインタフェースを簡略化する。例えば、ＲＥＳＴはＲＤＦプロセッサ１４のＡＰＩラッパ１２として用いることができる。したがって、図２のシステムアーキテクチャにおいて、クライアントアプリケーション２０は、明確なラッパＡＰＩ１２を通じてＲＤＦプロセッサ１４と相互作用する。ラッパＡＰＩ１２は、ＲＤＦプロセッサ１４に含まれるＳＰＡＲＱＬエンジンへクエリを送信するためのインタフェースを有しても良い。

【００７３】

ＲＤＦプロセッサ１４は、ＳＰＡＲＱＬエンジンを有しても良い。ＳＰＡＲＱＬエンジンは、ラッパＡＰＩ１２を介してクライアントアプリケーション２０から受信した「複雑なクエリ」を幾つかの単純な範囲クエリに分割するよう構成される。ＲＤＦトリプルデータは、図１に関連して上述したように、順序付けられたデータアイテムのセットとしてネットワークノード１０に渡って格納される。図２のシステムアーキテクチャでは、「キー値ストア」（Key-Value　store：KVS）は、このような順序付けられたデータセットの例として含まれ、図１の実施形態のデータアイテムはＫＶＳにキーとして格納されている。ＫＶＳを含むデータベースは、ＲＤＦトリプルデータを格納し、そのキーに対して範囲クエリを実行するよう構成される。

【００７４】

図３は、本発明の実施形態を実装する別のシステムアーキテクチャである。図３では、プロセスの各列は、特定のマシン又はネットワークノード１０で動作しているプロセスを表し得る。或いは、例えばアプリケーション２０及び／又はデータフィード２２は、データベースにアクセスする目的でラッパＡＰＩ１２、ＲＤＦプロセッサ１４及び分散型ＫＶＳ１６プロセスを動作させるネットワークノード１０へのデータコネクションを割り当てられているリモートプロセスであっても良い。プロセスを動作させている特定のマシンは、図１に示したような実施形態におけるネットワークノードであっても良い。代替として、ネットワークノード１０は、実装の詳細に依存して、図３の分散型ＫＶＳプロセス１６と等価であっても良い。

【００７５】

個々のプロセスは図２に関連して議論したプロセスの例であるので、ここでは各プロセスの詳細な説明を省略する。データフィード２２は、特定の種類のアプリケーションの例、この例では例えば「put」コマンドを介してデータベースにデータを書き込むアプリケーションである。矢印は、プロセス間の相互作用のためのデータコネクションを表す。分散型ＫＶＳプロセスが別の分散型ＫＶＳプロセスと相互作用し、特定のノードで受信した範囲クエリがマッピング関数に従ってサブ範囲クエリに分割され、サブ範囲クエリが他の分散型ＫＶＳプロセスへ送信され得るようにしていることが分かる。

【００７６】

図３のアーキテクチャでは、各ＲＤＦプロセッサ１４は、分散型順序付きＫＶＳ１６のネットワークノード（図１のネットワークノード１０の例である）と通信するよう構成される。

【００７７】

本願明細書に記載する、データアイテム又はキーとしてトリプルを格納するメカニズムは、ＫＶＳの実施形態において、全てのデータがキー／値記憶のキーに格納されることを保証する。したがって、このような実施形態では、値は、更に豊富なグラフデータベースを実現する情報を格納するために利用可能である。値に格納できるデータの幾つかの例を、幾つかの実装例と共に以下に説明する。

【００７８】

キー／値記憶は、キーを値にマッピングし、関連するキー、又は詳細には該キーにエンコードされた若しくは格納されたトリプルに関する情報の（値フィールドへの）記憶を可能にする。全てのグラフ情報をキー内に保持する本発明の実施形態は、グラフデータの迅速な範囲検索及び追加意味データを値フィールドに格納する可能性を実現し、したがってデータベースの機能を向上する。例えば、実施形態は、グラフデータを格納するのみなら

ず、ＫＶＳの値を情報及びコードコンテナとして用いることによりイベントをトリガしメタデータを保持する豊富なグラフデータベースを提供し得る。このようなコンテナは、関連付けられたトリプルに関する（例えば）メタデータ情報及び／又はトリプル内の何かが変更されるときにタスクを実行するトリガのリスト（つまり、ソフトウェアコード、又はサーバ内のどこかに格納されたソフトウェアコードへの参照）を格納するよう構成されるリッチオブジェクトとして実施されても良い。この文脈においてリッチオブジェクトは、異なる方法で表された幾つかのデータアイテムを有し得るオブジェクト、及びコードが格納されているマシンで動作できる該コードを含む。

【００７９】

例示的な実施形態では、（ＲＤＦトリプルである）ＫＶＳの各キーは、オブジェクトタイプContainer（コンテナ）にマッピングされる。他のデータの中でも、コンテナオブジェクトは、以下の情報を格納する。

【００８０】

・Ｒａｗデータこれは、ＫＶＳにおいて不可解オブジェクトとして見なされ取り扱われるアプリケーション依存データを保持する。この文脈における不可解オブジェクトは、定義されていない不明なクラスのオブジェクトを含む。プログラマは、このようなオブジェクトの種類又はクラスを知らずに、該オブジェクトを処理するコードを生成できる。この目的のため、コンテナオブジェクトは、アプリケーション依存情報を設定及びゲットするＡＰＩを提供する。

【００８１】

・トリガ（Trigger）アプリケーションは、特定のイベントに関心のあるアプリケーションを非同期に、つまりＲＤＦトリプルが変更されるとき及び／又は削除されるときに通知するために用いられるコールバック関数を登録できる。この目的のため、コンテナオブジェクトは、各イベント（「挿入」、「読み出し」、「更新」及び／又は「削除」）のコールバック関数のリストと、アプリケーションにこれらのコールバック関数を登録及び登録解除させるＡＰＩとを保持するよう構成されても良い。アプリケーションは、自由に、任意の又は全てのイベントに対してコールバック関数を登録又は登録解除し、イベント毎に異なる関数を選択する（このような関数及びイベントはＡＰＩで指定されても良い）。例えば、「挿入」イベントに対する関数は、挿入の時に登録される。

【００８２】

本発明の実施形態の第１の実装例を説明する。第１の実装例では、グラフデータ、つまりＲＤＦトリプル自体は、キー値ペアのキーとして格納され、ＲＤＦトリプルに関連するメタデータにより補完される。メタデータは、問題になっているＲＤＦトリプルを表すキーに対応する値記憶の値フィールドに格納される。従来のシステムでは、メタデータは、ＲＤＦトリプルをクワッドに拡張することにより格納されていた。これは、拡張性がないので望ましくない。

【００８３】

問題となっている第１の実装例におけるデータベースは、インポートされたＲＤＦトリプルを格納するデータベースである。格納されたデータにアクセスするアプリケーションは、ＲＤＦトリプルがインポートされたソースに関する情報、例えばソースの識別表示を要求しても良い。実際に、本発明の実施形態は、データベースがレポジトリ間で交換される実装及び複雑な知識ベースが発行される実装に順応できる。第１の実装例では、各トリプルの各部分は相対的ＵＲＩを用いて表され、元のデータソースも各ＲＤＦトリプルの間で格納される必要のあるコンテキスト情報として識別される。

【００８４】

キー値ペアの値を形成するコンテナは、追加情報（元のデータソースの識別表示）が格納されるRawデータフィールドを有する。データベースに挿入れるトリプル毎に、元のデータソースのＵＲＬがコンテナに挿入され、ＫＶＳ内の値として設定される。

【００８５】

その後、本実施形態のデータベース内に格納されたデータにアクセスするアプリケーシ

ョンがトリプルに関するコンテキスト情報を要求するとき、アプリケーションは単に、問題のトリプルに対応する値フィールド内のコンテナに対するgetコマンドを発行し、Rawデータフィールドを読み出すことができる。

【００８６】

本発明の実施形態の第２の実装例を次に説明する。第２の実装例では、キー値ペアの値は、コンテナオブジェクト（特定のアドレスに一緒に格納された又はコンテナに関連付けられた特定のアドレスを介してアクセス可能なデータの集合体）であり、該コンテナは、キー値ペアのキーに符号化されたトリプルに関連付けられたコールバック関数を格納する。この第２の実装例では、コンテナの内容は、キャッシュ管理の目的で用いられる。

【００８７】

分散型システムでは、キャッシングは、システム性能を向上させる確立された技術である。リモートデータをローカルキャッシュに格納することは、分散型システムの構成要素間で必要な通信量を低減するのに役立つ。本発明を実現するグラフデータベースでは、ピアツーピアネットワーク内の各プロセスは、グラフの一部のみを保持する。したがって、各プロセスは、ピアツーピアネットワーク内のどこかに格納されたリモートトリプルのローカルキャッシュを保持しても良い。例えば、プロセスは、リモートトリプルの正式バージョンに所定頻度で又はそれより多くアクセスしているとき、リモートトリプルをキャッシュしても良い。したがって、トリプルのコピーがローカルキャッシュに格納される。或いは、所定頻度で又はそれより多くアクセスされている特定のトリプルが存在しても良い。したがって、各プロセスは、該トリプルのコピーをローカルキャッシュに保持するよう構成される。リモートトリプルのローカルコピーをキャッシュすることは、リモートトリプルの正式バージョンの内容にアクセスする（読み取る）ためにリモートプロセスと頻繁に通信する必要を取り除く。

【００８８】

このようなキャッシュ管理システムを有する実装は、（ローカルキャッシュ内にコピーされた）トリプルの正式バージョンが変更又は削除されるとき、そのローカルキャッシュを無効にする又は更新するメカニズムを設けられることが望ましい。コールバック関数は、キャッシュされたトリプルの正式バージョンのコンテナに格納される。例えば、リモートトリプルにアクセスする１つのプロセスを生じるクエリの生成、したがってトリプルのキャッシュされたコピーの生成を担うクライアントアプリケーションは、リモートトリプルの正式バージョンと共にコールバック関数を登録しても良い。このコールバック関数により、キャッシュされたコピーを格納するプロセスは、トリプルの正式バージョンが削除又は変更されるとき、非同期に通知される。コールバック関数は、特定のイベントが生じるときに呼び出されるよう登録される関数である。この関数は、特定のイベントに応答して呼び出されるとき、通知又はデータの変更のような特定の機能を実行するよう構成される。コールバック関数の正確な形式は、実装に固有であるが、呼び出されると実行されるコードのブロックであり得る。

【００８９】

ローカルキャッシュを更新し又は無効にする例示的なメカニズムは、２つの異なるコールバック関数（又はコールバック方法）、つまりｉ）update()関数、ii)delete()関数の提供である。関数ｉ）は、update()関数がコンテナ内の「変更」イベントに登録されているとき、ローカルキャッシュをトリプルの正式バージョンの新しい値で更新するよう動作する。どちらの関数の引数も、例えばトリプルのキャッシュされたコピーであっても良い。関数ii)は、トリプルの正式バージョンが削除されるとき、該トリプルのキャッシュされたコピーを削除するよう動作する。delete()関数は、コンテナ内の「削除」イベントに登録されても良い。

【００９０】

本発明の一実施形態の第３の実装例を次に説明する。第３の実装例では、本発明を実現するグラフデータベースは、１より多い関係型データベースの連合のために用いられる。

【００９１】

有利なことに、（グラフデータベース内の）グラフのようなデータを表すことは、データベースのスキーマをより簡単に変更できるようにする。この文脈におけるデータベースのスキーマは、形式言語で記述されるデータベースの構造を含む。関係型データベースのスキーマを変更することは、プログラミングの変更を含む多くの管理業務を必要とする。例えば、データ及び該データにアクセスするアプリケーションのフォーマットの両方を変更する必要がある。これに対し、本発明の実施形態では、グラフデータベースのデータベーススキーマは、単純にデータ自体である。したがって、スキーマを変更することは、データを変更するように簡単である。

【００９２】

このデータベース連合の実装例では、異なるデータ及び異なるスキーマを有するデータベースは、本発明を実現するグラフデータベースを用いて相互に関連付けられる。

【００９３】

図４は、独立関係型データベース３０のセット、及びアプリケーション２０によりアクセス可能なグラフデータベース４０内のデータベース３０からのデータの表現を示す。各グラフノード４１は、データベース３０からのデータアイテムを表す。図４の破線は、グラフノード４１とグラフノード４１が表すデータアイテムを格納するデータベース３０との間のリンクを示す。幾つかのデータアイテムは、データベース間で共有される。例えば、人物のような特定のエンティティは、２つの独立したデータベース内に表されても良い。（例えば、「owl:sameAs」ＲＤＦ注釈を用いて）エンティティのアイデンティティを表す情報をグラフに追加することにより、望ましくは別個のデータベース内の独立したデータアイテムにより表される該エンティティは、単一のグラフノードとして格納できる。このようなグラフノードは、「共有された」データアイテムを表す。実線で示したグラフの端は、データアイテム間の関係を表す。

【００９４】

第２の実装例と同様に、この第３の実装例では、キー値ペアの値はコンテナオブジェクトである。しかしながら、本例のコンテナオブジェクトは、それぞれトリプルに関する１又は複数の以下の情報を格納する。

【００９５】

・トリプルにより表される情報を含む関係型データベースのリスト。

【００９６】

・上記データベースの各々について、該トリプルに含まれる情報を更新するために用いられる該データベース固有のコネクタ。

【００９７】

・他の関連するグラフアイテムを更新するためのトリガ。

【００９８】

コネクタは、例えば外部データソースと通信可能なソフトウェアライブラリとして実装されても良い。一例として、ＪＤＢＣドライバは、関係型データベースへのコネクタである。この文脈において、コネクタは、外部ソースで用いられるフォーマットからトリプルへ、及びその逆に、データを変換するよう構成されても良い。

【００９９】

Container、関連付けられたソフトウェアライブラリに格納された情報を用いて、データベース４０は、関係型データベース３０のデータを２つの方法で更新するよう動作可能である。

【０１００】

第一に、グラフ４０を直接に問い合わせ更新するよう構成され、及び関係型データベース３０に反映されるべき更新のためのアプリケーションが生成され得る。これを図５に示す。図５は、図４のシステムに点線を追加したものである。点線は更新を表す。したがって、クライアントアプリケーション２０は、グラフデータ４０を直接更新するよう構成され、グラフ４０内の特定のノード４１を変更する更新を実行する。更新されたノードにより表される情報は、２つのデータベースに格納される。したがって、更新されたノード４

１を関係型データベース３０にリンク付けする点線は、これらのデータベース３０の更新を示す。詳細には、更新されたノード４１に関する情報を表す１又は複数のトリプルが更新される。ＫＶＳ内のこれらのトリプルにリンク付けされた１又は複数のコンテナは、該トリプル内の変化を反映するために更新される必要のあるデータベースのリストと、該更新を実行するためのコネクタを含む。

【０１０１】

第２の方法では、関係型データベースのデータは、グラフデータ及び図６に示したコンテナ内に格納された情報を用いて更新できる。図６では、グラフデータベース４０は、異なる関係型データベース３０の間の中間物として実装される。例えば、ＫＶＳ記憶内のトリプルにリンク付けされたコンテナ内のデータベース３０のリストを格納することは、該トリプルにより記述されるリソースが１より多い関係型データベースに格納され、したがってリストが「X owl:sameAs Y」情報を反映する情報を含むことを間接的に示す。したがって、データベースアプリケーション２４ａが特定のデータベース３０ａ内のデータアイテムを更新する場合（点線で示した更新、本例では更新は注釈「発行された更新」と共に示される）、該更新は、グラフに反映され、コンテナ内に格納されたコネクタ（コールバック関数）を用いてデータアイテムが格納されている他の関係型データベース３０ｂ内の同じデータに同じ更新を実行する。したがって、独立関係型データベース３０ｂにアクセスする独立データベースアプリケーション２４ｂは、データベースアプリケーション２４ａにより発行されるデータの更新を反映する。

【０１０２】

コネクタは、コールバック関数の一例である。コールバック関数及びそれらが提供され得る例示的なメカニズムを以下に更に詳細に説明する。上述の第２及び第３の実施例では、及び本発明の実施形態において一般的に、ソフトウェアパッケージは、データ記憶システム内で生じる特定の動作又はイベントを「傾聴」するためにコールバック関数を登録できる。以下は、このようなコールバック関数が本発明の実施形態においてどのように実現できるかの一例である。例えば、通知及びその詳細を提供するためにコールバック関数を用いることができる動作は、トリプルに行われているリードアクセス、トリプルに行われているライトアクセス、新しいトリプルの生成、及び／又はトリプルの削除を含む。ソフトウェアパッケージは、コールバック関数が登録されているサーバで動作する。トリプルに動作が実行される（リード／ライト／生成／削除）ときは常に、記憶システムは、該トリプルに関連付けられたコンテナについてコールバック関数を調べるよう構成される。発生した特定の動作又はイベントに対して登録されたコールバック関数が存在する場合、関数が実行される。必要な関数及びリソースの詳細な特性は実装に依存し、関数のプログラマにより定められても良い。幾つかの実施形態では、クライアントアプリケーション２０にフィードバックを与えるために、ＲＤＦプロセッサ１４との通信が要求されても良い。このような例では、図３を参照すると、ＲＤＦプロセッサ１４とＫＶＳサーバ１６との間の通信チャネルは、双方向であり得る。ＲＤＦプロセッサ１４は、フィードバックをＡＰＩラッパ１２へ、さらにクライアントアプリケーション２０へ転送する。また、外部ソフトウェアへのコネクションが維持され得る。

【０１０３】

トリガイベントは、コールバック関数の一例である。コンテナは、関連付けられたトリプルが変更される場合、更新されるべき他のトリプルのリストを格納し得る。更新されたトリプルを格納するＫＶＳサーバ１６内に「updateTriple」として格納される方法は、トリプルが更新されることを要求する。ＫＶＳ記憶１６は、どこに他のトリプルが格納されているかを調べ、それらを削除し、該トリプルの新しい更新されたバージョンを挿入する。削除及び再挿入によりトリプルを更新することは、格納されたデータアイテム（トリプル）の再順序付けを実現するので、望ましい。新しいバージョンは、異なるサーバに格納され得る。

【０１０４】

上述の態様の何れにおいても、種々の特徴は、ハードウェアで、又は１若しくは複数の

プロセッサで動作するソフトウェアモジュールとして実施されても良い。ある態様の特徴は、他の態様の特徴に適用されても良い。

【０１０５】

本発明は、上述の任意の方法を実行するコンピュータプログラム又はコンピュータプログラムプロダクト、及び上述の任意の方法を実行するプログラムを格納しているコンピュータ可読媒体も提供する。本発明を実施するコンピュータプログラムは、コンピュータ可読媒体に格納されてもよい。或いは、例えば、インターネット・ウェブサイトから提供されるダウンロード可能なデータ信号のような信号形式又は任意の他の形式であってもよい。

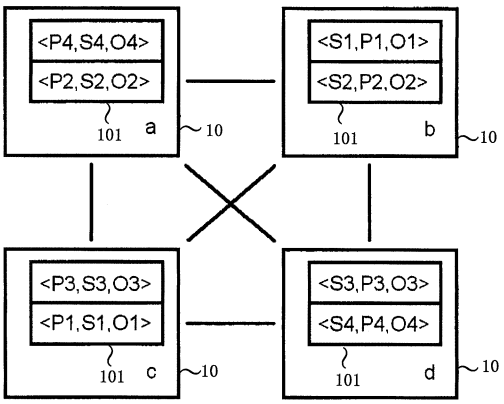
【符号の説明】

【０１０６】

１０　ネットワークノード
１２　ラッパＡＰＩ
１４　ＲＤＦプロセッサ
１６　分散型キー値ストア
２０　クライアントアプリケーション
２２　データフィード
３０　関係型データベース
４０　連合グラフデータベース
１０１　記憶ユニット

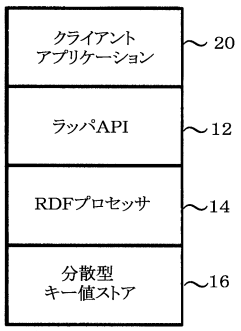
【図１】

本発明の実施形態の概略図



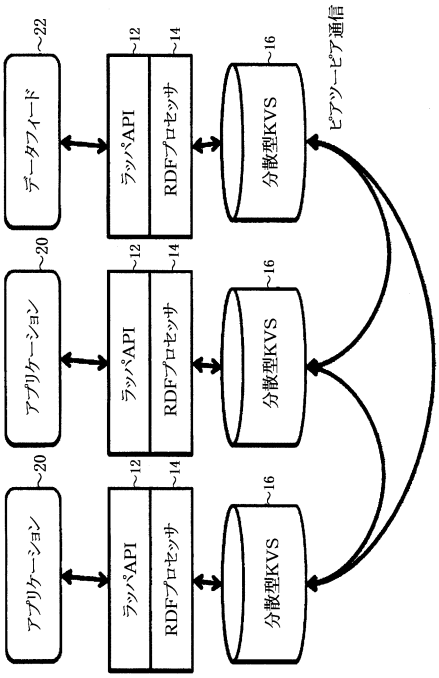
【図２】

本発明の実施形態を実装する例示的なシステムアーキテクチャ



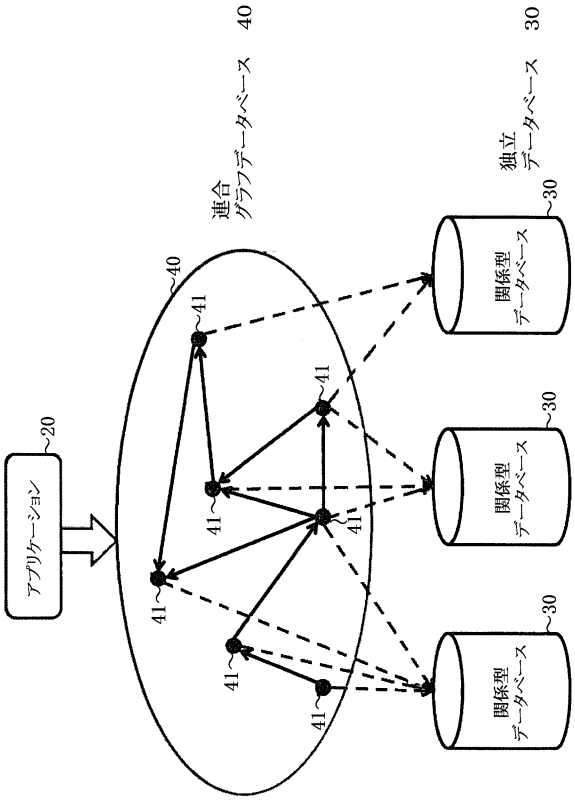
【図３】

本発明の実施形態を実装する別のシステムアーキテクチャ



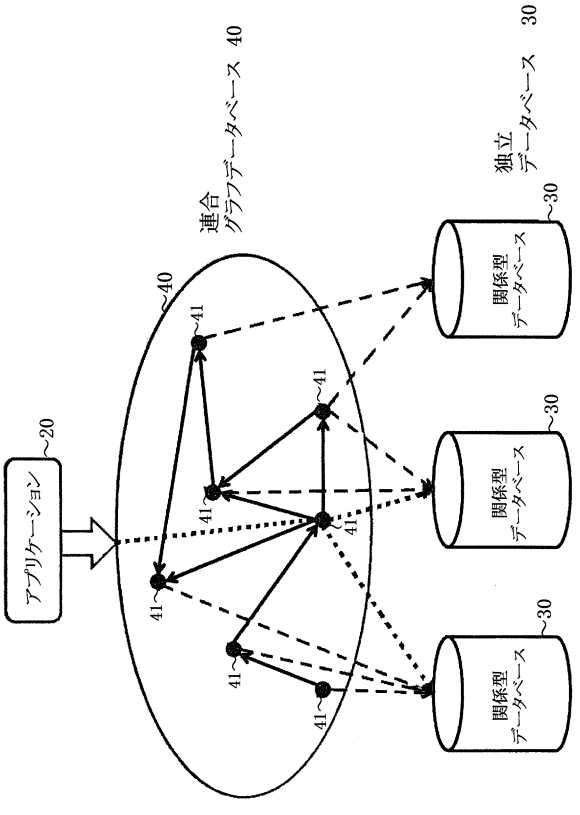
【図４】

本発明を実装するグラフデータベースがデータベース連合実装でどのように使用され得るかの説明



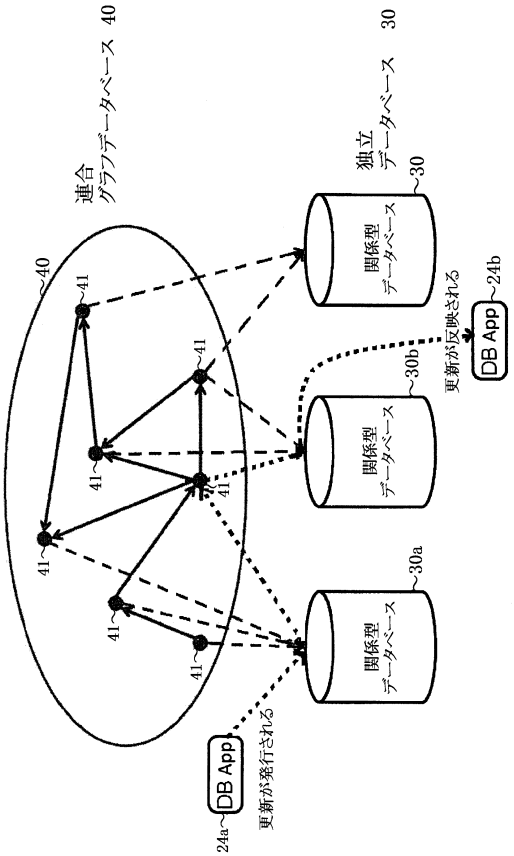
【図５】

本発明を実装するグラフデータベースがデータベース連合実装においてデータベース更新を実行するためにどのように用いられ得るかの第1の例



【図６】

本発明を実装するグラフデータベースがデータベース連合実装においてデータベース更新を実行するためにどのように用いられ得るかの第2の例



フロントページの続き

(72)発明者　松塚　貴英
イギリス国, ダブリュ３　０ジェイエヌ, ロンドン, チャーチル　ガーデンズ　３３番

審査官　川▲崎▼　博章

(56)参考文献　米国特許出願公開第２００４／０１９３６５３（ＵＳ, Ａ１）
米国特許出願公開第２００９／０２６５３０１（ＵＳ, Ａ１）
Ｍｉｎ　Ｃａｉ, Ｍａｒｔｉｎ　Ｆｒａｎｋ, RDFPeers: A Scalable Distributed RDF Repository based on A Structured Peer-to-Peer Network, Proceedings of the 13th Conference on World Wide Web, 米国, ＡＣＭ, ２００４年　5月２１日, ｐ.６５０－６５７, [online],<URL:http://dl.acm.org/citation.cfm?id=988760>

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ　１７／３０