

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2005-302004

(P2005-302004A)

(43) 公開日 平成17年10月27日(2005. 10. 27)

(51) Int.Cl.⁷

G06F 13/00

G06F 12/00

F I

G06F 13/00 520B

G06F 13/00 540C

G06F 12/00 510B

G06F 12/00 533J

テーマコード (参考)

5B082

審査請求 未請求 請求項の数 77 O L (全 41 頁)

(21) 出願番号 特願2005-73985 (P2005-73985)

(22) 出願日 平成17年3月15日 (2005. 3. 15)

(31) 優先権主張番号 10/825, 735

(32) 優先日 平成16年4月15日 (2004. 4. 15)

(33) 優先権主張国 米国 (US)

(特許庁注：以下のものは登録商標)

1. Bluetooth

(71) 出願人 500046438

マイクロソフト コーポレーション

アメリカ合衆国 ワシントン州 9805

2-6399 レッドモンド ワン マイ

クロソフト ウェイ

(74) 代理人 100077481

弁理士 谷 義一

(74) 代理人 100088915

弁理士 阿部 和夫

(72) 発明者 ダン テオドシュウ

アメリカ合衆国 98052 ワシントン

州 レッドモンド ワン マイクロソフト

ウェイ マイクロソフト コーポレーシ

ョン内

最終頁に続く

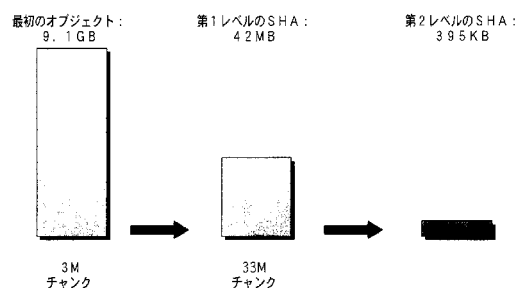
(54) 【発明の名称】 遠隔差分圧縮用の効率的アルゴリズムとプロトコル

(57) 【要約】

【課題】 限られた帯域幅のネットワーク上でオブジェクトを更新する方法およびシステムの提供。

【解決手段】 オブジェクトは、必要なデータ転送が最小化されるように遠隔差分圧縮 (RDC) 手法を使用して2つ以上のコンピューティングデバイス間で更新される。効率的な大きなオブジェクトの転送が、それ自体のメタデータに対してこのRDCアルゴリズムを再帰的に適用することによって達成される。すなわち、1つまたは複数の再帰的ステップをこのケースにおいて使用してこのRDCアルゴリズムによってこのネットワーク上で送信されるメタデータの量を減少させることができる。オブジェクト、および／または、シグネチャおよびチャンク長リストを、動的に決定されたロケーションに境界を位置づけることによってチャンキングすることができる。数学関数が、可能性のあるチャンク境界に関連してホライズンウィンドウ内に関連づけられたハッシュ値を評価する。

【選択図】 図6



【特許請求の範囲】

【請求項 1】

ローカルデバイスとリモートデバイスとの間で、ネットワーク上でオブジェクトを更新するためのシステムであって、

前記リモートデバイス上の第 1 のオブジェクトのバイトオフセットごとに第 1 のフィンガプリント関数を計算する手段と、

前記第 1 のフィンガプリント関数に基づいて前記リモートデバイス上の前記第 1 のオブジェクトをチャンキングする手段と、

前記リモートデバイス上で前記第 1 のオブジェクトに関連するチャンクごとにリモートシグネチャを計算する手段と、

前記リモートデバイス上で前記第 1 のオブジェクトに関連するリモートのシグネチャおよびチャンク長のリストを生成する手段と、

前記ローカルデバイス上で第 2 のオブジェクトのバイトオフセットごとに第 2 のフィンガプリント関数を計算する手段であって、前記第 1 および第 2 のオブジェクトが互いに関連しており、ここで前記第 1 のフィンガプリント関数が前記第 2 のフィンガプリント関数にマッチングさせられる手段と、

前記第 2 のフィンガプリント関数に基づいて前記ローカルデバイス上で前記第 2 のオブジェクトをチャンキングする手段であって、該第 2 のオブジェクトをチャンキングする手段に、前記リモートデバイス上で前記第 1 のオブジェクトをチャンキングする手段が前記ローカルデバイス上でマッチングさせられる手段と、

前記ローカルデバイス上で前記第 2 のオブジェクトに関連する各チャンクごとにローカルシグネチャを計算する手段であって、該ローカルシグネチャを計算する手段が、前記リモートシグネチャを計算する手段にマッチングさせられる手段と、

前記ローカルデバイス上でローカルのシグネチャおよびチャンク長のリストを生成する手段であって、該ローカルのシグネチャおよびチャンク長のリストが前記第 2 のオブジェクトに関連する手段と、

帯域幅の使用が前記ローカルデバイスへの前記リモートのシグネチャおよびチャンク長のリストの転送について最小になるように、前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスへの前記リモートのシグネチャおよびチャンク長のリストのチャンキングされた伝送をネゴシエーションする手段と、

前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストを前記リモートのシグネチャおよびチャンク長のリストと比較することによって前記第 1 のオブジェクトと前記第 2 のオブジェクトの間の差分を識別する手段と、

前記第 1 のオブジェクトと前記第 2 のオブジェクトの間の差分が前記ローカルデバイスによって識別されるとき前記リモートデバイスから少なくとも 1 つの更新されたオブジェクトチャンクの伝送を要求する手段と、

前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスに前記少なくとも 1 つの更新されたオブジェクトチャンクを伝送する手段と、

前記少なくとも 1 つの更新されたオブジェクトチャンクを用いて前記ローカルデバイス上で前記第 1 のオブジェクトのコピーを再組立てする手段と

を備えることを特徴とするシステム。

【請求項 2】

前記リモートデバイスから前記第 1 のオブジェクトについての更新を要求する手段をさらに備えることを特徴とする請求項 1 に記載のシステム。

【請求項 3】

前記ローカルデバイスから前記第 1 のオブジェクトについての更新を要求する手段をさらに備えることを特徴とする請求項 1 に記載のシステム。

【請求項 4】

前記リモートのシグネチャおよびチャンク長のリストのチャンキングされた伝送をネゴシエーションすることは、前記リモートデバイスから前記ローカルデバイスに前記リモート

10

20

30

40

50

トのシグネチャおよびチャンク長のリストの少なくとも一部分を送信することを含むことを特徴とする請求項 1 に記載のシステム。

【請求項 5】

前記ローカルデバイスおよび前記リモートデバイスのうちの一方は、クライアントであり、前記ローカルデバイスおよび前記リモートデバイスのうちの他方は、サーバであることを特徴とする請求項 1 に記載のシステム。

【請求項 6】

前記ネットワークは、直接配線接続、パラレルポート、シリアルポート、U S B ポート、I E E E 1 3 9 4 ポート、無線接続、I R ポート、ブルートゥースポート、有線ネットワーク、無線ネットワーク、ローカルエリアネットワーク、ワイドエリアネットワーク、
10
ウルトラワイドエリアネットワーク、インターネット、イントラネット、およびエクストラネットのうちの少なくともいずれか 1 つであることを特徴とする請求項 1 に記載のシステム。

【請求項 7】

前記ローカルデバイス上で前記第 2 のオブジェクトに関連するチャンクごとに前記ローカルシグネチャを計算する前記手段は、前記ローカルデバイス上で前記チャンクに対して適用される強化ハッシング関数を含むことを特徴とする請求項 1 に記載のシステム。

【請求項 8】

前記ローカルデバイス上で前記第 1 のオブジェクトと前記第 2 のオブジェクトの間の差分を識別する前記手段は、
20

前記リモートのシグネチャおよびチャンク長のリストを前記ローカルのシグネチャおよびチャンク長のリストと比較する手段と、

前記リモートのシグネチャおよびチャンク長のリストと前記ローカルのシグネチャおよびチャンク長のリストの間の少なくとも 1 つの差分を識別する手段と、

前記リモートのシグネチャおよびチャンク長のリストに前記少なくとも 1 つの差分をマッピングする手段と、

前記少なくとも 1 つの差分と前記リモートのシグネチャおよびチャンク長のリストの間での前記マッピングから前記少なくとも 1 つの更新されたオブジェクトチャンクを識別する手段と

を備えることを特徴とする請求項 1 に記載のシステム。
30

【請求項 9】

前記リモートデバイス上で前記第 1 のオブジェクトのバイトオフセットごとに前記第 1 のフィンガプリント関数を計算する前記手段は、

前記第 1 のオブジェクトに関連する各バイト位置の周囲で参照されるスモールウィンドウを提供する手段と、

バイト位置ごとに前記スモールウィンドウを使用してフィンガプリントを生成する手段と

を備えることを特徴とする請求項 1 に記載のシステム。

【請求項 10】

前記第 1 のオブジェクトに関連するデータタイプ、前記第 2 のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記ネットワークの特性と、前記第 1 のオブジェクトに関連する使用モデルと、前記第 2 のオブジェクトに関連する使用モデルのうちの少なくとも 1 つに基づいて、前記スモールウィンドウに関連するウィンドウサイズを調整する手段をさらに備えることを特徴とする請求項 9 に記載のシステム。
40

【請求項 11】

前記第 1 のフィンガプリント関数は、ラビン多項式を使用したハッシュ関数、循環シフティングハッシュ関数、3 2 ビットアドラーハッシュ関数のうちの少なくとも 1 つを含むことを特徴とする請求項 1 に記載のシステム。

【請求項 12】

前記リモートデバイス上で前記第1のオブジェクトをチャンキングする前記手段は、少なくとも1つのチャンキングパラメータを決定する手段を備えることを特徴とする請求項1に記載のシステム。

【請求項13】

前記リモートデバイス上で前記第1のオブジェクトをチャンキングする前記手段は、前記少なくとも1つのチャンキングパラメータからチャンキングホライズンを決定する手段と、

前記第1のオブジェクト内の各位置においてハッシュ値を計算する手段と、

前記第1のオブジェクト内の各位置の周囲の前記チャンキングホライズン内に位置づけられたハッシュ値に数学関数を適用する手段と、

前記数学関数が満たされるときにカットポイントおよびチャンキング境界のうちの少なくとも1つを指定する手段と、

前記指定されたカットポイントを用いて前記第1のオブジェクトをチャンキングする手段と

をさらに備えることを特徴とする請求項12に記載のシステム。

【請求項14】

前記数学関数は、前記ホライズン内の最大値を決定するステップ、前記ホライズン内の最小値を決定するステップ、および前記ホライズン内のハッシュ値の間の差分を評価するステップのうちの少なくとも1つを含むことを特徴とする請求項13に記載のシステム。

【請求項15】

前記リモートデバイス上で前記第1のオブジェクトをチャンキングする前記手段は、

前記少なくとも1つのチャンキングパラメータからホライズン、トリガ値、および他のトリガのリストを決定する手段と、

前記第1のオブジェクト内の位置ごとにハッシュ値を計算する手段と、

各計算済みのハッシュ値に数学関数を適用する手段と、

前記数学関数が、所与のオフセットにおいて前記トリガ値を獲得し、前記ホライズンが提供するすべての対応するオフセットにおいて前記他のトリガを獲得するときにカットポイントチャンキング境界の少なくとも1つを指定する手段と、

前記指定されたカットポイントを用いて前記第1のオブジェクトをチャンキングする手段と

を備えることを特徴とする請求項12に記載のシステム。

【請求項16】

前記数学関数は、ハッシュ値をブール値にマッピングする述語、およびハッシュ値を適切な小さなドメイン中に区分する別の数学関数のうちの少なくとも1つを含むことを特徴とする請求項13に記載のシステム。

【請求項17】

前記第1のオブジェクトに関連するデータタイプ、前記第2のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記ネットワークの特性と、前記第1のオブジェクトに関連する使用モデルと、前記第2のオブジェクトに関連する使用モデルのうちの少なくとも1つに基づいて、前記少なくとも1つのチャンキングパラメータを調整する手段をさらに備えることを特徴とする請求項12に記載のシステム。

【請求項18】

前記リモートデバイス上で前記少なくとも1つの更新されたオブジェクトチャンクの伝送を求める前記要求を受信する手段と、

前記少なくとも1つの更新されたオブジェクトチャンクの伝送を求める前記受信された要求に応答して前記リモートデバイス上で前記第2のオブジェクトから前記少なくとも1つの更新されたオブジェクトチャンクを抽出する手段と、

前記リモートデバイスを用いて前記ネットワーク上で前記少なくとも1つの更新されたオブジェクトチャンクを送信する手段と、

10

20

30

40

50

前記ローカルデバイスを用いて前記ネットワークから少なくとも1つの更新されたオブジェクトチャンクを受信する手段と、

前記少なくとも1つの更新されたオブジェクトチャンクを用いて前記ローカルデバイス上で前記第1のオブジェクトを更新する手段と

をさらに備えることを特徴とする請求項1に記載のシステム。

【請求項19】

前記第1のオブジェクトを更新する前記手段は、前記ローカルデバイス上で、前記少なくとも1つの更新されたオブジェクトチャンクを含む新しいオブジェクトを提供するように構成されていることを特徴とする請求項18に記載のシステム。

【請求項20】

前記ローカルデバイスを用いて前記ネットワークから前記少なくとも1つの更新されたオブジェクトチャンクを受信する手段と、

前記少なくとも1つの更新されたオブジェクトチャンクを用いて前記ローカルデバイス上で更新された第1のオブジェクトを組み立てる手段と

をさらに備えることを特徴とする請求項1に記載のシステム。

【請求項21】

前記更新された第1のオブジェクトを組み立てる前記手段は、さらに前記更新された第1のオブジェクトが、前記第1のオブジェクトからの少なくとも1つの変更されていないチャンクを含むように構成されることを特徴とする請求項20に記載のシステム。

【請求項22】

前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスに前記リモートのシグネチャおよびチャンク長のリストの前記チャンキングされた伝送をネゴシエーションする前記手段は、

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングしてチャンキングされたリモートのシグネチャおよびチャンク長のリストを提供する手段と、

前記リモートデバイス上で前記チャンキングされたリモートのシグネチャおよびチャンク長のリストに関連するチャンクごとに再帰的リモートシグネチャを計算する手段と、

前記再帰的リモートシグネチャと前記チャンキングされたリモートのシグネチャおよびチャンク長のリストを用いて前記リモートデバイス上で再帰的リモートシグネチャおよびチャンク長のリストを生成する手段と、

前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストをチャンキングする手段であって、前記ローカルのシグネチャおよびチャンク長のリストをチャンキングする前記手段が、前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段にマッチングさせられる手段と、

前記ローカルデバイス上で前記チャンキングされたローカルのシグネチャおよびチャンク長のリストに関連するチャンクごとに再帰的ローカルシグネチャを計算する手段であって、前記再帰的ローカルシグネチャを計算する前記手段が、前記再帰的リモートシグネチャを計算する前記手段にマッチングさせられる手段と、

前記再帰的ローカルシグネチャと、前記チャンキングされたローカルのシグネチャおよびチャンク長のリストとを用いて前記ローカルデバイス上で再帰的ローカルシグネチャおよびチャンク長のリストを生成する手段であって、前記再帰的ローカルシグネチャおよびチャンク長のリストを生成する前記手段が、前記再帰的リモートシグネチャおよびチャンク長のリストを生成する前記手段にマッチングさせられる手段と、

帯域幅の使用が、前記ローカルデバイスへの前記再帰的リモートシグネチャおよびチャンク長のリストの前記転送について最小になるように、前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスへの前記再帰的リモートシグネチャおよびチャンク長のリストの伝送をネゴシエーションする手段と、

前記ローカルデバイス上で前記再帰的リモートシグネチャおよびチャンク長のリストと前記再帰的ローカルシグネチャおよびチャンク長のリストとの間の差分を識別する手段と

10

20

30

40

50

、
前記再帰的リモートシグネチャおよびチャンク長のリストと前記再帰的ローカルシグネチャおよびチャンク長のリストとの間で前記ローカルデバイスによって差分が識別されるときに前記リモートデバイスからの少なくとも1つの更新されたシグネチャチャンクの伝送を要求する手段と、

前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスに前記少なくとも1つの更新されたシグネチャチャンクを伝送する手段であって、前記要求された少なくとも1つの更新されたシグネチャチャンクが、前記リモートのシグネチャおよびチャンク長のリストに関連する手段と、

前記少なくとも1つの更新されたシグネチャチャンクを用いて前記ローカルデバイス上で前記リモートのシグネチャおよびチャンク長のリストのコピーを組み立てる手段と
を備えることを特徴とする請求項1に記載のシステム。 10

【請求項23】

前記リモートデバイスから前記ローカルデバイスへの前記再帰的リモートシグネチャおよびチャンク長のリストの伝送をネゴシエーションする前記手段は、前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスに前記再帰的リモートシグネチャおよびチャンク長のリストの少なくとも一部分を送信するステップを含むことを特徴とする請求項22に記載のシステム。

【請求項24】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、 20

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストのバイトオフセットごとに第3のフィンガプリント関数を計算する手段と、

前記第3のフィンガプリント関数に基づいて前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングして前記チャンキングされたリモートのシグネチャおよびチャンク長のリストを提供する手段と
を備えることを特徴とする請求項22に記載のシステム。

【請求項25】

前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、 30

前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストのバイトオフセットごとに第4のフィンガプリント関数を計算する手段であって、前記第4のフィンガプリント関数は、前記第3のフィンガプリント関数にマッチングさせられる手段と

、
前記第4のフィンガプリント関数に基づいて前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストをチャンキングして前記チャンキングされたローカルのシグネチャおよびチャンク長のリストを提供する手段であって、前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストをチャンキングする前記手段が、前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段にマッチングさせられる手段と 40

を備えることを特徴とする請求項24に記載のシステム。

【請求項26】

前記リモートデバイス上での前記第3のフィンガプリント関数を計算する前記手段および前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、前記リモートデバイス上での前記第1のフィンガプリント関数を計算する前記手段および前記第1のオブジェクトをチャンキングする前記手段とは異なる方法を使用することを特徴とする請求項25に記載のシステム。

【請求項27】

前記リモートデバイス上での前記第3のフィンガプリント関数を計算する前記手段および前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、 50

前記リモートデバイス上での前記第1のフィンガプリント関数を計算する前記手段および前記第1のオブジェクトをチャンキングする前記手段と同じやり方を使用することを特徴とする請求項24に記載のシステム。

【請求項28】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストのバイトオフセットごとに前記第3のフィンガプリント関数を計算する前記手段は、

前記リモートのシグネチャおよびチャンク長のリストに関連する各バイト位置の周囲で参照されるスモールウィンドウを提供する手段と、

バイト位置ごとに前記スモールウィンドウを使用してフィンガプリントを生成する手段と

10

を備えることを特徴とする請求項24に記載のシステム。

【請求項29】

前記第1のオブジェクトに関連するデータタイプ、前記第2のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記ネットワークの特性と、前記第1のオブジェクトに関連する使用モデルと、前記第2のオブジェクトに関連する使用モデルのうちの少なくとも1つに基づいて、前記スモールウィンドウに関連するウィンドウサイズを調整する手段をさらに備えることを特徴とする請求項28に記載のシステム。

【請求項30】

前記第3のフィンガプリント関数は、ラビン多項式を使用したハッシュ関数、循環シフティングハッシュ関数、32ビットアドラーハッシュ関数のうちの少なくとも1つを含むことを特徴とする請求項24に記載のシステム。

20

【請求項31】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、少なくとも1つの再帰的チャンキングパラメータを決定する手段を備えることを特徴とする請求項22に記載のシステム。

【請求項32】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、

前記少なくとも1つの再帰的チャンキングパラメータから再帰的チャンキングホライズンを決定する手段と、

30

前記リモートのシグネチャおよびチャンク長のリスト内の各位置においてハッシュ値を計算する手段と、

前記リモートのシグネチャおよびチャンク長のリスト内の各位置の周囲の前記再帰的チャンキングホライズン内に位置づけられたハッシュ値に数学関数を適用する手段と、

前記数学関数が満たされるときに前記チャンキング境界中にカットポイントを指定する手段と、

前記指定されたカットポイントを用いて前記リモートのシグネチャおよびチャンク長のリストをチャンキングする手段と

をさらに備えることを特徴とする請求項31に記載のシステム。

40

【請求項33】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングする前記手段は、

前記少なくとも1つの再帰的チャンキングパラメータから再帰的ホライズン、再帰的トリガ値、および他の再帰的トリガのリストを決定する手段と、

前記リモートのシグネチャおよびチャンク長のリスト内の各位置においてハッシュ値を計算する手段と、

計算された各ハッシュ値に数学関数を適用する手段と、

前記数学関数が、所与のオフセットにおいて前記再帰的トリガ値を獲得し、前記再帰的ホライズンが提供するすべての対応するオフセットにおいて前記他の再帰的トリガを獲得

50

するときに、少なくとも1つのカットポイントおよびチャンキング境界を指定する手段と、

前記指定されたカットポイントを用いて前記リモートのシグネチャおよびチャンク長のリストをチャンキングする手段と

をさらに備えることを特徴とする請求項31に記載のシステム。

【請求項34】

前記数学関数は、ハッシュ値をブール値にマッピングする述語、およびハッシュ値を適切な小さなドメイン中に区分する他の任意の数学関数のうちの少なくとも1つを含むことを特徴とする請求項32に記載のシステム。

【請求項35】

前記数学関数は、前記ホライズン内の最大値の決定、前記ホライズン内の最小値の決定、前記ホライズン内のハッシュ値の間の差分の評価、前記ホライズン内のハッシュ値の合計、および前記ホライズン内のハッシュ値の平均値の計算のうちの少なくとも1つを含むことを特徴とする請求項32に記載のシステム。

【請求項36】

前記第1のオブジェクトに関連するデータタイプ、前記第2のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記ネットワークの特性と、前記第1のオブジェクトに関連する使用モデルと、前記第2のオブジェクトに関連する使用モデルのうちの少なくとも1つに基づいて、前記少なくとも1つの再帰的チャンキングパラメータを調整する手段をさらに備えることを特徴とする請求項33に記載のシステム。

【請求項37】

前記リモートデバイス上で前記チャンキングされたリモートのシグネチャおよびチャンク長のリストに関連するシグネチャチャンクごとに前記再帰的リモートシグネチャを計算する前記手段は、前記リモートデバイス上で前記シグネチャチャンクに対して適用される強化ハッシング関数を含むことを特徴とする請求項29に記載のシステム。

【請求項38】

前記リモートデバイス上で前記少なくとも1つの更新されたシグネチャチャンクの伝送を求める前記要求を受信する手段と、

前記少なくとも1つの更新されたシグネチャチャンクの伝送を求める前記受信された要求に応答して前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストから前記少なくとも1つの更新されたシグネチャチャンクを抽出する手段と、

前記リモートデバイスを用いて前記ネットワーク上で前記少なくとも1つの更新されたシグネチャチャンクを送信する手段と、

前記ローカルデバイスを用いて前記ネットワークから少なくとも1つの更新されたシグネチャチャンクを受信する手段と、

前記少なくとも1つの更新されたシグネチャチャンクを用いて前記ローカルデバイス上で前記リモートのシグネチャおよびチャンク長のリストのコピーを組み立てる手段と

をさらに備えることを特徴とする請求項22に記載のシステム。

【請求項39】

前記ローカルのシグネチャおよびチャンク長のリストを組み立てる前記手段は、前記ローカルデバイス上で、前記リモートのシグネチャおよびチャンク長のリストの新しいコピーを提供するように構成され、前記リモートのシグネチャおよびチャンク長のリストの前記新しいコピーは、前記少なくとも1つの更新されたシグネチャチャンクを含むことを特徴とする請求項38に記載のシステム。

【請求項40】

前記ローカルデバイスを用いて前記ネットワークから前記少なくとも1つの更新されたシグネチャチャンクを受信する手段と、

前記少なくとも1つの更新されたシグネチャチャンクを用いて前記ローカルデバイス上で前記リモートのシグネチャおよびチャンク長のリストのコピーを組み立てる手段と

10

20

30

40

50

をさらに備えることを特徴とする請求項 22 に記載のシステム。

【請求項 41】

前記リモートのシグネチャおよびチャンク長のリストの前記コピーを組み立てる前記手段は、前記リモートのシグネチャおよびチャンク長のリストの前記コピーが、前記ローカルのシグネチャおよびチャンク長のリストからの少なくとも 1 つの変更されていないチャンクを含むようにさらに構成されることを特徴とする請求項 38 に記載のシステム。

【請求項 42】

前記ローカルデバイス上で前記再帰的リモートシグネチャおよびチャンク長のリストと前記再帰的ローカルシグネチャおよびチャンク長のリストとの間の差分を識別する前記手段は、

10

前記再帰的リモートシグネチャおよびチャンク長のリストを前記再帰的ローカルシグネチャおよびチャンク長のリストと比較する手段と、

前記再帰的リモートシグネチャおよびチャンク長のリストと前記再帰的ローカルシグネチャおよびチャンク長のリストとの間の差分に関連する少なくとも 1 つのシグネチャチャンクを識別する手段と、

前記少なくとも 1 つのシグネチャチャンクを前記リモートのシグネチャおよびチャンク長のリストにマッピングする手段と、

前記少なくとも 1 つのシグネチャチャンクと前記リモートのシグネチャおよびチャンク長のリストの間の前記マッピングから前記少なくとも 1 つの更新されたシグネチャチャンクを識別する手段と

20

を備えることを特徴とする請求項 22 に記載のシステム。

【請求項 43】

前記ネットワーク上で前記リモートデバイスから前記ローカルデバイスへの前記リモートのシグネチャおよびチャンク長のリストのチャンキングされた伝送をネゴシエーションする前記手段は、

前記第 1 のオブジェクトに関連するデータサイズ、前記第 2 のオブジェクトに関連するデータサイズ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記ネットワークの特性と、前記第 1 のオブジェクトに関連する使用モデルと、前記第 2 のオブジェクトに関連する使用モデルと、前記第 1 のオブジェクトに関連するチャンクシグネチャの数、ならびに前記チャンキングされたりリモートのシグネチャおよびチャンク長のリストに関連するチャンクシグネチャの数のうちの少なくとも 1 つに基づいて、再帰的処理のための反復の回数を決定する手段を備えることを特徴とする請求項 1 に記載のシステム。

30

【請求項 44】

前記シグネチャおよびチャンク長のリストをチャンキングしてチャンキングされたシグネチャおよびチャンク長のリストを提供する手段と、

前記チャンキングされたシグネチャおよびチャンク長のリストに関連するチャンクごとに再帰的シグネチャを計算する手段と、

前記再帰的シグネチャと前記チャンキングされたシグネチャおよびチャンク長のリストとを用いて、再帰的なシグネチャおよびチャンク長のリストを生成する手段と、

40

追加の反復が再帰的な処理について必要とされるときに前記シグネチャおよびチャンク長のリストを前記再帰的なシグネチャおよびチャンク長のリストに初期化する手段と、

前記再帰的手続きが反復の回数を完了しているときに前記再帰的なシグネチャおよびチャンク長のリストを返す手段と

を備える、シグネチャおよびチャンク長のリストを処理するための再帰的手続きと、

前記リモートのシグネチャおよびチャンク長のリストを前記シグネチャおよびチャンク長のリストとして前記再帰的手続きに渡し、前記再帰的手続きから前記再帰的リモートシグネチャおよびチャンク長のリストを返すことにより、前記リモートデバイス上で前記再帰的手続きを用いて前記リモートのシグネチャおよびチャンク長のリストを処理する手段と、

50

前記ローカルのシグネチャおよびチャンク長のリストを前記シグネチャおよびチャンク長のリストとして前記再帰的手続きに渡し、前記再帰的手続きから前記再帰的ローカルシグネチャおよびチャンク長のリストを返すことにより、前記ローカルデバイス上で前記再帰的手続きを用いて前記ローカルのシグネチャおよびチャンク長のリストを処理する手段と

をさらに含むことを特徴とする請求項 4 3 に記載のシステム。

【請求項 4 5】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストを生成する前記手段は、さらに前記リモートのシグネチャおよびチャンク長のリストをコンパクトに符号化するように構成されることを特徴とする請求項 1 に記載のシステム。

10

【請求項 4 6】

前記ローカルデバイス上で前記ローカルのシグネチャおよびチャンク長のリストを生成する前記手段は、さらに前記ローカルのシグネチャおよびチャンク長のリストをコンパクトに符号化するように構成されることを特徴とする請求項 1 に記載のシステム。

【請求項 4 7】

前記リモートデバイス上で前記再帰的リモートシグネチャおよびチャンク長のリストを生成する前記手段は、さらに前記再帰的リモートシグネチャおよびチャンク長のリストをコンパクトに符号化するように構成されることを特徴とする請求項 2 2 に記載のシステム。

【請求項 4 8】

20

前記ローカルデバイス上で前記再帰的ローカルシグネチャおよびチャンク長のリストを生成する前記手段は、さらに前記再帰的ローカルシグネチャおよびチャンク長のリストをコンパクトに符号化するように構成されることを特徴とする請求項 2 2 に記載のシステム。

【請求項 4 9】

ローカルデバイスとリモートデバイスの間において通信媒体上でオブジェクトを更新するためのコンピュータ実行可能命令を有するコンピュータ読取り可能媒体であって、

前記リモートデバイス上で第 1 のオブジェクトをチャンキングするステップと、

前記リモートデバイス上で前記第 1 のオブジェクトに関連するチャンクごとにシグネチャを計算してリモートシグネチャを提供するステップと、

30

前記リモートシグネチャから前記リモートデバイス上でリモートのシグネチャおよびチャンク長のリストを組み立てるステップと、

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングし、

前記リモートデバイス上で前記チャンキングされたリモートのシグネチャおよびチャンク長のリストに関連するチャンクごとにシグネチャを計算して再帰的リモートシグネチャを提供し、

前記再帰的リモートシグネチャを用いて前記リモートデバイス上で再帰的リモートシグネチャおよびチャンク長のリストを組み立てることにより、

前記リモートデバイス上で再帰的リモートシグネチャおよびチャンク長のリストを生成するステップと、

40

前記ローカルデバイス上で第 2 のオブジェクトをチャンキングするステップと、

前記ローカルデバイス上で前記第 2 のオブジェクトに関連するチャンクごとにシグネチャを計算してローカルシグネチャを提供するステップと、

前記第 1 のオブジェクトが前記第 2 のオブジェクトにマッチングさせられるときに前記ローカルのシグネチャおよびチャンク長のリストが前記リモートのシグネチャおよびチャンク長のリストにマッチングさせられるように前記ローカルシグネチャから前記ローカルデバイス上でローカルのシグネチャおよびチャンク長のリストを組み立てるステップと、

前記ローカルのシグネチャおよびチャンク長のリストをチャンキングし、

前記チャンキングされたローカルのシグネチャおよびチャンク長のリストに関連する

50

チャンクごとにシグネチャを計算して再帰的ローカルシグネチャを提供し、

前記再帰的ローカルシグネチャを用いて再帰的ローカルシグネチャおよびチャンク長のリストを組み立てることにより、

前記ローカルデバイス上で再帰的ローカルシグネチャおよびチャンク長のリストを生成するステップと、

帯域幅の使用が、前記ローカルデバイスへの前記再帰的リモートシグネチャおよびチャンク長のリストの転送について最小になるように前記通信媒体上で前記リモートデバイスから前記ローカルデバイスへの前記再帰的リモートシグネチャおよびチャンク長のリストの伝送をネゴシエーションするステップと、

前記ローカルデバイス上で前記再帰的リモートシグネチャおよびチャンク長のリストと前記再帰的ローカルシグネチャおよびチャンク長のリストとを比較して差分を識別し、

前記第2のオブジェクトに関連する少なくとも1つのチャンクに前記差分をマッピングすることにより、

前記第1のオブジェクトと前記第2のオブジェクトとの間の少なくとも1つの差分を識別するステップと、

前記リモートデバイスからの少なくとも1つのチャンクの伝送を要求し、

前記通信媒体上で前記リモートデバイスから前記少なくとも1つのチャンクを含む伝送を受信し、

前記少なくとも1つのチャンクを用いてオブジェクトを組み立てることにより、

前記ローカルデバイス上で前記第1のオブジェクトを更新するステップと

を含むことを特徴とするコンピュータ読取り可能媒体。

【請求項50】

前記リモートデバイス上で前記第1のオブジェクトをチャンキングするステップは、前記第1のオブジェクトにフィンガプリント関数を適用して第1の組のフィンガプリントを生成するステップと、前記第1の組のフィンガプリントに基づいて前記第1のオブジェクトを第1の組のチャンクに区分するステップとを含むことを特徴とする請求項49に記載のコンピュータ読取り可能媒体。

【請求項51】

前記ローカルデバイス上で前記第2のオブジェクトをチャンキングするステップは、前記第2のオブジェクトに前記フィンガプリント関数を適用して第2の組のフィンガプリントを生成するステップと、前記第2の組のフィンガプリントに基づいて前記第2のオブジェクトを第2の組のチャンクに区分するステップとを含むことを特徴とする請求項50に記載のコンピュータ読取り可能媒体。

【請求項52】

前記通信媒体は、直接配線接続、パラレルポート、シリアルポート、USBポート、IEEE1394ポート、無線接続、IRポート、ブルートゥースポート、有線ネットワーク、無線ネットワーク、ローカルエリアネットワーク、ワイドエリアネットワーク、ウルトラワイドエリアネットワーク、インターネット、イントラネット、およびエクストラネットのうちの少なくとも1つであることを特徴とする請求項49に記載のコンピュータ読取り可能媒体。

【請求項53】

前記フィンガプリント関数は、前記第1のオブジェクトの各バイト位置の周囲で参照されるウィンドウを提供するステップと、前記ウィンドウ中に位置づけられる前記バイト値からハッシュを計算するステップとを含むことを特徴とする請求項50に記載のコンピュータ読取り可能媒体。

【請求項54】

前記第1のオブジェクトに関連するデータタイプ、前記第2のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記通信媒体の特性と、前記第1のオブジェクトに関連する使用モデルと、前記第2のオブジェクトに関連する使用モデルのうちの少なくとも1つに基づ

10

20

30

40

50

いて、前記ウィンドウに関連するウィンドウサイズを調整するステップをさらに含むことを特徴とする請求項 53 に記載のコンピュータ読取り可能媒体。

【請求項 55】

前記第 1 のオブジェクトと前記第 2 のオブジェクトとの間の少なくとも 1 つの差分を識別するステップは、

前記差分に基づいて前記リモートデバイス上で更新されたシグネチャチャンクを識別するステップと、

前記通信媒体上で前記リモートデバイスから前記ローカルデバイスへの前記更新されたシグネチャチャンクの伝送を要求するステップと、

前記通信媒体から前記ローカルデバイス上で前記更新されたシグネチャチャンクを受信するステップと、

前記更新されたシグネチャチャンクを用いて前記ローカルデバイス上で更新されたシグネチャおよびチャンク長のリストを組み立てるステップと

をさらに含むことを特徴とする請求項 49 に記載のコンピュータ読取り可能媒体。

【請求項 56】

前記第 2 のオブジェクトに関連する少なくとも 1 つのチャンクに前記差分をマッピングするステップは、前記更新されたシグネチャおよびチャンク長のリストを前記ローカルのシグネチャおよびチャンク長のリストと比較して前記リモートデバイス上で少なくとも 1 つの更新されたチャンクを識別するステップを含むことを特徴とする請求項 55 に記載のコンピュータ読取り可能媒体。

【請求項 57】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングするステップは、前記リモートのシグネチャおよびチャンク長のリストにフィンガプリント関数を適用して第 1 の組のフィンガプリントを生成するステップと、前記第 1 の組のフィンガプリントに基づいて前記リモートのシグネチャおよびチャンク長のリストを第 1 の組のチャンクに区分するステップとを含むことを特徴とする請求項 49 に記載のコンピュータ読取り可能媒体。

【請求項 58】

前記リモートデバイス上で前記ローカルのシグネチャおよびチャンク長のリストをチャンキングするステップは、前記ローカルのシグネチャおよびチャンク長のリストに前記フィンガプリント関数を適用して第 2 の組のフィンガプリントを生成するステップと、前記第 2 の組のフィンガプリントに基づいて前記ローカルのシグネチャおよびチャンク長のリストを第 2 の組のチャンクに区分するステップとを含むことを特徴とする請求項 57 に記載のコンピュータ読取り可能媒体。

【請求項 59】

前記フィンガプリント関数は、前記リモートのシグネチャおよびチャンク長のリストの各バイト位置の周囲で参照されるウィンドウを提供するステップと、前記ウィンドウ中に位置づけられる前記バイト値からハッシュ値を計算するステップとを含むことを特徴とする請求項 57 に記載のコンピュータ読取り可能媒体。

【請求項 60】

前記第 1 のオブジェクトに関連するデータタイプ、前記第 2 のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記通信媒体の特性と、前記第 1 のオブジェクトに関連する使用モデルと、前記第 2 のオブジェクトに関連する使用モデルのうちの少なくとも 1 つに基づいて、前記ウィンドウに関連するウィンドウサイズを調整するステップをさらに含むことを特徴とする請求項 59 に記載のコンピュータ読取り可能媒体。

【請求項 61】

前記リモートデバイス上で前記リモートのシグネチャおよびチャンク長のリストをチャンキングするステップは、

少なくとも 1 つの再帰的チャンキングパラメータを決定するステップと、

10

20

30

40

50

前記少なくとも1つの再帰的チャンキングパラメータから再帰的ホライズンおよび少なくとも1つの再帰的トリガ値のうちの少なくとも一方を決定するステップと、

前記リモートのシグネチャおよびチャンク長のリスト内の位置ごとにハッシュ値を計算するステップと、

計算された各ハッシュ値に数学関数を適用するステップと、

前記数学関数が、所与のオフセットにおいて前記少なくとも1つの再帰的トリガ値を獲得するときにチャンキング境界を指定するステップと、

前記指定済みのカットポイントを用いて前記リモートのシグネチャおよびチャンク長のリストをチャンキングするステップと

を含むことを特徴とする請求項60に記載のコンピュータ読取り可能媒体。

10

【請求項62】

前記数学関数は、ハッシュ値をブール値にマッピングする述語と、ハッシュ値を小さなドメイン中に区分する第1の関数と、前記ホライズン内の最大値を決定する第2の関数と、前記ホライズン内の最小値を決定する第3の関数と、前記ホライズン内のハッシュ値の間の差分を評価する第4の関数と、前記ホライズン内のハッシュ値を合計する第5の関数と、前記ホライズン内のハッシュ値の平均値を計算する第6の関数として構成されることを特徴とする請求項61に記載のコンピュータ読取り可能媒体。

【請求項63】

前記第1のオブジェクトに関連するデータタイプ、前記第2のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記通信媒体の特性と、前記第1のオブジェクトに関連する使用モデルと、前記第2のオブジェクトに関連する使用モデルのうちの少なくとも1つに基づいて、前記少なくとも1つの再帰的チャンキングパラメータを調整するステップをさらに含むことを特徴とする請求項61に記載のコンピュータ読取り可能媒体。

20

【請求項64】

ローカルデバイスとリモートデバイスとの間の通信チャンネル上でオブジェクトを更新するための、コンピュータによって実施される方法であって、

前記リモートデバイス上で第1のオブジェクトをチャンキングするステップと、

前記リモートデバイス上で前記第1のオブジェクトに関連するチャンクごとにシグネチャを計算してリモートシグネチャを提供するステップと、

30

前記リモートシグネチャから前記リモートデバイス上でリモートのシグネチャおよびチャンク長のリストを組み立てるステップと、

前記計算済みのフィンガプリント関数に基づいて前記ローカルデバイス上で第2のオブジェクトをチャンキングするステップと、

前記ローカルデバイス上で前記第2のオブジェクトに関連するチャンクごとにシグネチャを計算してローカルシグネチャを提供するステップと、

前記第1のオブジェクトが前記第2のオブジェクトにマッチングさせられるときに前記ローカルのシグネチャおよびチャンク長のリストが前記リモートのシグネチャおよびチャンク長のリストにマッチングさせられるように、前記ローカルシグネチャから前記ローカルデバイス上でローカルのシグネチャおよびチャンク長のリストを組み立てるステップと

40

、
前記指定されたシグネチャおよびチャンク長のリストをチャンキングしてチャンキングされたシグネチャおよびチャンク長のリストを提供し、

前記チャンキングされたシグネチャおよびチャンク長のリストに関連するチャンクごとに再帰的シグネチャを計算し、

前記再帰的シグネチャと前記チャンキングされたシグネチャおよびチャンク長のリストとを用いて再帰的なシグネチャおよびチャンク長のリストを生成し、

追加の反復が再帰的处理について必要とされるときに前記指定されたシグネチャおよびチャンク長のリストを前記再帰的なシグネチャおよびチャンク長のリストに初期化し、

前記再帰的手続きが、必要とされる反復の回数を完了しているときに、前記再帰的な

50

シグネチャおよびチャンク長のリストを返すことにより、

前記ローカルデバイス上でも前記リモートデバイス上でも共に、指定されたシグネチャおよびチャンク長のリストを処理するように構成された再帰的手続きを提供するステップと、

前記リモートのシグネチャおよびチャンク長のリストを前記指定されたシグネチャおよびチャンク長のリストとして前記再帰的手続きに渡し、前記再帰の手続きから前記再帰のリモートシグネチャおよびチャンク長のリストを返すことにより、前記リモートデバイス上で再帰のリモートシグネチャおよびチャンク長のリストを生成するステップと、

前記ローカルのシグネチャおよびチャンク長のリストを前記指定されたシグネチャおよびチャンク長のリストとして前記再帰の手続きに渡し、前記再帰の手続きから前記再帰のローカルシグネチャおよびチャンク長のリストを返すことにより、前記ローカルデバイス上で再帰的ローカルシグネチャおよびチャンク長のリストを生成するステップと、

前記通信チャネル上で前記リモートデバイスから前記ローカルデバイスに前記再帰のリモートシグネチャおよびチャンク長のリストを送信するステップと、

前記受信された再帰のリモートシグネチャおよびチャンク長のリストを前記再帰的ローカルシグネチャおよびチャンク長のリストと比較することにより、前記第1のオブジェクトと前記第2のオブジェクトの間の少なくとも1つの差分を識別するステップと、

前記少なくとも1つの差分に基づいて前記第2のオブジェクトに関連する少なくとも1つの更新されたチャンクを識別するステップと、

前記リモートデバイスから前記少なくとも1つの更新されたチャンクの伝送を要求し

、
前記通信チャネル上で前記リモートデバイスから前記少なくとも1つの更新されたチャンクを含む伝送を受信し、

前記少なくとも1つの更新されたチャンクを用いてオブジェクトを組み立てることにより、

前記ローカルデバイス上で前記第1のオブジェクトを更新するステップと
を含むことを特徴とする方法。

【請求項65】

前記リモートデバイス上で前記第1のオブジェクトをチャンキングするステップは、前記第1のオブジェクトにフィンガプリント関数を適用して第1の組のフィンガプリントを生成するステップと、前記第1の組のフィンガプリントに基づいて前記第1のオブジェクトを第1の組のチャンクに区分するステップとを含むことを特徴とする請求項64に記載のコンピュータによって実施される方法。

【請求項66】

前記ローカルデバイス上で前記第2のオブジェクトをチャンキングするステップは、前記第2のオブジェクトに前記フィンガプリント関数を適用して第2の組のフィンガプリントを生成するステップと、前記第2の組のフィンガプリントに基づいて前記第2のオブジェクトを第2の組のチャンクに区分するステップとを含むことを特徴とする請求項65に記載のコンピュータによって実施される方法。

【請求項67】

前記通信チャネルは、直接配線接続、パラレルポート、シリアルポート、USBポート、IEEE1394ポート、無線接続、IRポート、ブルートゥースポート、有線ネットワーク、無線ネットワーク、ローカルエリアネットワーク、ワイドエリアネットワーク、ウルトラワイドエリアネットワーク、インターネット、イントラネット、およびエクストラネットのうちの少なくとも1つであることを特徴とする請求項64に記載のコンピュータによって実施される方法。

【請求項68】

前記少なくとも1つの差分に基づいて前記第2のオブジェクトに関連する少なくとも1つのチャンクを識別するステップは、

前記再帰的ローカルシグネチャおよびチャンク長のリストとは異なる前記受信された再

10

20

30

40

50

帰的リモートシグネチャおよびチャンク長のリストの少なくとも1つの再帰的チャンクを識別するステップと、

前記リモートのシグネチャおよびチャンク長のリストの少なくとも1つのチャンクに前記少なくとも1つの再帰的チャンクをマッピングするステップと、

前記リモートデバイスから前記リモートのシグネチャおよびチャンク長のリストの前記少なくとも1つのチャンクの伝送を要求するステップと、

前記通信チャネル上で前記リモートデバイスから前記リモートのシグネチャおよびチャンク長のリストの前記少なくとも1つのチャンクを含む伝送を受信するステップと、

前記リモートのシグネチャおよびチャンク長のリストの前記受信された少なくとも1つのチャンクから、更新されたシグネチャおよびチャンク長のリストを組み立てるステップと

を含むことを特徴とする請求項64に記載のコンピュータによって実施される方法。

【請求項69】

前記少なくとも1つの差分に基づいて前記第2のオブジェクトに関連する少なくとも1つのチャンクを識別するステップは、前記更新されたシグネチャおよびチャンク長のリストを前記ローカルのシグネチャおよびチャンク長のリストと比較して、前記リモートデバイス上で前記少なくとも1つの更新されたチャンクを識別するステップを含むことを特徴とする請求項68に記載のコンピュータによって実施される方法。

【請求項70】

前記指定されたシグネチャおよびチャンク長のリストをチャンキングするステップは、前記指定されたシグネチャおよびチャンク長のリストにフィンガプリント関数を適用して1組のフィンガプリントを生成するステップと、前記1組のフィンガプリントに基づいて前記指定されたシグネチャおよびチャンク長のリストを1組のチャンクに区分するステップとを含むことを特徴とする請求項64に記載のコンピュータによって実施される方法。

【請求項71】

前記フィンガプリント関数は、前記指定されたシグネチャおよびチャンク長のリストに関連する各バイト位置の周囲で参照されるウィンドウを提供するステップと、前記ウィンドウ中に位置づけられる前記バイト値からハッシュ値を計算するステップとを含むことを特徴とする請求項70に記載のコンピュータによって実施される方法。

【請求項72】

前記第1のオブジェクトに関連するデータタイプ、前記第2のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記通信チャネルの特性と、前記第1のオブジェクトに関連する使用モデルと、前記第2のオブジェクトに関連する使用モデルのうちの少なくとも1つに基づいて、前記ウィンドウに関連するウィンドウサイズを調整するステップをさらに含むことを特徴とする請求項71に記載のコンピュータによって実施される方法。

【請求項73】

前記指定されたシグネチャおよびチャンク長のリストをチャンキングして前記チャンキングされたシグネチャおよびチャンク長のリストを提供するステップは、

少なくとも1つの再帰的チャンキングパラメータを決定するステップと、

前記少なくとも1つの再帰的チャンキングパラメータから再帰的ホライズン、再帰的トリガ値、および再帰的トリガのリストのうちの少なくとも1つを決定するステップと、

前記指定されたシグネチャおよびチャンク長のリスト内の位置ごとにハッシュ値を計算するステップと、

指定されたシグネチャおよびチャンク長のリスト内の各位置の周囲の前記チャンキングホライズン内に位置づけられたハッシュ値に数学関数を適用するステップと、

前記数学関数が、満たされるときにカットポイントおよびチャンキング境界のうちの少なくとも1つを指定するステップと、

前記指定されたカットポイントを用いて前記指定されたシグネチャおよびチャンク長のリストをチャンキングするステップと

を含むことを特徴とする請求項 6 4 に記載のコンピュータによって実施される方法。

【請求項 7 4】

前記数学関数は、ハッシュ値をブール値にマッピングする述語と、ハッシュ値を小さなドメイン中に区分する第 1 の関数と、前記ホライズン内の最大値を決定する第 2 の関数と、前記ホライズン内の最小値を決定する第 3 の関数と、前記ホライズン内のハッシュ値の間の差分を評価する第 4 の関数と、前記ホライズン内のハッシュ値を合計する第 5 の関数と、前記ホライズン内のハッシュ値の平均値を計算する第 6 の関数として構成されることを特徴とする請求項 7 3 に記載のコンピュータによって実施される方法。

【請求項 7 5】

前記第 1 のオブジェクトに関連するデータタイプ、前記第 2 のオブジェクトに関連するデータタイプ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記通信媒体の特性と、前記第 1 のオブジェクトに関連する使用モデルと、前記第 2 のオブジェクトに関連する使用モデルのうちの少なくとも 1 つに基づいて、前記少なくとも 1 つの再帰的チャンキングパラメータを調整するステップをさらに含むことを特徴とする請求項 7 3 に記載のコンピュータによって実施される方法。 10

【請求項 7 6】

前記第 1 のオブジェクトに関連するデータサイズ、前記第 2 のオブジェクトに関連するデータサイズ、前記リモートデバイスに関連する環境制約、および前記ローカルデバイスに関連する環境制約と、前記通信チャネルの特性と、前記第 1 のオブジェクトに関連する使用モデルと、前記第 2 のオブジェクトに関連する使用モデルと、前記第 1 のオブジェクトに関連するチャンクシグネチャの数、ならびに前記チャンキングされたりリモートのシグネチャおよびチャンク長のリストに関連するチャンクシグネチャの数のうちの少なくとも 1 つに基づいて、再帰的处理のために必要とされる反復回数を決定するステップをさらに含むことを特徴とする請求項 6 4 に記載のコンピュータによって実施される方法。 20

【請求項 7 7】

再帰的处理のために必要とされる反復回数が、少なくとも 1 回の反復に対応することを特徴とする請求項 6 4 に記載のコンピュータによって実施される方法。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、一般に帯域幅の限られたネットワーク上でデータオブジェクトを更新することに関する。より詳細には、本発明は、RDC (remote differential compression 遠隔差分圧縮) 手法を使用した、オブジェクトデータの差分転送用のシステムと方法に関する。この RDC 手法の再帰的適用を利用して大きなオブジェクトの転送のための帯域幅の使用率をさらに最小化することができる。 30

【背景技術】

【0002】

イントラネット、エクストラネット、インターネットなどのネットワークの普及によって、広範なネットワーク上で情報を共有するユーザ数が増大した。最大データ転送速度は、その伝送媒体や他のインフラストラクチャ関連の制限に関連する帯域幅に基づいて各物理ネットワークに関連づけられる。ネットワークの帯域幅が制限される結果として、ユーザは、ネットワーク上で大量のデータを検索し転送する際に長い遅延を経験する可能性がある。 40

【0003】

データ圧縮技法は、帯域幅の限られたネットワークをわたって大量のデータを転送するための評判のよい方法になっている。データ圧縮は、一般に可逆的または非可逆的なもののいずれかとして特徴づけることができる。可逆的圧縮は、伸張変換を適用することにより、そのデータセットを正確に再現して取り出すことができるようにデータセットの変換を行うものである。正確な再現が必要とされるときには、可逆的圧縮が最もしばしば使用されてデータが圧縮される。 50

【0004】

データオブジェクトのこの受取人がそのオブジェクトの以前の、またはより古いバージョンをすでに所有している場合には、遠隔差分圧縮（RDC）と呼ばれる可逆的圧縮手法を使用してそのオブジェクトの新しいバージョンと古いバージョンとの間の差分を決定し、この差分だけを転送することができる。RDC転送は、その新しいバージョンと古いバージョンの間のその観測された差分（例えば、ファイルの場合には、ファイルの修正もしくは最後のアクセス日付、ファイルの属性、またはそのファイルコンテンツに対するわずかな変更）を伝えるものにすぎないので、転送されるデータの全体の量を非常に減少させることができる。RDCを別の可逆的圧縮アルゴリズムと結合してこのネットワークトラフィックをさらに減少させることができる。RDCの利点は、コンピューティングデバイス間で大きなオブジェクトを頻繁に送受する必要がある、これらのオブジェクトの以前の複製を保持することが、困難または実行不可能であり、その結果局所差分アルゴリズムを使用することができないという場合に最も顕著になる。

10

【発明の開示】

【発明が解決しようとする課題】

【0005】

簡潔に述べると、本発明は、限られた帯域幅のネットワーク上でオブジェクトを更新するための方法およびシステムに関する。

【課題を解決するための手段】

【0006】

オブジェクトは、必要なデータ転送が最小化されるように遠隔差分圧縮（RDC）技法を使用して2つ以上のコンピューティングデバイス間で更新される。一態様においては、効率的な大きなオブジェクトの転送が、それ自体のメタデータに対してこのRDCアルゴリズムを再帰的に適用することによって達成される。すなわち、1つまたは複数の再帰的ステップをこのケースにおいて使用してこのRDCアルゴリズムによってこのネットワーク上で送信されるメタデータの量を減少させることができる。オブジェクトおよび／またはシグネチャ、ならびにチャンク長リストを、動的に決定されたロケーションに境界を位置づけることによってチャンキングすることができる。数学関数が、可能性のあるチャンク境界に関連してホライズン（horizon）ウィンドウ内に関連づけられたハッシュ値（hash value）を評価する。ここで説明する方法およびシステムは、ピアツーピアレプリケータ（peer-to-peer replicator）、電子メールクライアントおよび電子メールサーバ、クライアント側キャッシングシステム、汎用コピーユーティリティ、データベースレプリケータ、ポータル、ソフトウェア更新サービス、ファイル／データ同期化など様々なネットワーク化されたアプリケーションにおいて有用である。

20

30

【0007】

本発明およびその改良についてのより完全な理解は、本発明の例示の実施形態の以下の詳細なる説明、および添付特許請求の範囲に対して、以下に簡単に概説した添付図面を参照することによって得ることが可能である。

【0008】

本発明の非限定的かつ非網羅的な実施形態について、以下の図面を参照して説明する。

40

【発明を実施するための最良の形態】

【0009】

図面を参照して本発明の様々な実施形態を詳細に説明することにする。いくつかの図面にわたって、同様な参照番号は、同様な部品およびアセンブリを表している。様々な実施形態に対する言及が、本発明の範囲を限定することではなく、この本発明の範囲は、本明細書に添付された特許請求の範囲によってしか限定されない。さらに、本明細書中に記載のどの実施例も限定することを意図したものではなく、単にこの特許請求されている本発明についての多数の可能な実施形態のうちの一部を述べているにすぎない。

【0010】

50

本発明は、1つまたは複数の一般的に関連するオブジェクトが記憶されているローカルコンピューティングデバイスおよびリモートコンピューティングデバイス（または、略して「デバイス」）の文脈で説明される。用語「ローカル」および「リモート」は、この方法の1つのインスタンスを示している。しかし、この同じデバイスが、異なるインスタンスにおいて、「ローカル」の役割および「リモート」の役割を共に果たすこともできる。遠隔差分圧縮（RDC）手法を使用して、帯域幅の制限されたネットワーク上で一般的に関連するオブジェクトを効率的に更新することができる。オブジェクトの新しいコピーを有するデバイスが、この同じオブジェクトの、または同様なオブジェクトのより古いコピーを有するデバイスを更新する必要があるとき、このRDC手法を使用してこのネットワーク上でこれらのオブジェクト間の差分だけが伝送される。一実施例として説明するRDC手法では、（1）RDCメタデータを伝送するために再帰的な手法を使用して大きなオブジェクトについて伝送されるメタデータの量が減少させられ、（2）極大ベースのチャンキング手法を使用して、帯域幅の利用が最小化されるようにこのオブジェクトの差分に関連する精度が増大させられる。この説明するRDC手法からの恩恵を受ける一部の実施例のアプリケーションには、ただいくつかの名前を挙げると、ピアツーピア複製サービス、SMBなどのファイル転送プロトコル、大きなイメージを転送する仮想サーバ、電子メールサーバ、セルラ電話とPDAの同期化、データベースサーバの複製が含まれる。

【0011】

動作環境

図1は、本発明についての動作環境の一実施例を示す図である。この図に示すように、複数のデバイスが、ネットワーク上で情報をやりとりするように配置されている。これらのデバイスは、ネットワークに接続された汎用コンピューティングデバイス、専用コンピューティングデバイス、または他の任意の適切なデバイスでもよい。このネットワーク102は、それだけには限定されないが、直接配線接続（例えば、パラレルポート、シリアルポート、USB、IEEE1394など）、無線接続（例えば、IRポート、ブルートゥース（Bluetooth）ポートなど）、有線ネットワーク、無線ネットワーク、ローカルエリアネットワーク、ワイドエリアネットワーク、ウルトラワイドエリアネットワーク、インターネット、イントラネット、およびエクストラネットを含めて任意の接続トポロジに対応したものとすることができる。

【0012】

デバイスA（100）とデバイスB（101）の間の対話の一実施例においては、あるオブジェクトの異なるバージョンが、2つのデバイス上に、すなわちオブジェクトO_Aが100上に、オブジェクトO_Bが101上にローカルに記憶されている。あるポイントにおいて、デバイスA（100）は、オブジェクトO_AのそのコピーをデバイスB（101）上に記憶されたコピー（オブジェクトO_B）で更新しようと決定し、デバイスB（101）に要求を送信してRDC手法を開始する。代替実施形態においては、このRDC手法をデバイスB（101）が開始することができる。

【0013】

デバイスA（100）とデバイスB（101）は、共にそれらのローカルに記憶されたオブジェクトを処理し、データに依存するようにしてその関連するデータを可変数のチャンク（例えば、それぞれ、オブジェクトO_Bについてのチャンク1～n、またオブジェクトO_Aについてのチャンク1～k）に分割する。これらのチャンクについてのストロングハッシュ（strong hash）（SHA）など、1組のシグネチャが、両方のデバイスによってローカルに計算される。これらのデバイスは共に、シグネチャの別々のリストをコンパイルする。このRDC手法の次のステップ中に、デバイスB（101）は、このネットワーク102上で、シグネチャとチャンク長1～nのその計算済みのリストをデバイスA（100）に送信する。デバイスA（100）は、受信された各シグネチャをそれ自体の生成済みのシグネチャリスト1～kと比較することによってこのシグネチャリストを評価する。このシグネチャリストにおけるミスマッチは、訂正を要求するこれらオブジェクトにおける1つまたは複数の差分を示す。デバイスA（100）は、このシグネチ

ャリスト中のミスマッチによって識別されたチャンクをデバイスB(101)が送信するように求める要求を送信する。その後、デバイスB(101)は、この要求されたチャンクを圧縮し送信し、次いでこれらのチャンクは、受信と伸張が実現された後にデバイスA(100)によって再組立てされる。デバイスA(100)は、この受信済みのチャンクをそれ自体のマッチングチャンクと一緒に再組立てして、オブジェクトO_Bのローカルコピーを得る。

【0014】

コンピューティングデバイス例

図2は、本発明に従って構成されたコンピューティングデバイスの一実施例のブロック図である。基本的な構成においては、コンピューティングデバイス200は、一般に少なくとも1つの処理装置(202)およびシステムメモリ(204)を含んでいる。コンピューティングデバイスのその厳密な構成およびタイプに応じて、システムメモリ204は、(RAMなどの)揮発性、(ROM、フラッシュメモリなどの)不揮発性、またはこれら2つの何らかの組合せとすることができる。システムメモリ204は、一般にオペレーティングシステム(205)、1つまたは複数のプログラムモジュール(206)を含んでおり、またプログラムデータ(207)を含むこともある。この基本的な構成が、図2の破線208内のこれらのコンポーネントによって示されている。

【0015】

コンピューティングデバイス200はまた、追加の特徴または機能を有することができる。例えば、コンピューティングデバイス200は、例えば、磁気ディスク、光ディスク、テープなど(着脱可能および/または着脱不能な)追加のデータストレージデバイスも含むことができる。かかる追加のストレージは、図2において着脱可能なストレージ209および着脱不能なストレージ210によって示されている。コンピュータストレージ媒体は、コンピュータ読取り可能命令、データ構造、プログラムモジュール、他のデータなどの情報を記憶するための任意の方法または技術で実装された揮発性および不揮発性の着脱可能および着脱不能な媒体を含むことができる。システムメモリ204、着脱可能なストレージ209、および着脱不能なストレージ210は、すべてコンピュータストレージ媒体の実施例である。コンピュータストレージ媒体には、それだけには限定されないが、RAM、ROM、EEPROM、フラッシュメモリまたは他のメモリ技術、CD-ROM、DVD(digital versatile disk デジタル多用途ディスク)または他の光ストレージ、磁気カセット、磁気テープ、磁気ディスクストレージまたは他の磁気ストレージデバイス、あるいは所望の情報を記憶するために使用することができ、コンピューティングデバイス200によってアクセスすることが可能な他の任意の媒体が含まれる。かかるコンピュータストレージ媒体はどれも、デバイス200の一部とすることができる。コンピューティングデバイス200はまた、キーボード、マウス、ペン、ボイス入力デバイス、タッチ入力デバイスなどの1つ(または複数)の入力デバイス212を有することもできる。ディスプレイ、スピーカ、プリンタなどの1つ(または複数)の出力デバイス214もまた含めることができる。すべてのこれらのデバイスは、当技術分野において知られており、ここで長く論ずる必要はない。

【0016】

コンピューティングデバイス200はまた、このデバイスがネットワーク上などで他のコンピューティングデバイス218と情報をやりとりすることができるようにする1つ(または複数)の通信接続216を含んでもいる。1つ(または複数)の通信接続216は、通信媒体の一実施例である。通信媒体は一般に、搬送波や他の搬送メカニズムなどの被変調データ信号の形でコンピュータ読取り可能命令、データ構造、プログラムモジュールまたは他のデータを実施しており、任意の情報配信媒体を含んでいる。この用語「被変調データ信号」とは、信号に情報を符号化するように1つまたは複数のその特性が設定され変更された信号を意味する。一例として、限定するものではないが、通信媒体には、有線ネットワークや直接配線接続などの有線媒体、および音響、RF、マイクロ波、衛星、赤外線、他の無線媒体などの無線媒体が含まれる。本明細書中で使用される用語、コンピュ

ータ読取り可能媒体は、ストレージ媒体も通信媒体も共に含んでいる。

【0017】

システムメモリ204中に存在する1つまたは複数のアプリケーションプログラム中で、様々なプロシージャおよびインターフェースを実装することができる。一実施例においては、このアプリケーションプログラムは、このコンピューティングデバイス（例えば、クライアント）と別のリモートに位置づけられたコンピューティングデバイス（例えば、サーバ）との間でファイルの同期化をスケジュールする遠隔差分圧縮アルゴリズムである。別の実施例においては、このアプリケーションプログラムは、データを圧縮し伸張するための、システムメモリ204中で提供される圧縮／伸張プロシージャである。さらに他の実施例においては、このアプリケーションプログラムは、クライアントデバイスのシステムメモリ204中において提供される暗号解読プロシージャである。 10

【0018】

遠隔差分圧縮（RDC）

図3Aおよび3Bは、本発明の少なくとも一態様によるRDCプロシージャの実施例を示す図である。特にチャンク数は、この実際のオブジェクトO_AおよびO_Bに応じてインスタンスごとに変化することが可能である。

【0019】

図3Aを参照すると、この基本的なRDCプロトコルは、2つのコンピューティングデバイス（デバイスAとデバイスB）の間でネゴシエーションが行われる。このRDCプロトコルでは暗黙に、これらのデバイスAおよびBが、それぞれオブジェクトインスタンス（またはバージョン）O_AおよびO_Bによって識別される同じオブジェクトまたはリソースの2つの異なるインスタンス（またはバージョン）を有するという想定が行われる。この図に示す実施例では、デバイスAは、リソースの古いバージョンO_Aを有するが、デバイスBは、そのリソースに関連したコンテンツ（またはデータ）がわずかに（または増分的に）異なるバージョンO_Bを有する。 20

【0020】

デバイスBからデバイスAにこの更新されたオブジェクトO_Bを転送するためのプロトコルについて以下に説明する。同様なプロトコルを使用してあるオブジェクトをデバイスAからデバイスBに転送することができ、この転送を以下に示すプロトコルをあまり変更せずにデバイスAまたはデバイスBのいずれかの命令で開始することができる。 30

【0021】

1. RDCプロトコルを使用して、デバイスAは、デバイスBにオブジェクトO_Bを転送する要求を送信する。代替実施形態においては、デバイスBが、この転送を開始する。この場合には、このプロトコルは、ステップ1をスキップし、以下のステップ2から開始される。

【0022】

2. デバイスAは、オブジェクトO_Aをチャンク1～kに区分し、オブジェクトO_Aの各チャンク1...kごとにシグネチャS_{ig_A_i}および長さ（またはバイトサイズ）L_{en_A_i}を計算する。このチャンクへの区分について、以下に詳細に説明することにする。デバイスAは、このシグネチャおよびチャンク長のリスト（（S_{ig_A₁}, L_{en_A₁}）...（S_{ig_A_k}, L_{en_A_k}））を記憶する。 40

【0023】

3. デバイスBは、オブジェクトO_Bをチャンク1～nに区分し、オブジェクトO_Bの各チャンク1...nごとにシグネチャS_{ig_B_i}および長さL_{en_B_i}を計算する。ステップ3で使用されるこの区分アルゴリズムは、以上のステップ2のアルゴリズムとマッチする必要がある。

【0024】

4. デバイスBは、オブジェクトO_Bに関連するその計算済みのチャンクシグネチャおよびチャンク長のリスト（（S_{ig_B₁}, L_{en_B₁}）...（S_{ig_B_n}, L_{en_B_n}））をデバイスAに送信する。その後、このチャンク長情報をデバイスAが使用してそれ 50

らの開始オフセットおよびそれらの長さを用いてそれらを識別することによって特定の1組のチャンクを要求する。このリストの順次的性質のために、このリスト中のすべての先行するチャンクの長さを合計することによって各チャンク B_i のバイト中の開始オフセットを計算することが可能である。

【0025】

別の実施形態においては、チャンクシグネチャおよびチャンク長のリストは、デバイスAに送信する前に、可逆的圧縮アルゴリズムを使用してコンパクトに符号化され、さらに圧縮される。

【0026】

5. このデータを受信すると、デバイスAは、そのコンテンツの古いバージョンに関連する、デバイスAがステップ2でオブジェクト O_A について計算したそれらのシグネチャ $Sig_{A_1} \dots Sig_{A_k}$ に対してこの受信したシグネチャリストを比較する。 10

【0027】

6. デバイスAは、ステップ4においてデバイスBから受信したシグネチャが、ステップ2でデバイスAが計算したシグネチャのどれかとマッチングに失敗したすべてのチャンクについてデバイスBに対して要求を送信する。要求されたチャンク B_i ごとにこの要求は、ステップ4でデバイスAが計算したチャンク開始オフセットとそのチャンク長とを含む。

【0028】

7. デバイスBは、すべての要求されたチャンクに関連するそのコンテンツをデバイスAに対して送信する。デバイスBが送信するこのコンテンツは、デバイスAに送信する前に可逆的圧縮アルゴリズムを使用してさらに圧縮することができる。 20

【0029】

8. デバイスAは、ステップ7でデバイスBから受信したチャンクを使用することによってオブジェクト O_B のローカルコピー、ならびにステップ4でデバイスBが送信したシグネチャとマッチングしたオブジェクト O_A のそれ自体のチャンクを再構築する。このローカルチャンクとリモートチャンクがデバイスA上で再配置される順序は、ステップ4でデバイスAが受信したチャンクシグネチャのリストによって決定される。

【0030】

区分ステップ2および3は、関連するオブジェクト（それぞれ、 O_A および O_B ）中のあらゆるバイト位置ごとに計算されるフィンガプリント関数（`fingerprinting function`）を使用するデータに依存する方法で行うことができる。所与の位置では、このフィンガプリント関数は、このオブジェクト中のバイト位置を囲む小さなデータウィンドウを使用して計算される。したがって、このフィンガプリント（指紋）関数の値は、このウィンドウ内に含まれるオブジェクトのすべてのバイトに依存する。このフィンガプリント関数は、例えばハッシュ関数（`hash function`）やラビン多項式（`Rabin polynomial`）など適切な任意の関数とすることができる。 30

【0031】

チャンク境界は、選択された条件を満たす値になるまでこのフィンガプリント関数が計算するそのオブジェクト中の位置において決定される。このチャンクシグネチャは、暗号によるセキュアハッシュ関数（`secure hash function`）（SHA）、または衝突耐久性ハッシュ関数（`collision-resistant hash function`）など、他の何らかのハッシュ関数を使用して計算することができる。 40

【0032】

ステップ4で送信されるこのシグネチャおよびチャンク長のリストは、この最初のチャンクとこの識別済みの更新された、または新しいチャンクの両方を使用してこのオブジェクトを再構築するための基礎を提供する。ステップ6で要求されるチャンクは、そのオフセットおよび長さによって識別される。このオブジェクトは、ステップ4でデバイスAが 50

受信したシグネチャとそのシグネチャが同じ順序でマッチングするローカルチャンクおよびリモートチャンクを使用することによってデバイスA上で再構築される。

【0033】

この再構築ステップがデバイスAによって完了された後に、オブジェクト O_A は、削除し、デバイスA上で再構築済みのオブジェクト O_B のコピーで置き換えることが可能である。他の実施形態においては、デバイスAは、将来のRDC転送中にチャンクを「再使用」する可能性のためにオブジェクト O_A をそのまま保持しておくこともできる。

【0034】

大きなオブジェクトでは、例えオブジェクト O_A およびオブジェクト O_B が、非常に近い、または同一であったとしても、図3Aに示す基本的なRDCプロトコルインスタンスが、ステップ4においてかなりの固定オーバーヘッドを招く。平均のチャンクサイズCを想定すると、ステップ4においてこのネットワーク上で伝送される情報量は、オブジェクト O_B のサイズに比例し、特にこの情報量は、オブジェクトB、したがってステップ4で伝送される（チャンクシグネチャ、チャンク長）対のチャンク数となる、Cで分割されたオブジェクト O_B のサイズに比例する。

【0035】

例えば、図6を参照すると、大きなイメージ（例えば、マイクロソフト仮想サーバ（Microsoft Virtual Server）などの仮想マシンモニタが使用する仮想ハードディスクイメージ）は、9.1GBのサイズのオブジェクト（ O_B ）をもたらし得る。3KBに等しい平均チャンクサイズCでは、この9GBのオブジェクトは、ステップ4でそのネットワーク上で送信する必要がある42MBの関連するシグネチャおよびチャンク長の情報を伴って、オブジェクト O_B について生成される3百万個のチャンクをもたらすこともある。例えオブジェクト O_A とオブジェクト O_B の間の差分（および、したがってステップ7で送信する必要があるデータ量）が非常に小さいときでも、この42MBのシグネチャ情報をそのネットワーク上で送信する必要があるので、このプロトコルの固定オーバーヘッドコストは、過剰に高いものになる。

【0036】

この固定オーバーヘッドコストは、ステップ4におけるシグネチャ情報転送の代わりにこのRDCプロトコルの再帰的適用を利用することによってかなり低減することができる。図3Bを参照して、その基本RDCアルゴリズムのステップ4を置き換える追加のステップ4.2～4.8について、以下のように説明する。ステップ4.2～4.8は、前述の基本的なRDCプロトコルのステップ2～8の再帰的適用に対応する。この再帰的適用については、以下のステップ4.4に、また所望の任意の再帰性の深さまでなど、さらに適用することができる。

【0037】

4.2. デバイスAは、そのシグネチャおよびチャンク長のリスト（（ Sig_{A1} , Len_{A1} ）...（ Sig_{Ak} , Len_{Ak} ））の再帰的なシグネチャチャンクへの再帰的チャンキングを実施し、再帰的シグネチャおよび再帰的チャンク長の別のリスト（（ $RSig_{A1}$, $RLen_{A1}$ ）...（ $RSig_{As}$, $RLen_{As}$ ））、を得る（ここで $s < k$ ）。

【0038】

4.3. デバイスBは、シグネチャおよびチャンク長のリスト（（ Sig_{B1} , Len_{B1} ）...（ Sig_{Bn} , Len_{Bn} ））を再帰的にチャンキングして再帰的シグネチャおよび再帰的チャンク長のリスト（（ $RSig_{B1}$, $RLen_{B1}$ ）...（ $RSig_{Br}$, $RLen_{Br}$ ））を生成する（ここで $r < n$ ）。

【0039】

4.4. デバイスBは、再帰的シグネチャおよび再帰的チャンク長の順序づけされたリスト（（ $RSig_{B1}$, $RLen_{B1}$ ）...（ $RSig_{Br}$, $RLen_{Br}$ ））をデバイスAに送信する。この再帰的チャンクシグネチャおよび再帰的チャンク長のリストは、デバイスAに送信する前に、コンパクトに符号化され、可逆的圧縮アルゴリズムを使用し

10

20

30

40

50

てさらに圧縮することができる。

【0040】

4. 5. デバイスAは、ステップ4. 2で計算した再帰的シグネチャのそれ自体のリストとデバイスBから受信したこの再帰的シグネチャを比較する。

【0041】

4. 6. デバイスAは、デバイスAがそのセット ($RSig_{A1} \dots RSig_{As}$) 中にマッチングする再帰的シグネチャをもたないあらゆる別個の (再帰的シグネチャ $RSig_{Bk}$ を有する) 再帰的シグネチャチャンクを求める要求をデバイスBに送信する。

【0042】

4. 7. デバイスBは、この要求された再帰的なシグネチャチャンクをデバイスAに送信する。この要求された再帰的シグネチャチャンクは、デバイスAに送信する前に、可逆的圧縮アルゴリズムを使用してさらに圧縮することができる。 10

【0043】

4. 8. デバイスAは、このローカルにマッチングした再帰的シグネチャチャンク、およびステップ4. 7でデバイスBから受信したこれらの再帰的チャンクを使用してシグネチャおよびチャンク情報のリスト ($(Sig_{B1}, Len_{B1}) \dots (Sig_{Bn}, Len_{Bn})$) を再構築する。

【0044】

以上のステップ4. 8が完了した後に、前述のその基本的なRDCプロトコルのステップ5において実行が続行され、それについては図3Aに示されている。 20

【0045】

再帰的チャンキング動作の結果、そのオブジェクトに関連する再帰的シグネチャの数は、その平均チャンクサイズCに等しいファクタだけ低減され、再帰的シグネチャのかなり小さな数をもたらされる (それぞれ、オブジェクト O_A については $r \ll n$ 、オブジェクト O_B については $s \ll k$ となる)。一実施形態においては、最初のオブジェクト O_A および O_B をチャンキングするためのものと同じチャンキングパラメータを、これらのシグネチャをチャンキングするために使用することができる。代替実施形態においては、他のチャンキングパラメータを、再帰的ステップのために使用することができる。

【0046】

非常に大きなオブジェクトでは、以上の再帰的ステップは、 $k \geq 1$ となるk回にわたって適用することができる。Cの平均チャンクサイズでは、再帰的チャンキングにより、このネットワーク (ステップ4. 2ないし4. 8) 上でおよそ C^k に対応するファクタだけこのシグネチャトラフィックのサイズを低減することができる。Cは、比較的大きいので、1よりも大きな再帰性深さは、非常に大きなオブジェクトについてのみ必要になり得る。 30

【0047】

一実施形態においては、再帰的ステップ数は、以下の予想された平均チャンクサイズ、オブジェクト O_A および/または O_B のサイズ、オブジェクト O_A および/または O_B のデータフォーマット、デバイスAとデバイスBを接続するネットワークの待ち時間特性および帯域幅特性のうちの1つまたは複数を含むパラメータを考慮することによって動的に決定することができる。 40

【0048】

ステップ2で使用されるフィンガプリント関数は、ステップ3で使用されるフィンガプリント関数にマッチングさせられる。同様に、ステップ4. 2で使用されるフィンガプリント関数は、ステップ4. 3で使用されるフィンガプリント関数にマッチングさせられる。ステップ2～3からのフィンガプリント関数は、ステップ4. 2～4. 3からのフィンガプリント関数にオプションとしてマッチングさせることができる。

【0049】

以前に説明したように、各フィンガプリント関数は、そのオブジェクト中の位置を囲む小さなデータウィンドウを使用する。ここで、このフィンガプリント関数に関連する値は 50

、このデータウィンドウの内側に含まれるこのオブジェクトのすべてのバイトに依存する。このデータウィンドウのサイズは、1つまたは複数の判断基準に基づいて動的に調整することができる。さらに、このチャンキングプロシーダは、このフィンガプリント関数の値と1つまたは複数の追加のチャンキングパラメータを使用して以上のステップ2～3および4、2～4、3においてこのチャンク境界を決定する。

【0050】

ウィンドウサイズとチャンキングパラメータを動的に変更することにより、このチャンク境界は、どのような必要なデータ転送でもその使用可能な帯域幅を最小限に消費するだけで実現されるように調整される。

【0051】

このウィンドウサイズとチャンキングパラメータを調整するための判断基準の実施例は、そのオブジェクトに関連するデータタイプ、環境制約、使用モデル、デバイスAとデバイスBとを接続するネットワークの待ち時間特性および帯域幅特性、および平均データ転送ブロックサイズを決定するための他の適切な任意のモデルを含んでいる。データタイプの実施例には、ワードプロセッシングファイル、データベースイメージ、スプレッドシート、プレゼンテーションスライドショー、およびグラフィックイメージが含まれる。使用モデルの一実施例には、典型的なデータ転送で必要とされるその平均バイト数が監視される場合もある。

【0052】

アプリケーションプログラム内における1つの要素に対する変更が、その関連するデータおよび／またはファイルに対するいくつかの変更をもたらす可能性がある。ほとんどのアプリケーションプログラムは、関連するファイルタイプを有するので、このファイルタイプは、そのウィンドウサイズおよびそのチャンキングパラメータを調整するに際して考慮に値する可能性のある1つの判断基準である。一実施例においては、ワードプロセッシングドキュメントにおける1つのキャラクタの修正により、この関連するファイル中で約100バイトの変更がもたらされることになる。別の実施例においては、データベースアプリケーションにおける1つの要素の修正により、そのデータベース索引ファイル中で1000バイトの変更がもたらされることになる。各実施例では、その適切なウィンドウサイズおよびチャンキングパラメータは、このチャンキングプロシーダがその特定のアプリケーションに基づいて最適化される適切な細分性を有するように異なる場合もある。

【0053】

プロセスフローの実施例

図4Aおよび4Bは、本発明の少なくとも一態様に従って構成される一実施例のRDCプロセス中のローカルデバイス（例えば、デバイスA）と、リモートデバイス（例えば、デバイスB）との間の対話についてのプロセスフローを示す図である。図4Aの左側は、このローカルデバイスA上で動作させられるステップ400～413を示すが、一方、図4Aの右側は、このリモートデバイスB上で動作させられるステップ450～456を示している。

【0054】

図4Aに示すように、この対話は、ステップ400において、デバイスAがオブジェクトO_BのRDC転送を要求し、ステップ450において、デバイスBがこの要求を受信することによって開始される。これに続いて、このローカルデバイスAとリモートデバイスBは共に、それぞれステップ401および451において、フィンガプリントを独立に計算し、ステップ402および452において、それらの各オブジェクトをチャンクに分割し、ステップ403および453においてチャンクごとにシグネチャ（例えば、SHA）を計算する。

【0055】

ステップ454において、デバイスBは、ステップ452および453で計算したそのシグネチャおよびチャンク長のリストをデバイスAに送信し、デバイスAは、ステップ404においてこの情報を受信する。

10

20

30

40

50

【0056】

ステップ405において、ローカルデバイスAは、この要求されたチャンクリスト（REQUEST）を空リスト（Empty）に初期化し、このリモートチャンクについての追跡オフセットを0に初期化する。ステップ406において、ステップ404で受信されるそのリストから次の（シグネチャ、チャンク長）対（Sig_{B_i}, Len_{B_i}）を考慮するために選択する。ステップ407において、デバイスAは、ステップ406で選択されたこのシグネチャSig_{B_i}がステップ403中にデバイスAが計算したシグネチャのどれかとマッチングするかどうかを検査する。シグネチャがマッチングする場合は、ステップ409から実行が継続される。シグネチャがマッチングしない場合には、ステップ408において、この追跡リモートチャンクオフセットおよびバイト長Len_{B_i}がその要求リストに追加される。ステップ409において、この追跡オフセットは、現行のチャンク長Len_{B_i}分だけ増分される。 10

【0057】

ステップ410において、ローカルデバイスAは、ステップ404中に受信されたすべての（シグネチャ、チャンク長）対が処理されているかどうかをテストする。処理されていない場合には、ステップ406から実行が継続される。処理されている場合には、ステップ411において、このチャンク要求リストは、コンパクトな様式に適切に符号化され、圧縮され、リモートデバイスBに送信される。

【0058】

ステップ455において、リモートデバイスBは、この圧縮されたチャンクリストを受信し、ステップ456においてこのチャンクリストを伸張し、次いでこのチャンクデータを圧縮し返送する。 20

【0059】

ステップ412において、このローカルデバイスは、この要求されたチャンクデータを受信し伸張する。ステップ413において、オブジェクトO_Aのローカルコピーと受信されたチャンクデータとを使用して、このローカルデバイスは、O_Bのローカルコピーを再組立てする。

【0060】

図4Bは、図4Aからのステップ413についての詳細な実施例を示している。処理がステップ414から継続され、このステップにおいて、ローカルデバイスAは、この再構築されたオブジェクト（NEWOBJECT）を空（Empty）に初期化する。 30

【0061】

ステップ415において、次の（シグネチャ、チャンク長）対（Sig_{B_i}, Len_{B_i}）がステップ404で受信されたリストから考慮するために選択される。ステップ416において、デバイスAは、ステップ417中で選択されたシグネチャSig_{B_i}が、ステップ403中にデバイスAが計算したシグネチャのどれかとマッチングするかどうかを検査する。

【0062】

シグネチャがマッチングする場合、ステップ417から実行が継続され、このステップではその対応するローカルチャンクがその再構築済みのオブジェクトに追加される。シグネチャがマッチングしない場合には、ステップ418において、この受信され伸張されたリモートチャンクが、この再構築済みのオブジェクトに追加される。 40

【0063】

ステップ419においては、ローカルデバイスAは、ステップ404で受信されたすべての（シグネチャ、チャンク長）対が処理されているかどうかをテストする。処理されていない場合、ステップ415から実行が継続される。処理されている場合には、ステップ420において、この再構築済みのオブジェクトを使用してデバイスA上のオブジェクトO_Aの以前のコピーを置き換える。

【0064】

実施例の再帰的シグネチャ転送プロセスフロー

図 5 A および 5 B は、本発明の少なくとも一態様に従って構成される R D C プロシージャの一実施例におけるそのシグネチャおよびチャンク長のリストの再帰的な転送についてのプロセスフローを示す図である。以下で説明するプロシージャは、一般的に関連するオブジェクトを更新しようと試みるローカルデバイスとリモートデバイスの両方に対して適用することができる。

【0065】

図 5 A の左側は、このローカルデバイス A 上で動作させられるステップ 5 0 1 ~ 5 1 3 を示すが、一方、図 5 A の右側は、このリモートデバイス B 上で動作させられるステップ 5 5 1 ~ 5 5 6 を示している。ステップ 5 0 1 ~ 5 1 3 は、図 4 A 中のステップ 4 0 4 を置き換えるが、ステップ 5 5 1 ~ 5 5 6 は、図 4 A 中のステップ 4 5 4 を置き換えるものである。 10

【0066】

ステップ 5 0 1 および 5 5 1 において、これらのローカルデバイス A とリモートデバイス B は共に、それぞれステップ 4 0 2 / 4 0 3 と 4 5 2 / 4 5 3 で計算されているそのシグネチャおよびチャンク長のリスト ((S i g _{A 1} , L e n _{A 1}) . . . (S i g _{A k} , L e n _{A k})) と ((S i g _{B 1} , L e n _{B 1}) . . . (S i g _{B n} , L e n _{B n})) の再帰的フィンガプリントを独立に計算する。これらのデバイスは、それぞれステップ 5 0 2 および 5 5 2 において、それらの各シグネチャおよびチャンク長のリストを再帰的チャンクに分割し、ステップ 5 0 3 および 5 5 3 において、再帰的チャンクごとに再帰的シグネチャ (例えば、S H A) を計算する。 20

【0067】

ステップ 5 5 4 において、デバイス B は、このステップ 5 5 2 および 5 5 3 で計算された再帰的なシグネチャおよびチャンク長のリストをデバイス A に送信し、このデバイス A は、ステップ 5 0 4 において、この情報を受信する。

【0068】

ステップ 5 0 5 において、ローカルデバイス A は、この要求された再帰的チャンクリスト (R E Q U E S T) を空リスト (E m p t y) に初期化し、このリモート再帰的チャンクについてのこの追跡リモート再帰的オフセット (R O F F S E T) を 0 に初期化する。ステップ 5 0 6 において、次の (再帰的シグネチャ, 再帰的チャンク長) 対 (R S i g _{B i} , R L e n _{B i}) が、考慮するために、ステップ 5 0 4 で受信されたリストから選択される。ステップ 5 0 7 において、デバイス A は、ステップ 5 0 6 で選択されたこの再帰的シグネチャ R S i g _{B i} がステップ 5 0 3 中にデバイス A が計算したその再帰的シグネチャのどれかとマッチングするかどうかを検査する。シグネチャがマッチングする場合、ステップ 5 0 9 から実行が継続される。シグネチャがマッチングしない場合には、ステップ 5 0 8 において、この追跡リモート再帰的チャンクオフセットおよびバイト長 R L e n _{B i} が、その要求リストに追加される。ステップ 5 0 9 において、この追跡リモート再帰的オフセットは、現行の再帰的チャンク長 R L e n _{B i} 分だけ増分される。 30

【0069】

ステップ 5 1 0 において、ローカルデバイス A は、ステップ 5 0 4 で受信されたすべての (再帰的シグネチャ, 再帰的チャンク長) 対が、処理されているかどうかをテストする。処理されていない場合、ステップ 5 0 6 から実行が継続される。処理されている場合には、ステップ 5 1 1 において、この再帰的チャンク要求リストは、コンパクトに符号化され、圧縮され、リモートデバイス B に送信される。 40

【0070】

このリモートデバイス B は、ステップ 5 5 5 において、再帰的チャンクのその圧縮されたリストを受信し、そのリストを伸張し、次いでステップ 5 5 6 においてこの再帰的チャンクデータを圧縮し返送する。

【0071】

ステップ 5 1 2 において、このローカルデバイスは、この要求された再帰的チャンクデータを受信し伸張する。ステップ 5 1 3 において、そのシグネチャおよびチャンク長のリ 50

スト（（ Sig_{A1} ， Len_{A1} ）．．．（ Sig_{Ak} ， Len_{Ak} ））のそのローカルコピーとこの受信された再帰的チャンクデータとを使用して、このローカルデバイスは、そのシグネチャおよびチャンク長のリスト（（ Sig_{B1} ， Len_{B1} ）．．．（ Sig_{Bk} ， Len_{Bn} ））のローカルコピーを再組立てする。次いで、図4A中のステップ405から実行が継続される。

【0072】

図5Bは、図5Aからのステップ513についての詳細な実施例を示している。処理は、ステップ514から継続され、このステップにおいて、このローカルデバイスAは、リモートシグネチャおよびチャンク長のリストSIGCLを空リストに初期化する。

【0073】

ステップ515において、次の（再帰的シグネチャ，再帰的チャンク長）対（ RSig_{Bi} ， RLen_{Bi} ）が、考慮するために、ステップ504で受信されたリストから選択される。ステップ516において、デバイスAは、ステップ515で選択された再帰的シグネチャ RSig_{Bi} が、ステップ503中にデバイスAが計算した再帰的シグネチャのどれかとマッチングするかどうかを検査する。

【0074】

シグネチャがマッチングする場合、ステップ517から実行が継続され、このステップにおいてデバイスAは、対応するローカル再帰的チャンクをSIGCLに追加する。シグネチャがマッチングしない場合には、ステップ518において、このリモート受信された再帰的チャンクがSIGCLに追加される。

【0075】

ステップ519において、このローカルデバイスAは、ステップ504で受信されたすべての（再帰的シグネチャ，再帰的チャンク長）対が処理されているかどうかをテストする。処理されていない場合、ステップ515から実行が継続される。処理されている場合には、ステップ520において、このシグネチャおよびチャンク長のリスト（（ Sig_{B1} ， Len_{B1} ）．．．（ Sig_{Bk} ， Len_{Bn} ））のローカルコピーが、SIGCLの値に設定される。次いで、図4A中のステップ405に戻って実行が継続される。

【0076】

この再帰的シグネチャおよびチャンク長のリストをオプションとして評価して、前述のように帯域幅の利用を最小にするために追加の再帰的遠隔差分圧縮が必要かどうかは決定される。この再帰的シグネチャおよびチャンク長のリストは、ステップ504および554をこのRDCプロシーダの別のインスタンスなどで置き換えることにより、所望の圧縮レベルが達成されるまでこの説明したチャンキングプロシーダを使用して再帰的に圧縮することができる。この再帰的なシグネチャリストが十分に圧縮された後に、前述のようにこのリモートデバイスとローカルデバイスの間で伝送するために、この再帰的なシグネチャリストが返される。

【0077】

図6は、例示の本発明の実施形態に従って構成されるRDCシーケンスの実施例における再帰的圧縮の実施例を図示した図である。図6に示す実施例では、最初のオブジェクトは、9.1GBのデータである。シグネチャおよびチャンク長のリストは、チャンキングプロシーダを使用してコンパイルされ、ここでこのシグネチャおよびチャンク長のリストにより、3百万個のチャンク（または、42MBのサイズ）がもたらされる。第1の再帰的ステップの後に、このシグネチャリストは、3万3千個のチャンクに分割され、33KBのサイズの再帰的シグネチャおよび再帰的チャンク長のリストにまで減少する。このシグネチャリストを再帰的に圧縮することにより、このシグネチャリストを転送するための帯域幅利用率は、したがって42MBから395KBに劇的に減少する。

【0078】

オブジェクト更新の実施例

図7は、本発明の少なくとも一態様に従って構成されるRDCプロシーダの一実施例を使用したクライアントアプリケーションとサーバアプリケーションの対話を示す図であ

10

20

30

40

50

る。サーバとクライアントの両方の上の最初のファイルは、テキスト「The quick fox jumped over the lazy brown dog. The dog was so lazy that he didn't notice the fox jumping over him. (すばやい狐は、のろまな茶色の犬の上を飛び越えた。この犬はあまりにもものろまだったので、狐が自分の上を飛び越えたことに気がつかなかった。)」を含んでいた。

【0079】

後続の時刻において、このサーバ上のこのファイルは、「The quick fox jumped over the lazy brown dog. The brown dog was so lazy that he didn't notice the fox jumping over him. (すばやい狐は、のろまな茶色の犬の上を飛び越えた。この茶色の犬はあまりにもものろまだったので、狐が自分の上を飛び越えたことに気がつかなかった。)」に更新される。 10

【0080】

以前に説明したように、このクライアントは、このファイルが更新されるように定期的に要求する。これらのクライアントおよびサーバは共に、このオブジェクト（テキスト）を図に示すようにチャンクにチャンキングする。このクライアント上では、それらのチャンクは、「The quick fox jumped」、「over the lazy brown dog.」、「The dog was so lazy that he didn't notice」、および「the fox jumping over him.」であり、このクライアントシグネチャリストは、SHA_{1 1}、SHA_{1 2}、SHA_{1 3}、およびSHA_{1 4}として生成される。このサーバ上では、それらのチャンクは、「The quick fox jumped」、「over the lazy brown dog.」、「The brown dog was」、「so lazy that he didn't notice」、および「the fox jumping over him.」であり、このサーバシグネチャリストは、SHA_{2 1}、SHA_{2 2}、SHA_{2 3}、SHA_{2 4}、およびSHA_{2 5}として生成される。 20

【0081】

このサーバは、以前に説明したような再帰的シグネチャ圧縮技法を使用してこのシグネチャリスト（SHA_{2 1}～SHA_{2 5}）を送信する。このクライアントは、そのローカルに記憶済みのシグネチャリスト（SHA_{1 1}～SHA_{1 4}）が、その受信されたシグネチャリスト（SHA_{2 1}～SHA_{2 5}）にマッチングしないことを認識し、このサーバからの不足しているチャンク3および4を要求する。このサーバは、チャンク3および4（「The brown dog was」および「so lazy that he didn't notice」）を圧縮し送信する。このクライアントは、図7に示すように、これらの圧縮されたチャンクを受信し、それらを伸張し、そのファイルを更新する。 30

【0082】

チャンキング解析

前述のこの基本的なRDCプロシージャの有効性は、このオブジェクトデータをチャンキングし、かつ／またはこのシグネチャおよびチャンク長のリストをチャンキングするために使用されるチャンキングプロシージャを最適化することによって増大させることができる。 40

【0083】

この基本的なRDCプロシージャは、以下の合計に一致するネットワーク通信オーバーヘッドコストを有する。

【0084】

(S1) |Bからのシグネチャおよびチャンク長| = |O_B| × |SigLen| / C、式中で|O_B|は、オブジェクトO_Bのバイトサイズであり、SigLenは、(シグネチャ、チャンク長)対のバイトサイズであり、Cは、予想される平均チャンクのバイトサイズであり、

(S2) Σ chunk_length、式中で(signature, chunk_length) ∈ Bからのシグネチャ、および 50

【0085】

【数1】

signature ∈ Aからのシグネチャ

【0086】

である。

【0087】

したがってこの通信コストは、大きな平均チャンクサイズ、ならびにこのリモートチャンクとローカルチャンクの間大きな共通部分からの恩恵を受ける。どのようにしてオブジェクトをチャンクに切り分けるかの選択が、このプロトコルの品質を決定する。このローカルデバイスとリモートデバイスは、事前の通信なしにどこでオブジェクトを切り分けるかについて合意する必要がある。以下は、切れ目(cut)を見出すための様々な方法を説明し解析している。

【0088】

以下の特性が、このカッティングアルゴリズム(cutting algorithm)について知られているものと想定する。

【0089】

1. スラック(slack)：異なるファイル間でチャンクを一致させるために必要なバイト数。s₁、s₂、およびs₃を考慮して、連結によって2つのシーケンスs₁s₃、およびs₂s₃を形成する。これら2つのシーケンスについてのチャンク、Chunk s₁およびChunk s₂を生成する。Chunk s₁'およびChunk s₂'が、それぞれChunk s₁およびChunk s₂からのチャンク長の和である場合、第1の共通の接尾辞に到達するまで、バイトのスラックは、以下の式によって与えられる。

$$slack = Chunk s_1' - |s_1| = Chunk s_2' - |s_2|$$

【0090】

2. 平均チャンクサイズC：

オブジェクトO_AおよびO_Bが、平均サイズKを共通にもつS個のセグメントを有するとき、このクライアント上でローカルに取得することができるチャンク数は、次式で与えられる。

【0091】

【数2】

$$S * \lfloor (K - slack) / C \rfloor$$

【0092】

前記(S2)は、以下のように書き換えられる。

【0093】

【数3】

$$\lfloor O_A \rfloor - S * \lfloor (K - slack) / C \rfloor$$

【0094】

したがって、スラックを最小化するチャンキングアルゴリズムは、このワイヤ上で送信されるバイト数を最小化することになる。したがって、この予想されるスラックを最小化するチャンキングアルゴリズムを使用することが有利である。

【0095】

フィンガプリント関数

すべてのチャンキングアルゴリズムは、スモールウィンドウ(小型のウィンドウ)、すなわち限られたバイトシーケンスに依存するフィンガプリント関数またはハッシュを使用する。これらのアルゴリズムが有限の差分(強度削減)最適化に従うとき、チャンキングのために使用されるハッシュアルゴリズムの実行時間は、このハッシュウィンドウサイズから独立している。したがって、サイズkのハッシュウィンドウでは、b₀、b_k、およ

10

20

30

40

50

び# $[b_0, b_1, \dots, b_{k-1}]$ だけを使用してそのハッシュ# $[b_1, \dots, b_{k-1}, b_k]$ を計算することは簡単な（一定のステップ数しか必要としない）はずである。ラビン多項式を使用したハッシュ関数や、あらかじめ計算済みの乱数テーブルに基づいた計算上より効率的に見える他のハッシュ関数など、様々なハッシング関数（hashing function）を使用することができる。

【0096】

一実施例においては、ローリングチェックサム（rolling checksum）に基づいた32ビットのアドラーハッシュ（Adler hash）をフィンガプリントのためのハッシング関数として使用することができる。このプロシージャでは、それぞれあらかじめ計算済みの16ビットの乱数の256エントリを有する固定テーブルを使用することによって妥当で良好なランダムハッシュ関数が提供される。このテーブルを使用してフィンガプリントバイトがランダムな16ビットの数に変換される。この32ビットのハッシュが2つの16ビットの数sum1およびsum2に分割され、これらは、以下のプロシージャを想定して更新される。

$$\begin{aligned} \text{sum1} &+= \text{table}[b_k] - \text{table}[b_0] \\ \text{sum2} &+= \text{sum1} - k \times \text{table}[b_0] \end{aligned}$$

【0097】

他の実施例においては、循環シフティング（cyclic shifting）を伴う64ビットのランダムハッシュは、フィンガプリントのためのハッシング関数として使用することができる。循環シフトの周期は、そのハッシュ値のサイズによって縛られる。したがって、64ビットのハッシュ値の使用によって、このハッシュの周期は64に設定される。このハッシュを更新するためのプロシージャは、次式で与えられる。

$$\begin{aligned} \text{hash} &= \text{hash} \wedge ((\text{table}[b_0] \ll 1) | (\text{table}[b_0] \gg u)) \wedge \text{table}[b_k]; \\ \text{hash} &= (\text{hash} \ll 1) | (\text{hash} \gg 63); \end{aligned}$$

式中で $l = k \% 64$ 、および $u = 64 - l$

【0098】

さらに他の実施例においては、他のシフティング手法を使用してフィンガプリントを提供することができる。直接的循環シフティングは、限られた長さの周期を生成し、そのハッシュ値のサイズによって縛られる。他の順列は、より長い周期を有する。例えば、循環が提供する順列、 $(1 \ 2 \ 3 \ 0) \ (5 \ 6 \ 7 \ 8 \ 9 \ 10 \ 11 \ 12 \ 13 \ 14 \ 4) \ (16 \ 17 \ 18 \ 19 \ 20 \ 21 \ 15) \ (23 \ 24 \ 25 \ 26 \ 22) \ (28 \ 29 \ 27) \ (31 \ 30)$ は長さ $4 \times 3 \times 5 \times 7 \times 11 = 4620$ の周期を有する。この実施例の順列の1つの適用では、各間隔の先頭における位置にパッチアップ（patch up）する動作が後に続く右シフトを使用して計算することができる。

【0099】

所定のパターンにおいてチャンキングするための以前の技術の解析

以前のチャンキング手法では、所定のウィンドウサイズ k ($= 48$) を有するフィンガプリントハッシュを計算し、ハッシュビットのサブセットが所定のパターンとマッチングするかどうかに基づいてカットポイントを識別することによって決定される。ランダムハッシュ値を用いると、このパターンは、同様に0となることもあり、これに関連するサブセットも同様にこのハッシュの接頭辞になることもある。基本的な命令においては、これは、以下の形式の述語に変換される。

$$\text{CutPoint}(\text{hash}) \equiv 0 == (\text{hash} \& ((1 \ll c) - 1))$$

式中、 c はマッチングさせるべきビット数である。

【0100】

ランダムハッシュ関数の与えられたマッチングについての確率が 2^{-C} であるので、平均チャンクサイズ $C = 2^C$ がもたらされる。しかし、最小チャンクサイズも最大チャンクサイズもこのプロシージャによっては決定されない。 m の最小チャンク長が定められる場

合には、平均チャンクサイズは、次のようになる。

$$C = m + 2^C$$

【0101】

この予想されるスラックの大まかな推定値は、ストリーム s_1 s_3 および s_2 s_3 を考慮することによって得られる。 s_1 および s_2 におけるカットポイントは、どのような位置に現れてもよい。平均チャンク長は、 $C = m + 2^C$ であるので、 s_1 および s_2 における最後のカットポイントの $(2^C / C)^2$ 付近では、距離 m を超えることになる。これらは、およそ 2^C におけるスラックに寄与することになる。この残りの $1 - (2^C / C)^2$ は約 C の長さのスラックで寄与する。その場合には、この予想されるスラックは、およそ $(2^C / C)^3 + (1 - (2^C / C)^2) \times (C / C) = (2^C / C)^3 + 1 - (2^C / C)^2$ となり、これは、 $m = 2^{C-1}$ についてグローバル最小値を有し、約 $23 / 27 = 0.85$ の値を有する。より正確な解析では、残りの $1 - (2^C / C)^2$ 端数部についていくらか小さな推定値が提供されるが、 s_3 内側の距離 m の内部のカットについて補償する必要があることにもなり、これはより大きな推定値に寄与する。

【0102】

したがって、従来技術についての予想されるスラックは、約 $0.85 \times C$ となる。

【0103】

フィルタにおけるチャンキング（新しい技術）

フィルタにおけるチャンキングは、長さ m のパターンシーケンスとなっているフィルタを修正し、このフィルタに対してフィンガプリントハッシュシーケンスをマッチングさせることに基づいている。このフィルタが、ハッシュシーケンスをこのフィルタの接頭辞にも接尾辞にもマッチングさせないときには、任意の2つのマッチング部間の最小距離が、少なくとも m になる必要があることが推論できる。フィルタの一実施例は、以前の技術で使用されたカットポイント述語から、最初の $m-1$ 個のパターンを次式のように設定し、

$$0! = (\text{hash} \& ((1 \ll c) - 1))$$

また最後のパターンを次式のように

$$0 = (\text{hash} \& ((1 \ll c) - 1))$$

設定することによって得ることができる。

【0104】

このフィルタをマッチングさせる確率は、 $(1-p)^{m-1} p$ で与えられ、式中で p は 2^{-C} である。この予想されるチャンク長が、フィルタをマッチングさせる確率の逆数によって与えられることを算出することができ（このフィルタが、シーケンスが接頭辞にも接尾辞にもマッチングさせないようにする必要がある）、したがってこの実施例のフィルタの予想される長さは、 $(1-p)^{-m+1} p^{-1}$ となる。この長さは、 $p := 1/m$ を設定する際に最小になり、これは約 $(e \times m)$ であることが分かる。この平均スラックは、約 0.8 付近にあり、当業者なら検証することができよう。この方法の代替実施形態は、生入力を用いて直接に機能するパターンを使用し、ローリングハッシュは使用していない。

【0105】

極大におけるチャンキング（新しい技術）

極大 (Local Maxima) におけるチャンキングは、縛られたホライズン内部で最大となるカットポイント位置として選択することに基づいている。以下では、本発明者らは、ホライズン値については h を使用することにする。 $offset-h$ 、...、 $offset-1$ 、ならびに $offset+1$ 、...、 $offset+h$ におけるハッシュ値がすべて、 $offset$ におけるハッシュ値よりも小さい場合には、位置 $offset$ におけるハッシュは、 h -極大であると言う。換言すれば、 h ステップ左から h ステップ右までのすべての位置が、より小さなハッシュ値を有する。極大は、極小または（メジアンハッシュ値に対する「最近接」 (closest to the median hash value) など）比較に基づく他の任意の距離で置き換えることもできることが当業者には理解されよう。

10

20

30

40

50

【0106】

サイズ n のオブジェクトについての極大の組は、この極大の組を計算するコストが独立なチャンキングに基づいてこのカットポイントを計算するコストと近い、または同じになるように $2 \cdot n$ 個の演算によって縛られる時間内で計算することができる。極大を使用して生成されるチャンクは常に、約 $2h + 1$ の平均サイズを伴い、 h に対応する最小サイズを有する。Cut Point プロシーダを、図 8 および 9 に示し、これについて次のように以下に説明する。

【0107】

1. エントリがレコード $\{isMax = false, hash = 0, offset = 0\}$ で開始される長さ h のアレイ M を割り付ける。各フィールド中の第 1 のエントリ ($isMax$) は、候補が極大になりうるかどうかを示す。第 2 のフィールドエントリ ($hash$) は、そのエントリに関連するハッシュ値を示し、0 (またはその代わりに、可能な極大ハッシュ値) に初期化される。このエントリ中の最後のフィールド ($offset$) は、このフィンガプリントオブジェクトへの候補に対するバイトでの絶対オフセットを示す。

【0108】

2. アレイ M へのオフセット最小値および最大値を 0 に初期化する。これらの変数は、現在使用されているこのアレイの最初の要素および最後の要素を指し示す。

【0109】

3. Cut Point ($hash, offset$) は、図 8 のステップ 800 から開始され、このオブジェクトの各オフセットで呼び出されて M を更新し、特定のオフセットがカットポイントであるかどうかを示す結果を返す。

【0110】

このプロシーダはステップ 801 において $result = false$ を設定することによって開始される。

【0111】

ステップ 803 において、このプロシーダは、 $M[max].offset + h + 1 = offset$ が成立するかどうかを検査する。この条件が真である場合、ステップ 804 から実行が継続され、このステップで以下の割当てが実施される。すなわち、結果が $M[max].isMax$ に設定され、 max が $max - 1 \% h$ に設定される。次いでステップ 805 から実行が継続される。ステップ 803 における条件が偽である場合には、ステップ 805 から実行が継続される。

【0112】

ステップ 805 において、このプロシーダは、 $M[min].hash > hash$ が成立するかどうかを検査する。この条件が真である場合、ステップ 806 から実行が継続され、ここで、 min が $(min - 1) \% h$ に設定される。ステップ 807 から実行が継続され、ここで $M[min]$ は、 $\{isMax = false, hash = hash, offset = offset\}$ に設定され、ステップ 811 に進み、ここで計算結果が返される。

【0113】

ステップ 805 における条件が偽である場合には、ステップ 808 から実行が継続され、ここでこのプロシーダは $M[min].hash = hash$ が成立するかどうかを検査する。この条件が真である場合、ステップ 807 から実行が継続される。

【0114】

ステップ 808 における条件が偽である場合には、ステップ 809 から実行が継続され、ここでこのプロシーダは、 $min = max$ が成立するかどうかを検査する。この条件が真である場合、ステップ 810 から実行が継続され、ここで $M[min]$ は、 $\{isMax = true, hash = hash, offset = offset\}$ に設定される。次いで、ステップ 811 から実行が継続され、ここでその計算結果が返される。

【0115】

10

20

30

40

50

ステップ809における条件が偽である場合には、ステップ811から実行が継続され、ここでminは、 $(\min + 1) \% h$ に設定される。次いでステップ805に戻って実行が継続される。

【0116】

4. CutPoint (hash, offset) が真値を返すときには、それは、位置offset-h-1におけるオフセットが新しいカットポイントになる場合のほずである。

【0117】

極大プロシージャの解析

nバイトを有するオブジェクトは、所与のオブジェクトについて多くてもn個のエントリが挿入されるようにCutPointをn回呼び出すことによって処理される。n個より多いエントリを削除することがないように、ステップ805から開始されるループが反復されるたびに1つのエントリが除去される。したがって、この処理ループは、あらゆるエントリについて一度しか入ることができず、反復の結合回数は、多くてもn回とすることができる。これは、CutPointに対する各呼出しにおけるループ内のステップの平均数が2よりわずかに少なく、カットポイントを計算するステップ数が、hからは独立していることを意味する。

【0118】

これらの要素からのハッシュ値は、minとmaxの間の降順チェーンを形成するので、minとmaxの間の平均距離 $(|\min - \max| \% h)$ は、hの自然対数によって与えられることが分かる。M中の2つの隣接エントリの中に含まれないオフセットは、これら2つのエントリ以下のハッシュ値を有する。かかるチェーンの平均長は、漸化式 $f(n) = 1 + 1/n \times \sum_{k < n} f(k)$ によって与えられる。長さnの間隔上の最長の降順チェーンの平均長は、最大要素の位置から開始されるこの最長の降順チェーンの平均長よりも1だけ大きく、ここでこの最大要素は、任意の位置において確率 $1/n$ で見出すことができる。この循環関係は、調和次数に対応する解として $H_n = 1 + 1/2 + 1/3 + 1/4 + \dots + 1/n$ を有し、この式は、この式に H_n を代入し、nについての帰納法を実施することによって正当性を検証することができる。 H_n はnの自然対数に比例している。したがって、アレイMにはサイズhが割り付けられているが、どんなときにもサイズの小さな端数部 $1/n(h)$ しか使用されない。

【0119】

法hを用いてminおよびmaxを計算することにより、これらの数の間の距離がh内にとどまる限りは、Mの使用間隔を任意に大きくすることが可能である。

【0120】

Mについての初期値の選択は、カットポイントがこの最初のhオフセット内で生成できることを意味する。このアルゴリズムでは、これらの最初のhオフセットにおけるカットポイントを回避するようにすることもできる。

【0121】

このプロシージャが生成するチャンクの予想されるサイズは、約 $2h + 1$ である。所与の位置がカットポイントであるというその確率からこの数字が得られる。このハッシュがm個の可能な異なる値を有するものと想定する。その場合には、この確率は次式によって決定される。

$$\sum_{0 \leq k < m} 1/m (k/m)^{2h}$$

【0122】

積分 $\int_{0 \leq x < m} 1/m (x/m)^{2h} dx = 1/(2h + 1)$ を使用した近似は、mが十分に大きいときのこの確率を示している。

【0123】

この確率は、まずこの和を次式に簡略化することによってさらに高精度に計算することができる。

$$(1/m)^{2h+1} \sum_{0 \leq k < m} k^{2h}$$

10

20

30

40

50

【0124】

この式は、ベルヌーイ数 (Bernoulli number) B_k を使用して次式に展開される。

$$(1/m)^{2h+1} \frac{1}{(2h+1)} \sum_{0 \leq k < 2h} (2h+1)! / k! (2h+1-k)! B_k m^{2h+1-k}$$

【0125】

非ゼロの唯一の奇数のベルヌーイ数は、 B_1 であり、これは $-1/2$ の対応する値を有する。偶数のベルヌーイ数は、次式を満たす。

$$H_{\infty}^{(2n)} = (-1)^{n-1} 2^{2n-1} \pi^{2n} B_{2n} / (2n)!$$

【0126】

この左辺は無限の和 $1 + (1/2)^{2n} + (1/3)^{2n} + \dots$ を表し、これは、 n のちょうど穏当な値については、1 に非常に近くなる。

【0127】

m が h に比べてずっと大きいときには、積分によって分かったように、第1の項以外のすべての項を無視することができる。これらは、 h^{k-1}/m^k に比例する項で0と1の間の定数に乗ずることによって与えられる。第1項 (ここでは、 $B_0 = 1$) は、 $1/(2h+1)$ に簡略化される。(第2項は $-1/(2m)$ であり、第3項は $h/(6m^2)$ である)。

【0128】

この予想されるスラックの概略推定値については、ストリーム s_1 s_3 および s_2 s_3 を考慮する。 s_1 および s_2 の内側の最後のカットポイントは、どのような位置に現れてもよい。平均チャック長は約 $2h+1$ であるので、最後のカットポイントの約 $1/4$ までは、 s_1 でも s_2 でも距離 h の範囲内にあることになる。これらは、約 $7/8h$ におけるカットポイントに寄与する。他の $1/2$ の場合においては、一方のカットポイントは、距離 h の範囲内にあり、他方は、距離 h を超えている。これらは、約 $3/4h$ のカットポイントで寄与する。 s_1 および s_2 の最後のカットポイントの残りの $1/4$ は、 h よりも長い距離にあることになる。したがって、この予想されるスラックは、約 $1/4 \times 7/8 + 1/2 \times 3/4 + 1/4 \times 1/4 = 0.66$ になる。

【0129】

したがって、発明者らの独立なチャンキング手法についての予想されるスラックは、 $0.66 \times C$ となり、これは、従来技術 ($0.85 \times C$) よりも改善されている。

【0130】

h 、または平均して $\ln h$ にせいぜい比例するスペースを使用しながら、平均して少ない命令しか実行する必要のない、カットポイントを識別する代替手法が存在する。以上のプロシージャでは、長さ n のストリーム中のあらゆる位置 $0 \dots n-1$ ごとにエントリが挿入される。代替プロシージャにおける基本的なアイデアは、長さ h の間隔内の昇順チェーンの要素に出会う際にしか更新を行わないことである。発明者らは、間隔あたりに平均して $\ln h$ のこのような更新しか存在しないことを観察した。さらに、長さ h の2つの連続した間隔中で極大を比較することにより、これら2つの極大のそれぞれが、 h 極大にもなり得るかどうかを決定することができる。この代替プロシージャには、1つの特殊性がある。この代替プロシージャは、各ブロックが逆方向にトラバース (traverse) される、サイズ h のブロック中でこのストリームをトラバースすることによってその昇順チェーンを計算する必要がある。

【0131】

代替プロシージャ (図10および11参照) においては、簡潔に説明するためにハッシュのストリームがシーケンスとして提供されると想定している。このサブルーチン Cut Point は、(図中で「horizon」に展開された) 長さ h のサブシーケンスごとに呼び出される。このサブルーチンは、0 またはカットポイントであると決定された1つのオフセットを返す。Insert に対する呼出しの $\ln(h)$ だけがこの最初のテストに合格することになる。

10

20

30

40

50

【0132】

Aへの挿入は、それまでのA中の最大のエントリに対してこのオフセットにおけるそのハッシュ値をテストすることによって達成される。

【0133】

A[k]もB[k]、isMaxも共に更新するループは、平均してそのループ本体において1つのテストしか実施されないように最適化することができる。B[1].hash ≤ A[k].hash、およびB[1].isMaxのこのケースは、2つのループで処理され、この第1のものは、その第2の更新A[k]よりも小さくなるまでB[1].hashに対してそのハッシュ値を検査する。他方のケースは、B[1].isMaxへの更新が続くA[k]しか更新しないループを使用して処理することができる。

10

【0134】

CutPointに対する各呼出しは、Aに対して平均でlnh回のメモリ書込みを必要とし、極大を見出すことに関連したh+lnh回の比較を行うループを伴う。A[k].isMaxに対する最後の更新は、バイナリ探索によって、またはインデックス0から開始して平均して多くてもloglnh回のステップでBをトラバースすることによって実施することができる。CutPointに対する各呼出しは、更新されるウィンドウ中の最後の位置におけるそのローリングハッシュを再計算することも必要とする。これは、そのローリングハッシュウィンドウのサイズだけの多数のステップを必要とする。

【0135】

改善されたチャンキングアルゴリズムの観察された利点

20

前述の極大手法とフィルタ手法では共に、最小のチャンクサイズが構築される。従来技術の実装形態では、最小のチャンクサイズには、追加のパラメータを別に供給することが必要になる。

【0136】

この極大（または数学）ベースの手法では、測定可能なより良好なスラック推定値が生成され、この推定値は変換されてこのネットワーク上でさらに圧縮される。フィルタ手法でも、従来技術の方法に比べてより良好なスラック性能が生成される。

【0137】

これらの新しい方法は共に、カットポイントの局所特性を有する。ホライズンを超えるs3の内側のすべてのカットポイントは、ストリームs1s3とs2s3の両方についてのカットポイントになる。（換言すれば、ストリームs1s3を考慮すると、pが $|s1| + horizon$ となる位置であり、pがs1s3中のカットポイントである場合には、pは、s2s3中のカットポイントでもある。この同じ特性が他の方向についても（対称的に）当てはまり、pがs2s3中のカットポイントである場合には、pは、s1s2中のカットポイントにもなる）。以上は、従来技術の方法では成り立たず、従来方法では、カットがある最小のチャンクサイズを超えるという要件が、不利に妨害し合うこともある。

30

【0138】

代替的な数学関数

前述のチャンキングプロシージャは、極大計算を使用してカットポイントを位置づける手段について説明しているが、本発明は、それだけには限定されない。任意の数学関数を構成して可能性のあるカットポイントを検査することができる。可能性のある各カットポイントは、考慮されたカットポイントについてのホライズンウィンドウ内に配置されたハッシュ値を評価することによって評価される。このハッシュ値の評価は、このホライズン内の最大値の位置を見つけるステップ、このホライズン内の最小値の位置を見つけるステップ、ハッシュ値の間の差分を評価するステップ、任意の定数に対してその結果を比較するステップのうちの少なくとも1つを含み得る数学関数、ならびに他の何らかの数学関数または統計的関数によって達成される。

40

【0139】

極大について以前に説明した特定の数学関数は、2項の述語「 $_ > _$ 」である。pがこ

50

のオブジェクト中のオフセットであるケースでは、 $hash_p > hash_k$ がすべての k について成立し、ここで $p - horizon \leq k < p$ 、または $p < k \leq p + horizon$ である場合には、 p はカットポイントとして選択される。しかし、この2項の述語 $>$ は、本発明の趣旨を逸脱することなく他の任意の数学関数で置き換えることができる。

【0140】

以上の本明細書、実施例、およびデータは、本発明の製造および構成物の使用についての完全なる記述を提供している。本発明の多数の実施形態は、本発明の趣旨および範囲を逸脱することなく作成することができるので、本発明は、添付の特許請求の範囲内に存在している。

【産業上の利用可能性】

10

【0141】

上述したように、本発明において、オブジェクトは、必要なデータ転送が最小化されるように遠隔差分圧縮 (RDC) 手法を使用して2つ以上のコンピューティングデバイス間で更新される。一態様においては、効率的な大きなオブジェクトの転送が、それ自体のメタデータに対してこのRDCアルゴリズムを再帰的に適用することによって達成される。すなわち、1つまたは複数の再帰的ステップをこのケースにおいて使用してこのRDCアルゴリズムによってこのネットワーク上で送信されるメタデータの量を減少させることができる。オブジェクト、および/または、シグネチャおよびチャンク長リストを、動的に決定されたロケーションに境界を位置づけることによってチャンキングすることができる。数学関数が、可能性のあるチャンク境界に関連してホライズンウィンドウ内に関連づけられたハッシュ値を評価する。従って、本発明の方法およびシステムは、ピアツーピアレプリケータ、電子メールクライアントおよび電子メールサーバ、クライアント側キャッシングシステム、汎用コピーユーティリティ、データベースレプリケータ、ポータル、ソフトウェア更新サービス、ファイル/データ同期化など様々なネットワーク化されたアプリケーションにおいて有用である。

20

【図面の簡単な説明】

【0142】

【図1】本発明に係わる動作環境を示す図である。

【図2】本発明に従うコンピューティングデバイスの一実施例を示す図である。

【図3A】本発明に従うRDCプロシージャの一実施例を示す図である。

30

【図3B】本発明に従うRDCプロシージャの一実施例を示す図である。

【図4A】本発明に従うRDCプロシージャの一実施例中における、ローカルデバイスとリモートデバイスの間の対話についてのプロセスフローを示す図である。

【図4B】本発明に従うRDCプロシージャの一実施例中における、ローカルデバイスとリモートデバイスの間の対話についてのプロセスフローを示す図である。

【図5A】本発明に従うRDCプロシージャ中における対話の一実施例におけるシグネチャおよびチャンク長のリストの再帰的遠隔差分圧縮についてのプロセスフローを示す図である。

【図5B】本発明に従うRDCプロシージャ中における対話の一実施例におけるシグネチャおよびチャンク長のリストの再帰的遠隔差分圧縮についてのプロセスフローを示す図である。

40

【図6】本発明に従うRDCシーケンスの一実施例における再帰的圧縮の一実施例を図示する図である。

【図7】本発明に従うRDCプロシージャの一実施例を使用した、クライアントアプリケーションとサーバアプリケーションの対話を示す図である。

【図8】本発明に従うチャンキングプロシージャの一実施例についてのプロセスフローを示す図である。

【図9】本発明に従うチャンキングプロシージャの一実施例についての命令コードの実施例を示す図である。

【図10】本発明の少なくとも一態様に従って構成されるチャンキングプロシージャの他

50

の実施例についての命令コードの他の実施例を示す図である。

【図 1 1】本発明の少なくとも一態様に従って構成されるチャンキングプロシーダの他の実施例についての命令コードの他の実施例を示す図である。

【符号の説明】

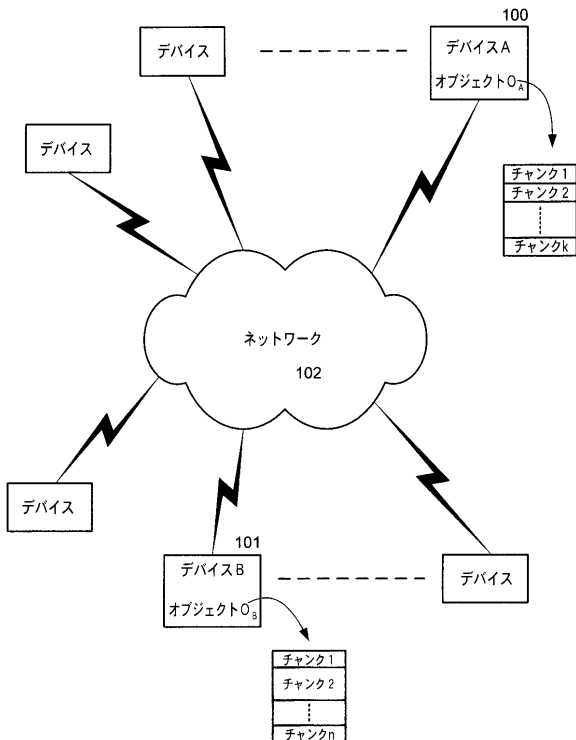
【0 1 4 3】

- 1 0 0 デバイス A
- 1 0 1 デバイス B
- 1 0 2 ネットワーク
- 2 0 0 コンピューティングデバイス
- 2 0 2 処理装置
- 2 0 4 システムメモリ
- 2 0 5 オペレーティングシステム
- 2 0 6 プログラムモジュール
- 2 0 7 プログラムデータ
- 2 0 9 着脱可能ストレージ
- 2 1 0 着脱不能ストレージ
- 2 1 2 1つ(または複数)の入力デバイス
- 2 1 4 1つ(または複数)の出力デバイス
- 2 1 6 1つ(または複数)の通信接続
- 2 1 8 他のコンピューティングデバイス

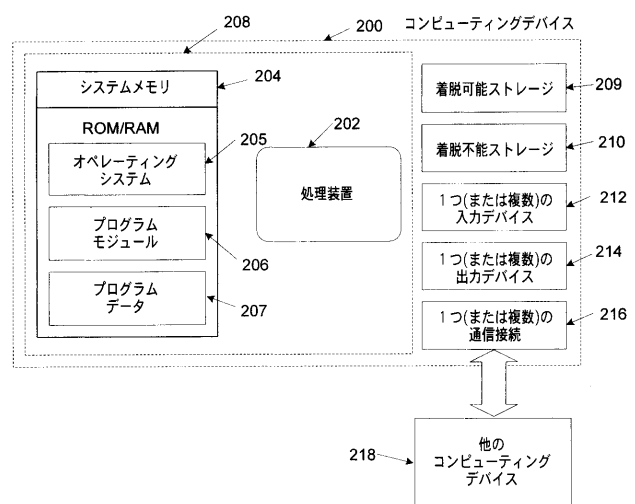
10

20

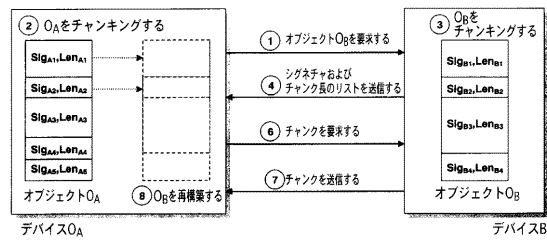
【図 1】



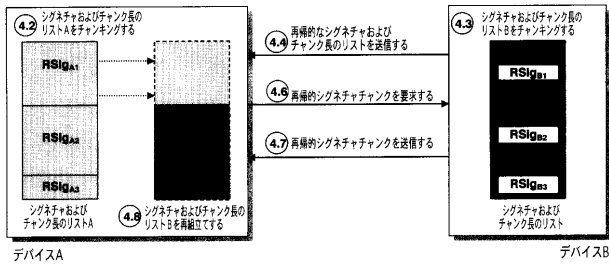
【図 2】



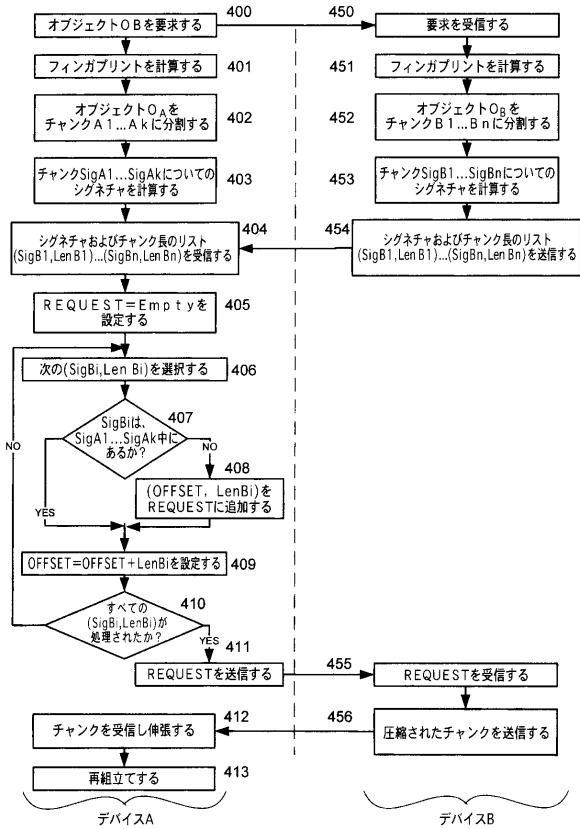
【図 3 A】



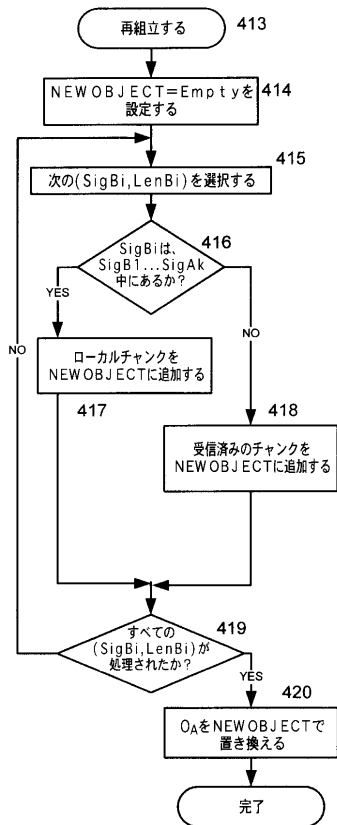
【図 3 B】



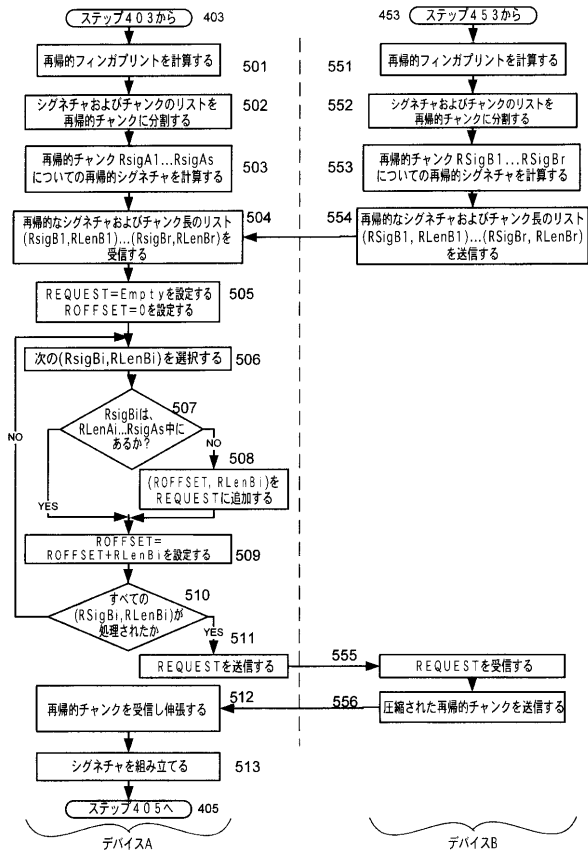
【図 4 A】



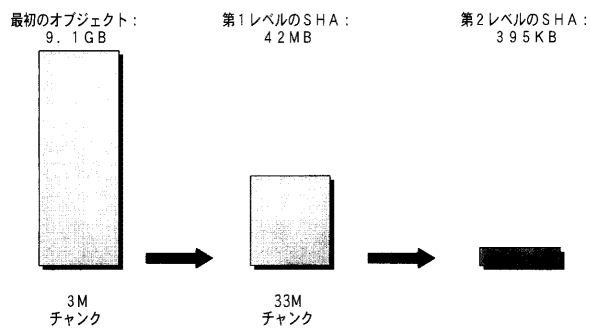
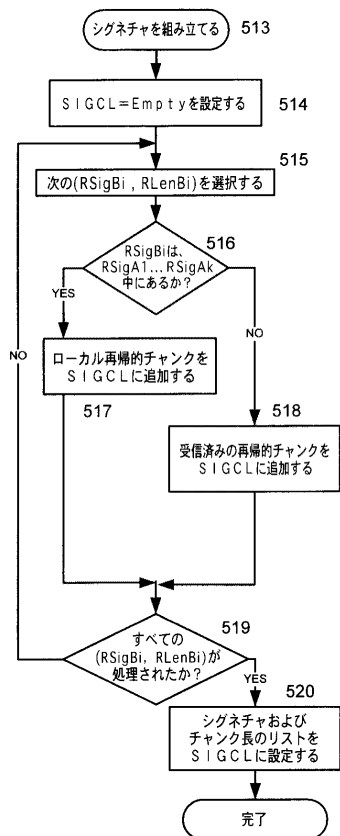
【図 4 B】



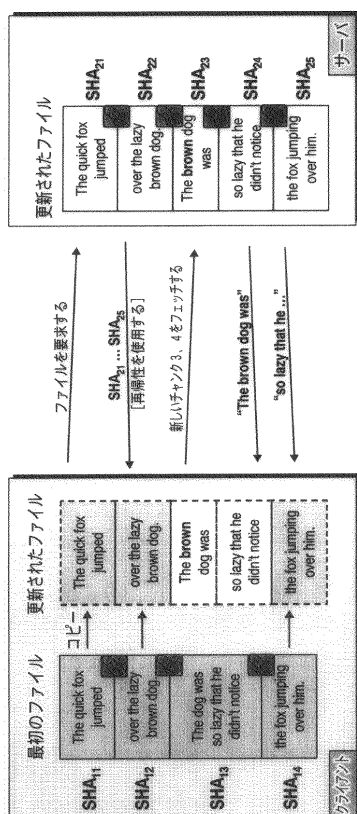
【図 5 A】



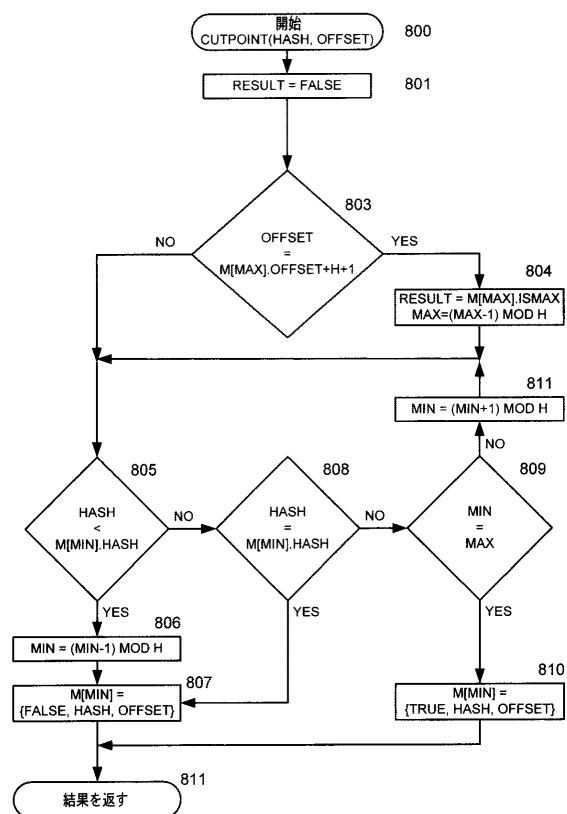
【图 6】



【图 7】



【图 8】



【図 9】

```

structure Entry
  var isMax as Boolean = false
  var hash as Integer = 0
  var offset as Integer = 0

class LocalMaxCut
  h as Integer
  var min as Integer = 0
  var max as Integer = 0
  var M as Array of entry = new Entry[h]

  CutPoint(hash as Integer, offset as Integer) as Boolean
    var result = false
    step
      if M[max].offset + h + 1 = offset then
        result := M[max].isMax
        max := (max+1) mod h
    step
    while true do step
      step
      if M[min].hash > hash then
        step
        min := (min-1) mod h
        step
        M[min] := Entry(false, hash, offset)
        return result
      if M[min].hash = hash then
        M[min] := Entry(false, hash, offset)
        return result
      if M[min].hash < hash and min = max then
        M[min] := Entry(true, hash, offset)
        return result
    step
    min := (min+1) mod h

```

【図 10】

```

structure Entry
  var offset as Integer = 0
  var isMax as Boolean = false
  var hash as Integer = 0

class LocalMaxCut
  horizon as Integer
  var hashes as Seq of Integer
  var k as Integer = 0
  var l as Integer = 0
  var A as Array of Entry = new Entry[horizon]
  var B as Array of Entry = new Entry[horizon]

  CutPoints() as Seq of Integer
    var cuts as Seq of Integer = []
    for window = 0 to Length(hashes)/horizon do step
      let first = window*horizon
      let last = min((window+1)*horizon, Length(hashes))-1
      cuts := cuts + CutPoint(first, last)
    return cuts

  CutPoint(first as Integer, last as Integer) as Seq of Integer
    step // Initialize A with the first entry at the offset
    k := 0
    A[0] = Entry(last, true, hashes[last])
    last := last - 1
    step // Update A[k] in the interval up to B[l]'s horizon
    while last > B[l].offset + horizon do step
      Insert(last)
      last := last - 1
    step // Update A[k] and B[l] in the remaining interval
    while last >= first do step
      Insert(last)
      if B[l].hash <= hashes[last] then
        B[l].isMax := false
        last := last - 1
    step // determine whether A[k] is a cutpoint with respect to B
    A[k].isMax := A[k].isMax and
      forall j in {0..} holds
        (B[j].offset + horizon < A[k].offset or
         B[j].hash < A[k].hash)
    step // Set B to A for the next round and return cut-point
    B, l := A, k
    return if B[l].isMax then [B[l].offset] else []

```

【図 11】

```

class LocalMaxCut
  Insert(offset as Integer)
    if hashes[offset] >= A[k].hash then
      if hashes[offset] = A[k].hash then
        // duplicated hashes within distance
        // of "horizon" are not maximal
        A[k].isMax := false
      else
        A[k+1] := Entry(offset, true, hashes[offset])
        k := k + 1

```

フロントページの続き

- (72)発明者 ニコライ エス. ビョルナー
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 パトリック イー. ボーズマン
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内
- (72)発明者 ユーリ グレビッチ
アメリカ合衆国 98052 ワシントン州 レッドモンド ワン マイクロソフト ウェイ マ
イクロソフト コーポレーション内

Fターム(参考) 5B082 GA01 HA03 HA08

【要約の続き】