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

特許第4268337号
(P4268337)

(45)発行日　平成21年5月27日(2009.5.27)　(24)登録日　平成21年2月27日(2009.2.27)

(51)Int.Cl.　F I
H04L 12/56 (2006.01)　H04L 12/56　100Z
H04L 12/46 (2006.01)　H04L 12/46　100R

請求項の数 23　(全 18 頁)

(21)出願番号　特願2000-581783(P2000-581783)
(86)(22)出願日　平成11年8月31日(1999.8.31)
(65)公表番号　特表2002-530011(P2002-530011A)
(43)公表日　平成14年9月10日(2002.9.10)
(86)国際出願番号　PCT/US1999/019995
(87)国際公開番号　WO2000/028704
(87)国際公開日　平成12年5月18日(2000.5.18)
審査請求日　平成17年9月29日(2005.9.29)
(31)優先権主張番号　09/188,014
(32)優先日　平成10年11月6日(1998.11.6)
(33)優先権主張国　米国(US)

(73)特許権者　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　９８０５２－６３９９　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　ドレイヴス，リチャード・ピー
アメリカ合衆国ワシントン州９８０２２，シアトル，サーティエイス・アベニュー　６０２

最終頁に続く

(54)【発明の名称】最適ルーティング・テーブル圧縮のためのルータ及び方法

(57)【特許請求の範囲】

【請求項１】

ルーティング・テーブルを圧縮する方法であって、

前記ルーティング・テーブルのバイナリ・ツリー表現を構築するステップであって、前記バイナリ・ツリーは前記ルーティング・テーブルにおける様々なルートを表すノードを有している、ステップと、

次のホップを前記ノードに割り当てるステップと、

普及している次のホップを前記ツリーの上方向に移動させるステップと、

前記ツリーにおける冗長分岐を排除し、出力ツリーを生じさせるステップと、

前記出力ツリーを新たなルーティング・テーブルに変換するステップと、を含み、

前記移動させるステップは、Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従って普及している次のホップを前記ツリーの上方向に移動させるステップを含み、ただし、Ａ＊Ｂは、親ノードにおいて、前記親ノードに対応する１対の子ノードに対して次のホップＡ及びＢの集合から形成される次のホップの集合であることを特徴とする方法。

【請求項２】

ルーティング・テーブルを圧縮する方法であって、

前記ルーティング・テーブルのバイナリ・ツリー表現を構築するステップであって、前記バイナリ・ツリーは前記ルーティング・テーブルにおける様々なルートを表すノードを有している、ステップと、

次のホップを前記ノードに割り当てるステップと、
普及している次のホップを前記ツリーの上方向に移動させるステップと、
前記ツリーにおける冗長分岐を排除し、出力ツリーを生じさせるステップと、
前記出力ツリーを新たなルーティング・テーブルに変換するステップと、を含み、
前記移動させるステップは、最初のノードで開始し最後のノードで終了する一連の単一分岐に圧縮経路を適用するステップであって、Ｘは前記最後のノードにおける次のホップの集合を表し、Ｙは前記最初のノードにおける次のホップの集合を表すものとして、Ｘ∩Ｙ＝φならば、Ｙを最初のノードに、Ｘを最後のノードに割り当て、Ｘ∩Ｙ≠φならば、Ｘ∩Ｙを、最初のノードと最後のノードとに割り当てるという演算に従って、次のホップを前記ツリーの上方向に移動させるステップを含むことを特徴とする方法。

【請求項３】
請求項１記載の方法において、前記排除するステップは、
最も近接した空でない祖先ノードから次のホップを選択するステップと、
前記親ノードから分岐している子ノードを調べ、前記選択された次のホップが前記子ノードに対する次のホップの要素であるかどうかを判断し、そうである場合には、前記子ノードに対する前記次のホップを排除するステップと、
を含むことを特徴とする方法。

【請求項４】
ルーティング・テーブルのバイナリ・ツリー表現を圧縮する方法であって、前記バイナリ・ツリーは、複数のレベルにおいて、前記ルーティング・テーブルにおける様々なルートを表す複数のノードを有しており、あるレベルにおける親ノードは次のより下位のレベルにおけるゼロ、１又は２の子ノードに分岐することができ、前記ノードのいくつかは関連する次のホップを有している、方法において、
子ノードを１つだけ有する親ノードに対して新たな子ノードを作成し、次のホップを前記親ノードから前記新たな子ノードに割り当てることによって、前記ツリーを正規化するステップと、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従い、子ノードの対応する対に対する次のホップＡ及びＢの集合を親ノードにおいてスーパーネットすることによって、普及している次のホップを前記ツリーの上方向に移動させるステップであって、ただし、Ａ＊Ｂは、前記親ノードにおいて生じる次のホップの集合である、ステップと、
前記ツリーを下方向にサブネットして子ノードを有する冗長分岐を排除するステップであって、少なくとも１つの次のホップが、対応する最も近接した空でない祖先ノードの次のホップの集合に含まれている、ステップと、
を含むことを特徴とする方法。

【請求項５】
請求項４記載の方法において、前記正規化し、移動させ及びサブネットするステップは、前記ツリーを通過する３つのパスにおいて実行されることを特徴とする方法。

【請求項６】
請求項４記載の方法において、前記正規化し移動させるステップは、前記ツリーを通過する単一のパスにおいて実行されることを特徴とする方法。

【請求項７】
請求項４記載の方法において、個別の親ノードに対してＡ＊Ｂ＝Ａ∩Ｂである場合には、前記個別の親ノードと関連する前記１対の子ノードにおける次のホップを除去するステップを更に含むことを特徴とする方法。

【請求項８】
請求項４記載の方法において、前記サブネットするステップは、
最も近接した空でない祖先ノードから次のホップを選択するステップと、
前記親ノードから分岐している子ノードを調べ、前記選択された次のホップが前記子ノードに対する次のホップの要素であるかどうかを判断し、そうである場合には、前記子ノ

ードに対する前記次のホップを排除するステップと、
を含むことを特徴とする方法。
【請求項９】
請求項８記載の方法において、前記選択するステップは、前記ツリー構造の変更を回避することを試みる次のホップを選択するステップを含むことを特徴とする方法。
【請求項１０】
ルーティング・テーブルのバイナリ・ツリー表現を圧縮する方法であって、前記バイナリ・ツリーは、複数のレベルにおいて、前記ルーティング・テーブルにおける様々なルートを表す複数のノードを有しており、あるレベルにおける親ノードは次のより下位のレベルにおけるゼロ、１又は２の子ノードに分岐することができる、方法において、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従い、普及している次のホップを前記ツリーの上方向に移動させるステップであって、ただし、Ａ＊Ｂは、親ノードにおいて、前記親ノードに対応する１対の子ノードに対する次のホップＡ及びＢの集合から形成される次のホップの集合である、ステップと、
前記ツリーにおける冗長分岐を排除するステップと、
を含むことを特徴とする方法。
【請求項１１】
請求項１０記載の方法において、個別の親ノードに対してＡ＊Ｂ＝Ａ∩Ｂである場合には、前記排除するステップに先立って、前記個別の親ノードと関連する前記１対の子ノードにおける次のホップを除去するステップを更に含むことを特徴とする方法。
【請求項１２】
請求項１０記載の方法において、前記排除するステップは、
最も近接した空でない祖先ノードから次のホップを選択するステップと、
前記親ノードから分岐している子ノードを調べ、前記選択された次のホップが前記子ノードに対する次のホップの要素であるかどうかを判断し、そうである場合には、前記子ノードに対する前記次のホップを排除するステップと、
を含むことを特徴とする方法。
【請求項１３】
請求項１２記載の方法において、前記選択するステップは、前記ツリー構造の変更を回避することを試みる次のホップを選択するステップを含むことを特徴とする方法。
【請求項１４】
ルーティング・テーブルを処理する方法であって、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従って人気のある次のホップを移動させることによって（ただし、Ａ＊Ｂは、次のホップＡ及びＢの集合から形成される次のホップの集合である）、より特定のルートからより一般的なルートに人気のある次のホップの集合を伝搬させるステップであって、より特定のルートは、より一般的なルートに対するＩＰアドレスと比較してプレフィクスのビット数が多いＩＰアドレスによって定義される、ステップと、
人気のある次のホップの集合から次のホップを選択して、前記より特定のルートを再定義するステップと、
を含むことを特徴とする方法。
【請求項１５】
請求項１４記載の方法において、共通のＩＰアドレスを有する冗長ルートをすべて排除するステップを更に含むことを特徴とする方法。
【請求項１６】
ルーティング・テーブルを圧縮する方法であって、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従って人気のある次のホップを移動させることによって（ただし、Ａ＊Ｂは、次のホップＡ及びＢの集合から形成される次のホップの集合である）、人気のある次のホップの集合をスーパーネットするステップと、

選択された次のホップを人気のある次のホップの集合からサブネットして、冗長ルートを排除するステップと、
を含むことを特徴とする方法。
【請求項１７】
ルーティング・テーブルを記憶するメモリと、
前記メモリに結合されており、前記ルーティング・テーブルにリスト化されているルートに従ってメッセージをルーティングするプロセッサと、
前記メモリに記憶されている前記ルーティング・テーブルを圧縮するテーブル圧縮器であって、前記ルーティング・テーブルのバイナリ・ツリー表現を構築して次のホップを前記ツリーにおけるノードに割り当て、普及している次のホップを前記ツリーの上方向に移動させて前記ツリーにおける冗長分岐を排除し、前記ツリーを圧縮されたルーティング・テーブルに変換して戻す、テーブル圧縮器と、を備え、
前記テーブル圧縮器は、Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従って、次のホップを前記ツリーの上方向に移動させ、ただし、Ａ＊Ｂは、親ノードにおいて、前記親ノードに対応する１対の子ノードに対して次のホップＡ及びＢの集合から形成される次のホップの集合であることを特徴とするルータ。
【請求項１８】
ルーティング・テーブルを記憶するメモリと、
前記メモリに結合されており、前記ルーティング・テーブルにリスト化されているルートに従ってメッセージをルーティングするプロセッサと、
前記メモリに記憶されている前記ルーティング・テーブルを圧縮するテーブル圧縮器であって、前記ルーティング・テーブルのバイナリ・ツリー表現を構築して次のホップを前記ツリーにおけるノードに割り当て、普及している次のホップを前記ツリーの上方向に移動させて前記ツリーにおける冗長分岐を排除し、前記ツリーを圧縮されたルーティング・テーブルに変換して戻す、テーブル圧縮器と、を備え、
前記テーブル圧縮器は、最初のノードで開始し最後のノードで終了する一連の単一分岐に経路圧縮技術を適用し、Ｘは前記最後のノードにおける次のホップの集合を表し、Ｙは前記最初のノードにおける次のホップの集合を表すものとして、Ｘ∩Ｙ＝φならば、Ｙを最初のノードに、Ｘを最後のノードに割り当て、Ｘ∩Ｙ≠φならば、Ｘ∩Ｙを、最初のノードと最後のノードとに割り当てるという演算に従って、次のホップを前記ツリーの上方向に移動させることを特徴とするルータ。
【請求項１９】
請求項１８記載のルータにおいて、前記テーブル圧縮器は、最も近接した空でない祖先ノードから次のホップを選択し、前記親ノードから分岐している子ノードを調べ、前記選択された次のホップが前記子ノードに対する次のホップの要素であるかどうかを判断し、そうである場合には、前記子ノードに対する前記次のホップを排除することを特徴とするルータ。
【請求項２０】
ルーティング・テーブルを記憶するメモリと、
前記メモリに結合されており、前記ルーティング・テーブルにリスト化されているルートに従ってメッセージをルーティングするプロセッサと、
前記メモリに記憶されている前記ルーティング・テーブルを圧縮し、より特定のルートからより一般的なルートに人気のある次のホップの集合を伝搬させるテーブル圧縮器であって、より特定のルートは、より一般的なルートに対するＩＰアドレスと比較してプレフィクスのビット数が多いＩＰアドレスによって定義され、Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従って、前記人気のある次のホップを移動させ（ただし、Ａ＊Ｂは、次のホップＡ及びＢの集合から形成される次のホップの集合である）、人気のある次のホップの集合から次のホップを選択して、前記より特定のルートを再定義し、共通のＩＰアドレスを有する冗長ルートを排除する、テーブ

ル圧縮器と、
を備えていることを特徴とするルータ。
【請求項２１】
コンピュータ可読な媒体であって、
子ノードを１つだけ有する親ノードに対して新たな子ノードを作成し、次のホップを前記親ノードから前記新たな子ノードに割り当てることによって、前記ツリーを正規化するステップと、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従い、子ノードの対応する対に対する次のホップＡ及びＢの集合を親ノードにおいてスーパーネットすることによって、普及している次のホップを前記ツリーの上方向に移動させるステップであって、ただし、Ａ＊Ｂは、前記親ノードにおいて生じる次のホップの集合である、ステップと、
前記ツリーを下方向にサブネットして子ノードを有する冗長分岐を排除するステップであって、少なくとも１つの次のホップが、対応する最も近接した空でない祖先ノードの次のホップの集合に含まれている、ステップと、
を実行するコンピュータ実行可能な命令を記憶していることを特徴とするコンピュータ可読な媒体。
【請求項２２】
コンピュータ可読な媒体であって、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従い、普及している次のホップを前記ツリーの上方向に移動させるステップであって、ただし、Ａ＊Ｂは、親ノードにおいて、前記親ノードに対応する１対の子ノードに対する次のホップＡ及びＢの集合から形成される次のホップの集合である、ステップと、
前記ツリーにおける冗長分岐を排除するステップと、
を実行するコンピュータ実行可能な命令を記憶していることを特徴とするコンピュータ可読な媒体。
【請求項２３】
コンピュータ可読な媒体であって、
より特定のルートからより一般的なルートに人気のある次のホップの集合を伝搬させるステップであって、より特定のルートは、より一般的なルートに対するＩＰアドレスと比較してプレフィクスのビット数が多いＩＰアドレスによって定義される、ステップと、
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪ＢでありＡ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂであるという演算に従って前記人気のある次のホップを移動させるステップと（ただし、Ａ＊Ｂは、次のホップＡ及びＢの集合から形成される次のホップの集合である）、
人気のある次のホップの集合から次のホップを選択して、前記より特定のルートを再定義するステップと、
共通のＩＰアドレスを有する冗長ルートをすべて排除するステップと、
を実行するコンピュータ実行可能な命令を記憶していることを特徴とするコンピュータ可読な媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、ネットワーク・ルータに関し、更に詳しくは、ルータがネットワーク・メッセージをルーティングするのに用いられるルーティング・テーブルを圧縮する方法に関する。
【０００２】
【従来の技術】
ルータは、ネットワーク上でデジタル・メッセージのルーティング（経路決定）を行う特別の計算装置である。ルータは、ある位置（例えば、発信元（ソース）コンピュータや、別のルータ）からメッセージを受け取り、そのメッセージを、最も効率的で利用可能な経

路を介して次の目的地（例えば、宛先（デスティネーション）コンピュータや、別のルータ）に送る。

【０００３】

ルータは、ルーティング・テーブルを用いて、ネットワーク上でメッセージ・トラフィックの方向を決める。ルーティング・テーブルは複数（例えば、数千から数万）のエントリを有しており、それぞれのエントリは宛先装置のアドレスである。例えば、インターネットの場合には、それぞれのエントリは、「１９２．５６．７．４８」などの３２ビットのＩＰ（インターネット・プロトコル）アドレスで構成されている。ＩＰアドレスは、８、１６又は２４など、予め定められた長さを有することもありうる。この予め定められた長さは、ルータが次の経路として何ビットを考慮すべきかを特定している。例えば、ＩＰアドレス「１９２．５６．７．４８」に１６という予め定められた長さが割り当てられている場合には、ルータは、次の経路を決定するのに最初の２バイトだけを考慮すればよいのであって、このＩＰアドレスを「１９２．５６．０．０」と読むことができる。

【０００４】

ルーティング・テーブルのそれぞれのエントリには、第２のテーブルへのインデクスを有する「次のホップ」値が関連付けられている。この第２のすなわち「次のホップ」のテーブルは、ネットワークにおける次のルータ又は宛先ポートのＩＰアドレスを識別するためのエントリをより少数だけ（例えば、１０ないし１００のエントリ）有している。例えば、ＩＰアドレス「１９２．５６．７．４８」は、「１７」という関連付けがなされた次のホップ値を有することがありうるが、これは、次のルータに対するＩＰアドレスが次のホップ・テーブルの中の位置１７にあることを意味する。

【０００５】

ルータは、ルーティング・テーブルと次のホップ・テーブルとを用いて、ネットワークを介してメッセージを最も効率的にルーティングする。パケットがあるルータに到達すると、このルータは、まず、宛先アドレスへの最も近接した一致を与えるアドレスをルーティング・テーブルにおいて見つける。宛先アドレスと比較して一致するプレフィクスが最も長いＩＰアドレスが選択されるというのが、一般的な規則である。次に、ルータは、選択されたＩＰアドレスと関連する次のホップを見つけ、その次のホップによってインデクスが付された次のホップ・テーブルにおけるＩＰアドレスによって指定される次のルータに向けてパケットをルーティングする。ルーティング及び次のホップ・テーブルにおいてアドレスを素早く見出す高速のアルゴリズムには多くの異なるものが存在する。

【０００６】

【発明が解決しようとする課題】

ワールド・ワイド・ウェブ上で利用可能になっているコンテンツが爆発的に増加していることにより、インターネットは、急速に成長を続け、世界のあらゆる片隅にまで到達している。インターネットのバックボーン・ルータの要求は、他と比較不可能なほどに増加している。ネットワークには、極めて速い速度で、ノードが追加されつつある。インターネットのバックボーンにおけるルートの数は、年に１万個の割合で増えつつあると考えられている。開発されるルータがより複雑かつ高速であるのならば、増加するルータの数に対応するために、より大きなルーティング・テーブルを備えていることになる。ルータの速度と信頼性とを向上させるには、ルータの機能性又は効率性を損なうことのないようにルーティング・テーブルのサイズを縮小することが望ましい。

【０００７】

この目的を達成するために、本発明は、ルーティング・テーブルのサイズを縮小する技術に向けられている。

【０００８】

【課題を解決するための手段】

本発明は、ルータにおいて用いられるルータ・テーブルを圧縮するようなルータ及び方法に関する。この方法によれば、より小型であるが機能的には同等なテーブルが作成される。

【０００９】
１つの実現形態では、ルーティング・テーブルを圧縮する方法は、そのルーティング・テーブルのバイナリ・ツリー表現を構築することを含む。このバイナリ・ツリーは複数のノードを有しており、あるレベルの分岐における親ノードは、それよりも低い次のレベルにはゼロ、１つ又は２つの子ノードを有している。この圧縮方法は、このツリーを介する３つの通過を行う。第１の通過では、この圧縮方法は、ツリーの葉ノード（すなわち、それよりも下位の子ノードには分岐しないノード）に向けて、下向きにルーティング情報を伝搬させる。この通過の間には、プログラムが、ツリーにおけるすべての葉ノードに、それよりも上位の祖先（ancestor）ノードから関連付けされた次のホップ又は相続された次のホップを割り当てる。
【００１０】
第２の通過では、この圧縮方法は、より普及している（prevalent）次のホップをツリーの上方向に移動させる。このボトムアップの通過は、次の演算に従って、ある親ノードにおいて、その親ノードに対応する１対の子ノードに対する次のホップＡ及びＢの集合をスーパーネットすることによって、次のホップの集合を形成することを含む：
【００１１】
【数１】
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪Ｂであり、
Ａ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂである。
ただし、Ａ＊Ｂは、親ノードにおいて形成された次のホップの集合である。
【００１２】
第３の通過では、この圧縮方法は、ツリーにおける冗長分岐を除去する。このトップダウンのパスは、根ノードと次いでそれぞれの親ノードとにおいて始まり、親ノードから次のホップを選択する。この方法は、次に、親ノードから分岐する子ノードを調べ、選択された次のホップがその子ノードに対する次のホップの要素であるかどうかを判断する。そうである場合には、この方法は、その子ノードに対するホップを除去する。
【００１３】
これらの３つのパスによるプロセスがツリーを再構成した後で、この圧縮方法は、ツリーを新たなルーティング・テーブルに変換する。
【００１４】
【発明の実施の形態】
本発明は、ルーティング・テーブルを、サイズはそれよりも小型だが機能的には同等なテーブルに圧縮するルータ及び方法に関する。ネットワーク構造の一般的な状況とルータとについて最初に説明し、次いで、圧縮プロセスについて説明する。
【００１５】
典型的なネットワーク・アーキテキチャ
図１は、ネットワーク２６を介して接続された発信元（ソース）コンピュータ２２と宛先（デスティネーション）コンピュータ２４とを有するコンピュータ・ネットワーク・システム２０を示している。ネットワーク２６は、相互に接続された複数のルータ２８（１）、２８（２）、・・・、２８（Ｎ）を備えている。発信元コンピュータ２２から発せられるメッセージはネットワーク２６上でルーティングされ、ルータ２８（１）－２８（Ｎ）の中の選択されたものを通過して宛先コンピュータ２４に至る。ネットワーク２６は、公共ネットワーク（例えば、インターネット）、ＬＡＮ（ローカル・エリア・ネットワーク）及びＷＡＮ（ワイド・エリア・ネットワーク）を表している。以下の議論では、このシステムを、発信元及び宛先コンピュータのＩＰアドレスを含むＩＰ（インターネット・プロトコル）パケットに含まれる形態でメッセージが転送されるインターネットの場合を用いて説明する。
【００１６】
それぞれのルータ２８は、大きなルーティング・テーブル３０と凝縮された次のホップ・テーブル３２とを維持している。ルーティング・テーブルはネットワーク上の他のノード

（すなわち、ルータ及び又はコンピュータ）のＩＰアドレスを保持し、それに対して、次のホップ・テーブル３２は直接に結合されている隣接するノードのＩＰアドレスを保持している。ルーティング・テーブル３０におけるそれぞれのＩＰアドレスは、次のホップ・テーブル３２における１つ又は複数の次のホップ・アドレスを指示している。

【００１７】

典型的なルータ

図２は、典型的なルータ２８を示している。このルータは、ネットワーク内の他のノードとのＩ／Ｏインターフェースを提供する。ルータ２８は、また、マイクロプロセッサ４２とメモリ４４（例えば、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、ハードディスクなど）とを有する。ルーティング・テーブル３０と次のホップ・テーブル３２とは、メモリ４４に記憶される。

【００１８】

ルータ２８は、既存のルーティング・テーブルを処理しそれをサイズは縮小されているが機能的には同等のテーブルに圧縮する技術を用いて構成されている。図解されている実現例では、ルーティング・テーブル圧縮プログラム４８は、メモリ４４に記憶されており、プロセッサ４２上で実行されてルーティング・テーブルを圧縮する。別の実現例では、このテーブル圧縮技術は、特定用途向け集積回路（ＡＳＩＣ）やそれ以外のＩＣ装置において具体化されている。

【００１９】

図２に示されているアーキテクチャは、汎用コンピュータなど、他の計算装置を表す場合もあることに注意すべきである。

一般的な圧縮プロセス

図３は、ルーティング・テーブルの圧縮に関係する一般的なプロセス・ステップを示している。ステップ５０では、圧縮プログラム４８が、ルーティング・テーブル３０を、ルーティング・テーブルにおけるアドレス・プレフィクスを表すバイナリ・ツリー構造に変換する。ツリーは、圧縮プロセスの間、揮発性メモリ４４に記憶される。

【００２０】

図４は、バイナリ・ツリー構造７０を示している。この構造は、複数のレベルに構成されたノード７２の階層を有する。３２ビットのＩＰアドレス・プレフィクスの場合、ツリー構造には、３３のレベル（すなわち、ｋ＝０ないし３２）が存在する。ほとんどのノードは、それよりも下位レベルにある２つのノードに分岐し、一方の分岐がバイナリ「１」の条件を表し、他方の分岐がバイナリ「０」の条件を表す。これらの２つの下位レベルのノードは、共通のすなわち「親」ノードの「子」ノードと称される。共通の親ノードを有する２つのノードは、「きょうだい（sibling）」ノードと称される。子のないノードは、その分岐に対する終点を形成しているので、「葉」と称される。

【００２１】

ＩＰアドレスのプレフィクスにおけるそれぞれの連続ビットは、ツリーにおける親ノードから子ノードへのリンクに対応する。プレフィクスにおける「０」ビットは、左側の子ノードに対応し、「１」ビットは右側の子ノードに対応する。ノード７２には、より小さい整数又はより小さい整数の組であるのが典型的な次のホップ値がラベル付けされている。

【００２２】

ステップ５２では、圧縮プログラム４８が、ツリー構造７０を圧縮する。圧縮プログラム４８は、サブステップ５４、５６及び５８によって表されているように、３つのパス（three-pass）によるプロセスを用いる。第１のパス（ステップ５４）では、圧縮プログラムは、ルーティング情報を下向きにツリーの葉まで伝搬させる。このパスの間は、プログラムは、ツリーにおけるすべての葉ノードに、関連する次のホップ、すなわち、より高位のレベルにある祖先ノードから相続した次のホップを割り当てる。

【００２３】

第２のパス（ステップ５６）では、圧縮プログラムは、次のホップの集合を葉ノードから上向きに根ノードに向かって浸透させることを含むスーパーネット・プロセス（supernet

ting process）によって、最も普及している（most prevalent）次のホップを見つける。プログラムは、最下位のレベルｋにおける葉ノードから開始して、最下位から最上位までツリー７０を移動する。圧縮プログラムは、２つのきょうだいノードＡ及びＢを調べ、それらの次のホップを、次のように定義される演算＊を用いて、親ノードに割り当てる。すなわち、
ＡとＢとの共通部分が空でない場合には、Ａ＊Ｂ＝Ａ∩Ｂであり、
ＡとＢとの共通部分が空である場合には、Ａ＊Ｂ＝Ａ∪Ｂとする。

【００２４】

Ａ＊Ｂの結果は、親ノードによって形成される次のホップの集合である。第２のパスの効果は、最も人気のある(popular)次のホップ値をツリーの最上位まで移動させることである。ルーティング・テーブルの場合には、スーパーネット・プロセスが、最も人気のある次のホップの集合を、より特定されたルート（すなわち、より長いＩＰプレフィクスを有するルート）から、より一般的なルート（すなわち、より短いＩＰプレフィクスを有するルート）に移動させる。

【００２５】

第３のパス（ステップ５８）では、圧縮プログラムが、トップダウン式のサブネット・プロセス（subnetting process）を用いて冗長ルートを排除する。第３のパスは、ツリー７０の根ノード（すなわち、レベル０）で始まり、可能性のある値の組から次のホップを選択する。圧縮プログラムは、次のようにして、それぞれの下位のノードを反復的にテストする。

【００２６】

テスト：選択された親の次のホップが、子の次のホップの集合の要素である場合には、子の次のホップの集合は取り除かれ、そうでない場合には、子の次のホップの集合から新たな次のホップを選択する。

【００２７】

新たな子の次のホップは、インテリジェントに又はランダムに、選択することができる。ここでインテリジェントなアプローチとは、ツリー構造をゲインなしに不必要に変更しない傾向を有する状態を選択しようとすることである。第３のパスは、ツリーから有用でない分岐及びノードを効果的に排除し、それにより、ツリー構造全体を縮小する。

【００２８】

ステップ６０では、圧縮プログラムが、ツリーを変換して、ＩＰアドレスのルーティング・テーブルに戻す。新たなルーティング・テーブルは、元のテーブルよりも小さいが、それ以外の点では、機能的に元のテーブルと同等である。

【００２９】

これら３つのパスによるプロセスは、多くの効果を奏する。直感的に、ツリーの根における又は根に近くにおけるプレフィクスがより短ければ、最も人気のある又は普及している次のホップにルーティングされるはずである。ツリーの葉の近くにあるプレフィクスがより長ければ、普及の程度がより低い次のホップにルーティングされるはずである。この３つのパスによるプロセスは、最も普及している次のホップを上向きにツリーの根に向かって移動させ、普及程度がより低い次のホップを下向きに葉に向かって移動させるのと同時に、ツリーから最大の数のルートを剪定する。

【００３０】

この圧縮プロセスは、最長のプレフィクスの一致を用いてＩＰアドレスへの発送情報（forwarding information）を提供するルーティング・テーブルが与えられた場合にルーティング・テーブルを提供するのであるが、このルーティング・テーブルは、（ａ）同じ発送情報を提供し、（ｂ）可能な限り最小の数のエントリを有する。このプロセスは、ＩＰｖ４のプレフィクスを用いて大きなバックボーン・ルータにおけるプレフィクスの数を約４０％減少させるという実験において示された。

【００３１】

圧縮ステップ５２の詳細な例

図５は、簡略化されているが、しかし、４つのルートに対するアドレス・プレフィクスの集合から構築された典型的なバイナリ・ツリー８０を示している。次の表１には、次のホップのテーブル３２を指示しているルートと関連する次のホップとのリストが挙げられている。また、表１は、次のホップの値を用いてラベル付けがなされた対応するノードのリストも有している。

【００３２】

【表１】

IP Address/Next Hop	Node
* → 1	82
00* → 2	84
10* → 2	86
11* → 3	88

根ノード８２は、デフォルトのルートが次のホップ値１によって示されているゼロのプレフィクスを表している。ノード８４はアドレス・プレフィクス００＊を表しており、このプレフィクスと関連付けられた次のホップ値は２である。ノード８６は、アドレス・プレフィクス１０＊を表し、関連付けられた次のホップ値は２である。ノード８８は、アドレス・プレフィクス１１＊を表し、関連付けられた次のホップ値は３である。

【００３３】

パス１：次のホップを葉ノードに割り当てる（ステップ５４）

圧縮ステップ５２（図３のステップ５４）の第１のパスは、第２及び第３のパスを準備するために、ルーティング・テーブルのバイナリ・ツリー表現を正規化する。これにより、ツリーにおけるすべてのノードが有する子ノードの数がゼロ又は２のいずれかであることが保証される。第１のパスは、新たな葉ノードを作成し、新たなノードのためにその次のホップを初期化する。ただし、この新たなノードは、次のホップを１つ有する直近の祖先からこの新たなノードが相続する次のホップを有している。

【００３４】

第１のパスは、バイナリ・ツリーのプレオーダ移動（pre-order traversal）か、又は、その代わりに、根から下位へのレベルの移動を用いることがありうる。いずれの場合にも、この移動は、次のホップ値を、親ノードから、次のホップを有していない子ノードに向かって下向きに押して、親ノードが子をただ１つしか有していないときには、新たな子ノードを作成する。

【００３５】

図６は、第１のパスが完了した後のバイナリ・ツリー８０を示している。第１のパスが、プレフィクス０１＊によってアドレシングされる新たな葉ノード９０を作成したことに注意してほしい。この新たな葉ノード９０は、その最も近接した祖先の次のホップを相続する。これは、この場合には、祖父母の根ノード８２からの次のホップ値１である。いったんツリーが葉ノードで完全に満たされると、内部ノードに対する次のホップはもはや問題とならず、廃棄されることがある。この例では、根ノード８２に対する次のホップ値１が、廃棄される。

【００３６】

パス２：スーパーネット（ステップ５６）

圧縮ステップ５２の第２のパス（すなわち、図３のステップ５６）は、人気のある（popular）次のホップの集合をツリーの上方向に浸透させることによって、ルーティング・テーブルの最も普及している（most prevalent）次のホップを計算する。第２のパスは、葉ノード８４、８６、８８及び９０から始めて、ボトムアップ式の移動を採用している。このボトムアップ式の移動の際に遭遇するそれぞれの親ノードでは、その親ノードに対する次のホップＡ＊Ｂの集合が、次の演算に従って、子ノードＡ及びＢにおいて見いだされる次のホップに基づいて計算される。

【００３７】

【数２】
Ａ∩Ｂ＝φならばＡ＊Ｂ＝Ａ∪Ｂであり、
Ａ∩Ｂ≠φならばＡ＊Ｂ＝Ａ∩Ｂである。
【００３８】
２つの子ノードの間に共通な次のホップが１つでも存在する（すなわち、Ａ∩Ｂ≠φ）場合には、それが、親ノードのレベルで最も普及している次のホップである。従って、その共通な次のホップだけが、共通部分をとる演算Ａ∩Ｂに従って、親ノードまで上向きに移動される。そうではなく、２つの子ノードの間に共通な次のホップが全く存在しない（すなわち、Ａ∩Ｂ＝φ）場合には、それらの子ノードからの次のホップすべてが、和集合をとる演算Ａ∪Ｂに従って、親ノードまで運ばれる。第２のパスが完了すると、ツリーにおけるすべてのノードが、潜在的な次のホップの集合を用いてラベル付けされる。
【００３９】
図７は、第２のパスが完了した後のバイナリ・ツリー８０を示している。きょうだいノード８４及び９０に対しては、「Ａ」はノード８４の次のホップの集合（すなわち、｛２｝）であり、「Ｂ」はノード９０の次のホップの集合（すなわち、｛１｝）である。第２のパスは、集合Ａ及びＢが共通な次のホップを全く有していないと判断し、従って、これらの集合の共通部分｛１，２｝をその親ノード９２まで運ぶ。同様に、きょうだいノード８６及び８８に対しては、「Ａ」はノード８６の次のホップの集合（すなわち、｛２｝）であり、「Ｂ」はノード８８の次のホップの集合（すなわち、｛３｝）であるから、第２のパスは、これらの集合は共通な次のホップを全く有していないことを見いだし、従って、これらの集合の共通部分｛２，３｝をその親ノード９４まで運ぶ。
【００４０】
この時点で、「Ａ」がノード９２の次のホップの集合（すなわち、｛１，２｝）であり、「Ｂ」がノード９４の次のホップの集合（すなわち、｛２，３｝）である、きょうだいノード９２及び９４に対して、第２のパスは、これらの集合が２という共通の次のホップを有していることを見いだして、これらの２つの集合の共通部分（すなわち、｛２｝）を親である根ノード８２に伝搬させる。第２のパスが、最も普及している次のホップ２を根ノード８２まで効率的に移動させていることに注意してほしい。
【００４１】
パス３：サブネット（ステップ５８）
圧縮ステップ５２の第３のパス（すなわち、図３のステップ５６）は、ツリーを下向きに移動し、サブネットを介して、プレフィクスに対する次のホップを選択し、冗長ルートを排除する。第３のパスは、ツリーのプレオーダーな移動か、又は、根から下向きへのレベルごとの移動のいずれかを用いることができる。途中で遭遇するそれぞれのノードは、可能性のある次のホップの集合を有している。これは、第２のパスで計算されたものである。根ノードを除いて、ノードは、次のホップを１つ有する最も近接した祖先ノードから次のホップを相続する。相続された次のホップがそのノードが有する潜在的な次のホップの集合に含まれる場合には、ノードは、それ自身の次のホップを必要とせず、適切な次のホップを相続していることになる。しかし、相続された次のホップがそのノードが有する潜在的な次のホップの集合に含まれない場合には、ノードは、次のホップを必要とする。そのノードが有する潜在的な次のホップの任意のものを、そのノードの次のホップとして選択することができる。このプロセスは、次のテストによって総括される。
【００４２】
テスト：選択された親である次のホップが、子の次のホップの集合の要素である場合には、子の次のホップの集合は取り除かれ、そうでない場合には、子の次のホップの集合から新たな次のホップが選択される。
【００４３】
第２のパスの後では、根ノードは、要素が１つである次のホップの集合｛２｝によってラベル付けされ、従って、第３のパスは、この根のために次のホップ２を選択する。次のホップ２は２つの子ノード９２及び９４の集合｛１，２｝及び｛２，３｝の要素であるから

、根の子ノードにおける次のホップのこれらの集合は取り除くことができる。ノード９２において開始する２つの子ノード８４及び９０は、それぞれ、集合｛２｝及び｛１｝を有する。このプロセスにより、子ノード８４に対して次のホップ２が、子ノード９０に対して次のホップ１が選択される。同様に、親ノード９４に対しては、このプロセスによって、子ノード８６に対して次のホップ２が、子ノード８８に対して次のホップ３が選択される。

【００４４】

図８は、トップダウン式のテスト手順がすべてのノードを訪問した後での第３のパスの間のバイナリ・ツリー８０を示している。第３のパスが進行するにつれて、ツリー８０は、テーブルを圧縮するために取り除くことができる冗長ルートを有することがありうる。この例では、２つの葉ノード８４及び８６は、次のホップを必要としない。その理由は、次のホップが根の集合に含まれているからである。従って、これらの葉ノードは、根から適切な次のホップを相続することができ、冗長であるとして、ツリーから取り除くことができる。あるいは、これらの冗長ルートは、第３のパスの最後に取り除くこともできる。

【００４５】

図９は、第３のパスが完了した後のバイナリ・ツリー８０を示している。最終的なツリーは、２つの葉ノード８８及び９０と、３つのルート＊、０１＊及び１１＊とを有している。表２には、ルートと、圧縮されたツリーに対する関連する次のホップ値とのリストが、対応するノードと共に示されている。

【００４６】

【表２】

IP Address/Next Hop	Node
* → 2	82
01* → 1	90
11* → 3	88

図５のツリー及び表１によって表現されている元のルーティング・テーブルと比較すると、図９及び表２によって表現される結果的なツリーは、有しているルートが１つ少なく、より普及している次のホップ２が葉ノードの位置ではなく根ノードの位置にあるという点で、より効率的である。

【００４７】

この３つのパスによる圧縮プロセスは、元のルーティング・テーブル（図５）と同じ発送行動を依然として維持しながら、可能な限り最小の数のプレフィクスを有する出力ルーティング・テーブル（図９）を与える。第３のパスでは、圧縮プログラムは、潜在的な次のホップの集合から次のホップを選択することができる。これは、このプロセスが与えられた入力テーブルに対して多くの異なる出力ルーティング・テーブルを生じさせることができるということを意味している。しかし、可能性がある出力ルーティング・テーブルは、すべてが、同じサイズで機能的に同等である。

【００４８】

別の実現例

圧縮プロセスにおけるステップ数を減らすことによってパフォーマンスを向上させる方法は複数存在する。１つの方法として、第１及び第２のパスを合成することがある。合成されたパスが子ノードをただ１つだけ有する親ノードに出会うと、このプロセスは、その時点で、新たな子ノードを作成して、相続された次のホップを新たな子に割り当てる。新たな子とこの新たな子が相続している祖先との間の任意の中間的なノードには、この相続された次のホップを割り当てることができる。これは、以後の相続演算の高速化に役立つ。

【００４９】

例えば、図６におけるツリー構造を考察しよう。圧縮プログラムは、新たな子ノード９０を作成する第１のパスの間に、次のホップ２を親ノード９２に割り当てることができた。

【００５０】
別のパフォーマンス向上によれば、第２のパスにおいて予測することにより、第３のパスにおける作業をいくらか節約することができる。第２のパスでは、親ノードに、その子ノードが有する潜在的な次のホップの集合の共通部分が割り当てられると、これら２つの子ノードに対する次のホップ値を除去することができる。例えば、図７における子ノード９２及び９４を考察しよう。親ノード８２には、これらのノードの次のホップである｛１，２｝及び｛２，３｝の共通部分である集合｛２｝が割り当てられる。第２のパスにおけるこの時点で、圧縮プログラムは、ノード９２及び９４において集合｛１，２｝及び｛２，３｝を取り除くことができる。というのは、そうでなくても、これらは、図８に示されているように、第３のパスの間に削除されるからである。

【００５１】
このパフォーマンスの向上は、直ちに、ルーティング・テーブルからこれらのプレフィクスを取り除くことになる。２つの集合の共通部分の要素は定義によって両方の集合の要素であるから、これは、安全な最適化である。第３のパスでは、親ノードには、この共通部分からの次のホップが割り当てられる（あるいは、そのような次のホップを相続する）。第３のパスは、子ノードを処理する場合には、それらが潜在的な集合の要素である次のホップを相続することを理解して、その時点でそれらを取り除く。

【００５２】
別のパフォーマンスの向上では、場合によっては、ルーティング・テーブルを「不必要に」変更せずに、圧縮することが好ましいことがありうる。第３のパスにおける次のホップの選択に２つの小さな修正を行うことにより、テーブルを不必要に変更しないように、安定性が向上する。第３のパスの間に、圧縮プログラムが、親ノードに対する次のホップを、潜在的な次のホップの集合Ｘから選択しなければならないと想定する。このノードのプレフィクスが入力ルーティング・テーブルの中に次のホップを有しており、それがＸの要素である場合には、それは、論理的な選択である。これによって、安定性が向上する。というのは、このプレフィクスの次のホップが保存されるからである。

【００５３】
圧縮プログラムが親ノードに対する次のホップを選択しなければならないが、そのプレフィクスが入力ルーティング・テーブルの中に次のホップを有していない場合には、安定性を向上させるために、そのプレフィクスに次のホップを割り当てる代わりに、そのプレフィクスを最適化されたルーティング・テーブルから取り除くことを望むであろう。親ノードが有する潜在的な次のホップの集合がその２つの子ノードである集合の和集合（union）によって形成される場合には、これは、安全な修正である。いずれの方法であっても、親ノードとその２つの子ノードとは、出力ルーティング・テーブルにおいて、２つのルートを発生する。

【００５４】
図５ないし図９とを参照しながら以上で説明した例では、本質的に、２つの仮定を行っている。第１の仮定は、入力ルーティング・テーブルがデフォルトのルート（ゼロ・プレフィクスに対する次のホップ）を含む、というものである。第２の仮定は、入力テーブルがそのプレフィクスに対して次のホップをただ１つ有する、というものである。しかし、上述したプロセスは、これらの仮定が成立しない状況においても用いることができる。

【００５５】
第１の仮定については、インターネットにおけるバックボーン・ルータには、「デフォルト・フリー」のルーティング・テーブルを用いているものがある。そのような場合には、１つのアプローチとして、０のダミーの次のホップへのデフォルトのルートを、第１のパスの最初において導入することがある。第３のパスの最後では、根におけるダミー・ルートは、出力テーブルに存在する場合には、取り除かれる。出力テーブルは次のホップ０へのルートを含みうることを注意すべきである。次のホップ０への発送は、一致するプレフィクスがちょうど見つからないのと同じように、単純に、エラーであると解釈される。

【００５６】

入力テーブルが次のホップをただ１つ含むという第２の仮定については、ルーティング・テーブルが次のホップを複数用いる場合がある。複数の次のホップを提供する方法は複数存在する。第１に、圧縮手順を適用して結果的なルーティング・テーブルを最適化する前に、何らかの計量（metric）を用いることにより、それぞれのプレフィクスに対して最良の次のホップを選択することが可能である。この計量によって、あるプレフィクスに対する次のホップの集合から最良の次のホップを１つ選択することができるのであれば、これは、適切な方法である。

【００５７】

複数の次のホップが最良のものを選択する際に互角である場合には、圧縮プロセスは、よりよい圧縮を達成するためにその中から選択できる次のホップを複数有することによって獲得する融通性を用いることができる。この技術では、入力テーブルは複数の次のホップを含む。第１のパスが僅かに修正されることにより、新たな子ノードがその祖先から複数の次のホップを相続することができる。このアプローチによれば、圧縮プロセスがより大きな圧縮を達成することが可能となる。その理由は、このアプローチが、普及している次のホップをツリーのより高いレベルにおいて見つける機会を多く有するからである。しかし、このアプローチでは、次のホップの集合は、入力ルーティング・テーブルの中に保存される。

【００５８】

次のホップの集合を入力ルーティング・テーブルに保存することが重要である場合には、「仮想的な」次のホップを作成するという修正された別のアプローチがある。ここで、仮想的な次のホップは、それぞれが、入力ルーティング・テーブルにおいて見いだされる次のホップの異なる集合を表している。圧縮プログラムは、仮想的な次のホップの使用を最適化し、そうすることで、次のホップの集合を操作する代わりに、実際には、次のホップの集合の集合を操作している。

【００５９】

経路圧縮の最適化

向上のための別の修正として、バイナリ・ツリーの複数の部分に経路圧縮技術を選択的に適用して圧縮速度を高速化するというものがある。図１０は、第１及び第２のパスが終了した後での入力ツリー１００と処理済のツリー１０２とを示している。入力ツリー１００は、次のホップ集合Ｄを備えた根ノードと、次のホップ集合Ａ及びＢを備えた葉ノードとを有している。根ノードと葉ノードとの間には、多くの単一分岐ノードが存在する。圧縮方法の第１のパスは、子をただ１つ有するそれぞれの親から複数の子ノードを作成し、最も近い祖先から相続された次のホップの集合を割り当てる。このようにして、新たな子ノードすべてに、処理済のツリー１０２に示されているように、次のホップの集合Ｄが割り当てられる。

【００６０】

第２のパスをツリーに適用することによって、もっとも普及している次のホップの集合が上向きに移動することになる。この場合には、次のホップの集合Ｄは、最終的に、根ノードＤに至ることになる。多くの重複的なステップが、この最終結果を達成することになる。更に、この作業は浪費されるように見えるのであるが、その理由は、根ノードにおける次のホップの集合Ｄが第１及び第２のステップの結果として変化していないからである。

【００６１】

圧縮を実行する演算の数は、式Ｏ（ｎｗ）によって与えられる。ここで「Ｏ」は、演算数の上界（upper bound）が「ｎｗ」に比例することを本質的に意味する広く理解されている記号である。変数「ｎ」は、バイナリ・ツリーによって表されるプレフィクス（又は、ルート）の数であり、変数「ｗ」はプレフィクスの長さである。３２ビットのＩＰアドレスの場合には、演算の数は３２ｎとなる。

【００６２】

経路圧縮技術は、更に演算数をＯ（ｎｗ）からＯ（ｎ）に減少させるのに用いることができる。図１１は、入力ツリー１００の凝縮された表現１１０を示している。ボックス１１

２は、単一分岐ノード及びドット１１４のストリングがこのストリングの最後で子ノードを表す前の親ノードを表している。曲線状の経路１１６は、分岐のストリングを表す。変数Ｘはノード１１４における次のホップの集合（すなわち、Ａ∪Ｂ）を記述し、変数Ｙはノード１１２における次のホップの集合（すなわち、Ｄ）を記述している。この構造に遭遇すると、圧縮プログラムは次の演算を行う。

【００６３】

【数３】

Ｘ∩Ｙ＝φならば、Ｙを最初のノードに、Ｘを最後のノードに、割り当てる。
Ｘ∩Ｙ≠φならば、Ｘ∩Ｙを、最初のノードと最後のノードとに割り当てる。

【００６４】

この場合には、「ドット」ノード１１４における次のホップＸの集合（すなわち、Ａ∪Ｂ）は、「ボックス」ノード１１２における集合Ｄと共通の値を有さない。従って、「ボックス」ノード１１２は、集合Ｄに割り当て、そして「ドット」ノード１１４は、集合Ｘに割り当てる。

【００６５】

圧縮ツリー１００においては、「ｎ」このプレフィクスがあるとき、多くて３ｎ個のノードがあり得るだけである。従って、演算の最大数は、高々、「３ｎ」である。ボックス・ノードは、その中の圧縮された経路の長さが少なくとも２であるときにだけ、用いられる。それ以外の場合には、ドット・ノードが用いられる。

【００６６】

結論

以上では、圧縮技術を、ルータにおけるルーティング・テーブルを圧縮する場合に即して説明した。より一般的に、同じ技術を適用して、プレフィクスの集合のサイズを最適に縮小することができる。この場合、プレフィクスは、最長の一致するプレフィクス・アルゴリズムに従って選択されている。

【００６７】

以上では、本発明は、構造的な特徴及び又は方法論的なステップに特有の言語を用いて説明されているが、冒頭の特許請求の範囲において定義されている本発明は、説明されたこれらの特定の特徴やステップには必ずしも限定されないことを理解すべきである。むしろ、これら特定の特徴やステップは、特許請求されている発明を実現する好適な形態として開示されている。

【図面の簡単な説明】

【図１】　コンピュータ・ネットワーク・システムの図解である。

【図２】　ルータのブロック図である。

【図３】　ルーティング・テーブルを圧縮する方法におけるステップを示す流れ図である。

【図４】　ルーティング・テーブルにおけるルートを表現するバイナリ・ツリー構造の図解である。

【図５】　４つのルートを表現するバイナリ・ツリーの図解である。

【図６】　圧縮方法の第１のパスの後での図５のバイナリ・ツリーの図解である。

【図７】　圧縮方法の第２のパスの後での図５のバイナリ・ツリーの図解である。

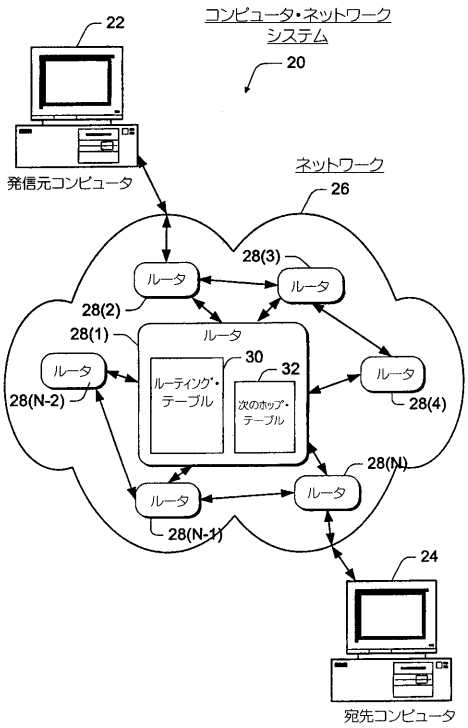
【図８】　圧縮方法の第３のパスの間での図５のバイナリ・ツリーの図解である。

【図９】　圧縮方法の第３のパスの後での図５のバイナリ・ツリーの図解である。

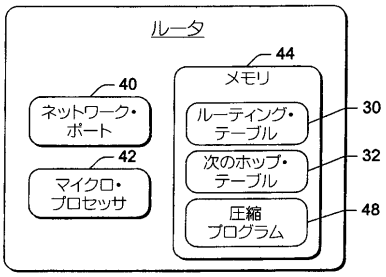
【図１０】　複数の単一分岐を有するバイナリ・ツリーの一部と、圧縮方法における第１及び第２のパスの結果としてそのツリーがどのように処理されるかとを示す図解である。

【図１１】　複数の単一分岐を有するバイナリ・ツリーの一部と、それらの単一分岐が経路圧縮を用いてどのように圧縮されうるかとを示す図解である。

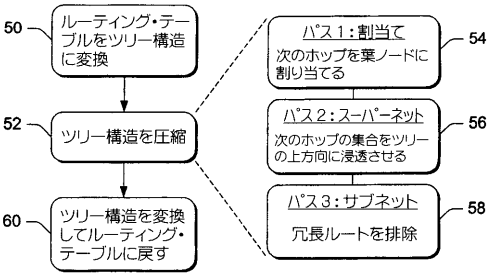
【図１】



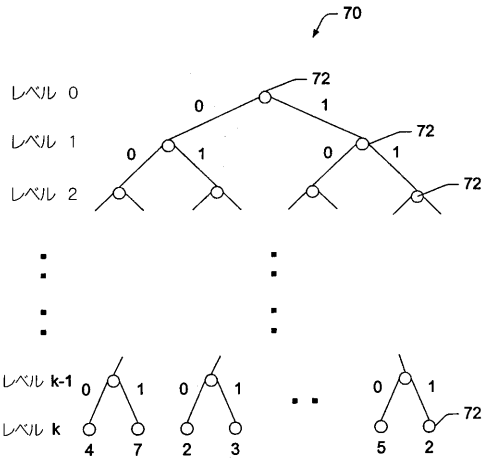
【図２】



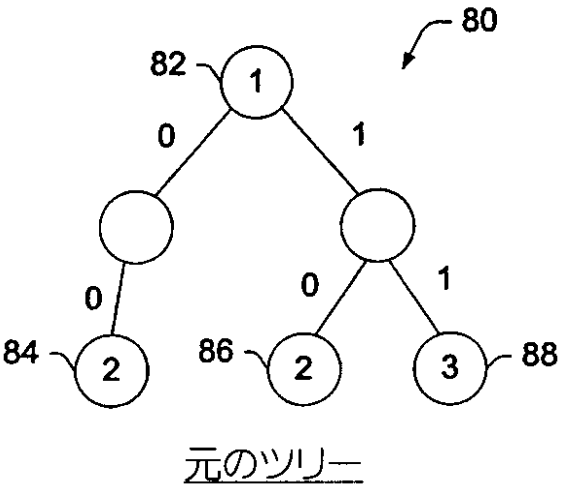
【図３】



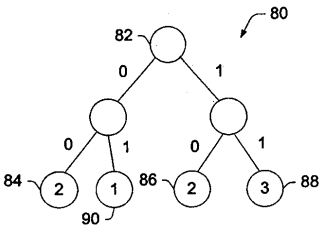
【図４】



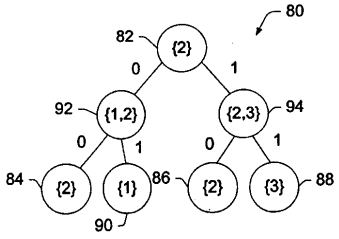
【図５】



【図６】

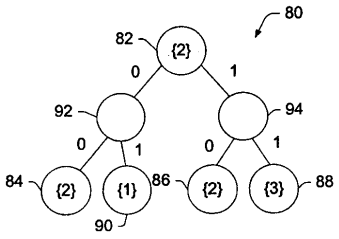


【図７】



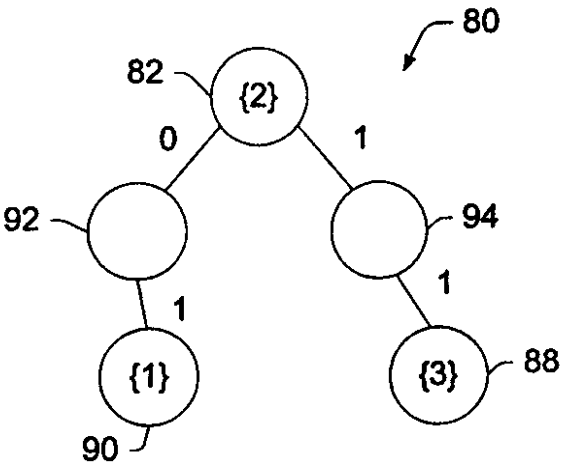
パス２の後のツリー

【図８】



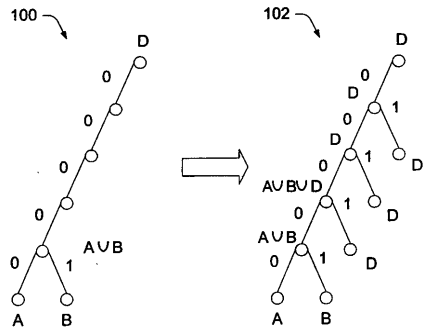
パス３の間のツリー

【図９】



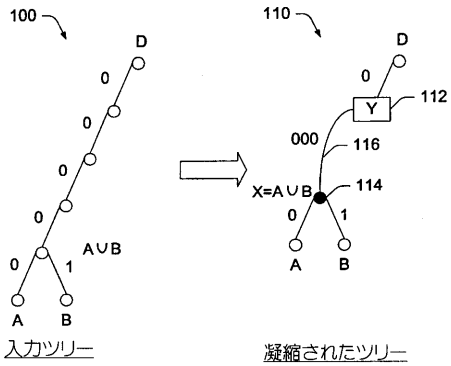
パス３の後のツリー

【図１０】



入力ツリー　　第1及び第2のパスの後のツリー

【図１１】



入力ツリー　　凝縮されたツリー

フロントページの続き

(72)発明者　キング，クリストファー・エス
アメリカ合衆国マサチューセッツ州０２１４４，サマーヴィル，キングストン・ストリート　２４
(72)発明者　ヴェンカタチャリー，スリニヴァサン
アメリカ合衆国ミシガン州６３１１２，セント・ルイス，ウォーターマン・ブルヴァード　５６３６，ナンバー　１１

審査官　吉田　隆之

(56)参考文献　1998 ACM SIGMETRICS, p1-10
IEEE INFOCOM'99, p88-97
ACM Computer Communications Review, Vol.27 No.4, p3-14
ACM Computer Communications Review, Vol.27 No.4, p25-36
IEEE Journal on Selected Areas in Communications, Vol.17 No.6, p1083-1092

(58)調査した分野(Int.Cl.，ＤＢ名)
H04L 12