(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4786908号
(P4786908)
(45)発行日　平成23年10月5日(2011.10.5)　(24)登録日　平成23年7月22日(2011.7.22)

(51)Int.Cl.　F I
H04L 12/26 (2006.01)　H04L 12/26

請求項の数 15　外国語出願　(全 35 頁)

(21)出願番号　特願2005-24158(P2005-24158)
(22)出願日　平成17年1月31日(2005.1.31)
(65)公開番号　特開2005-223906(P2005-223906A)
(43)公開日　平成17年8月18日(2005.8.18)
審査請求日　平成20年1月31日(2008.1.31)
(31)優先権主張番号　60/540,738
(32)優先日　平成16年1月30日(2004.1.30)
(33)優先権主張国　米国(US)
(31)優先権主張番号　10/881,695
(32)優先日　平成16年6月30日(2004.6.30)
(33)優先権主張国　米国(US)

前置審査

(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　110001243
特許業務法人　谷・阿部特許事務所
(74)復代理人　100115624
弁理士　濱中　淳宏
(74)復代理人　100130524
弁理士　藤田　英治

最終頁に続く

(54)【発明の名称】障害検出および診断

(57)【特許請求の範囲】
【請求項１】
管理機能を備える少なくとも１つのマネージャノードを含む複数のノードを備えたネットワークにおける障害の検出および診断のための方法であって、
前記ネットワークのオペレーションにおける食い違いを検出するステップと、
前記食い違いの根本原因を診断するステップと、を備え、
前記検出するステップは、
前記ノードのデータ収集モジュールが、実際のネットワークの挙動を記述しているトレースデータを前記実際のネットワークのオペレーションを通じて収集して、前記マネージャノードのネットワークシミュレータに供給するステップと、
前記ネットワークシミュレータが、前記トレースデータに基づいて実際のネットワークのネットワークシミュレーションを実行して前記実際のネットワークのパフォーマンスの値を推定するステップと、
前記マネージャノードの分析モジュールが、シミュレートされたネットワークパフォーマンスの前記推定値と、前記トレースデータに含まれる実際のネットワークの挙動を示すネットワークパフォーマンスの実際の値との差が、所定のしきい値以上であるかに基づいて、ネットワークオペレーションの食い違いが発生しているかどうかを判定するステップと、を備え、
前記診断するステップは、前記食い違いが発生していると判定された場合に、前記マネージャノードの分析モジュールが、前記推定値が前記実際の値に収斂するか否かの判断に

基づいて前記推定値が前記実際の値に近似すると判定されるまで、ネットワークの障害となる可能性がある複数の障害のうち１つまたは複数を前記ネットワークシミュレータに順次投入してネットワークシミュレーションを繰り返し実行させることにより、前記２つのパフォーマンス値の食い違いの根本原因と考えられる１つまたは複数の障害を判定し、

前記ネットワークシミュレーションを繰り返し実行させる場合、前記マネージャノードの分析モジュールは、

前記推定値と前記実際の値との差と、所定の障害のタイプに対応するしきい値との比較結果に基づき、前記１つまたは複数の障害のタイプを特定し、前記特定した障害のタイプごとに、

前記推定値と前記実際の値との差に基づいて変換された障害の大きさを前記特定した障害のタイプに対応するしきい値と比較し、

前記障害の大きさが前記対応するしきい値よりも小さい場合は、小さいと判定された障害を、食い違いの根本原因ではないと判定して、前記ネットワーク障害となる可能性がある障害から削除し、

削除後の前記推定値が前記実際の値に収斂すると判断された場合は、前記推定値が前記実際の値に近似すると判定され、収斂しないと判断された場合は、削除後の前記推定値と前記実際の値との比較結果に基づいて予想される前記ネットワークの障害となる可能性がある障害を追加する

ことを特徴とする方法。

【請求項２】

前記診断するステップは、

初期障害診断を行って初期障害集合を生成するステップと、

現在の障害集合に達するように前記初期障害集合を繰り返し精密化するステップと、をさらに備え、

前記現在の障害集合は、前記ネットワークシミュレータへの入力として使用された場合に、前記ネットワークシミュレータにより供給されるネットワークパフォーマンスの前記推定値で前記ネットワークパフォーマンスの実際の値を近似させる障害集合である

ことを特徴とする請求項１に記載の方法。

【請求項３】

前記供給するステップで収集するトレースデータは、前記複数のノードのうちの発信ノードと目標ノードとの間の、前記ネットワークを通じてデータパケットにより実行される一連のホップにより形成されるルートを記述することを特徴とする請求項１に記載の方法。

【請求項４】

前記供給するステップで収集するトレースデータは、ルーティングを含むネットワークトポロジのデータと、トラフィック統計量のデータと、物理的媒体のデータと、ネットワークパフォーマンスのデータとからなるグループから選択されることを特徴とする請求項１に記載の方法。

【請求項５】

コンピュータ実行可能命令を含むコンピュータ可読記憶媒体であって、前記命令がコンピュータによって実行された場合、コンピュータは請求項１に記載の方法を実行することを特徴とするコンピュータ可読記憶媒体。

【請求項６】

通信できるように順次に結合される複数のノードで形成される現実のネットワークにおいて、障害の検出および診断を行うためのシステムであって、

１つまたは複数の前記ノードは、実際のネットワークの挙動を記述し、前記ネットワーク上で通信するためのネットワーク設定を含むトレースデータを収集するエージェントモジュールを含み、

少なくとも１つの前記ノードは、前記エージェントモジュールからトレースデータを受け取り、前記トレースデータに基づいて前記ネットワークのシミュレーションを実行する

マネージャモジュールを備え、
前記マネージャモジュールは、
前記シミュレーションによって得られたネットワークパフォーマンスの推定結果と、前記トレースデータに含まれる実際のネットワークの挙動を示すネットワークパフォーマンスの観察結果との差が、所定のしきい値以上であるかに基づいて、ネットワークオペレーションの食い違いが発生しているかどうかを判定し、
前記食い違いが発生していると判定された場合に、前記推定結果が前記観察結果に収斂するか否かの判断に基づいて前記推定結果が前記観察結果に近似すると判定されるまで、ネットワークの障害となる可能性がある複数の障害のうち１つまたは複数を順次投入してネットワークシミュレーションを繰り返し実行させることにより、前記前記推定結果と前記観察結果との食い違いの根本原因と考えられる１つまたは複数の障害を判定し、
前記ネットワークシミュレーションを繰り返し実行させる場合、前記マネージャノードの分析モジュールは、
前記推定値と前記実際の値との差と、所定の障害のタイプに対応するしきい値との比較結果に基づき、前記１つまたは複数の障害のタイプを特定し、前記特定した障害のタイプごとに、
前記推定値と前記実際の値との差に基づいて変換された障害の大きさを前記特定した障害のタイプに対応するしきい値と比較し、
前記障害の大きさが前記対応するしきい値よりも小さい場合は、小さいと判定された障害を、食い違いの根本原因ではないと判定して、前記ネットワーク障害となる可能性がある障害から削除し、
削除後の前記推定値が前記実際の値に収斂すると判断された場合は、前記推定値が前記実際の値に近似すると判定され、収斂しないと判断された場合は、削除後の前記推定値と前記実際の値との比較結果に基づいて予想される前記ネットワークの障害となる可能性がある障害を追加する
ことを特徴とするシステム。
【請求項７】
少なくとも２つの前記ノードは、それぞれの前記マネージャモジュールを備えることを特徴とする請求項６に記載のシステム。
【請求項８】
それぞれの前記ノードは、それぞれの前記エージェントモジュールを備えることを特徴とする請求項６に記載のシステム。
【請求項９】
前記マネージャモジュールは、さらに、
前記検出された障害を是正するための是正処置を導出し、
前記エージェントモジュールに前記是正処置を実行させる
ことを実行可能とすることを特徴とする請求項６に記載のシステム。
【請求項１０】
前記マネージャモジュールは、さらに、
初期障害診断を行って初期障害集合を生成し、
現在の障害集合に達するように前記初期障害集合を繰り返し精密化することにより前記障害を診断し、
前記現在の障害集合は、前記ネットワークシミュレータへの入力として使用された場合に、前記ネットワークシミュレータにより供給されるネットワークパフォーマンスの前記推定値で前記ネットワークパフォーマンスの実際の値を近似させる障害集合である
ことを特徴とする請求項６に記載のシステム。
【請求項１１】
前記ネットワークは、メッシュネットワークであることを特徴とする請求項６に記載のシステム。
【請求項１２】

前記ネットワーク設定は、前記ネットワークの発信ノードと目標ノードとの間の、前記ネットワークを通じてデータパケットにより実行される一連のホップにより形成されるルートを記述することを特徴とする請求項６に記載のシステム。

【請求項１３】

通信できるように順次に結合される複数のノードで形成される現実のネットワークにおいて、障害を検出しおよび診断する手段を有する前記ノードであって、

データパケットのルーティングを行うための複数の手段を備えるネットワークのオペレーションを管理する手段を備え、

それぞれの前記ルーティングのための手段は、他の前記ルーティングのための手段に通信できるように結合され、

前記管理手段は、さらに、

実際のネットワークの挙動を記述しているトレースデータを前記実際のネットワークのオペレーションを通じて収集する手段と、

収集された前記トレースデータの供給を受けて、実際のネットワークのネットワークシミュレーションを実行して前記実際のネットワークのパフォーマンスの推定結果を出力するシミュレーション手段と、

シミュレートされたネットワークパフォーマンスの前記推定結果と、前記トレースデータに含まれる実際のネットワークの挙動を示すネットワークパフォーマンスの観察結果との差が、所定のしきい値以上であるかに基づいて、ネットワークオペレーションの食い違いが発生しているかどうかを判定する手段と、

前記食い違いが発生していると判定された場合に、前記推定結果が前記観察結果に収斂するか否かの判断に基づいて前記推定結果が前記観察結果に近似すると判定されるまで、ネットワークの障害となる可能性がある複数の障害のうち１つまたは複数を前記シミュレーション手段に順次投入してネットワークシミュレーションを繰り返し実行させることにより、前記推定結果と観察結果との食い違いの根本原因と考えられる１つまたは複数の障害を判定する手段とを備え、

前記ネットワークシミュレーションを繰り返し実行させる場合、前記シミュレーション手段は、

前記推定値と前記実際の値との差と、所定の障害のタイプに対応するしきい値との比較結果に基づき、前記１つまたは複数の障害のタイプを特定し、前記特定した障害のタイプごとに、

前記推定値と前記実際の値との差に基づいて変換された障害の大きさを前記特定した障害のタイプに対応するしきい値と比較し、

前記障害の大きさが前記対応するしきい値よりも小さい場合は、小さいと判定された障害を、食い違いの根本原因ではないと判定して、前記ネットワーク障害となる可能性がある障害から削除し、

削除後の前記推定値が前記実際の値に収斂すると判断された場合は、前記推定値が前記実際の値に近似すると判定され、収斂しないと判断された場合は、削除後の前記推定値と前記実際の値との比較結果に基づいて予想される前記ネットワークの障害となる可能性がある障害を追加する

ことを特徴とするノード。

【請求項１４】

前記ネットワークは、メッシュネットワークであることを特徴とする請求項１３に記載のノード。

【請求項１５】

前記管理する手段は、さらに、前記検出された障害を是正するための是正処置を導出する手段を備えることを特徴とする請求項１３に記載のノード。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、一般に、有線および無線ネットワークに関係しており、より具体的には、ネットワークに発生した障害の検出および診断のためのネットワークトラブルシューティングフレームワークに関する。

【背景技術】

【０００２】

ネットワーク管理（Ｎｅｔｗｏｒｋ　ｍａｎａｇｅｍｅｎｔ）は、マルチホップ無線ネットワークの配備を成功させるうえで欠かせない要素であるのに、業界も、研究機関もこれにはたいして注目していなかった。ネットワークのトラブルシューティングは、ネットワークの「健全な状態」を維持し、オペレーションを円滑に、継続して行えるようにする役割を持つネットワーク管理の一側面である。有線、無線を問わず、ネットワークのトラブルシューティングは、異なるネットワークエンティティの絡み合い、異なる障害の発生などで生じる相互作用により込み入った作業になる。

【０００３】

マルチホップ無線ネットワークのトラブルシューティングは、さまざまな要因が加わるためさらに込み入ってくる。例えば、典型的なマルチホップ無線ネットワークは、一般に、信号伝播の変動によって引き起こされるリンクエラーを生じやすい。信号伝播の変動の原因として、変動する環境条件などのさまざまな要因が考えられる。このような変動があると、ネットワークトポロジは動的で、予測不可能になる。ノードには機動性がある、つまり、ノードはネットワークに接続されたまま位置を次々に変えてゆく可能性があり、それによってネットワークの動的な予測不可能な性質が強まるため、これらの要因の重大性はいっそう深まる。さらに、マルチホップ無線ネットワークの容量は、一般的に、資源（例えば、帯域幅、電池の電力など）の希少性ゆえに限られており、ネットワークで許容できる管理トラフィックのオーバーヘッドの大きさが制約される。さらに、無線ネットワークは、悪意を持った第三者からのリンク攻撃に弱いと考えられる。例えば、攻撃者は、偽情報を流して、ネットワーク管理作業を攪乱したり妨害したりすることがある。

【発明の開示】

【発明が解決しようとする課題】

【０００４】

従来、ネットワークトラブルシューティングを実行するために使用されていた伝統的な発見的、理論的な手法では、「現実の」環境内で実行されたときのネットワークの挙動が捕捉されないのがふつうである。例えば、ネットワークの挙動は、次から次へのノードの間の相互作用だけでなくノードの近傍内に置かれている外部ノイズ源にも左右される。従来の発見的または理論的手法では、その周辺の環境も含めたネットワークのさまざまなコンポーネント間の相互作用を十分に取り扱うことはできず、したがってそのようなネットワークの挙動を捕捉しない。

【０００５】

そこで、改善された障害検出および診断機能を備えるネットワークトラブルシューティング用のフレームワークが必要である。

【課題を解決するための手段】

【０００６】

ネットワークトラブルシューティングフレームワークについて説明する。このフレームワークでは、現実のネットワークのシミュレーションを行って、現実のネットワークのオペレーションで生じる障害を検出し診断することができる。例えば、ネットワークシミュレーションは、現実のネットワークのオペレーションを記述したデータにより駆動することができる。実際には、ネットワークシミュレーションの駆動に使用するため収集される生データには、ハードウェア、ソフトウェア、および／またはネットワークのエラーなど、さまざまな理由によるエラーが含まれていることがある。ネットワークシミュレーションを駆動するために使用されるデータの一貫性を保証するために、生データから不要物を除去する。例えば、ネットワーク内の各ノードは、ネットワークシミュレーションの駆動に使用するデータを供給することができる。特定のノードによって供給されたデータは、

その特定のノードのオペレーションだけでなく、１つまたは複数の近隣ノードのオペレーションをも記述していることがある。したがって、ネットワーク内のノードから得られたデータは、冗長な場合がある。そこで、冗長なデータを次々に比較して、不一致点を識別し、その後、これをデータの平均処理、不一致データの除去などさまざまな方法で修正することができる。

【０００７】

その後、ネットワークシミュレーションにより、このデータに基づきネットワークパフォーマンスを推定することができる。推定されたネットワークパフォーマンスと現実のネットワークパフォーマンスの観察されたネットワークパフォーマンスとを比較して、現実のネットワークが期待通りの動作をしているかを検出する。期待通りの動作をしていない場合、現実のネットワークのオペレーションで障害が検出される。つまり、ネットワークシミュレーションが示す推定されたネットワークパフォーマンスと現実のネットワークが示す観察されたネットワークパフォーマンスとの違いを利用して、現実のネットワーク内の障害の発生を検出することができる。その後、ネットワークシミュレーションを利用し、ネットワークシミュレーションのネットワークパフォーマンスが現実のネットワークのネットワークパフォーマンスを近似するまで１つまたは複数の障害を選択してネットワークシミュレーションに投入することにより障害診断を行うことができる。

【０００８】

結果としてネットワークパフォーマンスの近似が得られた１つまたは複数の障害の集合が識別された後、１つまたは複数の修正点を識別し、障害の是正のため修正を実施することができる。例えば、ネットワークシミュレーションをその後で使用してｗｈａｔ－ｉｆ分析を実行し、シミュレートされたネットワークに修正を行い、修正により障害が是正されることおよび／または他の何らかの手段によるネットワークパフォーマンスの改善が行われることを検証することができる。したがって、ネットワークシミュレーションでは、障害を是正するために、および／またはネットワークパフォーマンスを改善するために行われた修正など、ネットワークに対し行うことができるさまざまな修正によるネットワークパフォーマンスへの影響に関する定量的フィードバックを行うことができる。

【発明を実施するための最良の形態】

【０００９】

類似のコンポーネントおよび図を参照するため開示および図全体を通して同じ番号を使用する。

【００１０】

概要

ネットワークトラブルシューティングフレームワークを、効率的で信頼性の高いネットワークオペレーションを維持するために有線および／または無線ネットワークで使用することについて説明する。本明細書で説明されているフレームワークは、オンライントレース駆動ネットワークシミュレーションを使用して、障害を検出し、障害の根本的原因分析（ｒｏｏｔ　ｃａｕｓｅ　ａｎａｌｙｓｉｓ）を実行する。ネットワークシミュレーションは、「現実の」ネットワークからネットワークパフォーマンスデータを取得できるという点で「オンライン」である。

【００１１】

フレームワークは、パケット破棄（ｐａｃｋｅｔ　ｄｒｏｐｐｉｎｇ）、リンク輻輳（ｌｉｎｋ　ｃｏｎｇｅｓｔｉｏｎ）、媒体アクセス制御（ＭＡＣ）（ｍｅｄｉｕｍ　ａｃｃｅｓｓ　ｃｏｎｔｒｏｌ）の不正挙動、外部ノイズなどにより生じた障害など、広範にわたるパフォーマンス問題（つまり、障害）の診断に適用することができる。さらに、このフレームワークを使用することにより、代替えネットワーク構成を評価し、ネットワークパフォーマンスを改善することもできる。以下の説明では、無線ネットワークの例におけるフレームワークについて取りあげるが、フレームワークは、有線ネットワークでも使用できる。

【００１２】

環境の実施例

すでに述べたように、ネットワーク管理に対する業界および研究機関の注目は限られていた。ネットワーク管理の実行は、ネットワークの機能の継続的監視、ネットワーク内のノードおよびリンクに関する情報の収集、報告されたデータからの不一致点およびノイズの除去、データの分析、およびネットワークの信頼性およびパフォーマンスを改善する適切な対策の実施を伴うと思われる。

【００１３】

ネットワークのトラブルシューティングは、ネットワークの「健全な状態」を維持し、オペレーションを円滑に、継続して行えるようにする役割を持つネットワーク管理の一側面である。有線、無線を問わず、ネットワークのトラブルシューティングは、異なるネットワークエンティティの絡み合い、異なる障害の発生などで生じる相互作用などさまざまな相互作用により込み入った作業になる。マルチホップ無線ネットワークのトラブルシューティングは、さまざまな要因が加わるためさらに込み入ってくる。例えば、典型的なマルチホップ無線ネットワークは、一般に、信号伝播の変動によって引き起こされるリンクエラーを生じやすく、したがってネットワークトポロジは動的で予測不可能である。さらに、マルチホップ無線ネットワークの容量は、一般的に、資源（例えば、帯域幅、電池の電力など）の希少性ゆえに限られており、ネットワークで許容できる管理トラフィックのオーバーヘッドの大きさも制約される。

【００１４】

これらの複雑な状態を取り扱うフレームワークについて説明する。このフレームワークは、オンライントレース駆動シミュレーションを使用して、障害を検出し、障害の根本的原因分析を実行することができる。シミュレーションを使用することにより、障害を引き起こしたネットワーク内で発生した事象を再現し、したがって、それらの障害を識別し、修正することができる。

【００１５】

図１は、複数のノード１０２（１）、１０２（２）、１０２（３）、．．．、１０２（ｎ）、．．．、１０２（Ｎ）を有するネットワーク１００を示す実施例における一環境の図である。図１の複数のノード１０２（１）～１０２（Ｎ）は、障害検出、診断、およびｗｈａｔ－ｉｆ分析にネットワーク１００のシミュレーションを使用するフレームワーク例を実装する。このフレームワークは、さまざまな有益な特性を備える。第１に、フレームワークは柔軟性があることである。シミュレーションはカスタマイズ性が非常に高く、さまざまな環境内に実装されたネットワークの大きなクラスに適用できるため、シミュレータの上に構築された障害診断機能は、そのような柔軟性を継承するように構成することができる。第２に、シミュレーションにより、さまざまな複雑な相互作用を捕捉できることである。例えば、ネットワーク内の相互作用、ネットワークと環境との間の相互作用、さらに、ネットワークのオペレーションで生じるさまざまな障害における相互作用を捕捉することができる。したがって、フレームワークは、シミュレーションの使用を通じて、広範にわたる障害に関する、それらの組み合わせを含む、システマティックな診断を実行できる。第３に、このフレームワークは、新しい障害を検出できる機能をフレームワークに組み込むために、システム内の他の障害とは独立したシミュレーションで障害をモデル化することができるという点で拡張性が高いということである。フレームワーク内にモデル化されている新しい障害と既存の障害との間の相互作用は、シミュレーションの実行を通じて暗黙のうちに捕捉される。第４に、シミュレータに内蔵されているネットワークの再現機能により、ｗｈａｔ－ｉｆ分析を簡単に実行することができ、ネットワークに対し実行できる修正のパフォーマンスに対する影響に関する定量的フィードバックが得られる。例えば、ネットワークのオペレーションの障害を是正する是正処置を実行すること、ネットワークのパフォーマンスを高める修正を行うことなどが可能である。

【００１６】

このフレームワークでは、ＱＵＡＬＮＥＴ（ＱＵＡＬＮＥＴはカリフォルニア州ロサンゼルスにあるＳｃａｌａｂｌｅ　Ｎｅｔｗｏｒｋ　Ｔｅｃｈｎｏｌｏｇｉｅｓ，Ｉｎｃ．

社の商標である）、ＯＰＮＥＴ　ＭＯＤＥＬＥＲ（ＯＰＮＥＴ　ＭＯＤＥＬＥＲはワシントン市にあるＯＰＮＥＴ　Ｔｅｃｈｎｏｌｏｇｉｅｓ，Ｉｎｃ．社の商標である）など、さまざまな既存のネットワークシミュレータのうち１つまたは複数を利用してネットワーク１００をシミュレートすることができる。シミュレータに供給されるトレースは、診断されるネットワーク、つまり「現実の」ネットワークから取得される。現実のネットワークからのトレースを使用することで、ハードウェア、ソフトウェア、および注目する特定のネットワークの環境の微妙な差異を捕捉できない汎用理論モデルへの依存を脱却し、それによってフレームワークの精度を改善することができる。

【００１７】

フレームワークは、さらに、障害診断スキーム（ｆａｕｌｔ　ｄｉａｇｎｏｓｉｓ　ｓｃｈｅｍｅ）を使用して根本的原因分析を実行することができる。例えば、このスキームでは、オンライントレース駆動シミュレータから吐き出された推定されたネットワークパフォーマンスデータを現実のネットワークの予想パフォーマンスのベースラインとして使用することができる。その後、予想パフォーマンスからの逸脱を使用して、潜在的障害を示す。さらに、このスキームでは、潜在的障害の集合をシミュレータに選択的に投入し、障害診断を障害の集合を探索する問題に還元することにより根本的原因分析を実行することができる。したがって、根本的原因は、投入されたシミュレーションにより現実のネットワークの観察されたパフォーマンスが近似されるような障害に基づいて識別することができる。したがって、このフレームワークでは、探索アルゴリズムを使用して、パケット破棄、リンク輻輳、外部ノイズ源、ＭＡＣ不正挙動などの障害を検出し、診断することができる。これらの障害は、パフォーマンスに対し比較的長い間持続する影響を及ぼし、停電または電池切れによりノードの電源が自動的にオフになった場合などのフェイルストップエラーを検出する場合よりも検出が困難である。

【００１８】

このような点で、このフレームワークは、シミュレーションを分析ツールとして使用し、ネットワーク内の代替えおよび潜在的パフォーマンス増強構成のトラブルシューティングおよびテストを実行することができる。以下の節では、シミュレータに供給されたときにネットワークの実際の挙動を正確に叙述するネットワークシミュレーションを実行するネットワークトレースが取りあげられている。トレースからエラーのあるデータを減らしたり、除去したりする手法も説明するが、詳細は、図８および９を参照すると理解できるであろう。したがって、シミュレータには高品質のデータが供給される。さらに、ネットワーク内の複数の障害を診断するうえで有効な探索アルゴリズムについて説明するが、詳細は、図７を参照すると理解できるであろう。シミュレータは、さらに、ｗｈａｔ－ｉｆ分析を実行し、現在のネットワークに対し実行可能な処置のパフォーマンスに関する利得を定量化するためにも使用できるが、詳細は、図１０～１３を参照すると理解できるであろう。

【００１９】

トラブルシューティングフレームワークは、さまざまなネットワーク構成で使用することができる。このような実施例の１つは、無線メッシュネットワークとして表されている、図１のネットワーク１００により例示されている。メッシュネットワークは、完全メッシュトポロジ（ｆｕｌｌ　ｍｅｓｈ　ｔｏｐｏｌｏｇｙ）または部分メッシュトポロジ（ｐａｒｔｉａｌ　ｍｅｓｈ　ｔｏｐｏｌｏｇｙ）など多様な配置を使用できる。完全メッシュトポロジでは、ネットワーク内でノード同士が直接接続されている。部分メッシュトポロジでは、それぞれのノードは少なくとも他の１つのノードに接続されているが、必ずしもネットワーク内で相互に接続されているわけではない。

【００２０】

例えば、メッシュネットワークは、ネイバー（近傍）同士が協力し合ってセルフマネージドコミュニティ（ｓｅｌｆ－ｍａｎａｇｅｄ　ｃｏｍｍｕｎｉｔｙ）型の無線メッシュネットワークを形成するイネーブリング技術として利用することができる。各ネイバーは、ネットワーク１００の複数のノード１０２（１）～１０２（Ｎ）のうちの１つまたは複

数を備えることができる。このようなネットワークでは、例えば、ネイバーは、費用効果のある方法でインターネットゲートウェイ１０４を共有することができる。

【００２１】

近傍内で使用されるようなメッシュネットワークの例では、複数のノード１０２（１）～１０２（Ｎ）を結合し通信するために使用されるルータは、自宅内に配置され、プラグが電気コンセントに差し込まれる。したがって、この実施例のルータはそれぞれ移動が限られている。しかし、障害はネットワークパフォーマンスに影響を及ぼし続けるであろうから、このようなネットワークの相対的安定性に関して、ネットワークトラブルシューティングはなおいっそう重要になる。この実施例でルータに移動性がなければ、環境の変化により無線リンクがアクセスできたりできなかったりするため、ネットワークトポロジにおけるダイナミズムが失われることに留意されたい。他の実施例では、メッシュネットワークのノードは、パーソナルデジタルアシスタント（ＰＤＡ）、タブレットパーソナルコンピュータ（ＰＣ）、ラップトップコンピュータなど、無線通信機能を備えるモバイルコンピューティングデバイスを使用するなどして、移動性を持たせることができる。

【００２２】

さらに、ユーザが機器を購入して設置しメッシュネットワークに参加するので、コミュニティメッシュネットワークの成長は有機的である。従来のメッシュネットワークは、ネットワーク管理を引き受ける一元化されたエンティティを欠いていた。しかし、本明細書で説明されているフレームワークを通じて実現される自己管理機能および自己回復機能では、それぞれのノード１０２（１）～１０２（Ｎ）がトラブルシューティング機能を実装するようにできる。例示されている実装では、管理機能を備える単一ノードが提示されている。

【００２３】

図１に例示されているネットワーク１００では、ノードはそれぞれプロセッサ、メモリ、およびネットワーク接続デバイスを備えており、その実施例は、プロセッサ１０６（ｎ）、メモリ１０８（ｎ）、およびネットワーク接続デバイス１１０（ｎ）を含むものとしてノード１０２（ｎ）により示されている。プロセッサ（例えば、プロセッサ１０６（ｎ）、１０６（Ｎ））は、形成に使用される材料、または使用される処理メカニズムの制限を受けない。例えば、プロセッサは、半導体および／またはトランジスタ（例えば、電子集積回路（ＩＣ））で構成できる。このような文脈では、プロセッサ実行可能命令は、電子機器により実行可能な命令としてよい。そうする代わりに、プロセッサのメカニズム、したがってノードの、またはノードのためのメカニズムとしては、量子コンピューティング（ｑｕａｎｔｕｍ　ｃｏｍｐｕｔｉｎｇ）、光コンピューティング、機械コンピューティング（ｍｅｃｈａｎｉｃａｌ　ｃｏｍｐｕｔｉｎｇ）（例えば、ナノテクノロジーを使用する）などが考えられる。

【００２４】

メモリ（例えば、メモリ１０８（ｎ）、１０８（Ｎ））は、読み取り専用メモリ（ＲＯＭ）、ランダムアクセスメモリ（ＲＡＭ）などの揮発性および／または不揮発性メモリの形態のコンピュータ記憶媒体を含む。メモリは、さらに、その他の取り外し可能／取り外し不可能な揮発性／不揮発性コンピュータ記憶媒体を含めることもできる。メモリは、コンピュータ可読命令、データ構造、ソフトウェアコンポーネント、およびノードのその他のデータを格納する機能を備える。

【００２５】

ネットワーク接続デバイス（例えば、ネットワーク接続デバイス１１０（ｎ）、１１０（Ｎ））は、ノードをネットワーク１００に通信できるように結合するさまざまな構成をとることができる。例えば、ローカルエリアネットワーク（ＬＡＮ）環境で使用する場合、ノード１０２（ｎ）はネットワークインターフェイスまたはアダプタを通じて、有線および／または無線のＬＡＮに通信可能に接続される。ワイドエリアネットワーク（ＷＡＮ）環境で使用する場合、ネットワーク接続デバイスは、デジタル加入者回線（ＤＳＬ）を経由する有線接続、衛星を利用した無線接続などの通信を確立するためのモデムまたはそ

の他の手段として構成することができる。論理接続は、矢印を使って図１に示されている。図１に示されているネットワーク１００は無線メッシュネットワークであるが、インターネット、イントラネットなどのさまざまなその他のネットワークを使用することもできる。

【００２６】

ノード１０２（ｎ）、１０２（Ｎ）は、ソフトウェアモジュールで構成される管理アーキテクチャの実施例を示している。一般に、本明細書で説明されている機能はどれも、ソフトウェア、ファームウェア（例えば、固定ロジック回路）、手動処理、またはそれらの実行方法の組み合わせを使用して、実施することができる。本明細書で使用されているような「モジュール」、「機能」、および「ロジック」という用語は、一般的に、ソフトウェア、ファームウェア、またはソフトウェアとファームウェアとの組み合わせを表している。ソフトウェア実装の場合、モジュール、機能、またはロジックは、１つまたは複数の中央演算処理装置（ＣＰＵ）などのプロセッサ上で実行された場合に規定されているタスクを実行するプログラムコードを表す。プログラムコードは、１つまたは複数のコンピュータ可読メモリデバイス内に格納することができる。後述のフレームワークの特徴は、プラットフォーム独立である、つまり、トラブルシューティング手法は、さまざまなプロセスを備えるさまざまな商用コンピューティングプラットフォーム上に実装することができる。

【００２７】

エージェントモジュール１１２（ｎ）は、ネットワーク１００の各ノード１０２（ｎ）上で実行されるように用意されている。エージェントモジュール１１２（ｎ）は、プロセッサ１０６（ｎ）上で実行されるものとして例示されており、メモリ１０８（ｎ）に格納可能である。エージェントモジュール１１２（ｎ）は、実行されると、さまざまなプロトコルレイヤおよび／またはネットワーク接続デバイス１１０（ｎ）からデータを収集することができるデータ収集モジュール１１４（ｎ）（これ以降「収集モジュール」と呼ぶ）を含む。その後、図１の例示されているネットワーク１００で、エージェントモジュール１１２（ｎ）は、このデータを、これ以降マネージャノードと呼ばれる管理機能を備えるノード１０２（Ｎ）に報告する。マネージャノード１０２（Ｎ）は、データの分析を実行し（例えば、データを入力として受け付けるシミュレーションの実行を通じて）、ネットワークのトラブルシューティングを行う適切な処置を実行する。ネットワークの管理は、図１のネットワーク１００に例示されているようにマネージャを単一ノード上に置くことにより一元化するか、またはネットワークの複数のノードがそれぞれ管理機能を備えるように分散させることができる。

【００２８】

エージェントモジュール１１２（ｎ）、１１２（Ｎ）は、それぞれのプロセッサ１０６（ｎ）、１０６（Ｎ）上で実行された場合、ネットワークの挙動の（ローカル）ビューを記述するデータを収集してマネージャノード１０２（Ｎ）に伝達する。送信されるデータの例として、トラフィック統計量、さまざまなリンク上の受信パケット信号強度、各リンク上の再送カウントなどがある。

【００２９】

マネージャノード１０２（Ｎ）は、メモリ１０８（Ｎ）に格納可能で、プロセッサ１０６（Ｎ）上で実行可能なマネージャモジュール１１６（Ｎ）を備え、ネットワーク１００のトラブルシューティングのためエージェント１１２（ｎ）、１１２（Ｎ）から送られたデータを処理する。例えば、マネージャモジュール１１６（Ｎ）は、プロセッサ１０６（Ｎ）上で実行可能で、メモリ１０８（Ｎ）に格納可能なネットワークシミュレータ１１８（Ｎ）（これ以降、「シミュレータ」と呼ぶ）を備え、ネットワーク１００をシミュレートする。

【００３０】

マネージャノード１０２（Ｎ）がさまざまなエージェント１１２（ｎ）、１１２（Ｎ）からデータを受信すると、ネットワーク１００のビューに不一致が生じることがある。こ

のような不一致は、トポロジーおよび環境の変化、測定誤差、不正挙動のノードなどの結果として生じる。したがって、マネージャノード１０２（Ｎ）は、このような不一致を解決するためプロセッサ１０６（Ｎ）上で実行可能なデータクリーニングモジュール１２０（Ｎ）（これ以降「クリーニングモジュール」と呼ぶ）を備える。その後、クリーニングモジュール１２０（Ｎ）から出力された浄化済みデータが供給されると、根本原因分析モジュール１２２（Ｎ）（これ以降「分析モジュール」と呼ぶ）により処理されるが、詳細は以下の図を参照すると理解できるであろう。マネージャノード１０２（Ｎ）はエージェントモジュール１１２（Ｎ）およびマネージャモジュール１１６（Ｎ）を含むように例示されているが、他の実装では、マネージャノード１０２（Ｎ）は、エージェントモジュール１１２（Ｎ）を含まないという点で専用マネージャノードである。さらに、すでに説明したように、マネージャモジュール１１６（Ｎ）の機能は、ネットワーク１００内の複数のノードにより実現することができる。

【００３１】

図２は、図１の分析モジュール１２２（Ｎ）を詳しく示している一実施例２００の図である。データ内の不一致が図１のクーリングモジュール１２０（Ｎ）により解決された後、浄化済みデータは分析モジュール１２２（Ｎ）に供給され、そこでさらに調査される。

【００３２】

分析モジュール１２２（Ｎ）は、オンライントレース駆動シミュレーションを使用して、シミュレートされたネットワーク実行により示される予想されるネットワークパフォーマンスとの不一致の根本原因を突き止める。以下の説明では、予想されるネットワークパフォーマンスおよびシミュレートされたネットワークパフォーマンスは入れ替えて使用し、ネットワークシミュレーションにより与えられるネットワークパフォーマンスを示すことができる。分析モジュール１２２（Ｎ）では、トレースユーティリティから得られた浄化済みデータ２０２を利用することができ、そのようなデータのいくつかの例が、図２に、リンク受信信号強度（ＲＳＳ）２０４、リンクロケーション２０６、およびルーティング更新２０８として例示されており、与えられたネットワーク構成およびトラフィックパターンの下でオンラインシミュレーションを駆動し、予想されるパフォーマンスを確定する。

【００３３】

分析モジュール１２２（Ｎ）は、シミュレータ１１８（Ｎ）の実行を通じて与えられるネットワークシミュレーション２１０を含むものとして例示されている。ネットワークシミュレーション２１０は、ネットワークの特性のシミュレーションを行う１つまたは複数のソフトウェアモジュールを実行することにより得られる、インターフェアレンスインジェクションモジュール（ｉｎｔｅｒｆｅｒｅｎｃｅ　ｉｎｊｅｃｔｉｏｎ　ｍｏｄｕｌｅ）２１２、トラフィックシミュレータモジュール２１４、およびトポロジ変更モジュール２１６による実施例が図２に例示されている。インターフェアレンスインジェクションモジュール２１２は、ネットワークシミュレーション２１０に対する外部ノイズの影響を投入することにより、外部ノイズ源をシミュレートするように実行可能である。トラフィックシミュレータモジュール２１４は、ネットワークシミュレーション２１０のトラフィックは現実のネットワークのトラフィックを近似するように実行可能である。トポロジ変更モジュール２１６は、ネットワークシミュレーション２１０内のノードの追加および／または削除などによりトポロジへの変更をシミュレートするように実行可能である。

【００３４】

分析モジュール１２２（Ｎ）は、ネットワークシミュレーション２１０により示されるような予想されるパフォーマンスと観察されたパフォーマンスとを比較することにより図１のネットワーク１００内の障害を検出する。不一致が観察された場合、分析モジュール１２２（Ｎ）は、シミュレートされたネットワークパフォーマンスと観察されたネットワークパフォーマンスとの間の最良一致が得られる障害ディレクトリ２１８に格納されている１つまたは複数の障害を検索することにより不一致の根本原因を突き止める。

【００３５】

分析モジュール１２２（Ｎ）は、例えば、図２で、「損失率、スループット、およびノイズ２２０」と例示されている、損失率、スループット、およびノイズ２２０を記述する図１のエージェントモジュール１１２（ｎ）のうちの１つまたは複数から観察されたデータ２２０を受信することができる。ネットワークシミュレーション２１０では、図２に「予想損失率、スループット、およびノイズ２２２」と例示されている、予想損失率、予想スループット、および予想ノイズを記述する予想データ２２２を計算する。観察されたデータ２２０は、遅延２２４を通じて、比較器２２６に伝達され、比較器２２６は観察されたデータ２２０および予想されるデータ２２２を同時に受信する。その後、比較器２２６は、観察されたデータ２２０が予想されるデータ２２２を超えたかどうかを判定する。超えている場合、比較的２２６は、ネットワーク管理者に伝達するためのエラーメッセージ２２８を出力し、エラーを障害ディレクトリ２１８に送り、エラーの根本原因を突き止める。

【００３６】

障害ディレクトリ２１８から障害のうち１つまたは複数を選択することによりエラーの根本原因が識別された後、分析モジュール１２２（Ｎ）は、その障害を是正するための１つまたは複数の代替え処置をシミュレートすることができる。これらの代替え処置は、それぞれトラフィックシミュレータ２１４およびトポロジ変更モジュール２１６により与えられるような現在のトラフィックパターンおよびネットワークトポロジの下でシミュレートすることができる。シミュレーションに基づき、分析モジュール１２２（Ｎ）は、障害を緩和しネットワークのパフォーマンス全体を高める１つまたは複数の適切な処置を提案することができるが、その実施例は、図２のリンクノード障害２３０として例示されている。例えば、ソフトウェアまたはハードウェアが故障していることが疑われる場合にネットワーク管理者に通知する、トポロジは、接続の不良が検出された場合に送信電力調整を介して変更する、ルータは、輻輳を緩和するレート制限を使用する、といったことができる。

【００３７】

オンライン診断にネットワークシミュレーション２１０を使用することには、従来の発見的または理論的な診断手法に勝るさまざまな利点がある。例えば、ネットワークシミュレーション２１０は、従来の発見的または理論的手法に比べて、ネットワークの挙動の洞察に優れる。例えば、稼働している無線ネットワークは、トラフィックフロー、ネットワーキングプロトコル、信号処理アルゴリズム、ハードウェア、無線周波（ＲＦ）伝播などの錯綜した断片（ｉｎｔｒｉｃａｔｅ　ｐｉｅｃｅ）を含む複合システムである。さらに、ネットワークのすべての断片の間には相互作用も発生しうる。障害の相互作用は、「現実の」ネットワークの実線の挙動に対応するネットワークシミュレーション２１０が行われる障害ディレクトリ２１８から１つまたは複数の障害を選択することにより効果的に診断し、対処することができる。

【００３８】

さらに、ネットワークの挙動は、次から次へのノードの間の相互作用だけでなくノードの近傍内に置かれている外部ノイズ源にも左右される。従来の発見的または理論的手法では、そのようなネットワークの挙動を捕捉せず、ネットワークのさまざまなコンポーネント間の相互作用を十分に取り扱うことはできない。

【００３９】

例えば、図３に示されている７×３のグリッドトポロジネットワーク３００を考える。５つの流れが、ネットワーク３００内に例示されており、F_1　３０２、F_2　３０４、F_3　３０６、F_4　３０８、およびF_5　３１０として示されている。示されている実施例では、フロー３０２～３１０はそれぞれ、通信するトラフィックの量は類似している。例えば、フロー３０２～３１０はそれぞれ、各アプリケーションから実質的に類似している量のデータを受け取ることができる。

【００４０】

さらに、この実施例では、隣接ノードは、互いに「聴き合う」ことができ、干渉域は共

通範囲の２倍である。例えば、ノードＡ　３１２とノードＯ　３１４との間のトラフィックは、ノードＣとＱ　３１６、３１８との間のトラフィックに干渉する。同様に、ノードＧとＵ　３２０、３２２との間のトラフィックは、ノードＥとＳ　３２４、３２６との間のトラフィックと干渉する。しかし、ＧとＵ　３２０、３２２との間のトラフィックおよびノードＡとＯ　３１２、３１４との間のトラフィックは、ノードＤとＲ　３２８、３３０との間のトラフィックとは干渉しない。

【００４１】

以下の表は、各フローが固定ビットレート（ＣＢＲ）トラフィックを１１Ｍｂｐｓの速さで送信するときにフロー３０２～３１０のスループットの一実施例を示している。

【００４２】

【表１】

F_1	F_2	F_3	F_4	F_5
2.50 Mbps	0.23 Mbps	2.09 Mbps	0.17 Mbps	2.53 Mbps

【００４３】

上記の表に示されているように、フローF_3　３０６では、F_2　３０４およびF_4　３０８よりも高いスループットを受け取る。つまり、フローF_3　３０６は、ネットワーク３００の他のフローよりも帯域幅の高域部分を消費する。

【００４４】

従来、発見的手法の適用だと、フローF_3　３０６が帯域幅の極端に大きな割り当て部分を受け取るという結論に至る可能性がある。しかし、オンライントレース駆動シミュレーションを使用すれば、マネージャノード１０２（Ｎ）は、これが通常の挙動であると結論することができる。例えば、ネットワークシミュレーションで、リンク品質を考慮することができ、したがって、フローF_1　３０２およびF_5　３１０がフローF_2　３０４およびF_4　３０８と干渉すると判定する。したがって、フローF_3　３０６は、フローF_2　３０４およびF_4　３０８とは反対に、フローF_1　３０２およびF_5　３１０からの干渉がないため、追加帯域幅を与えられる。このようにして、シミュレーションにより、すべてのフローが同じアプリケーションレベルの送信速度を持つことができるとしても、観察されたスループットが予想されると判定することができる。しかし、単純な発見的手法だと、ノードＤおよびＲ　３２８、３３０は不正挙動であると誤った結論に至る可能性がある。

【００４５】

分析モジュール１２２（Ｎ）はネットワークシミュレーションを使用して、現在のトラフィックフローおよびリンク品質が与えられた場合にネットワークに「何を期待すべきか」を知ることによりネットワークを管理する。つまり、分析モジュール１２２（Ｎ）は、ネットワークシミュレーションにより与えられる推定に基づいて何が通常の挙動なのかについてのコメントを作成することができる。前記の実施例では、F_3　３０６が、ネットワーク３００内の他のフローよりも、ネットワーク３００の帯域幅の大きな割り当て部分を使用するとしても、この挙動は予想されているため、マネージャモジュールにより障害のフラグが立てられない。観察された挙動が予想される挙動から逸脱する場合、マネージャモジュールは、図２の障害ディレクトリ２１８を利用する障害探索アルゴリズムを呼び出して、逸脱の根本原因を突き止めることができる。

【００４６】

さらに、特定の環境および構成の下で従来の署名ベースまたはルールベースの障害診断アプローチを特定のタイプのネットワークに適用することも可能であるが、単純な署名またはルールは、一般設定で障害診断の固有の複雑さを捕捉するのには不十分である。それと対照的に、シミュレータは非常にカスタマイズ性が高く、適切なパラメータ設定により、さまざまな環境で使用することを目的に構成されている大きなクラスのネットワークに適用することができる。そのようなシミュレータの上に構築された障害診断機能は、このような一般性を継承している。

【００４７】
シミュレーションベースのアプローチのさらに他の利点として、ｗｈａｔ－ｉｆ分析を実行できるという点が挙げられる。つまり、設定を修正するか、またはシミュレータで特定のアクションを実行することにより、シミュレータは想像上のシナリオに対するパフォーマンスを予測できる。このデータに基づき、マネージャモジュールは、ネットワークのパフォーマンスを最適化する適切な処置を講じるようにエージェントモジュール（例えば、図１のエージェントモジュール１１２（ｎ））に指示を与えることができる。すでに説明したように、ネットワーク内には複数の要因の相互作用があり是正処置の結果を予測することは困難であるため、このようなｗｈａｔ－ｉｆ分析はとても役立つ。例えば、送信機出力を高めるとリンク品質を改善できるが、出力を上げると、ネットワーク内の他のノードに影響を及ぼす干渉もさらに増える可能性がある。
【００４８】
障害検出および診断
現実のネットワークの機能を近似するシミュレータ内に１つの環境を作成すること（例えば、ネットワークシミュレーション２１０）を考えるシミュレーションベースの診断アプローチについて説明する。そこで、作成された環境（つまり、ネットワークシミュレーション）を使用して、現実のネットワークの予想される挙動を判定するだけでなく、現実のネットワークのオペレーションの食い違いが生じた場合にそのことを判別することができる。これらの食い違いの根本原因を特定するために、マネージャモジュールを実行して障害空間内を探索し、現実のネットワーク内で観察されるネットワークパフォーマンスを近似するネットワークパフォーマンスを再現できる障害または障害の集合を判定する。シミュレートされたネットワークは、ネットワークトポロジ、ルーティングの挙動、現実のネットワーク内で観察されるトラフィックパターンなど、さまざまなネットワークの態様を再現することができる。
【００４９】
オンライントレース駆動シミュレーションを基本構成要素として使用して、障害の根本原因を特定するために実行可能な診断アルゴリズムについて説明する。例えば、診断アルゴリズムは、まず、与えられた障害集合の下でネットワークのパフォーマンスを推定することができる。次に、診断アルゴリズムは、推定されたパフォーマンスと観察されたパフォーマンスとの間の差に基づいて、障害空間を探索し、観察された任意の食い違いを再現する。一実施形態では、診断アルゴリズム（診断ソフト）は、同じタイプ（例えば、ネットワークトポロジ）の複数の障害を診断するだけでなく、複数のタイプ（例えば、ノイズとトポロジ）の障害の存在をも診断することができる。
【００５０】
シミュレーションを駆動するために使用されるトレースデータがエラーを含んでいる場合でも障害を診断することができる。例えば、図１のエージェントモジュール１１２（ｎ）により供給されるデータは、測定誤差、偽情報、ノード１０２（ｎ）の実行に際してのソフトウェア／ハードウェアエラー、ネットワーク通信エラーなど、さまざまな理由によるエラーを含む場合がある。クリーニングモジュール１２０（Ｎ）は、マネージャノード１０２（Ｎ）により実行され、トレースから誤りのあるデータを低減または排除し、品質トレースデータを使用してシミュレーションベースの障害診断を駆動する。クリーニングモジュール１２０（Ｎ）の実行の詳細については図８～９を参照されたい。
【００５１】
トレース駆動シミュレーション
図４は、シミュレータ１１８（Ｎ）および図２のネットワークシミュレーション２１０を含むシステム４００示す実施例の図である。現実のネットワークのオペレーションを通じて得られたトレースデータを使用することにより、シミュレータ１１８（Ｎ）は現実のネットワークのネットワークオペレーションを表し、現実のネットワークに対する与えられた障害の集合が及ぼす影響を調査することを正確に実行できる。シミュレータに入力するさまざまなトレースデータを収集することができるが、それらの実施例について以下で

説明する。

【００５２】

ネットワークトポロジ４０２

ネットワークトポロジ４０２のデータは、どのノードが現在ネットワークのメンバーであり、ノード間のリンクに対応しているかなどのネットワークのトポロジを記述している、例えば、ネットワーク内のそれぞれのノードは、近隣ノードおよびノードの１つまたは複数のルーティングテーブルで参照されているノードのステータス（例えば、接続状態または切断状態）に関して報告するように構成することができる。このようにして、ネットワーク内のノードのメンバ関係を図１のマネージャノード１０２（Ｎ）に伝達することができる。一実施形態では、ネイバーまたはルートの変更しか報告されない。このデータは、ルートシミュレーションを駆動するために使用することができ、これについては、図４のルートシミュレータを参照して詳しく説明する。

【００５３】

トラフィック統計量４０４

トラフィック統計量４０４のデータは、そのデータを伝達するネットワークおよび特定のノードを通じて伝達されるデータの量を記述するために使用することができる。トラフィック統計量４０４は、図２のトラフィックシミュレータモジュール２１４により入力として使用することができ、ネットワークシミュレーション２１０のトラフィックフローにより、現実のネットワークのトラフィックフローが近似される。ネットワークのそれぞれのノードは、すぐそばのネイバーとの間でやり取りされたトラフィックの量を記述する１つまたは複数のカウンタを保持することができる。このデータは、トラフィックシミュレーションモジュール２１４により実行されるルートトラフィックシミュレーションを駆動するために使用されるが、これについては、さらに、図４を参照しつつ詳細に説明する。

【００５４】

物理的媒体４０６

物理的媒体４０６のデータは、ネットワークを実現するために使用される物理的媒体のネットワークパフォーマンスに対する影響を記述することができる。例えば、無線ネットワークでは、それぞれのノードは、そのノイズレベルおよび近接ノードからの無線リンクの信号強度を報告することができる。一実施形態では、信号強度の変動は、時間平均処理、標準偏差、またはその他の統計量集計を通じて定期的に捕捉される。

【００５５】

ネットワークオペレーション４０８

ネットワークオペレーション４０８のデータは、現実のネットワークのネットワークオペレーション４０８を記述する。すでに説明したように、観察されたネットワークオペレーションとネットワークシミュレーションから出力された推定されたネットワークオペレーションとを比較して、ネットワークオペレーションの食い違いを検出する。ネットワークオペレーションは、リンクオペレーションと終端間オペレーションの両方を含み、両方とも、パケット損失率、遅延、およびスループットなどのさまざまな尺度を通じて測定することができる。以下の説明では、リンクレベルのオペレーションを重点的に取りあげる。

【００５６】

データ収集は、（１）ローカルノードで生のパフォーマンスデータを収集するステップと（２）収集されたデータを分析のため収集ポイントに分配するステップの２つステップを伴う。ネイティブのルーティングプロトコルおよびパケットスニファなど、さまざまなツールをローカルのデータ収集に使用することができる。

【００５７】

一実施形態では、マネージャモジュール上のデータの分配によりネットワークオーバーヘッドが生じるとしても、ネットワークオーバーヘッドは低く、ネットワーク内のデータトラフィックに対する影響はほとんどない。さらに、ネットワークオーバーヘッドは、圧縮、差分符号化（ｄｅｌｔａ　ｅｎｃｏｄｉｎｇ）、マルチキャスト、時間的尺度の適応

的変化（ａｄａｐｔｉｖｅ ｃｈａｎｇｅｓ ｏｆ ａ ｔｉｍｅ ｓｃａｌｅ）、および／または分配の空間的範囲（ｓｐａｔｉａｌ ｓｃｏｐｅ ｏｆ ｄｉｓｔｒｉｂｕｔｉｏｎ）などを使用して低減することができる。例えば、ネットワークの通常オペレーションで、最小のデータの集合が収集され、交換される。追加データが必要になったら（例えば、収集される情報が食い違いを示している場合）、マネージャモジュールは、追加情報を要求し、監視増強を必要とするノードの部分集合に対するデータ収集の頻度を上げることができる。

【００５８】

シミュレーションの方法

シミュレータによりモデル化されるネットワーク特性は、トラフィック負荷、ルーティング、無線信号、障害など、さまざまなカテゴリに分類することができる。以下の節では、これらのカテゴリ例のそれぞれのシミュレーション実施例を対応するネットワーク特性をシミュレータにシミュレートさせるために使用される個別モジュールとして説明する。

【００５９】

トラフィック負荷シミュレータ４０１

シミュレータにより生成されるネットワークシミュレーションは、現実のネットワークのトラフィックパターンを近似するトラフィックパターンが得られるように構成することができる。グラフィックかシミュレーションの方法の一実施例では、終端間アプリケーションの要求をシミュレートする必要がある。しかし、Ｎノードネットワークには、潜在的にN^2の要求が含まれる可能性がある。さらに、アプリケーションの要求に異質性があり、伝送制御プロトコル（ＴＣＰ）、ユーザデータグラムプロトコル（ＵＤＰ）、ラピッドトランスポートプロトコル（ＲＴＰ）（ｒａｐｉｄ ｔｒａｎｓｐｏｒｔ ｐｒｏｔｏｃｏｌ）などの異なるトランスポートプロトコルを使用するとすれば、終端間アプリケーションの要求は、取得が困難な場合がある。

【００６０】

一実施形態では、トラフィック負荷シミュレータ４１０モジュールは、図２のトラフィックシミュレータモジュール２１４の一部であり、スケーラビリティに使用され、さらに終端間アプリケーション要求を取得する必要性をなくすために使用されるリンクベースのトラフィックシミュレーションを実行する。リンクベースのトラフィックシミュレーションは、実行されると、各リンクでアプリケーションレベルの送信速度を調整し、現実のネットワークのリンクレベルのトラフィックカウントと一致するようにできる。このようにして、レイヤが上がるほど（例えば、トランスポートレイヤ、アプリケーションレイヤなど）、抽象化され、これにより、シミュレーションはパケットサイズおよびトラフィック速度に集中することができる。

【００６１】

アプリケーションレベルの送信速度のみが調整することができる、媒体アクセス制御（ＭＡＣ）プロトコルを取り扱わなければならないなど、リンク上の送信速度は直接制御できないためシミュレータでリンク毎に送信速度を一致させることは、自明である。例えば、リンクのアプリケーション送信速度が１Ｍｂｐｓに設定されている場合、実際の送信速度（電波で）ＭＡＣレイヤでのバックオフのせいで低いか、またはＭＡＣレベル再送のせいで高くなる。この問題は、さらに、異なるリンクでの送信速度間の相互依存性を持ち込む干渉によりさらに複雑になる。

【００６２】

反復探索手法を使用すると、各リンクでの送信速度を判定することによりこれらの問題に対処することができる。（ｉ）乗法的増加および乗法的減少、（ｉｉ）加法的増加および加法的減少など、さまざまな反復探索手法を使用することができる。擬似コード例を使用して表されている以下のプロシージャに示されているように、各リンクでは、個別に、シミュレータの現在の送信速度と現実のネットワークの実際の送信速度との差を小さくしようとする。

【００６３】

```
   while (not converged and i<maxIterations)
        i=i+1
        If (option==multiplicative)
              for each link(j)
                    prevRatio=targetMacSent(j)/simMacSent(J);
                    currRatio=(1-a)+a*prevRatio;
                    simAppSent(J)=prevAppSent(j)*currRatio;
        else // 加法
              for each link(j)
                    diff=targetMacSent(j)-PrevMacSent(j);
                    simAppSent(j)=prevAppSent(j)+a*diff;
        simAppSentを入力として使用しシミュレーションを実行
        シミュレーション結果からすべてのリンクについてsimMacSentを判定する
        conveyed=isConverge(simMacSent, targetMacSent)
```

したがって、上記の疑似コードは、乗法的増加／乗法的減少または加法的増加／加法的減少のいずれかを使用してアプリケーションレベルの送信速度の例を示している。上記のプロシージャ例では、パラメータaを導入しており、a≦１（例えば、a＝０．５）として、発振を減衰する。このプロセスは、その速度が目標速度（targetMacSentと表されている）を近似するか、または最大反復回数に到達するまで繰り返される。

【００６４】

ルートシミュレータ４１２

ルーティングは、ネットワークパフォーマンス、特にマルチホップ無線ネットワークでは重要な役割を果たす。ルートシミュレーションの方法の１つは、シミュレータ内で、現実のネットワークで使用されるルーティングプロトコルをシミュレートする方法を伴う。現実のネットワークと同じルーティング挙動を再現するために、パケットの詳細トレースを取得してルーティングをセットアップする。

【００６５】

パケットが辿る実際のルートは、ルートシミュレータ４１２モジュールへの入力として使用することができる。ルートが頻繁に変動しないのであれば、マネージャ側でパケット毎にルートを収集する代わりに、ルーティングの変更を追跡することができる。この目的のために、ルートシミュレータ４１２モジュールはトレース駆動であってよい。例えば、ルートシミュレーションモジュールは、ＱＵＡＬＮＥＴシミュレータ（ＱＵＡＬＮＥＴはカリフォルニア州ロサンゼルスＳｃａｌａｂｌｅ Ｎｅｔｗｏｒｋ Ｔｅｃｈｎｏｌｏｇｉｅｓ，Ｉｎｃ．社の商標である）などのシミュレータ１１８（Ｎ）の内部に実装することができる。ルートシミュレーション４１２モジュールは、ルーティングの更新および対応するタイムスタンプを入力として受け付け、ネットワークシミュレーションでのパケットが実際のネットワークと同じルートをたどるようにする。

【００６６】

信号強度シミュレータ４１４

信号強度は、有線および無線の両方のネットワークパフォーマンスに営業を及ぼす。異なるネットワーク接続デバイス（例えば、無線カード）および環境にまたがるバラツキにより、これらの要因すべてを捕捉する一般的な伝播モデルを導出することは困難であると

思われる。このような問題点に対処するために、信号強度シミュレータ４１４を、ネットワーク接続デバイス自体から得られるような、現実のネットワーク内の信号強度の実際の測定結果から導出することができる。

【００６７】

フォルトインジェクション（Ｆａｕｌｔ　Ｉｎｊｅｃｔｉｏｎ）４１６

フレームワークは、ホストでのパケット破棄、外部ノイズ源、ＭＡＣ不正挙動などさまざまなタイプの障害をシミュレータに投入するように実行可能なフォルトインジェクション４１６モジュールを備えることができる。このようにして、分析モジュールは、ネットワークに対する障害の影響を調べることができる。例えば、ホストでのパケット破棄は、ハードウェア／ソフトウェアのエラー、バッファオーバーフロー、悪意ある破棄などにより、不正挙動のノードが１つまたは複数の近隣ノードからトラフィックの一部を破棄した場合に発生する。このようなエンドホストパケット破棄（ｅｎｄ－ｈｏｓｔ　ｐａｃｋｅｔ　ｄｒｏｐｐｉｎｇ）を検出できる機能は、マネージャがこれを使用してエンドホストにより引き起こされた損失をネットワークにより引き起こされた損失とを区別できるため、有用である。

【００６８】

このフレームワークは、フォルトインジェクション４１６モジュールの実行により、さらに、ネットワーク内に外部ノイズ源を投入する機能もサポートする。したがって、このフレームワークでは、ネットワークの外部にある（つまり、ノードが提供しない）がネットワークに影響を及ぼさないノイズ源の効果を複製するシミュレーションを行うことができる。

【００６９】

ＭＡＣ不正挙動は、障害が発生しているノードがＭＡＣエチケットに従わず、チャネル帯域幅の不正な割り当てを取得すると発生する。例えば、ＩＥＥＥ　８０２．１１では、障害のあるノードは、積極的にトラフィックを送信するためにより小さなコンテンションウィンドウ（ｃｏｎｔｅｎｔｉｏｎ　ｗｉｎｄｏｗ）（ＣＷ）を選択することができる。

【００７０】

リング輻輳も、フレームワークにより、シミュレートされたネットワーク上で高いデータ送信負荷をかけることで、シミュレートすることができる。他のタイプの障害とは異なり、リンク輻輳は、各ノードから収集されたトラフィック統計量により暗黙のうちに捕捉される。したがって、トレース駆動シミュレーションでは、現実のネットワークに対するリンク輻輳の影響を直接評価することができる。障害診断の詳細については、以下の節を参照されたい。

【００７１】

障害診断

不具合およびパフォーマンスの問題の根本原因は、図２の分析モジュール１２２（Ｎ）を実行することにより診断できる。ネットワークシミュレーションに障害を適用することにより、ネットワークの不一致の診断を、シミュレートされたネットワーク内に投入した場合に現実のネットワークの観察されたパフォーマンスを近似するシミュレートされたネットワークによる推定パフォーマンスが得られる障害の集合を探索することに還元できる。正式にいうと、ネットワーク設定ＮＳが与えられたとすると、ＦａｕｌｔＳｅｔは以下のようになる。

【００７２】

【数１】

$$\mathrm{SimPerf}\ (\mathrm{NS};\mathrm{FaultSet}) \approx \mathrm{RealPerf}$$

【００７３】

ただし、ネットワークパフォーマンスは、さまざまな異なる尺度を使用して定量化することができる機能値である。

【００７４】

障害の探索空間は、発生しうる障害のさまざまな組み合わせによる複数の探索次元を含むことができる。一実施形態では、分析モジュール１２２（Ｎ）は、さまざまなタイプの障害が少数の特定のネットワークパフォーマンス尺度を変えることが多いという認識から、効率のよい探索を行えるように最適化されている。例えば、ホストでのパケット破棄は、一般に、リンク損失率に影響を及ぼすが、他のネットワークパフォーマンス尺度には影響を及ぼさない。したがって、ネットワークパフォーマンス尺度は、尺度によって示される観察されたネットワークパフォーマンスと推定されたネットワークパフォーマンスとの間の差に注目することによりネットワークパフォーマンスの診断に使用することができる。

【００７５】

一実施形態では、障害診断の目的のために予測モデルを用意する必要はない。むしろ、事実に倣ってネットワーク内に生じた事をシミュレートするので十分である。例えば、エージェントモジュールは、リンク状態およびトラフィックパターンに関する情報を定期的にマネージャモジュールに報告することができる。この情報は、処理された後、シミュレータに供給され、そこでネットワークシミュレーションが作成されるので、それを使用して、障害の考えられる根本原因を突き止めることができる。

【００７６】

初期診断

図５は、同じタイプの障害が最初に次々に診断される一実施例のプロシージャ５００を示す流れ図である。説明を簡単にするため、以下のでは、（１）ホストでのパケット破棄、（２）外部ノイズ、および（３）ＭＡＣ不正挙動の３種類の障害例を取りあげる。しかし、さまざまな他の複数の障害および障害の組み合わせも同様の方法で対処できることは明白であろう。以下の説明は、説明されているシステムおよびデバイスを使用して実行することができるプロシージャを含む。これらのプロシージャのそれぞれの態様は、ハードウェア、ファームウェア、またはソフトウェア、またはそれらの組み合わせで実行することができる。これらのプロシージャは、１つまたは複数のデバイスにより実行されるオペレーションを指定する一組のブロックとして示されているが、必ずしも、それぞれのブロックによりオペレーションを実行するのに示されている順序に限定されるわけではない。

【００７７】

すでに説明したように、トレース駆動シミュレーションは、現実のネットワークの現在ネットワーク設定を供給されたときに、ネットワークの推定されたネットワークパフォーマンスを確定するために使用することができる。推定されたネットワークパフォーマンスと観察されたネットワークパフォーマンスとの差に基づいて、決定木を使用して障害のタイプを判定することができるが、その例は、図６に示されている。

【００７８】

さまざまな要因があるため、推定されたネットワークパフォーマンスは、障害が存在していなくても、観察されたネットワークパフォーマンスと一致することはありえない。したがって、ネットワークパフォーマンスの食い違いは、しきい値を使用して判定することができる。例えば、推定されたネットワークパフォーマンス値と観察された（つまり、現実の）ネットワークパフォーマンス値との差が対応するしきい値を超えているかどうかに基づいて判定することができる。このしきい値は、シミュレートされたネットワークパフォーマンスと実際のネットワークパフォーマンスの間の時間推移による差を観察することなど、さまざまな方法で計算することができる。

【００７９】

障害分類スキーム（ｆａｕｌｔ　ｃｌａｓｓｉｆｉｃａｔｉｏｎ　ｓｃｈｅｍｅ）は、図６に示されている一例であるが、異なる障害がそれぞれの異なる挙動を示すことに注目することにより食い違いを引き起こした障害のタイプを判定するように構成される。それでも障害のそれぞれによって示される挙動はオーバーラップする場合があるが（例えば、ノイズ源とホストでのパケット破棄の両方により損失率が高まり、コンテンションウィンドウを下げることでトラフィックの量が増え、したがって干渉ノイズが増大する、など）

、最初に、違いのあるそれぞれの挙動をチェックすることにより障害を分類することができる。例えば、外部ノイズ源は、近隣ノードで生じるノイズのレベルを高めるが、ノードの送信速度を高めることはない。したがって、外部ノイズ源は、ＭＡＣ不正挙動およびホストでのパケット破棄と区別することができる。

【００８０】

再び図５を参照することにする。以下の説明では、関係する図の説明に含まれる疑似コード例で使用されている他の表記を説明するのにイタリック体のテキストを括弧で囲んでいる。ブロック５０２で、分析モジュールは、図２の障害ディレクトリ２１８などの複数の障害うちから１つまたは複数の障害を選択する。プロシージャ５００の第１の反復では、通常の動作条件の下で、つまり障害なしで、ネットワークの予想されるパフォーマンスを求めるのに複数の障害のうちどれも選択されない。他の実施形態では、図５のプロシージャ５００は、初期診断を実行するために使用されるが、反復的ではない、つまり、「１パス」プロシージャである。そのような実施形態では、ブロック５０２はプロシージャ５００から削除することができ、障害集合は空集合 {} として与えられる。

【００８１】

ブロック５０４で、障害集合（ＦＳ）およびネットワーク設定（ＮＳ）は、入力としてネットワークシミュレーションに供給される。信号強度、トラフィック統計量、ルーティングテーブルなどさまざまなネットワーク設定を行うことができる。

【００８２】

ブロック５０６で、予想されるパフォーマンス（ＳｉｍＰｅｒｆ）は、与えられた入力とともにネットワークシミュレーションを実行することにより予測される。決定ブロック５０６で、予想されるパフォーマンス（ＳｉｍＰｅｒｆ）と現実のパフォーマンス（ＲｅａｌＰｅｒｆ）との差（Ｄｉｆｆ）はしきい値よりも大きいのかどうかに関して判別が行われる。この差がしきい値よりも大きい場合（ブロック５０６）、障害タイプ（ＦＴ）が判定される（ブロック５１０）。障害タイプの判定の詳細な説明は、図６を参照されたい。

【００８３】

障害タイプが判定された後、観察されたネットワークパフォーマンスと予想されるネットワークパフォーマンスとの差がその特定の障害タイプのしきい値を超えている（ブロック５１４）ノードおよびリンクの集合を見つけることにより障害が特定される（ブロック５１２）。障害タイプにより、パフォーマンスの差を定量化するのに使用されるネットワークパフォーマンスの尺度が決定される。例えば、パケット破棄は、予想される損失率と観察された損失率との著しい差を有するリンクを見つけることにより識別することができる。

【００８４】

ブロック５１６で、障害の大きさが判定される。例えば、ある関数（「ｇ（）」で表されている）を、障害の影響を対応する大きさにマッピングするために使用することができる。例えば、エンドホストのパケット破棄のシナリオでは、リンクの損失率の差はリンク上のパケット破棄率の変化（障害の大きさ）に直接マッピングできるためｇ（）関数は恒等関数である。外部ノイズの障害のシナリオでは、ｇ（）関数はノイズ信号の伝播関数である。ブロック５１０～５１６は、リンクまたはノード毎に繰り返すことができる。その後、５１６で、対応する大きさを持つ障害を障害集合に追加することができる。

【００８５】

以下では、以下のような図５のプロシージャ５００に類似しているプロシージャの実装のため実行できる疑似コード例を示す。

【００８６】

ＮＳはネットワーク設定（つまり、信号強度、トラフィック統計量、ルーティングテーブル）を表す。

【００８７】

ＲｅａｌＰｅｒｆは現実のネットワークのパフォーマンスを表す。

【００８８】
ＦａｕｌｔＳｅｔ＝{}
ＳｉｍＰｅｒｆを予測するのに、入力（ＮＳ；ＦａｕｌｔＳｅｔ）でシミュレーションを実行する
ｉｆ ｜Ｄｉｆｆ（ＳｉｍＰｅｒｆ，ＲｅａｌＰｅｒｆ）｜＞しきい値
各リンクまたはノードｉに対する決定木を使用して障害タイプｆｔを判別する
ｉｆ （｜$Diff_{ft}$（ＳｉｍＰｅｒｆ（ｉ），ＲｅａｌＰｅｒｆ（ｉ））｜＞しきい値）
ｆａｕｌｔ（ｆｔ，ｉ）を追加する、ただし
ｍａｇｎｉｔｕｄｅ（ｉ）＝ｇ（$Diff_{ft}$（ＳｉｍＰｅｒｆ（ｉ），ＲｅａｌＰｅｒｆ（ｉ））
この疑似コードは、障害が発生したかどうかを検出するために使用することができる診断アルゴリズムを説明している。以下のプロシージャは、検出された障害のタイプを判別するために使用することができるアルゴリズムの一実施例である。

【００８９】
図６は、決定木を使用して障害のタイプを判定する一実施例のプロシージャ６００を説明する流れ図である。図６に示されているプロシージャ６００は、図５のブロック５１０に対応する場合もあれば対応しない場合もある。決定ブロック６０２で、送信されたパケットのシミュレートされた量（ＳｉｍＳｅｎｔ）から送信されたパケットの実際の量（ＲｅａｌＳｅｎｔ）を引いた値の絶対値が、ＴｈｒｅｓｈＳｅｎｔＤｉｆｆと表されるしきい値よりも大きいかどうかの判定が行われる。大きい場合、障害が送られコンテンションウィンドウ（ＣＷ）の設定が低すぎることを示す（ブロック６０４）。

【００９０】
ブロック６０２のしきい値以下の場合、決定ブロック６０６で、現実のネットワーク上で示される現実のノイズ（ＲｅａｌＮｏｉｓｅ）とシミュレートされたネットワークの予想されるノイズ（ＳｉｍＮｏｉｓｅ）との間に食い違いがあるかどうか（つまり、ノイズしきい値差分ＴｈｒｅｓｈＮｏｉｓｅＤｉｆｆを超えたかどうか）について判定が行われる。食い違いがあれば、ノイズ障害と判定される（ブロック６０８）。

【００９１】
ノイズしきい値以下であった場合（ブロック６０６）、決定ブロック６１０で、シミュレートされたパケット損失（ＳｉｍＬｏｓｓ）、つまり、予想されるパケット損失が現実のパケット損失（ＲｅａｌＬｏｓｓ）と複数のしきい値損失の差の分（ＴｈｒｅｓｈＬｏｓｓＤｉｆｆ）だけ異なるかどうかの判定が行われる。そのように異なる場合、パケット破棄障害が発生している（ブロック６１２）。異ならない場合、ノードは正常に動作している（ブロック６１４）。さまざまな他の障害タイプも同様にして判定できることは明白であろう。

【００９２】
図７は、反復診断アルゴリズムを使用して同じタイプの障害が次々に診断される一実施例のプロシージャ７００を示す流れ図である。一般に、ネットワークでは、障害の相互作用は複数のタイプが発生しうる。障害が同じタイプであるとしても、相互作用はそれでも起こりえ、したがって、１パスの診断アルゴリズムでは不十分な場合がある。したがって、図７に示されているように、反復診断アルゴリズム（反復診断プログラム）を実装して根本原因を見つけるようにすることができる。このアルゴリズムは、（ｉ）図５のプロシージャ５００に似た初期診断段階と（ｉｉ）反復的精密化の２つの段階を含む。

【００９３】
初期診断段階では、１パス診断アルゴリズムが適用され、初期障害集合が導かれる。第２段階では、（ｉ）すでに障害集合に挿入されている障害の大きさを調整する、および（ｉｉ）必要ならば新しい障害を障害集合に追加することにより障害集合を反復的に精密化する。プロシージャ７００は、障害集合の変化が無視できるくらい小さくなるまで、例え

ば、障害タイプおよびロケーションが変化しなくなる、障害の大きさの変化が最小であるなどとなるまで、繰り返すことができる。

【００９４】

反復的方法は、障害の大きさについて探索する場合にも使用することができる。この方法は、高いレベルで、図５に関して説明されている、リンクベースのシミュレーションに類似しており、そこでは目標値と現在値との間の差が目標に徐々に移行する際のフィードバックとして使用されている。

【００９５】

ブロック７０２で、例えば、反復毎に既存の障害集合の下で予想されるネットワークパフォーマンスが推定される。例えば、予想されるネットワークパフォーマンスは、現実のネットワークから得られたネットワーク設定を使用しネットワークのシミュレーションを通じて推定することができる。ネットワーク設定を行うには、それぞれのノード上でエージェントモジュールを実行する。各ノードで行われるネットワーク設定では、ノードのローカルのネットワークパフォーマンスだけでなく、近隣ノードのネットワークパフォーマンスをも記述することができる。

【００９６】

ブロック７０４で、推定されたネットワークパフォーマンス（既存の障害集合による）と現実のパフォーマンスとの差が計算される。例えば、この差は、マネージャモジュールの実行を通じてマネージャノードにより計算することができる。マネージャモジュールは、実行されると、複数のエージェントから得られた追加ネットワーク設定により示されているようにシミュレートされたネットワークから得られた推定された（つまり、予想される）ネットワークパフォーマンスと現実の（つまり、観察された）ネットワークパフォーマンスとを比較する。

【００９７】

図７のプロシージャ７００は、まず、図５に関して説明されているプロシージャ５００に似た方法で初期障害診断を行う。決定ブロック７０６で、例えば、計算で求めた差が対応するしきい値よりも大きいかどうかについて判定が行われる。そうでない場合、障害集合が報告される（ブロック７０８）。この場合、計算で求めた差はしきい値以下なので、これは、ネットワークが正常に稼働していることを分析モジュールに示すことになる。しかし、計算で求めた差が対応するしきい値よりも大きい場合（ブロック７０６）、障害タイプが判定される（ブロック７１０）。障害タイプは、さまざまな方法で判定できるが、その一例は、図６に関して説明されている。

【００９８】

ブロック７１２で、この差は、障害の大きさの変化に翻訳され、障害の大きさは計算された変化に応じて調整される（ブロック７１４）。例えば、関数ｇ（）は、図５に関してすでに説明されているように、予想されるネットワークパフォーマンスと現実のネットワークパフォーマンスとのそれぞれの差に基づいて障害のそれぞれについて障害の大きさを計算するために使用することができる。このようにして、障害を次々に比較し、観察された食い違いに対応するネットワークパフォーマンスに対する影響を持つ障害を判定することができる。一実施形態では、最大の障害の大きさを最初に使用して、食い違いを説明し、それによって、食い違いを引き起こした特定の障害を識別する。他の実施形態では、障害の大きさを比較して、計算された差を近似する差が得られる障害を特定する。例えば、複数の障害はそれぞれ、予想されるネットワークパフォーマンスと現実のネットワークパフォーマンスとのそれぞれの差を持ちうる。それぞれの差をネットワークパフォーマンスの計算で求めた差と照合して障害のうち１つまたは複数を選択することができる。ブロック７１６で、対応するしきい値よりも低い大きさを持つ障害を除去し、それによって障害集合を最適化する。

【００９９】

決定ブロック７１８で、現在の障害集合を使用して予想されるネットワークのパフォーマンスが現実のネットワークパフォーマンスに収斂するかどうかの判定が行われる。例え

ば、分析モジュールは、ネットワークシミュレーションにおける障害集合およびその結果のパフォーマンス値の１つまたは複数の前の反復を記述する発見的データを格納することができる。目標値（つまり、現実のネットワークパフォーマンス値）と現在の値（つまり、シミュレートされたネットワークパフォーマンス値）との差は、分析モジュールによりフィードバックとして使用され、これにより、ネットワークシミュレーションを徐々に「移動し」、現実のネットワークを近似する。

【０１００】

予想されるパフォーマンスが現実のネットワークパフォーマンスに収斂しない場合（ブロック７１８）、新しい障害候補が障害集合に追加される。障害の正しい大きさを探索することのほかに、例えば、予想されるネットワークパフォーマンスと現実のネットワークパフォーマンスとの差を最もよく説明できる新しい障害候補を選択することにより、障害集合のメンバ関係を反復的に精密化することができる（ブロック７２０）。これらの新しい障害は障害集合に追加される（ブロック７２２）。その後、新しい障害候補を含む障害集合をネットワークシミュレーションの入力として使用し、既存を障害集合の下で予想されるネットワークパフォーマンスを推定する（ブロック７０２）。一実施形態では、最大の食い違いを説明できるプロシージャ７００の反復毎に障害を追加し、それによって偽陽性を制御する。その後、シミュレートされたネットワークの予想されるパフォーマンスが現実のネットワークの実際のパフォーマンスを近似するまでプロシージャ７００を繰り返すことができる。このようにして、シミュレートされたネットワークを、障害を取り込むことにより移動し、現実のネットワーク内の観察されたネットワークパフォーマンスの原因となる障害の正確な記述を得ることができる。

【０１０１】

以下では、図７のプロシージャ７００を実現するために実行することができる疑似コード例を示す。

【０１０２】

１）ＮＳでネットワーク設定を表す
（つまり、信号強度、トラフィック統計量、およびルーティングテーブル）
ＲｅａｌＰｅｒｆで現実のネットワークのパフォーマンスを表す
２）ＦａｕｌｔＳｅｔ＝{}
３）ＳｉｍＰｅｒｆを予測するのに、入力（ＮＳ；ＦａｕｌｔＳｅｔ）でシミュレーションを実行する
４）ｉｆ　｜Ｄｉｆｆ（ＳｉｍＰｅｒｆ，ＲｅａｌＰｅｒｆ）｜＞しきい値
　　ｇｏ　ｔｏ（５）
　ｅｌｓｅ
　　ｇｏ　ｔｏ（７）
５）初期診断：図５のアルゴリズムを適用することによりＦａｕｌｔＳｅｔを初期化する
６）ｗｈｉｌｅ（収束しない）
　ａ）障害の大きさを調整する
　　ＦａｕｌｔＳｅｔ内の障害タイプｆｔ毎に（図６の決定木の順序で）
　　（ＦａｕｌｔＳｅｔ，ｆｔ）内の障害ｉ毎に
　　ｍａｇｎｉｔｕｄｅ（ｉ）＝ｇ（$Diff_{ft}$（ＳｉｍＰｅｒｆ（ｉ），ＲｅａｌＰｅｒｆ（ｉ））
　　ｉｆ（｜ｍａｇｎｉｔｕｄｅ（ｉ）｜＜しきい値）
　　ｆａｕｌｔ（ｆｔ，ｉ）を削除する
　ｂ）必要ならば、新しい候補障害を追加する
　　障害タイプｆｔ毎に（図６の決定木の順序で）
　　ｉ）　ＦａｕｌｔＳｅｔ内になく、最大の｜$Diff_{ft}$（ＳｉｍＰｅｒｆ（ｉ）；ＲｅａｌＰｅｒｆ（ｉ））｜を持つような障害を見つける
　　ｉｉ）ｉｆ　（｜$Diff_{ft}$（ＳｉｍＰｅｒｆ（ｉ），ＲｅａｌＰｅ

ｒｆ（ｉ））｜＞しきい値）
（ｆｔ，ｉ）をｍａｇｎｉｔｕｄｅ（ｉ）＝ｇ（Ｄｉｆｆ$_{ft}$（ＳｉｍＰｅｒｆ（ｉ），ＲｅａｌＰｅｒｆ（ｉ））となるＦａｕｌｔＳｅｔに追加する
ｃ）シミュレーションを実行する
７）ＦａｕｌｔＳｅｔを報告する
したがって、上記の擬似コードは、複数のタイプの障害を診断するように構成されている診断アルゴリズムの実施例を記述している。

【０１０３】
トレースデータ内の誤りの除去
前の節では、トレースデータを使用してオンラインシミュレーションを駆動する障害診断について説明した。実際には、それぞれのノードによって実行されときにエージェントモジュールによって収集される生トレースデータは、ハードウェア、ソフトウェア、および／またはネットワークのエラーなど、上述のさまざまな理由によるエラーを含むことがある。したがって、図１のクリーニングモジュール１２０（Ｎ）を実行し、複数のエージェントから受け取った「生」トレースデータを浄化し、障害診断のため浄化済みトレースデータをシミュレータ１１８（Ｎ）に入力として供給することができる。

【０１０４】
図８は、図１の複数のノード１０２（１）～１０２（Ｎ）がネイバー監視を行うために実行可能なエージェントモジュールを含む一実施例内のネットワーク８００の図である。ネットワーク内のノードのそれぞれで実行されるエージェントモジュールは、ネイバー監視を実行するが、これは、複数のノード１０２（１）～１０２（Ｎ）のそれぞれが、入る／出る自リンクだけでなく、通信範囲内にある他のリンクのパフォーマンスおよびトラフィック統計量を報告する手法である。ネイバー監視は、いろいろな方法で実行できる。例えば、第１のノード上のエージェントモジュールを実行し、ネットワーク内の第２のノードを調べ、第２のノードからネットワークパフォーマンスデータを取得することができる。他の場合には、第１のノードは、第２のノードから、ネットワークパフォーマンスデータを含むブロードキャストなどの通信を受け取る。他の場合には、第１のノードは、第２のノードにより送信されたデータを監視し、ネットワークを通じて通信し、ネットワークパフォーマンスを監視する。例えば、第１のノードは、ノードのネットワーク接続デバイスがその特定のノードに到着する各データパケットを丸ごと解釈し読み込むための「無差別」モードで動作することができる。

【０１０５】
ネイバー監視により、異なる発信源（つまり、ノード）からの複数のレポートはリンク毎に送信される可能性が高い。例えば、ノード１０２（３）は、ノード１０２（２）のネットワークパフォーマンスだけでなく、ノード１０２（１）、１０２（ｎ）をも記述するノード１０２（２）からレポート８０２（２）を取得することができる。図８でレポートの参照番号で使用される括弧は、それぞれのノード、例えば、ノード１０２（２）を持つレポートとレポート８０２（２）の対応関係を示すために選択されている。

【０１０６】
ノード１０２（３）は、マネージャノード１０２（Ｎ）と通信するために形成されたレポート８０２（３）内のレポート８０２（２）（図８では破線で示されている）からのネットワークパフォーマンスデータを含む。レポート８０２（３）は、さらに、ノード１０２（２）、１０２（１）を監視することによりノード１０２（３）により得られるネットワークパフォーマンスデータを含むことができる。一実施形態では、レポート８０２（３）は、エージェントモジュールを実行し、冗長な情報を除去することで最適化されている。例えば、ノード１０２（３）のエージェントモジュールは、整合性がとれており、それぞれのレポート８０２（２）、８０２（３）内のノード１０２（２）、１０２（３）により繰り返される情報を除去し、データ内の不一致を記述したデータを残すことができる。同様に、ノード１０２（ｎ）は、収集モジュール１１４（ｎ）を実行し、ノード１０２（２）、１０２（３）からネットワークパフォーマンスデータを取得することができる。ネ

ットワークパフォーマンスデータは、マネージャノード１０２（Ｎ）への通信のためのレポート８０２（ｎ）として構成される。

【０１０７】

マネージャノード１０２（Ｎ）では、冗長なレポートを使用して、ネットワークパフォーマンスの１つまたは複数の不一致を検出することができる。例えば、マネージャノード１０２（Ｎ）では、クリーニングモジュール１２０（Ｎ）を実行して、レポート８０２（２）、８０２（３）同士を比較し、そこに記述されているネットワークパフォーマンスデータ内の不一致を見つけることができる。不一致は、さまざまな方法により見つけることができ、その一実施例は以下の図で説明されている。

【０１０８】

図９は、ネットワーク内の不正挙動ノードを特定するために近隣ノードを記述しているレポートを比較する一実施例のプロシージャ９００を示す流れ図である。この実施形態では、プロシージャ９００は、レポート内で食い違いを説明できるノードの最小集合として不正挙動ノードを識別する。

【０１０９】

図９に関して説明されているプロシージャ９００では、送信ノードｉは送信されたパケットの個数および有向リンクｌについて受信されたＭＡＣレベル肯定応答の個数を（$sent_i(l)$，$ack_i(l)$）として報告する。受信ノードｊは、リンク上で受信されたパケットの個数を$recv_j(l)$として報告する。さらに、送信または受信ノードのすぐそばのネイバーｋも、リンク上で送信または受信されたパケットの個数およびＭＡＣレベル肯定応答の個数を（$sent_k(l)$，$recv_k(l)$，$ack_k(l)$）として報告する。レポート内の不一致は、以下の場合のうちの１つとして定義される。

【０１１０】

決定ブロック９０２で、送信先により報告される、リンク上で受信されたパケットの個数が、送信元によって報告される、同じリンク上で送信されたパケットの個数よりも著しく多い（しきい値により記述されているように）かどうかの判定が行われる。つまり、ノードｉからノードｊへのリンクｌについて、しきい値ｔが与えられたとすると、以下の判定が行われる。

【０１１１】

$$recv_j(l) - sent_i(l) > t$$

しきい値ｔが使用されているのは、それぞれのノードによるレポートの通信は、通常、同期していないからである。受信されたパケットの個数が送信されたパケットの個数よりも著しく多い場合、レポート内の不一致が指摘されるが、これについては、ブロック９１２に関して詳述する。それぞれのノードにより受信され送信されるパケットの個数が対応している場合、プロシージャ９００はブロック９０４に進む。

【０１１２】

決定ブロック９０４で、送信元によって報告される、リンク上で送信されたＭＡＣレベル肯定応答の個数は、送信先によって報告される、そのリンク上で受信されたパケットの個数に対応するかどうかの判定が行われる。つまり、ノードｉからノードｊへのリンクｌについて、しきい値ｔが与えられたとすると、以下が判定される。

【０１１３】

$$|ack_i(l) - recv_j(l)| > t$$

したがって、肯定応答の数が受信されたパケットの個数に対応しない場合（つまり、近似）（ブロック９０４）、レポート内の不一致が指摘される。受信された肯定応答およびパケットの数が呼応する場合（ブロック９０４）、プロシージャ９００は、ブロック９０６に進行する。

【０１１４】

決定ブロック９０６で、送信先のネイバーにより報告される、リンク上で受信されたパケットの個数が、送信元によって報告される、同じリンク上で送信されたパケットの個数よりも著しく多いかどうかの判定が行われる。つまり、ノードｊのネイバーがノードｋで

ある、ノードｉからノードｊへのリンクｌについて、しきい値ｔが与えられたとすると、以下の判定が行われる。

【０１１５】

$recv_k(l) - sent_i(l) > t$

したがって、受信されたパケットの個数が送信されたパケットの個数に対応しない場合（つまり、近似）（ブロック９０６）、レポート内の不一致が指摘される。そうでない場合、プロシージャ９００はブロック９０８に進む。

【０１１６】

決定ブロック９０８で、送信元のネイバーにより報告される、リンク上で送信されたパケットの個数が、送信元によって報告される、同じリンク上で送信されたパケットの個数よりも著しく多いかどうかの判定が行われる。つまり、ノードｉからノードｊへのリンクｌ、ｉのネイバーｋについて、しきい値ｔが与えられたとすると、以下が判定される。

【０１１７】

$sent_k(i) - sent_i(l) > t$

上記の式に示されているように、送信されたパケットの個数が、それぞれ、送信元および近隣ノードにより指示された送信された（ブロック９０８）バケットの個数を近似している場合、レポート内の不一致が指摘される。そうでない場合、レポートに不一致はない（ブロック９１０）。

【０１１８】

決定ブロック９１２で、ノードの不一致のある対がすでに不一致グラフ内に含まれているかどうかの判定が行われる。含まれていなければ、ノードは不一致グラフに追加される（ブロック９１４）。ノードの不一致のある対がすでに不一致グラフ内にあるか（ブロック９１２）、または不一致グラフに追加されている場合（ブロック９１４）、不一致グラフ内のノード間に辺が加えられる（ブロック９１６）。

【０１１９】

不一致のある対のそれぞれが識別された後、ブロック９１８で、観察された不一致を説明できるノードの最小集合（つまり、最小数）が不一致グラフ内に見つかる。例えば、ネットワーク内の大半のノードが信頼できるレポートを送信するという仮定を置くことができる。したがって、観察された不一致を説明できるノードの最小集合が見つかる。これは、例えば、不一致グラフを被覆する頂点の最小集合を見つけることにより得られ、その場合、識別された頂点は不正挙動ノードを表す。

【０１２０】

頂点の最小集合は、ＮＰ困難であることが知られている、最小頂点被覆問題を利用することにより見つけることができる。残された辺がなくなるまで最も多い辺およびつながっている辺を持つノードを繰り返し選択して削除する貪欲アルゴリズム（ｇｒｅｅｄｙ　ａｌｇｏｒｉｔｈｍ）を適用する。

【０１２１】

レポートの履歴を使用して、さらに、不一致検出の精度を改善することができる。例えば、ブロック９２０で、新しいレポートを追加して、前の情報を削除することなく不一致グラフを更新することができる。その後、プロシージャ９００のブロック９１２～９１８を使用して、新しいレポート内のノードの不一致の対を処理することができる。例えば、ブロック９１８の同じ貪欲アルゴリズムを再適用して、不正挙動ノードを識別することができる。

【０１２２】

Ｗｈａｔ－Ｉｆ分析

前の節では、障害をネットワークシミュレーションに選択的に投入して、予想されるネットワークパフォーマンスと観察されたネットワークパフォーマンスとの差の原因と考えられる障害がもしあれば、その障害を識別した。ネットワークシミュレーションをさらに使用して、「ｗｈａｔ－ｉｆ」分析を実行して、ネットワークのオペレーションを改善することもできる。ｗｈａｔ－ｉｆ分析を使用すると、マネージャモジュールは、実行され

たときに、ネットワーク流パフォーマンスに対するさまざまな可能なネットワークおよびノード構成の影響を判別することができる。ｗｈａｔ－ｉｆ分析の結果は、ネットワーク内の選択されたノード上のエージェントモジュールにそれぞれのノードを相応に構成させるなどして、マネージャモジュールがネットワークを効率よく運用できるようにする処置の集まりである。

【０１２３】

例えば、ｗｈａｔ－ｉｆ分析は、すでに説明しているように、オンライントレース駆動シミュレーションを使用して実行することができる。以下の説明では、シミュレータを駆動するために集められるトレース例が示されている（例えば、図２のシミュレータ１１８（Ｎ））。例えば、シミュレータを使用することにより、現実のネットワークのネットワークシミュレーションを行うことができる。ネットワークシミュレーションは、さまざまなノードおよびネットワーク構成をテストし、既存のトラフィック条件に対し最良の総合的ネットワークパフォーマンスが得られる構成を決定するように再構成することができる。その後、マネージャモジュールは、構成に基づいてネットワーク内の特定のノードによる実装に対する処置の集合を決定することができる。

【０１２４】

ｗｈａｔ－ｉｆ分析に使用された従来の手法では、簡略化されたネットワークモデルを使用し、予想されるパフォーマンスを解析的に導いていた。しかし、オンライントレース駆動シミュレーションは、シミュレータを使用することにより発見的または理論的手法自体で可能である以上の、ネットワークの挙動に対するより高度な洞察が得られるという点で理論的分析に勝る利点を有する。例えば、稼働している無線ネットワークは、複雑なシステムであり、トラフィックフロー、ネットワーキングプロトコル、信号処理アルゴリズム、ハードウェア、ＲＦ伝播、および最も重要である、それらのこれらの断片のそれぞれの間の相互作用を含む多数の入り組んだ断片を伴う。さらに、ネットワークの挙動は、互いの範囲内にあるノード同士の相互作用および近傍内のノイズ源により左右される。発見的手法も理論的手法も、そのようなネットワークの挙動および異なるコンポーネント間の相互作用を捕捉しない。

【０１２５】

図１０は、ｗｈａｔ－ｉｆ分析がオンライントレース駆動シミュレーションに基づき実行される一実施例のプロシージャ１０００を示す流れ図である。高レベルでは、プロシージャ１０００は、最初に、ネットワークシミュレーションを使用して現実のネットワークを再現する。ネットワークに加えた修正の結果は、現実のネットワークに適用された場合、その後、ネットワークシミュレーションにおける変更を適用し、ネットワークパフォーマンスの関係を定量化することにより判定される。

【０１２６】

ブロック１００２で、マネージャモジュールを実行することにより、複数の修正のうちの１つまたは複数を選択する。修正は、いろいろな方法で選択できる。例えば、修正は、マネージャモジュールにより、ネットワークパフォーマンスの低下ではなく、増大を引き起こす障害としてみなされることも可能である。このような場合の修正は、図２の障害ディレクトリ２１８内に格納し、タイプに基づいて配列することができる。ブロック１００４では、分析モジュールは、現実のネットワークのネットワーク設定および選択された修正を含む修正集合をネットワークシミュレーションに入力として供給する。

【０１２７】

ブロック１００６で、ネットワークの予想されるパフォーマンスが入力に基づいて予測される。例えば、シミュレータは、現実のネットワークのネットワーク設定および修正集合に基づいてネットワークシミュレーションを実行することができる。ネットワークシミュレーションは、すでに説明されているように、現実のネットワークに対する修正の結果を判定するために使用することができる。

【０１２８】

分析モジュールは、実行されると、修正を実行するためネットワークのエージェントモ

ジュールにより実行される１つまたは複数の処置を導出する（ブロック１００８）。例えば、分析モジュールは、対応する修正にマッピングされる処置のディレクトリを含むことができる。その後、分析モジュールは、それらの修正に基づいて対応する処置を取得することができる。

【０１２９】

ブロック１０１０で、分析モジュールは、対応するエージェントモジュールとの通信のための１つまたは複数の処置を記述する通信を形成する。その後、対応するエージェントモジュールは、ネットワークのそれぞれのノードにそこに記述されている処置を実行させることができる。したがって、マネージャおよびエージェントモジュールを使用することにより、障害検出と似た方法でオンライントレース駆動シミュレーションに基づいてｗｈａｔ－ｉｆ分析実行することができる。ｗｈａｔ－ｉｆ分析は、障害を是正し、ネットワークパフォーマンスを改善するために使用できる。

【０１３０】

他の実施例では、シミュレーションは、反復法を使用してｗｈａｔ－ｉｆ分析を実行することなどによりネットワークパフォーマンスを改善するためネットワークに対し行う修正を決定する場合に使用される。この方法は、図５および７に関して説明されているようにシミュレーションに類似している。したがって、反復精密化は、複数の修正処置が必要な場合に使用することが可能である。

【０１３１】

図１１は、損害を与える流れの診断に基づいてネットワークへの修正が導かれる一実施例のプロシージャ１１００を示す流れ図である。ブロック１１０２で、マネージャモジュール（例えば、図１および２のマネージャモジュール１１６（Ｎ））は、ネットワーク内の１つまたは複数のフローで、対応する予想される目標スループット値よりも低いスループット値が生じていることを判定するために実行される。ブロック１１０４で、マネージャモジュールは、ネットワーク内にフローがある場合に、どれが「ダメージを与えるフロー」なのかを判別する。ダメージを与えるフローは、存在しているとネットワークスループットの重大な低下を引き起こすタイプの障害であり、ダメージを与えるフローはそれ自体では健全である場合もあるが、他の競合するフローと適切に相互作用しないという点で前の障害とは異なる。

【０１３２】

例えば、ブロック１１０６で、目標の終端間の要求および使用しているルーティングプロトコルを記述しているネットワーク設定が収集される。プロシージャ１１００は、ネットワーク（例えば、リンク負荷およびルーティング）がネットワーク構成の変化にどのように反応するかを調べるため、これらのネットワーク設定は、トラブルシューティングに使用されるトレースとは異なる場合があることに留意されたい。

【０１３３】

ブロック１１０８で、ネットワークスループット全体に対する影響が、ネットワークシミュレーションから各フローを一度に１つずつ削除することに基づいて調べられる。一実施形態では、ダメージを与えるフローは、除去がネットワークパフォーマンスに対する最も著しい総合的改善をもたらす１つのフローとして識別される。例えば、複数のフロー１２０２から１２１６を含むネットワーク１２００が図１２に示されている。フロー８　１２１６（図１２でF_8として例示されている）は、例示されているネットワーク１２００内の他のフロー１２０２～１２１４のそれぞれを横断する。したがって、フロー８　１２０８を削除すると、他のフロー１２０２～１２１４のどれかを削除した場合とは反対に、スループットの増大が最大となる可能性がある。つまり、フロー８　１２１６が存在すると、ネットワーク１２００のパフォーマンスに対するダメージが最大になるということである。このようにして、ネットワークパフォーマンスの向上が最大になるネットワーク１２００に対する修正（例えば、システムの他のフローに対するフロー８　１２１６の影響の除去または低減）を決定することができる。

【０１３４】

ブロック１１１０で、ネットワークパフォーマンスを改善するために使用できる修正に基づいて１つまたは複数の処置が導かれる。処置の例として、フロー８　１２１６の律速、ルート変更、およびトポロジ制御がある。ネットワークシミュレーションを行うことにより、マネージャモジュールは、さらに、それらの処置のメリットを正確に評価することができる。例えば、以下の表は、是正処置の実施例に対する予想されるスループットを示している。

【０１３５】

【表２】

処置	トータルスループット（Mbps）
処置なし	1.064
フロー8の速度を半分にする	1.148
グリッド境界を介してフロー8のルートを決定する	1.217
伝送出力を20dBMに上げる	0.990
伝送出力を25dBMに上げる	1.661

【０１３６】

表に示されているように、伝送出力を２５ｄＢＭに上げると、考察対象の４つの処置の実施例（および１つの無処置）の間で最も高いスループットが得られるが、それは、送信先に到達するために必要なホップ回数が減るからである。これらの結果に基づき、マネージャモジュールは、それぞれのノード上のエージェントの１つまたは複数に、出力を上げてネットワークパフォーマンス問題を緩和させる通信を形成する。

【０１３７】

フレームワーク実施例

説明されているフレームワークの例は、ＷＩＮＤＯＷＳ（登録商標）ＸＰプラットフォーム（ＷＩＮＤＯＷＳ（登録商標）ＸＰは、ワシントン州レッドモンドのＭｉｃｒｏｓｏｆｔ　Ｃｏｒｐ．の商標である）上に実装されている。実施例のコンポーネント、設計原理、およびその特徴については、この節で説明される。

【０１３８】

この場合のフレームワーク実施例は、エージェントモジュールとマネージャモジュールの２つのコンポーネントを含む。図１に関してすでに説明されているように、エージェントモジュールは、ネットワークの各ノード上で実行され、ローカルデータを定期的にまたはオンデマンドで報告する。マネージャモジュールは、エージェントモジュールから関連データを収集し、図２に関して説明されているように含まれている分析モジュールの実行などを通じて実行されるとデータを分析する。

【０１３９】

フレームワーク実施例では、簡素であることと拡張性があることという設計原理を採用している。例えば、監視および管理のため収集され伝播されるデータは、ＷＩＮＤＯＷＳ（登録商標）（ＷＩＮＤＯＷＳ（登録商標）は、ワシントン州レッドモンドのＭｉｃｒｏｓｏｆｔ　Ｃｏｒｐ．の商標である）上でサポートされているパフォーマンスカウンタに投入することができる。パフォーマンスカウンタは、カテゴリ別にグループ分けされた（名前、値）の対として実現することができる。

【０１４０】

説明されているフレームワークも拡張可能である。監視対象のデータに追加するには、パフォーマンスカウンタの新しいカテゴリーを作成し、情報の変更に合わせてパフォーマンスカウンタ値を更新するモジュールを書く必要がある。伝送制御プロトコル（ＴＣＰ）、ユーザデータグラムプロトコル（ＵＤＰ）、インターネットプロトコル（ＩＰ）、およびワークステーションリモートアプリケーションプログラミングインターフェイス（ＷＲＡＰＩ）に関係するパフォーマンスデータは、ほとんど追加作業もなくフレームワークに組み込むことができる。

【０１４１】
これらのパフォーマンスカウンタ内の値は、読み取り専用または書き込み可能である。例えば、書き込み可能カウンタは、許可されたマネージャノードが、異なるそれぞれのノート上で実行されるエージェントモジュールとマネージャモジュールとの通信などを通じて問題を解決したり、リモートから実験を開始したりするため、値を変更し、ノートの挙動に影響を及ぼす手段となる。
【０１４２】
それぞれのマネージャノードは、さらに、実施例が図１３に示されているグラフィカルユーザインターフェイス（ＧＵＩ）１３００を備え、ネットワーク管理者と対話形式でやり取りすることもできる。ＧＵＩを使用すると、管理者は、ネットワークをビジュアル化するだけでなく、管理要求をマネージャモジュールを通じて発行することもできる。ＧＵＩ　１３００には、ネットワークのたたき台となるトポロジが表示される。その場合、ＧＵＩ　１３００は、エージェントが２３個のノードからなるたたき台上に配備されているマネージャウィンドウを表示する。マネージャモジュールは、直接得られた、または推論されたノードの相対座標に基づいてトポロジを表示することができる。ＧＵＩ　１３００を使用すると、さらに、管理者はネットワークの特定の部分を拡大して詳細情報を表示し、リンクをクリックすることで、特定のリンクに関するネットワークパフォーマンスデータを表形式で表示することができる。
【０１４３】
結論
本発明は構造的機能および／または方法論的活動に固有の言語で説明されているが、付属の請求項で定められている発明は、説明した特定の機能または活動に必ずしも限られないことは理解されるであろう。むしろ、特定の機能および活動は請求されている発明を実施するための複数の実施形態の例として開示されている。
【図面の簡単な説明】
【０１４４】
【図１】複数のノードを有するネットワークを示す実施例における一環境の図である。
【図２】図１の分析モジュールを詳しく示している一実施例の図である。
【図３】７×３グリッドトポロジ（ｇｒｉｄ　ｔｏｐｏｌｏｇｙ）を有するネットワークの図である。
【図４】シミュレータを含むシステムおよび図２のネットワークシミュレーションを示す実施例の図である。
【図５】同じタイプの障害が最初に次々に診断される一実施例のプロシージャを示す流れ図である。
【図６】推定されたパフォーマンスと観察されたパフォーマンスとの違いに基づいて障害のタイプを判定するために使用できる実施例の決定木（ｄｅｃｉｓｉｏｎ　ｔｒｅｅ）の図である。
【図７】反復診断アルゴリズムを使用して同じタイプの障害を次々に診断する一実施例のプロシージャを示す流れ図である。
【図８】図１の複数のノードがネイバー監視を行うために実行可能なエージェントモジュール（ａｇｅｎｔ　ｍｏｄｕｌｅｓ）を備える一実施例内のネットワークの図である。
【図９】ネットワーク内の不正挙動ノードを特定するために近隣ノードを記述しているレポートを比較する一実施例のプロシージャを示す流れ図である。
【図１０】ｗｈａｔ－ｉｆ分析がオンライントレース駆動シミュレーション（ｏｎｌｉｎｅ　ｔｒａｃｅ－ｄｒｉｖｅｎ　ｓｉｍｕｌａｔｉｏｎ）に基づき実行される一実施例のプロシージャを示す流れ図である。
【図１１】損害を与える流れの診断に基づいてネットワークへの修正が導かれる一実施例のプロシージャを示す流れ図である。
【図１２】複数の流れを含み、そのうちの１つが損害を与える流れである、ネットワークの図である。

【図１３】ネットワーク管理者がネットワークをビジュアル化し、管理要求をネットワークに対し発行できるようにするマネージャノード（ｍａｎａｇｅｒ　ｎｏｄｅ）が備えるグラフィカルユーザインターフェイス（ＧＵＩ）（ｇｒａｐｈｉｃａｌ　ｕｓｅｒ　ｉｎｔｅｒｆａｃｅ）を示す実施例の図である。

【符号の説明】

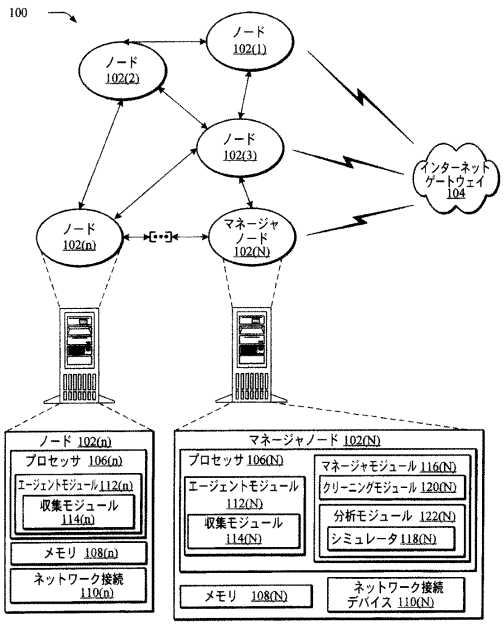
【０１４５】

１０２　ノード

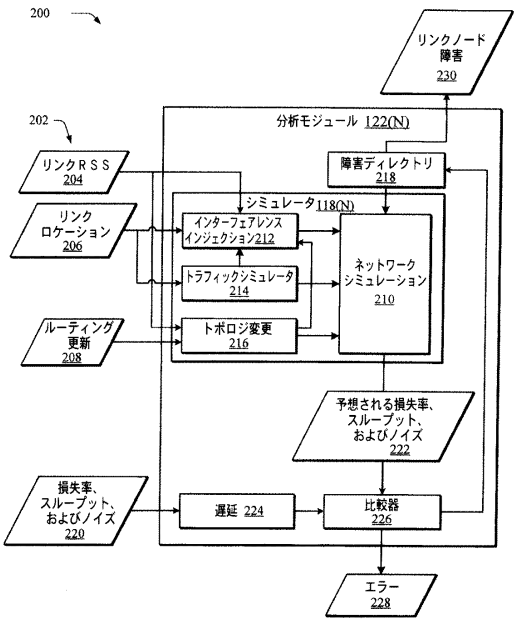
１０６　プロセッサ

１１４　収集モジュール

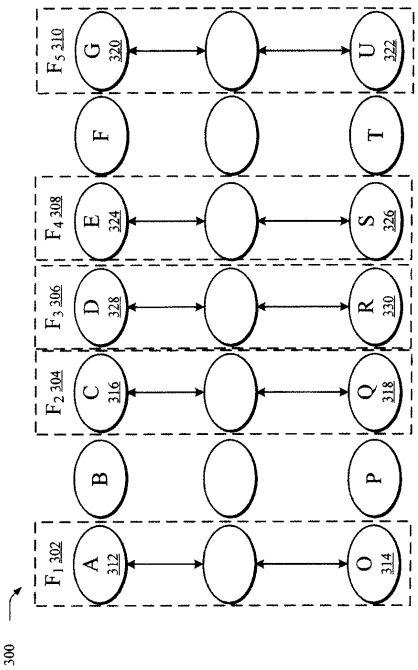
【図１】



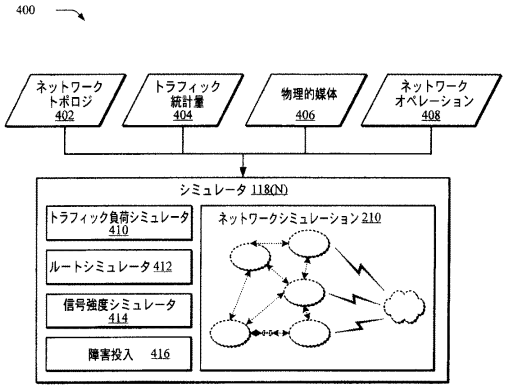
【図２】



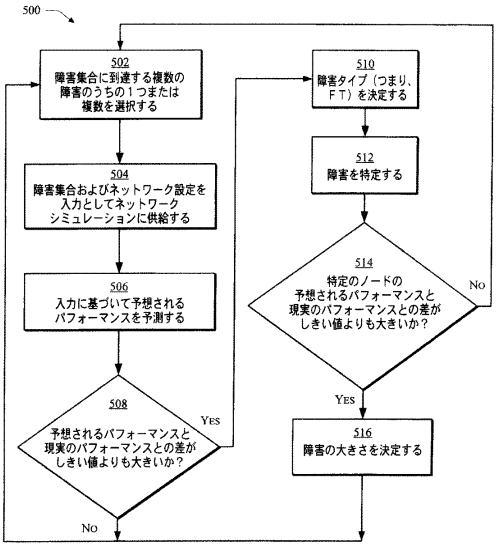
【図３】


F5 310
G 320
U 322
F
T
F4 308
E 324
S 326
F3 306
D 328
R 330
F2 304
C 316
Q 318
B
P
F1 302
A 312
O 314
300

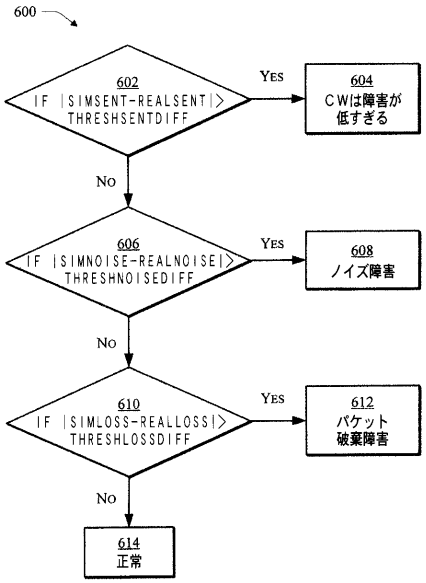
【図４】


400
ネットワーク トポロジ 402
トラフィック 統計量 404
物理的媒体 406
ネットワーク オペレーション 408
シミュレータ 118(N)
トラフィック負荷シミュレータ 410
ルートシミュレータ 412
信号強度シミュレータ 414
障害投入 416
ネットワークシミュレーション 210

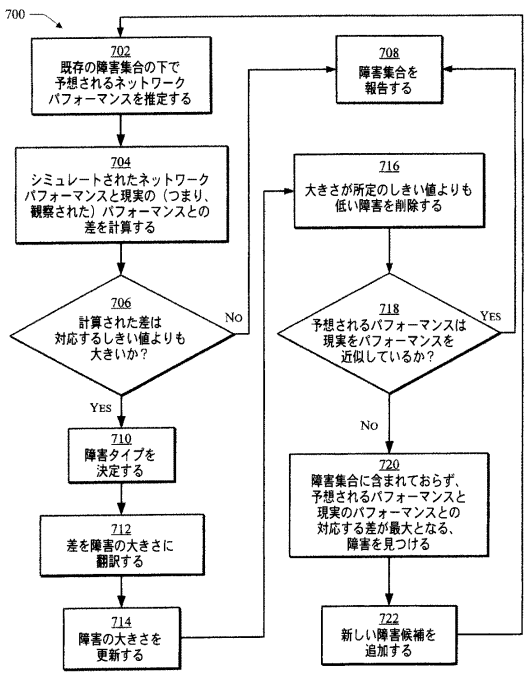
【図５】


500
502 障害集合に到達する複数の障害のうちの１つまたは複数を選択する
504 障害集合およびネットワーク設定を入力としてネットワークシミュレーションに供給する
506 入力に基づいて予想されるパフォーマンスを予測する
508 予想されるパフォーマンスと現実のパフォーマンスとの差がしきい値よりも大きいか？
Yes
No
510 障害タイプ（つまり、FT）を決定する
512 障害を特定する
514 特定のノードの予想されるパフォーマンスと現実のパフォーマンスとの差がしきい値よりも大きいか？
No
Yes
516 障害の大きさを決定する

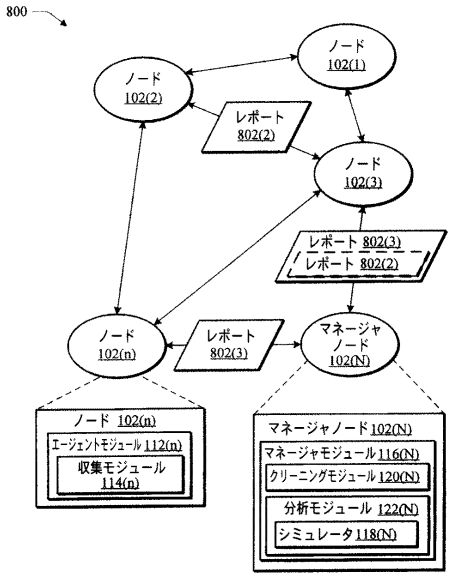
【図６】


600
602 IF |SIMSENT-REALSENT|> THRESHSENTDIFF
Yes
604 CWは障害が低すぎる
No
606 IF |SIMNOISE-REALNOISE|> THRESHNOISEDIFF
Yes
608 ノイズ障害
No
610 IF |SIMLOSS-REALLOSS|> THRESHLOSSDIFF
Yes
612 パケット破棄障害
No
614 正常

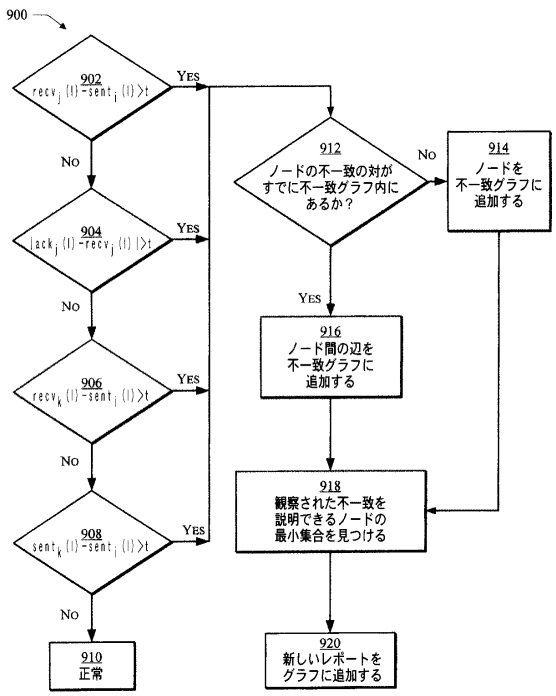
【図７】


700
702
既存の障害集合の下で
予想されるネットワーク
パフォーマンスを推定する
704
シミュレートされたネットワーク
パフォーマンスと現実の（つまり、
観察された）パフォーマンスとの
差を計算する
706
計算された差は
対応するしきい値よりも
大きいか？
No
Yes
708
障害集合を
報告する
710
障害タイプを
決定する
712
差を障害の大きさに
翻訳する
714
障害の大きさを
更新する
716
大きさが所定のしきい値よりも
低い障害を削除する
718
予想されるパフォーマンスは
現実をパフォーマンスを
近似しているか？
Yes
No
720
障害集合に含まれておらず、
予想されるパフォーマンスと
現実のパフォーマンスとの
対応する差が最大となる、
障害を見つける
722
新しい障害候補を
追加する

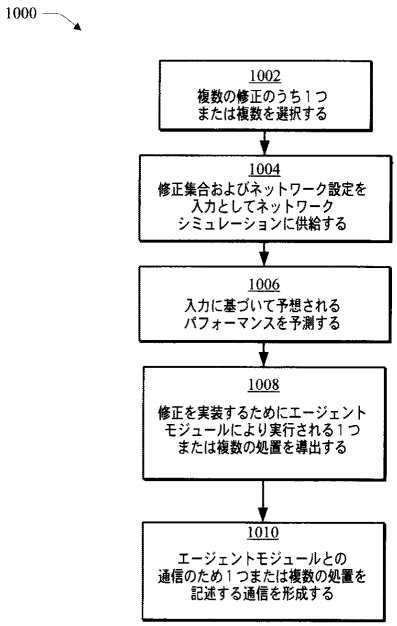
【図８】


800
ノード
102(1)
ノード
102(2)
レポート
802(2)
ノード
102(3)
レポート 802(3)
レポート 802(2)
ノード
102(n)
レポート
802(3)
マネージャ
ノード
102(N)
ノード 102(n)
エージェントモジュール 112(n)
収集モジュール
114(n)
マネージャノード 102(N)
マネージャモジュール 116(N)
クリーニングモジュール 120(N)
分析モジュール 122(N)
シミュレータ 118(N)

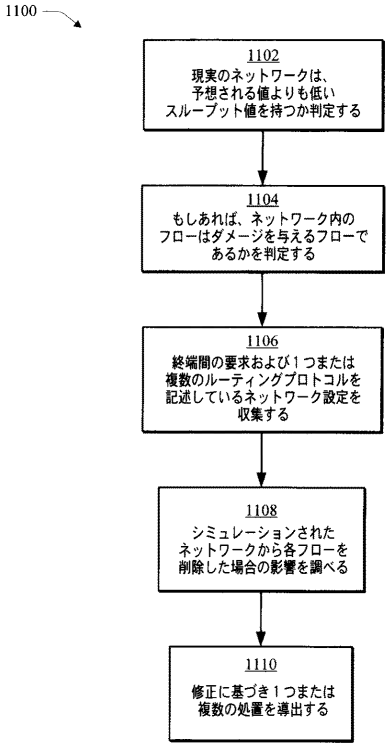
【図９】


900
902
recv_j(t)-sent_i(t)>t
Yes
No
904
|ack_j(t)-recv_j(t)|>t
Yes
No
906
recv_k(t)-sent_i(t)>t
Yes
No
908
sent_k(t)-sent_i(t)>t
Yes
No
910
正常
912
ノードの不一致の対が
すでに不一致グラフ内に
あるか？
No
Yes
914
ノードを
不一致グラフに
追加する
916
ノード間の辺を
不一致グラフに
追加する
918
観察された不一致を
説明できるノードの
最小集合を見つける
920
新しいレポートを
グラフに追加する

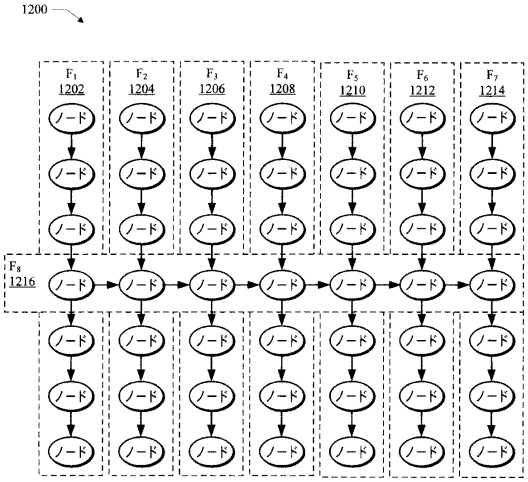
【図１０】


1000
1002
複数の修正のうち１つ
または複数を選択する
1004
修正集合およびネットワーク設定を
入力としてネットワーク
シミュレーションに供給する
1006
入力に基づいて予想される
パフォーマンスを予測する
1008
修正を実装するためにエージェント
モジュールにより実行される１つ
または複数の処置を導出する
1010
エージェントモジュールとの
通信のため１つまたは複数の処置を
記述する通信を形成する

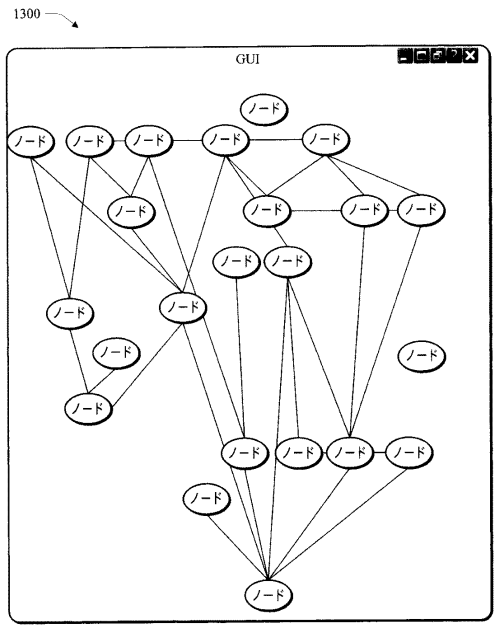
【図１１】


1100
1102
現実のネットワークは、
予想される値よりも低い
スループット値を持つか判定する
1104
もしあれば、ネットワーク内の
フローはダメージを与えるフローで
あるかを判定する
1106
終端間の要求および１つまたは
複数のルーティングプロトコルを
記述しているネットワーク設定を
収集する
1108
シミュレーションされた
ネットワークから各フローを
削除した場合の影響を調べる
1110
修正に基づき１つまたは
複数の処置を導出する

【図１２】


1200
F1 1202
F2 1204
F3 1206
F4 1208
F5 1210
F6 1212
F7 1214
F8 1216
ノード

【図１３】


1300
GUI
ノード

フロントページの続き

(72)発明者　リドン　チョウ
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
(72)発明者　リリー　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
(72)発明者　パラミール　バル
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　アナス　ラージャゴーパーラ　ラオ
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

審査官　永井　啓司

(56)参考文献　特開平０９－２０５４２９（ＪＰ，Ａ）
特開２００１－１６８８６２（ＪＰ，Ａ）
特開平０８－３２８９８４（ＪＰ，Ａ）
特開平０７－２９５８５３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｈ０４Ｌ　１２／００－１２／２６
Ｈ０４Ｌ　１２／５０－１２／６６