

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第5736042号  
(P5736042)

(45) 発行日 平成27年6月17日 (2015. 6. 17)

(24) 登録日 平成27年4月24日 (2015. 4. 24)

(51) Int. Cl.

F I

GO 6 N 3/04 (2006. 01)  
 GO 6 N 3/063 (2006. 01)  
 GO 6 N 99/00 (2010. 01)  
 GO 6 T 7/00 (2006. 01)

GO 6 N 3/04  
 GO 6 N 3/063  
 GO 6 N 99/00 1 5 0  
 GO 6 T 7/00 3 0 0 B

請求項の数 37 (全 29 頁)

(21) 出願番号 特願2013-514372 (P2013-514372)  
 (86) (22) 出願日 平成23年6月9日 (2011. 6. 9)  
 (65) 公表番号 特表2013-533542 (P2013-533542A)  
 (43) 公表日 平成25年8月22日 (2013. 8. 22)  
 (86) 国際出願番号 PCT/US2011/039849  
 (87) 国際公開番号 W02011/156634  
 (87) 国際公開日 平成23年12月15日 (2011. 12. 15)  
 審査請求日 平成26年6月9日 (2014. 6. 9)  
 (31) 優先権主張番号 12/943, 551  
 (32) 優先日 平成22年11月10日 (2010. 11. 10)  
 (33) 優先権主張国 米国 (US)  
 (31) 優先権主張番号 61/353, 546  
 (32) 優先日 平成22年6月10日 (2010. 6. 10)  
 (33) 優先権主張国 米国 (US)

(73) 特許権者 595168543  
 マイクロン テクノロジー, インク,  
 アメリカ合衆国, アイダホ州 8 3 7 1 6  
 - 9 6 3 2, ボイズ, サウス フェデ  
 ラル ウェイ 8 0 0 0  
 (74) 代理人 100106851  
 弁理士 野村 泰久  
 (74) 代理人 100074099  
 弁理士 大菅 義之  
 (72) 発明者 ドラゴッシュ, ポール ディー,  
 アメリカ合衆国, アイダホ州 8 3 6 1 6  
 , イーグル, ウェスト モーガン クリー  
 ク コート 4 4 0 2

審査官 石川 亮

最終頁に続く

早期審査対象出願

(54) 【発明の名称】 階層構造を使用するデータの分析

(57) 【特許請求の範囲】

【請求項 1】

第 1 のデータが第 1 のパターンに一致するかを決定することによって、複数のハードウェア要素上で実行する第 1 の並行処理機械を使用して前記第 1 のデータを分析することと、  
 前記第 1 の並行処理機械から第 2 のデータを出力することであって、前記第 2 のデータが、前記第 1 のデータの分析から生成され、前記第 1 のデータを分析した結果の表示を提供する、ことと、

前記第 2 のデータが第 2 のパターンに一致するかを決定することによって、複数のハードウェア要素上で実行する第 2 の並行処理機械を使用して前記第 2 のデータを分析することと、

前記第 2 のデータを分析した結果を、前記第 2 の並行処理機械から出力することと、

前記第 2 の並行処理機械によって前記第 2 のデータ内で識別される前記第 2 のパターンに基づいて、引き続き動作のために前記第 1 の並行処理機械を構成することと、

を含む、データを分析するための方法。

【請求項 2】

前記第 2 のデータが、前記第 1 のデータが前記第 1 のパターンと一致するかどうかの表示を提供する、請求項 1 に記載の方法。

【請求項 3】

前記第 2 のデータを分析した結果が、前記第 2 のデータが前記第 2 のパターンと一致したかどうかの表示を含む、請求項 1 に記載の方法。

**【請求項 4】**

前記第 2 のデータが、前記第 1 のパターンとの一致を示す単一のビットを含む、請求項 1 に記載の方法。

**【請求項 5】**

前記第 2 のデータが、複数のパターンに対する一致および非一致を示す複数のビットを備えるワードを備え、前記複数のパターンが前記第 1 のパターンを含む、請求項 1 に記載の方法。

**【請求項 6】**

前記第 2 のデータが、前記第 1 の並行処理機械のプログラマブル要素の状態に対応する状態ベクトルを含む、請求項 1 に記載の方法。

10

**【請求項 7】**

第 1 のデータを分析することが、前記第 1 のデータが関心のあるシーケンスを含むかどうかを決定することを含む、請求項 1 に記載の方法。

**【請求項 8】**

前記第 1 の並行処理機械から第 2 のデータを出力することが、前記第 1 のデータに対する、前記第 1 の並行処理機械のプログラマブル要素の反応に基づいて第 2 のデータを出力することを含む、請求項 1 に記載の方法。

**【請求項 9】**

第 1 のデータを分析することが、データシーケンスが、前記第 1 のデータ内で識別されるかどうかを決定することを含む、請求項 1 に記載の方法。

20

**【請求項 10】**

前記第 1 のデータが未処理データを備える、請求項 1 に記載の方法。

**【請求項 11】**

前記第 2 のデータを分析することが、前記第 2 の並行処理機械を使用して前記第 1 のデータを分析することを含む、請求項 1 に記載の方法。

**【請求項 12】**

前記第 2 のデータが、前記第 1 の並行処理機械の検出状態の表示を提供する、請求項 1 に記載の方法。

**【請求項 13】**

前記第 2 のデータが、前記第 1 の並行処理機械の検出状態の表示を提供する、請求項 1 に記載の方法。

30

**【請求項 14】**

前記第 2 の並行処理機械に、前記第 2 のデータを分析する前記結果をフィードバックすることをさらに含み、前記第 2 の並行処理機械を使用して前記第 2 のデータを分析することが、前記第 2 のデータを分析する前記結果を分析することをさらに含む、請求項 1 に記載の方法。

**【請求項 15】**

前記第 2 のパターンが、前記第 1 のデータで識別されるパターンの内のパターンを備える、請求項 3 に記載の方法。

**【請求項 16】**

前記第 1 の並行処理機械を構成することが、

前記第 2 のデータで識別される前記第 2 のパターンを分析することと、前記第 2 のデータで識別される前記第 2 のパターンに基づいて前記第 1 の並行処理機械のためのプログラムをコンパイルすることとを含む、請求項 1 に記載の方法。

40

**【請求項 17】**

前記プログラムが、更新されたプログラムを備える、請求項 16 に記載の方法。

**【請求項 18】**

データを分析する方法であって、

第 1 のデータが第 1 のパターンに一致するかを決定することによって、複数のハードウェア要素上で実行する第 1 の並行処理機械を使用して前記第 1 のデータを分析することと

50

、前記第1の並行処理機械から第2のデータを出力することであって、前記第2のデータが、前記第1のデータを分析した結果の表示を提供する、ことと、

前記第2のデータが第2のパターンに一致するかどうかを決定することによって、前記複数のハードウェア要素上で実行する第2の並行処理機械を使用して前記第2のデータを分析することと、

前記第2のデータが、第2の複数のパターンの内の1つ以上<sup>10</sup>に一致するかどうかを決定することを含み、結果を出力することと、

第3のデータを出力することによって、前記第2のデータを分析した結果を前記第2の並行処理機械から出力することであって、前記第3のデータが、前記第2の複数のパターンの内の1つ以上<sup>10</sup>が前記第2のデータで識別されたかどうかの表示を提供する、ことと、

第3の並行処理機械を使用して、前記第3のデータが第3の複数のパターンの内の1つ以上と一致するかどうかを決定することと、

前記第3の複数のパターンの内の1つ以上<sup>10</sup>が前記第3のデータで識別されたかどうかの表示を、前記第3の並行処理機械から出力することと、

前記第3の並行処理機械からの出力に基づいて、前記第1の並行処理機械および前記第2の並行処理機械の内の少なくとも一方<sup>10</sup>を構成することと、

を含む、方法。

#### 【請求項19】

第1のデータ入力ポートおよび第1の出力ポートを有する、複数のハードウェア要素上で実行するように構成された第1の並行処理機械であって、前記第1の並行処理機械が、前記第1のデータ入力ポートで第1のデータを受け取り、第1のパターンが前記第1のデータで識別されるかどうかの表示を提供する第2のデータを、前記第1の出力ポートで出力するように構成される、第1の並行処理機械と、

第2のデータ入力ポートおよび第2の出力ポートを有する、前記複数のハードウェア要素上で実行するように構成された第2の並行処理機械であって、前記第2のデータ入力ポートが前記第1の並行処理機械の前記第1の出力ポートに結合され、前記第2の並行処理機械が、前記第1の並行処理機械から前記第2のデータ入力ポートで前記第2のデータを受け取り、第2のパターンが前記第2のデータで識別されるかどうかの表示を提供する第3のデータを、前記第2の出力ポートで出力するように構成される、第2の並行処理機械と、

を備え、

前記第1の並行処理機械は、前記第3のデータに基づいて、前記第1の並行処理機械を再プログラミングするように構成されたプログラムを受け取るように構成されるプログラミングインタフェースを含む、データを処理するための装置。

#### 【請求項20】

第3のデータ入力ポートおよび第3の出力ポートを有する第3の並行処理機械をさらに備え、前記第3の出力ポートが、前記第2の並行処理機械の前記第2のデータ入力ポートに結合される、請求項19に記載の装置。

#### 【請求項21】

前記第2の並行処理機械が、前記第2のデータ入力ポートで前記第1のデータを受け取るようにさらに構成され、かつ、前記第2のデータおよび前記第1のデータの組合せで前記第2のパターンを識別するようにさらに構成される、請求項19に記載の装置。

#### 【請求項22】

前記プログラムが、更新されたプログラムを含み、前記更新されたプログラムが、前記第1の並行処理機械にすでにロードされたプログラムを更新するように構成される、請求項19に記載の装置。

#### 【請求項23】

前記プログラムが、前記第1の並行処理機械のためのオリジナルプログラムを備える、請求項19に記載の装置。

#### 【請求項24】

前記第 2 の並行処理機械の前記出力ポートに結合され、前記第 2 の並行処理機械の前記出力ポートを分析し、前記第 2 の並行処理機械から出力された前記第 3 のデータの関数として、前記第 1 の並行処理機械のための更新プログラムをコンパイルするように構成される処理装置をさらに備え、

前記第 1 の並行処理機械が、前記処理装置から更新されたプログラムを受け取るように構成される、請求項 19 に記載の装置。

【請求項 25】

第 3 のデータ入力ポートおよび第 3 の出力ポートを有する第 3 の並行処理機械を備え、前記第 3 の並行処理機械が、

前記第 2 の並行処理機械からの前記表示を受け取り、

第 3 のパターンが前記表示内にあるかどうかを決定し、

出力を生成する、

ように構成される、請求項 24 に記載の装置。

【請求項 26】

複数のハードウェア要素上で実行する第 1 の並行処理機械で第 1 のデータを受け取ることと、

前記第 1 のデータの分析から生成される第 2 のデータを、前記第 1 の並行処理機械から出力することと、

前記複数のハードウェア要素上で実行する第 2 の並行処理機械で前記第 2 のデータを受け取ることと、

前記第 2 のデータの分析から生成される第 3 のデータを、前記第 2 の並行処理機械から出力することと、

前記第 1 の並行処理機械での引き続く動作のために、前記第 2 の並行処理機械から出力される前記第 3 のデータに基づいて前記第 1 の並行処理機械を構成することと、

を含み、

前記第 1 の並行処理機械は前記第 1 のデータ内で第 1 のパターンを識別するように構成され、前記第 2 の並行処理機械は前記第 2 のデータ内で第 2 のパターンを識別するように構成される、

データを分析するための方法。

【請求項 27】

構成することが再プログラムすることを含む、請求項 26 に記載の方法。

【請求項 28】

前記第 1 の並行処理機械を構成することが、

前記第 3 のデータに基づいて、前記第 1 の並行処理機械のための更新されたプログラムをコンパイルすることと、

前記第 1 の並行処理機械に前記更新プログラムをロードすることと、を含む、請求項 26 に記載の方法。

【請求項 29】

前記第 2 の並行処理機械で前記第 1 のデータを受け取ることとをさらに含み、前記第 2 の並行処理機械から第 3 のデータを出力することが、前記第 1 のデータおよび前記第 2 のデータの分析から生成される第 3 のデータを出力することを含む、請求項 26 に記載の方法。

【請求項 30】

複数のハードウェア要素上で実行する、階層構造の第 1 の層における第 1 の並行処理機械を用いて、入力データに関する下位レベルのパターン認識分析を実行することと、

前記第 1 の層によって、前記下位レベルのパターン認識分析に基づいて第 1 の出力を提供することと、

前記複数のハードウェア要素上で実行する、前記階層構造の第 2 の層における第 2 の並行処理機械を用いて、前記第 1 の出力に関する高レベルのパターン認識分析を実行することと、

10

20

30

40

50

前記第 2 の層によって、前記高レベルのパターン認識分析に基づく第 2 の出力を提供することと、

前記第 1 の層または前記第 2 の層の引き続くパターン認識分析を修正するよう、前記第 1 の出力または前記第 2 の出力にそれぞれ基づいて、前記第 1 の層または前記第 2 の層の少なくとも一方を変更することと、

前記階層構造の前記第 2 の層から、前記階層構造の前記第 1 の層にフィードバック情報を提供することと、

を含む、方法。

【請求項 3 1】

前記フィードバック情報が、前記下位レベルのパターン認識分析を更新するために使用される、請求項 3 0 に記載の方法。

【請求項 3 2】

前記フィードバック情報が、前記下位レベルのパターン認識分析を実行するために使用されるプログラムを更新するために使用される、請求項 3 1 に記載の方法。

【請求項 3 3】

前記フィードバック情報は、前記入力データに相当する情報を識別することを含む、請求項 3 1 に記載の方法。

【請求項 3 4】

複数のハードウェア要素上で実行する、階層構造の第 1 の層における第 1 の並行処理機械を用いて、入力データに関する下位レベルのパターン認識分析を実行することと、

前記第 1 の層によって、前記下位レベルのパターン認識分析に基づいて第 1 の出力を提供することと、

前記複数のハードウェア要素上で実行する、前記階層構造の第 2 の層における第 2 の並行処理機械を用いて、前記第 1 の出力に関する高レベルのパターン認識分析を実行することと、

前記第 2 の層によって、前記高レベルのパターン認識分析に基づく第 2 の出力を提供することと、

前記第 1 の層または前記第 2 の層の引き続くパターン認識分析を修正するよう、前記第 1 の出力または前記第 2 の出力にそれぞれ基づいて、前記第 1 の層または前記第 2 の層の少なくとも一方を変更することと、

前記高レベルのパターン認識分析に基づいて前記下位レベルのパターン認識分析を構成することと、

を含む、方法。

【請求項 3 5】

複数のハードウェア要素上で実行する、階層構造の第 1 の層における第 1 の並行処理機械を用いて、入力データに関する下位レベルのパターン認識分析を実行することと、

前記第 1 の層によって、前記下位レベルのパターン認識分析に基づいて第 1 の出力を提供することと、

前記複数のハードウェア要素上で実行する、前記階層構造の第 2 の層における第 2 の並行処理機械を用いて、前記第 1 の出力に関する高レベルのパターン認識分析を実行することと、

前記第 2 の層によって、前記高レベルのパターン認識分析に基づく第 2 の出力を提供することと、

前記第 1 の層または前記第 2 の層の引き続くパターン認識分析を修正するよう、前記第 1 の出力または前記第 2 の出力にそれぞれ基づいて、前記第 1 の層または前記第 2 の層の少なくとも一方を変更することと、

前記下位レベルのパターン認識分析に基づいて前記高レベルのパターン認識分析を構成することと、

を含む、方法。

【請求項 3 6】

10

20

30

40

50

データを処理するための装置であって、

複数のハードウェア要素上で実行するように構成された第1の並行処理機械であって、  
前記第1の並行処理機械は第1のデータ入力ポートおよび第1の出力ポートを有し、前記第1の並行処理機械は、前記第1のデータ入力ポートで第1のデータを受け取り、第1のパターンが前記第1のデータ内で識別されるかに基づいて、前記第1の出力ポートで第2のデータを出力するように構成される、第1の並行処理機械と、

複数のハードウェア要素上で実行するように構成された第2の並行処理機械であって、  
前記第2の並行処理機械は第2のデータ入力ポートおよび第2の出力ポートを有し、前記第2のデータ入力ポートは前記第1の並行処理機械の前記第1の出力ポートに接続され、前記第2の並行処理機械は、前記第2のデータ入力ポートで前記第2のデータを受け取り、前記第2の出力ポートで、第2のパターンが前記第2のデータで識別されるかどうかの表示を出力するように構成される、第2の並行処理機械と、

前記第2の並行処理機械の前記出力ポートに接続され、前記第2の並行処理機械の前記出力ポートを分析し、前記第2の並行処理機械から出力される前記表示の関数として、前記第1の並行処理機械に対し更新プログラムをコンパイルするように構成される処理デバイスと、

を含み、

前記第1の並行処理機械は、前記処理デバイスから更新プログラムを受け取るように構成されている、装置。

【請求項37】

複数のハードウェア要素上で実行するように構成された第3の並行処理機械をさらに含み、前記第3の並行処理機械は第3のデータ入力ポートおよび第3の出力ポートを有し、前記第3の並行処理機械は、

前記第2の並行処理機械から前記表示を受け取り、

前記表示中に第3のパターンがあるかどうかを決定し、

出力を生成する、

ように構成される、請求項36に記載の装置。

【発明の詳細な説明】

【技術分野】

【0001】

(優先権主張)

本特許出願は、参照することによりとも本明細書に含まれる、2010年6月10日に出願された「Hierarchical Pattern Recognition」と題する米国仮特許出願番号第61/353,546号に対し、合衆国法典第35巻、第119条(e)の下で利益を主張する、2010年11月10日に出願された「Analyzing Data Using A Hierarchical Structure」と題する米国出願番号第12/943,551号に基づく優先権を主張する。

【背景技術】

【0002】

複雑なパターン認識は、従来のフォンノイマン型コンピュータで実行するには不十分である場合がある。しかし、生物学的な脳、特に人間の脳は、パターン認識の実行に長けている。現在の研究は、人間の脳が新皮質の一連の階層的に編成された神経細胞層を使用してパターン認識を実行していることを示唆している。階層の下層の神経細胞は、たとえば感覚器官からの「未処理信号」を分析する。一方、上位層の神経細胞は下位レベルの神経細胞からの信号出力を分析する。新皮質のこの階層システムが、おそらくは脳の他の領域と組み合わせられて、人間が空間推論、意識的思考、および複雑な言語等の高レベルの機能を実行できるようにする複雑なパターン認識を達成している。

【図面の簡単な説明】

【0003】

【図1】本発明の多様な実施形態に係る並行処理機械の例を示す図である。

【図2】本発明の多様な実施形態に係る有限状態機械の例を示す図である。

10

20

30

40

50

【図 3】本発明の多様な実施形態に従って並行処理機械で実装される 2 レベル階層の例を示す図である。

【図 4】本発明の多様な実施形態に従って並行処理機械で実装される 2 レベル階層の別の例を示す図である。

【図 5】本発明の多様な実施形態に従って並行処理機械で実装される 4 レベル階層の例を示す図である。

【図 6】本発明の多様な実施形態に従って並行処理機械で実装されるフィードバックを有する 4 レベル階層の例を示す図である。

【図 7】本発明の多様な実施形態に従って並行処理機械で実装されるフィードバックを有する 4 レベル階層の別の例を示す図である。

【図 8】本発明の多様な実施形態に従って有限状態機械エンジンとして実装される、図 1 の並行処理機械の例を示す図である。

【図 9】本発明の多様な実施形態に係る図 8 の有限状態機械エンジンのブロックの例を示す図である。

【図 10】本発明の多様な実施形態に係る図 9 のブロックの行の例を示す図である。

【図 11】本発明の多様な実施形態に係る図 10 の行の 2 のグループの例を示す図である。

【図 12】コンパイラが、本発明の多様な実施形態に従って図 8 の並行処理機械のプログラミングのための画像にソースコードを変換する方法の例を示す図である。

【図 13】本発明の多様な実施形態に係るフォンノイマンをベースにしたアーキテクチャを有するコンピュータの例を示す図である。

【発明を実施するための形態】

【0004】

以下の説明および図面は特定の実施形態を十分に示し、当業者がそれらを実行できるようにする。他の実施形態は、構造上の変更、論理の変更、電気的変更、プロセス変更、および他の変更を組み込んでよい。いくつかの実施形態の部分および特長は、他の実施形態の部分および特長の中に含まれてよい、または他の実施形態の部分および特長に代わってよい。

【0005】

本書は、とりわけ、階層構造を使用してデータを分析するための方法および装置を説明する。階層構造は複数の層を含むことがあり、各層は、入力データに対する分析を実行し、その分析に基づいた出力を提供する。階層構造の下層からの出力は、上位層に対する入力として提供できる。このようにして、下層は下位レベルの分析（たとえば、基本的 / 根本的な分析）を実行できる。一方、上位層は、1 つまたは複数の下層からの出力を使用して高レベルの分析（たとえば、より複雑な分析）を実行できる。例では、階層構造は、パターン認識を実行する。例では、パターン認識は、記号のシーケンスを識別することを含む。パターンの識別のための例の記号は、音素（音声）、画像中のピクセル、ASCII 文字、機械データ（たとえば 0 と 1）に相当することがある。

【0006】

例では、階層構造は、カスケード式でともに結合された複数の並行処理機械で実装される。たとえば、第 1 の並行処理機械および第 2 の並行処理機械は、第 2 の並行処理機械が、第 1 の並行処理機械からの出力を入力として受け取るように、直列で結合できる。任意の数の並行処理機械が、この階層構造でともに結合できる。

【0007】

階層構造を使用してデータを分析することに加えて、本書は、階層のあるレベルで実行される分析からの情報を使用して、階層の別のレベルで実行される分析を修正するための方法および装置も説明する。高レベルの分析を実現する第 2 の並行処理機械は、上述された並行処理機械の例を使用して、下位レベルの分析を実現する第 1 の並行処理機械にフィードバック情報を提供することができる。フィードバック情報は、第 1 の並行処理機械によって使用され、生物学的な脳での学習に類似した方法で第 1 の並行処理機械によって実

10

20

30

40

50

行される分析を更新できる。

【 0 0 0 8 】

図 1 は、データを分析するための階層構造を実装するために使用できる例の並行処理機械 1 0 0 を示す。並行処理機械 1 0 0 は、入力データを受け取り、その入力データに基づいた出力を提供することができる。並行処理機械 1 0 0 は、入力データを受け取るためのデータ入力ポート 1 1 0、および別の装置に出力を提供するための出力ポート 1 1 4 を含むことがある。データ入力ポート 1 1 0 は、並行処理機械 1 0 0 に入力されるためのデータにインタフェースを提供する。

【 0 0 0 9 】

並行処理機械 1 0 0 は、それぞれが 1 つまたは複数の入力 1 0 4、および 1 つまたは複数の出力 1 0 6 を有する複数のプログラマブル要素 1 0 2 を含む。プログラマブル要素 1 0 2 は、複数の状態の内の 1 つにプログラムすることができる。プログラマブル要素 1 0 2 の状態は、プログラマブル要素 1 0 2 が、所与の入力（複数の場合がある）に基づいてどの出力（複数の場合がある）を提供するのかを決定する。すなわち、プログラマブル要素 1 0 2 の状態は、所与の入力に基づいて、プログラマブル要素がどのように反応するのかを決定する。データ入力ポート 1 1 0 に対するデータ入力は、複数のプログラマブル要素 1 0 2 に提供され、プログラマブル要素 1 0 2 にそれに対して処置を講じさせることができる。プログラマブル要素 1 0 2 の例は、以下に説明される状態機械要素（SME）、および構成可能論理ブロックを含むことがある。例では、SME は、所与の入力がデータ入力ポート 1 1 0 で受け取られるときに、特定の出力（たとえば、高、つまり「1」信号）を提供するように所与の状態で設定できる。所与の入力以外の入力がデータ入力ポート 1 1 0 で受け取られるとき、SME は異なる出力（たとえば、低、つまり「0」信号）を提供できる。例では、構成可能な論理ブロックは、データ入力ポート 1 1 0 で受け取られる入力に基づいてブール論理関数（たとえば、AND、OR、NOR、ext.）を実行するように設定できる。

【 0 0 1 0 】

並行処理機械 1 0 0 は、並行処理機械 1 0 0 の上にプログラム（たとえば、画像）をロードするためのプログラミングインタフェース 1 1 1 も含むことがある。画像は、プログラマブル要素 1 0 2 の状態をプログラム（たとえば、設定）できる。つまり、画像は、所与の入力に特定のやり方で反応するようにプログラマブル要素 1 0 2 を構成できる。たとえば、プログラマブル要素 1 0 2 は、文字「a」がデータ入力ポート 1 1 0 で受け取られるときに高信号を出力するように設定できる。いくつかの例では、並行処理機械 1 0 0 は、プログラマブル要素 1 0 2 の動作のタイミングを制御するためのクロック信号を使用できる。特定の例では、並行処理機械 1 0 0 は、プログラマブル要素 1 0 2 とのインタフェースをとるため、および特殊目的の機能を実行するための特殊目的要素 1 1 2（たとえば、RAM、論理ゲート、カウンタ、ルックアップテーブル等）を含むことがある。いくつかの実施形態では、データ入力ポート 1 1 0 で受け取られるデータは、経時的にまたは一度に受け取られるデータの固定セット、もしくは経時的に受け取られるデータのストリームを含むことがある。データは、並行処理機械 1 0 0 に結合される、データベース、センサ、ネットワーク等の任意のソースから受け取られてよい、または任意のソースによって生成されてよい。

【 0 0 1 1 】

並行処理機械 1 0 0 は、並行処理機械 1 0 0 の異なる要素（たとえば、プログラマブル要素 1 0 2、データ入力ポート 1 1 0、出力ポート 1 1 4、プログラミングインタフェース 1 1 1、および特殊目的要素 1 1 2）をともに選択的に結合するための複数のプログラマブルスイッチ 1 0 8 も含む。したがって、並行処理機械 1 0 0 は、要素の間で形成されるプログラマブルマトリックスを含む。例では、プログラマブルスイッチ 1 0 8 は、プログラマブル要素 1 0 2、データ入力ポート 1 1 0、プログラミングインタフェース 1 1 1、または特殊目的要素 1 1 2 の入力 1 0 4 が、1 つまたは複数のプログラマブルスイッチ 1 0 8 を通して、プログラマブル要素 1 0 2、出力ポート 1 1 4、プログラミングインタ

10

20

30

40

50



フェース 1 1 1、または特殊目的要素 1 1 2 の出力 1 0 6 に結合できるように、2 つ以上の要素を選択的に互いに結合できる。したがって、要素間の信号の経路選択は、プログラマブルスイッチ 1 0 8 を設定することによって制御できる。図 1 は、所与の要素とプログラマブルスイッチ 1 0 8 との間の特定数の導線（たとえばワイヤ）を示しているが、他の例では、異なる数の導線を使用できることが理解されるべきである。また、図 1 は、プログラマブルスイッチ 1 0 8 に個々に結合される各プログラマブル要素 1 0 2 を示しているが、他の例では、複数のプログラマブル要素 1 0 2 が、プログラマブルスイッチ 1 0 8 に対するグループ（たとえば、図 8 に示されるようなブロック 8 0 2）として結合できる。例では、データ入力ポート 1 1 0、データ出力ポート 1 1 4、および/またはプログラミングインタフェース 1 1 1 は、レジスタへの書込みがそれぞれの要素へデータを提供する、またはそれぞれの要素からデータを提供するようにレジスタとして実装できる。

10

#### 【 0 0 1 2 】

例では、単一の並行処理機械 1 0 0 が、物理デバイス上に実装されているが、他の例では、2 台以上の並行処理機械 1 0 0 を単一の物理デバイス（たとえば、物理チップ）上に実装することができる。例では、複数の並行処理装置 1 0 0 のそれぞれが、別個のデータ入力ポート 1 1 0、別個の出力ポート 1 1 4、別個のプログラミングインタフェース 1 1 1、プログラマブル要素 1 0 2 の別個のセットを含むことがある。さらに、プログラマブル要素 1 0 2 の各セットは、その対応する入力データポート 1 1 0 でデータに反応する（たとえば、高信号または低信号を出力する）ことができる。たとえば、第 1 の並行処理機械 1 0 0 に対応するプログラマブル要素 1 0 2 の第 1 のセットは、第 1 の並行処理機械 1 0 0 に対応する第 1 のデータ入力ポート 1 1 0 でデータに反応できる。第 2 の並行処理機械 1 0 0 に対応するプログラマブル要素 1 0 2 の第 2 のセットは、第 2 の並行処理機械 1 0 0 に対応する第 2 のデータ入力ポート 1 1 0 に反応できる。したがって、各並行処理機械 1 0 0 は、プログラマブル要素 1 0 2 のセットを含み、プログラマブル要素 1 0 2 の異なるセットが、異なる入力データに反応できる。同様に、各並行処理機械 1 0 0、およびプログラマブル要素 1 0 2 の対応する各セットは、別個の出力を提供できる。いくつかの例では、第 1 の並行処理機械 1 0 0 からの出力ポート 1 1 4 は、第 2 の並行処理機械 1 0 0 の入力ポート 1 1 0 に結合でき、したがって、第 2 の並行処理機械 1 0 0 のための入力データは、第 1 の並行処理機械 1 0 0 からの出力データを含むことができる。

20

#### 【 0 0 1 3 】

例では、並行処理機械 1 0 0 にロードするための画像は、プログラマブル要素 1 0 2 の状態を設定し、プログラマブルスイッチ 1 0 8 をプログラムし、並行処理機械 1 0 0 内部で特殊目的要素 1 1 2 を構成するための複数のビットの情報を含む。例では、画像は、特定の入力に基づいて所望される出力を提供するように並行処理機械 1 0 0 をプログラムするために、並行処理機械 1 0 0 にロードできる。出力ポート 1 1 4 は、データ入力ポート 1 1 0 でのデータに対するプログラマブル要素 1 0 2 の反応に基づいて並行処理機械 1 0 0 から出力を提供できる。出力ポート 1 1 4 からの出力は、所与のパターンの一致を示す単一ビット、複数のパターンに対する一致および不一致を示す複数ビットを含むワード、および所与の瞬間でのすべてのまたは特定のプログラマブル要素 1 0 2 の状態に対応する状態ベクトルを含むことがある。

30

40

#### 【 0 0 1 4 】

並行処理機械 1 0 0 の例の使用は、パターン認識（たとえば、音声認識、画像認識等）信号処理、画像処理、コンピュータビジョン、暗号法および他を含む。特定の例では、並行処理機械 1 0 0 は、有限状態機械（F S M）エンジン、フィールドプログラマブルゲートアレイ（F P G A）、およびその変形を含むことがある。さらに、並行処理機械 1 0 0 は、コンピュータ、ページャ、セルラー電話、個人用整理手帳、携帯型オーディオプレーヤー、ネットワーク装置（たとえば、ルータ、ファイアウォール、スイッチ、または任意のその組み合わせ）、制御回路、カメラ等のより大きな装置内の構成要素であってよい。

#### 【 0 0 1 5 】

図 2 は、並行処理機械 1 0 0 によって実装できる有限状態機械（F S M）の例のモデル

50

を示す。並行処理機械 100 は、FSM の物理的な実装として構成（たとえば、プログラム）できる。FSM は、1 つまたは複数のルートノード 202 を含む、グラフ 200（たとえば、有向グラフ、無向グラフ、準グラフ）として表現できる。ルートノード 202 に加えて、FSM は、いくつかの標準ノード 204、ならびに 1 つまたは複数のエッジ 206 を通してルートノード 202 および他の標準ノード 204 に接続される端末ノード 208 から構成できる。ノード 202、204、208 は、FSM 内の状態に相当する。エッジ 206 は、状態間の遷移に相当する。

#### 【0016】

ノード（202、204、208）のそれぞれは、アクティブ状態または非アクティブ状態のどちらかであることがある。非アクティブ状態にあるとき、ノード（202、204、208）は、入力データに反応（たとえば、応答）しない。アクティブ状態にあるとき、ノード（202、204、208）は、入力データに反応できる。上流ノード（202、204）は、入力データが、上流ノード（202、204）と下流ノード（204、208）との間でエッジ 206 によって指定される基準に一致するときに、そのノードから下流にあるノード（204、208）を活性化することによって入力ノードに反応できる。たとえば、文字「b」を指定する第 1 のノード 204 は、第 1 のノード 204 がアクティブで、文字「b」が入力データとして受け取られるときに、エッジ 206 によって第 1 のノード 204 に接続される第 2 のノード 204 を活性化する。本明細書に使用されるように、「上流」は、1 つまたは複数の他のノードの上流（またはループの場合にはそれ自体の上流）にある第 1 のノードが、第 1 のノードが 1 つまたは複数の他のノードを活性化できる（またはループの場合にはそれ自体を活性化できる）状態を指す、1 つまたは複数のノード間の関係性を指す。同様に、「下流」は、1 つまたは複数の他のノードの下流（ループの場合にはそれ自体の下流）にある第 1 のノードが、1 つまたは複数の他のノードによって活性化できる（またはループの場合にはそれ自体によって活性化できる）関係性を指す。したがって、用語「上流」および「下流」は、本明細書中で、1 つまたは複数のノード間の関係性を指すために使用されるが、これらの用語は、ノード間のループまたは他の非線形経路の使用を排除しない。

#### 【0017】

グラフ 200 では、ルートノード 202 は初期に活性化され、入力データが、ルートノード 202 からのエッジ 206 に一致するときに下流ノード 204、208 を活性化できる。グラフ 200 全体でノード 204、208 は、入力データが受け取られると、このようにして活性化できる。端末ノード 208 は、入力データによる関心のあるシーケンスの一致に相当する。したがって、端末ノード 208 の活性化は、入力データとして関心のあるシーケンスが受け取られたことを示す。パターン認識機能を実装する並行処理機械 100 との関連で、端末ノード 208 に到達することは、関心のある特定のパターンが、入力データで検出されたことを示すことができる。

#### 【0018】

例では、それぞれのルートノード 202、標準ノード 204、および端末ノード 208 は、並行処理機械 100 のプログラマブル要素 102 に相当することがある。各エッジ 206 は、プログラマブル要素 102 間の接続に相当することがある。したがって、別の標準ノード 204 または端末ノード 208 に遷移する（たとえば、別の標準ノード 204 または端末ノード 208 に接続するエッジ 206 を有する）標準ノード 204 は、別のプログラマブル要素 102 に遷移する（たとえば、別のプログラマブル要素 102 に出力を提供する）プログラマブル要素 102 に相当する。いくつかの例では、ルートノード 202 は、対応するプログラム要素 102 を有していない。

#### 【0019】

並行処理機械 100 が FSM としてプログラムされるとき、プログラマブル要素 102 のそれぞれも、アクティブ状態または非アクティブ状態のどちらかであることがある。非アクティブであるときの所与のプログラマブル要素 102 は、その対応するデータ入力ポート 110 で入力データに反応しない。アクティブプログラマブル要素 102 は、入力

データおよびデータ入力ポート 110 に反応することができ、入力データがプログラマブル要素 102 の設定に一致するときに下流のプログラマブル要素 102 を活性化できる。プログラマブル要素 102 が、端末ノード 208 に相当するとき、プログラマブル要素 102 は、出力ポート 114 に結合され、外部装置に一致の表示を提供できる。

#### 【0020】

プログラミングインタフェース 111 を介して並行処理機械 100 にロードされる画像は、所望される FSM がデータ入力ポート 110 でのデータの反応に基づいてノードの順次的な活性化によって実装されるように、プログラマブル要素 102 と他の要素 112 との間の接続だけではなく、プログラマブル要素 102 および他の要素 112 も構成できる。例では、プログラマブル要素 102 は、単一のデータサイクル（たとえば、単一文字、文字のセット、単クロックサイクル）の間活性のままとなり、次いで上流のプログラマブル要素 102 によって再活性化されない限り、非アクティブに切り替わる。

10

#### 【0021】

端末ノード 208 は、過去のイベントの圧縮履歴を記憶するために検討できる。たとえば、端末ノード 208 に達するために必要とされる入力データの 1 つまたは複数のパターンは、その端末ノード 208 の活性化によって表すことができる。例では、端末ノード 208 によって提供される出力はバイナリである。すなわち、出力は、関心のあるパターンが一致しているかどうかを示す。グラフ 200 中の端末ノード 208 の標準ノード 204 に対する率はきわめて小さい場合がある。言い換えると、FSM に高い複雑度がある場合があるが、FSM の出力は、比較すると小さい場合がある。

20

#### 【0022】

例では、並行処理機械 100 の出力は、並行処理機械用の状態ベクトルを含むことがある。状態ベクトルは並行処理機械 100 のプログラマブル要素 102 の状態（たとえば、活性化済みまたは未活性）を含む。例では、状態ベクトルは、端末ノード 208 に相当するプログラマブル要素 102 のための状態を含む。したがって、出力は、グラフ 200 のすべての端末ノード 208 によって提供される表示の集合体を含むことがある。状態ベクトルはワードとして表すことができ、各端末ノード 208 によって提供されるバイナリ表示は、1 ビットのワードを含む。端末ノード 208 のこの符号化は、並行処理機械 100 のための検出状態（たとえば、関心のあるシーケンスが検出されたかどうか、および関心のあるどのシーケンスが検出されたのか）の効果的な表示を提供できる。別の例では、状態ベクトルは、プログラマブル要素 102 が端末ノード 208 に相当するかどうかに関係なく、プログラマブル要素 102 のすべてまたは部分集合の状態を含むことがある。

30

#### 【0023】

上述されたように、並行処理機械 100 は、パターン認識機能を実装するようにプログラムできる。たとえば、並行処理機械 100 によって実装される FSM は、入力データ中の 1 つまたは複数のデータシーケンス（たとえば、シグナチャ、パターン）を認識するように構成できる。関心のあるデータシーケンスが並行処理機械 100 によって認識されるとき、その認識の表示は、出力ポート 114 で提供できる。例では、パターン認識は、たとえば、ネットワークデータ中のマルウェアまたは他の情報を識別するために記号（たとえば、ASCII 文字）の文字列を認識できる。

40

#### 【0024】

図 3 は、階層構造 300 を使用してデータを分析するように構成された第 1 の並行処理機械 302 および第 2 の並行処理機械 304 の例を示す。それぞれの並行処理機械 302、304 は、データ入力ポート 302 A、304 A、プログラミングインタフェース 302 B、304 B、および出力ポート 302 C、304 C を含む。

#### 【0025】

第 1 の並行処理機械 302 は、入力データ、たとえばデータ入力ポート 302 A での未処理データを受け取るように構成される。第 1 の並行処理機械 302 は、上述されたように入力データに応答し、出力ポート 302 C で出力を提供する。第 1 の並行処理機械 302 からの出力は、第 2 の並行処理機械 304 のデータ入力ポート 304 A に送信される。

50

次いで、第2の並行処理機械304は、第1の並行処理機械302によって提供される出力に基づいて反応し、出力ポート304Cで対応する出力を提供できる。2つの並行処理機械302、304の直列でのこの階層上の結合は、第1の並行処理機械302から第2の並行処理機械304に圧縮ワードで過去のイベントに関する情報を転送するための手段を提供する。転送される情報は、事実上、第1の並行処理機械302によって記録された複雑なイベントの要約（たとえば、関心のあるシーケンス）である場合がある。

#### 【0026】

図3に示される並行処理機械302、304の2レベル階層300は、2つの独立したプログラムが、同じデータストリームに基づいて動作できるようにする。2段階階層は、異なる部位としてモデル化される生物学的な脳内での視覚的な認識に類似する場合がある。このモデルの下では、部位は、事実上、異なるパターン認識エンジンであり、それぞれが類似する計算機能（パターン一致）を実行するが、他のプログラム（シグナチャ）を使用する。複数の並行処理機械302、304をともに接続することによって、データストリーム入力について高められた知識が取得され得る。

#### 【0027】

（第1の並行処理機械302によって実装される）階層の第1のレベルは、未処理データストリームに対して直接的に処理を実行する。すなわち、未処理データストリームは入力インタフェース302Aで受け取られ、第1の並行処理機械302のプログラマブル要素が未処理データストリームに反応できる。階層の（第2の並行処理機械304によって実装される）第2のレベルは、第1のレベルからの出力を処理する。すなわち、第2の並行処理機械304は、入力インタフェース304Bで第1の並行処理機械302から出力を受け取り、第2の並行処理機械304のプログラマブル要素は第1の並行処理機械302の出力に反応できる。したがって、この例では、第2の並行処理機械304が入力として未処理データストリームを受け取るのではなく、むしろ、第1の並行処理機械302によって決定される未処理データストリームが一致する関心のあるパターンの表示を受け取る。第2の並行処理機械304は、第1の並行処理機械302からの出力データストリーム中のパターンを認識するFSMでプログラムできる。

#### 【0028】

図4は、階層の1つのレベルが複数の並行処理機械で実装される2レベル階層400の別の例を示す。ここでは、階層400の第1のレベルは、3つの並行処理機械402で実装される。3つの第1のレベルの並行処理機械402のそれぞれからの出力が、第1のレベルの並行処理機械402からの出力でパターンを認識する（たとえば、識別する）単一の第2のレベルの並行処理機械404に提供される。他の例では、異なる数の並行処理機械が、異なるレベルで実装できる。各並行処理機械402、404は、データ入力ポート402A、404、プログラミングインタフェース402B、404B、および出力ポート402C、404Cを含む。

#### 【0029】

図5は、4台の並行処理機械502、504、506、および508で実装され、各レベルによって識別されるパターンの例を示す4レベル階層500を示す。上述されたように、各並行処理機械502、504、506および508は、データ入力ポート502A、504A、506A、および508A、プログラミングインタフェース502B、504B、506B、および508B、ならびに出力ポート502C、504C、506C、および508Cを含む。4レベル階層500は、画像中の黒のピクセルまたは白のピクセルに基づいた作成された言語の視覚的な識別に相当する。階層が高レベルに進むにつれ、入力ストリームの蓄積された知識は相応して増大する。並行処理機械502、504、506、508は、階層認識機能を達成するためにカスケードされる。階層500のそれぞれの連続レベルは、従前のレベルの圧縮された出力に適用される新しい規則（パターンシグナチャ）を実装できる。このようにして、きわめて詳細なオブジェクトが、基本的なプリミティブな情報の初期検出に基づいて識別できる。

#### 【0030】

たとえば、レベル1（第1の並行処理機械502）に対する未処理データ入力ストリームは、視角映像のための情報（たとえば、所与のビットが黒なのか／白なのか、ONか／OFFか）を含むことがある。第1の並行処理機械502は、ピクセル情報によって形成されるプリミティブパターンを識別するようにプログラムできる。たとえば、第1の並行処理機械502は、隣接するピクセルが垂直線、水平線、弧等をいつ形成するのかを識別するように構成できる。これらのパターン（たとえば、垂直線、水平線弧等）のそれぞれは、第1の並行処理機械502からのそれぞれの出力ビット（または信号）によって示すことができる。たとえば、第1の並行処理機械502が少なくとも3ビットの垂直線を識別するとき、高信号（たとえば、論理「1」）が、第2の並行処理機械504に対する出力ワードの第1のビットで出力できる。第1の並行処理機械502が少なくとも3ビットの水平線を識別するとき、高信号が、第2の並行処理機械504に対する初期ワードの第2のビットで出力できる。

10

#### 【0031】

第2のレベル（第2の並行処理機械504）は、第1の並行処理機械502からの出力信号のパターンを識別するようにプログラムできる。たとえば、第2の並行処理機械504は、第1の並行処理機械502によって識別されるプリミティブパターン（線、弧、他）の組合せによって形成されるパターンを識別するようにプログラムできる。第2の並行処理機械504は、水平線および垂直線がいつ交差し、文字「t」を形成するのかを識別するようにプログラムできる。上述されたように、第2の並行処理機械504によって実装されるFSM内のノードは、第1の並行処理機械502からの出力に反応する。したがって、プリミティブパターンの組合せは、第1の並行処理機械502からの出力ビット中のパターンを識別することによって識別される。

20

#### 【0032】

次いで、第2の並行処理機械504からの出力が、第2の並行処理機械506によって識別される文字の組合せからワードを識別できる第3のレベル（第3の並行処理機械506）に入力される。第4のレベル（第4の並行処理機械508）は、次いで、第3の並行処理機械506によって識別されるワードによって形成されるフレーズを識別できる。したがって、高レベルが、下位レベルの出力のパターンを識別するようにプログラムできる。さらに、下位レベルが、高レベルで識別されるパターンを構成する構成要素を識別するようにプログラムできる。

30

#### 【0033】

ピクセル情報からの文字、ワードおよびフレーズの視覚的な識別は、例として使用される。ただし、本明細書に説明される階層方法および装置は、他のデータに、および他の用途のために適用できる。たとえば、階層分析は、音声に対応するデータで使用され、第1のレベルで音素の組合せから音節を、第2のレベルで音節の組合せからワードを識別できる。他の例では、階層分析は、それ自体を階層状に構築する機械データ（たとえば、未処理の0および1）に適用できる。

#### 【0034】

図5は、層間での特定の接続および個々の接続を示しているが、1つのレベルからの出力が階層の他のレベルに前方にまたは後方に送られる階層が、実装できることが理解されるべきである。たとえば、第2の並行処理機械504からの出力は、第4の並行処理機械508に送信できるだろう。一方、第4の並行処理機械508からの出力は、第3の並行処理機械506にフィードバックされる可能性がある。一般的には、階層は、並行処理機械からの検出状態情報が、他の並行処理機械の1台または複数台もしくはすべてに送られるように実装できる。

40

#### 【0035】

いくつかの例では、フィードバックは、階層構造内で使用され、1つまたは複数のレベルで使用されるプログラムを更新する。たとえば、第1のレベルからの出力は、第2のレベルに提供され、第2のレベルを再プログラムできる。これは、第1のレベルで識別される（または識別されない）パターンに基づいて第2のレベルによって適用される規則を更

50

新するために使用できる。例では、第1のレベルは、第2のレベルよりも階層では高レベルである。たとえば、下位レベルは、高レベルによって識別されるパターンに基づいてプログラムによって最初に指定されなかった追加のパターンを探すように再プログラムできる。別の例では、下位レベルは、下位レベルによって識別される特定のパターンが、特定のパターンが他のパターンと結合し、重要なイベントを形成するという点で重要であることを通知され得る。さらに別の例では、下位レベルは、識別された特定のパターンが特定の値を有さず、したがって下位レベルがそのパターンを識別するのを停止できると通知されてよい。例では、再プログラミングは経時的に実行でき、したがって所与のレベルのプログラムは、一定期間での小さな調整によって増分的に修正される。

#### 【0036】

図6は、フィードバックを使用して階層の部分を再プログラムする4レベル階層600の例を示す。4レベル階層600は、それぞれが、データ入力ポート602A、604A、606A、608A、プログラミングインタフェース602B、604B、606B、608B、および出力ポート602C、604C、606C、608Cを有する4つの並行処理機械602、604、606、608で実装される。第1の並行処理機械602は、階層600の第1のレベルを実装し、階層600の第2のレベルを実装する第2の並行処理機械604に出力を提供する。第3の並行処理機械および第4の並行処理機械606、608は、同様に、階層600の第3のレベルおよび第4のレベルを実装する。例では、第4の並行処理機械608からの出力が、第1の並行処理機械602によって受け取られる入力データに関する階層600の分析に基づいた階層600の出力として外部装置に送られる。したがって、第4の並行処理機械608からの出力は、階層600のための集成的な出力に相当する。他の例では、他の並行処理機械608からの出力は、階層600のための集成的な出力に相当することがある。

#### 【0037】

第2の、第3のおよび第4の並行処理機械604、606、608は、それぞれ下のレベルで並行処理機械602、604、606のプログラミングインタフェース602B、604B、606Bにフィードバックされる。たとえば、第4の並行処理機械608からの出力は、第3の並行処理機械606のプログラミングインタフェース606Bにフィードバックされる。したがって、第3の並行処理機械606は、第4の並行処理機械608からの出力に基づいて再プログラムできる。相応して、第3の並行処理機械606は、実行時間にそのプログラムを修正できる。第1の並行処理機械および第2の並行処理機械602、604は、それぞれ第2のおよび第3の並行処理機械604、606からの出力に基づいて実行時間に同様に再プログラムできる。

#### 【0038】

例では、並行処理機械604、606、608が分析され、コンパイルされて、並行処理機械602、604、606を再プログラムするためのプログラム（たとえば、画像）を形成する。たとえば、並行処理機械608からの出力は、プログラミングインタフェース606Bに送られる前に処理装置614によって分析され、コンパイルされる。処理装置614は、並行処理機械608からの出力に基づいて並行処理機械606のための更新プログラムを生成できる。処理装置614は、出力を分析し、第3の並行処理装置606のための更新されたプログラムをコンパイルできる。更新されたプログラムは、次いで、プログラミングインタフェース606Bを通して第3の並行処理機械606にロードされ、第3の並行処理機械606を再プログラムできる。例では、更新されたプログラムは、現在のプログラムからの部分的な変更だけを含んでよい。したがって、例では、更新されたプログラムは、並行処理機械602、604、606、608上で現在のプログラムの一部だけを置換できる。別の例では、更新されたプログラムは現在のプログラムのすべてまたはより大部分を置換する。同様に、処理装置610、612は、第2のおよび第3の並行処理機械604、606からの出力に基づいて、同様にフィードバックを分析し、更新されたプログラムをコンパイルできる。処理装置610、612、614は、1つまたは複数の追加の並行処理機械で実装できる、または異なるタイプの機械（たとえば、フォ

10

20

30

40

50

ンノイマンアーキテクチャを有するコンピュータ)で実装できる。

【0039】

いくつかの例では、処理装置610、612、614は、新しいプログラムをコンパイルする前に高レベルからの出力を分析する。例では、処理装置610、612、614は、出力を分析して、下位レベルのプログラムを更新する方法を決定して、次いで新しい(たとえば、更新された、最初の)下位レベルのプログラムを分析に基づいてコンパイルする。階層600では、所与の並行処理機械でのフィードバックは、所与の並行処理機械のすぐ上のレベルから受け取られるが、フィードバックは、任意の並行処理機械から高レベル、下位レベル、または同レベルで別の並行処理機械になる場合がある。たとえば、フィードバックは、その同じ並行処理機械の出力から、または同じレベル、高レベル、または下位レベルでの別の並行処理機械の出力から、並行処理機械のプログラミング入力を受け取ることができる。さらに、並行処理機械は、複数の異なる並行処理機械からフィードバックを受け取ることができる。フィードバックに基づいた並行処理機械の再プログラミングは、入力データのパターンの識別から時間内に(たとえば、未処理データの処理とリアルタイムではない)切断されてよい。

10

【0040】

情報を階層下方に送り返し、下位レベルの再プログラミングに影響を及ぼす目的は、下位レベルが関心のあるパターンの差異を認識することにより効率的になるためである場合がある。情報を下方に送る別の目的は、下位レベルで高レベルの鋭さを達成することである。いくつかの例では、階層の高レベルに情報を転送するには時間を要することを認識し、情報を高レベルに送るプロセスは、可能な時には回避される。いくつかの例では、高レベルは、本来、システムにとって新しいパターンの識別を解決するために使用できる。これは、生物学的な脳のニューロコル皮質で起こる、使用されているプロセスに類似することがある。例では、パターンが下位レベルで完全に解決できるのであれば、パターンは解決されるべきである。フィードバック機構は、階層の下位レベルに「学習」を転送する1つの方法である。階層を下方に情報を押し下げるこのプロセスは、新しいパターンまたは見慣れていないパターンを処理するために階層の上位レベルを保存するのに役立つ。さらに、認識プロセス全体は、階層の多様なレベルを通じたデータ転送の量を削減することによって加速できる。

20

【0041】

フィードバックは、階層の下位レベルを、入力時にデータストリームに対してより鋭く敏感にすることができる。この情報の「押し下げ」の結果として、決定は、階層の下位レベルで下すことができ、非常に迅速に下すことができる。したがって、例では、下位レベルの並行処理機械(たとえば、第1の並行処理機械602)からの出力は、第4の並行処理機械608からの出力とともに、階層600から別の装置への集合的な出力に相当することがある。外部装置は、たとえば、これらの並行処理機械602、608のそれぞれからの出力を監視して、いつ階層600によってパターンが識別されたかを決定できる。

30

【0042】

例では、フィードバック情報は、分析されるデータストリームに対応する識別情報を含む場合がある。たとえば、識別情報は、データの識別特性、データのフォーマット、データのプロトコル、および/または任意の他のタイプの識別情報を含むことがある。識別情報は、たとえば、処理装置610による入力データののための分析方法を修正する(たとえば、適応する)ために収集され、分析され、使用されてよい。並行処理機械100は、次いで適応された分析方法でプログラムされてよい。識別情報は、たとえば、入力データの言語を含むことがある。並行処理機械100は、初期に、入力データの言語を決定するためにプログラムすることができ、いったん言語が入力に対応して識別されると、実行時の間に適応され(たとえば、再プログラムされ)てよい。並行処理機械100のための適応された分析方法は、より具体的には、識別された言語に対する分析方法に相当することがある。最後に、並行処理機械100は、適応された分析方法を使用してさらに入力データを分析してよい。フィードバックプロセスは、反復的であってよく、したがって追加の識

40

50

別情報が入力データで検出され、分析方法のさらなる適応化を可能にする。

【 0 0 4 3 】

並行処理機械 1 0 0 にロードするためのプログラム（たとえば、画像）は、図 1 2 に関して以下に説明されるコンパイラによって生成できる。一般に、コンパイルは、計算集約的なプロセスである場合があり、初めてパターンシグナチャの大型データベースをコンパイルする際に最も明白になることがある。実行時動作では、高レベルの並行処理機械 1 0 0 が、下位レベルの並行処理機械のための増分プログラム更新の形で下位レベルにフィードバックを提供できる。したがって、下位レベルの並行処理機械に対するフィードバック情報は、コンパイルするにはより計算集約的ではない元のプログラムに対するはるかに小さい、増分更新である場合がある。

10

【 0 0 4 4 】

図 7 は、4 つの並行処理機械 7 0 2、7 0 4、7 0 6、7 0 8 で実装される 4 レベル階層 7 0 0 の別の例を示す。それぞれがデータ入力ポート 7 0 2 A、7 0 4 A、7 0 6 A、7 0 8 A、プログラミングインタフェース 7 0 2 B、7 0 4 B、7 0 6 B、7 0 8 B、および出力ポート 7 0 2 C、7 0 4 C、7 0 6 C、7 0 8 C を有する 4 つの並行処理機械 7 0 2、7 0 4、7 0 6、7 0 8。さらに、いくつかの例では、4 レベル階層 7 0 0 は、処理装置 7 1 0、7 1 2、7 1 4 を含み、並行処理機械 7 0 2、7 0 4、および 7 0 6 のためのプログラムをコンパイルできる。4 レベル階層 7 0 0 では、第 2 レベルの、第 3 レベルの、および第 4 レベルの並行処理機械 7 0 4、7 0 6、7 0 8 が、未処理データストリームからの入力データだけではなく、下位レベルの並行処理機械 7 0 2、7 0 4、7 0 6 の出力からの入力データも受け取る。したがって、レベル 2、レベル 3、およびレベル 4 は、下位レベルからのパターンおよび未処理データの組合せからパターンを識別できる。

20

【 0 0 4 5 】

図 6 および図 7 からわかるように、並行処理機械 1 0 0 は、並行処理機械 1 0 0 からの出力だけではなく、階層に対する未処理データ入力も、それ自体を含む他の任意の並行処理機械 1 0 0 に送ることができる場合にほぼ任意の方法でカスケードできる。さらに、所与の並行処理機械 1 0 0 からの出力は、データ入力ポート 1 1 0 に対する入力データとして、および / または並行処理機械 1 0 0 のためのプログラムを更新するためのフィードバックとして別の並行処理機械 1 0 0 に送信できる。

【 0 0 4 6 】

30

並行処理機械 1 0 0 が、入力データの 1 つのデータサイクル（たとえば、ビット、ワード）を処理するための時間のため、並行処理機械 1 0 0 を直列でカスケードすることは、すべての並行処理機械 1 0 0 を通して入力データストリームを完全に処理して、階層のための集約的な出力を生成するために要する時間を増加することがある。階層の下位レベルは、下位（最も粒状の）レベルの入力を受け取ることができるので、下位レベルは、高レベルの出力よりもよりアクティブであることが予想されるべきである。つまり、階層の中の各連続レベルは、高レベルのオブジェクトをアセンブルできる。例では、並行処理機械 1 0 0 は、並行処理機械 1 0 0 にどれほど早く入力データを送ることができるのかを制限する最大入力レートを持有する。この入力速度は、単一のデータサイクルとして考えることができる。それぞれの連続データサイクルで、並行処理機械 1 0 0 は、多くの端末ノードを活性化する可能性を持有する。これによって、（特に階層の下位レベルにある）並行処理機械 1 0 0 は、かなりの量の出力データを生成するだろう。たとえば、入力が最下位レベルの並行処理機械 1 0 0 にバイトのストリームとして提供される場合、並行処理機械 1 0 0 が、任意の所与のデータサイクルで、結果情報の複数のバイトを生成することが可能である場合がある。1 バイトの情報が複数のバイトの情報を生成できる場合、次いで並行処理機械 1 0 0 の階層全体は、上方が階層の上方に渡されるように同期される必要がある。いくつかの例では、フィードバックは同期される必要がない。ただし、フィードバックが下位レベルで受け取られるのが早いほど、下位レベルが適応するのが早くなり、分析はより効果的になる。

40

【 0 0 4 7 】

50



例として、(単一の並行処理機械100で実装される)階層のレベルごとの最大サイズ出力は、1024バイトに等しくなることがあり、階層の深さは4レベルに等しくなることがある。並行処理機械100の入力データストリームのデータ転送速度は、128MB/秒に等しくなることがある。以上の条件の場合、階層の各レベルは、7.63マイクロ秒で横断できるだろう。4レベル階層を用いる場合、並行処理機械100のスタック全体の総整定時間は、7.63マイクロ秒の4倍、つまり30.5マイクロ秒となるだろう。30.5マイクロ秒の整定時間では、入力データ周波数が32KB/秒に制限される必要があるようである。

#### 【0048】

とりわけ、これは、並行処理機械100の構成にきわめて左右される。並行処理機械100は、入力データ転送速度対状態機械のサイズをトレードオフするように構成可能である。さらに、並行処理機械に対する入力ワードサイズは、並行処理機械100上にロードされる個々の画像を生成したコンパイラに対して対応する修正が行われる場合、調整できる。

#### 【0049】

例では、上述された階層構造は、フォンノイマンアーキテクチャを有する機械でソフトウェアにより実装できるだろう。したがって、ソフトウェア命令が、プロセッサに、未処理データに対する第1のレベルの分析FSMを実装させるだろう。第1のレベルのFSMからの出力は、次いで、第2のレベルの分析等々を使用してプロセッサによって処理できるだろう。さらに、上述されたフィードバックループは、第1のレベルの分析が、たとえば第2のレベルの分析の出力に基づいて修正されるように実装できるだろう。

#### 【0050】

図8から図11は、本明細書で「FSMエンジン800」と呼ばれている並行処理機械の例を示す。例では、FSMエンジン800は、有限状態機械のハードウェア実装を含む。したがって、FSMエンジン800は、FSM内の複数の状態に相当する、複数の選択的に結合可能なハードウェア要素(たとえば、プログラマブル要素)を実装する。FSM内の状態と同様に、ハードウェア要素は、入力ストリームを分析し、その入力ストリームに基づいて下流のハードウェア要素を活性化できる。

#### 【0051】

FSMエンジン800は、汎用要素および特殊目的要素を含む複数のプログラマブル要素を含む。汎用要素は、多くの異なる機能を実装するようにプログラムできる。これらの汎用要素は、行806(図9および図10に示される)に階層状に編成されるSME804、805(図11に図示)、およびブロック802(図8および図9に図示)を含む。階層状に編成されたSME804、805の間で信号の経路を定めるために、ブロック間スイッチ803(図8および図9に図示)、ブロック内スイッチ808(図9および図10に図示)、ならびに行内スイッチ812(図10に図示)を含む、プログラマブルスイッチの階層が使用される。SME804、805は、FSMエンジン800によって実装されるFSMの状態に相当することがある。SME804、805は、以下に説明されるプログラマブルスイッチを使用することによって結合できる。したがって、FSMは、SME804、805を、状態の機能に対応するようにプログラムすることによって、およびFSM内の状態間の遷移に相当するようにSME804、805をともに選択的に結合することによってFSMエンジン800上で実装できる。

#### 【0052】

図8は、例のFSMエンジン800の全体図を示す。FSMエンジン800は、プログラマブルブロック間スイッチ803とともに選択的に結合できる複数のブロック802を含む。さらに、ブロック802は、信号(たとえば、データ)を受信し、そのデータをブロック802に提供するための入力ブロック809(たとえば、データ入力ポート)に選択的に結合できる。また、ブロック802は、ブロック802からの信号を外部装置(たとえば、別のFSMエンジン800)に提供するための出力ブロック813(たとえば、出力ポート)に選択的に結合できる。FSMエンジン800は、FSMエンジン800の

上にプログラム（たとえば、画像）をロードするためのプログラミングインタフェース 811 も含むことがある。画像は、SME 804、805 の状態をプログラム（たとえば、設定）できる。つまり、画像は、入力ブロック 809 で所与の入力に特定のやり方で反応するように SME 804、805 を構成できる。たとえば、SME 804 は、文字「a」が入力ブロック 809 で受け取られるときに高信号を出力するように設定できる。

#### 【0053】

例では、入力ブロック 809、出力ブロック 813、および／またはプログラミングインタフェース 811 は、レジスタへの書込みがそれぞれの要素ヘデータを提供する、またはそれぞれの要素からデータを提供するようにレジスタとして実装できる。したがって、プログラミングインタフェース 811 に相当するレジスタに記憶される画像からのビットは、SME 804、805 にロードできる。図 8 は、ブロック 802 と、入力ブロック 809 と、出力ブロック 813 と、ブロック間スイッチ 803 との間の特定数の導線（たとえば、ワイヤ、トレース）を示しているが、他の例では、より少ない導線またはより多い導線を使用できることが理解されるべきである。

10

#### 【0054】

図 9 は、ブロック 802 の例を示す。ブロック 802 は、プログラマブルブロック内スイッチ 808 と選択的に結合できる複数の行 806 を含むことがある。さらに、行 806 は、ブロック間スイッチ 803 により、別のブロック 802 内の別の行 806 に選択的に結合できる。例では、バッファ 801 が、ブロック間スイッチ 803 への／ブロック間スイッチ 803 からの信号のタイミングを制御するために含まれる。本明細書で行 806 は、2 のグループ（GOT）810 と呼ばれる要素の組に編成される複数の SME 804、805 を含む。例では、ブロック 802 は、十六（16）の行 806 を含む。

20

#### 【0055】

図 10 は、行 806 の例を示す。GOT 810 は、プログラマブル行内スイッチ 812 によって他の GOT 810、および行 806 内の任意の他の要素 824 に選択的に結合できる。GOT 810 は、ブロック内スイッチ 808 で他の行 806 内の他の GOT 810 に、またはブロック間スイッチ 803 で他のブロック 802 内の他の GOT 810 に結合することもできる。例では、GOT 810 は、第 1 の入力および第 2 の入力 814 および 816、ならびに出力 818 を有する。第 1 の入力 814 は、GOT 810 の第 1 の SME 804 に結合され、第 2 の入力 816 は、GOT 810 の第 2 の SME 804 に結合される。

30

#### 【0056】

例では、行 806 は、第 1 のおよび第 2 の複数の行相互接続導線 820、822 を含む。例では、GOT 810 の入力 814、816 は、1 本または複数の行相互接続導線 820、822 に結合することができ、出力 818 は、1 つの行相互接続導線 820、822 に結合できる。例では、第 1 の複数の行相互接続導線 820 は、行 806 内の各 GOT 810 の各 SME 804 に結合できる。第 2 の複数の行相互接続導線 822 は、行 806 内の各 GOT 810 の 1 つの SME 804 に結合できるが、GOT 810 の他の SME 804 には結合できない。例では、第 2 の複数の行相互接続導線 822 の第 1 の半分が、行 806 内部の SME 804 の第 1 の半分（各 GOT 810 から 1 つの SME 804）に結合でき、第 2 の複数の行相互接続導線 822 の第 2 の半分は行 806 内の SME 804 の第 2 の半分（各 GOT 810 からの他の SME 804）に結合できる。第 2 の複数の行相互接続導線 822 と SME 804、805 との間の限られた接続性が、本明細書では「パリティ」と呼ばれている。例では、行 806 は、カウンタ、プログラマブルブール論理要素、フィールドプログラマブルゲートアレイ（FPGA）、特定用途向け集積回路（ASIC）、プログラマブルプロセッサ（たとえば、マイクロプロセッサ）、および他の要素等の特殊目的要素 824 を含むこともある。

40

#### 【0057】

例では、特殊目的要素 824 は、カウンタ（本明細書では、カウンタ 824 とも呼ばれている）を含む。例では、カウンタ 824 は、12 ビットプログラマブルダウンカウンタ

50

を含む。12ビットプログラマブルカウンタ824は、計数入力、リセット入力、およびゼロカウント出力を有する。計数入力は、アサート時、カウンタ824の値を1、減分する。リセット入力は、アサート時、カウンタ824に、関連するレジスタからの初期値をロードさせる。12ビットカウンタ824の場合、最高12ビットの数が初期値としてロードできる。カウンタ824の値がゼロ(0)に減分されるとき、ゼロカウント出力がアサートされる。カウンタ824は、少なくとも2つのモード、パルスおよび保持を有する。カウンタ824がパルスモードに設定されるとき、ゼロカウント出力は、カウンタ824がゼロに減分するときのクロックサイクル中にアサートされ、次のクロックサイクルでは、ゼロカウント出力はもはやアサートされない。カウンタ824が保持モードに設定されるとき、ゼロカウント出力は、カウンタ824がゼロに減分するときのクロックサイクル中にアサートされ、カウンタ824が、アサートされているリセット入力によってリセットされるまでアサートされた状態のままとなる。例では、特殊目的要素824はブール論理を含む。いくつかの例では、このブール論理は、FSMエンジン800の端末状態SMEから情報を抽出するために使用できる。抽出された情報は、他のFSMエンジン800に状態情報を転送するため、および/またはFSMエンジン800を再プログラムするために使用されるプログラミング情報を転送するため、または別のFSMエンジン800を再プログラムするために使用できる。

#### 【0058】

図11は、GOT810の例を示す。GOT810は、入力814、816を有し、ORゲート830および3対1マルチプレクサ842に結合されるそれらの出力826、828を有する、第1のSME804および第2のSME805を含む。3対1マルチプレクサ842は、GOT810の出力818を、第1のSME804、第2のSME805、またはORゲート830に結合するように設定できる。ORゲート830を使用して、両方の出力826、828を結合し、GOT810の共通の出力818を形成できる。例では、第1のSMEおよび第2のSME804、805は、上述されたパリティを示し、第1のSME804の入力814は、行相互接続導線822のいくつかに結合することができ、第2のSME805の入力816は、他の行相互接続導線822に結合できる。例では、GOT810内の2つのSME804、805は、スイッチ840のどちらかまたは両方を設定することによってそれら自体にカスケードおよびまたはループバックすることができる。SME804、805は、SME804、805の出力826、828を、他のSME804、805の入力814、816に結合することによってカスケードできる。SME804、805は、出力826、828をそれら自体の入力814、816に結合することによってそれら自体にループバックできる。したがって、第1のSME804の出力826は、第1のSME804の入力814および第2のSME805の入力816のどれにも結合できないか、そのうちの1つに結合できるか、または両方に結合できる。

#### 【0059】

例では、状態機械要素804、805は、検出ライン834に並列で結合される、多くの場合、動的ランダムアクセスメモリ(DRAM)で使用されるセル等の複数のメモリセル832を含む。1つのかかるメモリセル832は、高値または低値(たとえば、1または0)のどちらかに対応する状態等のデータ状態に設定できるメモリセルを含む。メモリセル832の出力は、検出ライン834に結合され、メモリセル832に対する入力は、データストリームライン836でのデータに基づいて信号を受信する。例では、データストリームライン836での入力は、メモリセル832の内の1つを選択するために復号される。選択されたメモリセル832は、その記憶されたデータ状態を検出ライン834に対する出力として提供する。たとえば、データ入力ポート809で受信されたデータは、デコーダ(不図示)に提供することができ、デコーダはデータストリームライン836の内の1つを選択できる。例では、デコーダはASCII文字を、256ビットの内の1に変換できる。

#### 【0060】

したがって、メモリセル 8 3 2 は、メモリセル 8 3 2 が高値に設定され、データストリームライン 8 3 6 上のデータがメモリセル 8 3 2 に相当するときに、検出ライン 8 3 4 に高信号を出力する。データストリームライン 8 3 6 上のデータがメモリセル 8 3 2 に相当し、メモリセル 8 3 2 が低値に設定されると、メモリセル 8 3 2 は、検出ライン 8 3 4 に低信号を出力する。検出ライン 8 3 4 上でのメモリセル 8 3 2 からの出力は、検出回路 8 3 8 によって感知される。例では、入力ライン 8 1 4、8 1 6 上の信号は、それぞれの検出回路 8 3 8 をアクティブ状態または非アクティブ状態のどちらかに設定する。非アクティブ状態に設定されると、検出回路 8 3 8 はそれぞれの検出ライン 8 3 4 上の信号に関わりなく、それぞれの出力 8 2 6、8 2 8 上で低信号を出力する。アクティブ状態に設定されると、検出回路 8 3 8 は、それぞれの S M E 8 0 4、8 0 5 のメモリセル 8 3 4 の内の 1 つから高信号が検出されるときに、それぞれの出力ライン 8 2 6、8 2 8 上で高信号を出力する。アクティブ状態にあるとき、検出回路 8 3 8 は、それぞれの S M E 8 0 4、8 0 5 のメモリセル 8 3 4 の内のすべてからの信号が低であるときに、それぞれの出力ライン 8 2 6、8 2 8 上で低信号を出力する。

#### 【 0 0 6 1 】

例では、S M E 8 0 4、8 0 5 は、2 5 6 個のメモリセル 8 3 2 を含み、各メモリセル 8 3 2 は異なるデータストリームライン 8 3 6 に結合される。したがって、S M E 8 0 4、8 0 5 は、データストリームライン 8 3 6 の内の選択された 1 つまたは複数がその上に高信号を有するときに高信号を出力するようにプログラムできる。たとえば、S M E 8 0 4 は、第 1 のメモリセル 8 3 2 (たとえば、ビット 0) を高に設定させ、他のすべてのメモリセル 8 3 2 (たとえば、ビット 1 から 2 5 5) を低に設定させることができる。それぞれの検出回路 8 3 8 がアクティブ状態にあるときに、S M E 8 0 4 は、ビット 0 に対応するデータストリームライン 8 3 6 がその上に高信号を有すると出力 8 2 6 に対する高信号を出力する。他の例では、S M E 8 0 4 は、複数のデータストリームライン 8 3 6 の内の 1 つが、適切なメモリセル 8 3 2 を高値に設定することによってその上に高信号を有するときに高信号を出力するように設定できる。

#### 【 0 0 6 2 】

例では、メモリセル 8 3 2 は、関連するレジスタからビットを読み取ることによって高値または低値に設定できる。したがって、S M E 8 0 4 は、コンパイラによって作成された画像をレジスタの中に記憶し、関連するメモリセル 8 3 2 の中にレジスタ内のビットをロードすることによってプログラムできる。例では、コンパイラによって作成される画像は、高ビットおよび低ビット(たとえば、1 および 0)のバイナリ画像を含む。画像は、S M E 8 0 4、8 0 5 をカスケードすることによって F S M として動作するように F S M エンジン 8 0 0 をプログラムできる。たとえば、第 1 の S M E 8 0 4 は、検出回路 8 3 8 をアクティブ状態に設定することによってアクティブ状態に設定できる。第 1 の S M E 8 0 4 は、ビット 0 に対応するデータストリームライン 8 3 6 がその上に高信号を有するときに高信号を出力するように設定できる。第 2 の S M E 8 0 5 は、当初、非アクティブ状態に設定できるが、アクティブ時、ビット 1 に対応するデータストリームライン 8 3 6 がその上に高信号を有すると高信号を出力するように設定できる。第 1 の S M E 8 0 4 および第 2 の S M E 8 0 5 は、第 2 の S M E 8 0 5 の入力 8 1 6 に結合するように、第 1 の S M E 8 0 4 の出力 8 2 6 を設定することによってカスケードできる。したがって、高信号がビット 0 に対応するデータストリームライン 8 3 6 で検知されるとき、第 1 の S M E 8 0 4 は出力 8 2 6 で高信号を出力し、第 2 の S M E 8 0 5 の検出回路 8 3 8 をアクティブ状態に設定する。高信号がビット 1 に対応するデータストリームライン 8 3 6 で検知されるとき、第 2 の S M E 8 0 5 は別の S M E 8 0 5 を活性化するために出力 8 2 8 で、または F S M エンジン 8 0 0 からの出力のために高信号を出力する。

#### 【 0 0 6 3 】

図 1 0 は、コンパイラが、ソースコードを、並行処理機械をプログラムするように構成された画像に変換するための方法 1 0 0 0 の例を示す。方法 1 0 0 0 は、構文木にソースコードをパースすること(ブロック 1 0 0 2)と、構文木をオートマトンに変換すること

10

20

30

40

50

(ブロック1004)と、オートマトンを最適化すること(ブロック1006)と、オートマトンをネットリストに変換すること(ブロック1008)と、ハードウェア上にネットリストを格納すること(ブロック1010)と、ネットリストの経路を定めること(ブロック1012)と、結果として生じる画像を公開すること(ブロック1014)とを含む。

#### 【0064】

例では、コンパイラは、ソフトウェア開発者が、FSMエンジン800上でFSMを実装するために画像を作成できるようにするアプリケーションプログラミングインタフェース(API)を含む。コンパイラは、ソースコードの正規表現の入力セットを、FSMエンジン800をプログラムするように構成される画像に変換する方法を提供する。コンパイラは、フォンノイマンアーキテクチャを有するコンピュータ用の命令によって実装できる。これらの命令は、コンピュータ上のプロセッサにコンパイラの機能を実行させることができる。たとえば、命令は、プロセッサによって実行されるときに、プロセッサがアクセス可能であるソースコードに対して、ブロック1002、1004、1006、1008、1010、1012、および1014で説明されるアクションをプロセッサに実行させることができる。フォンノイマンアーキテクチャを有する例のコンピュータは、図13に示され、後述される。

#### 【0065】

例では、ソースコードは、記号のグループ内で記号のパターンを識別するためのサーチ文字列を記述する。サーチ文字列を説明するために、ソースコードは、複数の正規表現(regex)を含むことがある。regexは、記号検索パターンを記述するための文字列である場合がある。regexは、プログラミング言語、テキストエディタ、ネットワークセキュリティ、および他等の多様なコンピュータの領域で幅広く使用されている。例では、コンパイラによってサポートされている正規表現は、非構造化データの検索のための検索基準を含む。非構造化データは、自由形式を有し、データ内のワードに何のインデクシングも適用されないデータを含むことがある。ワードは、データの内部にバイト、プリンタブルおよび非プリンタブルの任意の組合せを含むことがある。例では、コンパイラは、Perl(Perl互換性正規表現(PCRE))言語、PHP言語、Java(登録商標)言語、および.NET言語を含むregexを実装するための複数の異なるソースコード言語をサポートできる。

#### 【0066】

ブロック1002で、コンパイラはソースコードを分析し、相対的に接続された演算子の配列を形成することができ、異なるタイプの演算子は、ソースコードによって実装される異なる関数(たとえば、ソースコード内でregexによって実装される異なる関数)に相当する。ソースコードを分析すると、ソースコードの一般的表現を生じさせることがある。例では、一般的表現は、構文木として知られているツリーグラフの形でソースコード内のregexの符号化された表現を含む。本明細書に説明されている例は、他の例では構文木(「抽象構文木」としても知られている)としての配置を指すが、具象構文木または他の配置も使用できる。

#### 【0067】

上述されたように、コンパイラはソースコードの複数の言語をサポートできるので、分析は、言語に関わりなく、たとえば構文木等の非言語特性表現にソースコードを変換する。したがって、コンパイラによる追加の処理(ブロック1004、ブロック1006、ブロック1008、ブロック1010)は、ソースコードの言語に関わりなく、共通の入力構造から作業できる。

#### 【0068】

上記に留意されたように、構文木は、相対的に接続されている複数の演算子を含む。構文木は、複数の異なるタイプの演算子を含むことがある。つまり、異なる演算子は、ソースコード内のregexによって実装される異なる関数に相当することがある。

#### 【0069】

ブロック 1 0 0 4 で、構文木はオートマトンに変換される。オートマトンは、F S M のソフトウェアモデルを含み、相応して決定論的または非決定論的と分類できる。決定論的オートマトンは、所与のときに実行の単一の経路を有する。一方、非決定論的オートマトンは、実行の複数の並行経路を有する。オートマトンは、複数の状態を含む。構文木をオートマトンに変換するために、演算子および構文木内の演算子間の関係性が、状態間の遷移を含む状態に変換される。例では、オートマトンは、F S M エンジン 8 0 0 のハードウェアに部分的に基づいて変換できる。

【 0 0 7 0 】

例では、オートマトンの入力記号は、アルファベット、数字 0 から 9、および他のプリンタブル文字を含む。例では、入力記号は、0 から 2 5 5 を含めたバイト値によって表される。例では、オートマトンは、グラフのノードが状態のセットに対応する有向グラフとして表すことができる。例では、入力記号 上の状態 p から状態 q への遷移、つまり ( p , ) は、ノード p からノード q への有向接続によって示される。例では、オートマトンの逆転は、なんらかの記号 上の各遷移 p q が、同じ記号上で逆転される q p である新規オートマトンを生じさせる。逆転では、開始状態は最終状態になり、最終状態が開始状態になる。例では、オートマトンによって受け入れられる（たとえば、一致される）言語は、オートマトンに順次に入力されるときに最終状態に到達するすべての考えられる文字列のセットである。オートマトンによって受け入れられる言語の各文字列は、開始状態から 1 つまたは複数の最終状態への経路をたどる。

【 0 0 7 1 】

ブロック 1 0 0 6 で、オートマトンが構築されたのち、オートマトンは、とりわけその複雑度およびサイズを削減するために最適化される。オートマトンは、冗長な状態を組み合わせることによって最適化できる。

【 0 0 7 2 】

ブロック 1 0 0 8 で、最適化されたオートマトンが、ネットリストに変換される。オートマトンをネットリストに変換すると、オートマトンの各状態は F S M エンジン 8 0 0 上のハードウェア要素（たとえば、S M E 8 0 4、8 0 5、他の要素 8 2 4）にマッピングされ、ハードウェア要素間の接続が決定される。

【 0 0 7 3 】

ブロック 1 0 1 0 で、ネットリストは、ネットリストの各ノードに対応するターゲット装置（たとえば、S M E 8 0 4、8 0 5、特殊目的要素 8 2 4）の特定のハードウェア要素を選択するために格納される。例では、格納すると、F S M エンジン 8 0 0 のための一般的な入力および出力の制約に基づいて、それぞれの特定のハードウェア要素が選択される。

【 0 0 7 4 】

ブロック 1 0 1 2 で、格納されたネットリストは、選択されたハードウェア要素とともに結合して、ネットリストによって記述される接続を達成するために、プログラマブルスイッチ（たとえば、ブロック間スイッチ 8 0 3、ブロック内スイッチ 8 0 8、および行内スイッチ 8 1 2）のための設定値を決定するために経路が選択される。例では、プログラマブルスイッチのための設定が、選択されたハードウェア要素、およびプログラマブルスイッチのための設定値を接続するために使用される P S M エンジン 8 0 0 の特定の導線を決定することによって決定される。経路選択は、ブロック 1 0 1 0 での格納、ハードウェア要素間の接続のより多くの制限を考慮に入れることがある。したがって、経路選択は、F S M エンジン 8 0 0 に対する導線の実際の制限を考えると、適切な接続を行うために大局的な格納で決定されるハードウェア要素のいくつかの場所を調整してよい。

【 0 0 7 5 】

いったんネットリストが格納され、経路選択されると、格納され、経路選択されたネットリストは、F S M エンジン 8 0 0 のプログラミングのための複数のビットに変換できる。この複数のビットは、本明細書で画像と呼ばれている。

【 0 0 7 6 】

ブロック 1 0 1 4 で、画像はコンパイラによって公表される。画像は、特定のハードウェア要素および/または F S M エンジン 8 0 0 のプログラマブルスイッチをプログラムするための複数のビットを含む。画像が複数のビット（たとえば、0 および 1）を含む実施形態では、画像はバイナリ画像と呼ぶことができる。ビットは、F S M エンジン 8 0 0 にロードされ、プログラムされた F S M エンジン 8 0 0 が、ソースコードによって記述される機能性を有する F S M を実装するように、S M E 8 0 4、8 0 5、特殊目的要素 8 2 4、およびプログラマブルスイッチの状態をプログラムできる。格納（ブロック 1 0 1 0）および経路選択（1 0 1 2）は、F S M エンジン 8 0 0 内での特定の場所にある特定のハードウェア要素を、オートマトンの特定の状態にマッピングできる。したがって、画像内のビットは、特定のハードウェア要素および/またはプログラマブルスイッチをプログラムし、記述される関数（複数の場合がある）を実装できる。例では、画像は、コンピュータ可読媒体に機械コードを保存することによって公表できる。別の例では、画像は、画像を表示装置上で表示することによって公表できる。さらに別の例では、画像は、F S M エンジン 8 0 0 上に画像をロードするためのプログラミング装置等の別の装置に画像を送ることによって公表できる。さらに別の例では、画像は、並行処理機械（たとえば、F S M エンジン 8 0 0）上に画像をロードすることによって公表できる。

10

#### 【 0 0 7 7 】

例では、画像は、画像からのビット値を S M E 8 0 4、8 0 5 および他のハードウェア要素 8 2 4 に直接的にロードする、もしくは 1 つまたは複数のレジスタの中に画像をロードし、次いで S M E 8 0 4、8 0 5 および他のハードウェア要素 8 2 4 にレジスタからのビット値を書き込むかのどちらかによって F S M エンジン 8 0 0 上にロードできる。例では、プログラマブルスイッチ（たとえば、ブロック間スイッチ 8 0 3、ブロック内スイッチ 8 0 8、および行内スイッチ 8 1 2）の状態。例では、F S M エンジン 8 0 0 のハードウェア要素（たとえば、S M E 8 0 4、8 0 5、他の要素 8 2 4、プログラマブルスイッチ 8 0 3、8 0 8、8 1 2）は、プログラミング装置および/またはコンピュータが、1 つまたは複数のメモリアドレスに画像を書き込むことによって F S M エンジン 8 0 0 上に画像をロードできるように、メモリマッピングされる。

20

#### 【 0 0 7 8 】

本明細書に説明される方法の例は、少なくとも部分的に機械でまたはコンピュータで実装できる。いくつかの例は、上記の例に説明される方法を実行するように電子機器を構成するために動作可能な命令で符号化されたコンピュータ可読媒体または機械可読媒体を含むことがある。かかる方法の実装は、マイクロコード、アセンブリ言語コード、高水準言語コード等のコードを含むことがある。かかるコードは、多様な方法を実行するためのコンピュータ可読命令を含むことがある。コードは、コンピュータプログラム製品の部分を形成してよい。さらに、コードは、実行中、または他のときに 1 つまたは複数の揮発性の、または不揮発性のコンピュータ可読媒体で有形で記憶されてよい。これらのコンピュータ可読媒体は、ハードディスク、リムーバブル磁気ディスク、リムーバブル光ディスク（たとえば、コンパクトディスクおよびデジタルビデオディスク）、磁気カセット、メモリカードまたはメモリスティック、ランダムアクセスメモリ（R A M）、読取専用メモリ（R O M）等を含んでよいが、これらに限定されない。

30

40

#### 【 0 0 7 9 】

図 1 3 は、一般に、フォンノイマンアーキテクチャを有するコンピュータ 1 5 0 0 の例を示す。本開示の内容を読み、理解すると、当業者は、ソフトウェアプログラムがどのようにしてコンピュータベースのシステムでコンピュータ可読媒体から起動され、ソフトウェアプログラムに定められる関数機能を実行するのかを理解するだろう。当業者は、本明細書に開示される方法を実装し、実行するように設計される 1 つまたは複数のソフトウェアプログラムを作成するために利用できる多様なプログラミング言語もさらに理解するだろう。プログラムは、J a v a（登録商標）言語、C + + 言語、もしくは 1 つまたは複数の言語等のオブジェクト指向言語を使用するオブジェクト指向フォーマットで構造化できる。代わりに、プログラムは、アセンブリ、C 等の手続き言語を使用して手続き指向フォ

50

ーマットで構造化できる。ソフトウェア構成要素は、リモートプロシージャコールまたは他を含む、アプリケーションプログラムインタフェース、またはプロセス間通信技法等の、当業者にとって周知の多くの機構の内のどれかを使用して通信できる。多様な実施形態の教示は、任意の特定のプログラミング言語または環境に制限されていない。

#### 【0080】

したがって、他の実施形態は実現できる。たとえば、コンピュータ、メモリシステム、磁気ディスクまたは光ディスク、なんらかの他の記憶装置、もしくは任意のタイプの電子機器またはシステム等の製造品は、1台または複数のプロセッサ1502によって実行されるときに上記方法に関して説明されるアクションのいずれかを実行することになる、その上に命令1524（たとえば、コンピュータプログラム命令）が記憶されるメモリ（たとえば、電気的な導体、光学的な導体、または電磁的な導体を含む任意のメモリだけではなく、リムーバブル記憶媒体等の）等のコンピュータ可読媒体1522に結合される1台または複数のプロセッサ1502を含むことがある。

10

#### 【0081】

コンピュータ1500は、多くの構成要素に直接的に結合されるプロセッサ1502を有する、および/またはバス1508を使用するコンピュータシステムの形をとることができる。かかる構成要素は、メインメモリ1504、スタティックメモリまたは不揮発性メモリ1506、およびマスメモリ1516を含むことがある。プロセッサ1502に結合される他の構成要素は、ビデオディスプレイ等の出力装置1510、キーボード等の入力装置1512、およびマウス等のカーソル制御装置1514を含むことがある。プロセッサ1502および他の構成部品をネットワーク1526に結合するためのネットワークインタフェース装置1520は、バス1508に結合することもできる。命令1524は、多くの周知の転送プロトコル（たとえば、HTTP）の内の任意の1つを活用するネットワークインタフェース装置1520を介して、ネットワーク1526上でさらに送信または受信できる。バス1508に結合されるこれらの要素のいずれかは、実現される特定の実施形態に応じて、存在しない、単独で存在する、または複数の数で存在することがある。

20

#### 【0082】

例では、プロセッサ1502、メモリ1504、1506、または記憶装置1516の内の1つまたは複数、実行時にコンピュータ1500に本明細書に説明される方法の内の任意の1つまたは複数を実行させることができる命令1524を含むことがある。代替実施形態では、コンピュータ1500は、スタンドアロン装置として動作する、または他の装置に接続する（たとえば、ネットワーク化する）ことができる。ネットワーク化された環境では、コンピュータ1500は、サーバ-クライアントネットワーク環境でのサーバ装置またはクライアント装置の資格で、もしくはピアツーピア（または分散型）ネットワーク環境でピアデバイスとして動作できる。コンピュータ1500は、パーソナルコンピュータ（PC）、タブレットPC、セットトップボックス（STB）、パーソナルデジタルアシスタント（PDA）、セルラー電話、ウェブ機器、ネットワークルータ、ネットワークスイッチまたはネットワークブリッジ、もしくはその装置によって講じられるアクションを指定する命令のセット（順次またはそれ以外）を実行できる任意の装置を含むことがある。さらに、単一のコンピュータ1500だけが示されているが、用語「コンピュータ」は、本明細書に説明される方法論の任意の1つまたは複数を実行するために命令のセット（または複数のセット）を個別にまたは共同で実行する装置の任意の集合体を含むとも解釈されるものとする。

30

40

#### 【0083】

コンピュータ1500は、1つまたは複数の通信プロトコル（たとえば、ユニバーサルシリアルバス（USB）、IEEE1394等）を使用して周辺装置と通信するための出力コントローラ1528を含むこともある。出力コントローラ1528は、たとえば、コンピュータ1500に通信で結合されるプログラミング装置1530に画像を提供できる。プログラミング装置1530は、並行処理機械（たとえば、並行処理機械100、FS

50



Mエンジン 800) をプログラムするように構成できる。他の例では、プログラミング装置 1530 は、コンピュータ 1500 と統合され、バス 1508 に結合できるか、ネットワークインタフェース装置 1520 または別の装置を介してコンピュータ 1500 と通信することができる。

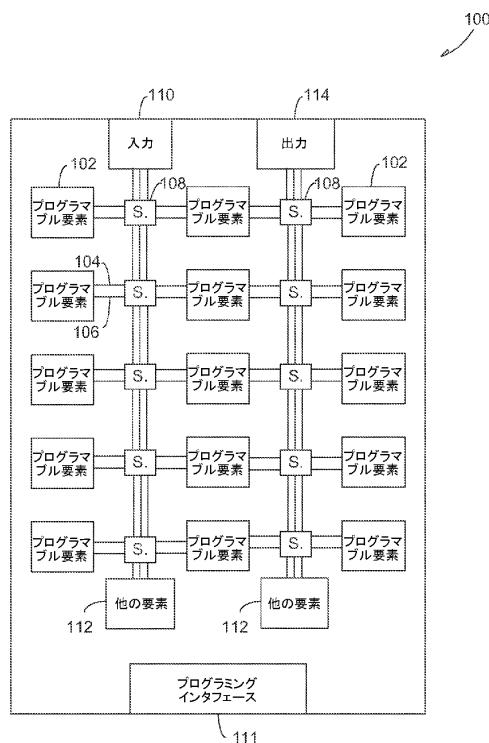
#### 【0084】

コンピュータ可読媒体 1524 は、単一の媒体として示されているが、「コンピュータ可読媒体」は、命令 1524 の 1 つまたは複数のセットを記憶する、単一の媒体または複数の媒体（たとえば、集中データベースまたは分散型データベース、もしくは関連付けられたキャッシュおよびサーバ、ならびにもしくはプロセッサ 1502 レジスタ、メモリ 1504、1506、および記憶装置 1516 等のいろいろな記憶媒体等）を含むと解釈されるべきである。また、用語「コンピュータ可読媒体」は、コンピュータによる実行のための命令のセットを記憶する、符号化する、または搬送することができる、および本発明の方法論の任意の 1 つまたは複数を実行させる、またはかかる命令のセットによって活用される、またはかかる命令のセットと関連付けられるデータ構造を記憶する、符号化する、または搬送することができる任意の媒体を含むとも解釈されるものとする。用語「コンピュータ可読媒体」は、したがって、ソリッドステートメモリ、光媒体、および磁気媒体等の有形的表現媒体を含むが、これらに限定されないと解釈されるものとする。

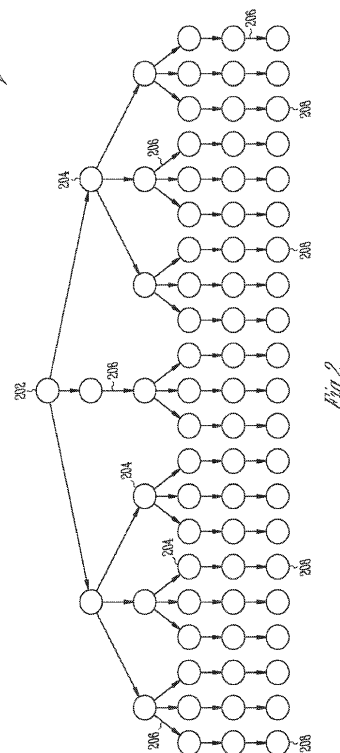
#### 【0085】

要約書は、読者が技術的な開示の性質および要旨を確かめることを可能にする要約書を要求する、連邦法施行規則第 37 巻、第 1.72(b) 項に準拠するために提供される。要約書は、それが請求項の範囲または意味を制限する、または解釈するために使用されないという理解で提出される。以下の特許請求の範囲は、発明を実施するための形態にこれにより組み込まれ、各請求項は、それ自体、別個の実施形態として効力がある。

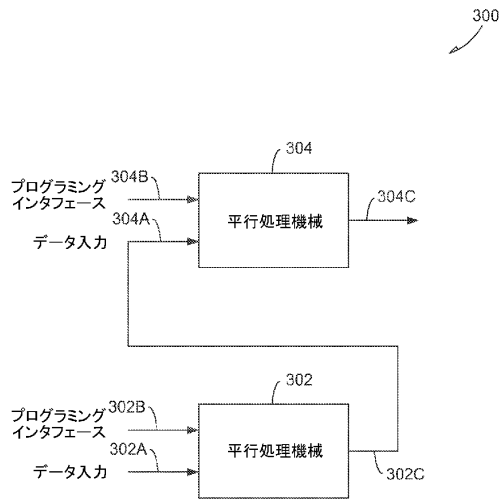
【図 1】



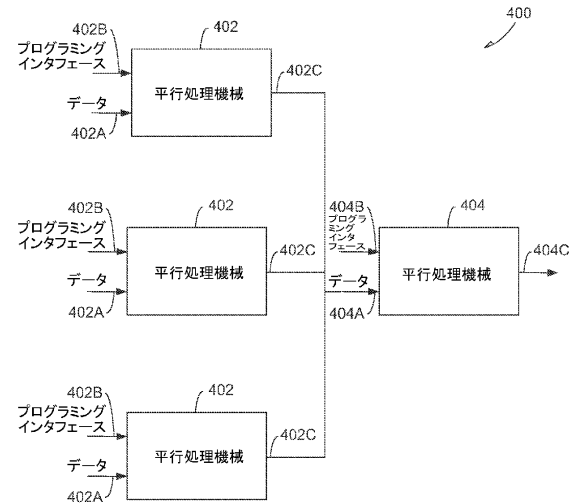
【図 2】



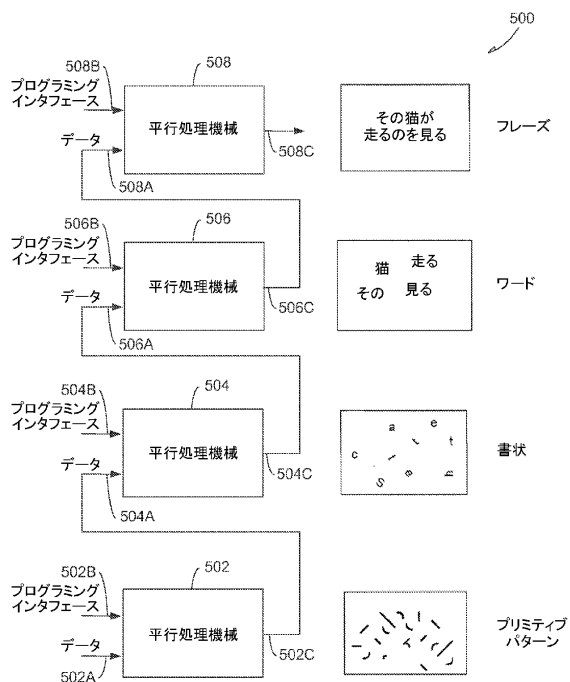
【図 3】



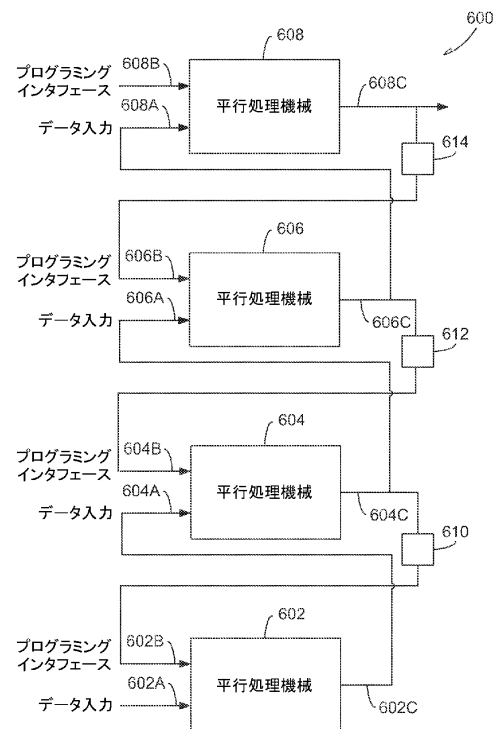
【図 4】



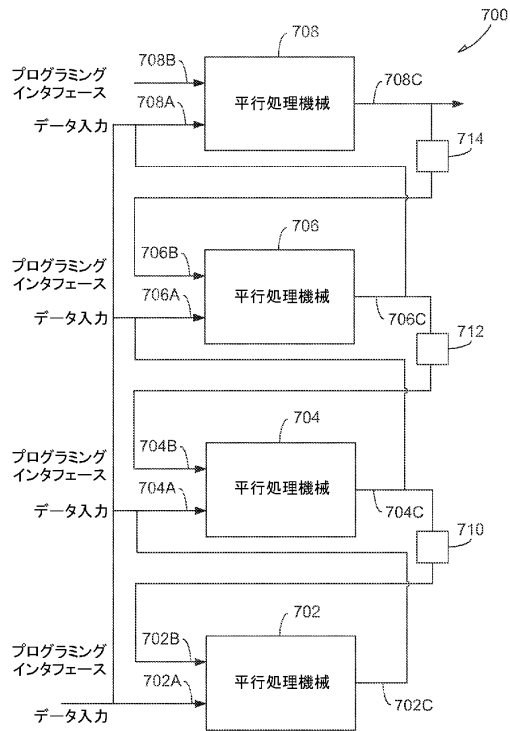
【図 5】



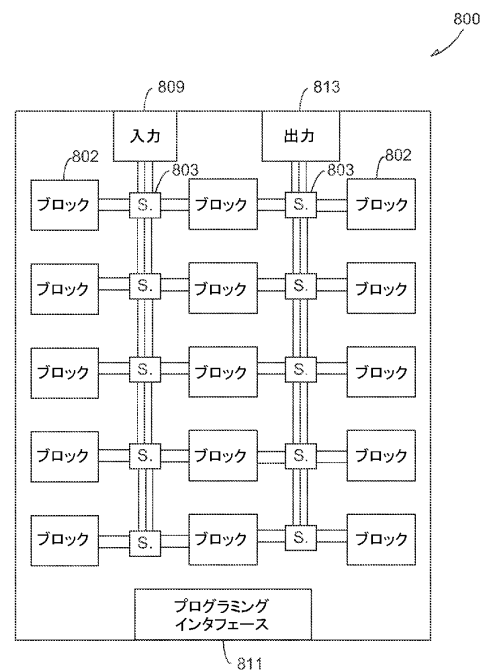
【図 6】



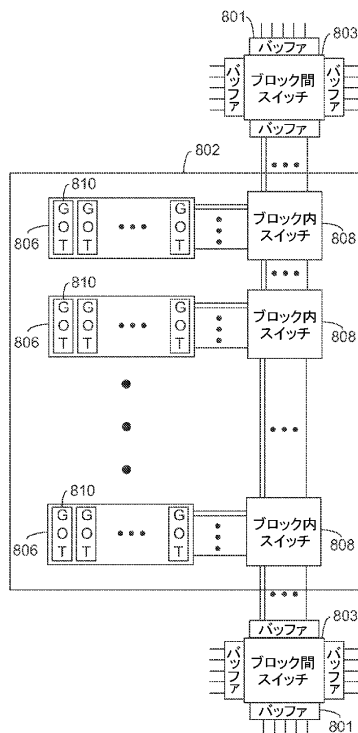
【図 7】



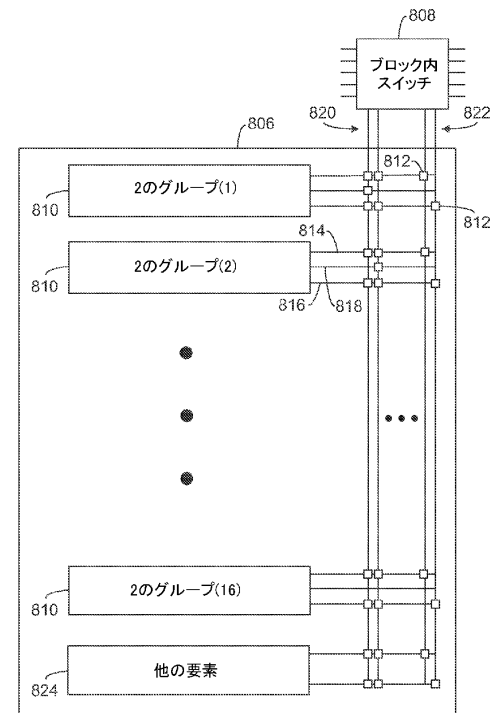
【図 8】



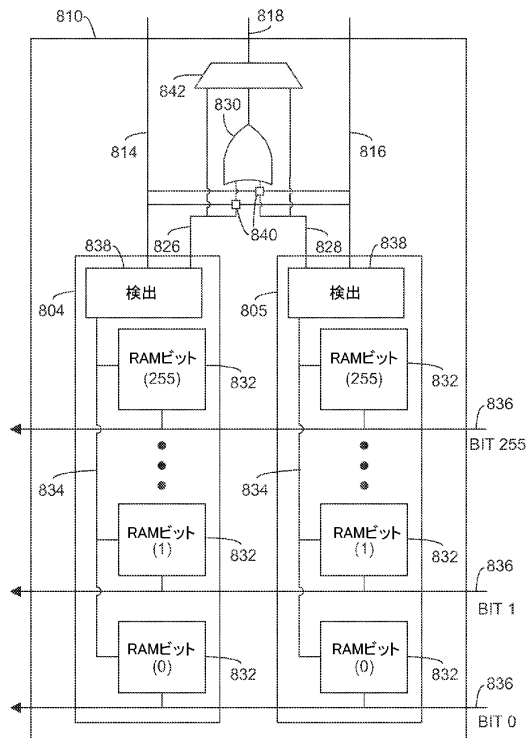
【図 9】



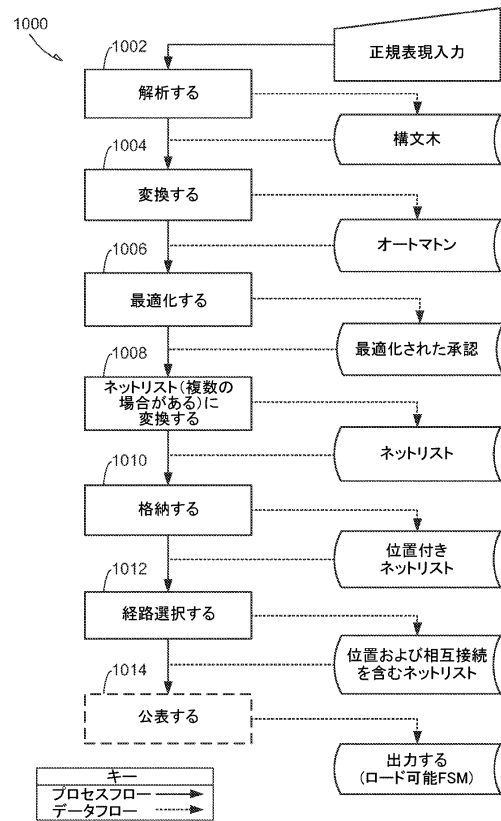
【図 10】



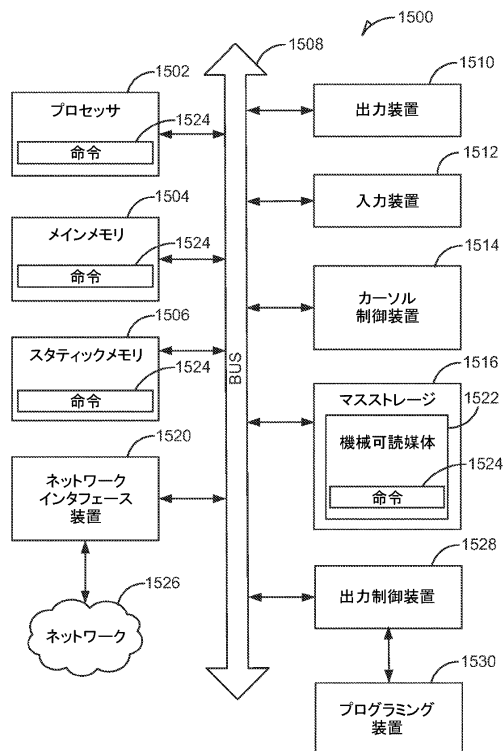
【図 1 1】



【図 1 2】



【図 1 3】



---

フロントページの続き

(56)参考文献 特開2005-122466(JP,A)  
特開平04-291662(JP,A)  
特開2002-358500(JP,A)

(58)調査した分野(Int.Cl., DB名)

G06N	3/00 - 3/12
G06N	99/00
G06T	7/00
G06F	9/44