【公報種別】特許公報の訂正
【部門区分】第６部門第３区分
【発行日】令和6年10月2日(2024.10.2)

【特許番号】特許第7546571号(P7546571)
【登録日】令和6年8月29日(2024.8.29)
【特許公報発行日】令和6年9月6日(2024.9.6)
【年通号数】登録公報(特許)2024-168
【出願番号】特願2021-539860(P2021-539860)
【訂正要旨】特許権者の住所の誤載により、下記のとおり全文を訂正する。
【国際特許分類】
*Ｇ０６Ｆ　16/215(2019.01)*
【ＦＩ】
Ｇ０６Ｆ　16/215
【記】別紙のとおり

(19)日本国特許庁(JP)

# (12)特許公報(B2)

(11)特許番号
特許第7546571号
(P7546571)

(45)発行日 令和6年9月6日(2024.9.6)

(24)登録日 令和6年8月29日(2024.8.29)

(51)国際特許分類
*G06F 16/215(2019.01)*

FI
G06F 16/215

請求項の数 13 (全96頁)

(21)出願番号 特願2021-539860(P2021-539860)
(86)(22)出願日 令和2年6月18日(2020.6.18)
(65)公表番号 特表2022-538705(P2022-538705A)
(43)公表日 令和4年9月6日(2022.9.6)
(86)国際出願番号 PCT/US2020/038431
(87)国際公開番号 WO2020/263675
(87)国際公開日 令和2年12月30日(2020.12.30)
審査請求日 令和5年5月25日(2023.5.25)
(31)優先権主張番号 62/865,797
(32)優先日 令和1年6月24日(2019.6.24)
(33)優先権主張国・地域又は機関
米国(US)
(31)優先権主張番号 16/904,298
(32)優先日 令和2年6月17日(2020.6.17)
最終頁に続く

(73)特許権者 502303739
オラクル・インターナショナル・コーポレイション
アメリカ合衆国 ９４０６５ カリフォルニア州 レッドウッド ショアーズ，メール ストップ ５オーピー７ オラクル パークウェイ ５００
(74)代理人 110001195
弁理士法人深見特許事務所
(72)発明者 マラック，マイケル
アメリカ合衆国、９４０６５ カリフォルニア州、レッドウッド・シティー、オラクル・パークウェイ、５００
(72)発明者 リーバス，ルイス・イー
アメリカ合衆国、９４０６５ カリフォ
最終頁に続く

(54)【発明の名称】 コンテキストを用いた陰性例のための正規表現生成

(57)【特許請求の範囲】
【請求項１】
正規表現を生成する方法であって、
１つまたは複数のプロセッサを備える正規表現生成器が、１つまたは複数の陽性のキャラクタシーケンスを含む第１の選択を受け取ることを含み、前記１つまたは複数の陽性のキャラクタシーケンスの各々は、前記正規表現生成器によって生成される正規表現によってマッチされるべき陽性例に対応し、前記方法はさらに、
前記正規表現生成器が第１の正規表現を生成することを含み、前記第１の正規表現は前記陽性例にマッチし、前記方法はさらに、
前記正規表現生成器が、１つまたは複数の陰性のキャラクタシーケンスを含む第２の選択を受け取ることを含み、前記１つまたは複数の陰性のキャラクタシーケンスの各々は、前記正規表現生成器によって生成される前記正規表現によってマッチされるべきでない陰性例に対応し、前記方法はさらに、
前記第２の選択を受け取ることに応答して、前記陰性例に対応する前記１つまたは複数の陰性のキャラクタシーケンスのコンテキストを判断することと、
前記１つまたは複数の陰性のキャラクタシーケンスの前記判断されたコンテキストに基づいて、前記第１の正規表現を更新することとを含む、方法。
【請求項２】
前記第１の選択を受け取ることは、ユーザインターフェイスを介して、データセットの第１のデータセルにおいて前記１つまたは複数の陽性のキャラクタシーケンスの選択を受

け取ることを含む、請求項１に記載の方法。
【請求項３】
前記正規表現生成器が、前記データセット内の複数のデータセル内において、前記１つまたは複数の陽性のキャラクタシーケンスを含む第１の選択に対応するキャラクタシーケンスを自動的に選択することをさらに含む、請求項２に記載の方法。
【請求項４】
前記第２の選択を受け取ることは、前記ユーザインターフェイスを介して、前記データセットの第２のデータセルにおいて前記１つまたは複数の陰性のキャラクタシーケンスの選択を受け取ることを含む、請求項３に記載の方法。
【請求項５】
前記正規表現生成器によって、前記データセット内の前記複数のデータセル内において、前記１つまたは複数の陰性のキャラクタシーケンスを含む第２の選択に対応するキャラクタシーケンスを自動的に選択することをさらに含む、請求項４に記載の方法。
【請求項６】
前記第１の選択は、第１の強調表示フォーマットで強調表示され、前記第２の選択は、前記第１の強調表示フォーマットとは異なる第２の強調表示フォーマットで強調表示される、請求項３から５のいずれか１項に記載の方法。
【請求項７】
前記陰性例に対応する前記１つまたは複数の陰性のキャラクタシーケンスの前記コンテキストを判断することは、
前記第２の選択の、埋め込まれた強調表示位置を識別することと、
前記第２の選択の前記埋め込まれた強調表示位置の左側のデータからコンテキストを判断することと、
前記強調表示された第２の選択の前記埋め込まれた強調表示位置の右側のデータからコンテキストを判断することととを含む、請求項６に記載の方法。
【請求項８】
前記陰性例に対応する前記１つまたは複数の陰性のキャラクタシーケンスの前記コンテキストを判断することは、さらに、
前記埋め込まれた強調表示位置の左側のデータからの前記判断されたコンテキストおよび前記埋め込まれた強調表示位置の右側のデータからの前記判断されたコンテキストに基づいて自動的に選択された、前記１つまたは複数の陰性のキャラクタシーケンスを含む前記第１の選択に対応する、前記データセット内の前記複数のデータセル内におけるキャラクタシーケンスをフィルタリングすることと、
前記データセット内の前記複数のデータセル内の前記選択されたキャラクタシーケンスから、前記選択された１つまたは複数の陰性のキャラクタシーケンスに対応する前記フィルタリングされたキャラクタシーケンスを除去することとを含む、請求項７に記載の方法。
【請求項９】
前記埋め込まれた強調表示位置の左側のデータから前記コンテキストを判断することは、前記埋め込まれた強調表示位置の左側の第１のスパンを識別することを含み、
前記選択された１つまたは複数の陰性のキャラクタシーケンスに対応する前記データセット内の前記複数のデータセル内の前記キャラクタシーケンスをフィルタリングすることは、前記選択された１つまたは複数の陰性のキャラクタシーケンスに対応する前記複数のデータセル内の前記キャラクタシーケンスにおいて、前記埋め込まれた強調表示位置の左側の前記第１のスパンとマッチしないスパンを識別することをさらに含む、請求項８に記載の方法。
【請求項１０】
前記埋め込まれた強調表示位置の左側のデータからコンテキストを判断することは、前記埋め込まれた強調表示の左側の第２のスパンを識別することをさらに含み、
前記選択された１つまたは複数の陰性のキャラクタシーケンスに対応する前記データセット内の前記複数のデータセル内の前記キャラクタシーケンスをフィルタリングすること

は、前記選択された１つまたは複数の陰性のキャラクタシーケンスに対応する前記複数のデータセル内の前記キャラクタシーケンスにおいて、前記埋め込まれた強調表示位置の左側の前記第２のスパンとマッチしないスパンを識別することをさらに含む、請求項９に記載の方法。

【請求項１１】

前記埋め込まれた強調表示位置の右側のデータから前記コンテキストを判断することは、前記埋め込まれた強調表示位置の右側の第１のスパンを識別することを含み、

前記１つまたは複数の陰性のキャラクタシーケンスを含む第２の選択に対応する前記データセット中の前記複数のデータセル中の前記キャラクタシーケンスをフィルタリングすることは、前記１つまたは複数の陰性のキャラクタシーケンスを含む第２の選択に対応する前記複数のデータセル中の前記キャラクタシーケンスにおいて、前記埋め込まれた強調表示位置の右側の前記第１のスパンとマッチしないスパンを識別することをさらに含む、請求項７に記載の方法。

【請求項１２】

正規表現生成器サーバコンピュータであって、

プロセッサと、

メモリと、

前記プロセッサに結合される記憶媒体とを備え、前記記憶媒体は、請求項１から１１のいずれか１項に記載の方法を実現するために前記プロセッサによって実行可能な命令を格納する、正規表現生成器サーバコンピュータ。

【請求項１３】

１つまたは複数のプロセッサによって実行されると、当該１または複数のプロセッサに、請求項１から１１のいずれか１項に記載の方法を実行させるための命令を有したプログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
関連出願の相互参照
本出願は、２０１９年６月１１日に提出された、「USER INTERFACE FOR REGULAR EXPRESSION GENERATION（正規表現生成のためのユーザインターフェイス）」と題される米国特許出願第１６／４３８，３２７号の一部係属であり、米国特許法第１１９条（ｅ）に基づき、２０１８年６月１３日に提出された「AUTOMATED GENERATION OF REGULAR EXPRESSIONS（自動化された正規表現生成）」と題される米国仮特許出願第６２／６８４，４９８号に対する優先権を主張し、および米国特許法第１１９条（ｅ）に基づき、２０１８年１０月２２日に提出された「AUTOMATED GENERATION OF REGULAR EXPRESSIONS（自動化された正規表現生成）」と題される米国仮特許出願第６２／７４９，００１号に対する優先権を主張する。本出願はまた、米国特許法第１１９条（ｅ）に基づき、２０１９年６月２４日に提出された「AUTOMATED GENERATION OF REGULAR EXPRESSIONS（正規表現の自動生成）」と題される米国仮特許出願第６２／８６５，７９７号に対する優先権を主張する。それらの全内容は、あらゆる目的のために参照により本明細書に組み込まれる。
【背景技術】
【０００２】
背景
ビッグデータ解析システムは、予測解析、ユーザ挙動解析、および他の高度なデータ解析に使用することができる。しかしながら、有用な結果を提供するために任意のデータ解析が効果的に実行される前に、最初のデータセットは、クリーンかつキュレートされたデータセットにフォーマットされる必要があり得る。このデータオンボード化は、多くの場合、さまざまな異なるデータソースおよび／またはデータストリームからのデータが単一のデータリポジトリにコンパイルされ得るクラウドベースのデータリポジトリならびに他のビッグデータシステムに対する課題を提示する。そのようなデータは、複数の異なるフォーマットの構造化データ、異なるデータモデルに従った半構造化データ、およびさらには非構造化データを含み得る。そのようなデータのリポジトリは、多くの場合、さまざまな異なるフォーマットおよび構造内のデータ表現を含み、重複データおよび誤ったデータを含む場合もある。これらのデータリポジトリが報告、予測モデリング、および他の解析タスクのために解析されるとき、初期データセットの低信号対雑音比は、不正確であるかまたは有用でない結果につながり得る。
【０００３】
データフォーマッティングおよび前処理の問題に対する多くの現在の解決策は、データ解析を実行する前にデータを共通のフォーマットに操作するために、データを浄化およびキュレートするための手動およびアドホック処理を含む。これらの手動処理は、特定のより小さいデータセットに対しては有効であり得るが、そのような処理は、大規模なデータセットを前処理およびフォーマットすることを試みる場合は、非効率的かつ非実用的であり得る。
【発明の概要】
【課題を解決するための手段】
【０００４】
概要
本明細書で説明される態様は、正規表現を生成するためのさまざまな技法を提供する。本明細書で使用される場合、「正規表現」は、より長い入力テキストストリング内のマッチを検索するために使用され得る、パターンを定義するキャラクタのシーケンスを指し得る。いくつかの実施形態では、正規表現は、記号のワイルドカードマッチング言語を使用して構成されてもよく、正規表現によって定義されたパターンは、キャラクタストリングとマッチするよう、および／または入力として与えられるキャラクタストリングから情報

を抽出するよう、使用されてもよい。本明細書で説明するさまざまな実施形態では、データ処理システムとして実現される正規表現生成器を使用して、入力テキストデータを受信および表示し、クライアントユーザインターフェイスを介して入力テキストの特定のキャラクタサブセットの選択を受け取り、次いで、選択されたキャラクタサブセットに基づいて１つまたは複数の正規表現を生成することができる。１つまたは複数の正規表現を生成した後、正規表現エンジンを使用して、正規表現のパターンを１つまたは複数のデータセットに対してマッチさせることができる。さまざまな実施形態において、正規表現にマッチするデータは、抽出、再フォーマット、または修正などされてもよい。場合によっては、正規表現にマッチするデータに基づいて、追加の列、テーブル、または他のデータセットが作成されてもよい。

【０００５】

本明細書で説明するいくつかの態様によれば、データ処理システムを介して実現される正規表現生成器は、１つまたは複数の正規表現コードの、異なるセットによって共有される、判断された最長共通サブシーケンス（ＬＣＳ）に基づいて、正規表現を生成することができる。正規表現コード（カテゴリコードとも称され得る）は、たとえば、英語アルファベットの文字に対するL、数字に対するN、空白に対するZ、句読点に対するP、および他の記号に対するSを含み得る。１つまたは複数の正規表現コードの各セットは、ユーザインターフェイスを介して入力データとして受信された１つまたは複数のキャラクタの異なるシーケンスから変換され得る。ＬＣＳから除外される正規表現コードは、任意選択および／または代替として表され得る。いくつかの実施形態では、正規表現コードは、正規表現コードの最小発生数に関連付けられてもよい。追加または代替として、正規表現コードは、正規表現コードの最大発生数に関連付けられてもよい。たとえば、あるカテゴリコードのセットは、ＬＣＳの特定の部分が、ある文字を、もしあったとしてもせいぜい１回含む旨を示すよう、L〈0,1〉を含んでもよい。以下でより詳細に説明するように、入力データを中間正規表現コード（ＩＲＥＣ）として一般化することは、非常にわずかな入力データを使用することを含むさまざまな技術的利点を提供し得、それは、まだ見られていないデータにおける偽陽性マッチまたは偽陰性マッチに屈しない正規表現のほぼ即時的な生成を可能にする。

【０００６】

本明細書で説明される追加の態様によれば、正規表現は、３つ以上のキャラクタシーケンスを含む入力データに基づいて生成され得る。３つ以上のキャラクタシーケンスが入力データとして識別される場合、キャラクタシーケンスのＬＣＳを識別する正規表現生成器は、ランタイムの指数関数的な増加をもたらし得る。すべてのキャラクタシーケンスのＬＣＳを充分に機能する態様で識別するために、正規表現生成器は、２つのキャラクタシーケンスの各別個の組み合わせに対してＬＣＳアルゴリズムを実行してもよい。ＬＣＳアルゴリズムの結果に基づいて、全結合グラフを生成してもよく、各グラフノードは異なるキャラクタシーケンスを表し、各グラフエッジの長さは、グラフエッジを定義するノードのＬＣＳに対応する。次いで、全結合グラフに対して最小スパニングツリーの深さ優先トラバースを実行することによって、キャラクタシーケンスを選択する順序を判断してもよい。

【０００７】

本明細書で説明されるさらなる態様は、陽性キャラクタシーケンス例および陰性キャラクタシーケンス例の両方を含む入力に基づいて正規表現を生成することに関する。陽性例は、生成されるべき正規表現にマッチするキャラクタのシーケンスを指し得、陰性例は、生成されるべき正規表現にマッチしないキャラクタのシーケンスを指し得る。いくつかの実施形態では、陽性例および陰性例の両方が受け取られた場合、正規表現生成器は、弁別子、つまり陽性例を陰性例から区別する１つまたは複数のキャラクタの最短サブシーケンスを識別してもよい。選択された弁別子は、（たとえばカテゴリコードで表現された）最短シーケンスであってもよく、陽性または陰性のいずれかであってもよく、したがって、陽性例はマッチし、陰性例はマッチしないことになる。次いで、弁別子は、正規表現生成器によって生成される正規表現にハードコード化されてもよい。場合によっては、最短サ

ブシーケンスは、陰性例のプレフィックス部分またはサフィックス部分に含まれてもよい。

【０００８】

本明細書で説明されるさらなる態様は、正規表現を生成するために入力データが提供され得る１つまたは複数のユーザインターフェイスに関する。いくつかの実施形態では、ユーザインターフェイスは、正規表現生成サーバに通信可能に結合されたクライアントデバイスに表示されてもよい。ユーザインターフェイスは、サーバによって、クライアントデバイスによって、またはサーバおよびクライアントにおいて実行されるソフトウェアコンポーネントの組み合わせによってプログラム的に生成され得る。ユーザインターフェイスを介して受信された入力データは、陽性例または陰性例を表し得る１つまたは複数のキャラクタシーケンスのユーザ選択に対応し得る。場合によっては、ユーザインターフェイスは、第２のキャラクタシーケンス内の第１のキャラクタシーケンスの選択を含む入力データをサポートしてもよい。たとえば、ユーザは、より大きな、以前に強調表示されたキャラクタシーケンス内の、１つまたは複数のキャラクタを強調表示することができ、第２のユーザ選択は、より大きな第１のユーザ選択のためのコンテキストを提供することができる。これは、入力データが、より高い特異性で正規表現生成器に提供されることを可能にし、正規表現生成器に「コンテキスト」を提供して、それが、偽陽性を回避する正規表現を生成できるようにすることを可能にする。ユーザインターフェイスを介してユーザがキャラクタシーケンスを選択することに応答して、正規表現生成器は正規表現を生成して表示してもよい。たとえば、ユーザがキャラクタの第１のシーケンスを強調表示すると、正規表現生成器は、キャラクタの第１のシーケンスにマッチする正規表現、ならびに他の同様のキャラクタシーケンス（たとえば、マッチするシーケンスについてユーザの意図と整合する）を生成し、表示することができる。ユーザがキャラクタの第２のシーケンスを強調表示すると、正規表現生成器は、キャラクタの第１のシーケンスとキャラクタの第２のシーケンスとの両方を包含する更新された正規表現を生成してもよい。次いで、ユーザが（たとえば第１のシーケンスまたは第２のシーケンスのいずれか内で）キャラクタの第３のシーケンスを強調表示すると、正規表現生成器は正規表現を再び更新してもよいなどとなる。

【０００９】

本明細書で説明される追加の態様によれば、正規表現は、１つまたは複数の入力シーケンス例からの最長共通サブシーケンスに基づいて生成され得るが、例のいくつかにのみ存在するキャラクタを取り扱うこともできる。いくつかの入力例においてのみ存在するキャラクタを取り扱うために、正規表現コードの最小発生数および最大発生数の両方が追跡されるスパンを定義してもよい。所与の入力例のすべてにスパンが存在しない可能性がある場合、最小発生数はゼロにセットされてもよい。次いで、これらの最小数および最大数は、正規表現マルチプリシティ構文にマッピングされ得る。最長共通サブシーケンス（ＬＣＳ）アルゴリズムを、すべての入力例には現れない「任意選択の」スパン（たとえばゼロの最小長さ）を含む、入力例から導出されたキャラクタのスパン上で実行してもよい。以下で説明するように、連続するスパンは、ＬＣＳアルゴリズムの実行中にマージされてもよい。そのような場合において、一緒に担持されている追加の任意選択のスパンが連続して出現することに終わるとき、ＬＣＳアルゴリズムは、それらの任意選択のスパン上でも同様に再帰的に実行されてもよい。

【００１０】

本明細書で説明されるさらなる態様は、正規表現生成器によって実行されるＬＣＳアルゴリズムが複数回実行されて、「正しい」正規表現（たとえば、すべての所与の陽性例と適切にマッチし、すべての所与の陰性例を適切に除外する正規表現）を生成してもよく、および／または最も望ましいもしくは最適な正規表現が選択され得る複数の正しい正規表現を生成してもよい、コンビナトリック探索に関する。いくつかの実施形態では、ＬＣＳアルゴリズムは一般に、正規表現を生成するために、入力例において右から左に実行され得る。しかしながら、比較の目的のために、および代替的な正規表現を見つけるために、ＬＣＳアルゴリズムは、入力例において逆方向に（たとえば左から右への方向で）別途実

行されてもよい。たとえば、ユーザ入力として受信された例示的なキャラクタシーケンスは、それらがＬＣＳアルゴリズムを通過する前に反転されてもよく、次いで、ＬＣＳアルゴリズムからの結果を、（元のテキストフラグメントを含んで）反転して戻してもよい。さらに、いくつかの実施形態では、ＬＣＳアルゴリズムは、正規表現生成器によって、複数回、通常のキャラクタシーケンス順序および逆の順序の両方で、行の始まりでの位置指定、行の終わりでの位置固定、行の始まりまたは終わりでの位置指定なしで、実行されてもよい。したがって、場合によっては、ＬＣＳアルゴリズムは、少なくともこれら６回実行されてもよく、最短の成功裡な正規表現が、これらの実行から選択されてもよい。

【図面の簡単な説明】

【００１１】

【図１】さまざまな実施形態が実現され得る、正規表現を生成するための例示的な分散システムの構成要素を示すブロック図である。

【図２】本明細書で説明する１つまたは複数の実施形態による、ユーザインターフェイスを介して受信された入力に基づいて正規表現を生成するための処理を示すフローチャートである。

【図３】本明細書で説明する１つまたは複数の実施形態による、正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して正規表現を生成するための処理を示すフローチャートである。

【図４】本明細書で説明する１つまたは複数の実施形態による、正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、２つのキャラクタシーケンス例に基づいて正規表現を生成するための例示的な図である。

【図５】本明細書で説明する１つまたは複数の実施形態による、より大きな正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して正規表現を生成するための処理を示すフローチャートである。

【図６】本明細書で説明する１つまたは複数の実施形態による、正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、５つのキャラクタシーケンス例に基づいて正規表現を生成するための例示的な図である。

【図７】本明細書で説明する１つまたは複数の実施形態による、より大きな正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムについて実行の順序を判断するための処理を示すフローチャートである。

【図８Ａ】本明細書に記載する１つまたは複数の実施形態による、より大きな正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムについて実行の順序を判断するために用いられる、全結合グラフを示す。

【図８Ｂ】本明細書に記載する１つまたは複数の実施形態による、より大きな正規表現コードのセット上で最長共通サブシーケンス（ＬＣＳ）アルゴリズムについて実行の順序を判断するために用いられる、全結合グラフの最小スパニングツリー表現を示す。

【図９】本明細書に記載する１つまたは複数の実施形態による、陽性キャラクタシーケンス例および陰性キャラクタシーケンス例に基づいて正規表現を生成するための処理を示すフローチャートである。

【図１０Ａ】本明細書に記載する１つまたは複数の実施形態による、陽性キャラクタシーケンス例に基づく正規表現の生成を示す例示的なユーザインターフェイス画面である。

【図１０Ｂ】本明細書に記載する１つまたは複数の実施形態による、陽性キャラクタシーケンス例および陰性キャラクタシーケンス例に基づく正規表現の生成を示す例示的なユーザインターフェイス画面である。

【図１１】本明細書で説明する１つまたは複数の実施形態による、ユーザインターフェイス内で受信されるユーザデータ選択に基づいて正規表現を生成するための処理を示すフローチャートである。

【図１２】本明細書で説明する１つまたは複数の実施形態による、ユーザインターフェイス内で受信されるユーザデータ選択を介して、正規表現を生成し、キャプチャグループに基づいてデータを抽出する処理を示すフローチャートである。

【図１３】本明細書に記載する１つまたは複数の実施形態によるテーブル形式データディスプレイを示す例示的なユーザインターフェイス画面である。
【図１４】本明細書に記載する１つまたは複数の実施形態による、テーブル形式ディスプレイからのデータの選択に基づく正規表現およびキャプチャグループの生成を示す例示的なユーザインターフェイス画面である。
【図１５】本明細書に記載する１つまたは複数の実施形態による、テーブル形式ディスプレイからのデータの選択に基づく正規表現およびキャプチャグループの生成を示す例示的なユーザインターフェイス画面である。
【図１６Ａ】本明細書に記載する１つまたは複数の実施形態による、テーブル形式ディスプレイからの陽性例および陰性例の選択に基づく正規表現の生成を示す例示的なユーザインターフェイス画面である。
【図１６Ｂ】本明細書に記載する１つまたは複数の実施形態による、テーブル形式ディスプレイからの陽性例および陰性例の選択に基づく正規表現の生成を示す例示的なユーザインターフェイス画面である。
【図１７】本明細書に記載する１つまたは複数の実施形態による、テーブル形式ディスプレイからのデータの選択に基づく正規表現およびキャプチャグループの生成を示す別の例示的なユーザインターフェイス画面である。
【図１８】本明細書で説明する１つまたは複数の実施形態による、最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、任意選択のスパンを含む、正規表現を生成するための処理を示すフローチャートである。
【図１９】本明細書で説明する１つまたは複数の実施形態による、最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、任意選択のスパンを含む、正規表現を生成するための例示的な図である。
【図２０】本明細書で説明する１つまたは複数の実施形態による、最長共通サブシーケンス（ＬＣＳ）アルゴリズムのコンビナトリックな実行に基づいて正規表現を生成するための処理を示すフローチャートである。
【図２１】本発明のさまざまな実施形態が実現され得る例示的な分散システムの構成要素を示すブロック図である。
【図２２】本発明の実施形態によって提供されるサービスがクラウドサービスとして提供され得るシステム環境の構成要素を示すブロック図である。
【図２３】本発明の実施形態が実現され得る例示的なコンピュータシステムを示すブロック図である。
【図２４】いくつかの例示的実施形態による、正規表現生成器を示す。
【図２５Ａ】いくつかの例示的実施形態による、分割コマンドを実現するためのユーザインターフェイスを示す。
【図２５Ｂ】いくつかの例示的実施形態による、分割コマンドを実現するためのユーザインターフェイスを示す。
【図２６】いくつかの例示的実施形態による、分割コマンドの結果をデータセットに表示するユーザインターフェイスを示す。
【図２７】いくつかの例示的な実施形態による、分割コマンドを実行するための方法のフローチャートを示す。
【図２８】いくつかの例示的実施形態による、削除コマンドを実現するためのユーザインターフェイスを示す。
【図２９】いくつかの例示的実施形態による、データセット上に削除コマンドの結果を表示するユーザインターフェイスを示す。
【図３０】いくつかの例示的な実施形態による、削除コマンドを実行するための方法のフローチャートを示す。
【図３１】いくつかの例示的実施形態による、難読化コマンドを実現するためのユーザインターフェイスを示す。
【図３２】いくつかの例示的実施形態による、データセット上に難読化コマンドの結果を

表示するユーザインターフェイスを示す。
【図３３】いくつかの例示的実施形態による、難読化コマンドを実行するための方法のフローチャートを示す。
【図３４】いくつかの例示的実施形態による、置換コマンドを実現するためのユーザインターフェイスを示す。
【図３５】いくつかの例示的実施形態による、置換コマンドを実現するためのユーザインターフェイスを示す。
【図３６】いくつかの例示的実施形態による、置換コマンドの結果をデータセットに表示するユーザインターフェイスを示す。
【図３７】いくつかの例示的な実施形態による、置換コマンドを実行するための方法のフローチャートを示す。
【図３８】いくつかの例示的実施形態による、行フィルタリングコマンドを実現するためのユーザインターフェイスを示す。
【図３９】いくつかの例示的実施形態による、行フィルタリングコマンドを実現するためのユーザインターフェイスを示す。
【図４０】いくつかの例示的実施形態による、データセット上に行フィルタリングコマンドの結果を表示するユーザインターフェイスを示す。
【図４１】いくつかの例示的な実施形態による、行フィルタリングコマンドを実行するための方法のフローチャートを示す。
【図４２】いくつかの例示的実施形態による、単一レベルモードでデータセットのビューを表示するユーザインターフェイスを示す。
【図４３】いくつかの例示的実施形態による、ネスト化された完全制御モードで強調表示されたデータを表示するユーザインターフェイスを示す。
【図４４】いくつかの例示的実施形態による、ネスト化された完全制御モードで強調表示されたデータを表示するユーザインターフェイスを示す。
【図４５】いくつかの例示的な実施形態による、例を提供するためのユーザインターフェイスを示す。
【図４６】いくつかの例示的実施形態による、更新された生成されたレゲックスを表示するユーザインターフェイスを示す。
【図４７】いくつかの例示的実施形態による、代替データ強調表示を表示するユーザインターフェイスを示す。
【図４８】いくつかの例示的実施形態による、更新された生成されたレゲックスを表示するユーザインターフェイスを示す。
【図４９】いくつかの例示的な実施形態による、多重強調表示を実行するための方法のフローチャートである。
【図５０】いくつかの例示的な実施形態による、陰性例を提供するための方法のフローチャートを示す。
【図５１】いくつかの例示的な実施形態による、陰性例からコンテキストを判断するための方法のフローチャートを示す。
【図５２】いくつかの例示的実施形態による、スパン強調表示整列に基づいて正規表現を生成するための図を示す。
【図５３】いくつかの例示的実施形態による、スパン強調表示整列を実行するための方法のフローチャートを示す。
【図５４】いくつかの例示的実施形態による、スパンを追跡する方法のフローチャートを示す。
【図５５】いくつかの例示的実施形態による、句読点スパンおよび記号スパンを表示するユーザインターフェイスを示す。
【発明を実施するための形態】
【００１２】
詳細な説明

以下の記載では、説明を目的として、本発明のさまざまな実施形態の完全な理解のために、多数の具体的な詳細が記載される。しかしながら、本発明の実施形態は、これらの具体的な詳細のいくつかを伴わずに実施され得ることが当業者には明白であろう。他の例では、周知の構造およびデバイスがブロック図の形で示される。

【００１３】

以下の説明は、例示的な実施形態のみを提供するものであり、本開示の範囲、適用可能性、または構成を限定することを意図したものではない。むしろ、例示的な実施形態の以下の説明は、例示的な実施形態を実施するための実施可能な説明を当業者に提供するであろう。特許請求の範囲に記載されている本発明の精神および範囲から逸脱することなく、要素の機能および構成にさまざまな変更を加えることができることを理解されたい。

【００１４】

具体的な詳細が、以下の説明において、実施の形態の十分な理解を与えるよう与えられる。しかしながら、当業者には、実施の形態はこれらの具体的な詳細なしに実施されてもよいことが理解される。たとえば、回路、システム、ネットワーク、プロセスおよび他のコンポーネントは、実施の形態を不必要な詳細で曖昧にしないように、ブロック図形式におけるコンポーネントとして示され得る。他の例では、周知の回路、プロセス、アルゴリズム、構造および技術は、実施の形態を曖昧にすることを回避するために、不必要な詳細なしに示され得る。

【００１５】

さらに、個々の実施の形態は、フローチャート、フロー図、データフロー図、構造図またはブロック図として示されるプロセスとして記載され得ることが注記される。フローチャートは動作をシーケンシャルなプロセスとして記載するかもしれないが、動作の多くは並列または同時に実行され得る。加えて、動作の順序は再構成されてもよい。プロセスは、その動作が完了されるときに終結されるが、図に含まれない追加のステップを含み得る。プロセスは、方法、関数、プロシージャ、サブルーチン、サブプログラムなどに対応し得る。プロセスが関数に対応する場合では、その終結は、その関数が呼出関数または主関数に戻ることに対応し得る。

【００１６】

「コンピュータ読取可能媒体」という語は、命令および／もしくはデータを記憶するか、含むか、または担持することができるポータブルまたは固定された記憶装置、光記憶装置、ならびにさまざまな他の媒体のような非一時的媒体を含むが、それらに限定はされない。コードセグメントまたはコンピュータ実行可能な命令は、プロシージャ、関数、サブプログラム、プログラム、ルーチン、サブルーチン、モジュール、ソフトウェアパッケージ、クラス、または、命令、データ構造もしくはプログラム文の任意の組合せを表し得る。コードセグメントは、情報、データ、引数、パラメータまたはメモリコンテンツを受け渡すおよび／または受け取ることによって、別のコードセグメントまたはハードウェア回路に結合されてもよい。情報、引数、パラメータ、データなどは、メモリ共有、メッセージ受渡し、トークン受渡し、ネットワーク伝送などを含む任意の好適な手段を介して渡されるか、転送されるか、または伝送されてもよい。

【００１７】

さらに、実施の形態は、ハードウェア、ソフトウェア、ファームウェア、ミドルウェア、マイクロコード、ハードウェア記述言語、またはそれらの任意の組み合わせによって実現されてもよい。ソフトウェア、ファームウェア、ミドルウェアまたはマイクロコードにおいて実現される場合には、必要なタスクを実行するプログラムコードまたはコードセグメントを機械読取可能媒体に記憶してもよい。プロセッサは必要なタスクを実行してもよい。

【００１８】

本明細書では、１つまたは複数の入力データ例内で識別されるパターンに対応する正規表現を生成するためのさまざまな技法（たとえば、方法、システム、１つまたは複数のプロセッサによって実行可能な複数の命令を記憶する非一時的なコンピュータ読取可能記憶

メモリなど）について説明する。特定の実施形態では、入力データの選択を受け取ることに応答して、入力データ内の１つまたは複数のパターンが自動的に識別され、識別されたパターンを表すよう、正規表現（または短く「レゲックス」）が自動的かつ効率的に生成され得る。そのようなパターンは、キャラクタのシーケンス（たとえば、文字、数字、空白、句読点、記号等のシーケンス）に基づくことができる。本明細書では、方法、システム、１つもしくは複数のプロセッサによって実行可能なプログラム、コード、または命令を記憶する非一時的コンピュータ読取可能記憶媒体などを含むさまざまな実施形態について説明する。

【００１９】

いくつかの実施形態では、正規表現は、キャラクタストリングにマッチするために、および／または入力として与えられるキャラクタストリングから情報を抽出するために、記号的なワイルドカードマッチング言語を使用して構成され得る。

【００２０】

【数１】

たとえば、第１の例示的な正規表現[A-Za-z]{3} \d?\d, \d\d\d\dは、ある日付（たとえばApril 3, 2018）にマッチしてもよく、第２の例示的な正規表現[A-Za-z]{3} \d?\d, (\d\d\d\d)を使用して、マッチする日付から年を抽出してもよい。

【００２１】

正規表現生成システムによって受信される入力データは、たとえば、１つもしくは複数の「陽性」データ例および／または１つもしくは複数の「陰性」データ例を含み得る。本明細書で使用される場合、陽性例は、入力として受信され、その入力に基づいて生成される正規表現によってマッチされることになるキャラクタシーケンスを指し得る。一方、陰性例は、その入力に基づいて生成される正規表現によってマッチされることにならない、入力されたキャラクタシーケンスを指し得る。

【００２２】

いくつかの技術的利点が、本明細書に記載されるさまざまな実施形態および例内で実現され得る。たとえば、本開示で説明するいくつかの技法は、正規表現生成処理の速度および効率を向上させ得る（たとえば、レゲックス解は、１秒未満で生成され得、ユーザインターフェイスは、対話型リアルタイム使用に適し得る）。本明細書で説明されるさまざまな技法はまた、判断論的であってもよく、トレーニングデータを必要としなくてもよく、いかなる初期正規表現入力も必要とせずに解を生成してもよく、完全に自動化されてもよい（たとえば、任意の人的介入を必要とする範囲内で正規表現を生成する）。さらに、本明細書で説明されるさまざまな技法は、効果的に処理され得るデータ入力のタイプに関して限定される必要はなく、そのような技法は、結果として得られる正規表現の人間による可読性を改善し得る。

【００２３】

本明細書で説明するいくつかの実施形態は、最長共通サブシーケンス（ＬＣＳ）アルゴリズムの１つまたは複数の実行を含む。ＬＣＳアルゴリズムは、ある状況においては、２つのテキストファイル間の差分を判断し示すように構成された差分エンジン（たとえば、Unix（登録商標）“diff”utilityの背後にあるエンジン）として使用され得る。いくつかの実施形態では、入力データ（たとえば、ストリングおよび他のキャラクタシーケンス）は、抽象的なトークンに変換され得、次いで、それらは、ＬＣＳアルゴリズムへの入力として提供され得る。そのような抽象的なトークンは、たとえば、正規表現キャラクタクラスを表す正規表現コード（たとえば、Loogleコードまたは他のキャラクタクラスコード）に基づくトークンであってもよい。そのようなコードのさまざまな異なる例が考えられ、本明細書では「正規表現コード」または「中間正規表現コード」（ＩＲＥＣ）と称され得る。たとえば、入力されたキャラクタシーケンス「May 3」は、ＩＲＥＣコード「LLLZN

」に変換されてもよく、その後、トークン化されたストリングは、他のトークン化されたストリングとともにＬＣＳアルゴリズムに供されてもよい。いくつかの実施形態では、入力されたキャラクタシーケンスが共通に有さないＩＲＥＣ（たとえば正規表現コード）が、最終的に生成された正規表現において、任意選択（たとえば任意選択のスパン）として現れてもよい。特定の実施形態では、正規表現コードは、https://www.regular-expressions.info/unicode.html#categoryで示されるユニコードカテゴリコードに基づくカテゴリコードであってもよく、またはhttp://unicode.org/reports/tr18/#General_Category_Propertyの一般的なカテゴリプロパティコードであってもよい。たとえば、コードLは文字を表してもよく、コードNは数字を表してもよく、コードZは空白を表してもよく、コードSは記号を表してもよく、コードPは句読点を表してもよい等である。

【００２４】

【数２】

たとえば、コードLは、ユニコード\p{L}に対応し、コードNは、ユニコード\p{N}に対応し得る。これは、ＬＣＳ出力から正規表現への１対１のマッピングが働くことを可能にし（たとえば、\pN\pN\pZ\pL\pLは「10 am」にマッチすることができる）、これは、人間による可読性の利点を提供し得る。

【００２５】

加えて、これらの異なるカテゴリは、互いに素、つまり相互に排他的であってもよい。すなわち、この例では、カテゴリL、N、Z、PおよびSは、カテゴリのメンバ間に重複がないように、互いに素にされてもよい。

【００２６】

さらなる技術的利点は、正規表現コード（たとえばカテゴリコード）、スパンなどの使用に基づく正規表現のより効率的な生成を含むさまざまな実施形態において実現され得る。そのようなコードを使用することによって、ＬＣＳアルゴリズムが入力ストリング内のキャラクタのすべてまたは実質的にすべてを異なるものとして首尾よく識別する場合には、計算資源を無駄にする必要がない。本明細書においてさまざまな実施形態によって提供されるさらなる技術的利点は、生成された正規表現の改善された可読性、ならびに陽性例および陰性例の両方を入力データとしてサポートすること、およびさまざまな有利なユーザインターフェイス特徴を提供すること（たとえば、ユーザが、抽出のために、より大きなキャラクタシーケンスまたはデータセル内のテキストフラグメントを強調表示することを可能にする）を含む。

【００２７】

Ｉ．全体の概要

本明細書に開示されるさまざまな実施形態は、正規表現の生成に関連する。いくつかの実施形態では、正規表現生成器として構成されるデータ処理システムは、正規表現コード（たとえばカテゴリコード）の異なるセットによって共有される最長共通サブシーケンス（ＬＣＳ）を識別することによって正規表現を生成することができる。正規表現コードの各セットは、ユーザインターフェイスを介して入力データとして受け取られるキャラクタのシーケンスから変換され得る。本明細書で説明する技術的利点のうち、入力データを中間コード（たとえば、正規表現コード、スパンなど）として抽象化することにより、非常に少ない入力データを使用して正規表現を効率的に生成することができる。

【００２８】

図１は、さまざまな実施形態が実現され得る、正規表現を生成するための例示的な分散システムの構成要素を示すブロック図である。この例に示されるように、クライアントデバイス１２０は、正規表現生成サーバ１１０（または正規表現生成器）と通信し、ユーザインターフェイスと対話してテーブル形式データを取り出し表示し、ユーザインターフェイスを介した入力データ（たとえば例）の選択に基づいて正規表現を生成することができる。いくつかの実施形態では、クライアントデバイス１２０は、クライアントウェブブラ

ウザ１２１および／またはクライアント側正規表現アプリケーション１２２（たとえば、サーバ１１０によって生成された正規表現を受信／消費するクライアント側アプリケーション）を介して正規表現生成器１１０と通信してもよい。正規表現生成器１１０内で、クライアントデバイス１２０からの要求は、ネットワークインターフェイスにおいてさまざまな通信ネットワークを介して受信され、ＲＥＳＴ　ＡＰＩ１１２などのアプリケーションプログラミングインターフェイス（ＡＰＩ）によって処理され得る。ユーザインターフェイスデータモデル生成器１１４コンポーネントは正規表現生成器１１０とともに、サーバ側プログラミングコンポーネントおよびロジックを提供して、本明細書で説明されるさまざまなユーザインターフェイス特徴を生成し、レンダリングすることができる。そのような特徴は、ユーザがデータリポジトリ１３０からテーブル形式データを取り出し、表示すること、入力データ例を選択して正規表現の生成を開始し、生成された正規表現に基づいてデータを修正および／または抽出することを可能にする機能を含み得る。この例では、正規表現生成器コンポーネント１１６は、入力キャラクタシーケンスを正規表現コードおよび／またはスパンに変換すること、入力データに対してアルゴリズム（たとえばＬＣＳアルゴリズム）を実行すること、および正規表現を生成／単純化することを含んで、正規表現を生成するように実現され得る。正規表現生成器１１６によって生成された正規表現は、ＲＥＳＴサービス１１２によってクライアントデバイス１２０に送信されてもよく、そこで、クライアントブラウザ１２１（または対応するクライアント側アプリケーションコンポーネント１２２）上のJavascriptコードは、次いで、ブラウザ内でレンダリングされたスプレッドシート列内のすべてのセルに対して正規表現を適用することができる。他の場合では、サーバ側でマッチするデータ／マッチしないデータを識別するために、サーバ側で別個の正規表現エンジンコンポーネントを実現して、生成された正規表現を、ユーザインターフェイス上に表示されるテーブル形式データおよび／またはデータリポジトリ１３０に格納された他のデータ内で比較してもよい。さまざまな実施形態において、マッチする／マッチしないデータは、ユーザインターフェイス内で自動的に選択（たとえば強調表示）されてもよく、抽出、修正、削除などのために選択されてもよい。正規表現の生成に基づいて、ユーザインターフェイスを介して抽出または修正される任意のデータは、１つまたは複数のデータリポジトリ１３０に記憶され得る。さらに、いくつかの実施形態では、生成された正規表現（および／またはＬＣＳアルゴリズムへの対応する入力）は、将来の検索および使用のために正規表現ライブラリ１３５に記憶されてもよい。いくつかの実施形態では、生成された正規表現は、実際に「ライブラリ」に格納される必要はなく、「変換スクリプト」に組み込まれてもよい。たとえば、ＥＴ．Ｓ．特許番号１０，２１０，２４６（すべての目的のために、ここに引用により援用する）においてより詳細に記載されるように、そのような変換スクリプトは、受信されたデータを変換するために１つまたは複数の処理ユニットによって実行可能であり得るプログラム、コード、または命令を含み得る。変換スクリプトの他の考えられ得る例は、「改名列」、「大文字列データ」、または「ファーストネームから性別を推測し性別を伴う新たな列を作成する」などを含み得る。

【００２９】

図２は、本明細書に記載される１つまたは複数の実施形態に係る、ユーザインターフェイスを介して受信される入力に基づいて正規表現を生成するための処理２００を示すフローチャートである。ステップ２０１において、正規表現生成器１１０は、正規表現生成器ユーザインターフェイスにアクセスし、ユーザインターフェイスを介して特定のデータを閲覧するための要求をクライアントデバイス１２０から受信することができる。ステップ２０１における要求は、ＲＥＳＴ　ＡＰＩ１１２、および／またはウェブサーバ、認証サーバなどを介して受信されてもよく、ユーザの要求はパーズおよび認証されてもよい。たとえば、ビジネスまたは組織内のユーザは、取引データ、顧客データ、実績データ、予測データ、および／または組織のデータリポジトリ１３０に記憶され得るデータの任意の他のカテゴリを解析および／または修正するために、正規表現生成器１１０にアクセスすることができる。ステップ２０２において、正規表現生成器１１０は、選択された入力デー

タに基づく正規表現の生成をサポートするユーザインターフェイスを介して、要求されたデータを取り出し、表示することができる。そのようなユーザインターフェイスのさまざまな実施形態および例は、以下で詳細に説明される。

【００３０】

ステップ２０３において、ユーザは、正規表現生成器１１０によって提供されるユーザインターフェイスに表示されたデータから１つまたは複数の入力キャラクタシーケンスを選択し得る。いくつかの実施形態では、データは、特定のデータタイプおよび／またはデータのカテゴリを有するラベル付き列を含んで、ユーザインターフェイス内にテーブル形式で表示されてもよい。そのような場合、ステップ２０３における入力データの選択は、ユーザがデータセルを選択すること、またはデータセル内の個々のテキストフラグメントを選択する（たとえば強調表示すること）ことに対応し得る。しかしながら、他の実施形態では、正規表現生成器１１０は、ユーザインターフェイスを介して半構造化および非構造化データの検索ならびに表示をサポートしてもよく、ユーザは、半構造化または非構造化データからキャラクタシーケンスを選択することによって正規表現生成のための入力データを選択してもよい。後述する例で説明されるように、ユーザが表示されたテーブル形式データから入力キャラクタシーケンスを選択することは、単なる使用ケースの一例である。他の例では、ユーザ（たとえば、おそらく、Linux（登録商標）コマンドラインツールgrep、sed、またはawkなどのために正規表現を構築することを試みるソフトウェア開発者またはパワーユーザ）は、スプレッドシートから例を拾うのではなく、例において初めからタイプしてもよい。

【００３１】

ステップ２０４において、正規表現生成器１１０は、ステップ２０３においてユーザにより選択された入力データに基づいて、１つまたは複数の正規表現を生成してもよい。ステップ２０５において、正規表現生成器１１０は、たとえば、生成された正規表現を表示するように、および／または表示されたデータ内のマッチする／マッチしないデータを強調表示するように、ユーザインターフェイスを更新してもよい。いくつかの実施形態では任意選択であり得るステップ２０６において、ユーザインターフェイスは、ユーザが生成された正規表現に基づいて基礎となるデータを修正することを可能にする機能をサポートしてもよい。たとえば、ユーザインターフェイスは、ユーザが、テーブル形式データから、特定のデータフィールドを、それらフィールドが正規表現とマッチするか否かに基づいて、フィルタリング、修正、削除、または抽出することを可能にする特徴をサポートしてもよい。データをフィルタリングまたは修正することは、リポジトリ１３０に格納された基礎となるデータを修正することを含むことができ、場合によっては、抽出したデータを新たな列および／または新たなテーブルとしてリポジトリ１３０に格納することができる。

【００３２】

これらのステップは、正規表現生成器１１０のユーザインターフェイスとの例示的なユーザ対話の一般的かつハイレベルの概要を示すが、他の実施形態では、さまざまな追加の特徴および機能性をサポートしてもよい。たとえば、いくつかの実施形態では、正規表現コード（またはカテゴリコード）は、コードの最小発生数に関連付けられ得る。追加的または代替的に、正規表現コードは、コードの最大発生数に関連付けられてもよい。一例として、正規表現コードのセットは、ＬＣＳの特定の部分がある文字を少なくとも０回、および最大でも１回のいずれかで含むことを示すコードL〈0,1〉を含むことができる。

【００３３】

さらに、いくつかの実施形態では、入力データは、３つ以上のキャラクタシーケンスを含み得る。そのような実施形態では、さまざまな技法を用いて、３つ以上のキャラクタシーケンスに対してＬＣＳアルゴリズムを実行するための順序を判断し、結果として生じる正規表現が充分に機能する態様で生成され得るようにして、３つ以上の入力キャラクタシーケンスによって引き起こされるランタイムの指数関数的な増加を回避することができる。なお、正規表現生成器１１０は、かわりに、一度に２つのキャラクタシーケンス上でＬＣＳアルゴリズムを実行し、グラフに基づいてキャラクタシーケンスのペアを選択するた

めの順序を判断してもよい。たとえば、全結合グラフは、ＬＣＳアルゴリズムの第１の実行（たとえばＬＣＳ１）は、シーケンス１およびシーケンス３に対して実行されるべきであることを示し、次いで、ＬＣＳアルゴリズムの第２の実行（たとえばＬＣＳ２）は、ＬＣＳ１およびシーケンス２に対して実行されるべきであることなどを示し得る。グラフは、全結合グラフであってもよく、ノードは、キャラクタシーケンスを表し、エッジはノードを接続して、接続されたノードによって共有されるＬＣＳの長さを表す。グラフ内の各ノードは、グラフ内の他のすべてのノードに接続されてもよく、キャラクタシーケンスを選択する順序は、グラフについて最小スパニングツリーの深さ優先トラバースを実行することによって判断されてもよい。

【００３４】

さらなる実施形態では、入力データは、いくつかの異なる方法でユーザインターフェイスを介して提供されてもよい。たとえば、入力データは、キャラクタのセットの第２のユーザ選択内の、１つまたは複数のキャラクタの第１のユーザ選択を示してもよい。たとえば、ユーザは、以前に強調表示されたキャラクタのセット内のうちのあるキャラクタを強調表示してもよい。したがって、第２のユーザ選択は、第１のユーザ選択のためのコンテキストを提供し得、これは、入力データが、より高い特異性で正規表現生成器１１０に提供されることを可能にし得る。いくつかの実施形態では、正規表現生成器１１０は、各ユーザ選択に応答して、ほぼリアルタイムで正規表現を生成し、表示することができる。たとえば、ユーザがキャラクタの第１の範囲を強調表示する場合、正規表現生成器１１０は、キャラクタの第１の範囲を表す正規表現を表示してもよい。そして、ユーザが、キャラクタの第１の範囲内の、キャラクタの第２の範囲を強調表示すると、正規表現生成器１１０は、表示される正規表現を更新してもよい。

【００３５】

さらに、いくつかの実施形態では、正規表現生成器１１０は、陽性および陰性の両方の例を含む入力に基づいて正規表現を生成することができる。上述したように、陽性例は、正規表現に包含されるべきキャラクタのシーケンスを指し得、陰性例は、正規表現に包含されるべきでないキャラクタのシーケンスを指し得る。そのような場合、正規表現生成器１１０は、特定の位置において、陽性例を陰性例から区別する、１つまたは複数のキャラクタからなる最短サブシーケンスを識別することができる。次いで、最短サブシーケンスは、正規表現生成器１１０によって生成された正規表現内でハードコード化され得る。さまざまな例では、最短サブシーケンスは、プレフィックス／サフィックス部分、または陰性例内のミッドスパンに含まれ得る。

【００３６】

特定の実施形態による、正規表現を自動的に生成するさらなる例を以下に記載する。これらの例は、図２の一般的技法のさまざまな具体的な考えられ得る実現例に対応し得、それぞれのシステムの１つまたは複数の処理ユニット（たとえば、プロセッサ、コア）によって実行されるソフトウェア（たとえば、コード、命令、プログラムなど）、ハードウェア、またはそれらの組み合わせにおいて実現され得る。ソフトウェアは、非一時的記憶媒体上に（たとえばメモリデバイス上に）記憶され得る。以下で説明するさらなる例は、例示的かつ非限定的であることを意図している。これらの例は、特定の順番または順序で生じるさまざまな処理ステップを示すが、これは限定することを意図するものではない。いくつかの代替実施形態では、ステップは、ある異なる順序で実行されてもよく、またはいくつかのステップは、並行して実行されてもよい。

【００３７】

いくつかの例では、ユーザインターフェイス（たとえばステップ２０３）を介して受信されるユーザ入力は、正規表現出力によってマッチされる１つまたは複数の「陽性例」と、正規表現出力によってマッチされない０またはそれより多い「陰性例」とを含み得る。任意選択で、陽性例のうちの１つまたは複数を強調表示して、キャラクタの特定の範囲（またはサブシーケンス）を選択することができる。場合によっては、ステップ２０４において、ユーザインターフェイスを介して受信された陽性例は、正規表現コード（たとえば

、ユニコードカテゴリコードのようなキャラクタカテゴリコード）のスパンに変換され得る。各陽性例に対して、スパンのシーケンスが生成され得る。いくつかの実施形態では、各頂点が、スパンのシーケンスのうちの１つに対応し、エッジ重みが、エッジのエンドポイントに対応するスパンのそれらの２つのシーケンス上で実行されるＬＣＳアルゴリズムからの出力の長さに等しい、グラフが生成されてもよい。グラフについて、最小スパニングツリーを判断することができる。たとえば、いくつかの実施形態では、Ｐｒｉｍのアルゴリズムを用いて、最小スパニングツリーを得てもよい。深さ優先トラバースを最小スパニングツリー上で実行してトラバース順序を判断してもよく、その後、ＬＣＳアルゴリズムを、トラバースの最初の２つの要素上で実行してもよい。次いで、１つずつ、トラバースの各追加要素が、前のＬＣＳ反復の出力および次の現在のトラバース要素に対してＬＣＳアルゴリズムを再び実行することによって、現在のＬＣＳ出力に順番にマージされてもよい。次いで、スパンのシーケンスであり得る、ＬＣＳアルゴリズムの最終出力が、正規表現に変換されてもよい。この変換は、いくつかの実施形態では、１対１変換であり得るが、本明細書で説明される特定の任意選択的な実施形態は、１対１変換に対応しないかもしれない。最後に、得られた正規表現は、ステップ２０３でユーザインターフェイスを介して受信されたすべての陽性例および陰性例に対してテストされてもよい。テストのいずれかが失敗した場合、すべての陽性例および失敗した任意の陰性例を使用して、上述の処理を繰り返してもよい。

【００３８】

ＩＩ．正規表現コード上で最長共通サブシーケンスアルゴリズムを用いた正規表現生成

上述したように、本明細書で説明するいくつかの態様は、入力データに対応する正規表現コードの異なるセットによって共有される最長共通サブシーケンス（ＬＣＳ）の計算に基づく正規表現の生成に関する。

【００３９】

図３は、本明細書で説明する１つまたは複数の実施形態による、正規表現コードのセットに対してＬＣＳアルゴリズムを使用して正規表現を生成するための処理３００を示すフローチャートである。ステップ３０１において、正規表現生成器１１０は、入力データとして１つまたは複数のキャラクタシーケンスを受け取り得る。上述したように、いくつかの例では、入力データは、ユーザインターフェイスに表示されたテーブル形式データ内から選択された陽性例データに対応し得るが、いくつかの実施形態ではユーザインターフェイスは任意選択であり、入力データは、さまざまな例では任意の他の通信チャネル（たとえば非ユーザインターフェイス）を介して受信される任意のキャラクタシーケンスに対応し得ることを理解されたい。

【００４０】

ステップ３０２において、ステップ３０１において受け取られた各キャラクタシーケンスは、対応する正規表現コードに変換され得る。さまざまな実施形態において、正規表現コードは、Loogleコード、ユニコードカテゴリコード、または正規表現キャラクタクラスを表す任意の他のキャラクタクラスコードであり得る。たとえば、ある入力キャラクタシーケンス「May 3」をLoogleコード「LLLZN」に変換してもよい。いくつかの実施形態では、正規表現コードは、https://www.regular-expressions.info/unicode.html#categoryに示されるユニコードカテゴリコードに基づくカテゴリコードであり得る。たとえば、コードLは文字を表してもよく、コードNは数字を表してもよく、コードZは空白を表してもよく、コードSは記号を表してもよく、コードPは句読点を表してもよい、等である。

【００４１】

【数３】

たとえば、コードLは、ユニコード\p{L}に対応し、コードNは、ユニコード\p{N}に対応し得る。

【００４２】

ステップ３０３では、ステップ３０２で生成された正規表現コードのセットの中から最

長共通サブシーケンスを判断することができる。いくつかの実施形態では、ＬＣＳアルゴリズムは、入力として正規表現コードの２つのセットを使用して実行され得る。ＬＣＳアルゴリズム（たとえば、処理の方向、位置指定、空白のプッシュ、低濃度スパンの合体（coalescing）、共通トークン上の整列など）の実行のさまざまな異なる特性が、異なる実施形態において使用され得る。ステップ３０４では、ＬＣＳアルゴリズムの出力に基づいて正規表現を生成することができる。場合によっては、ステップ３０４は、ＬＣＳアルゴリズムの出力を正規表現コードで捕捉すること、および正規表現コードを正規表現に変換することを含み得る。ステップ３０５において、正規表現は、たとえば、ユーザインターフェイスを介してユーザに正規表現を表示することによって、単純化され、出力されてもよい。

【００４３】

図４は、２つのキャラクタシーケンス例に基づき、正規表現コードのセットに対して最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、正規表現を生成するための例示的な図である。したがって、図４は、図３において上述した処理を適用する例を示している。図４に示すように、本例の正規表現は、２つの入力ストリング「iPhone 5」および「iPhone X」に基づいて生成される。この例における各シーケンスは、正規表現コードのそれぞれのセットに変換され得る。したがって、iPhone 5は「LLLLLLZN」に変換されてもよく、iPhone Xは「LLLLLLZL」に変換されてもよい。図４に示されるように、これらのカテゴリコードは、次いで、ＬＣＳアルゴリズムへの入力として提供され、ＬＣＳアルゴリズムは、ＩＲＥＣ（またはカテゴリコード）の両方のセットは６つのLおよび１つのZを含む、と判断する。ＬＣＳから除外されるZカテゴリコードは、任意選択および／または代替として表され得る。

【００４４】

【数４】

したがって、両方のキャラクタシーケンスを包含する正規表現は、\pL{6}\pZ\pN?\pL?のように表され得る。この例において、正規表現はユニコードカテゴリコード（たとえば、文字については\pL、空白については\pZ、数字については\pN）を含む。

【００４５】

数字6を含む中括弧は、文字の６つのインスタンスを示し、疑問符は、最後の数字／文字が任意選択であることを示す。最後に、正規表現生成器によって単純化処理を実行することができ、その間に、正規表現は、共通テキストフラグメント「iPhone」を最終正規表現に挿入し戻すことによって単純化され、正規表現のより広い

【００４６】

【数５】

「\pL{6}\」

【００４７】

部分を置き換える。

この例に示されるように、正規表現生成器１１０によって受け取られた入力ストリングは、正規表現ブロードカテゴリ（これは、「カテゴリコード」とも呼ばれ得る）を表す「正規表現コード」に変換されてもよく、ＬＣＳアルゴリズムは、それらの正規表現コード上で実行されてもよい。いくつかの実施形態では、正規表現コードのために、ユニコードカテゴリコードを使用し得る。たとえば、入力テキストストリングは、レゲックスユニコードブロードカテゴリ

【００４８】

【数６】

（たとえば、文字については\pL、句読点については\pP等である。）

【００４９】
を表すコードに変換され得る。図３および図４によって示されるこのアプローチは、間接的アプローチと称され得る。しかしながら、他の実施形態では、ＬＣＳアルゴリズムが入力として受け取られたキャラクタシーケンスに対して直接実行される直接的アプローチが使用されてもよい。
【００５０】
いくつかの実施形態では、間接的アプローチは、大量のトレーニングデータを必要としないという点で、追加の技術的利点を提供し得、比較的より少ない数の入力例で有効な正規表現を生成し得る。これは、間接的なアプローチが、正規表現生成における不確実性を低減するために、および潜在的な偽陽性および偽陰性を除去するために、ヒューリスティックを使用するためである。たとえば、入力ストリング「May 3」および「Apr 11」に基づいて正規表現を生成する際に、直接的アプローチは、日付パターンにマッチする有効な正規表現を生成するために月毎に少なくとも１つの例を必要とし得る。それら２つの例のみに依拠して、直接的アプローチは、「[AM][ap][yr] [13]1?」のレゲックスを生成してもよい。対照的に、間接的アプローチは、ユニコードブロードカテゴリに基づいて、
【００５１】
【数７】
「\pL{3} \d{1,2}」

【００５２】
の、より効果的な正規表現を生成してもよい。加えて、上述のように、本明細書で説明される技術的利点の１つは、場合によっては単一の例からさえ、非常にわずかな入力データを使用して正規表現を効率的に生成することを含む。たとえば、単一の例「am」からの正規表現の生成に関して、あるヒューリスティックは、正規表現のために「am」を生成するか
【００５３】
【数８】
「\pL\pL」

【００５４】
を生成するかを判断することができる。どちらも、おそらくは正しいが、プログラムされたヒューリスティックは、最適な正規表現を生成する方法（たとえば、それが「pm」にもマッチすべきか否か）を判断するために、ユーザ選好および／または基準を実現し得る。
【００５５】
加えて、間接的アプローチは、さらに、生成された正規表現
【００５６】
【数９】
「\pL{3} \d{1,2}」～「[A-Za-z]{3} \d{1,2}」

【００５７】
を単純化して、それをより人間によって可読なものにすることができる。これは、いくつかの実施形態において、たとえば、正規表現のためのユニコード表現に精通していない可能性がある高度な知識のない正規表現ユーザに出力する場合に有益であり得る。
【００５８】

さらに、いくつかの実施形態では、ＬＣＳアルゴリズムを実行する際に各キャラクタを独立して扱う代わりに、シーケンシャルかつ等しい正規表現コードが、スパンデータ構造（スパンとも称され得る）に変換され得る。場合によっては、スパンは、単一の正規表現コード（たとえばユニコードブロードカテゴリコード）の表現を、繰り返しカウント範囲（たとえば最小数および／または最大数）とともに含み得る。正規表現コードからスパンへの変換は、交代（オルタネーション）（たとえば分離）を認識するなど、以下に説明されるいくつかのさまざまな追加の特徴を容易にすることができ、また、生成された正規表現をさらに単純化するために、隣接する任意選択のスパンのマージを容易にすることもできる。

【００５９】

上述したように、ＬＣＳアルゴリズムは、図４のストリング「iPhone」のような、最終的な正規表現に潜在的に挿入して戻すことができる、入力キャラクタシーケンス内の基礎となるテキストフラグメントを記憶し、保持するように構成することができる。そのスパンに割り当てられたカテゴリコードを元々もたらしたテキストフラグメントを追跡することによって、そのような実施形態は、リテラルテキスト（たとえばamおよびpm）が、生成された正規表現に直接含まれることを可能にし、偽陽性を低減し、正規表現出力をより人間により可読なものにすることができる。

【００６０】

いくつかの実施形態では、あるヒューリスティックを用いて、広くマッチする正規表現コードとは対照的に、生成された正規表現にいつストリングリテラルを出力するべきかを判断することができる。上述したように、場合によっては、正規表現がストリングフラグメントに対して正確なマッチを必要とすることが望ましい場合がある。したがって、いくつかの実施形態では、あるヒューリスティックを用いて、正確なストリングフラグメントが正規表現に出力されるかどうかを判断してもよい。例えば、あるヒューリスティックは、特定のスパンについて、今までにたった１つの特定のストリングフラグメント（例えば、"pm"）に遭遇した場合、そしてそのスパンについて２つ以上の例が存在する場合、その正確なストリングフラグメントが正規表現に出力される、と判断してもよい。句読点スパンの場合、ヒューリスティックは、（例えば、句読点は変化する可能性がより低いという仮定に基づいて）そのスパンについて閾値を１回だけの発生に低下させてもよい。例えば、

【００６１】

【数１０】

記号 \pS および句読点 \pP

【００６２】

の両方について、正規表現内にリテラルストリングを出力するために、閾値は、１回だけの発生まで下げられてもよい。ヒューリスティックは、正確なストリングフラグメントを正規表現に出力するのに必要な例の閾値数に基づいて、ある程度制限的になるように修正および／または調整されてもよい。ヒューリスティックが（リテラルストリングフラグメントをあまりに容易に正規表現に出力することによって）過剰に制限的であることにおいて誤る場合には、ユーザは、追加の陽性例を導入することによって補ってもよいことに留意されたい。同様に、ヒューリスティックエラーが（あまりにしばしば広範な正規表現コードを出力することによって）過剰に許容的であることにおいて誤る場合には、ユーザは、追加の陰性例を導入することによって補ってもよい。ヒューリスティックは、以前のユーザインタラクションおよび実験に基づいて、適切な量の制限を伴って最初にプログラムされ（およびその後調整され）ることができる。

【００６３】

図４を参照して上述したように、単純化処理は、正規表現生成器１１０によって実行されてもよく、その間、正規表現は、様々な技法を用いて単純化されてもよい。場合によっては、単純化は、（例えば、中括弧を用いて）長形正規表現繰り返しコードを正規表現短

手コード（例えば、*, +,および?）に置換することを含み得る。例えば、スパンが、例えば、０と３（最小と最大）との間の繰り返しを表すストリングフラグメント例から来る場合、正規表現生成器１１０は、偽陰性を防止し、判読性を改善するために、正規表現に{0,3}の代わりに*を出力することによって、表現を単純化することができる。さらに、{a,b}構文を頻繁に用いる代わりに、正規表現生成器１１０は、代わりに、a〉2またはb〉4の場合に、+構文を用いてもよい。これは、生成された正規表現をより判読性にし、偽陰性を回避する可能性を高める。

【００６４】

場合によっては、予め定義されたキャラクタ範囲のバンクが用いられてもよい。例えば、文字に対する正規表現Unicodeキャラクタカテゴリは

【００６５】

【数１１】

\pL

【００６６】

であり、それは、[A-Z]などの、より古いキャラクタ範囲と比較して、周知でないかまたは容易に判読可能でないかもしれない。したがって、英語入力の場合、正規表現生成器１１０は、[A-Z], [a-z], および [A-Za-z]を

【００６７】

【数１２】

\pL

【００６８】

に置換することを、所与の例がそれら置換物の１つと上手くマッチし得る場合には、行ってもよい。

【００６９】

さらに、正規表現生成器１１０は、いくつかの実施形態では、正規表現における空白に関して最大許容度までデフォルトするよう構成され得る。本明細書で説明されるように、正規表現生成器１１０は、例えば、場合によっては、広範なマッチするコードよりもリテラルストリングフラグメントを好み、中括弧内の繰り返し出力に最小および最大境界を適用することによって、より特定の正規表現を生成するよう構成され得る。しかしながら、空白に関しては、正規表現生成器１１０は、いくつかの実施形態では、最大許容度までデフォルトするよう構成されてもよい。すなわち、任意の空白（例えば、単語間に現れる１つの空白であっても）について、正規表現生成器１１０は、それを正規表現カテゴリコード

【００７０】

【数１３】

\s

【００７１】

として維持するよう構成されてもよく、繰り返しを、１つまたは複数を意味する+であるよう指定してもよい。したがって、そのような実施形態は、余分の空白、または正確にマッチされることができる（例えば、完全に構造化されていないテキストにおけるような）タブおよび復帰さえも、予想し、マッチし得る。

【００７２】

場合によっては、生成された正規表現内の繰り返しは、文字通り繰り返す正規表現コードに展開され得る。

【００７３】

【数１４】

例えば、ある年に対して、\d{4}の代わりに、規則表現生成器１１０は、判読性を容易にするために、およびあるユーザは規則表現繰り返し構文に精通していない可能性があるため、\d{4}を\d\d\d\dに展開するよう構成されてもよい。

【００７４】

例えば、正規表現生成器１１０は、Unicodeカテゴリコードの長さが２つのキャラクタ（例えば

【００７５】

【数１５】

\d

【００７６】

）であり、繰り返し数が４以下の場合、またはUnicodeカテゴリコードの長さが３つのキャラクタ（例えば

【００７７】

【数１６】

\pP

【００７８】

）であり、繰り返し数が３以下の場合、またはUnicodeカテゴリコードの長さが３つのキャラクタより長く、繰り返し数が２以下である場合、繰り返しを展開してもよい。さらに、いくつかの実施形態では、リテラルキャラクタを出力する必要がある場合（例えば、正規表現「特殊文字」）、正規表現生成器１１０は、それを逃れるよう構成されてもよい。

【００７９】

様々な追加の実施形態では、正規表現生成器１１０は、様々なタイプの対になった括弧（たとえば、{および}）を検出し、否定されたキャラクタクラス（たとえば、[^}]*）を有する正規表現を生成するよう構成され得る。例えば、ユーザによって強調表示された（または別様に選択された）テキスト内、またはシステムへの陽性／陰性例他の入力内では、開いた括弧または閉じた括弧が検出され、対応する括弧とマッチされるかまたはペアリングされてもよい。いくつかの例では、

【００８０】

【数１７】

“(“->”)”, “[“->”]”, “{“->”}”, “<“->”>”, ““->””, ““\”->“\””, “\u0060”->“\u00b4”, “\u2018”->“\u2019”, “\u201c”->“\u201d”, “\u00ab"->“\u00bb”, “\u3008”->“\u3009”

【００８１】

のうちのいくつかまたはすべてを含む複数の異なるタイプの括弧が入力テキスト内で検出されてもよい。したがって、一例として、ＨＴＭＬタグの場合、〈[a-z]+(?: [a-z]+=[a-z]+〉のようなもの代わりに〈[^〉]*〉を生成してもよい。様々な種類の括弧を検出およびペアリングするとき、正規表現生成器１１０は、括弧内にネスト化された括弧を適切に検出およびペアリングするよう構成されてもよいし、入力の強調表示された（またはそうでなければ、ユーザ選択された）部分と重なる括弧を無視するよう構成されてもよい。また、場合によっては、正規表現生成器１１０は、コンテンツが一意である場合に括弧間に新たな否定されたキャラクタクラスを用いないかもしれない。すなわち、コンテンツが常にすべて同じである場合（例えば、ＨＴＭＬの例を続けると、識別されるすべてのタグが〈h1〉である場合）、リテラルは、否定された大括弧キャラクタクラスの代わりに、生成された正規表現に出力されてもよい（例えば、〈[^〉]*〉の代わりに〈h1〉だけの出力）。

【００８２】

例示的な実施形態では、強調表示がちょうど括弧によって包含されるとき、括弧内に隠すこと（collapsing）ことが許容されてもよい。これにより、挿入句的注釈の抽出が可能になる。例えば、(Windows NT)からWindows NTを抽出するためには、生成されたレゲックスは、

【００８３】

【数１８】

\(([A-Za-z]+ [A-Z][A-Z])\)の代わりに\(([^)]*)\)であり、

【００８４】

それは、より一般的であり、もたらされる偽陰性がより少ないであろう。

ＩＩＩ．正規表現コードの組み合わせ上で最長共通サブシーケンスアルゴリズムを用いた正規表現生成

本明細書で説明するさらなる態様は、３つ以上のストリング（たとえば３つ以上の別個のキャラクタシーケンス）を含む入力データに基づく正規表現の生成に関する。３つ以上のストリングが入力データとして識別される場合、正規表現生成器１１０は、ＬＣＳアルゴリズム実行のシーケンスに対して最適な順序が判断される性能最適化特徴を使用し得る。以下で説明するように、３つ以上のストリングに関する性能最適化機能は、各ストリングに対応する頂点と、各ストリングと他のすべてのストリングとの間のＬＣＳ出力のサイズに基づき得るエッジ長さ／重みとでグラフを構築することを含み得る。次いで、それらのエッジ重みを使用して最小スパニングツリーが導出され得、入力ストリングの順序を判断するために、深さ優先トラバースが実行され得る。最後に、判断された入力ストリングの順序を使用して、一連のＬＣＳアルゴリズムが行われてもよい。

【００８５】

図５は、正規表現コードの、より大きなセット（たとえば３つ以上のキャラクタシーケンス）に対して、最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、正規表現を生成するための処理５００を示すフローチャートである。したがって、この例におけるステップ５０２～５０５は、図３で上述したステップ３０３に対応し得る。しかしながら、この例は、３つ以上の入力キャラクタシーケンスに基づいて正規表現を生成することに関するため、ＬＣＳアルゴリズムは複数回実行されてもよい。たとえば、３つ以上の入力ストリングのランタイムの指数関数的な増加を避けるために、ＬＣＳアルゴリズムは複数回実行されてもよく、各実行は２つの入力ストリングだけに対して実行される。たとえば、正規表現生成器１１０は、２つのストリング（たとえば２つの入力キャラクタシーケンスまたは２つの変換された正規表現コード）に対してＬＣＳアルゴリズムの最初の実行を実行してもよく、次いで、第１のＬＣＳアルゴリズムの出力および第３のストリングに対してＬＣＳアルゴリズムの第２の実行を実行してもよく、次いで、第２のＬＣＳアルゴリズムの出力および第４のストリングに対してＬＣＳアルゴリズムの第３の実行を実行してもよい等となる。

【００８６】

そのような実施形態の性能を改善および／または最適化するために、ＬＣＳアルゴリズムのシーケンスを実行する入力ストリング（たとえば入力キャラクタシーケンスまたは正規表現コード）の最適な順序を判断することが望ましい場合がある。たとえば、入力ストリングを取り込むための良好な順序は、任意選択のスパンの数を最小にすることなどによって、生成された正規表現の可読性に影響を及ぼし得る。生成されたレゲックスを簡潔に保つために、現在のレゲックスにＬＣＳされる追加のストリングは、現在のレゲックス（既に見られたストリングをＬＣＳしたことからの中間結果）に既にいくらか類似していることが好ましい。

【００８７】

これにより、ステップ５０１では、複数（たとえば３つ以上）の入力キャラクタシーケンスが正規表現コードに変換される。ステップ５０２では、ＬＣＳアルゴリズムを使用し

て正規表現コードを処理する順序を判断する。ステップ５０２における順序の判断については、図７を参照して後述する。ステップ５０３において、判断された順序における最初の２つの正規表現コードが（ステップ５０３の最初の反復のために）選択されるか、または判断された順序における次の正規表現コードが（ステップ５０３の後続の反復のために）選択される。ステップ５０４において、ＬＣＳアルゴリズムは、正規表現コードのフォーマットに対応する２つの入力ストリングに対して実行される。ステップ５０４の第１の反復では、ＬＣＳアルゴリズムは、判断された順序における最初の２つの正規表現コードに対して実行され、ステップ５０４の後続の反復では、ＬＣＳアルゴリズムは、判断された順序における次の正規表現コードおよび前のＬＣＳアルゴリズムの出力（同じ正規表現コードのフォーマットであってもよい）に対して実行される。ステップ５０５において、正規表現生成器１１０は、ＬＣＳアルゴリズムへの入力としてまだ提供されていない、判断された順序における追加の正規表現コードがあるか否かを判断する。判断された順序におけるそのような追加の正規表現コードがある場合、処理は、ＬＣＳアルゴリズムの別の実行のためにステップ５０３に戻る。そうでない場合には、ステップ５０６において、ＬＣＳアルゴリズムの最後の実行の出力に基づいて、正規表現が生成される。

【００８８】

図６は、５つの入力キャラクタシーケンス例に基づいて正規表現を生成するための例示的な図である。この例では、各入力されたキャラクタシーケンスを正規表現コードに変換し、次いで、判断された正規表現コードの順序に基づいてＬＣＳアルゴリズムを繰り返し実行する。したがって、図６は、図５において上述した処理を適用する一例を示す。この例では、５つの正規表現コードについて判断された順序は、コード＃１～コード＃５であり、各コードは、判断された順序でＬＣＳアルゴリズムに入力されて、正規表現出力が生成される。最終の正規表現出力（Ｒｅｇ　Ｅｘ＃４）は、入力キャラクタシーケンスの５つすべてに基づいて生成された最終正規表現に対応する。

【００８９】

図７は、正規表現コードの、より大きいセット（たとえば３つ以上）上における最長共通サブシーケンス（ＬＣＳ）アルゴリズムについての実行順序を判断するための処理を示すフローチャートである。したがって、この例に示されるように、ステップ７０１～７０４は、上述のステップ５０２における順序判断に対応し得る。ステップ７０１において、ＬＣＳアルゴリズムは、入力データに対応する正規表現コードの各他とは異なるペア上で実行されてもよく、得られた出力ＬＣＳは、実行ごとに記憶されてもよい。したがって、ｋ個の入力データについて、これは、ＬＣＳアルゴリズムを通して実行されるストリングのすべての（ｋ（ｋ－１））／２個の可能なペア形成、またはいくつかの実施形態ではｋ（ｋ－１）を表し得る。たとえば、ｋ＝３の入力キャラクタシーケンスが受け取られた場合、ＬＣＳアルゴリズムは、ステップ７０１において３回実行されてもよく；ｋ＝４の入力キャラクタシーケンスが受け取られた場合、ＬＣＳアルゴリズムは、ステップ７０１において６回実行されてもよく；ｋ＝５の入力キャラクタシーケンスが受け取られた場合、ＬＣＳアルゴリズムは、ステップ７０１において１０回実行されてもよい等となる。ステップ７０２において、全結合グラフは、（ｋ（ｋ－１））／２個のエッジのエッジ重みが２つのノード間の未処理ＬＣＳ出力の長さでストリングを表すｋ個のノードから構築され得る。ステップ７０３では、ステップ７０２における全結合グラフから最小スパニングツリーを導出し得る。ステップ７０４では、深さ優先トラバースが最小スパニングツリー上で実行され得る。このトラバースの出力は、正規表現コードがＬＣＳアルゴリズム実行のシーケンスに入力される順序に対応し得る。

【００９０】

図８Ａおよび図８Ｂを簡単に参照すると、図５に、受け取られたｋ＝５の入力キャラクタシーケンスに基づいて生成された全結合グラフの例が示され、図８Ｂに、全結合グラフに対する最小スパニングツリー表現が示されている。

【００９１】

いくつかの実施形態では、図５～図８Ｂに記載されるアプローチは、性能に関して追加

の技術的利点を提供し得る。たとえば、ＬＣＳアルゴリズムの、特定の従来の実現例は、Ｏ（ｎ$^{2}$）のランタイム性能を示すことができ、ここで、ｎはストリングの長さである。そのような実現例を２だけの代わりにｋ個のストリングに拡張すると、指数関数的なランタイム性能Ｏ（ｎ$^{k}$）をもたらし得、なぜならば、ＬＣＳアルゴリズムはｋ次元空間を探索するよう求められ得るからである。ＬＣＳアルゴリズムのそのような従来の実現例は、リアルタイムのオンラインユーザ体験に対しては、充分に機能しないか、または十分に適していないかもしれない。

【００９２】

上述したように、ＬＣＳアルゴリズムは、（ｋ（ｋ－１））／２回実行されてもよく、その場合、時として重複は以前に見られたのと全く同じであり、なぜならば、ＬＣＳアルゴリズムは、ユーザからの生の入力例がレゲックスカテゴリコードに変換された場合であり得るためである。したがって、いくつかの場合においては、記憶化が実現され得、キャッシュを使用して、以前に見られたＬＣＳ問題を以前に機能したＬＣＳ解にマッピングすることができる。

【００９３】

ある例示的な実施形態も、キャッシュテーブルが大きすぎる場合にキャッシュサイズを制御する有効な方法を提供する。キャッシュの一部を削除してもよい。例えば、切り捨てることができるハッシュマップテーブルを用いることができる。ハッシュマップテーブルは、任意に切り捨てることができる。

【００９４】

ＩＶ．陽性および陰性パターンマッチング例に基づく正規表現の生成

本明細書で説明するさらなる態様は、陽性および陰性の両方の例に対応する入力データに基づいて正規表現を生成することに関する。上述したように、陽性例は、正規表現生成器によって生成される正規表現にマッチするべき例のストリングとして指定される入力データキャラクタシーケンスを指し得る。一方、陰性例は、正規表現生成器によって生成される正規表現とマッチするべきでない例のストリングとして指定される入力データキャラクタシーケンスを指し得る。以下で説明するように、いくつかの実施形態では、正規表現生成器１１０は、位置、およびその位置で陽性例を陰性例から区別するキャラクタの最短サブシーケンスを識別するように構成され得る。次いで、最短サブシーケンスは、生成された正規表現にハードコード化され得、陽性例は正規表現とマッチすることになり、陰性例は正規表現によって除外される（たとえばマッチしない）ことになる。

【００９５】

図９は、陽性のキャラクタシーケンス例および陰性のキャラクタシーケンス例に基づいて正規表現を生成するための処理９００を示すフローチャートである。ステップ９０１において、正規表現生成器１１０は、陽性例に対応する１つまたは複数の入力データキャラクタシーケンスを受け取ることができる。ステップ９０２において、正規表現生成器１１０は、受け取った陽性例に基づいて正規表現を生成し得る。したがって、ステップ９０１～９０２は、入力データキャラクタシーケンスに基づいて正規表現を生成するために、上で論じた図３または図５において実行されるステップの一部またはすべてを含み得る。

【００９６】

ステップ９０３において、正規表現生成器１１０は、陰性例に対応する１つの追加入力データキャラクタシーケンスを受け取ることができる。したがって、ステップ９０２で生成された正規表現とマッチしないように、陰性例を具体的に指定する。いくつかの実施形態では、ステップ９０３において受け取られた陰性例は、ステップ９０２において生成された正規表現に対して最初にテストされてもよく、陰性例が正規表現とマッチしないと判断された場合、さらなるアクションはとられない。しかしながら、この例では、ステップ９０３において受け取られた陰性例のうちの少なくとも１つはステップ９０２において生成された正規表現とマッチする、と仮定されてもよい。したがって、ステップ９０４では、ステップ９０２で生成された正規表現内で曖昧性除去位置を判断してもよい。いくつかの実施形態では、曖昧性除去位置は、（たとえば正規表現の始まりにおける）プレフィッ

クス位置または（たとえば正規表現の終わりにおける）サフィックス位置のいずれかとして選択され得る。たとえば、正規表現生成器１１０は、陽性例を陰性例から区別するためにプレフィックスで必要とされるであろう第１の数のキャラクタと、陽性例を陰性例から区別するためにサフィックスで必要とされるであろう第２の数のキャラクタとを判断してもよい。次いで、正規表現生成器１１０は、必要とされる置換キャラクタの最短数に基づいてサフィックスまたはプレフィックスを選択し得る。場合によっては、曖昧性除去位置としてプレフィックスを使用することが、可読性のために好まれる（たとえば重み付けされる）ことがある。さらに他の例では、曖昧性除去位置は、正規表現のプレフィックスまたはサフィックスに対応しないミッドスパン位置であり得る。

【００９７】

ステップ９０５において、正規表現生成器１１０は、判断された位置で正規表現に挿入されると、陽性例を陰性例から区別することができるカスタムキャラクタクラスの置換シーケンス（またはより具体的には、カスタム「大括弧」正規表現キャラクタクラスのシーケンス）を判断することができる。いくつかの実施形態では、ステップ９０５において、正規表現生成器１１０は、陽性例および陰性例の各々から、曖昧性除去位置（または置換位置）に対応するテキストフラグメントを取り出し、次いで、それらテキストフラグメントを使用して、陽性例を陰性例から区別する置換シーケンスとして使用される弁別子を判断することができる。さらに、ステップ９０５で判断される弁別子置換シーケンスは、カスタムキャラクタクラスの、複数の異なる置換シーケンスを含むことができ、それらは、正規表現内の同じ位置または異なる位置のいずれかで置換されることができる。

【００９８】

上述したように、場合によっては、ステップ９０５における置換シーケンスの判断は、ステップ９０４における曖昧性除去位置（または置換位置）の判断との関連で実行されてもよい。たとえば、正規表現生成器１１０は、第１の可能な置換位置において陽性例を陰性例から区別し得る１つまたは複数の置換シーケンスを判断し得る。正規表現生成器１１０はまた、第２の異なる可能な置換位置において陽性例を陰性例から区別し得る１つまたは複数の他の置換シーケンスも判断し得る。この例において、異なる可能な置換位置および対応する置換シーケンスの間で選択するとき、正規表現生成器１１０は、ヒューリスティック式を適用して、置換位置のキャラクタのサイズ、ならびに対応する置換シーケンスの数および／またはサイズ、のうちの１つまたは複数に基づいて選択を実行することができる。最後に、ステップ９０６において、正規表現は、１つまたは複数の判断された置換シーケンスを判断された位置に挿入して正規表現の以前の部分を置換することによって、修正され得る。場合によっては、ステップ９０６における正規表現の修正に続いて、陽性例および／または陰性例を、修正された正規表現に対してテストして、陽性例が正規表現とマッチし、陰性例が正規表現とマッチしないことを確認してもよい。

【００９９】

図１０Ａおよび図１０Ｂは、陽性および陰性のキャラクタシーケンスの例に基づく正規表現の生成を示す例示的なユーザインターフェイス画面である。したがって、図１０Ａおよび図１０Ｂに示される例は、上述の図９の処理の実行中に表示されるユーザインターフェイスに対応し得る。図１０Ａにおいて、ユーザは、データ入力キャラクタシーケンスの３つの陽性例１００１を提供し、正規表現生成器１１０は、陽性例の各々にマッチする正規表現１００２を生成する。次に、図１０Ｂにおいて、ユーザは、１つの陰性例１００４を提供し、正規表現生成器１１０は、陽性例の現在のセット１００３および陰性例の現在のセット１００４の両方に基づく修正された正規表現１００５を生成する。

【０１００】

上で注記したように、いくつかの実施形態では、陽性例および陰性例の両方が受け取られると、正規表現生成器１１０は、弁別子、つまり陽性例を陰性例から区別する１つまたは複数のキャラクタの最短サブシーケンスを識別してもよい。選択された弁別子は、最短シーケンス（たとえばカテゴリコードで表現される）であってもよく、陽性または陰性のいずれかであってもよく、陽性例はマッチし、陰性例はマッチしないことになる。場合に

よっては、弁別子は、次いでステップ９０５で正規表現にハードコード化され得る置換サブシーケンスに対応してもよい。一例として、「[AL][a-z]+」において、[AL]は、それが街路サフィックスに適用されると仮定すると、「Alley（小路）」、「Avenue（大通り）」、および「Lane（横町）」とはマッチする（またはそれらを認める）であろうが、他のすべてのものとはマッチしない（または許容しない）であろう陽性の弁別子である。別の例として、「[BC][o][a-z]+」において、[BC][o]は、「Boulevard（並木道）」および「Court（袋小路）」とマッチするであろう２つのキャラクタクラスのシーケンスからなる陽性の弁別子である。さらに別の例として、「[^A][a-z]+」において、[LA]は、「Alley」および「Avenue」を許容しないであろう陰性の弁別子であってもよい。場合によっては、アルゴリズムは、正しく弁別するために陰性後読みを生成してもよい。たとえば、(?〈!Av)[A-Za-z]+は、「Avenue」は除外するが、「Alley」は認めるであろう。

【０１０１】

別の例として、ユーザが陽性例「202-456-7800」および「313-678-8900」ならびに陰性例「404-765-9876」および「515-987-6570」を供給する場合、ある実施形態では、正規表現生成器１１０は、正規表現

【０１０２】

【数１９】

「\d\d\d-\d\d\d-\d\d00」

【０１０３】

を生成してもよい。すなわち、置換キャラクタサブシーケンスは、正規表現のサフィックスについて、（たとえば、目的が、ビジネス電話番号とマッチする正規表現であると仮定して、）00で終わる電話番号は陽性例を陰性例から区別するという判断に基づいて、識別され得る。これは、サフィックスによる陰性例の例（より具体的に言えば、陽性のサフィックスを使用することによって陰性例に対処する例）であるが、さまざまな他の実施形態は、プレフィックス、サフィックス、またはミッドスパン位置のいずれかでの置換をサポートしてもよい。ミッドスパン位置での置換の例では、スパン内にオフセットされるキャラクタが追跡され、ミッドスパンポイントで分割され得る。

【０１０４】

プレフィックスまたはサフィックスを使用するかどうかを判断するために、いくつかの実施形態では、$k_a$およびプレフィックス／サフィックスのすべての組み合わせにわたって最小スコアが選択されるヒューリスティックが使用される：

【０１０５】

【数２０】

$$\text{スコア} = k_a \min^2\left\{\frac{|\mathbf{F}_p|}{1+|\mathbf{E}_p|}, \frac{|\mathbf{F}_n|}{1+|\mathbf{E}_n|}\right\} + \begin{cases} 0.0 & \text{（プレフィックスの場合）} \\ 0.1 & \text{（サフィックスの場合）} \end{cases}$$

【０１０６】

$k_a$＝アフィックス（プレフィックスまたはサフィックス）の曖昧性を除去すると考えられるキャラクタの数

$|\mathbf{F}_p|$＝アフィックスの曖昧性を除去するために必要とされる陽性例からの一意のテキストフラグメントの数

$|\mathbf{F}_n|$＝アフィックスの曖昧性を除去するために必要とされる陰性例からの一意のテキストフラグメントの数

$|\mathbf{E}_p|$＝ユーザによって提供される（完全な）陽性例の数

$|\mathbf{E}_n|$＝ユーザによって提供される（完全な）陰性例の数

上記の例では、ヒューリスティックは、より長い曖昧性除去テキストフラグメント（したがって、たとえば、$k_a$による乗算）よりもより短い曖昧性除去テキストフラグメントを

好むように設計される。ヒューリスティックはまた、可読性を改善するよう、サフィックス（したがって、たとえば、サフィックスに対する０．１のペナルティ）よりもプレフィックスを好むように設計される。最後に、ヒューリスティックは、より長いプレフィックスまたはサフィックスの曖昧性除去（たとえば置換）を、より多数のストリングフラグメント（したがって、たとえば、置換されるべきストリングフラグメントの数を二乗）を使用することによる曖昧性除去よりも好むように設計される。

【０１０７】

上述したように、いくつかの実施形態はまた、陰性のミッドスパン例、ならびに陰性の後読み例および陰性の先読み例をサポートしてもよい。

【０１０８】

いったんプレフィックス／サフィックスおよびk（曖昧性除去すべきキャラクタの数）が判断されると、正規表現生成器１１０は、さらに、その曖昧性除去を、生成された正規表現においてどのように表現するかを判断することができる。生成された正規表現は、陽性例のように見えるアフィックス（たとえばプレフィックスまたはサフィックス）について許容的（permissive）であってもよく、または陰性例のように見えるアフィックスを除外してもよい。

【０１０９】

【数２１】

$$usePermissive = \frac{|\mathbf{E}_p|}{|\mathbf{F}_p|} - \frac{|\mathbf{E}_n|}{|\mathbf{F}_n|}$$

【０１１０】

usePermissiveがゼロよりも大きい場合、陽性例のように見えるものは、（各キャラクタ位置）について１つずつ、陽性例から取られたキャラクタを許容する正規表現を生成することによって通される。他の場合では、正規表現生成器１１０は（各キャラクタ位置について）１つずつ、陰性例から取られたキャラクタを許容しない正規表現を生成することにより、陰性例のように見えるものを許容しないアプローチをとってもよい。

【０１１１】

別の例として、陽性例8amおよび陰性例9pmについて生成された正規表現は、

【０１１２】

【数２２】

```
\d[^p]m
```

【０１１３】

であるかもしれない。これは、カレット構文を使用する。場合によっては、正規表現生成器１１０は、より短い正規表現を好むように構成されてもよく、それは、ユーザとってより可読であり得るだけでなく、より正しい可能性があり得る。その原理は、今後、頻繁に出現するキャラクタは、今後再び出現する可能性がよりありそうであり、したがって、頻繁に出現するキャラクタに強調がおかれるべきである、というものである。一意のキャラクタ｜$\mathbf{E}_p$｜がより少ない場合（出現するものはより頻繁に出現するため、一意性はより少ない）、これは、分母内にそれを有することによって、ヒューリスティックにおいて報酬を与えられる。

【０１１４】

再び上記のusePermissive例のヒューリスティックを参照すると、１つの一意の陽性アフィックスを判断することは、ユーザから１つの陽性例しかない場合、大きな特色ではない。したがって、このヒューリスティックでは低い｜$\mathbf{E}_p$｜は、分子内にそれを有することによってペナルティを科せられる（すなわち、高い｜$\mathbf{E}_p$｜は、このヒューリスティックにおいて報酬を与えられる）。

【０１１５】

さらに、いくつかの実施形態では、陰性例は、後読みおよび／または先読みに基づいてもよい。たとえば、ユーザは、「323-1234」の陽性例および「202-754-9876」の陰性例を提供し、その場合、それはエリアコードを有する電話番号を除外するためにレゲックス後読み構文(?〈!)の使用を伴う。

【０１１６】

場合によっては、陰性例は、任意選択のスパンに基づいてもよい。たとえば、ユーザは、「ab」および「a2b」の陽性例および「a3b」の陰性例を提供してもよい。この場合、ある例示的な実現例は失敗し得、なぜならば、それは、必要とされるスパンのみに基づいて弁別しようとし得、「2」の桁は任意選択のスパンにあるからである。この例において、失敗とは、生成された正規表現が、陽性例のすべてに（正しく）マッチし、また、陰性例の１つまたは複数に（誤って）マッチする状況を指し得る。そのような場合、ユーザは、失敗に対して警告されることができ、生成された正規表現を手動で修復するために、および／または陰性例の一部を除去するために、ユーザインターフェイスを介して選択肢を提供されることができる。

【０１１７】

Ｖ．正規表現生成のためのユーザインターフェイス

本明細書で説明する追加の態様は、正規表現の生成に関連するグラフィカルユーザインターフェイス内にいくつかの異なる特徴および機能を含む。以下で説明するように、これらの特徴のうちのあるものは、ユーザ選択のためのさまざまな選択肢、および陽性例および陰性例の強調表示、陽性例および陰性例のカラーコード化、ならびにデータセル内の複数の重複／ネストされた強調表示を含み得る。

【０１１８】

図１１は、ユーザインターフェイス内で受信されたユーザデータ選択に基づいて正規表現を生成するための処理１１００を示すフローチャートである。図１１の例示的な処理は、入力データキャラクタシーケンスに基づいて正規表現を生成する前述の例のいずれかに対応し得る。しかしながら、図１１は、クライアントデバイス１２０上で生成および表示され得るユーザインターフェイスに関する処理を説明する。ステップ１１０１において、ユーザインターフェイスを介したユーザからの要求に応答して、正規表現生成器１１０は、（たとえばデータリポジトリ１３０から）データを取り出し、そのデータをグラフィカルユーザインターフェイス内においてテーブル形式でレンダリング／表示してもよい。なお、この例ではテーブル形式データを用いているが、他の例ではテーブル形式データを使用および表示しなくてもよいことを理解されたい。たとえば、場合によっては、ユーザは、（ユーザインターフェイスからデータを選択するのではなく）生データを直接タイプすることができる。さらに、データがユーザインターフェイスを介して提示されるとき、データはテーブル形式である必要はなく、構造化されていないデータ（たとえばドキュメント）または半構造化（たとえば、ツイートまたはポストのような非フォーマット化／非構造化データアイテムのスプレッドシート）であってもよい。さまざまな例では、テーブル形式データは、取引データ、顧客データ、実績データ、予測データ、および／またはビジネスもしくは他の組織のためにデータリポジトリ１３０に記憶され得るデータの任意の他のカテゴリに対応し得る。ステップ１１０２において、入力データのユーザ選択がユーザインターフェイスを介して受信され得る。選択された入力データは、たとえば、ユーザによって選択されたあるデータセル全体、またはあるデータセル内のキャラクタのサブシーケンスに対応し得る。ステップ１１０３において、正規表現生成器１１０は、ステップ１１０２において受信された入力データ（たとえばデータセルまたはその一部分）に基づいて正規表現を生成し得る。ステップ１１０４において、ユーザインターフェイスは、正規表現の生成に応答して更新され得る。場合によっては、ユーザーインターフェイスは、単に、生成された正規表現をユーザーに表示するために更新されてもよく、一方、他の場合では、ユーザーインターフェイスは、以下で説明するさまざまな他の方法で更新されてもよい。この例に示されるように、ユーザは、ユーザインターフェイスを介して複数の異なる入力データキャラクタシーケンスを選択してもよく、受信された各新たな入力データに

応答して、正規表現生成器１１０は、キャラクタシーケンスの第１および第２の（陽性の）例の両方を包含する更新された正規表現を生成してもよい。次いで、ユーザがキャラクタの第３のシーケンスを（たとえば、両方のキャラクタシーケンス外、または第１もしくは第２のキャラクタシーケンス内で）を強調表示すると、正規表現生成器１１０は、正規表現を再び更新してもよい等となる。いくつかの実施形態では、正規表現生成器１１０は、アルゴリズムをリアルタイム（またはほぼリアルタイム）で実行し得、全面的に新たな正規表現が、ユーザによってなされた各新たなキーストロークまたは各新たな強調表示されたセクションに応答して生成され得る。さらに、ユーザが、前の強調表示上を部分的に強調表示（または他の態様で選択）する場合、古い強調表示は削除されてもよく、新たな強調表示が追加されてもよい。

【０１１９】

したがって、図１１に示すように、ユーザインターフェイスを介するキャラクタシーケンスのユーザ選択に応答して、正規表現生成器１１０は、正規表現を生成および表示し得る。たとえば、ユーザがキャラクタの第１のシーケンスを強調表示すると、正規表現生成器は、キャラクタの第１のシーケンスを表す正規表現を生成し、表示し得る。ユーザがキャラクタの第２のシーケンスを強調表示すると、正規表現生成器は、キャラクタの第１のシーケンスとキャラクタの第２のシーケンスとの両方を包含する更新された正規表現を生成し得る。次いで、ユーザがキャラクタの第３のシーケンスを（たとえば第１のシーケンスまたは第２のシーケンスのいずれか内で）強調表示すると、正規表現生成器は正規表現を再び更新し得る等となる。

【０１２０】

図１２は、ユーザインターフェイス内で受信されたユーザデータ選択を介して、正規表現を生成し、キャプチャグループに基づいてデータを抽出するための処理１２００を示す別のフローチャートである。ステップ１２０１において、ステップ１１０１で上述したように、正規表現生成器１１０は、（たとえばデータリポジトリ１３０から）データを取り出し、そのデータをグラフィカルユーザインターフェイス内においてテーブル形式でレンダリング／表示することができる。ステップ１２０２において、正規表現生成器１１０は、特定のデータセル内のテキストフラグメントのユーザ強調表示の選択を受け取ることができる。ステップ１２０３において、正規表現生成器１１０は、選択されたデータセルの陽性例に基づいて正規表現を生成することができ、ステップ１２０４において、セル内で強調表示されたテキストフラグメントに基づいて正規表現キャプチャグループを作成することができる。ステップ１２０５において、正規表現生成器１１０は、生成された正規表現とマッチする、表示されたテーブル形式データ内の１つまたは複数の追加セルを判断することができ、ステップ１２０６において、生成された正規表現とマッチする追加セル内の対応するテキストフラグメントを抽出することができる。

【０１２１】

したがって、陽性例を供給することに加えて、ユーザは、（たとえばマウステキスト強調表示を介して、）選択された陽性例のいずれか内でテキストフラグメントを選択してもよい。これに応答して、正規表現生成器１１０は、例から、そのテキストフラグメントを抽出し、正規表現が適用されているテキスト中の他のすべてのマッチから、対応するフラグメントを抽出するために、正規表現キャプチャグループを作成してもよい。マッチするデータセルからテキストフラグメントを抽出することは、削除および修正も含み得、場合によっては、半構造化または非構造化テキストの既存の列からデータの新たな列を作成するために用いられ得る。

【０１２２】

ユーザが陽性のデータ例を選択する例を用いて、ユーザが年を強調表示した場合、正規表現生成器１１０は正規表現

【０１２３】

【数２３】

(?:[A-Z]{3}\s+\d\d,\s+|\d\d/\d\d)(\d\d\d\d)

【０１２４】
を生成することができる。この例に示すように、正規表現生成器１１０は、年の周りに括弧を付けてあり、また、月および日の周りの古い括弧（交代のために使用）を?:レゲックス構文の使用により「非キャプチャ」グループに変換している。いくつかの実施形態では、抽出／キャプチャグループは、スパン境界上にあることが要求され得、そのような実施形態では、正規表現生成器１１０は、強調表示されたキャラクタ範囲を入力として取り得、最も近い位置指定子スパン境界を包含するようにそれを拡張し得る。しかしながら、他の例では、ミッドスパン抽出／キャプチャは、ユーザインターフェイスによってサポートされてもよい。
【０１２５】
いくつかの実施形態では、ユーザインターフェイスは、第２のキャラクタシーケンス内の第１のキャラクタシーケンスの選択を含むユーザからの入力データをサポートしてもよい。たとえば、ユーザは、より大きな以前に強調表示されたキャラクタシーケンス内の１つまたは複数のキャラクタを強調表示することができ、第２のユーザ選択は、より大きな第１のユーザ選択のためのコンテキストを提供してもよい。そのような実施形態は、入力データが、より高い特異性で正規表現生成器１１０に提供されることを可能にし得る。
【０１２６】
さらに、いくつかの例では、ユーザがユーザインターフェイス内で選択する（たとえばテキストを強調表示する）ことに応答して、動作を開始し、ダイアログを開くことができる。場合によっては、ダイアログは、メイン画面とのユーザ対話を妨げないフローティングツールボックスウィンドウなどの非モーダルダイアログであってもよい。ダイアログはまた、ユーザがどのような主要な操作を行っているかに応じて外観および／または機能性を変化させることもできる。したがって、そのような場合、ユーザは、キャプチャグループテキストフラグメントの修正、抽出などを開始するために、選択されたテキストを強調表示した後にさらなるメニューアイテムを検索する必要がない。
【０１２７】
さらに、特定の実施形態では、正規表現を生成するために提供されるユーザインターフェイスは、３つの強調表示モード、すなわち、ネスト化自動、ネスト化手動、および単一レベル、を含み得る。ネスト化自動は、ネスト化（自動-外側）としても知られる。ネスト化手動は、ネスト化（完全-制御）としても知られる。これらの例では、単一レベルは、１つのレベルの強調表示（または他の形態のテキスト選択）を指し得、これは、正規表現を、キャプチャグループを伴わずに生成させ得る。ネスト化手動強調表示モードは、２つのレベルの強調表示（または他の形態のテキスト選択）の識別を可能にする。これにより、キャプチャーグループを伴う正規表現の生成を引き起こす。ネスト化自動強調表示は、外側強調表示（または他の形態のテキスト選択）が、テキスト全体（例えば、スプレッドシートセル全体）のコンテンツとなるように自動的に設定され得ることを除き、ネスト化手動強調表示モードと同一であってもよい。
【０１２８】
場合によっては、デフォルト動作モードは、セル全体が強調表示された領域として識別されることであってもよく、ユーザは、強調されたセル内の１つまたは複数の追加のサブシーケンスをさらに強調表示してもよい。他のモードでは、ユーザは、テーブル形式データディスプレイのデータセル内で両方の強調表示を手動で指定することを許可されてもよい。さらに他のモードでは、ユーザは、内側強調表示なしで外側強調表示を手動で指定することを許可されてもよい。これらのその他のモードは、半構造化データ、たとえば、ブラウザ「ユーザエージェント」ストリングなどのツイートまたは他の長いストリングからなるデータの列に、より適し得る。「半構造化」データは、ユーザインターフェイス内でテーブル形式で表示され得るデータを指すが、テーブル内の列は非構造化テキストからなる。
【０１２９】
いくつかのそのような実施形態では、ユーザインターフェイスを介したユーザによる内

側および外側の選択（たとえば強調表示）は、カラーコード化によって区別され得る。たとえば、陽性例の外側強調表示は、第１のテキスト／背景色の組み合わせで示されてもよく、陽性例の内側強調表示は、異なる対照的なテキスト／背景色の組み合わせで示されてもよい。

【０１３０】

上述したように、ユーザは、キャラクタサブシーケンスの選択を介してキャプチャグループの選択を指定することができる。ＧＵＩを用いて、強調表示（または他の表示）を介するユーザ選択を容易にしてもよい。一例を図１３に示し、例示的ユーザインターフェイス画面がテーブル形式データディスプレイとともに示される。この例では、図１３は、たとえば、ユーザが列値の１つまたは複数の所望の要素を横切ってマウスをドラッグすることによって引き起こされる、列値内の強調表示を示す。なお、ユーザ強調表示が実行される「セル」は、列値の選択を示す色変化を示してもよい。この色変化は、ユーザ強調表示に応答する自動化された強調表示と解釈されてもよい。

【０１３１】

図１４および図１５は、テーブル形式ディスプレイからのデータの選択に基づく正規表現およびキャプチャグループの生成を示す例示的なユーザインターフェイス画面である。これらの例では、図１４および図１５は、テーブル形式データディスプレイ内のユーザ強調表示１４０１の検出が自動的に表示される追加のユーザインターフェイスウィンドウを示す。ウィンドウは、陽性例を表示するためのフィールド１４０２と、陰性例を表示するためのフィールドと、テーブル形式データディスプレイからの陽性例の選択に応答して動的に（およびほぼ瞬時に）生成される正規表現を表示するためのフィールドとを含む。これらの例では、列値１４０１内のユーザ強調表示は、自動化された強調表示内のユーザ強調表示と同等であり得る。したがって、エリアコードのユーザ強調表示により、ユーザ強調表示されたエリアコード１４０１だけでなく、電話番号の残りの部分も陽性例フィールド１４０２にポピュレートされる。

【０１３２】

しかしながら、ユーザ強調表示は、自動強調表示内の性能に限定されないことを理解されたい。たとえば、ユーザ強調表示は、代替的に、他のユーザ強調表示内で実行されてもよい。別の例として、ユーザ強調表示は、代替的に、内側強調表示（たとえば、強調表示されたテキスト内でのさらに強調表示）なしで実行されてもよい。これらの代替例は、「ツイート」または他の長いストリング（たとえばブラウザ「ユーザエージェント」ストリング）を含むデータの列などの半構造化データに特に適している。

【０１３３】

さらに、対応する正規表現が生成されると、正規表現にマッチする他の列値１４０２が、追加の自動化された強調表示に基づいて識別され得る。図１４および図１５に示される例では、追加の自動化された強調表示は、生成された正規表現のキャプチャグループにマッチする、これらの他の列値の要素を示す。追加の自動化された強調表示は、ユーザ強調表示に使用される色とは異なる色を使用して実行されてもよい。

【０１３４】

図１５に示すように、他の例のユーザ選択を示すために、追加のユーザ強調表示が示される。追加のユーザ強調表示は、上述の方法と同様の方法で実行され得る。したがって、図１５のユーザインターフェイスは、陽性例を表示するためのフィールド１５０２における他の例のポピュレーションを示す。これは、追加のユーザ強調表示の検出に応答して起こり得る。さらに、生成された正規表現１５０３は、それが陽性例１５０２のすべてにマッチするように、動的にかつほぼ瞬時に更新されてもよい。更新された正規表現の生成に応答して、更新された正規表現にマッチする他の列値１５０４の自動化された強調表示も更新され得る。いくつかの実現形態では、動的カラーコード化も使用され得る。たとえば、マッチは、第１の色（たとえば青色）を使用してカラーコード化されてもよく、陽性例は、第２の色（たとえば緑色）を使用してカラーコード化されてもよく、陰性例は、第３の色（たとえば赤色）を使用してカラーコード化されてもよい。テーブル形式のデータビ

ュー（スプレッドシートなど）またはその他の（たとえば半構造化データまたは非構造化データの）無限スクロールデータビュー内では、ビューが実際の追加データにスクロールダウンされると、新たに可視となるデータがカラーコード化されてもよい。

【０１３５】

図１６Ａおよび図１６Ｂは、テーブル形式ディスプレイからの陽性例および陰性例の選択に基づく正規表現の生成を示すユーザインターフェイス画面例である。図１６Ａ～図１６Ｂでは、陽性例フィールド１６０２からの個々の例は、陽性例フィールド１６０３から除去され得、および／または陰性例フィールド１６０３に移され得る。ユーザインターフェイス内で、これは、たとえば、ユーザが例の１つをクリック（たとえば右クリック）してそれを選択することによって実行されてもよい。選択は、ユーザーインターフェイスに、削除オプションおよび変更オプションを含むメニュー１６０２を表示させることができる。その後、オプションをクリックすると、対応する機能が実行される。

【０１３６】

図１６Ａおよび図１６Ｂに示される例において、変更オプションのユーザ選択の結果は、選択された例を陰性例フィールド１６０３に移動させ、正規表現１６０１を正規表現１６０４に更新させ、この正規表現１６０４は、動的にかつほぼ瞬時に生成され得る（たとえばある実施形態では、３０ｍｓ～９０００ｍｓの間、または１００ｍｓメジアンである）。更新された正規表現１６０４の生成に応答して、更新された正規表現にマッチする他の列値の自動化された強調表示も、テーブル形式データディスプレイ内で更新され得る。さらに、自動化された強調表示は、陰性例に対応する任意の列値を含む、陰性例の一部または全部に対して実行されてもよく、それは、上で使用された色のいずれとも異なる色を使用して強調表示されてもよく、または他の態様では他の視覚的技法を使用してユーザインターフェイス内で区別されてもよい。

【０１３７】

いくつかの実施形態では、ユーザインターフェイスを介して陰性例を指定することは、図１６Ａおよび図１６Ｂに示されるように、最初にその例を陽性例として指定し、次いでそれを陰性例に変換することを必要とする必要はない。むしろ、陰性例は、さまざまな方法で指定され得る。たとえば、ユーザは、ユーザインターフェイスを介して列値（たとえば、自動化された強調表示が実行されて、生成された正規表現とマッチすることを示した、他の列値のうちの１つ）を選択する（たとえば右クリックする）ことができ、それによって、オプション（たとえば「新たな反例を作成する」）を含むメニューのディスプレイに、選択された列値を陰性例として指定させることができる。

【０１３８】

したがって、図１６Ａおよび図１６Ｂに示される例を使用して、更新された正規表現１６０４の生成に応答して、更新された正規表現にマッチする他の列値の自動化された強調表示も更新され得る。これらの例では、更新された正規表現は、「９」で終わる電話番号を指定する。

【０１３９】

図１４および図１５を簡単に参照すると、「抽出」ボタンがユーザによってクリックされるか、または他の態様で選択されると、現在の正規表現１４０３または１５０３にマッチするすべてのセル内の強調表示されたテキストフラグメントを抽出するための動作が、開始され得る。図１４および図１５には示されていないが、いくつかの実施形態では、ユーザインターフェイスは、「抽出」ボタンに加えて、またはその代わりに、他の選択可能なボタンを提供してもよい。たとえば、「置換」ボタンが、ユーザ強調表示された要素をユーザ指定された要素に置換するためのオプションとして提示されてもよい。追加または代替として、１つまたは複数の「削除」ボタンが、事実上、ユーザ強調表示された要素を何にも置換しないオプションとして提示されてもよい。たとえば、「フラグメントを削除」操作および／または「行を削除」操作の一方または両方が実現されてもよく、それは、それぞれ、ユーザ強調表示されたテキストフラグメントまたは行全体のいずれかを削除することになる。さまざまな実施形態において実現され得る追加の操作は、「行を保持」操

作、「分割」操作（たとえば、コンマを強調表示し、次いで、コンマ分離成分を別々の複数の新たな列に抽出する）、および「難読化」操作（たとえば、強調表示されたテキスト／キャプチャグループを「＃」または他の記号のシーケンスで置き換える）を含み得る。

【０１４０】

この例では、「抽出」ボタンが選択されたことに応答して、抽出操作が、下流の操作によって実行されるべき変換スクリプトのリストに追加され得る。いくつかの実施形態では、変換スクリプトのリストは、ユーザによるレビュー／修正のためにユーザインターフェイスの一部分に表示されてもよい。代替的に、抽出操作は、その場で実行されて、レゲックスキャプチャグループの内容（たとえば、陽性例のユーザ強調表示部分に対応する要素）を含む新たな列を生成してもよい。図１４および図１５に示される例では、エリアコードの新たな列および／または新たなテーブルが、「抽出」ボタンの選択に応答して生成されてもよい。

【０１４１】

図１７は、本明細書に記載する１つまたは複数の実施形態による、テーブル形式ディスプレイからのデータの選択に基づく正規表現およびキャプチャグループの生成を示す別の例示的なユーザインターフェイス画面である。

【０１４２】

Ａ．複数強調表示

いくつかの実施形態では、ユーザインターフェイスはまた、例につき複数の強調表示をサポートしてもよい。図４２，図４３，図４４，図４５，図４６，図４７，および図４８は、いくつかの例示的実施形態による、複数の強調表示を実現するためのユーザインターフェイス４２００，４３００，４４００，４５００，４６００，４７００，および４８００を示す。異なる強調表示方法を以下に記載する。例示的な実施形態では、内側強調表示および外側強調表示は、異なるカラーコードを用いることによって区別することができる。

【０１４３】

図４２は、いくつかの例示的実施形態による、単一レベルモードでデータセットのビューを表示するユーザインターフェイス４２００を示す。図４２は、単一レベル強調表示モードにある間、強調表示がユーザインターフェイス上にどのように現れるかを示す。図４２において、ユーザは、アプリケーション４２３６（例えばMozilla）のバージョン番号４２３５（例えば、"5.0"）を強調表示する。ユーザがバージョン番号４２３５を強調表示するのに応答して、ユーザインターフェイスデータモデル生成器は、バージョン番号４２３７（"6.1"）、４２３８（"1.9"）、４２３９（"2.2"）、および４２４０（"3.6"）を強調表示する。

【０１４４】

強調表示に基づいて、例別レゲックスダイアログボックス４２３０が現れる。例別レゲックスダイアログボックス４２３０は、ユーザによって指定された、指定された例４２３３を含む。この例では、バージョン番号"5.0"がユーザによって指定される。また、例別レゲックスダイアログボックス４２３０は、強調表示モード４２５０が単一レベル４２５１であることを示す。

【０１４５】

図４３は、いくつかの例示的実施形態による、ネスト化された完全制御モードで強調表示されたデータを表示するユーザインターフェイス４３００を示す。図４３に示す例では、ダイアログボックス４３３０の強調表示モードパネル４３５０から「ネスト化（完全制御）」４３５１が選択されている。また、図４３に示すように、ユーザは、外側強調表示を、ソフトウェア名４２３６（例えば「Mozilla」）およびバージョン番号４２３５（例えば"5.0"）として表示する。例別レゲックスダイアログボックス４３３０は、強調表示モード４３５０がネスト化（完全制御）４３５１にあることを示す。さらに、データセットは、現在、マルチ強調表示モード４３６０にある。図４３において実行される強調表示は、外側強調表示を、ネスト化された全自動モードで行う。

【０１４６】

例示的な実施形態では、現在実行されている外側強調表示である進行中外側強調表示は、金色のようなそれ自体の緊急性の色で、第１のクラスの態様で取り扱うことができる。例示的な実施形態では、強調表示は、強調表示の緊急性、優先度、および／または重要性を示すために用いることができる異なる色で現れることができる。

【０１４７】

図４４は、いくつかの例示的実施形態による、ネスト化された完全制御モードで強調表示されたデータを表示するユーザインターフェイス４４００を示す。図４４に示すように、マルチ強調表示モード４４６０が選択される。ユーザは、外側強調表示を、ソフトウェア名４２３６およびバージョン番号４２３５として表示している。外側強調表示として選択されるソフトウェア名４２３６（例えば、Mozilla）およびバージョン番号４２３５（例えば、5.0）は、第１の色で強調表示され得る。ユーザは、外側強調表示を選択した後、内側強調表示を識別することができる。例えば、ユーザは、バージョン番号４２３５（例えば、"5.0"）のみを内側強調表示として選択することができる。内側強調表示は、第１の色とは異なる第２の色で強調表示され得る。内側強調表示と外側強調表示とが異なる色で現れるので、外側強調表示と内側強調表示とを容易に識別することができる。図４４に示される例は、ネスト化された完全制御モード強調表示にある。したがって、強調表示モードパネル４４５０は、データセットがネスト化された完全制御モード４４５１にあることを示す。

【０１４８】

例示的な実施形態では、内側強調表示および外側強調表示は、異なる色またはカラーコードを用いることによって区別することができる。例えば、陽性例の外側強調表示は、淡緑色背景上に黒色テキストで示され得、陽性例の内側強調表示は、濃緑色背景上に淡緑色テキストで示され得る。

【０１４９】

図４４では、ユーザは、要素４２３５として識別されるバージョン番号"5.0"を選択している。ユーザがバージョン番号"5.0"（要素４２３５）を選択すると、ユーザインターフェイスデータモデル生成器は、バージョン番号"3.6"（要素４２４０）および"5.1"（要素４２４１）を自動的に選択することができる。ユーザインターフェイスデータモデル生成器は、ユーザによって選択されたバージョン番号に加えて、バージョン番号を強調したので、ユーザインターフェイスデータモデル生成器によって識別される追加のバージョン番号がユーザによって望まれない場合には、ユーザは追加の例を与えることができる。あるいは、ユーザがユーザインターフェイスデータモデル生成器によって実行される追加の強調表示に同意する場合、ユーザは強調表示されたデータにコマンドを適用することに進むことができる。

【０１５０】

例別レゲックスダイアログボックス４４３０は、指定された例４４３３を表示する。さらに、例別レゲックスダイアログボックス４４３０は、強調表示モード４４５０がネスト化（完全制御）４４５１であることを示す。また、例別レゲックスダイアログボックス４４３０は、マルチ強調表示モード４４６０にあることを示す。生成された正規表現４４３２も、例別レゲックスダイアログボックス４４３０において識別される。

【０１５１】

図４５、図４６、図４７に示すように、ユーザは、さらなる例を与えることができる。図４５は、いくつかの例示的実施形態による、２つの陽性例が提供されるユーザインターフェイス４５００を示す。図４６は、いくつかの例示的実施形態による、２つの陽性例が提供されるユーザインターフェイス４６００を示す。図４７は、いくつかの例示的実施形態による、３つの陽性例が提供されるユーザインターフェイス４５００を示す。

【０１５２】

図４５に示すように、陽性例を与えるために、ユーザは、外側強調表示をソフトウェア名４５３６およびバージョン番号４５３５として示すことができる。外側強調表示として選択されたソフトウェア名４５３６（例えば、WindowsNT）およびバージョン番号４５

３５（例えば、6.1）は、第１の色で強調表示され得る。ユーザは、外側強調表示を選択した後、内側強調表示を識別することができる。例えば、ユーザは、バージョン番号４５３５（例えば、"6.1"）を内側強調表示として選択することができる。内側強調表示は、第１の色とは異なる第２の色で強調表示され得る。正規表現４５３２は、強調表示に基づいて生成することができる。

【０１５３】

ユーザーによって与えられる強調表示例は、例別レゲックスダイアログボックス４５３０内の指定された例４５３３において識別される。

【０１５４】

図４６は、いくつかの例示的実施形態による、更新された生成されたレゲックスを表示するユーザインターフェイス４６００を示す。図４６に示されるように、生成されたレゲックス４６３２は、２つの指定された例４６３３（例えば、"Mozilla 5.0"および"WindowsNT 6.1"）に基づいて更新される。追加の外側および内側強調表示が、ユーザによって、指定された例４６３３において与えられるので、生成されたレゲックス４６３２もそれに応じて変化する。生成されたレゲックス４６３２は、例別レゲックスダイアログボックス４６３０に現れる。

【０１５５】

図４７は、いくつかの例示的実施形態による、代替データ強調表示を表示するユーザインターフェイス４７００を示す。図４７に示すように、レゲックスを生成するための例としてアンダースコアを伴うバージョン番号を与えることができる。

【０１５６】

図４７に示すように、別の陽性例を与えるために、ユーザは、外側強調表示をソフトウェア名４７３５およびバージョン番号４７３５として示すことができる。外側強調表示として選択されたソフトウェア名４７３５（例えばMacOS X）およびバージョン番号４７３６（例えば10_6_8）は、第１の色で強調表示され得る。ユーザは、外側強調表示を選択した後、内側強調表示を識別することができる。例えば、ユーザは、バージョン番号４７３６（例えば10_6_8）を内側強調表示として選択することができる。内側強調表示は、第１の色とは異なる第２の色で強調表示され得る。

【０１５７】

ユーザーによって与えられる強調表示例は、例別レゲックスダイアログボックス４７３０内の指定された例４７３３において識別される。図４７に示すように、生成されたレゲックス４７３２は、３つの指定された例４７３３（例えば、"Mozilla 5.0"、"WindowsNT 6.1" および"MacOS X 10_6_8"）に基づいて更新される。追加の外側および内側強調表示が、ユーザによって、指定された例４７３３において与えられるので、生成されたレゲックス４７３２もそれに応じて変化する。

【０１５８】

図４８は、いくつかの例示的実施形態による、更新された生成されたレゲックスを表示するユーザインターフェイス４８００を示す。図４８に示されるように、ユーザがアンダースコアを伴うバージョン番号を選択することに応答して、生成されたレゲックス４８３２は、アンダースコア情報を含むように更新される。

【０１５９】

生成されたレゲックス４８３２は、３つの指定された例４８３３（例えば、"Mozilla 5.0"、"WindowsNT 6.1"および"MacOS X 10_6_8"」）に基づいて更新される。追加の外側および内側強調表示が、ユーザによって、指定された例４８３３において与えられるので、生成されたレゲックス４８３２もそれに応じて変化する。生成されたレゲックス４８３２は、例別レゲックスダイアログボックス４８３０に表示される。

【０１６０】

図４９は、いくつかの例示的実施形態による、多重強調表示を実行するための方法４９００のフローチャートである。

【０１６１】

ステップ４９１０において、ユーザは、データセット内のデータ（例えば、データフラグメント）を選択することができる。すなわち、ユーザは、外側強調表示（例えば、"Mozilla 5.0"）および内側強調表示（例えば、"5.0"）を選択することができる。ユーザは、第１のデータレコード４３２０において外側強調表示および内側強調表示を実行することができる。

【０１６２】

ステップ４９２０で、ユーザが初期強調表示を実施した後、ユーザインターフェイスデータモデル生成器は、ステップ４９１０でユーザによって行われた強調表示に対応する、データセット内の同じレコードおよび／または他のレコード内のデータ（例えば、データフラグメント）を自動的に強調表示することができる。

【０１６３】

ステップ４９３０では、正規表現を生成することができる。正規表現は、ステップ４９１０でユーザによって与えられる強調表示に基づいて、およびステップ４９２０でユーザインターフェイスデータモデル生成器によって実行される追加の強調表示に基づいて、生成される。

【０１６４】

ユーザが強調表示結果に満足しない場合、ユーザは、追加の例を与えるために、追加のデータを強調表示することができる。したがって、ステップ４９１０，４９２０および４９３０は、ユーザがデータセット上の最後の強調表示に満足するまで、繰り返すことができる。あるいは、ユーザは、生成された正規表現（例えば、４８３２）を修正して、所望の強調表示を生成することができる。

【０１６５】

例示的な実施形態では、ネスト化された（自動外側）モードにおける複数の強調表示について、必要な最小の数の周辺スパンに対して検索が実行される。ユーザインターフェイスが、複数の強調表示を伴うネスト化された（自動外側）モードである場合、バックエンドアルゴリズムは、特別モードに入ることができる。特別モードの理由は、例当たり複数の強調表示が、例示ストリングが通常よりも長いことを意味する傾向があるからである。ＬＣＳアルゴリズムは指数関数的なランタイムを有するので、これは外側強調表示の全長で実行するには長くかかりすぎるであろう。特別モードでは、アルゴリズムは内側強調表示で始まり、満足のいくレゲックスが生成されるまで、内側強調表示のまわりで外側強調表示を徐々に成長させる。

【０１６６】

特別モードを実行するための例示的なコードは、以下のコードの一部またはすべてを含むことができる：

【０１６７】

【数２４】

```
            if (mustUseLookAheadAndBehind && examplesScala.forall(e =>
e.highlights.forall(h => h.outer.start == 0 && h.outer.end == e.s.length-1))) {
              var result = InductResult(false, "", new java.util.ArrayList[Range]())
              var reachedMaximal = false
              var numLookAheadAndBehindSpans = 0
              do {
                var numLookBehindSpans = 0
                do {
                  val shortenedPosOnly = flaggedPosOnly.map(e => {
                    val es = toSpans(e.e.s)
                    val highlightSpanRange = highlightSpanRangeForExample(es,
e.e.highlight)
                    val expandedSpanRange = Math.max(0, highlightSpanRange.min -
numLookBehindSpans) to Math.min(es.length - 1, highlightSpanRange.max +
numLookAheadAndBehindSpans - numLookBehindSpans)
                    def toString(r:Range) = es.slice(r.start,
r.end+1).map(_.orig.head.s).mkString
                    val expandedSpanString = toString(expandedSpanRange)
                    val lookBehindLength = toString(expandedSpanRange.start until
expandedSpanRange.start + numLookBehindSpans - 1).length
                    FlaggedExample(e.useForBothTrainAndTest, NumberedExample(e.e.id,
expandedSpanString, e.e.isPositive, lookBehindLength to lookBehindLength +
toString(highlightSpanRange).length-1))
                  })
                  result = shortestRegEx(shortenedPosOnly, false, false,
inductParameters.useContext, mustUseLookAheadAndBehind, false).r
```

【０１６８】
【数２５】

```
            if (!result.isSuccess) {
                val someNeg = numberedExamples.filter(!_.isPositive).map(e =>
FlaggedExample(isMatch(result.r, e.s) && e.s.length > 0, e))
                if (someNeg.count(_.useForBothTrainAndTest) > 0)
                  result = shortestRegEx(shortenedPosOnly ++ someNeg,
inductParameters.isStartAligned, inductParameters.isEndAligned, inductParameters.useContext,
mustUseLookAheadAndBehind, false).r
                }
                numLookBehindSpans += 1
                reachedMaximal = shortenedPosOnly.indices.forall(i =>
shortenedPosOnly(i).e.s.length == flaggedPosOnly(i).e.s.length)
              } while (!result.isSuccess && !reachedMaximal &&
numLookBehindSpans < numLookAheadAndBehindSpans)
              numLookAheadAndBehindSpans += 1
            } while (!result.isSuccess && !reachedMaximal)
            result
```

【０１６９】

多重強調表示により、ユーザは、コマンド（例えば、抽出する、分割する、削除する、難読化する、置換する、および行をフィルタリングする）のいずれでも適用することができるデータ（例えばデータフラグメント）を容易に選択することができる。

【０１７０】

例単位複数強調表示の能力について、個々の強調表示は、全行の代わりにクリック可能であってもよい。場合によっては、チェックボックス（またはユーザインターフェイス内の他の技術）を用いて、例単位１強調表示モードと例単位複数強調表示モードとの間で切り換えてもよい。例えば、例につき複数の強調表示をサポートするモードでは、ユーザは、正規表現が生成され得る複数の異なる例を与えるために、単一のデータ例（例えば、スプレッドシートセル）内で複数の異なる強調表示された部分を選択し得る。例ごとの複数の強調表示において、陽性および陰性の両方の「例」が識別され得、１つの例のリストにおいてカラーコードとして示され得る。例えば、マルチ強調表示モードでは、ある強調表示をクリックすると、その１つのサブ例のみが陰性になる。例単位複数強調表示の機能が図４２～図４９に示されている。

【０１７１】

この一連の例に示すように、ユーザは、外側の強調表示された領域、それに続いて内側の強調表示された領域を順次選択して、対応する正規表現を生成／更新する。さらに、これらの図に示されるように、ユーザは、単一のデータセル内で複数の異なる内側／外側強調表示例を指定することができ、生成された正規表現は、ユーザインターフェイスを介して強調表示された（または他の方法でユーザによって選択された）各新たな例で更新され得る。

【０１７２】

いくつかの実施形態では、特別なマルチ強調表示ネスト化自動外側検索モードは、所定の数のスパン（例えば、１０スパン）の直径に制限され得る。そのような場合、最後のdo-whileループに関する条件は、while (!result.isSuccess && !reachedMaximal && numLookAheadAndBehindSpans 〈 10)のようであり得る。

【０１７３】
特定のコマンドを実現するときに、マルチ強調表示を用いてもよい。
Ｂ．コマンド
上述したように、ユーザインターフェイスは、様々な実施形態において抽出コマンドに加えて追加のタイプのコマンドをサポートしてもよい。ユーザインターフェイスデータモデル生成器の要素は、それらのコマンドを実現するよう構成され得る。
【０１７４】
図２４は、いくつかの例示的実施形態による、正規表現生成サーバ２４００を示す。正規表現生成サーバ２４００は、図１に示した正規表現生成器サーバ１１１に対応する。正規表現生成サーバ２４００は、複数のプロセッサおよびメモリを含むことができる。正規表現生成サーバ２４００は、正規表現生成器２４１０およびユーザインターフェイスデータモデル生成器２４２０を含むことができる。正規表現生成器２４１０は、レゲックス生成器とも称され得る。正規表現生成器２４１０は、図１の正規表現生成器１１６に対応し得る。ユーザインターフェイスデータモデル生成器２４２０は、図１のユーザインターフェイスデータモデル生成器１１４に対応し得る。
【０１７５】
正規表現生成器２４２０は、抽出する、分割する、削除する、難読化する、置換する、および行をフィルタリングする、を含むいくつかの異なるコマンドを実行することができる。したがって、正規表現生成器２４２０は、抽出データ生成器２４１１、分割データ生成器２４１２、削除データ生成器２４１３、難読化データ生成器２４１４、置換データ生成器２４１５、行フィルタリングデータ生成器２４１６を含むことができる。以下、正規表現作成サーバ２４００の各構成要素についてより詳細に説明する。
【０１７６】
抽出、分割、削除、難読化、置換、および行フィルタリングコマンドが説明されるが、例示的な実施形態は、追加のコマンドを実行するよう構成され得る。また、正規表現生成器２４１０は、図２４に示した以外のコマンド生成部を含むこともできる。
【０１７７】
１．抽出
上述したように、抽出コマンドにおいては、正規表現のパターンマッチが識別されてもよく、マッチするデータが、新たに作成された列に抽出されてもよい。抽出関数は、図１４および図１５に関して上で説明されている。
【０１７８】
２．分割
図２５Ａ、図２５Ｂ、および図２６は、ユーザインターフェイスを介して実現される分割コマンドの例を示す。図２５Ａおよび図２５Ｂは、いくつかの例示的実施形態による、分割コマンドを実現するためのユーザインターフェイス２５００およびユーザインターフェイス２５０１を示す。図２６は、いくつかの例示的実施形態による、データセット上の分割コマンドの結果を示す。
【０１７９】
分割コマンドは、デリミタとして用いられる特定の正規表現の存在に基づいて、複数の異なる列の作成をもたらすことができる。例えば、コンマ（「,」）がマッチされるべきパターンとして与えられる場合、コンマは、抽出操作中にデリミタとして用いられる。単一のコンマを含む抽出されるべき各データは、コンマにおいて分割され、２つの異なる列（すなわち、コンマの前のデータおよびコンマの後のデータ）に抽出される。分割されるべきデータ項目が２つのコンマを含む場合、そのデータはコンマに基づいて３つの部分に分割され、３つの新たな列に抽出される。したがって、分割コマンドは、（マッチが単一の列に入れられるであろう）パターンマッチに基づく抽出ではなく、デリミタに基づく（１つまたは複数の新たな列への）抽出を実行することができる。
【０１８０】
他の例では、分割コマンドは、特定のタイプのコンテンツに対して、マッチされるべき

パターン（またはデリミタ）における中括弧または大括弧、アスタリスク、「〈」もしくは「〉」またはハイフンおよびダッシュ文字を用いるなどして、実行されてもよい。追加の記号を、デリミタとして用いてもよい。

【０１８１】

図２５Ａに示されるように、データセットがユーザインターフェイス２５００上に表示される。データセットはスプレッドシートとすることができる。データセットは、電話番号を含む列２５１０を含むことができる。電話番号は、第１のハイフン２５１１および第２のハイフン２５１２を含むことができる。分割コマンドが電話番号の列２５１０上で実現される場合、電話番号は、ユーザによって所望のように分割され得る。

【０１８２】

図２７は、いくつかの例示的実施形態による、分割コマンドを実行するための方法２７００のフローチャートを示す。図２７の例では、データを分割するためのデリミタとしてハイフン文字が用いられる。

【０１８３】

ステップ２７１０において、ユーザは、「電話番号」列２５１０内において第１のレコード２５２０のハイフンを選択することができる。ハイフンは、ユーザが対話型ユーザインターフェイス上にいる場合、ユーザがデバイスのディスプレイ上で例えばマウスまたはジェスチャーを用いてハイフンを強調表示することによって、選択され得る。図２５Ａに示される例では、ユーザは、第１のレコード２５２０内の第１のハイフン２５１１を強調表示している。

【０１８４】

ステップ２７２０において、ユーザが第１のレコード２５２０内の第１のハイフンを強調表示した後、ユーザインターフェイスデータモデル生成器は、データレコード（例えば、図２５Ａに示すようなレコード２～２５）の各々について、ユーザによって行われた選択に対応する電話番号列内の第１のハイフンのすべてを自動的に強調表示することができる。すなわち、ユーザインターフェイスデータモデル生成器は、第１のレコード２５２０に対してユーザにより行われた強調表示に基づいて、電話番号列の残りのレコード（例えば、２～２５）に対してどのような強調表示をすべきかを判断することができる。図面では２５個のレコードが示されているが、例示的な実施形態は、２５個よりも多いかまたは少ないレコードに分割コマンドを適用することができる。さらに、データセットは、何千または数千のレコードなど、多数のレコードを含むことができる。正規表現は、ユーザによって行われた最初の強調表示に基づいて、どのような追加の強調表示を同じレコード内、または追加のレコード内で実行すべきかを判断することができる。

【０１８５】

ステップ２７３０では、強調表示が行われた後、強調表示に基づいて正規表現を生成することができる。図２５Ａは、ユーザによって実行された強調表示、および正規表現生成器によって実行された自動強調表示に基づいて生成される、生成されたレゲックス２５３２を示す。生成される正規表現は、ユーザによって行われる強調表示の変化に基づいて更新される。

【０１８６】

ステップ２７４０では、「例別レゲックス」ダイアログボックス２５３０が自動的に画面上に現れる。図２５Ａに示す例では、電話番号列２５１０のすべてのレコード（例えば、レコード１～２５）に対して強調表示が行われた後に「例別レゲックス」ダイアログボックス２５３０が現れるが、「例別レゲックス」ダイアログボックス２５３０は初期入力（例えば、ユーザによる第１のレコードの強調表示）後に現れることもできる。

【０１８７】

図２５Ａには、指定された例２５３３が示されており、ユーザは、選択されたデータを容易に強調表示するかまたは選択されたデータに対して行われる強調表示を容易に修正できる。ユーザは、「電話番号」列２５１０のデータにおける強調表示を変更すること、指定された例２５３３における強調表示を変更すること、または生成されたレゲックス２５

３２を修正することによって、強調表示を修正することができる。「電話番号」列２５１０に対してなされた変更は、生成される正規表現２５３２に反映される。

【０１８８】

ある例示的な実施形態では、最小スパニングツリーを、強調表示上ではなく、例上で用いることができる。したがって、与えられる各例に対して複数の強調表示があり、強調表示が重複する場合、最長共通サブシーケンスアルゴリズムの単一のコピーのみが、各例に対して実行される必要がある。与えられる例に対する強調表示が重複しない場合、最長共通サブシーケンスアルゴリズムは、強調表示の各々について実行することができる。

【０１８９】

例示的な実施形態では、最長スパンリストが最小スパニングツリー（ＭＳＴ）頂点に対して選択され、最初にＬＣＳキューに供給される。ＬＣＳキューは、最長共通サブシーケンスアルゴリズムが適用されるスパンを含む。複数の強調表示を表す１つの例がgraphLcs()に渡される配列に含まれている例あたりの複数の強調表示で、最も長いそのような例は、スパンの順序リストがＬＣＳキューに渡されると判断する最小スパニングツリーのグラフにおける頂点となるように選択される。

【０１９０】

ステップ２７５０において、「分割」コマンドの選択が受信される。例えば、ユーザは、例別レゲックスダイアログボックス２５３０上の「分割」ボタン２５３１を選択できる。例別レゲックスダイアログボックス２５３０に示すように、マルチ強調表示チェックボックス２５３４および強調表示モードドロップダウンリストは、分割が選択されると無効化される。

【０１９１】

例示的な実施形態では、チェックボックスは、ユーザが第２の強調表示を行うことに基づいてＵＩがチェックボックスを自動的にチェックすると、ユーザの注意を喚起するように、活性化（animated）されてもよい。ある例示的な実施形態は、ユーザがＵＩの他の場所で対話することによって引き起こされる状態の変化に基づいてチェックボックスを活性化する。チェックボックスは、サーバシステムが、画面上の他の場所におけるユーザ対話に基づいてチェックボックスにその状態を自動的に変化させた場合に、活性化される。

【０１９２】

「分割」コマンドの選択が受信された後、ステップ２７６０において、正規表現生成器２４１０の分割データ生成器２４１２は、強調表示されたデリミタに基づいて電話番号を自動的に分割することができる。図２５Ｂに示される例は、ユーザが図２５Ａの「分割」コマンド２５３１を選択した後に生ずる。

【０１９３】

図２５Ｂに示されるように、分割コマンドは、強調表示設定をマルチ強調表示に変更する。図２５Ａでは、分割コマンドの選択に先立って、マルチ強調表示モード２５３４は選択されていない。図２５Ｂでは、分割コマンドの選択後、マルチ強調表示モード２５３４が選択される。いくつかの実施形態では、「分割」コマンドが選択されると、ユーザインターフェイスは、マルチ強調表示チェックボックスおよび強調表示モードドロップダウンリストを有効にすることができる。例えば、図２５Ｂを参照すると、ユーザが「分割」コマンドをクリックすると、強調表示モードは自動的に単一レベルに変更され得、その後、マルチ強調表示テキストボックスが有効化され得る。

【０１９４】

ステップ２７７０では、分割コマンド結果を表示することができる。図２６に示されるユーザインターフェイス２６００に示されるように、電話番号列２５１０からの電話番号は、ここで３つの列２６１０，２６２０および２６３０に分割されている。列２６１０は、第１のハイフンの前に現れる電話番号の一部を含み、列２６２０は、第１のハイフンと第２のハイフンとの間に現れる電話番号の一部を含み、列２６３０は、第２のハイフンの後に現れる電話番号の一部を含む。デリミタとして機能するハイフンは２つあるため、電話番号は３つの列に分割される。一部の番号の場合、ある列は市外局番を市内局番とも

に含み得、なぜならば、市外局番は第１のハイフンの前に現れるからである。電話番号について、その電話番号内の情報に基づいて、フィールドがポピュレートされることになる。電話番号が１つのハイフンのみを有する場合、その番号に対して２列の情報のみが現れる。

【０１９５】

３つの追加の列は、図２６に示されるように、スプレッドシートビュー内に生成される。電話番号が分割されているので、ユーザは所望の情報をより容易に識別することができる。例えば、ユーザは、データレコード内の市外局番をより容易に識別することができる。また、ユーザは、列２６１０，２６２０および２６３０の各々に対して追加処理を行うことができる。したがって、ユーザはより容易にデータを利用することができる。例えば、ユーザは顧客の一次市外局番を識別することができる。

【０１９６】

分割コマンドが実行されると、実行された分割コマンド（例えば、電話列分割）は、データセットのための変換スクリプトに追加され得る。変換スクリプトは、データの列に適用されるコマンドを示す。変換スクリプトは、同じコマンドを別のデータセットに適用するために用いることができる。

【０１９７】

したがって、例示的な実施形態は、データセット内のデータを分割するための迅速かつ効率的なユーザインターフェイスを提供する。さらに、正規表現生成器は、ユーザによって望まれる結果を生成する可能性がより高い設定を識別するように構成することができる。

【０１９８】

フロー図は特定のステップで説明されるが、ステップの順序は変更することができる。例えば、ユーザにより実行される初期強調表示に基づいて正規表現を生成することができる。

【０１９９】

３．削除

図２８は、いくつかの例示的実施形態による、削除コマンドを実現するためのユーザインターフェイス２８００を示す。図２９は、いくつかの例示的実施形態による、データセット上に削除コマンドの結果を表示するユーザインターフェイス２９００を示す。削除コマンドにおいては、ユーザは、テキスト例を強調表示（または他の方法で選択）して、パターンに対応する正規表現を生成し、次いで、スプレッドシートまたは他のデータビューからパターンとマッチするデータを削除してもよい。削除機能は、データを空のストリングで置換する（例えば、削除コマンドは、生成されたレゲックスを３つのキャプチャグループに変換することによって実現することができる）。

【０２００】

図２８に示されるように、データセットがユーザインターフェイス２８００に表示される。データセットは、ストリートアドレスを含む列２８１０を含むことができる。図２８に示す例では、ユーザはアドレスからストリート番号２８１１を削除したい。削除コマンドがレコード２８２０の「street_address」の列２８１０で実施される場合、ストリートアドレス情報は、ユーザによって所望されるように削除されることができる。図２８において、ユーザは、スプレッドシートの「ストリートアドレス」列内のストリート番号を識別するためにいくつかの陽性例を選択している。これらの例は、指定された例２８３３として現れる。次いで、ユーザは「削除」ボタンをクリックして、削除操作を開始する。図２９は、ストリート番号が削除された結果のスプレッドシートを示す。したがって、削除コマンドは本質的に（空のストリング）と置換される。これは、「置換」コマンドに関して以下で論じるように、生成されたレゲックスを３つのキャプチャグループに変換することによって実現され得る。

【０２０１】

図３０は、いくつかの例示的な実施形態による、削除コマンドを実行するための方法３０００のフローチャートを示す。

【０２０２】
ステップ３０１０において、ユーザは、削除のためにアドレス情報の一部を選択することができる。図２８に示す例では、第１レコード２８２０について、アドレス情報のストリート番号部分２８１１が選択されている。アドレス情報のストリート番号部分２８１１は、ユーザが対話型ユーザインターフェイス上にいる場合、ユーザがデバイスのディスプレイ上で例えばマウスまたはジェスチャーを用いてストリート番号部分を強調表示することによって、選択され得る。
【０２０３】
ステップ３０２０において、ユーザが第１のレコード２８２０内のストリート番号部分２８１１を強調表示した後、ユーザインターフェイスデータモデル生成器は、データレコード（例えば、図２８に示すレコード２～２５）の各々について、「street_address」列２８１０内のストリート番号部分のすべてを自動的に強調表示することができる。すなわち、ユーザインターフェイスデータモデル生成器は、第１のレコード２８２０に対して実行される強調表示に基づいて、「street_address」列２８１０内の残りのレコード（例えば、２～２５）に対してどのような強調表示が実行されるべきかを判断することができる。図面では２５個のレコードが示されているが、例示的な実施形態は、２５個よりも多いかまたは少ないレコードに削除コマンドを適用することができる。正規表現は、ユーザによる最初の強調表示に基づいて、どのような追加の強調表示をすべきかを判断することができる。
【０２０４】
ステップ３０３０では、正規表現が生成される。図２８には、生成された正規表現２８３２が示されている。生成された正規表現により、ユーザは、どのデータを選択するかを容易に強調表示することができる。削除コマンドの場合、生成された正規表現により、ユーザは、どのデータを削除するかを容易に強調表示することができる。生成された正規表現は、ユーザおよび正規表現生成器によって行われる強調表示に対応する。ユーザは、「street_address」列２８１０のデータにおける強調表示を変更することによって、または正規表現２８３２における強調表示を変更することによって、強調表示を修正することができる。「street_address」列２８１０に対する変更は、生成される正規表現２８３２に反映される。生成される正規表現は、ユーザによって行われる強調表示の変更に基づいて更新される。
【０２０５】
ステップ３０４０では、「street_address」列２８１０のすべてのストリート番号に対して強調表示を行った後、「例別レゲックス」ダイアログボックス２８３０を画面上に自動的に表示することができる。図２８に示す例では、「street_address」列２８１０のすべてのレコード（例えば、レコード１～２５）に対して強調表示が行われた後に「例別レゲックス」ダイアログボックス２８３０が現れるが、「例別レゲックス」ダイアログボックス２８３０は初期入力（例えば、第１のレコードの強調表示）後に現れることもできる。指定された例２８３３は、ユーザによって与えられる例を識別する。
【０２０６】
ステップ３０５０では、削除コマンドの選択を受け付けることができる。例えば、ユーザは、「例別レゲックス」ダイアログボックス２８３０において、「削除」ボタン２８３１を選択することができる。
【０２０７】
削除コマンドの選択を受け付けた後、ステップ３０６０において、正規表現生成器２４１０の削除データ生成器２４１３は、強調表示に基づいてストリート番号を自動的に削除することができる。図２８に示す例では、「street_address」レコードのすべてが強調表示を含んでいるため、図２８に示すレコードの各々に対して削除コマンドが適用される。
【０２０８】
ステップ３０７０において、削除結果を表示することができる。図２９に表示されるユーザインターフェイス２９００に示されるように、「street address」列２８１０のスト

リートアドレスからのストリート番号は除去されている。ストリート番号は、適所で削除される。

【０２０９】

削除コマンドが実行された後、実行された削除コマンド（例えば、ストリートアドレス列削除）は、データセットのための変換スクリプトに追加され得る。変換スクリプトは、データの列に適用されるコマンドを示す。変換スクリプトは、同じコマンドを別のデータセットに適用するために用いることができる。

【０２１０】

したがって、例示的な実施形態は、データセット内のデータを削除するための迅速かつ効率的なユーザインターフェイスを提供する。

【０２１１】

４．難読化

図３１は、いくつかの例示的実施形態による、難読化コマンドを実現するためのユーザインターフェイス３１００を示す。図３２は、いくつかの例示的実施形態による、データセット上に難読化コマンドの結果を示すユーザインターフェイス３２００を示す。

【０２１２】

難読化コマンドでは、ユーザは、テキスト例を強調表示（または他の方法で選択）して、パターンに対応する正規表現を生成し、次いで、スプレッドシートまたは他のデータビューからパターンとマッチするデータを難読化（例えば、削除するのではなく曖昧化）してもよい。データは、レコード内のデータのすべてが見えないように難読化され得る。例えば、ユーザは、セキュリティまたはプライバシーの理由から情報を難読化することを望む場合がある。

【０２１３】

図３１において、ユーザは、スプレッドシートの「ssn」列３１１０内の中央の２桁３１１２を選択している。次に、ユーザは「難読化」ボタンをクリックして難読化操作を開始する。図３２は、「ssn」列全体の中央の２桁を２つのポンド符号に置き換えた結果のスプレッドシートを示す。

【０２１４】

図３１に示されるように、データセットがユーザインターフェイス３１００に表示される。データセットは、例えば社会保障番号に向けられた列３１１０を含む。難読化コマンドを示すために社会保障番号が用いられているが、データの列内のデータのいずれでも、ユーザによって望まれるように難読化され得る。例えば、ユーザは、クレジットカード情報または銀行口座情報など、任意の種類の取り扱いに注意を要する情報を難読化することを望む場合がある。

【０２１５】

社会保障番号は、第１のフィールド３１１１、第２のフィールド３１１２、および第３のフィールド３１１３に分けることができる。第１のフィールド３１１１は、第１のダッシュの前にあり得、第２のフィールド３１１２は、ダッシュ間にあり得、第３のフィールド３１１３は、社会保障番号の第２のダッシュの後にあり得る。図３１に示される例では、ユーザは、社会保障番号のすべてが見えないように、社会保障番号の第２のフィールド３１１２（例えば、第１のダッシュと第２のダッシュとの間の数字）を難読化すること望む。第２のフィールドが難読化されているが、ユーザは社会保障番号内のフィールドのいずれかまたはすべてを難読化することを選択することができる。

【０２１６】

図３３は、いくつかの例示的実施形態による、難読化コマンドを実行するための方法３３００のフローチャートを示す。

【０２１７】

ステップ３３１０において、社会保障番号列３１１０内の第２のフィールド３１１２の選択が受け付けられる。選択されたフィールド３１１２は、第１のレコード３１２０について選択される。第２のフィールド３１１２は、ユーザが対話型ユーザインターフェイス

上にいる場合、デバイスのディスプレイ上で例えばマウスまたはジェスチャーを用いることによって、選択され得る。ユーザによって選択された第２のフィールドは、第１の色で強調表示され得る。ユーザによってなされた選択は、ダイアログボックス３１３０内の指定された例３１３３として与えられる。

【０２１８】

ステップ３３２０では、社会保障番号列内の社会保障番号のすべてを、ステップ３３２０において実行される強調表示に対応するよう強調することができる。すなわち、列３１１０の社会保障番号内のすべての第２のフィールドは、第１の色で強調表示され、列３１１０の社会保障番号内のすべての第１のフィールド３１１１および第３のフィールド３１１３は、第２の色で強調表示されることになる。フィールドは、ユーザが修正されるフィールドと修正されないフィールドを容易に区別できるように、異なる色で強調表示される。

【０２１９】

ユーザインターフェイスデータモデル生成器は、すべてのレコード（例えば、２～２５）を、第１のレコード３１２０に対してユーザによって実行された強調表示に対応するように、強調表示することができる。すなわち、ユーザインターフェイスデータモデル生成器は、第１のレコード３１２０に対して実行された強調表示に基づいて、社会保障番号列内の残りのレコード（例えば２～２５）に対してどのような強調表示が実行されるべきかを判断することができる。図面では２５個のレコードが示されているが、例示的な実施形態は、２５個よりも多いかまたは少ないレコードに難読化コマンドを適用することができる。

【０２２０】

ステップ３０３０では、正規表現を生成することができる。図３１には、生成された正規表現３１３２が例別ダイアログボックス３１３０に示されている。生成された正規表現により、ユーザは、どのデータを難読化すべきかを容易に強調することができる。ユーザは、「ssn」列３１１０のデータにおける強調表示を変更することによって、または生成された正規表現３１３２における強調表示を変更することによって、強調表示を修正することができる。「ssn」列３１１０への変更は、生成される正規表現３１３２に反映される。すなわち、ユーザが行ったデータセットに対する強調表示の変更に基づいて、生成される正規表現が更新される。

【０２２１】

ステップ３３４０において、社会保障番号に対して強調表示が行われた後、「例別レゲックス」ダイアログボックス３１３０が画面上に自動的に現れることができる。図３１に示す例では、社会保障番号列３１１０のすべてのレコード（例えば、レコード１～２５）に対して強調表示が行われた後に「例別レゲックス」ダイアログボックス３１３０が現れるが、「例別レゲックス」ダイアログボックス３１３０は初期入力（例えば、第１のレコードの強調表示）後に現れることもできる。「例別レゲックス」ダイアログボックス３１３０は、ユーザにより入力される指定された例３１３３を含む。また、「例別レゲックス」ダイアログボックス３１３０は、生成された正規表現３１３２を含む。

【０２２２】

ステップ３３５０において、「難読化」コマンドの選択が受け付けられる。例えば、ユーザは、「例別レゲックス」ダイアログボックス３１３０において、「難読化」ボタン３１３１を選択することができる。図３１に示すように、ユーザは、難読化ボタン３１３１を選択することができる。

【０２２３】

ステップ３３６０において、難読化コマンドが受信された後、正規表現生成サーバ２４００の難読化生成器２４１４は、「ssn」列３１１０内の社会保障番号の第２のフィールドのすべてを自動的に難読化することができる。

【０２２４】

ステップ３３７０では、難読化結果を表示することができる。図３２は、いくつかの例示的実施形態による、データセット上に難読化コマンドの結果を表示するユーザインター

フェイス３２００を示す。図３２に示すように、ユーザによって選択された第２のフィールド３１１２は、難読化されるべき強調表示されたフィールドの代わりに「##」に置き換えられる。したがって、社会保障列３１１０内の社会保障番号の第２のフィールド３１１２は、すべて、「##」に置換されている。すなわち、すべての社会保障番号に対する第２のフィールド３１１２が難読化されている。

【０２２５】

難読化コマンドが実行された後、実行された難読化コマンド（例えば、obfuscate_column_ssn）は、データセットのための変換スクリプト３２３０に追加され得る。変換スクリプトは、データの列に適用されるコマンドを示す。変換スクリプトは、同じコマンドを別のデータセットに適用するために用いることができる。

【０２２６】

したがって、例示的な実施形態は、データセット内のデータを難読化するための迅速かつ効率的なユーザインターフェイスを提供し、ユーザ情報のプライバシーを維持することができる。

【０２２７】

５．置換

いくつかの例示的実施形態に従い、図３４は、置換コマンドを実現するためのユーザインターフェイス３４００を示し、図３５は、置換コマンドを実現するためのユーザインターフェイス３５００を示す。置換コマンドにおいては、ユーザは、テキスト例を強調表示（または他の方法で選択）して、パターンに対応する正規表現を生成し、次いで、パターンにマッチするテキストを他の選択されたテキストと置換することができる。

【０２２８】

置換コマンドは、ダイアログ内に例の前後を示す２列テーブルを動的に表示することを含む。置換表現におけるキャプチャグループのコンテンツを利用するために、ユーザはJava（登録商標）およびJavascriptレゲックスＡＰＩからの標準的なレゲックス置換構文である$1を用いることができる。

【０２２９】

図３７は、いくつかの例示的な実施形態による、置換コマンドを実行するための方法３７００のフローチャートを示す。

【０２３０】

ステップ３７１０において、ユーザは、「street_address」列２８１０内で、ユーザが他のデータで置換することを望むフィールドを選択することができる。図３４に示す例において、ユーザは、ストリートアドレスの「Drive」を「Dr.」に置換することを望む。図３４に示すように、ユーザは、データセットの第５のレコード３４２０から「Drive」を選択することができる。置換されるべきデータは、ユーザが対話型ユーザインターフェイス上にいる場合、ユーザがデバイスのディスプレイ上で例えばマウスまたはジェスチャーを用いて置換されるべきデータを強調表示することによって、選択され得る。

【０２３１】

ステップ３７２０において、ユーザが、第５のレコードにおいて置換されるべきデータ（例えば、「Drive」）を強調表示した後、ユーザインターフェイスデータモデル生成器は、第５のレコード３４２０において実行された強調表示に対応する、残りのレコード内のすべてのフィールドを自動的に強調表示することができる。ユーザインターフェイスデータモデル生成器は、「Drive」を含むストリートアドレスを自動的に強調表示することができる。図３４に示すように、レコード７，１３，１６，１８，および２０には「Drive」が含まれているため、レコード７，１３，１６，１８，および２０において、「Drive」が自動的にユーザインターフェイスデータモデル生成部によって強調表示される。「Drive」は、レコード５，７，１３，１６，１８，および２０において第１の色で強調表示され得る。「Drive」を含むストリートアドレスの残りは、第１の色とは異なる第２の色で強調表示され得る。したがって、置換されるべき部分を容易に識別することができる。

【０２３２】

ステップ３７３０において、正規表現３４３２が生成され得る。図３５に示すように、３つの部分からなる生成された正規表現３５３２が示されている。３つの部分からなる生成された正規表現により、ユーザは、データの「Drive」部分、「Drive」よりも前の部分、または「Drive」よりも後の部分を容易に置換することができる。この例では、ユーザは、データの「Drive」部分を「Dr」と置き換え、ユーザは、ポップアップ「例別レゲックス」ダイアログボックス３５３０のプレビュースプレッドシートで結果を直ちに見ることができる。

【０２３３】

図３４において、ユーザは、スプレッドシートの「street_address」列から十分な数の例３４３３（陽性および陰性）を選択して、「Drive」で終わるアドレスに対応する正規表現を生成している。アドレス３４３４は陽性例であり、アドレス３４３５は陰性例である。

【０２３４】

ステップ３７４０では、すべてのレコードに対して強調表示が行われた後、「例別レゲックス」ダイアログボックス３４３０が画面上に自動的に現れる。図３４に示す例では、「例別レゲックス」ダイアログボックス３４３０は、置換されるべきデータを含むすべてのレコードに対して強調表示が行われた後に現れるが、「例別レゲックス」ダイアログボックス３４３０は、初期入力（例えば、第５のレコードにおける「Drive」の強調表示）の後に現れることもできる。「例別レゲックス」ダイアログボックス３４３０は、指定された例３４３３と、生成された正規表現３４３２とを含むことができる。アドレス３４３４は陽性例であり、アドレス３４３５は陰性例である。

【０２３５】

ステップ３７５０において、「置換」コマンドの選択が受け付けられる。例えば、ユーザは、「例別レゲックス」ダイアログボックス３４３０上の「置換」ボタン３４３１を選択して、置換コマンドを開始することができる。

【０２３６】

ステップ３７６０では、パネル３５３８および「～と置換」フィールド３５３７を表示して、どの情報が置換されているかを判断することにおいてユーザが支援することができる。図３５に示されるように、パネル３５３８内で、ユーザは、初期アドレス、およびそのアドレスが置換されたときにどのように現れるかを見ることができる。例えば、アドレス内の「Drive」という語は、要素３５３９によって示されるように、アドレス内に「Dr.」として現れる。パネル３５３８は、置換コマンドが適用された場合に、データがどのように現れるかに関するプレビューをユーザに提供することにおいてユーザを支援することができる。

【０２３７】

図３５に示されるように、レゲックス３５３２は、三重キャプチャグループに変換される。生成されたレゲックスは「^(.*?)(D[a-z]+)()$」である。したがって、ユーザは、強調表示前（^(.*?)）、強調表示上（(D[a-z]+)）、強調表示後（()）にどのようなデータが出現するかを容易に見ることができる。図３５に示す例では、強調表示後はデータが現れない（()）。強調表示上のデータは、ユーザによって選択されるデータである。この例では、強調表示上のデータは、ユーザが置換のために選択したデータである「Drive」である。

【０２３８】

ユーザはまた、選択された語が何と置き換えられるかを識別できる。「～と置換」フィールド３５３７は、選択された語（例えば「Drive」）が置換されるべき語（例えば、「Dr」）を識別する。ユーザは、「～と置換」フィールド３５３７において置換語を修正することができる。「～と置換」フィールド３５３７においてユーザによって選択された置換語は、強調表示されたレコードに適用される。ユーザが置換に同意すれば、ユーザは作成ボタン３５３６を選択し、置換データ生成器２４１５は置換を行うことができる。

【０２３９】

上に示されるように、置換コマンドはまた、ダイアログ内に、例の前後を示す２列テーブルを動的に表示することを含み得る（例えば、ポップアップ「例別レゲックス」ダイアログボックスのプレビュースプレッドシート）。置換表現におけるキャプチャグループのコンテンツを利用するために、ユーザはJavaおよびJavascriptレゲックスＡＰＩからの標準的なレゲックス置換構文である＄１を用いることができる。さらに、ユーザが置換コマンドを選択し、ユーザがキャプチャグループを生成するモード（すなわち、ネスト化-自動-外側）にある場合、ダイアログに表示される生成されたレゲックスは、合計３つ、すなわち強調表示の前、強調表示上、および強調表示の後のための追加の２つのキャプチャグループを得る。次いで、ユーザは、置換表現内においてこれらを$1、$2および$3として言及することができる。さらに、キャプチャグループがない場合、システムは、生成されたレゲックス全体をキャプチャグループにラップすることができ、ユーザは、$1を用いて元のテキストを参照することができる。

【０２４０】

例示的な実施形態で開示されるように、強調表示モードは、ネスト化自動、ネスト化手動、および単一レベルを含むことができる。ネスト化自動は、ネスト化（自動外側）としても知られる。ネスト化手動は、ネスト化（完全制御）としても知られる。

【０２４１】

ステップ３７７０では、置換を行うことができる。「作成」ボタン３５３６をクリックし変更を確定した後、データが置換される。

【０２４２】

ステップ３７８０では、置換結果を表示することができる。図３６は、いくつかの例示的実施形態による、置換コマンドの結果をデータセットに表示するユーザインターフェイス３６００を示す。図３６に示されるユーザインターフェイス３６００に示されるように、「Drive」を含むレコードは、今、「Dr.」と置換される。これにより、「street_address」列２８１０における「Drive」を含むレコードは、「Dr.」と置換される。

【０２４３】

置換コマンドが実行された後、実行された置換コマンド（例えば、ストリートアドレス列置換）は、データセットのための変換スクリプトに追加され得る。変換スクリプトは、データの列に適用されるコマンドを示す。変換スクリプトは、同じコマンドを別のデータセットに適用するために用いることができる。

【０２４４】

したがって、例示的な実施形態は、データセット内のデータを置換するための迅速かつ効率的なユーザインターフェイスを提供する。

【０２４５】

６．行のフィルタリング

図３８および図３９は、いくつかの例示的実施形態による、行フィルタリングコマンドを実現するためのユーザインターフェイス３８００および３９００を示す。図４０は、いくつかの例示的実施形態による、データセット上に行フィルタリングコマンドの結果を表示するユーザインターフェイス４０００を示す。

【０２４６】

フィルタリング操作において、ユーザは、テキスト例を強調表示（または他の方法で選択）して、パターンに対応する正規表現を生成し、次いで、データをフィルタリングして、パターンとマッチするデータを含む（または除外する）ようにすることができる。「行フィルタリング」操作の例を図３８、図３９、図４０および図４１に示す。

【０２４７】

図４１は、いくつかの例示的な実施形態による、行フィルタリングコマンドを実行するための方法４１００のフローチャートを示す。図４１に示される例では、ユーザは、ストリートアドレスにおいて「Avenue」を有するレコードを識別するようにデータセット内のレコードをフィルタリングすることを望む。

【０２４８】

ステップ４１１０において、ユーザは、レコードをフィルタリングするために用いられるデータを選択することができる。図３８に示す例では、ユーザは、「street address」列２８１０から「Avenue」を選択している。データは、ユーザがフィルタリングに用いられるデータを強調表示することによって、選択することができる。例えば、ユーザが対話型ユーザインターフェイス上にいる場合、データは、マウスを用いて、またはデバイスのディスプレイ上でジェスチャーを介して、選択することができる。

【０２４９】

ステップ４１２０において、ユーザがフィルタリングに用いられるデータを強調表示した後、ユーザインターフェイスデータモデル生成器は、他のレコード内の対応するデータを自動的に強調表示することができる。例えば、ユーザインターフェイスデータモデル生成器は、レコード８、１３、および１９において「Avenue」という語を強調表示し得る。すなわち、ユーザインターフェイスデータモデル生成器は、第１のレコード３８２０に対して実行される強調表示に基づいて、「street_address」列２８１０の残りのレコード（例えば、２～２５）に対してどのような強調表示が実行されるべきかを判断することができる。

【０２５０】

ステップ４１３０では、ユーザによって選択された強調表示に基づいて正規表現を生成することができる。図３９は、生成された正規表現を含むダイアログボックス３９３０を示す。

【０２５１】

図３９には、生成された正規表現３９３２が示されており、ユーザは、フィルタリングの基礎として用いられるデータを容易に識別することができる。図３９では、３つの例がユーザによって強調表示され、指定された例３９３３に表示されている。したがって、指定された例３９３３には、３つのアドレスが現れる。ユーザは、スプレッドシートの「street_address」列から十分な数の例３９３３（陽性３９３４および陰性３９３５）を選択して、「Avenue」で終わるアドレスに対応する正規表現を生成している。３つの例が示されているが、ユーザは、ユーザによって望まれる結果に基づいて１つまたは複数の例を用いることができる。

【０２５２】

陽性例および陰性例は、異なる色で、または陽性例および陰性例がダイアログボックス３９３０内において、およびユーザインターフェイス上に表示されるデータセット内において、異なるように表示される、任意の方法で、表示され得る。図３９に示す指定された例３９３３では、「street address」列２８１０から陰性例として、「street」を含む２つのアドレスがユーザによって選択される。すなわち、それらは、ユーザがレコードに出現することを望まないデータの例である。しかしながら、追加の例がユーザによって与えられる場合、フィルタリングに適用される、より正確なレゲックス表現を生成することができる。

【０２５３】

ユーザは、「street_address」列２８１０内のデータにおける強調表示を変更することによって、または正規表現３９３２における強調表示を変更することによって、強調表示を修正することができる。「street_address」列２８１０に対する変更は、生成される正規表現３９３２に反映される。他の例では、ユーザは、同様の技法を用いて、（保持されるべきではなく）フィルタリング除去されるべきパターンに対応する正規表現を定義してもよい。

【０２５４】

ステップ４１４０では、強調表示が行われた後、「例別レゲックス」ダイアログボックス３９３０が自動的に画面上に現れる。図３９に示す例では、「street_address」列２８１０のすべてのレコード（例えば、レコード１～２５）に対して強調表示が行われた後に「例別レゲックス」ダイアログボックス３９３０が現れるが、「例別レゲックス」ダイアログボックス３９３０は初期入力（例えば、第１のレコードの強調表示）後に現れること

もできる。
【０２５５】
ステップ４１５０において、「行フィルタリング」コマンドの選択が受け付けられる。例えば、ユーザは、「例別レゲックス」ダイアログボックス３９３０において、「行フィルタリング」ボタン３９３１を選択することができる。ユーザは、ユーザインターフェイス内の「行フィルタリング」ボタン３９３１をクリックして、図３８に示されるように、すべての他のタイプのアドレスをフィルタリング除去して「Avenue」アドレスのみを残すフィルタオプションを開始することができる。
【０２５６】
ステップ４１６０では、行フィルタリングコマンドに関する確定が受け付けられる。具体的には、例３９３３によって識別されるレコードを保持する（保持ボタン３９３７）か、例３９３３によって識別されるレコードを削除する（削除ボタン３９３８）かの選択を受け付ける。「保持」ボタン３９３４または「削除」ボタン３９３５を選択した後、ユーザは、作成ボタン３９３６を選択してフィルタリングを開始することができる。
【０２５７】
ステップ４１７０において、正規表現生成器２４１０の行フィルタリングデータ生成器２４１６は、ユーザによって入力されたフィルタリング基準に従ってレコードを自動的にフィルタリングすることができる。
【０２５８】
ステップ４１８０では、行フィルタリングの結果を表示する。図４０に示されるユーザインターフェイス４０００に示されるように、レコードは、アドレスに「Avenue」を有するレコードのみが示されるようにフィルタリングされている。この例では、スプレッドシートは２５を超えるレコードを含む。説明のため、「street_address」に「Avenue」を有する最初の２５個のレコードが示されている。
【０２５９】
行フィルタリングコマンドが実行されると、実行された行フィルタリングコマンド（たとえば、ストリートアドレス列行フィルタリング）は、データセットのための変換スクリプトに追加され得る。変換スクリプトは、データの列に適用されるコマンドを示す。変換スクリプトは、同じコマンドを別のデータセットに適用するために用いることができる。
【０２６０】
したがって、例示的な実施形態は、データセット中のデータの行をフィルタリングするための迅速かつ効率的なユーザインターフェイスを提供する。
【０２６１】
ＶＩ．陰性例のコンテキスト
例示的な実施形態では、ネスト化手動における陰性例に対して、陽性例に対応するコンテキストについて検索を実行することができる。
【０２６２】
ユーザインターフェイスがネスト化手動モードにある場合、「Windows NT 6.1」の外側強調表示および「6.1」の内側強調表示を有する例が受信され得る。例えばユーザによって与えられる初期例に応答して、ＵＩは「Windows NT 6.0」を強調表示するかもしれない。ＵＩは、正規表現生成器を介して、「Windows NT 6.0」を強調表示し得、なぜならば、「Windows NT 6.0」は、例として最初に与えられた「Windows NT 6.1」とのマッチであると判断されるからである。次いで、ユーザは、「Windows NT 6.0」を選択して、「Windows NT 6.0」が「Windows NT 6.1」との所望のマッチではないことを示してもよい。すなわち、ユーザは、「Windows NT 6.1」を選択することにより、陰性例を与えることができる。ユーザが「Windows NT 6.0」をクリックして陰性例を作成する場合、ＵＩは、その陰性例として「6.0」だけを送信するかもしれない。したがって、例示的な実施形態によれば、ＵＩは、ＬＣＳを実行する前にマッチ「Windows NT」コンテキストを検索することができる。例示的な実施形態は、与えられる陰性例に基づいてコンテキストを検索する。

【０２６３】
図５０は、いくつかの例示的な実施形態による、陰性例を与えるための方法５０００のフローチャートを示す。
【０２６４】
ステップ５０１０において、初期選択を受け付けることができる。初期選択に基づいて正規表現が生成される。
【０２６５】
例えば、ユーザは、「Windows NT 6.1」の外側強調表示および「Windows NT 6.1」の「6.1」の内側強調表示を選択することができる。外側強調表示および内側強調表示は、データを強調表示することにより作成することができる。外側強調表示の場合、ユーザは「Windows NT 6.1」を強調表示することができ、内側強調表示の場合、ユーザは既に強調表示された「Windows NT 6.1」の「6.1」を強調表示することができる。外側強調表示はアプリケーション名に対応し、内側強調表示はバージョン番号に対応する。外側強調表示は、内側強調表示とは異なる色で現れることができる。強調表示は、ユーザが対話型ユーザインターフェイス上にいる場合、デバイスのディスプレイ上で例えばマウスまたはジェスチャーを用いることによって、実行することができる。
【０２６６】
ステップ５０２０において、ユーザが所望のデータフラグメントを強調表示（例えば、内側および外側強調表示）した後、ユーザインターフェイスデータモデル生成器は、同じレコードまたは残りのレコード内の対応するデータを自動的に強調表示することができる。例えば、ユーザインターフェイスデータモデル生成器は、レコード内の「6.1」のすべてのインスタンスを強調表示することができる。しかしながら、正規表現生成器は、データセット中のレコードから「Windows NT 6.0」のすべてのインスタンスを強調することもできる。したがって、結果において高い精度を保証するために、ユーザは陰性例を与えることができる。
【０２６７】
ステップ５０３０において、ユーザは陰性例を与えることができる。ユーザは、ＵＩによって自動的に強調表示されたアプリケーションおよびバージョン番号のすべてを望むわけではないため、陰性例を与えてもよい。例えば、ユーザは、陰性例を与えるために「Windows NT 6.0」を選択してもよい。次いで、ＵＩは「6.0」のバージョン番号を含むすべてのレコードを識別し得る。しかしながら、これは「6.0」バージョン番号を含む「Windows NT」以外のアプリケーションを識別する結果となり得る。
【０２６８】
ステップ５０４０において、ＵＩは、陰性例からコンテキストを判断する。「6.0」のバージョン番号を含むすべてのバージョン番号を強調表示する代わりに、ＵＩは、ユーザによって与えられた例からコンテキストを判断する。すなわち、ＵＩは、データレコードからバージョン番号「6.0」を識別するだけでなく、陰性例を含むレコードを識別するときに、レコードからアプリケーション名「Windows NT」を識別する。したがって、ＵＩは、陰性例を含むレコードを識別するときに単に「6.0」を含むレコードを検索する代わりに、陰性例を含むレコードを識別するときに「Windows NT 6.0」を検索する。
【０２６９】
陰性例からコンテキストを判断することは、図５１に関して以下でより詳細に説明される。
【０２７０】
ステップ５０５０において、データセット上の強調表示は、識別されたコンテキストに従って更新される。ＵＩは、「6.0」だけではなく「Windows NT 6.0」含む語を強調し、それによって陰性例をより正確にする。
【０２７１】
生成される正規表現は、陰性例から識別されるコンテキストに基づいて更新され得る。
したがって、例示的な実施形態は、陰性例を与える、より正確な方法を提供する。コン

テキストは、１つまたは複数の正規表現コードの異なるセットによって共有される最長共通サブシーケンスを判断するようＬＣＳアルゴリズムを実行する前に判断される。与えられる例からコンテキストを判断することによって、より正確な正規表現を生成することができる。

【０２７２】

図５１は、いくつかの例示的な実施形態による、陰性例からコンテキストを判断するための方法５１００のフローチャートを示す。図５１は、図５０のステップ５０４０をより詳細に説明している。

【０２７３】

ステップ５１１０では、陰性例が与えられた後、複数の強調表示があるかどうか、およびそれらの複数の強調表示がネスト化されているかどうか（内側＋外側強調表示）が判断される。図５１に示されるように、陰性例のコンテキストは、陰性例が与えられる（例えば、図５０のステップ５０３０）のでなければ、判断されない。

【０２７４】

ステップ５１２０において、コンテキストが、陰性例の左側のデータから取得される。具体的には、コンテキストは、陰性の強調表示が埋め込まれている場所の左側のデータから得られる。陰性の強調表示の埋め込まれた位置は、コンテキストを取得する前に判断され得る。このステップを実行するためのコードは、val eLookBehindStart = r.s.slice(0, r.highlightSpans.min).foldRight(es.slice(0, highlightSpanRange.start).reverse)((sElem,esLookBehind) =〉 esLookBehind.dropWhile(_.cc == sElem.cc)).length.を含むことができる。

【０２７５】

ある例示的実施形態においては、最も遠い左のスパンが用いられる。eLookBehindStartは、左側歩行から（陰性の強調表示の左縁から）最も左のスパンのスパン番号を得る。

【０２７６】

ステップ５１２０は、反復プロセスであり得る。例えば、フィルタリングは、陰性の強調表示が埋め込まれる場所の左側の各スパンについて実行され得る。データセットのフラグメントの強調表示は、陰性例のスパンとマッチしないフラグメントの各スパンについて除去され得る。すべての反復において、考慮中のスパンのリストは短くされることになる。

【０２７７】

ステップ５１３０において、陰性例の右側のデータからコンテキストが取得される。具体的には、陰性の強調表示が埋め込まれている場所の右側のデータからコンテキストを取得する。本ステップに対するコードは、val eLookAheadEnd = r.s.slice(r.highlightSpans.max+1, r.s.length).foldLeft(es.slice(highlightSpanRange.end, es.length))((esLookAhead,sElem) =〉 esLookAhead.dropWhile(_.cc == sElem.cc)).lengthを含むことができる。

【０２７８】

ある例示的実施形態においては、最も遠い右のスパンが用いられる。eLookAheadEndは、右側歩行から（陰性の強調表示の右縁から）最も右のスパンのスパン番号を得る。

【０２７９】

ステップ５１３０は、反復プロセスであり得る。例えば、フィルタリングは、陰性の強調表示が埋め込まれる場所の右側のフラグメントの各スパンに対して実行され得る。データセットのフラグメントの強調表示は、陰性例のスパンとマッチしないフラグメントごとに除去され得る。すべての反復において、考慮中のスパンのリストは短くされることになる。

【０２８０】

ステップ５１４０では、フラグメントのフィルタリングが実行される。陰性例の左側および右側のコンテキスト（たとえば、ステップ５１２０および５１３０で識別されるコンテキスト）に対応しない強調表示されたフラグメントが除去される。

【０２８１】

コンテキストは、陰性例の左側および右側で用いられるが、例えば、左側コンテキストデータのみまたは右側コンテキストデータのみが存在する場合、左側または右側コンテキストのみが、陰性例についてコンテキストを識別するために用いられてもよい。また、図５１で説明した例では、左側コンテキストを右側コンテキストの前に識別しているが、右側コンテキストを左側コンテキストの前に識別してもよい。さらに、上記の例は、陰性例に関してコンテキストを判断することに関して説明されているが、陽性例に関してコンテキストを判断することもできる。

【０２８２】

ネスト化手動モードにおいて陰性例に対応するコンテキストを検索するための例示的なコードは、以下のコードの一部または全部を含むことができる：

【０２８３】

【数２６】

```
val numberedExamplesWithContextualizedNegative =

if (mustUseLookAheadAndBehind && examplesScala.exists(_.highlights.exists(h
=> !h.isPositive && h.outer == h.inner))) {

val r = shortestRegEx(flaggedPosOnly.filter(_.e.isPositive),
inductParameters.isStartAligned, inductParameters.isEndAligned, inductParameters.useContext,
mustUseLookAheadAndBehind, true)

numberedExamples.map(e => if (!r.r.isSuccess || e.isPositive) e else {

val es = toSpans(e.s)

val highlightSpanRange = highlightSpanRangeForExample(es, e.highlight)

val eLookBehindStart = r.s.slice(0, r.highlightSpans.min).foldRight(es.slice(0,
highlightSpanRange.start).reverse)((sElem,esLookBehind) => esLookBehind.dropWhile(_.cc ==
sElem.cc)).length

val eLookAheadEnd = r.s.slice(r.highlightSpans.max+1,
r.s.length).foldLeft(es.slice(highlightSpanRange.end, es.length))((esLookAhead,sElem) =>
esLookAhead.dropWhile(_.cc == sElem.cc)).length

val lookBehindLength = (eLookBehindStart to highlightSpanRange.start-1).filter(_ >=
0).map(es(_).orig.head.s.length).sum

val lookAheadLength = (highlightSpanRange.end+1 to eLookAheadEnd).filter(_ <
es.length).map(es(_).orig.head.s.length).sum
```

【０２８４】

【数２７】

```
    val start = e.highlight.start - lookBehindLength
    val end = e.highlight.end + lookAheadLength
    NumberedExample(e.id, e.s.substring(start, end+1), e.isPositive, start to end)
  })
}
else
  numberedExamples
```

【０２８５】

ＶＩＩ．スパン上で最長共通サブシーケンスアルゴリズムを用いた正規表現生成

本明細書で説明されるさらなる態様は、１つまたは複数のデータ入力キャラクタシーケンスからのＬＣＳアルゴリズムに基づく正規表現の生成に関するが、正規表現生成器１１０は、例のいくつかのみに存在するキャラクタを取り扱うこともできる。いくつかの入力例においてのみ存在するキャラクタを取り扱うために、正規表現コードの最小発生数および最大発生数の両方が追跡されるスパンが定義され得る。たとえば、「9pm」および「9 pm」のキャラクタシーケンス入力については、数字と「pm」テキストとの間に任意選択の空白が存在する。そのような場合、所与の入力例のすべてに一定のスパン（たとえば「9」と「pm」との間の単一の空白）が存在しないかもしれない場合、最小発生数はゼロに設定されてもよい。次いで、これらの最小数および最大数は、正規表現マルチプリシティ構文にマッピングされ得る。最長共通サブシーケンス（ＬＣＳ）アルゴリズムを、すべての入力例に現れない「任意選択の」スパン（たとえばゼロの最小長さ）を含む、入力例から導出されたキャラクタのスパン上で実行してもよい。以下で説明するように、連続するスパンは、ＬＣＳアルゴリズムの実行中にマージされてもよい。そのような場合において、一緒に担持される追加の任意選択のスパンが連続して出現することに終わるとき、ＬＣＳアルゴリズムは、それらの任意選択のスパン上でも同様に再帰的に実行されてもよい。すなわち、ＬＣＳアルゴリズムの実行は、その性質上、再帰的であるが、これらの場合、ＬＣＳアルゴリズム全体を、再帰的に実行してもよい（たとえば、再帰的ＬＣＳアルゴリズムを再帰的に実行する）。他の技術的利点の中でもとりわけ、これは、より短く、よりクリーンで、より可読性のある正規表現生成を可能にし得る。たとえば、(am｜ am)（すなわち、amの前に任意選択の空白を有する）は、ＬＣＳアルゴリズムを再帰的に実行せずに生成されるかもしれず、一方、ＬＣＳアルゴリズムを再帰的に実行すると、正規表現は、より短く、よりクリーンな( ?am)として生成される結果となり得る。

【０２８６】

図１８は、本明細書で説明する１つまたは複数の実施形態による、最長共通サブシーケンス（ＬＣＳ）アルゴリズムを使用して、任意選択のスパンを含む正規表現を生成するための処理１８００を示すフローチャートである。ステップ１８０１において、正規表現生成器１１０は、陽性の正規表現例に対応する１つまたは複数のキャラクタシーケンスを入力データとして受け取ってもよい。ステップ１８０２において、正規表現生成器１１０は、キャラクタシーケンスを正規表現コードに変換してもよい。したがって、ステップ１８０１およびステップ１８０２は、上述の、先の対応する例と同様でも同一でもよい。次いで、ステップ１８０２において、正規表現コードは、スパンデータ構造（またはスパン）にさらに変換され得る。上述したように、各スパンは、キャラクタクラスコード（たとえばレゲックスコード）および繰り返しカウント範囲（たとえば最小カウントおよび／また

は最大カウント）を記憶するデータ構造を含むことができる。ステップ１８０４において、正規表現生成器１１０は、ＬＣＳアルゴリズムを実行して、アルゴリズムへの入力としてスパンのセットを提供することができる。この例におけるＬＣＳアルゴリズムの出力は、ＬＣＳアルゴリズムの出力内の任意選択のスパンに対応する、ゼロに等しい最小繰り返しカウント範囲を有する少なくとも１つのスパンを含むスパンの出力セットを含むことができる。最後に、ステップ１８０５において、正規表現生成器１１０は、任意選択のスパンを含むＬＣＳアルゴリズムの出力の出力に基づいて正規表現を生成することができる。

【０２８７】

いくつかの実施形態では、判読性を改善し、かつ適切なレベルの制限を達成するために、正規表現生成器１１０は、単一スパン交代を用いてもよい。説明のため、ＬＣＳアルゴリズムがステップ１８０４で実行された後、その出力（すなわち、正規表現への最終的な変換前の中間出力）は、最小および最大繰り返しカウントを各々が有するスパンの連なりとすることができる。スパンのリストから正規表現への単純な直接変換は、（例えば、疑問符「?」修飾子コードを用いて）オプションとして印される複数の正規表現コードをもたらし得る。対照的に、場合によっては、交代を含む正規表現を生成することが望ましい場合があり、これは、括弧で閉じられた垂直方向のバーで区切られた代替物として表現され得る（https://www.regular-expressions.info/altemation.htmlを参照のこと）。例えば、時間に関するスパンの正規表現への直接的な変換は、

【０２８８】

【数２８】

\d\d:\d\d (am)?(pm)?

【０２８９】

であり得る。しかしながら、この例では、正規表現生成器１１０は、その正規表現を、より判読性のある

【０２９０】

【数２９】

\d\d:\d\d (am|pm)

【０２９１】

に変換するよう構成され得る。その変換を実行するために、正規表現生成器１１０は、スパンの下にある元のストリングフラグメントを追跡し続けてもよく、すべての元のストリングフラグメントが少なくとも２つの所与の陽性例に現れる場合、それらの元のストリングフラグメントのリストが、（例えば、複数の任意選択の正規表現コードではなく、）垂直バーで区切られた代替物として出力されるべきである、と判断してもよい。

【０２９２】

図１９は、最長共通サブシーケンス（ＬＣＳ）アルゴリズムを用いた正規表現の生成を示す例示的な図であり、生成された正規表現は、任意選択のスパンを含む。この例では、２つの入力データキャラクタシーケンスは、「8am」および「9 pm」である。入力データキャラクタシーケンスは、上述したように、まず正規表現コードに変換され（ステップ１８０２）、次にスパンに変換される（ステップ１８０３）。スパンは、ＬＣＳアルゴリズムへの入力として提供され得（ステップ１８０４）、ＬＣＳ出力は、任意選択のスパン$Z^{\langle 0,\,1\rangle}$を含み、任意選択の単一の空白が数字および２文字のテキストシーケンスであり得ることを示す。すなわち、この例における上付き表記は、先行するコード（たとえばZ＝空白）に適用される２つの数字、最小繰り返しカウント範囲（たとえば０）および最大繰り返しカウント範囲（たとえば１）を含み得る。最後に、正規表現は、ＬＣＳアルゴリズムの出力スパンに基づいて生成されてもよく、任意選択のスパンは、対応する正規表現コード「$pZ^{*}$」に変換されてもよい。

【０２９３】
いくつかの実施形態では、ＬＣＳアルゴリズムの実行中の正規表現生成器１１０による任意選択の空白の描出および使用は、性能および可読性に関してさらなる技術的利点を提供し得る。たとえば、正規表現を生成するとき、場合によっては、すべての所与の例の間で共通するキャラクタと、それらの例のうちのいくつかにおいてのみ存在するキャラクタとの両方を扱うことができることが望ましい。
【０２９４】
ある実施形態では、各スパンデータ構造について、カテゴリコードの最小発生数およびカテゴリコードの最大発生数の両方が追跡され得る。所与の例の１つまたは複数においてスパンが全く存在しない場合、最小はゼロに設定される。別の例として、綴りで示された月を扱うための正規表現を生成するために、最小数および最大数を、次いで、中括弧を伴う正規表現マルチプリシティ構文（たとえば[A-Za-z]{3,9}）にマッピングしてもよい。
【０２９５】
いくつかの実施形態では、正規表現生成器１１０は、各スパンについて最小発生数および最大発生数を追跡してもよいが、追加の実施詳細を処理してもよい。たとえば、任意選択のスパンを取り扱うこととキャラクタのスパン上でＬＣＳを実行することとの組み合わせの結果として、正規表現生成器１１０は、ＬＣＳアルゴリズムの実行を通して、連続的なスパンを検出し、マージするように構成され得る。加えて、一緒に担持されている任意の追加の任意選択のスパンが、時々、連続的に現れ、ＬＣＳアルゴリズムがそれら上でも同様に再帰的に実行されることが望ましい場合がある。たとえば、場合によっては、正規表現生成器１１０は、任意選択のシーケンス要素と必要なシーケンス要素（たとえばスパン）との間のより少ない遷移を好む（または重み付けする）よう、ＬＣＳアルゴリズムを修正および／または拡張する。たとえば、任意選択のスパンを一緒にグループ化することは、正規表現内で使用されなければならないグループ化括弧の数を最小にすることができ、したがって、生成された正規表現の人間の可読性を改善することができる。場合によっては、結果として生じる長さが、任意選択のスパンを考慮した後でさえ等しい場合、正規表現生成器１１０は、任意選択のスパンと必要なスパンとの間の遷移がより少ない代替物に対する選好を示してもよい。たとえば、場合によっては、ある標準ＬＣＳアルゴリズムは、その判断点でより長いシーケンスの選択を好むように実現され得る。しかしながら、選択肢が等しい長さのものである判断点では、構成選好が正規表現生成器１１０にプログラムされてもよい。１つのそのような構成選好は、たとえば、（任意選択のスパンが考慮されると）より短いシーケンスを好むことであり得る。したがって、この構成内のカスタマイズされたＬＣＳは、（必要なスパンの）より長いシーケンスおよび（必要なスパンおよび任意選択のスパンの合計の）より短いシーケンスを同時に最適化することができる。
【０２９６】
いくつかの実施形態では、正規表現生成器１１０によるＬＣＳアルゴリズムの実現例は、より短い出力を選択するよう構成され得る。上述のように、ＬＣＳアルゴリズムは、一般に、最長共通サブシーケンスを見つけるために用いられ得る。スパンを用いる正規表現の生成のために、ＬＣＳアルゴリズムを用いて、必要なシーケンス要素（本発明者らの場合、必要なスパン）の最長シーケンスを識別することができる。しかしながら、任意選択のスパンを含む実施形態では、ＬＣＳアルゴリズムは、共通シーケンス要素（例えばスパン）の数を最大化するという元の目標を、任意選択のスパンの数も同時に最小化しながら、保持するよう構成されてもよい。すなわち、元のＬＣＳアルゴリズムにおいて、（共通シーケンス要素の数を最大化する観点から、）第１の例のストリングからシーケンス要素（例えばスパン）を消費するか、または第２の例のストリングからシーケンス要素を消費するかに関して、任意の選択を行うことができる。そのような場合、正規表現生成器１１０は、任意選択のスパンも考慮されると、より短い全体的な正規表現長をもたらすものを選択することができるＬＣＳアルゴリズムの修正バージョンを実現してもよい。最終結果における正規表現が短いほど、判読性が向上する。
【０２９７】

さらに、いくつかの実施形態では、正規表現生成器１１０によるＬＣＳアルゴリズムの実現は、より多くの数の必要なスパンを好むよう構成され得る。すなわち、選択権を与えられると、規則表現生成部１１０は、人間の判読性を向上させるために、より少ない任意選択のスパンを伴うＬＣＳ出力を選択してもよい。

【０２９８】

いくつかの実施形態では、生成された正規表現は、任意選択のスパンで正規表現を開始するのではなく、必要なスパン（これは、人間の読者に対する精神的拠り所としても働き得る）で開始する場合、より可読性があり得る。したがって、場合によっては、結果として得られる選択肢が等しい数の遷移を有する場合、より早期の非任意選択のスパンを有する選択肢が選択され得る。加えて、正規表現生成器１１０によって実行されるＬＣＳアルゴリズムは、いくつかの実施形態では、正規表現内においてすべての空白（空白に対応する任意選択のスパンを含む）を右にプッシュするように構成されてもよい。すべての空白を右にプッシュすることによって、空白のスパンが一緒にマージされ得る機会が増大する可能性があり、これは、結果として生じる正規表現を単純化し、かつ可読性を改善し得る。このように、ＬＣＳアルゴリズムの実行中に、サブストリングの２つのセットが同じＬＣＳを有すると判断された場合、サブストリングの２つのセットのうちの１つのセットを恣意的に選択する代わりに、可読性の改善を容易にするセットを選択してもよい。さらに、いくつかの実施形態では、ＬＣＳアルゴリズムは、可読性を改善するために、より多くの数の必要なスパンおよび／またはより少ない数の任意選択のスパンを好むように構成されてもよい。

【０２９９】

上述したように、場合によっては、陰性例は、任意選択のスパンに基づいてもよい。たとえば、ユーザは、「ab」および「a2b」の陽性例ならびに「a3b」の陰性例を提供してもよい。この場合、ある例示的な実現例は失敗し得、なぜならば、それは、必要とされるスパンのみに基づいて区別しようとし得、「2」の桁は任意選択のスパンにあるからである。そのような場合、ユーザは、失敗に対して警告されることができ、生成された正規表現を手動で修復するために、および／または陰性例の一部を除去するために、ユーザインターフェイスを介して選択肢を提供されることができる。

【０３００】

いくつかの実施形態では、ＲＥＳＴサービスから戻って来るＪＳＯＮの一部として返されるisSuccessが存在し得る。いくつかの実施形態では、生成されたレゲックスは、isSuccess＝偽のとき、異なる色（たとえば赤色）となってもよい。

【０３０１】

上述したように、正規表現生成器１１０は、場合によっては、単一スパン交代を用いることができる。さらに、いくつかの実施形態では、正規表現生成器１１０は、マルチスパン交代を実行するよう構成され得る。すなわち、時として、行内のいくつかのスパンは、

【０３０２】

【数３０】

```
([A-Z]{3}\s+\d\d,\s+|\d\d/\d\d/)\d\d\d\d
```

【０３０３】

として表される日付のような交代を形成し得る。これらを検出するために、正規表現生成器１１０は、まず、ＬＣＳアルゴリズムの出力からのどのスパンが、所与の入力された陽性例のすべてにおいて実際に用いられるかを発見してもよい。これらの共通のスパンは、最初のスパンの前のファントムスパンおよび最後のスパンの後のファントムスパンとともに、アンカーポイントを構成し得る。連続するアンカーポイントのすべての対（少なくとも１つの非アンカースパンも間に有する）の間には、渡らなければならない「ブリッジ」がある。次いで、ブリッジごとに、正規表現生成器１１０は、（ａ）入力された陽性例ごとにブリッジスパンによってカバーされるストリングフラグメントを判断し、（ｂ）本明

細書で説明される技法を用いて各ストリングフラグメントを正規表現に変換し、次いで、（ｃ）上記から重複を除去し、それらを正規表現交代構文に入れてもよい。

【０３０４】

任意の順序で出現する交代内の代替物の代わりに、それらをアルファベット順にソートすることもできる。より決定論的な出力を有することによって、生成されたレゲックスは、ユーザがタイピングする各キャラクタとともにレゲックスがリアルタイムで生成される場合のタイピングの最中など、入力される例におけるわずかな変化に対しては、あまり変化しないかもしれない。さらに、いくつかの進歩したレゲックスインタプリタは、交代に遭遇すると、greedy定量化子が存在するかどうかに応じて、greedy対非greedy戦略を明示的に用いることができる。しかしながら、標準JavaおよびJavascriptＡＰＩに見られるものなど、他のレゲックスインタプリタは、交代内の代替物を、単に、それらが現れる順序で試行してもよい。この共通して見出された弱点を補償するために、レゲックス生成は、最初に、交代内における、より長い代替物を出力してもよい。また、いかなる結びつきも断つために、アルファベット順を二次ソート基準として用いてもよい。

【０３０５】

いくつかの実施形態では、正規表現生成器は、少数の陽性例において現れるスパンを非greedyワイルドカード.*?まで隠す（collapse）ように構成することができる。すなわち、入力例が少しばかり饒舌になる場合がある。正規表現生成器は、各すべての単語およびそれらの特定の単語長にマッチしようとするよりも、スパンの数および元のフラグメントコンテンツにおける有意な分散を検出する場合に、ワイルドカード.*?を用いてもよい。いくつかの実施形態では、スパンが、ワイルドカード.*?にそのように隠すことなどに加わるために考慮されるためには、スパンは、以下の３つの条件のうちの１つを満たさなければならない。（１）スパンは、（厳密に）入力された例の総数の３分の１未満に関与してもよく、（２）スパンはSPACE（空白）スパンであり、（３）スパンに対する元のテキストフラグメントのすべてが希にしか発生せず、解に「一意の句読点」が存在する。「希に」とは、（厳密に）例の数の３分の２未満を意味し得る。「一意の句読点」とは、解のどこかにSYMBOL（記号）もしくはPUNCTUATION（句読点）スパンが存在すること、または解は、テキストの始まりに^で位置指定されるか、もしくはテキストの終わりに$で位置指定されることを意味する。

【０３０６】

場合によっては、eolワイルドカードが、強調表示の内側の隠されたワイルドカードに接するのを防いでもよい。すなわち、^または$が用いられる場合、ワイルドカードは反対側の終わりに現れることを想起されたい。例えば、^([A-Z]+).*?は、最初の単語とマッチする。ワイルドカードの無差別に隠すことは、レゲックスインタプリタにマッチさせる基準点となる具体的なものを何も与えないであろう、^(.*?).*?のような何かに終わる場合がある。これを解決するために、^が用いられ、キャプチャグループ内の最後のスパンが、ワイルドカード.*?にまで隠すことの一部である場合には、ワイルドカードに接する陰性キャラクタクラスに用いられるように、行の終わりワイルドカードが発せられる前の出力に、キャプチャグループの後の１つの追加のスパンが用いられる。同様に、$が用いられ、キャプチャグループ内の最初のスパンが、ワイルドカードにまで隠すことの一部である場合には、キャプチャグループの前の１つの追加のスパンが明示的に出力に用いられる。

【０３０７】

いくつかの実施形態では、記号および句読点は、例えば、強調表示エンドポイントが２つの句読点符号の間に来ることを可能にするようスパン可能ではもはやない場合がある。そのような例では、各句読点符号および各記号は、それ自体のPUNCTUATIONまたはSYMBOLスパンを取り上げ得る。他の場合では、「A.,S」は、３つのスパン、（元フラグメントＡを伴う）LETTER、元フラグメント(.,)を伴うPUNCTUATION、および（元フラグメントSを伴う）LETTERをもたらすかもしれない。しかしながら、これらの実施形態では、４つのスパンが存在してもよい。この例では、コンマがコンマ分離リストにおけるデリミタとして機能する場合、特に、例ごとに複数のユーザ強調表示が存在する場合、役立ち

得る。この例では、ドット（期間）は、余剰であり得、コンマと同じスパンにあるべきではない。

【０３０８】

さらに、タイプALPHANUMERICのスパンの場合、正規表現生成器は、いくつの文字スパンおよび数字スパンが置き換えられたかを追跡することができる。全体的な解における内側強調表示の始まりおよび終わりを判断するなど、アルゴリズムの多くの局面は、スパンインデックスによって位置の追跡を維持するので、複数のLETTERスパンおよびNUMBERスパンを単一のALPHANUMERICスパンと置換すると、これらのインデックスおよび位置は捨て去られる。したがって、置換前インデックスから置換後インデックスへのマッピングを維持することが必要となり得る。さらに、これらは、例ごとに追跡される必要があり、なぜならば、概して、各例が（例の１００％に表されないスパンについて任意選択のスパンを含む）全体解からのすべてのインデックスを含むわけではないからである。クラスメンバnumPreSubstitutedSpansが、Spanクラスに追加され、タイプおよびデフォルト値は以下のとおりである：

```
numPreSubstitutedSpans:Map[ID,Int]=orig.flatMap(_.fromExamples).distinct.map((_,1)).toMap
```

いくつかの実施形態では、一般に利用可能なレゲックスＡＰＩを用いて、キャプチャグループの終了オフセットを見つけることができる。たとえば、いくつかのユーザインターフェイスにおいて実現され得る置換コマンドに関連して説明されるように、ネスト化自動外側モードで置換コマンドを用いる場合、正規のキャプチャグループの後に追加のキャプチャグループが（全部で３つに対して、前のものと同様に）生成され得る。ユーザインターフェイスは、この置換を実行することができ、上記と同様に、ユーザインターフェイスにおいて完全なパーサを実現する必要を避けるために、キャプチャグループの右括弧を見つけるための技法を採用することができる。これは、キャプチャグループの左括弧を見つけることよりも複雑であり得、なぜならば、キャプチャグループの右括弧(〈stufff〉)を非キャプチャグループの右括弧(?:〈stuff〉)から区別する容易な方法がないためである。左括弧を区別することは、場合によっては、ルックアヘアッドを用いて?:を探すことにより、可能であり得る。いくつかの実施形態では、検索において、正規表現とマッチすることが知られている何かの例を必要とする技法が採用されてもよい。一緒に連結されたキャプチャされたグループを元の例のテキストと比較することによって、正規表現がキャプチャグループで隙間なく完全にカバーされていることが確認されてもよい。この技法のためのコードを以下に示す：

【０３０９】

【数３１】

```
self.addEpilogueCaptureGroup = function(r, example) {
    let ret = """";
    const rightParenRegex = /(?<!\\)\)[^?+*]/g;
    let lpm = null;
    while (!ret && (lpm = rightParenRegex.exec(r)) != null)
      try {
        ret = r.substring(0, lpm.index+1) + ""("" + r.substring(lpm.index+1) + "")"";
        const m = XRegExp(ret).exec(example);
        if (m.length != 4 || m[1]+m[2]+m[3] != example)
          ret = """";
      } catch(err) {
        ret = """";
      }
    return ret;
}
```

【０３１０】

ＶＩＩＩ．コンビナトリック最長共通サブシーケンスアルゴリズムを用いた正規表現生成

本明細書で説明されるさらなる態様は、正規表現生成器１１０によって実行されるＬＣＳアルゴリズムが複数回実行されて、「正しい」正規表現（たとえば、すべての所与の陽性例と適切にマッチし、すべての所与の陰性例を適切に除外する正規表現）を生成し得る、および／または最も望ましいもしくは最適な正規表現が選択され得る複数の正しい正規表現を生成し得るコンビナトリック探索に関する。たとえば、コンビナトリック探索中、全ＬＣＳアルゴリズムおよび正規表現生成処理は、テキスト処理方向の異なる組み合わせ／置換、異なる位置指定、およびＬＣＳアルゴリズムの他の異なる特性を含めて、複数回実行されてもよい。

【０３１１】

図２０は、最長共通サブシーケンス（ＬＣＳ）アルゴリズムのコンビナトリックな実行に基づいて正規表現を生成するための処理２０００を示すフローチャートである。ステップ２００１において、正規表現生成器１１０は、陽性例に対応する入力データキャラクタシーケンスを受け取り得る。ステップ２００２において、正規表現生成器１１０は、ＬＣＳアルゴリズムのための実行技法のさまざまな異なる組み合わせに対して反復することができる。この例に示されるように、ステップ２００２の各反復の間に、正規表現生成器１１０は、以下のＬＣＳアルゴリズム実行パラメータ（または特性）、すなわち位置指定子（すなわち、位置指定なし、行の始まりで位置指定、行の終わりで位置指定）、処理方向

（すなわち、右から左の順序、左から右の順序）、空白プッシュ（すなわち、空白プッシュを行うかまたは行わない）、およびスパンを隠す（collapse）（すなわち、スパンを隠すことを行うかまたは行わない）、の異なる組み合わせを選択し得る。ステップ２００３において、ＬＣＳアルゴリズムは、入力データキャラクタシーケンスにおいて（または、入力キャラクタシーケンスが最初に変換された場合には正規表現コードにおいて）実行され、ＬＣＳアルゴリズムは、ステップ２００２において選択されたパラメータ／特性に基づいて構成される。ステップ２００４において、ＬＣＳアルゴリズムの出力は、正規表現生成器１１０によって格納されてもよく、アルゴリズムによってＬＣＳが成功裡に識別されたか否か、および対応する正規表現の長さなどのデータを含み得る。ステップ２００５において、処理は、ＬＣＳアルゴリズムがコンビナトリック探索のパラメータ／特性のすべての可能な組み合わせで実行されるまで、反復してもよい。最後に、ステップ２００６において、ＬＣＳの１つからの特定の出力が、最適出力（たとえば、成功および正規表現長に基づく）として選択され、正規表現が、選択されたＬＣＳアルゴリズム出力に基づいて生成され得る。

【０３１２】

さまざまな実施形態において、図２０を参照して上述したもののようなコンビナトリック探索は、パラメータ／特性のさまざまな異なる組み合わせに対して実行されてもよい。たとえば、幾つかの実施形態では、ＬＣＳアルゴリズムは、正規表現をテキストの始まりに位置指定するためにカレット記号^を使用し、および／または正規表現をテキストの終わりに位置指定するためにドル記号$を使用してもよい。場合によっては、そのような位置指定は、より短い正規表現を生成する結果となり得る。位置指定子は、ユーザがストリングの始まりおよび／または終わりに特定のパターンを発見することを望む場合に特に有用であり得る。たとえば、ユーザは始まりに製品名を望む場合がある。ＬＣＳアルゴリズムを、製品名を記述するさまざまな数の単語と混同するのを避けるために、下の画像に示されるように、カレットを使用して、レゲックスをストリングの始まりに位置指定することができる。

【０３１３】

さらに、いくつかの実施形態では、ＬＣＳアルゴリズムは、順方向または逆方向のいずれかである入力データを用いて実行され得る（または同様に、ＬＣＳアルゴリズムは、通常の順序で入力データを受け取り、次いでアルゴリズムを実行する前に順序を逆にするように構成されてもよい）。したがって、いくつかの実施形態では、入力キャラクタシーケンスまたはコードのペアに対して実行され得るＬＣＳアルゴリズムのコンビナトリック探索は、以下のようであってもよい。

【０３１４】

１．通常（右から左へ）の順序、開始または終了に対して位置指定しない
２．通常（右から左へ）の順序、カレット^を使用して行の始まりに対して位置指定する
３．通常（右から左へ）の順序、ドル$を使用して行の終わりに対して位置指定する
４．逆（左から右へ）の順序、始まりまたは終わりに対して位置指定しない
５．逆（左から右へ）の順序、カレット^を使用して行の始まりに対して位置指定する
６．逆（左から右へ）の順序で、ドル$を使用して行の終わりに対して位置指定する

この例では、ＬＣＳの６つの実行のうち、最も短い結果の正規表現が選択されてもよい（ステップ２００６）。

【０３１５】

幾つかの実施形態では、ＬＣＳアルゴリズムのコンビナトリック探索は、greedy量指定子「?」および非greedy量指定子「??」に対して反復してもよい。たとえば、デフォルトでは、任意選択のスパンが存在する場合、１つの疑問符が発せられ、たとえば、任意選択のミドルイニシャルを有するファーストネームおよびラストネームについては

【０３１６】

【数３２】

[A-Z]+(?: [A-Z]\.)? [A-Z]+

【０３１７】
である。greedy量指定子を使用する場合に満足のいく正規表現が見つからない場合には、コンビナトリック探索は、すべての疑問符量指定子を二重疑問符量指定子（たとえば
【０３１８】
【数３３】
[A-Z]+(?: [A-Z]\.)?? [A-Z]+)

【０３１９】
に置き換えることを試みることができる。二重疑問符は、非greedy量指定子に対応し、それは、マッチを見つけるために、下流の正規表現マッチャーにバックトラッキングモードに入るように命令することができる。
【０３２０】
加えて、いくつかの実施形態では、ＬＣＳアルゴリズムのコンビナトリック探索は、右側の空白を好むかどうかに対して反復することもできる。たとえば、上記のように、空白を右にプッシュするいくつかの実施形態において、たとえば、ＬＣＳアルゴリズムが、他の態様であれば等しい選択肢の恣意的な選択に直面する場合、空白スパンがともにマージされ、全体のスパンの数がより少なくなる結果となることを期待して、ある戦略が使用されてもよい。この特徴は、別の選択肢をコンビナトリック探索に追加し、すなわち、空白を右にプッシュするか、または判断を任意のままにする従来のＬＣＳアプローチに従って実行するかのいずれかにするようにする。
【０３２１】
さらに、いくつかの実施形態では、ＬＣＳアルゴリズムのコンビナトリック探索はまた、元のストリング上でＬＣＳを実行することによって、すべての例で共通のリテラルに対する走査／非走査に対して反復してもよい。そのような実施形態では、ＬＣＳアルゴリズムは、共通単語を識別し、整列するように構成され得る。本明細書で使用される場合、「共通単語」は、すべての陽性例において現れる単語を指し得る。いったん共通単語が識別されると、そのスパンタイプは、LETTERからWORDに変換されてもよく、次いで、ＬＣＳアルゴリズムを介する後続の実行は、それに自然に整列してもよい。
【０３２２】
したがって、以下の例においては、コンビナトリック探索は、完全なＬＣＳアルゴリズムが実行される９６回に達するように、いくつかのパラメータ／特性に対して反復してもよい。この例において反復されるべきさまざまなパラメータ／特性は以下のとおりである：
・位置指定子（３）（値＝^, $, またはどちらもない）
・空白をプッシュ（２）（値＝ＹｅｓまたはＮｏ）
・低濃度スパンのワイルドカードへの合体（２）（値＝ＹｅｓまたはＮｏ）
・Greedy量指定子?（２）（値＝ＹｅｓまたはＮｏ）
・ＬＣＳアルゴリズムの共通トークン上での整列（２）（値＝ＹｅｓまたはＮｏ）
【０３２３】
【数３４】
・別のスパンとして扱われる文字「\pL」および数字「\pN」を保持することに対して、英数字を表すよう「\w」を使用（２）（値＝ＹｅｓまたはＮｏ）

【０３２４】
上述したように、この例では、完全なＬＣＳアルゴリズムは９６回（たとえば３＊２＊２＊２＊２＊２＝９６）実行される。
【０３２５】

しかしながら、他の実施形態では、正規表現生成器１１０は、性能向上を提供してもよく、それによって、上記のリストのうちの最初の３つの特性のみ（位置指定子、空白のプッシュ、および低濃度スパンのワイルドカードへの合体）が、コンビナトリック探索に加わってもよい。これは、遙かにより少ない数の完全なＬＣＳアルゴリズムが実行されることになる結果となり得る（たとえば３＊２＊２＝１２回）。そのような実施形態では、上記リストの最後の３つの特性（Greedy量指定子、ＬＣＳアルゴリズムの共通トークン上での整列、ならびに

【０３２６】

【数３５】

別のスパンとして扱われる文字「\pL」および数字「\pN」を保持することに対して、英数字を表すよう「\w」を使用

【０３２７】

）は、コンビナトリック探索に加わらないが、これらの特性は、最後に、個々におよび逐次、テストされ得る。そのような実施形態においては技術的利点が実現され得、なぜならば、探索空白をこのように分割することは、それでも、満足のいく正規表現が、性能において約８倍の高速化を伴って、見いだされる結果となり得るからである。

【０３２８】

説明すると、コンビナトリック探索の以下の例は、前の例よりも性能上の利点を提供し得る。この例では、コンビナトリック探索は、反復されるべき以下のパラメータ／特性に基づいて実行され得る：

・位置指定（３）：BEGINNING_OF_LINE_MODE（行の始まりモード）, END_OF_LINE_MODE(行の終わりモード), NO_EOL_MODE（行の終わりなしモード）

・順序／方向（２）：右から左（通常）ＬＣＳ対左から右（逆）ＬＣＳ

・プッシュ（２）：ＬＣＳアルゴリズム内で空白を右にプッシュしようとするか否か

・ワイルドカードに圧縮（２）：時々発生するにすぎないスパンの長いシーケンスをワイルドカード.*?に圧縮しようとするか否か

この例におけるコンビナトリックは、完全なアルゴリズムを３＊２＊２＊２＝２４回実行する結果となり得る。次いで、正規表現生成器１１０は、ＬＣＳアルゴリズムの２４個の結果のうち最良のものを取り得、ここで、「最良」とは、（ａ）ＬＣＳアルゴリズムが成功したこと、および（ｂ）最短正規表現が生成されたこと、を意味し得る。次いで、正規表現生成器１１０は、以下の３つの追加のタスクを実行することができる。

１．空白、句読点、または記号によって途切れない文字および数字のシーケンスを、

【０３２９】

【数３６】

\w

【０３３０】

の生成されたレゲックスに対応する、ALPHANUMERICと呼ばれる新たなスパンタイプまで圧縮することを試みる。これは、クリックストリームログからのＩＰｖ６アドレスに見られる１６進数に対して有用であり得る。

２．greedy量指定子?の代わりに非greedy量指定子??を使用することを試みる。

３．リテラル上で整列を試みる。

【０３３１】

Ａ．スパン強調表示整列

陽性例または陰性例は、複数の強調表示を含むことができる。与えられる各例は、複数の強調表示（例えば、外側強調表示および内側強調表示）を含むことができる。したがって、例示的な実施形態は、複数の例が各々それら自体の強調表示を有する場合に、効率的かつ正確な処理のための方法を提供する。

【０３３２】

各例は、強調表示前、強調表示上、強調表示後に分解される。完全なアルゴリズムが各セット上で実行され、各セットは、強調表示前、強調表示上、および強調表示後を含む。複数の例上のユーザ強調表示は、与えられる例の間に大きなばらつき度があってもサポートされ、強調表示の失敗が回避されることになる。

【０３３３】

図５２は、いくつかの例示的実施形態による、スパン強調表示整列に基づいて正規表現を生成するための図５２００を示す。

【０３３４】

図５２は、２つのデータ例「Jane Doe」５２１０および「David Williams Jr.」５２２０に対する強調表示整列の実行を示す。図５２に示すように、各エントリの姓（例えば、DoeおよびWilliams）が強調表示されている。例えば、ユーザは、第１のレコードにおいて例として「Doe」を選択し、別のデータレコードにおいて例として「Williams」を選択してもよい。説明を容易にするために、２つの例を示す。しかしながら、複数のデータ例に対して、強調表示整列を実行することができる。

【０３３５】

図５２に示す例では、レゲックスは、部分（例えば、３部分）で生成される。データレコードのデータセル内の情報に応じて、より多い部分またはより少ない部分があり得る。レゲックスは、「強調表示前」フラグメント、「強調表示上」フラグメント、および「強調表示後」フラグメントのすべてについて生成することができる。第１のレゲックス５２３１は「強調表示前」フラグメントに対して生成され、第２のレゲックス５２３２は「強調表示上」フラグメントに対して生成され、第３のレゲックス５２３３は「強調表示後」フラグメントに対して生成される。

【０３３６】

生成されたレグックスのスパン（すなわち、最終レゲックス生成前の中間結果）は、一緒に連結される。部分（例えば、強調表示前フラグメント、強調表示上フラグメント、および強調表示後フラグメント）についての生成されたレゲックスのスパンは、一緒に連結される。

【０３３７】

図５３は、いくつかの例示的実施形態による、スパン強調表示整列を実行するための方法５３００のフローチャートを示す。図５３のステップは、図１および図２４に示す正規表現生成器のような正規表現生成器によって実行され得る。

【０３３８】

ステップ５３１０において、初期強調表示選択を受け付けることができる。例えば、ユーザは、データの列内の第１のレコードからテキストフラグメントを選択することができる。図５２に示す例では、ユーザは名前列において姓５２１０「Doe」を強調表示している。選択されたフラグメントは、「強調表示上」フラグメントとして識別することができる。選択されたフラグメントの前のフラグメントは、「強調表示前」フラグメントとして識別することができる。選択されたフラグメントの後のフラグメントは、「強調表示後」フラグメントとして識別することができる。

【０３３９】

ステップ５３２０で、第２の強調表示選択を受け付けることができる。例えば、ユーザは、最初の強調表示選択と同じデータ列（例えば、名前列）内の第２のレコードからテキストフラグメントを選択することができる。図５２に示す例では、名前列における姓５２２０「Williams」が強調表示されている。図５２に示す第２の例では、ユーザは、ある姓も強調表示している。選択されたフラグメントは「強調表示上」フラグメントとして識別することができる。選択されたフラグメントの前のフラグメントは、「強調表示前」フラグメントとして識別することができる。選択されたフラグメントの後のフラグメントは、「強調表示後」フラグメントとして識別することができる。ユーザは、複数の例を与えることを望む場合があり、したがって、ユーザは、追加の強調表示された選択を与えてもよい。しかしながら、与えられる例は、明確に整列しなくてもよい。

【０３４０】
ステップ５３３０では、フラグメントの整列が実行される。強調表示の前に生ずるフラグメントは一緒に整列され、強調表示後に生ずるフラグメントは一緒に整列され、強調表示後のフラグメントは一緒に整列される。したがって、図５２に示す例では、「Jane」を含むフラグメント５２１１および「David」を含むフラグメント５２２１は、強調表示されたフラグメントの前に生ずるため、整列される。フラグメント５２１０「Doe」およびフラグメント５２２０「Williams」は、強調表示上で生ずるので、一緒に整列される。キャラクタを含まないフラグメント５２１５、およびフラグメント５２２２は、強調表示後に生ずるので整列される。
【０３４１】
ステップ５３４０において、「強調表示前」フラグメントのためのレゲックスが生成される。例えば、第１のレゲックス５２３１が生成され得る。第１のレゲックス５２３１は、強調表示されたフラグメントの前のデータに基づいて、レゲックス生成器によって生成され得る。
【０３４２】
ステップ５３５０において、「強調表示上」フラグメントのためのレゲックスが生成される。例えば、第２のレゲックス５２３１が生成され得る。第２のレゲックス５２３２は、強調表示されたフラグメント内のデータに基づいて、レゲックス生成器によって生成され得る。
【０３４３】
ステップ５３６０において、「強調表示後」フラグメントのためのレゲックスが生成される。例えば、第３のレゲックス５２３１が生成され得る。第３のレゲックス５２３３は、強調表示されたフラグメントの後のデータに基づいて、レゲックス生成器によって生成され得る。
【０３４４】
図５３に示すように、３つの部分５２３１，５２３２，５２３３においてレゲックスが生成される。「強調表示前」フラグメントに対して第１のレゲックス５２３１が生成され、「強調表示上」フラグメントに対して第２のレゲックス５２３２が生成され、次いで、「強調表示後」フラグメントに対して第３のレゲックス４２３３が生成される。第１、第２、および第３のレゲックスを生成する順序を説明したが、レゲックスを生成する順序は様々であり得る。さらに、３つの部分について説明しているが、データ例の、３つよりも多い部分について、強調表示を行ってもよい。
【０３４５】
ステップ５３７０では、３つのレゲックスに対するスパンが連結される。すなわち、結果として生じる３つのレグックスを一緒に連結する代わりに、スパン（すなわち、最終レゲックス生成前の中間結果）が一緒に連結される。
【０３４６】
ステップ５３８０において、連結された３つのスパンについて最長共通サブシーケンスが判断される。ＬＣＳは、強調表示前スパン、強調表示上スパン、および強調表示後スパン上でＬＣＳアルゴリズムを実行した出力について、判断され得る。
【０３４７】
したがって、例示的な実施形態は、最長共通サブシーケンスが、同様に位置するデータに基づいて判断されるので、最長共通サブシーケンスを判断する、より正確な方法を提供する。
【０３４８】
ある例示的な実施形態は、英数字スパンも与える。内側強調表示の開始および終了を判断すること、スパンインデックスによって位置の追跡を維持すること、および複数の文字スパンおよび数字スパンを単一の英数字スパンに置換することにより、インデックスおよび位置を捨て去るすることができる。したがって、例示的な実施形態は、置換前インデックスから置換後インデックスへのマッピングを与える。英数字スパンのインデックスを１

つまたは複数の数字および１つまたは複数の文字のインデックスに関連付けるマッピングが生成される。

【０３４９】

さらに、各例は全体解からのインデックスのすべてを含むわけではない場合があるので、マッピングは例ごとに追跡される。多数の予め置換されたスパンを識別することができる。

【０３５０】

図５４は、いくつかの例示的実施形態による、スパンを追跡する方法５４００のフローチャートを示す。具体的には、例示的な実施形態は、ある英数字スパンのうちのいくつの文字スパンおよびいくつの数字スパンが置換されるかを判断するための方法を提供する。

【０３５１】

ステップ５４１０では、１つまたは複数の文字スパンおよび１つまたは複数の数字スパンが、単一の英数字スパンで置換される。例えば、ＨＴＭＬ１６進数カラーコードは、文字スパンおよび数字スパンのような３つの異なるスパンの代わりに、単一の英数字スパンに置換することができる。〈span style="color:#FF030A"〉BUY NOW!〈/span〉のようなＨＴＭＬ１６進数カラーコードでは、３つの異なるスパン（例えば、文字スパン、数字スパン、および別の文字スパン）の代わりに単一の英数字スパンFF030Aが用いられる。

【０３５２】

いくつかのデータフラグメント例は、１６進数「桁」Ａ～Ｆを含むことができる。この１６進数桁は、hexコード内のどこにでも出現することができる。したがって、英数字スパンを用いて、そのようなデータフラグメント例についてのマッチを容易に識別することができる。

【０３５３】

ステップ５４２０では、１つまたは複数の文字スパンおよび１つまたは複数の数字スパンを置き換えた英数字スパンを含むデータフラグメント例が識別される。

【０３５４】

ステップ５４３０では、データフラグメント例について置換前インデックスおよび置換後インデックスを識別するマッピングが生成される。各データフラグメント例に対してマッピングが生成される。

【０３５５】

ステップ５４４０では、多数の予め置換されたスパンを識別することができる。

図５５は、いくつかの例示的実施形態による、句読点スパンおよび記号スパンを表示するユーザインターフェイス５５００を示す。図５５に示すように、ユーザは、指定された例５５３３を与えている。ユーザによって指定された例は、例５５３４および５５３５を含む。レゲックス５５３２は、指定された例５５３３に基づいて生成される。

【０３５６】

図５５に示すように、記号および句読点は、強調表示エンドポイントが２つの句読点符号の間に来ることができるようスパン可能ではもはやない。

【０３５７】

各句読点符号および各記号は、それ自体の句読点スパンまたは記号スパンを有することができる。例えば、「A.,S」を含む例は、４つのスパン「A」「.」「,」および「S」を含むであろう。

【０３５８】

例示的な実施形態は、改善されたレゲックス生成美観を与える。具体的には、フラグメントは、それらを生じさせた例と関連付けることができる。フラグメントは、それらのフラグメントを生じさせた例に戻って関連付けられる。ある例示的な実施形態においては、フラグメントを生じさせた例は、例番号のリストに追加され得る。

【０３５９】

さらに、ある例示的な実施形態は、フラグメントを、それを生じさせた例と関連付ける、さらにより詳細な方法を提供する。例は、単にスパンオブジェクト全体としてではなく

、特定のテキストフラグメントに結び付けることができる。このようにして、ユーザは、例番号のリストにおいて例番号を見るだけの代わりに、例それ自体を見ることができるので、ユーザは、どの例がフラグメントに結び付けられているかをより容易に判断することができる。

【０３６０】

例えば、以下の２つの入力例、すなわち例１：「a#s#」および例２：「a#」に関して、解スパンは、[(LETTER, [("a",[1,2])]), (SYMBOL, [("#",[1,2])]), (LETTER, ["s",[1]]), (SYMBOL, ["#",[1]])]を含むことができる。個々のフラグメントごとに利用可能な例番号を有することによって、アルゴリズムは、特定の例を処理するときに強調表示の開始および／または終了を（またはコンテキストを確立するよう強調表示を超えて）検索する際に、前後（左または右）に歩くことができる。各例は、スパンに分解することができ、それらスパンの、それ自体のスパンのセットへのＬＣＳマージが実行される。

【０３６１】

別の２つの入力例、すなわち、例１：「8pm」および例２：「9am」に関して、解スパンは、[(NUMBER, [("8",[1]),("9",[2])]), (LETTER, [("pm",[1]),("am",[2])]), (PUNCTUATION, [(".",[1,2])])]を含むことができる。したがって、フラグメントは、それを生じさせた特定の例に関連付けられる。

【０３６２】

別の例では、レゲックス生成美観は、スパンは同じ組の例に由来しなければマージされないことを含み得る。マッチングを容易にするために、ある例示的な実施形態では、スパンは同じ組の例に由来しなければマージされなくてもよい。

【０３６３】

したがって、本発明の実施形態は、正確な正規表現を生成するための様々なメカニズムを提供する。正規表現をデータに適用して、ユーザが望むデータ結果を得ることができる。ユーザは、広範囲の検索やデータ操作を行うことなく、より容易かつ効率的に所望の情報を得ることができる。

【０３６４】

ＩＸ．ハードウェア概要

図２１は、ある実施形態を実現するための分散型システム２１００の簡略図を示す。図示される実施形態において、分散型システム２１００は、１つ以上の通信ネットワーク２１１０を介してサーバ２１１２に結合された１つ以上のクライアントコンピューティングデバイス２１０２、２１０４、２１０６、および２１０８を含む。クライアントコンピューティングデバイス２１０２、２１０４、２１０６、および２１０８は、１つ以上のアプリケーションを実行するように構成され得る。

【０３６５】

さまざまな実施形態において、サーバ２１１２は、本開示に記載される正規表現の自動化された生成を可能にする１つ以上のサービスまたはソフトウェアアプリケーションを実行するように適合され得る。たとえば、特定の実施形態では、サーバ２１１２は、クライアントデバイスから送信されたユーザ入力データを受信することができ、ユーザ入力データは、クライアントデバイスで表示されたユーザインターフェイスを介して、クライアントデバイスによって受信される。次いで、サーバ２１１２は、ユーザ入力データを、ユーザインターフェイスを介して表示するためにクライアントデバイスに送信される正規表現に変換することができる。

【０３６６】

特定の実施形態では、サーバ２１１２はまた、非仮想環境および仮想環境を含み得る他のサービスまたはソフトウェアアプリケーションを提供し得る。いくつかの実施形態では、これらのサービスは、クライアントコンピューティングデバイス２１０２、２１０４、２１０６および／または２１０８のユーザに対して、サービスとしてのソフトウェア（Software as a Service：ＳａａＳ）モデルのようなウェブベースのサービスまたはクラウドサービスとして提供され得る。クライアントコンピューティングデバイス２１０２、２

１０４、２１０６および／または２１０８を操作するユーザは、１つ以上のクライアントアプリケーションを利用してサーバ２１１２とやり取りすることで、これらのコンポーネントによって提供されるサービスを利用し得る。

【０３６７】

図２１に示される構成では、サーバ２１１２は、サーバ２１１２によって実行される機能を実現する１つ以上のコンポーネント２１１８、２１２０および２１２２を含み得る。これらのコンポーネントは、１つ以上のプロセッサ、ハードウェアコンポーネント、またはそれらの組合わせによって実行され得るソフトウェアコンポーネントを含み得る。分散型システム２１００とは異なり得る多種多様なシステム構成が可能であることが認識されるはずである。したがって、図２１に示される実施形態は、実施形態のシステムを実現するための分散型システムの一例であり、限定するよう意図されたものではない。

【０３６８】

ユーザは、クライアントコンピューティングデバイス２１０２、２１０４、２１０６および／または２１０８を用いて、１つまたは複数のアプリケーションを実行し、それは、本開示の教示に従って正規表現を生成してもよい。クライアントデバイスは、当該クライアントデバイスのユーザが当該クライアントデバイスと対話することを可能にするインターフェイスを提供し得る。クライアントデバイスはまた、このインターフェイスを介してユーザに情報を出力してもよい。図２１は４つのクライアントコンピューティングデバイスだけを示しているが、任意の数のクライアントコンピューティングデバイスがサポートされ得る。

【０３６９】

クライアントデバイスは、ポータブルハンドヘルドデバイス、パーソナルコンピュータおよびラップトップのような汎用コンピュータ、ワークステーションコンピュータ、ウェアラブルデバイス、ゲームシステム、シンクライアント、各種メッセージングデバイス、センサまたはその他のセンシングデバイスなどの、さまざまな種類のコンピューティングシステムを含み得る。これらのコンピューティングデバイスは、さまざまな種類およびバージョンのソフトウェアアプリケーションおよびオペレーティングシステム（たとえばＭｉｃｒｏｓｏｆｔ Ｗｉｎｄｏｗｓ（登録商標）、Ａｐｐｌｅ Ｍａｃｉｎｔｏｓｈ（登録商標）、ＵＮＩＸ（登録商標）またはＵＮＩＸ系オペレーティングシステム、Ｌｉｎｕｘ（登録商標）またはＬｉｎｕｘ系オペレーティングシステム、たとえば、各種モバイルオペレーティングシステム（たとえばＭｉｃｒｏｓｏｆｔ Ｗｉｎｄｏｗｓ Ｍｏｂｉｌｅ（登録商標）、ｉＯＳ（登録商標）、Ｗｉｎｄｏｗｓ Ｐｈｏｎｅ（登録商標）、Ａｎｄｒｏｉｄ（登録商標）、ＢｌａｃｋＢｅｒｒｙ（登録商標）、Ｐａｌｍ ＯＳ（登録商標））を含むＧｏｏｇｌｅ Ｃｈｒｏｍｅ（登録商標）ＯＳ）を含み得る。ポータブルハンドヘルドデバイスは、セルラーフォン、スマートフォン（たとえばｉＰｈｏｎｅ（登録商標））、タブレット（たとえばｉＰａｄ（登録商標））、携帯情報端末（ＰＤＡ）などを含み得る。ウェアラブルデバイスは、Ｇｏｏｇｌｅ Ｇｌａｓｓ（登録商標）ヘッドマウントディスプレイおよびその他のデバイスを含み得る。ゲームシステムは、各種ハンドヘルドゲームデバイス、インターネット接続可能なゲームデバイス（たとえばＫｉｎｅｃｔ（登録商標）ジェスチャ入力デバイス付き／無しのＭｉｃｒｏｓｏｆｔ Ｘｂｏｘ（登録商標）ゲーム機、Ｓｏｎｙ ＰｌａｙＳｔａｔｉｏｎ（登録商標）システム、Ｎｉｎｔｅｎｄｏ（登録商標）が提供する各種ゲームシステムなど）を含み得る。クライアントデバイスは、各種インターネット関連アプリケーション、通信アプリケーション（たとえばＥメールアプリケーション、ショートメッセージサービス（ＳＭＳ）アプリケーション）のような多種多様なアプリケーションを実行可能であってもよく、各種通信プロトコルを使用してもよい。

【０３７０】

ネットワーク２１１０は、利用可能な多様なプロトコルのうちのいずれかを用いてデータ通信をサポートできる、当該技術の当業者には周知のいずれかの種類のネットワークであればよく、上記プロトコルは、ＴＣＰ／ＩＰ（伝送制御プロトコル／インターネットプ

ロトコル）、ＳＮＡ（システムネットワークアーキテクチャ）、ＩＰＸ（インターネットパケット交換）、ＡｐｐｌｅＴａｌｋ（登録商標）などを含むがこれらに限定されない。単に一例として、ネットワーク２１１０は、ローカルエリアネットワーク（ＬＡＮ）、Ｅｔｈｅｒｎｅｔ（登録商標）に基づくネットワーク、トークンリング、ワイドエリアネットワーク（ＷＡＮ）、インターネット、仮想ネットワーク、仮想プライベートネットワーク（ＶＰＮ）、イントラネット、エクストラネット、公衆交換電話網（ＰＳＴＮ）、赤外線ネットワーク、無線ネットワーク（たとえば電気電子学会（ＩＥＥＥ）８０２．１１プロトコルスイートのいずれかの下で動作する無線ネットワーク、Ｂｌｕｅｔｏｏｔｈ（登録商標）および／または任意の他の無線プロトコル）、および／またはこれらおよび／または他のネットワークの任意の組み合わせを含み得る。

【０３７１】

サーバ２１１２は、１つ以上の汎用コンピュータ、専用サーバコンピュータ（一例としてＰＣ（パーソナルコンピュータ）サーバ、ＵＮＩＸ（登録商標）サーバ、ミッドレンジサーバ、メインフレームコンピュータ、ラックマウント型サーバなどを含む）、サーバファーム、サーバクラスタ、またはその他の適切な構成および／または組み合わせで構成されてもよい。サーバ２１１２は、仮想オペレーティングシステムを実行する１つ以上の仮想マシン、または仮想化を伴う他のコンピューティングアーキテクチャを含み得る。これはたとえば、サーバに対して仮想記憶装置を維持するように仮想化できる論理記憶装置の１つ以上のフレキシブルプールなどである。各種実施形態において、サーバ２１１２を、上記開示に記載の機能を提供する１つ以上のサービスまたはソフトウェアアプリケーションを実行するように適合させてもよい。

【０３７２】

サーバ２１１２内のコンピューティングシステムは、上記オペレーティングシステムのうちのいずれかを含む１つ以上のオペレーティングシステム、および、市販されているサーバオペレーティングシステムを実行し得る。また、サーバ２１１２は、ＨＴＴＰ（ハイパーテキスト転送プロトコル）サーバ、ＦＴＰ（ファイル転送プロトコル）サーバ、ＣＧＩ（コモンゲートウェイインターフェイス）サーバ、ＪＡＶＡ（登録商標）サーバ、データベースサーバなどを含むさまざまなさらに他のサーバアプリケーションおよび／または中間層アプリケーションのうちのいずれかを実行し得る。例示的なデータベースサーバは、Ｏｒａｃｌｅ（登録商標）、Ｍｉｃｒｏｓｏｆｔ（登録商標）、Ｓｙｂａｓｅ（登録商標）、ＩＢＭ（登録商標）（International Business Machines）などから市販されているものを含むが、それらに限定されない。

【０３７３】

いくつかの実現例において、サーバ２１１２は、クライアントコンピューティングデバイス２１０２、２１０４、２１０６および２１０８のユーザから受信したデータフィードおよび／またはイベントアップデートを解析および整理統合するための１つ以上のアプリケーションを含み得る。一例として、データフィードおよび／またはイベントアップデートは、センサデータアプリケーション、金融株式相場表示板、ネットワーク性能測定ツール（たとえば、ネットワークモニタリングおよびトラフィック管理アプリケーション）、クリックストリーム解析ツール、自動車交通モニタリングなどに関連するリアルタイムのイベントを含んでもよい、１つ以上の第三者情報源および連続データストリームから受信される、Ｔｗｉｔｔｅｒ（登録商標）フィード、Ｆａｃｅｂｏｏｋ（登録商標）アップデートまたはリアルタイムのアップデートを含み得るが、それらに限定されない。サーバ２１１２は、データフィードおよび／またはリアルタイムのイベントをクライアントコンピューティングデバイス２１０２、２１０４、２１０６および２１０８の１つ以上の表示デバイスを介して表示するための１つ以上のアプリケーションも含み得る。

【０３７４】

分散型システム２１００はまた、１つ以上のデータリポジトリ２１１４、２１１６を含み得る。特定の実施形態において、これらのデータリポジトリを用いてデータおよびその他の情報を格納することができる。たとえば、データリポジトリ２１１４、２１１６のう

ちの１つ以上を用いて、システムにより生成された正規表現とマッチする新たなデータの列のような情報を格納することができる。データリポジトリ２１１４、２１１６は、さまざまな場所に存在し得る。たとえば、サーバ２１１２が使用するデータリポジトリは、サーバ２１１２のローカル位置にあってもよく、またはサーバ２１１２から遠隔の位置にあってもよく、ネットワークベースの接続または専用接続を介してサーバ２１１２と通信する。データリポジトリ２１１４、２１１６は、異なる種類であってもよい。特定の実施形態において、サーバ２１１２が使用するデータリポジトリは、データベース、たとえば、Ｏｒａｃｌｅ　Ｃｏｒｐｏｒａｔｉｏｎ（登録商標）および他の製造業者が提供するデータベースのようなリレーショナルデータベースであってもよい。これらのデータベースのうちの１つ以上を、ＳＱＬフォーマットのコマンドに応じて、データの格納、アップデート、およびデータベースとの間での取り出しを可能にするように適合させてもよい。

【０３７５】

特定の実施形態では、データリポジトリ２１１４、２１１６のうちの１つ以上は、アプリケーションデータを格納するためにアプリケーションによって用いられてもよい。アプリケーションが使用するデータリポジトリは、たとえば、キー値ストアリポジトリ、オブジェクトストアリポジトリ、またはファイルシステムがサポートする汎用ストレージリポジトリのようなさまざまな種類のものであってもよい。

【０３７６】

特定の実施形態において、本開示に記載される機能は、クラウド環境を介してサービスとして提供され得る。図２２は、特定の例に係る、各種サービスをクラウドサービスとして提供し得るクラウドベースのシステム環境の簡略化されたブロック図である。図２２に示される例において、クラウドインフラストラクチャシステム２２０２は、ユーザが１つ以上のクライアントコンピューティングデバイス２２０４、２２０６および２２０８を用いて要求し得る１つ以上のクラウドサービスを提供し得る。クラウドインフラストラクチャシステム２２０２は、サーバ２１１２に関して先に述べたものを含み得る１つ以上のコンピュータおよび／またはサーバを含み得る。クラウドインフラストラクチャシステム２２０２内のコンピュータは、汎用コンピュータ、専用サーバコンピュータ、サーバファーム、サーバクラスタ、またはその他任意の適切な配置および／または組み合わせとして編成され得る。

【０３７７】

ネットワーク２２１０は、クライアント２２０４、２２０６、および２２０８と、クラウドインフラストラクチャシステム２２０２との間におけるデータの通信および交換を容易にし得る。ネットワーク２２１０は、１つ以上のネットワークを含み得る。ネットワークは同じ種類であっても異なる種類であってもよい。ネットワーク２２１０は、通信を容易にするために、有線および／または無線プロトコルを含む、１つ以上の通信プロトコルをサポートし得る。

【０３７８】

図２２に示される例は、クラウドインフラストラクチャシステムの一例にすぎず、限定を意図したものではない。なお、その他いくつかの例において、クラウドインフラストラクチャシステム２２０２が、図２２に示されるものよりも多くのコンポーネントもしくは少ないコンポーネントを有していてもよく、２つ以上のコンポーネントを組み合わせてもよく、または、異なる構成または配置のコンポーネントを有していてもよいことが、理解されるはずである。たとえば、図２２は３つのクライアントコンピューティングデバイスを示しているが、代替例においては、任意の数のクライアントコンピューティングデバイスがサポートされ得る。

【０３７９】

クラウドサービスという用語は一般に、サービスプロバイダのシステム（たとえばクラウドインフラストラクチャシステム２２０２）により、インターネット等の通信ネットワークを介してオンデマンドでユーザにとって利用可能にされるサービスを指すのに使用される。典型的に、パブリッククラウド環境では、クラウドサービスプロバイダのシステム

を構成するサーバおよびシステムは、顧客自身のオンプレミスサーバおよびシステムとは異なる。クラウドサービスプロバイダのシステムは、クラウドサービスプロバイダによって管理される。よって、顧客は、別途ライセンス、サポート、またはハードウェアおよびソフトウェアリソースをサービスのために購入しなくても、クラウドサービスプロバイダが提供するクラウドサービスを利用できる。たとえば、クラウドサービスプロバイダのシステムはアプリケーションをホストし得るとともに、ユーザは、アプリケーションを実行するためにインフラストラクチャリソースを購入しなくても、インターネットを介してオンデマンドでアプリケーションをオーダーして使用し得る。クラウドサービスは、アプリケーション、リソースおよびサービスに対する容易でスケーラブルなアクセスを提供するように設計される。いくつかのプロバイダがクラウドサービスを提供する。たとえば、ミドルウェアサービス、データベースサービス、Ｊａｖａ（登録商標）クラウドサービスなどのいくつかのクラウドサービスが、カリフォルニア州レッドウッド・ショアーズのＯｒａｃｌｅ　Ｃｏｒｐｏｒａｔｉｏｎ（登録商標）から提供される。

【０３８０】

特定の実施形態において、クラウドインフラストラクチャシステム２２０２は、ハイブリッドサービスモデルを含む、サービスとしてのソフトウェア（ＳａａＳ）モデル、サービスとしてのプラットフォーム（ＰａａＳ）モデル、サービスとしてのインフラストラクチャ（ＩａａＳ）モデルなどのさまざまなモデルを使用して、１つ以上のクラウドサービスを提供し得る。クラウドインフラストラクチャシステム２２０２は、各種クラウドサービスのプロビジョンを可能にする、アプリケーション、ミドルウェア、データベース、およびその他のリソースのスイートを含み得る。

【０３８１】

ＳａａＳモデルは、アプリケーションまたはソフトウェアを、インターネットのような通信ネットワークを通して、顧客が基本となるアプリケーションのためのハードウェアまたはソフトウェアを購入しなくても、サービスとして顧客に配信することを可能にする。たとえば、ＳａａＳモデルを用いることにより、クラウドインフラストラクチャシステム２２０２がホストするオンデマンドアプリケーションに顧客がアクセスできるようにし得る。Ｏｒａｃｌｅ　Ｃｏｒｐｏｒａｔｉｏｎ（登録商標）が提供するＳａａＳサービスの例は、人的資源／資本管理のための各種サービス、カスタマー・リレーションシップ・マネジメント（ＣＲＭ）、エンタープライズ・リソース・プランニング（ＥＲＰ）、サプライチェーン・マネジメント（ＳＣＭ）、エンタープライズ・パフォーマンス・マネジメント（ＥＰＭ）、解析サービス、ソーシャルアプリケーションなどを含むがこれらに限定されない。

【０３８２】

ＩａａＳモデルは一般に、インフラストラクチャリソース（たとえばサーバ、ストレージ、ハードウェアおよびネットワーキングリソース）を、クラウドサービスとして顧客に提供することにより、柔軟な計算およびストレージ機能を提供するために使用される。各種ＩａａＳサービスがＯｒａｃｌｅ　Ｃｏｒｐｏｒａｔｉｏｎ（登録商標）から提供される。

【０３８３】

ＰａａＳモデルは一般に、顧客が、環境リソースを調達、構築、または管理しなくても、アプリケーションおよびサービスを開発、実行、および管理することを可能にするプラットフォームおよび環境リソースをサービスとして提供するために使用される。Ｏｒａｃｌｅ　Ｃｏｒｐｏｒａｔｉｏｎ（登録商標）が提供するＰａａＳサービスの例は、Oracle Java Cloud Service（ＪＣＳ）、Oracle Database Cloud Service（ＤＢＣＳ）、データ管理クラウドサービス、各種アプリケーション開発ソリューションサービスなどを含むがこれらに限定されない。

【０３８４】

クラウドサービスは一般に、オンデマンドのセルフサービスベースで、サブスクリプションベースで、柔軟にスケーラブルで、信頼性が高く、可用性が高い、安全なやり方で提

供される。たとえば、顧客は、サブスクリプションオーダーを介し、クラウドインフラストラクチャシステム２２０２が提供する１つ以上のサービスをオーダーしてもよい。次いで、クラウドインフラストラクチャシステム２２０２は、処理を実行することにより、顧客のサブスクリプションオーダーで要求されたサービスを提供する。クラウドインフラストラクチャシステム２２０２を、１つまたは複数のクラウドサービスを提供するように構成してもよい。

【０３８５】

クラウドインフラストラクチャシステム２２０２は、さまざまなデプロイメントモデルを介してクラウドサービスを提供し得る。パブリッククラウドモデルにおいて、クラウドインフラストラクチャシステム２２０２は、第三者クラウドサービスプロバイダによって所有されていてもよく、クラウドサービスは一般のパブリックカスタマーに提供される。このカスタマーは個人または企業であってもよい。プライベートクラウドモデルでは、クラウドインフラストラクチャシステム２２０２がある組織内で（たとえば企業組織内で）機能してもよく、サービスはこの組織内の顧客に提供される。たとえば、この顧客は、人事部、給与部などの企業のさまざまな部署であってもよく、企業内の個人であってもよい。コミュニティクラウドモデルでは、クラウドインフラストラクチャシステム２２０２および提供されるサービスは、関連コミュニティ内のさまざまな組織で共有されてもよい。上記モデルの混成モデルなどのその他各種モデルが用いられてもよい。

【０３８６】

クライアントコンピューティングデバイス２２０４、２２０６、および２２０８は、異なるタイプであってもよく（たとえば図２１に示されるデバイス２１０２、２１０４、２１０６および２１０８）、１つ以上のクライアントアプリケーションを操作可能であってもよい。ユーザは、クライアントデバイスを用いることにより、クラウドインフラストラクチャシステム２２０２が提供するサービスを要求することなど、クラウドインフラストラクチャシステム２２０２とのやり取りを行い得る。

【０３８７】

いくつかの実施形態において、クラウドインフラストラクチャシステム２２０２が、管理関連サービスを提供するために実行する処理は、ビッグデータ解析を含み得る。この解析は、大きなデータセットを使用し、解析し、処理することにより、このデータ内のさまざまな傾向、挙動、関係などを検出し可視化することを含み得る。この解析は、１つ以上のプロセッサが、場合によっては、データを並列に処理し、データを用いてシミュレーションを実行するなどして、実行してもよい。たとえば、自動化された態様で正規表現を決定するために、ビッグデータ解析がクラウドインフラストラクチャシステム２２０２によって実行されてもよい。この解析に使用されるデータは、構造化データ（たとえばデータベースに格納されたデータもしくは構造化モデルに従って構造化されたデータ）および／または非構造化データ（たとえばデータブロブ（blob）（binary large object：バイナリ・ラージ・オブジェクト））を含み得る。

【０３８８】

図２２の例に示されるように、クラウドインフラストラクチャシステム２２０２は、クラウドインフラストラクチャシステム２２０２が提供する各種クラウドサービスのプロビジョンを容易にするために利用されるインフラストラクチャリソース２２３０を含み得る。インフラストラクチャリソース２２３０は、たとえば、処理リソース、ストレージまたはメモリリソース、ネットワーキングリソースなどを含み得る。

【０３８９】

特定の実施形態において、異なる顧客に対しクラウドインフラストラクチャシステム２２０２が提供する各種クラウドサービスをサポートするためのこれらのリソースを効率的にプロビジョニングし易くするために、リソースを、リソースのセットまたはリソースモジュール（「ポッド」とも処される）にまとめてもよい。各リソースモジュールまたはポッドは、１種類以上のリソースを予め一体化し最適化した組み合わせを含み得る。特定の実施形態において、異なるポッドを異なる種類のクラウドサービスに対して予めプロビジ

ョニングしてもよい。たとえば、第１のポッドセットをデータベースサービスのためにプロビジョニングしてもよく、第１のポッドセット内のポッドと異なるリソースの組み合わせを含み得る第２のポッドセットをＪａｖａサービスなどのためにプロビジョニングしてもよい。いくつかのサービスについて、これらのサービスをプロビジョニングするために割り当てられたリソースをサービス間で共有してもよい。

【０３９０】

クラウドインフラストラクチャシステム２２０２自体が、クラウドインフラストラクチャシステム２２０２の異なるコンポーネントによって共有されるとともにクラウドインフラストラクチャシステム２２０２によるサービスのプロビジョニングを容易にするサービス２２３２を、内部で使用してもよい。これらの内部共有サービスは、セキュリティ・アイデンティティサービス、統合サービス、エンタープライズリポジトリサービス、エンタープライズマネージャサービス、ウィルススキャン・ホワイトリストサービス、高可用性、バックアップリカバリサービス、クラウドサポートを可能にするサービス、Ｅメールサービス、通知サービス、ファイル転送サービスなどを含み得るが、これらに限定されない。

【０３９１】

クラウドインフラストラクチャシステム２２０２は複数のサブシステムを含み得る。これらのサブシステムは、ソフトウェア、またはハードウェア、またはそれらの組み合わせで実現され得る。図２２に示されるように、サブシステムは、クラウドインフラストラクチャシステム２２０２のユーザまたは顧客がクラウドインフラストラクチャシステム２２０２とやり取りすることを可能にするユーザインターフェイスサブシステム２２１２を含み得る。ユーザインターフェイスサブシステム２２１２は、ウェブインターフェイス２２１４、クラウドインフラストラクチャシステム２２０２が提供するクラウドサービスが宣伝広告され消費者による購入が可能なオンラインストアインターフェイス２２１６、およびその他のインターフェイス２２１８などの、各種異なるインターフェイスを含み得る。たとえば、顧客は、クライアントデバイスを用いて、クラウドインフラストラクチャシステム２２０２がインターフェイス２２１４、２２１６、および２２１８のうちの１つ以上を用いて提供する１つ以上のサービスを要求（サービス要求２２３４）してもよい。たとえば、顧客は、オンラインストアにアクセスし、クラウドインフラストラクチャシステム２２０２が提供するクラウドサービスをブラウズし、クラウドインフラストラクチャシステム２２０２が提供するとともに顧客が申し込むことを所望する１つ以上のサービスについてサブスクリプションオーダーを行い得る。このサービス要求は、顧客と、顧客が申しむことを所望する１つ以上のサービスを識別する情報を含んでいてもよい。たとえば、顧客は、クラウドインフラストラクチャシステム２２０２によって提供される正規表現の自動生成関連サービスの申し込み注文を出すことができる。

【０３９２】

図２２に示される例のような特定の実施形態において、クラウドインフラストラクチャシステム２２０２は、新しいオーダーを処理するように構成されたオーダー管理サブシステム（order management subsystem：ＯＭＳ）２２２０を含み得る。この処理の一部として、ＯＭＳ２２２０は、既に作成されていなければ顧客のアカウントを作成し、要求されたサービスを顧客に提供するために顧客に対して課金するのに使用する課金および／またはアカウント情報を顧客から受け、顧客情報を検証し、検証後、顧客のためにこのオーダーを予約し、各種ワークフローを調整することにより、プロビジョニングのためにオーダーを準備するように、構成されてもよい。

【０３９３】

適切に妥当性確認がなされると、ＯＭＳ２２２０は、処理、メモリ、およびネットワーキングリソースを含む、このオーダーのためのリソースをプロビジョニングするように構成されたオーダープロビジョニングサブシステム（ＯＰＳ）２２２４を呼び出し得る。プロビジョニングは、オーダーのためのリソースを割り当てることと、顧客オーダーが要求するサービスを容易にするようにリソースを構成することとを含み得る。オーダーのためにリソースをプロビジョニングするやり方およびプロビジョニングされるリソースのタイ

プは、顧客がオーダーしたクラウドサービスのタイプに依存し得る。たとえば、あるワークフローに従うと、ＯＰＳ２２２４を、要求されている特定のクラウドサービスを判断し、この特定のクラウドサービスのために予め構成されたであろうポッドの数を特定するように構成されてもよい。あるオーダーのために割り当てられるポッドの数は、要求されたサービスのサイズ／量／レベル／範囲に依存し得る。たとえば、割り当てるポッドの数は、サービスがサポートすべきユーザの数、サービスが要求されている期間などに基づいて決定してもよい。次に、割り当てられたポッドを、要求されたサービスを提供するために、要求している特定の顧客に合わせてカスタマイズしてもよい。

【０３９４】

クラウドインフラストラクチャシステム２２０２は、要求されたサービスがいつ使用できるようになるかを示すために、レスポンスまたは通知２２４４を、要求している顧客に送ってもよい。いくつかの例において、顧客が、要求したサービスの利益の使用および利用を開始できるようにする情報（たとえばリンク）を顧客に送信してもよい。特定の実施形態では、正規表現の自動生成関連サービスを要求する顧客に対して、応答は、実行されるとユーザインターフェイスの表示を引き起こす命令を含み得る。

【０３９５】

クラウドインフラストラクチャシステム２２０２はサービスを複数の顧客に提供し得る。各顧客ごとに、クラウドインフラストラクチャシステム２２０２は、顧客から受けた１つ以上のサブスクリプションオーダーに関連する情報を管理し、オーダーに関連する顧客データを維持し、要求されたサービスを顧客に提供する役割を果たす。また、クラウドインフラストラクチャシステム２２０２は、申し込まれたサービスの顧客による使用に関する使用統計を収集してもよい。たとえば、統計は、使用されたストレージの量、転送されたデータの量、ユーザの数、ならびにシステムアップタイムおよびシステムダウンタイムの量などについて、収集されてもよい。この使用情報を用いて顧客に課金してもよい。課金はたとえば月ごとに行ってもよい。

【０３９６】

クラウドインフラストラクチャシステム２２０２は、サービスを複数の顧客に並列に提供してもよい。クラウドインフラストラクチャシステム２２０２は、場合によっては著作権情報を含む、これらの顧客についての情報を格納してもよい。特定の実施形態において、クラウドインフラストラクチャシステム２２０２は、顧客の情報を管理するとともに管理される情報を分離することで、ある顧客に関する情報が別の顧客に関する情報からアクセスされないようにするように構成された、アイデンティティ管理サブシステム（ＩＭＳ）２２２８を含む。ＩＭＳ２２２８は、アイデンティティサービス、情報アクセス管理、認証および許可サービス、顧客のアイデンティティおよび役割ならびに関連する能力などを管理するためのサービスなどの、各種セキュリティ関連サービスを提供するように構成されてもよい。

【０３９７】

図２３は、コンピュータシステム２３００の例を示す。いくつかの実施形態では、コンピュータシステム２３００は、上述のシステムのいずれかを実現するために用いられ得る。図２３に示されるように、コンピュータシステム２３００は、バスサブシステム２３０２を介して他のいくつかのサブシステムと通信する処理サブシステム２３０４を含むさまざまなサブシステムを含む。これらの他のサブシステムは、処理加速ユニット２３０６、Ｉ／Ｏサブシステム２３０８、ストレージサブシステム２３１８、および通信サブシステム２３２４を含み得る。ストレージサブシステム２３１８は、記憶媒体２３２２およびシステムメモリ２３１０を含む非一時的なコンピュータ読取り可能記憶媒体を含み得る。

【０３９８】

バスサブシステム２３０２は、コンピュータシステム２３００のさまざまなコンポーネントおよびサブシステムに意図されるように互いに通信させるための機構を提供する。バスサブシステム２３０２は単一のバスとして概略的に示されているが、バスサブシステムの代替例は複数のバスを利用してもよい。バスサブシステム２３０２は、さまざまなバス

アーキテクチャのうちのいずれかを用いる、メモリバスまたはメモリコントローラ、周辺バス、ローカルバスなどを含むいくつかのタイプのバス構造のうちのいずれかであってもよい。たとえば、このようなアーキテクチャは、業界標準アーキテクチャ（Industry Standard Architecture：ＩＳＡ）バス、マイクロチャネルアーキテクチャ（Micro Channel Architecture：ＭＣＡ）バス、エンハンストＩＳＡ（Enhanced ISA：ＥＩＳＡ）バス、ビデオ・エレクトロニクス・スタンダーズ・アソシエーション（Video Electronics Standards Association：ＶＥＳＡ）ローカルバス、およびＩＥＥＥ　Ｐ１３８６．１規格に従って製造されるメザニンバスとして実現され得る周辺コンポーネントインターコネクト（Peripheral Component Interconnect：ＰＣＩ）バスなどを含み得る。

【０３９９】

処理サブシステム２３０４は、コンピュータシステム２３００の動作を制御し、１つ以上のプロセッサ、特定用途向け集積回路（ＡＳＩＣ）、またはフィールドプログラマブルゲートアレイ（ＦＰＧＡ）を含み得る。プロセッサは、シングルコアまたはマルチコアプロセッサを含み得る。コンピュータシステム２３００の処理リソースを、１つ以上の処理ユニット２３３２、２３３４などに組織することができる。処理ユニットは、１つ以上のプロセッサ、同一のまたは異なるプロセッサからの１つ以上のコア、コアとプロセッサとの組み合わせ、またはコアとプロセッサとのその他の組み合わせを含み得る。いくつかの実施形態において、処理サブシステム２３０４は、グラフィックスプロセッサ、デジタル信号プロセッサ（ＤＳＰ）などのような１つ以上の専用コプロセッサを含み得る。いくつかの実施形態では、処理サブシステム２３０４の処理ユニットの一部または全部は、特定用途向け集積回路（ＡＳＩＣ）またはフィールドプログラマブルゲートアレイ（ＦＰＧＡ）などのカスタマイズされた回路を使用し得る。

【０４００】

いくつかの実施形態において、処理サブシステム２３０４内の処理ユニットは、システムメモリ２３１０またはコンピュータ読取り可能記憶媒体２３２２に格納された命令を実行し得る。さまざまな例において、処理ユニットはさまざまなプログラムまたはコード命令を実行するとともに、同時に実行する複数のプログラムまたはプロセスを維持し得る。任意の所定の時点で、実行されるべきプログラムコードの一部または全部は、システムメモリ２３１０および／または潜在的に１つ以上の記憶装置を含むコンピュータ読取り可能記憶媒体２３２２に常駐していてもよい。適切なプログラミングを介して、処理サブシステム２３０４は、上述のさまざまな機能を提供し得る。コンピュータシステム２３００が１つ以上の仮想マシンを実行している例において、１つ以上の処理ユニットが各仮想マシンに割り当ててもよい。

【０４０１】

特定の実施形態において、コンピュータシステム２３００によって実行される全体的な処理を加速するように、カスタマイズされた処理を実行するために、または処理サブシステム２３０４によって実行される処理の一部をオフロードするために、処理加速ユニット２３０６を任意に設けることができる。

【０４０２】

Ｉ／Ｏサブシステム２３０８は、コンピュータシステム２３００に情報を入力するための、および／またはコンピュータシステム２３００から、もしくはコンピュータシステム２３００を介して、情報を出力するための、デバイスおよび機構を含むことができる。一般に、「入力デバイス」という語の使用は、コンピュータシステム２３００に情報を入力するためのすべての考えられ得るタイプのデバイスおよび機構を含むよう意図される。ユーザインターフェイス入力デバイスは、たとえば、キーボード、マウスまたはトラックボールなどのポインティングデバイス、ディスプレイに組み込まれたタッチパッドまたはタッチスクリーン、スクロールホイール、クリックホイール、ダイアル、ボタン、スイッチ、キーパッド、音声コマンド認識システムを伴う音声入力デバイス、マイクロフォン、および他のタイプの入力デバイスを含んでもよい。ユーザインターフェイス入力デバイスは、ユーザが入力デバイスを制御しそれと対話することを可能にするＭｉｃｒｏｓｏｆｔ

Ｋｉｎｅｃｔ（登録商標）モーションセンサ、Ｍｉｃｒｏｓｏｆｔ Ｘｂｏｘ（登録商標）３６０ゲームコントローラ、ジェスチャおよび音声コマンドを用いる入力を受信するためのインターフェイスを提供するデバイスなど、モーションセンシングおよび／またはジェスチャ認識デバイスも含んでもよい。ユーザインターフェイス入力デバイスは、ユーザから目の動き（たとえば、写真を撮っている間および／またはメニュー選択を行っている間の「まばたき」）を検出し、アイジェスチャを入力デバイス（たとえばＧｏｏｇｌｅ Ｇｌａｓｓ（登録商標））への入力として変換するＧｏｏｇｌｅ Ｇｌａｓｓ（登録商標）瞬き検出器などのアイジェスチャ認識デバイスも含んでもよい。また、ユーザインターフェイス入力デバイスは、ユーザが音声コマンドを介して音声認識システム（たとえばＳｉｒｉ（登録商標）ナビゲータ）と対話することを可能にする音声認識感知デバイスを含んでもよい。

【０４０３】

ユーザインターフェイス入力デバイスの他の例は、三次元（３Ｄ）マウス、ジョイスティックまたはポインティングスティック、ゲームパッドおよびグラフィックタブレット、ならびにスピーカ、デジタルカメラ、デジタルカムコーダ、ポータブルメディアプレーヤ、ウェブカム、画像スキャナ、指紋スキャナ、バーコードリーダ３Ｄスキャナ、３Ｄプリンタ、レーザレンジファインダ、および視線追跡デバイスなどの聴覚／視覚デバイスも含んでもよいが、それらに限定されない。また、ユーザインターフェイス入力デバイスは、たとえば、コンピュータ断層撮影、磁気共鳴撮像、ポジションエミッショントモグラフィー、および医療用超音波検査デバイスなどの医療用画像化入力デバイスを含んでもよい。ユーザインターフェイス入力デバイスは、たとえば、ＭＩＤＩキーボード、デジタル楽器などの音声入力デバイスも含んでもよい。

【０４０４】

一般に、出力デバイスという語の使用は、コンピュータシステム２３００からユーザまたは他のコンピュータに情報を出力するための考えられるすべてのタイプのデバイスおよび機構を含むことを意図している。ユーザインターフェイス出力デバイスは、ディスプレイサブシステム、インジケータライト、または音声出力デバイスなどのような非ビジュアルディスプレイなどを含んでもよい。ディスプレイサブシステムは、陰極線管（ＣＲＴ）、液晶ディスプレイ（ＬＣＤ）またはプラズマディスプレイを使うものなどのフラットパネルデバイス、計画デバイス、タッチスクリーンなどであってもよい。たとえば、ユーザインターフェイス出力デバイスは、モニタ、プリンタ、スピーカ、ヘッドフォン、自動車ナビゲーションシステム、プロッタ、音声出力デバイスおよびモデムなどの、テキスト、グラフィックスおよび音声／映像情報を視覚的に伝えるさまざまな表示デバイスを含んでもよいが、それらに限定されない。

【０４０５】

ストレージサブシステム２３１８は、コンピュータシステム２３００によって使用される情報およびデータを格納するためのリポジトリまたはデータストアを提供する。ストレージサブシステム２３１８は、いくつかの例の機能を提供する基本的なプログラミングおよびデータ構成を格納するための有形の非一時的なコンピュータ読取り可能記憶媒体を提供する。処理サブシステム２３０４によって実行されると上述の機能を提供するソフトウェア（たとえばプログラム、コードモジュール、命令）が、ストレージサブシステム２３１８に格納されてもよい。ソフトウェアは、処理サブシステム２３０４の１つ以上の処理ユニットによって実行されてもよい。ストレージサブシステム２３１８はまた、本開示の教示に従って使用されるデータを格納するためのリポジトリを提供してもよい。

【０４０６】

ストレージサブシステム２３１８は、揮発性および不揮発性メモリデバイスを含む１つ以上の非一時的メモリデバイスを含み得る。図２３に示すように、ストレージサブシステム２３１８は、システムメモリ２３１０およびコンピュータ読取り可能記憶媒体２３２２を含む。システムメモリ２３１０は、プログラム実行中に命令およびデータを格納するための揮発性主ランダムアクセスメモリ（ＲＡＭ）と、固定命令が格納される不揮発性読取

り専用メモリ（ＲＯＭ）またはフラッシュメモリとを含む、いくつかのメモリを含み得る。いくつかの実現例において、起動中などにコンピュータシステム２３００内の要素間における情報の転送を助ける基本的なルーチンを含むベーシックインプット／アウトプットシステム（basic input/output system：ＢＩＯＳ）は、典型的には、ＲＯＭに格納されてもよい。典型的に、ＲＡＭは、処理サブシステム２３０４によって現在操作および実行されているデータおよび／またはプログラムモジュールを含む。いくつかの実現例において、システムメモリ２３１０は、スタティックランダムアクセスメモリ（ＳＲＡＭ）、ダイナミックランダムアクセスメモリ（ＤＲＡＭ）などのような複数の異なるタイプのメモリを含み得る。

【０４０７】

一例として、限定を伴うことなく、図２３に示されるように、システムメモリ２３１０は、ウェブブラウザ、中間層アプリケーション、リレーショナルデータベース管理システム（ＲＤＢＭＳ）などのような各種アプリケーションを含み得る、実行中のアプリケーションプログラム２３１２、プログラムデータ２３１４、およびオペレーティングシステム２３１６を、ロードしてもよい。一例として、オペレーティングシステム２３１６は、Ｍｉｃｒｏｓｏｆｔ Ｗｉｎｄｏｗｓ（登録商標）、Ａｐｐｌｅ Ｍａｃｉｎｔｏｓｈ（登録商標）および／またはＬｉｎｕｘオペレーティングシステム、市販されているさまざまなＵＮＩＸ（登録商標）またはＵＮＩＸ系オペレーティングシステム（さまざまなＧＮＵ／Ｌｉｎｕｘオペレーティングシステム、Ｇｏｏｇｌｅ Ｃｈｒｏｍｅ（登録商標）ＯＳなどを含むがそれらに限定されない）、および／または、ｉＯＳ（登録商標）、Ｗｉｎｄｏｗｓ（登録商標） Ｐｈｏｎｅ、Ａｎｄｒｏｉｄ（登録商標） ＯＳ、ＢｌａｃｋＢｅｒｒｙ（登録商標） ＯＳ、Ｐａｌｍ（登録商標） ＯＳオペレーティングシステムのようなさまざまなバージョンのモバイルオペレーティングシステムなどを、含み得る。

【０４０８】

コンピュータ読取り可能記憶媒体２３２２は、いくつかの例の機能を提供するプログラミングおよびデータ構成を格納することができる。コンピュータ読取り可能記憶媒体２３２２は、コンピュータシステム２３００のための、コンピュータ読取り可能命令、データ構造、プログラムモジュール、および他のデータのストレージを提供することができる。処理サブシステム２３０４によって実行されると上記機能を提供するソフトウェア（プログラム、コードモジュール、命令）は、ストレージサブシステム２３１８に格納されてもよい。一例として、コンピュータ読取り可能記憶媒体２３２２は、ハードディスクドライブ、磁気ディスクドライブ、ＣＤ ＲＯＭ、ＤＶＤ、Ｂｌｕ－Ｒａｙ（登録商標）ディスクなどの光ディスクドライブ、またはその他の光学媒体のような不揮発性メモリを含み得る。コンピュータ読取り可能記憶媒体２３２２は、Ｚｉｐ（登録商標）ドライブ、フラッシュメモリカード、ユニバーサルシリアルバス（ＵＳＢ）フラッシュドライブ、セキュアデジタル（ＳＤ）カード、ＤＶＤディスク、デジタルビデオテープなどを含んでもよいが、それらに限定されない。コンピュータ読取り可能記憶媒体２３２２は、フラッシュメモリベースのＳＳＤ、エンタープライズフラッシュドライブ、ソリッドステートＲＯＭなどのような不揮発性メモリに基づくソリッドステートドライブ（ＳＳＤ）、ソリッドステートＲＡＭ、ダイナミックＲＡＭ、スタティックＲＡＭのような揮発性メモリに基づくＳＳＤ、ＤＲＡＭベースのＳＳＤ、磁気抵抗ＲＡＭ（ＭＲＡＭ）ＳＳＤ、およびＤＲＡＭとフラッシュメモリベースのＳＳＤとの組み合わせを使用するハイブリッドＳＳＤも含み得る。

【０４０９】

特定の実施形態において、ストレージサブシステム２３１８は、コンピュータ読取り可能記憶媒体２３２２にさらに接続可能なコンピュータ読取り可能記憶媒体リーダ２３２０も含み得る。リーダ２３２０は、ディスク、フラッシュドライブなどのようなメモリデバイスからデータを受け、読取るように構成されてもよい。

【０４１０】

特定の実施形態において、コンピュータシステム２３００は、処理およびメモリリソースの仮想化を含むがこれに限定されない仮想化技術をサポートし得る。たとえば、コンピ

ュータシステム２３００は、１つ以上の仮想マシンを実行するためのサポートを提供し得る。特定の実施形態において、コンピュータシステム２３００は、仮想マシンの構成および管理を容易にするハイパーバイザなどのプログラムを実行し得る。各仮想マシンには、メモリ、演算（たとえばプロセッサ、コア）、Ｉ／Ｏ、およびネットワーキングリソースを割り当てられてもよい。各仮想マシンは通常、他の仮想マシンから独立して実行される。仮想マシンは、典型的には、コンピュータシステム２３００によって実行される他の仮想マシンによって実行されるオペレーティングシステムと同じであり得るかまたは異なり得るそれ自体のオペレーティングシステムを実行する。したがって、潜在的に複数のオペレーティングシステムがコンピュータシステム２３００によって同時に実行され得る。

【０４１１】

通信サブシステム２３２４は、他のコンピュータシステムおよびネットワークに対するインターフェイスを提供する。通信サブシステム２３２４は、他のシステムとコンピュータシステム２３００との間のデータの送受のためのインターフェイスとして機能する。たとえば、通信サブシステム２３２４は、コンピュータシステム２３００が、１つ以上のクライアントデバイスとの間で情報を送受信するために、インターネットを介して１つ以上のクライアントデバイスへの通信チャネルを確立することを可能にし得る。

【０４１２】

通信サブシステム２３２４は、有線および／または無線通信プロトコルの両方をサポートし得る。ある実施形態において、通信サブシステム２３２４は、（たとえば、セルラー電話技術、３Ｇ、４ＧもしくはＥＤＧＥ（グローバル進化のための高速データレート）などの先進データネットワーク技術、ＷｉＦｉ（ＩＥＥＥ８０２．ＸＸファミリー規格、もしくは他のモバイル通信技術、またはそれらのいずれかの組み合わせを用いて）無線音声および／またはデータネットワークにアクセスするための無線周波数（ＲＦ）送受信機コンポーネント、グローバルポジショニングシステム（ＧＰＳ）受信機コンポーネント、および／または他のコンポーネントを含み得る。いくつかの実施形態において、通信サブシステム２３２４は、無線インターフェイスに加えてまたはその代わりに、有線ネットワーク接続（たとえばEthernet（登録商標））を提供し得る。

【０４１３】

通信サブシステム２３２４は、さまざまな形式でデータを受信および送信し得る。いくつかの実施形態において、通信サブシステム２３２４は、他の形式に加えて、構造化データフィードおよび／または非構造化データフィード２３２６、イベントストリーム２３２８、イベントアップデート２３３０などの形式で入力通信を受信してもよい。たとえば、通信サブシステム２３２４は、ソーシャルメディアネットワークおよび／またはＴｗｉｔｔｅｒ（登録商標）フィード、Ｆａｃｅｂｏｏｋ（登録商標）アップデート、ＲｉｃｈＳｉｔｅ Ｓｕｍｍａｒｙ（ＲＳＳ）フィードなどのウェブフィード、および／または１つ以上の第三者情報源からのリアルタイムアップデートなどのような他の通信サービスのユーザから、リアルタイムでデータフィード２３２６を受信（または送信）するように構成されてもよい。

【０４１４】

特定の実施形態において、通信サブシステム２３２４は、連続データストリームの形式でデータを受信するように構成されてもよく、当該連続データストリームは、明確な終端を持たない、本来は連続的または無限であり得るリアルタイムイベントのイベントストリーム２３２８および／またはイベントアップデート２３３０を含んでもよい。連続データを生成するアプリケーションの例としては、たとえば、センサデータアプリケーション、金融株式相場表示板、ネットワーク性能測定ツール（たとえばネットワークモニタリングおよびトラフィック管理アプリケーション）、クリックストリーム解析ツール、自動車交通モニタリングなどを挙げることができる。

【０４１５】

通信サブシステム２３２４は、コンピュータシステム２３００からのデータを他のコンピュータシステムまたはネットワークに伝えるように構成されてもよい。このデータは、

構造化および／または非構造化データフィード２３２６、イベントストリーム２３２８、イベントアップデート２３３０などのような各種異なる形式で、コンピュータシステム２３００に結合された１つ以上のストリーミングデータソースコンピュータと通信し得る１つ以上のデータベースに、伝えられてもよい。

【０４１６】

コンピュータシステム２３００は、ハンドヘルドポータブルデバイス（たとえばｉＰｈｏｎｅ（登録商標）セルラーフォン、ｉＰａｄ（登録商標）コンピューティングタブレット、ＰＤＡ）、ウェアラブルデバイス（たとえばＧｏｏｇｌｅ　Ｇｌａｓｓ（登録商標）ヘッドマウントディスプレイ）、パーソナルコンピュータ、ワークステーション、メインフレーム、キオスク、サーバラック、またはその他のデータ処理システムを含む、さまざまなタイプのうちの１つであればよい。コンピュータおよびネットワークの性質が常に変化しているため、図２３に示されるコンピュータシステム２３００の記載は、具体的な例として意図されているに過ぎない。図２３に示されるシステムよりも多くのコンポーネントまたは少ないコンポーネントを有するその他多くの構成が可能である。当業者であれば、本明細書における開示および教示に基づいて、さまざまな例を実現するための他の態様および／または方法を認識するだろう。

【０４１７】

特定の例について説明したが、さまざまな変形、変更、代替構成、および均等物が可能である。例は、特定のデータ処理環境内の動作に限定されず、複数のデータ処理環境内で自由に動作させることができる。さらに、例を特定の一連のトランザクションおよびステップを使用して説明したが、これが限定を意図しているのではないことは当業者には明らかであるはずである。いくつかのフローチャートは動作を逐次的プロセスとして説明しているが、これらの動作のうちの多くは並列または同時に実行されてもよい。加えて、動作の順序を再指定してもよい。プロセスは図に含まれない追加のステップを有し得る。上記の例の各種特徴および局面は、個別に使用されてもよく、またはともに使用されてもよい。

【０４１８】

さらに、特定の例をハードウェアとソフトウェアとの特定の組み合わせを用いて説明してきたが、ハードウェアとソフトウェアとの他の組み合わせも可能であることが理解されるはずである。特定の例は、ハードウェアでのみ、またはソフトウェアでのみ、またはそれらの組み合わせを用いて実現されてもよい。本明細書に記載されたさまざまなプロセスは、同じプロセッサまたは任意の組み合わせの異なるプロセッサ上で実現されてもよい。

【０４１９】

デバイス、システム、コンポーネントまたはモジュールが特定の動作または機能を実行するように構成されると記載されている場合、そのような構成は、たとえば、動作を実行するように電子回路を設計することにより、動作を実行するようにプログラミング可能な電子回路（マイクロプロセッサなど）をプログラミングすることにより、たとえば、非一時的なメモリ媒体に格納されたコードもしくは命令またはそれらの任意の組み合わせを実行するようにプログラミングされたコンピュータ命令もしくはコード、またはプロセッサもしくはコアを実行するなどにより、達成され得る。プロセスは、プロセス間通信のための従来の技術を含むがこれに限定されないさまざまな技術を使用して通信することができ、異なる対のプロセスは異なる技術を使用してもよく、同じ対のプロセスは異なる時間に異なる技術を使用してもよい。

【０４２０】

本開示では具体的な詳細を示すことにより例が十分に理解されるようにしている。しかしながら、例はこれらの具体的な詳細がなくとも実施し得るものである。たとえば、周知の回路、プロセス、アルゴリズム、構造、および技術は、例が曖昧にならないようにするために不必要な詳細事項なしで示している。本明細書は例示的な例のみを提供し、他の例の範囲、適用可能性、または構成を限定するよう意図されたものではない。むしろ、例の上記説明は、各種例を実現することを可能にする説明を当業者に提供する。要素の機能および構成の範囲内でさまざまな変更が可能である。

【０４２１】
したがって、明細書および図面は、限定的な意味ではなく例示的なものとみなされるべきである。しかしながら、請求項に記載されているより広範な精神および範囲から逸脱することなく、追加、削減、削除、ならびに他の修正および変更がこれらになされ得ることは明らかであろう。このように、具体的な例を説明してきたが、これらは限定を意図するものではない。さまざまな変形例および同等例は添付の特許請求の範囲内にある。
【０４２２】
上記の明細書では、本開示の局面についてその具体的な例を参照して説明しているが、本開示はそれに限定されるものではないということを当業者は認識するであろう。上記の開示のさまざまな特徴および局面は、個々にまたは一緒に用いられてもよい。さらに、例は、明細書のさらに広い精神および範囲から逸脱することなく、本明細書に記載されているものを超えて、さまざまな環境および用途で利用することができる。したがって、明細書および図面は、限定的ではなく例示的であると見なされるべきである。
【０４２３】
上記の説明では、例示の目的で、方法を特定の順序で記載した。代替の例では、方法は記載された順序とは異なる順序で実行されてもよいことを理解されたい。また、上記の方法は、ハードウェアコンポーネントによって実行されてもよいし、マシン実行可能命令であって、用いられると、そのような命令でプログラムされた汎用もしくは専用のプロセッサまたは論理回路などのマシンに方法を実行させてもよいマシン実行可能命令のシーケンスで具体化されてもよいことも理解されたい。これらのマシン実行可能命令は、ＣＤ－ＲＯＭもしくは他の種類の光ディスク、フロッピー（登録商標）ディスク、ＲＯＭ、ＲＡＭ、ＥＰＲＯＭ、ＥＥＰＲＯＭ、磁気もしくは光学カード、フラッシュメモリのような、１つ以上の機械可読媒体、または電子命令を記憶するのに適した他の種類の機械可読媒体に保存できる。代替的に、これらの方法は、ハードウェアとソフトウェアとの組み合わせによって実行されてもよい。
【０４２４】
構成要素が特定の動作を実行するように構成されるとして記載されている場合、そのような構成は、たとえば、特定の動作を実行するよう電子回路もしくは他のハードウェアを設計すること、特定の動作を実行するようプログラミング可能な電子回路（たとえばマイクロプロセッサもしくは他の好適な電子回路）をプログラミングすること、またはそれらの任意の組み合わせによって達成されてもよい。
【０４２５】
本願の説明のための例をここに詳細に記載したが、本発明の概念は、他の態様で様々に具現化および採用され得ること、および特許請求の範囲は、先行技術によって制限される場合を除き、そのような変形を含むように解釈されるよう意図されることを理解されたい。
【０４２６】
構成要素が特定の動作を実行する「ように構成される」として記載されている場合、そのような構成は、たとえば、特定の動作を実行するよう電子回路もしくは他のハードウェアを設計すること、特定の動作を実行するようプログラミング可能な電子回路（たとえばマイクロプロセッサもしくは他の好適な電子回路）をプログラミングすること、またはそれらの任意の組み合わせによって達成されてもよい。

【図面】

【図１】

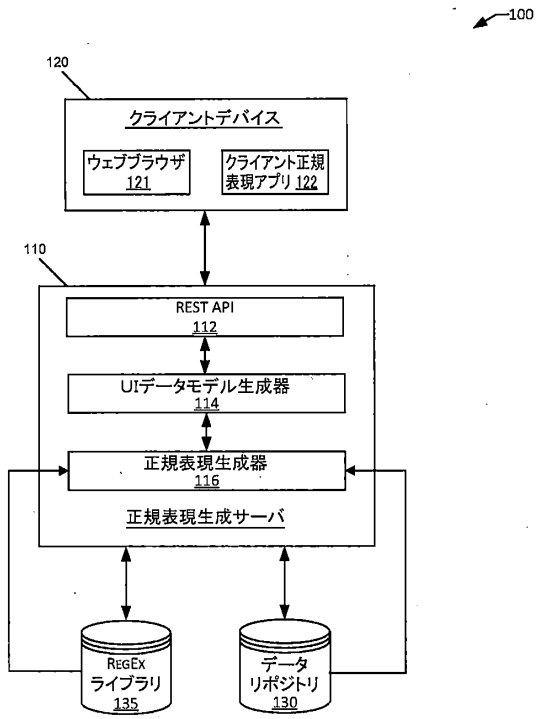



FIG. 1

【図２】

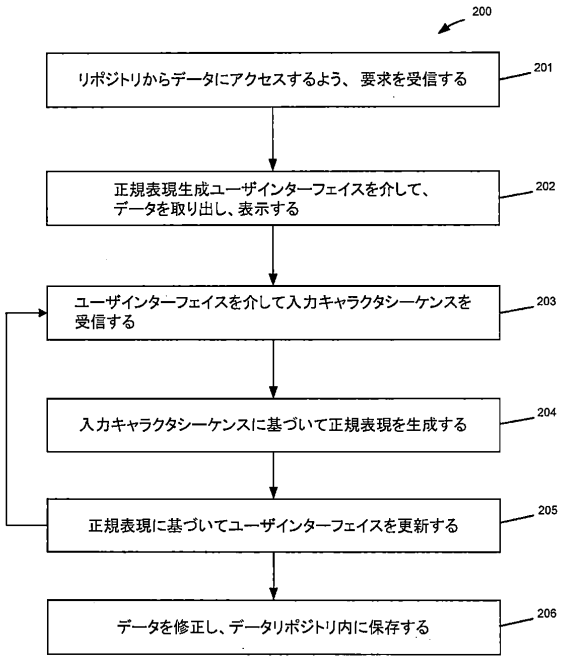



FIG. 2

【図３】

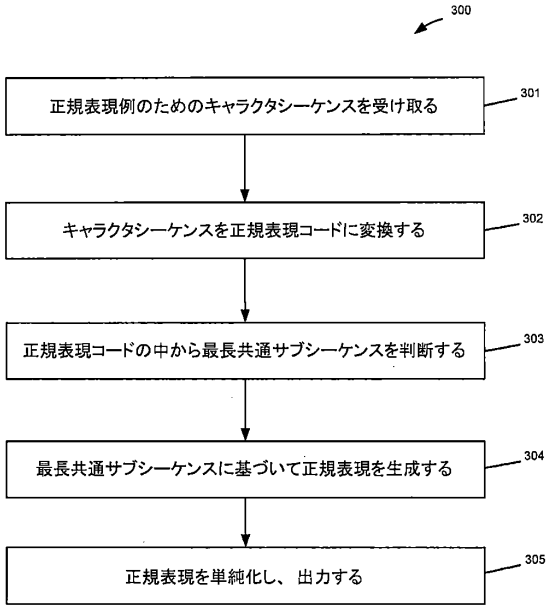



FIG. 3

【図４】

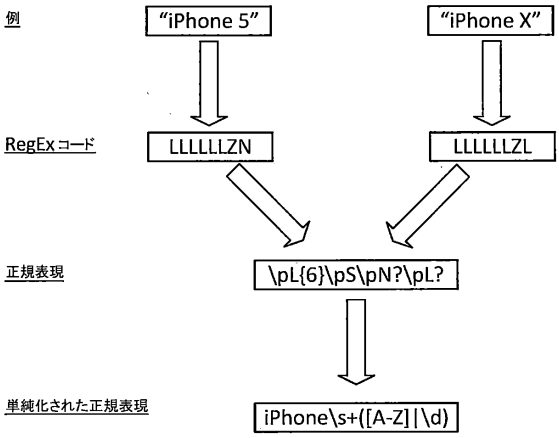



FIG. 4

【図５】

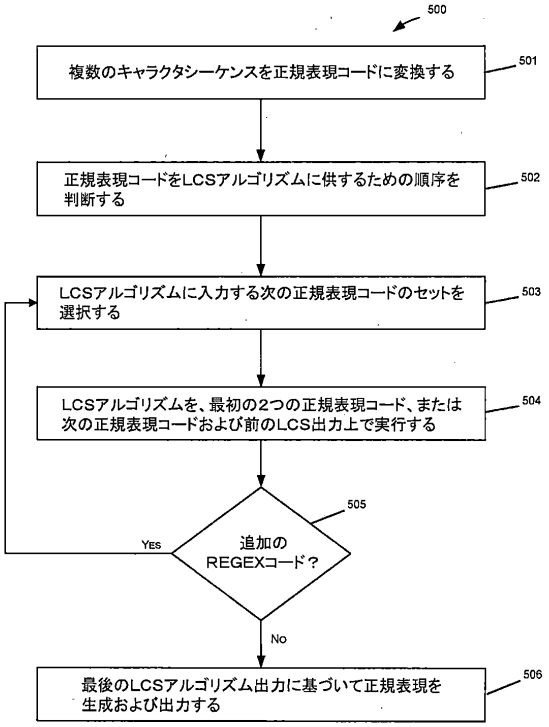



FIG. 5

【図６】

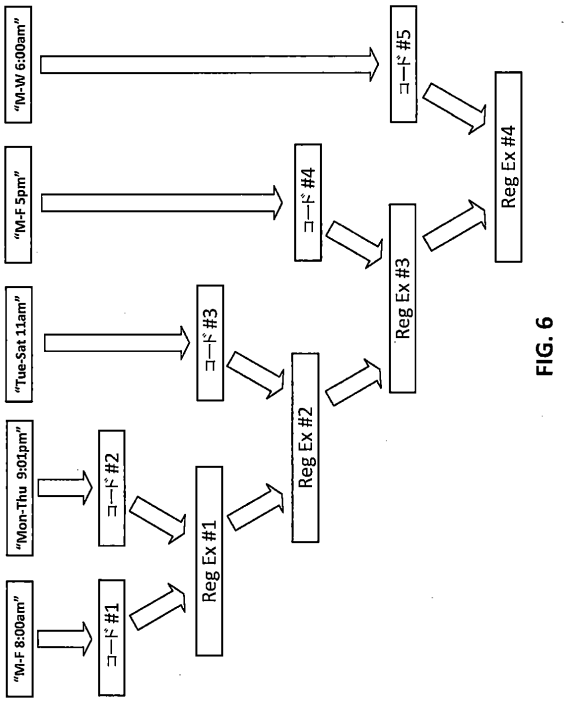



FIG. 6

【図７】

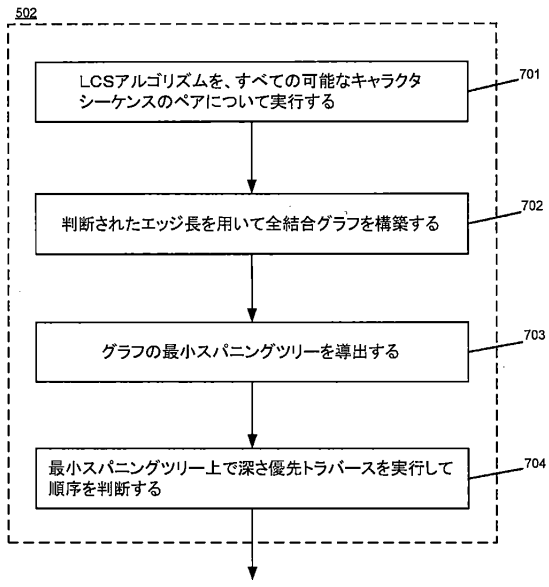



FIG. 7

【図８Ａ】

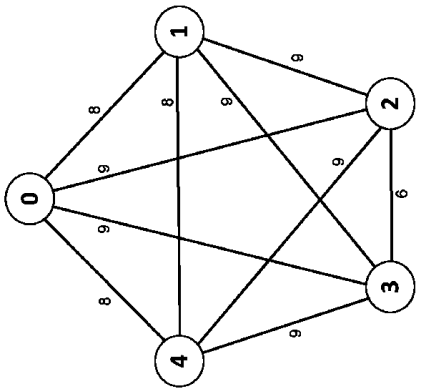



FIG. 8A

【図８Ｂ】

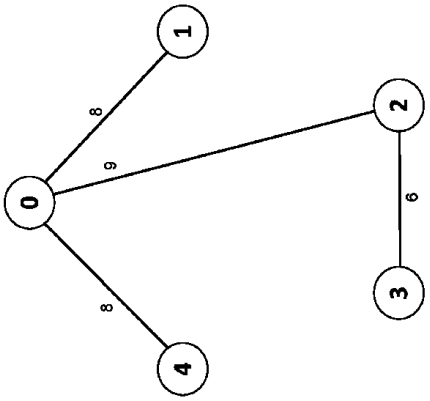



FIG. 8B

【図９】

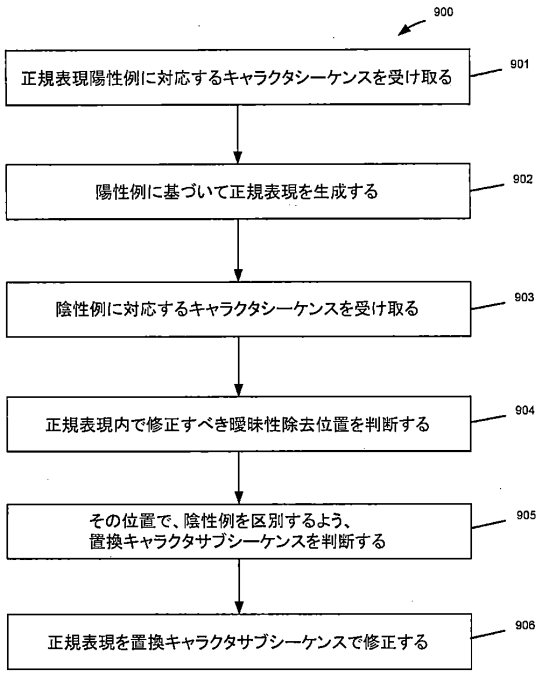



FIG. 9

【図１０Ａ】

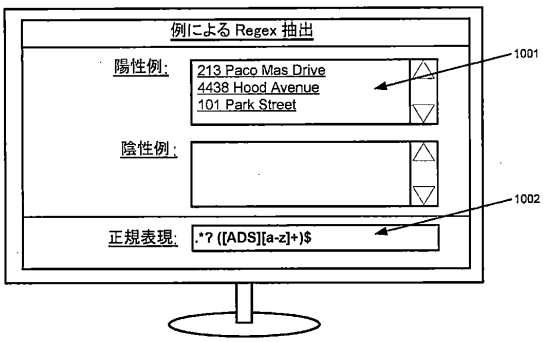



FIG. 10A

【図１０Ｂ】

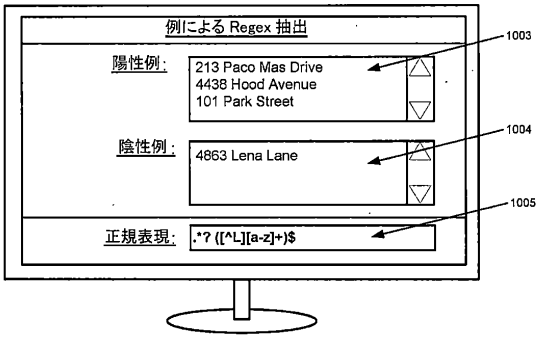



FIG. 10B

【図１１】

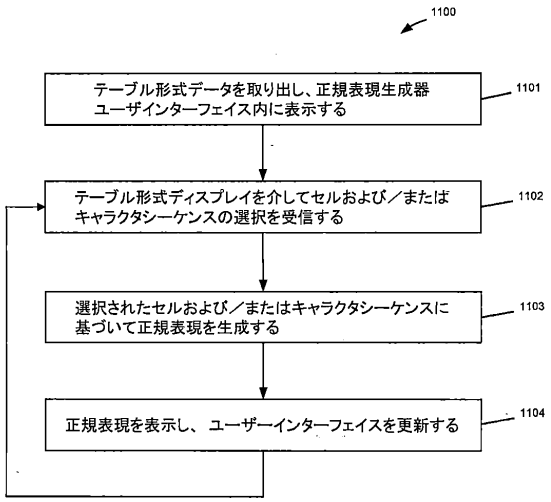



FIG. 11

【図１２】

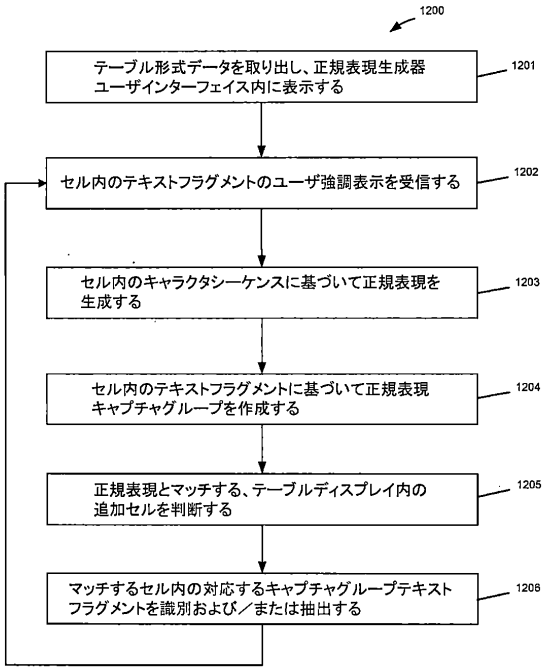



FIG. 12

【図１３】

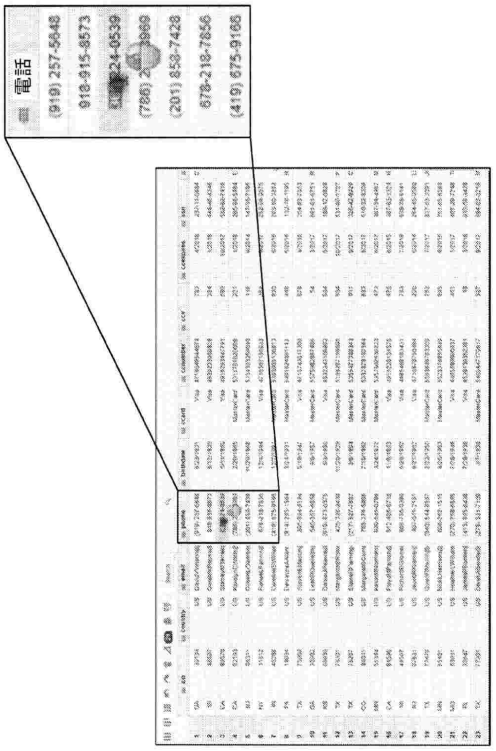



FIG. 13

【図１４】

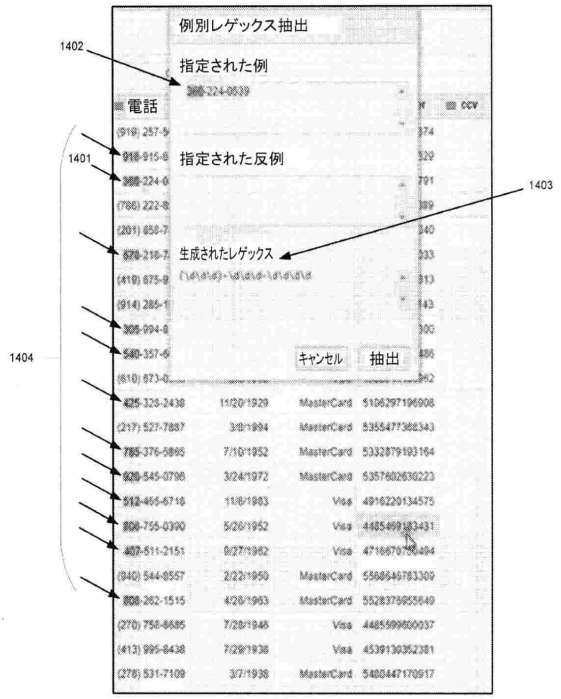



FIG. 14

【図１５】

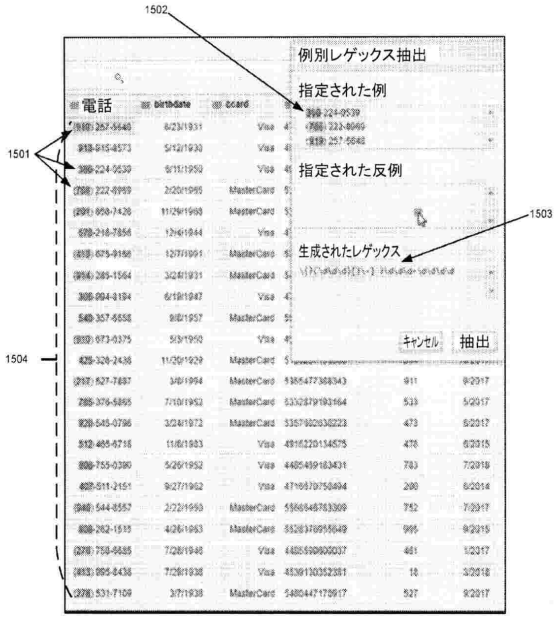



FIG. 15

【図１６Ａ】

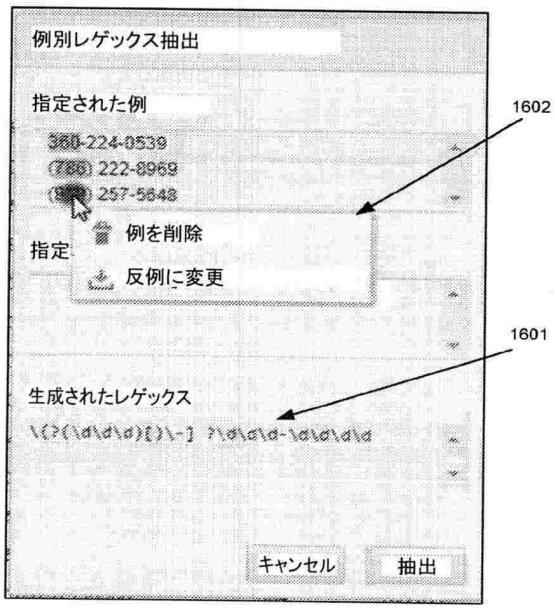



FIG. 16A

【図１６Ｂ】

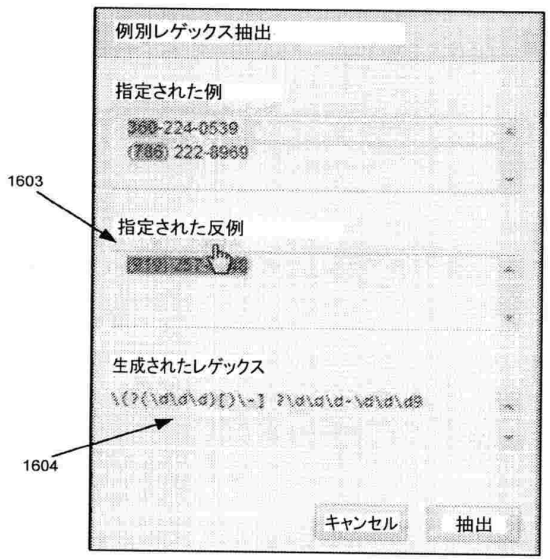



FIG. 16B

【図１７】

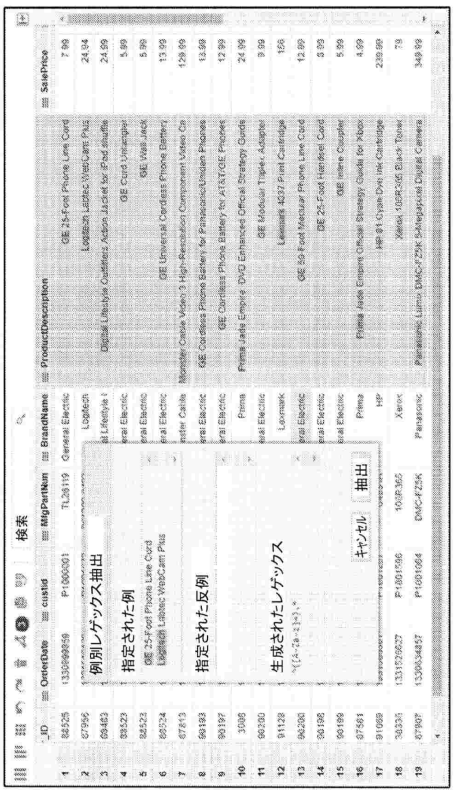



FIG. 17

【図１８】

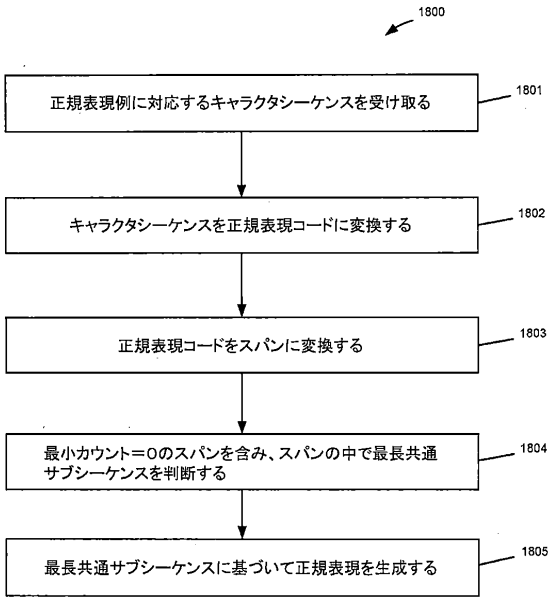



FIG. 18

【図１９】

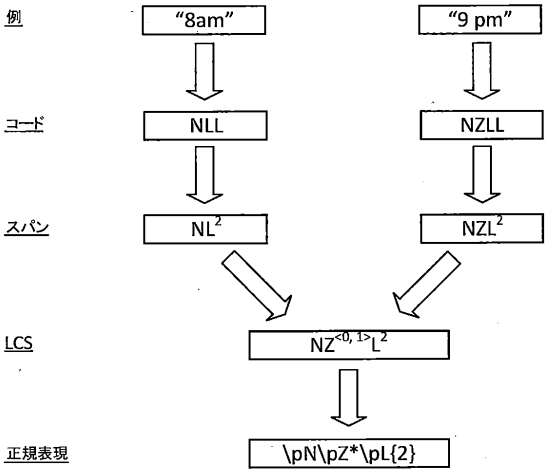



FIG. 19

【図２０】

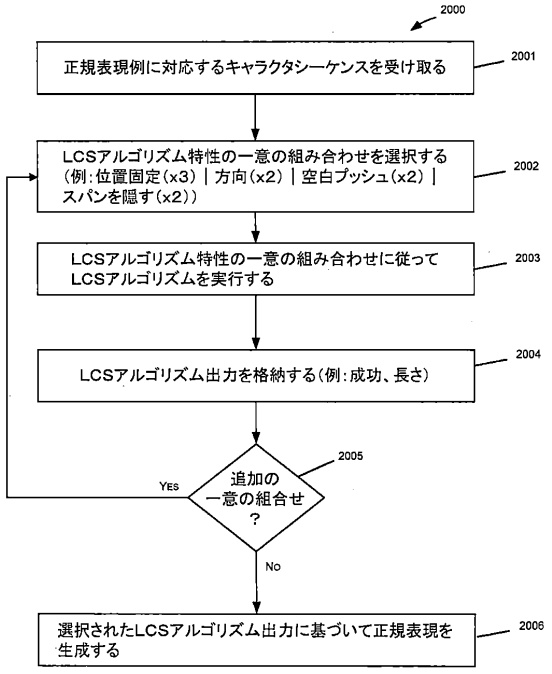



FIG. 20

【図２１】

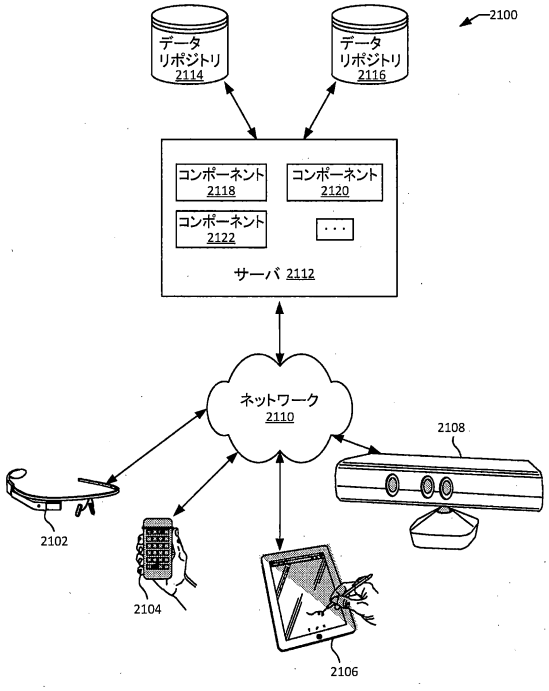



FIG. 21

【図２２】

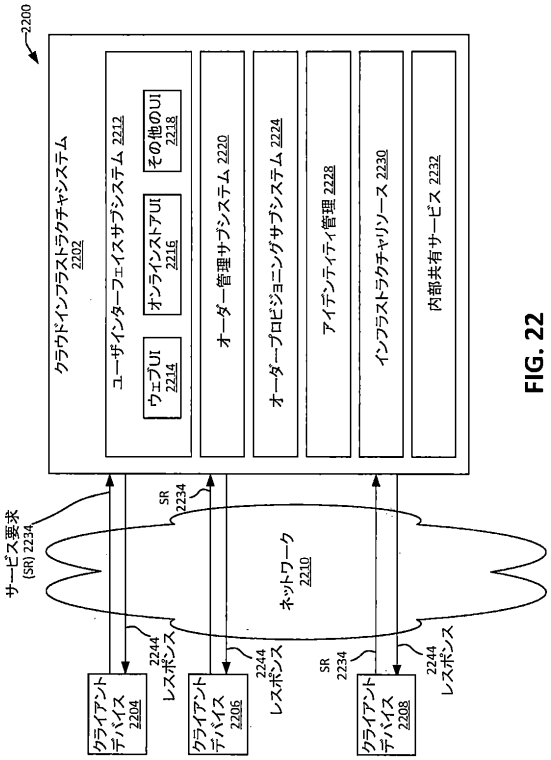



FIG. 22

【図２３】

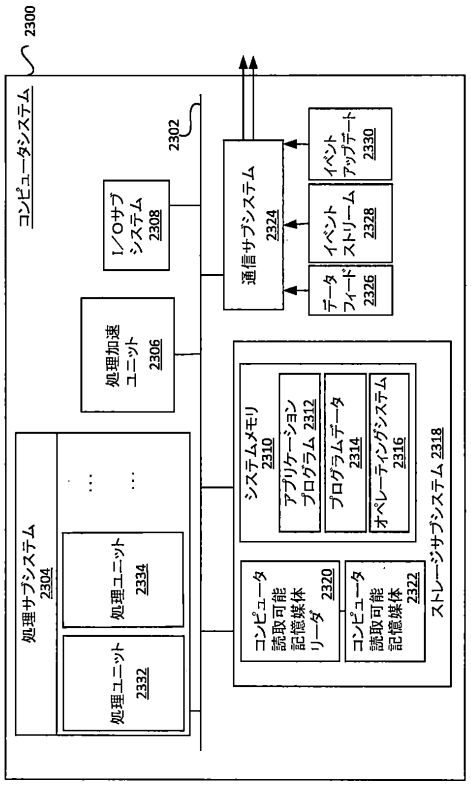



FIG. 23

【図２４】

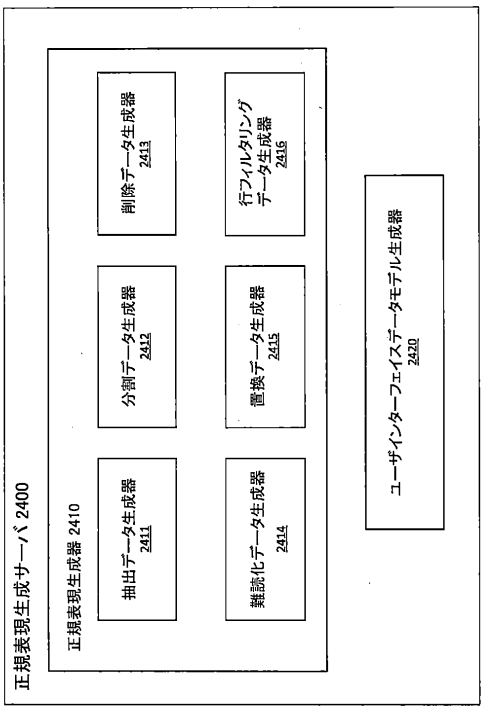



FIG. 24

【図２５Ａ】

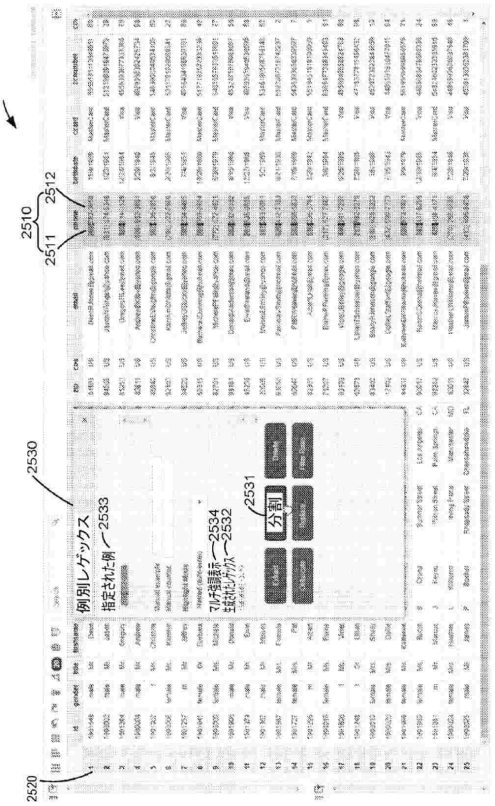



FIG. 25A

【図２５Ｂ】

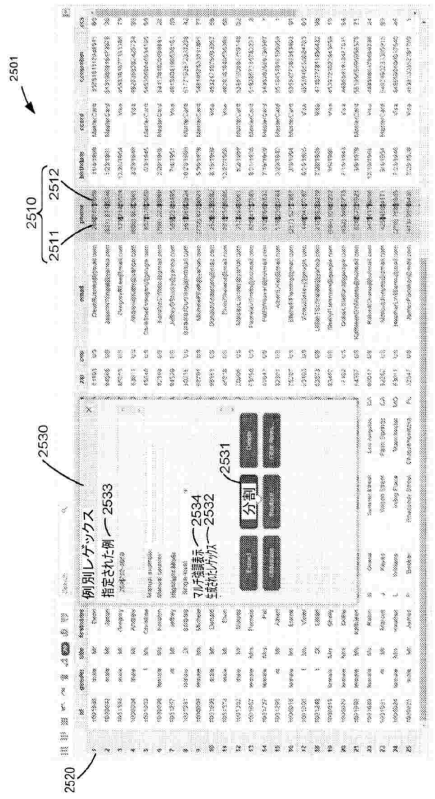



FIG. 25B

【図２６】

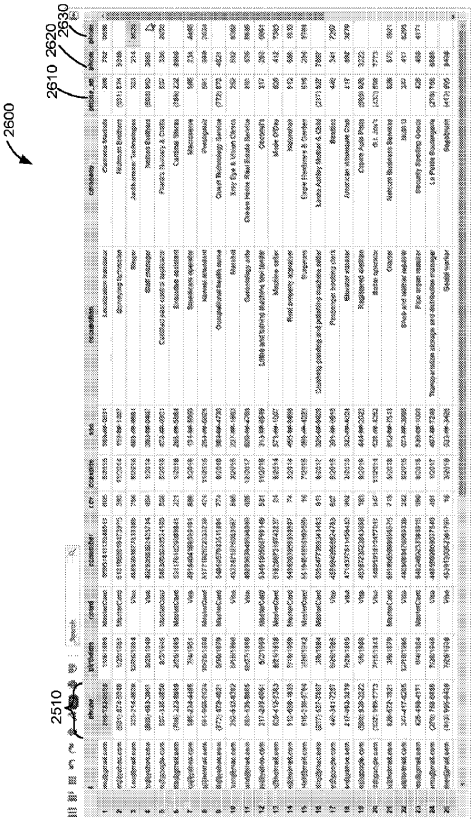



FIG. 26

【図２７】

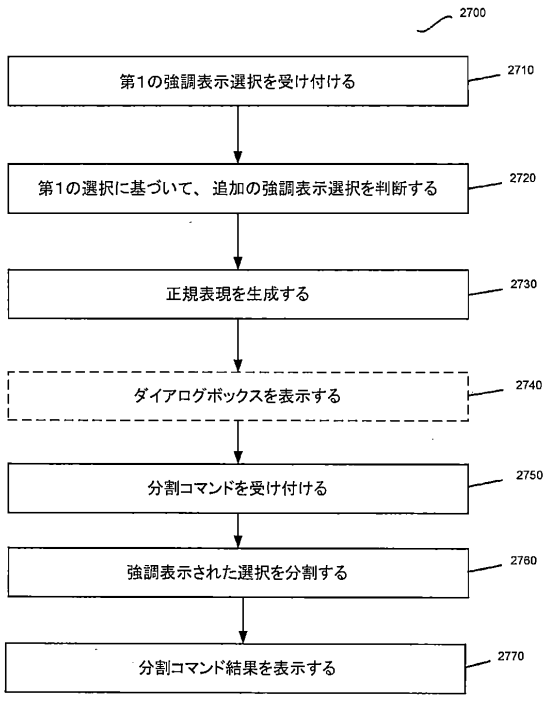



FIG. 27

【図２８】

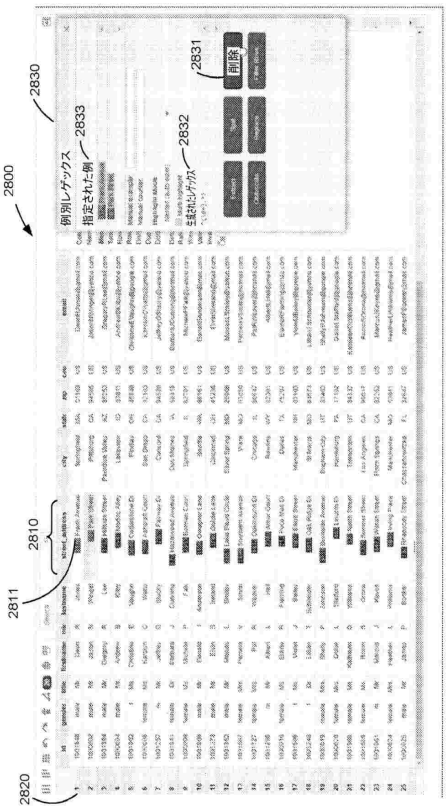



FIG. 28

【図２９】

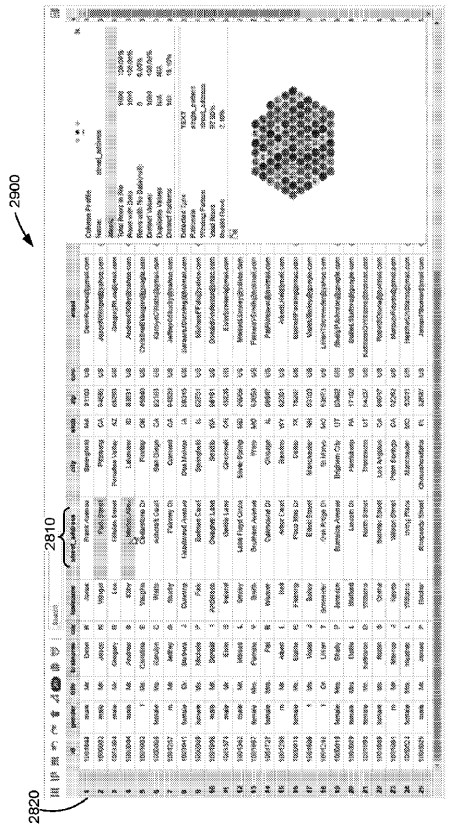



FIG. 29

【図３０】

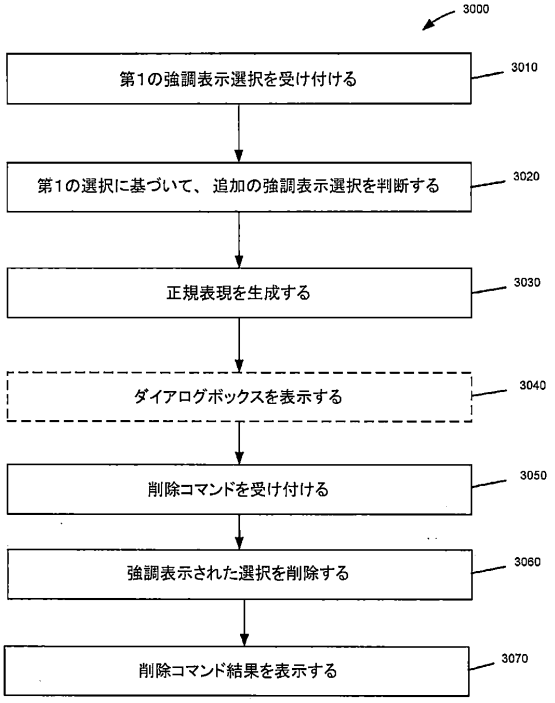



FIG. 30

【図３１】

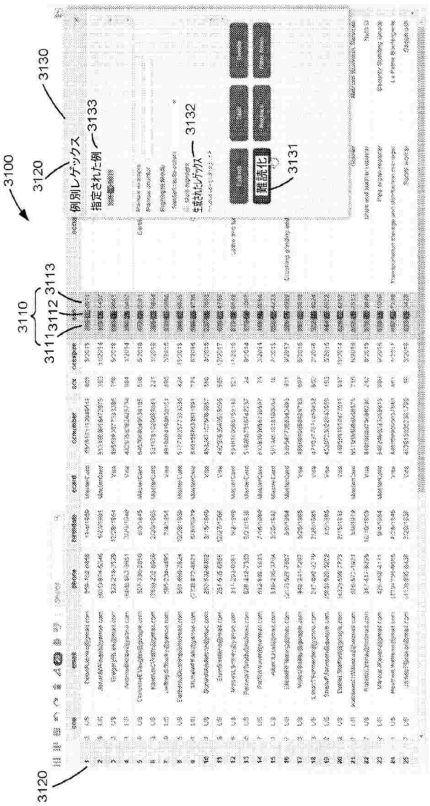



FIG. 31

【図３２】

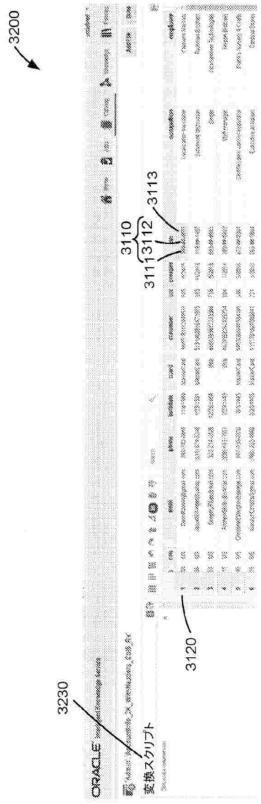



FIG. 32

【図３３】

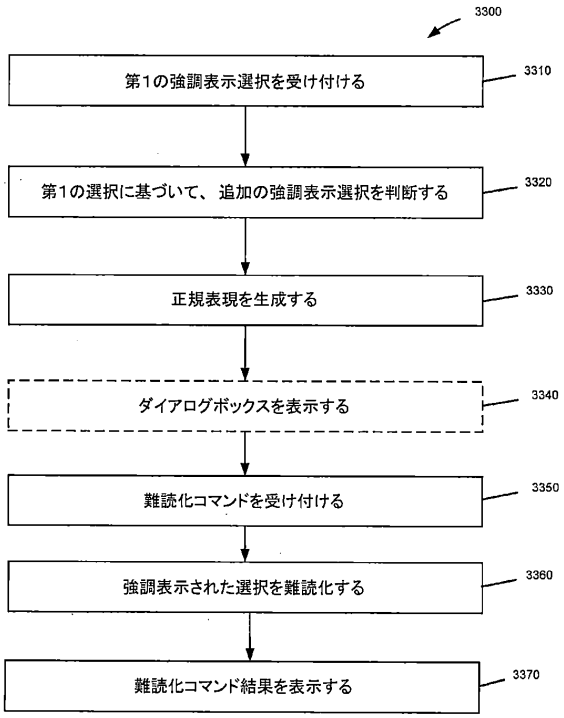



FIG. 33

【図３４】

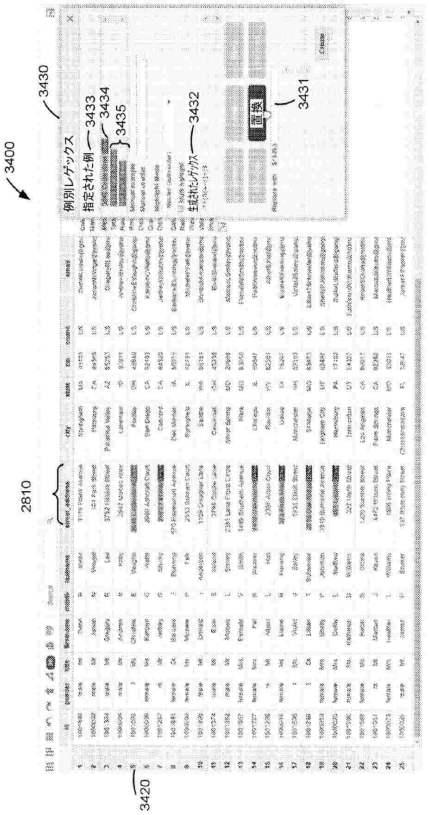



FIG. 34

【図３５】

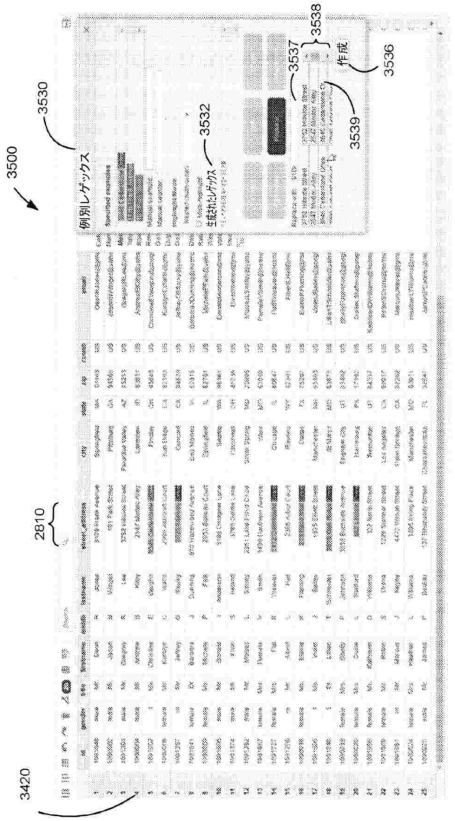



FIG. 35

【図３６】

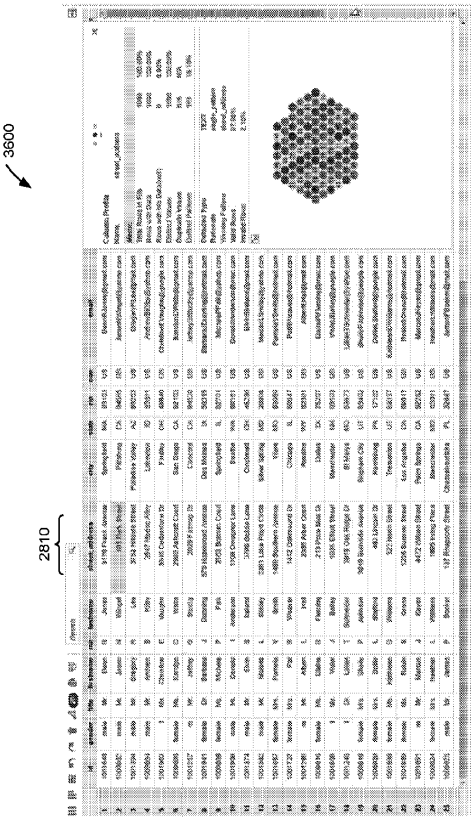



FIG. 36

【図３７】

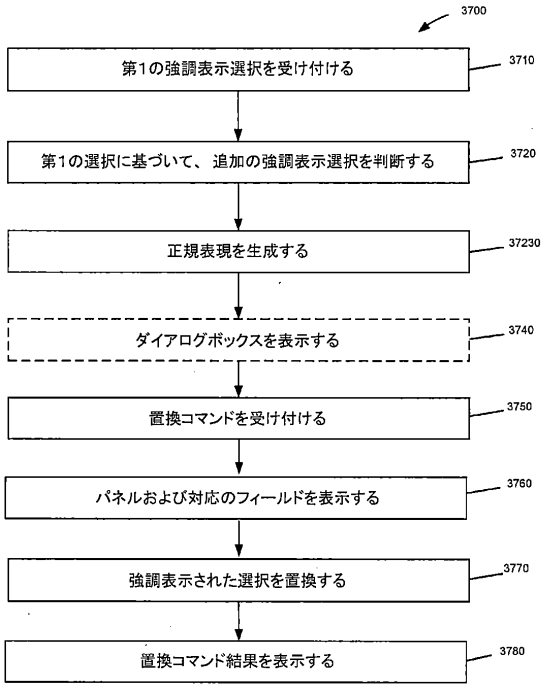



FIG. 37

【図３８】

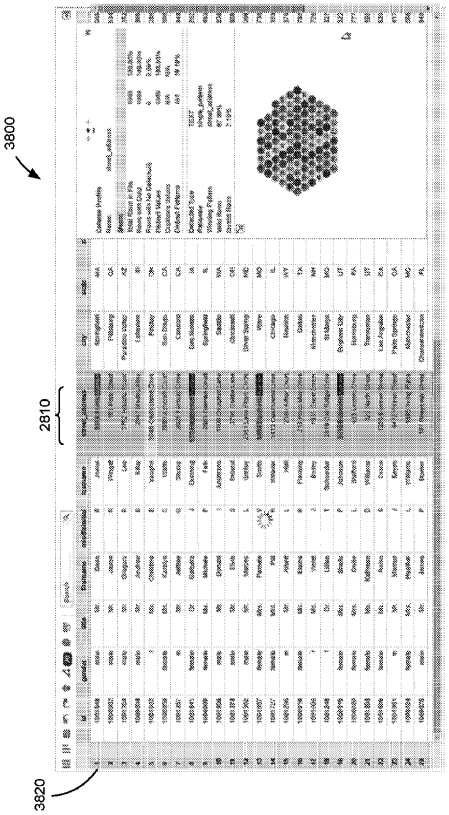



FIG. 38

【図３９】

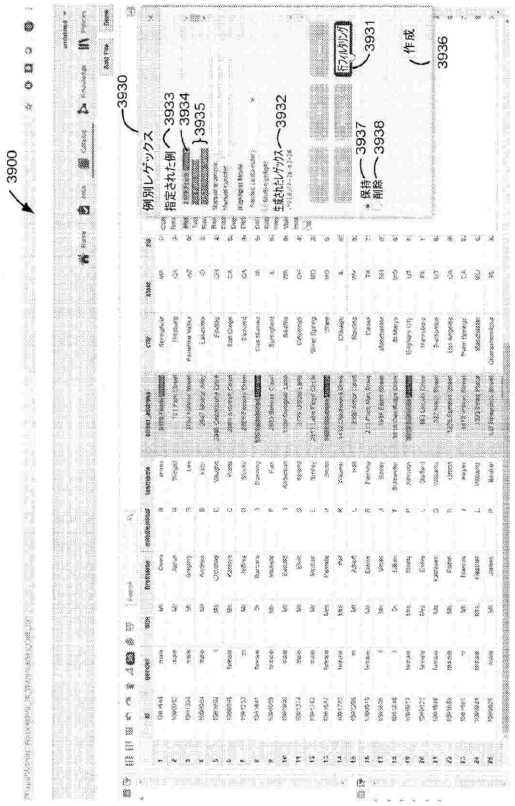



FIG. 39

【図４０】

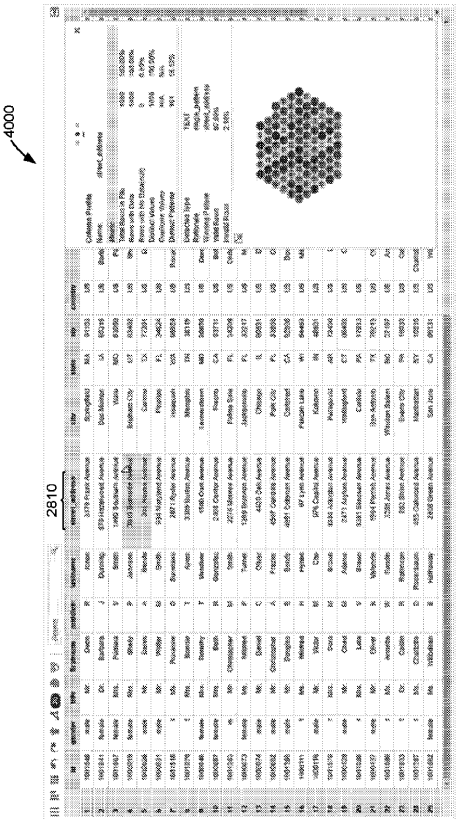



FIG. 40

【図４１】

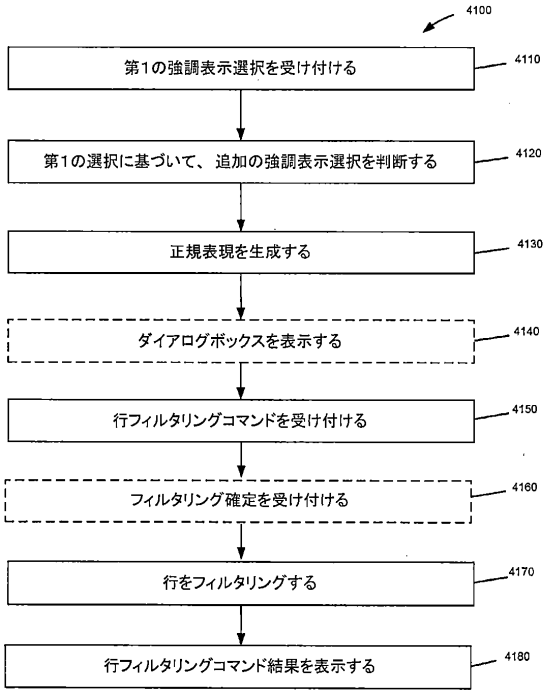



FIG. 41

【図４２】

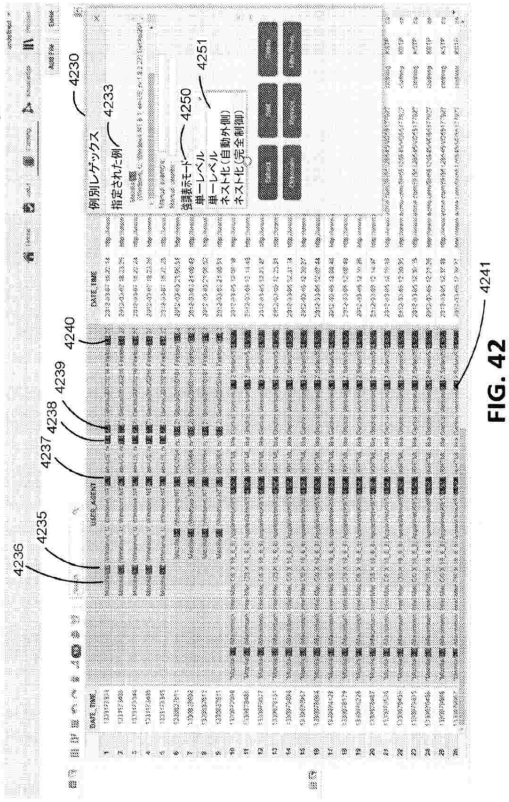



FIG. 42

【図４３】

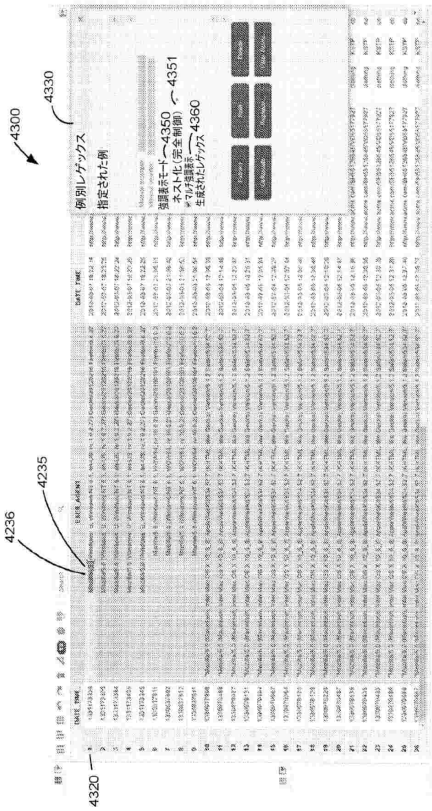



FIG. 43

【図４４】

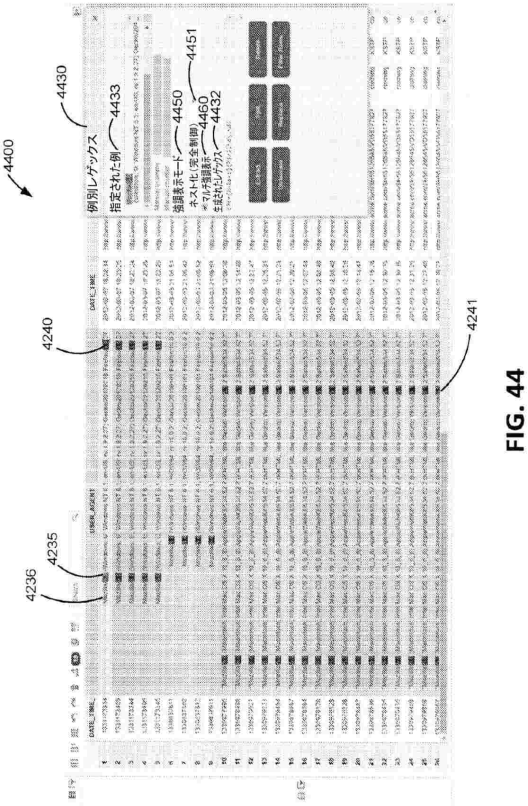



FIG. 44

【図４５】

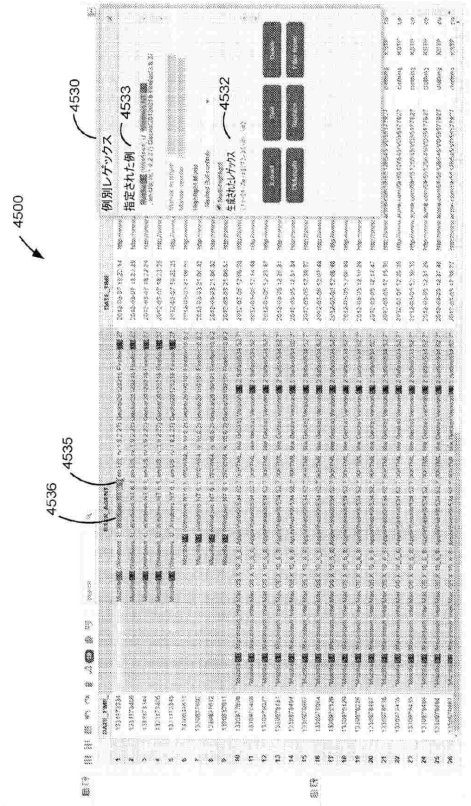



FIG. 45

【図４６】

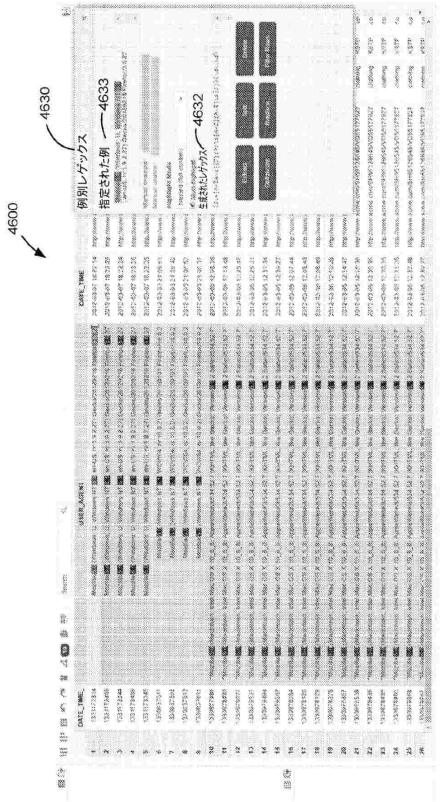



FIG. 46

【図４７】

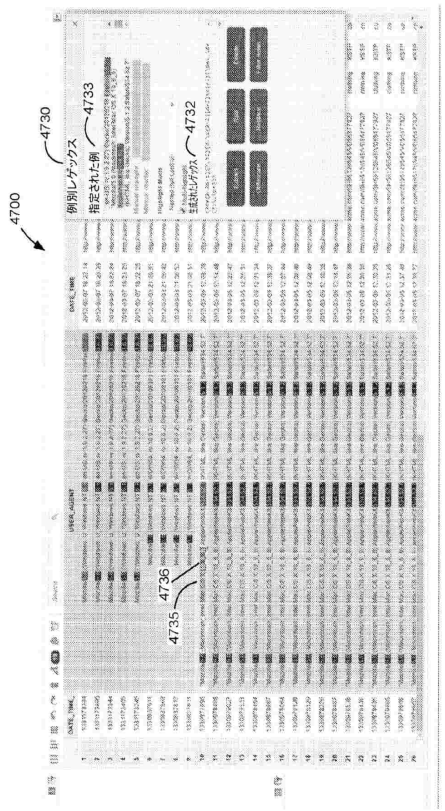



FIG. 47

【図４８】

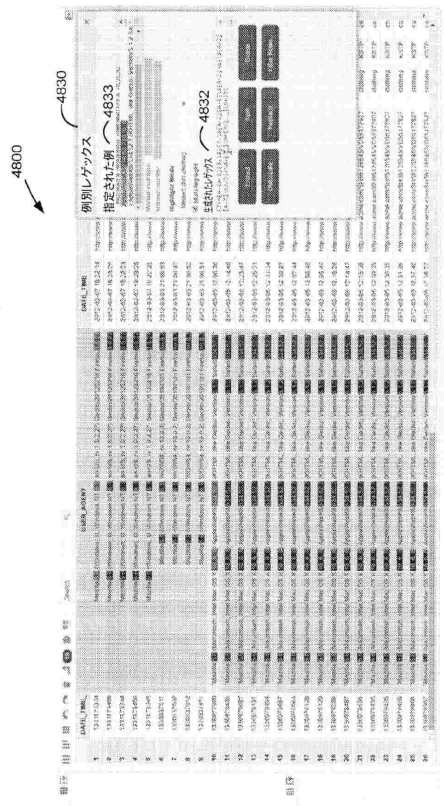



FIG. 48

【図４９】

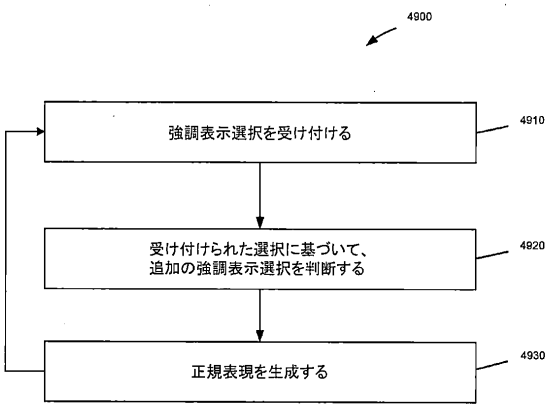



FIG. 49

【図５０】

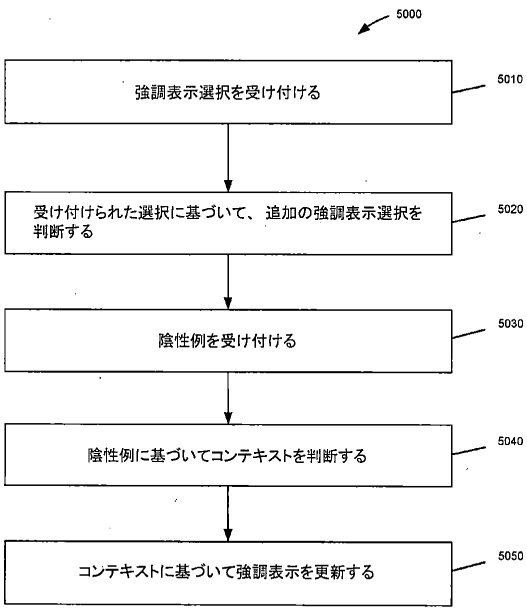



FIG. 50

【図５１】

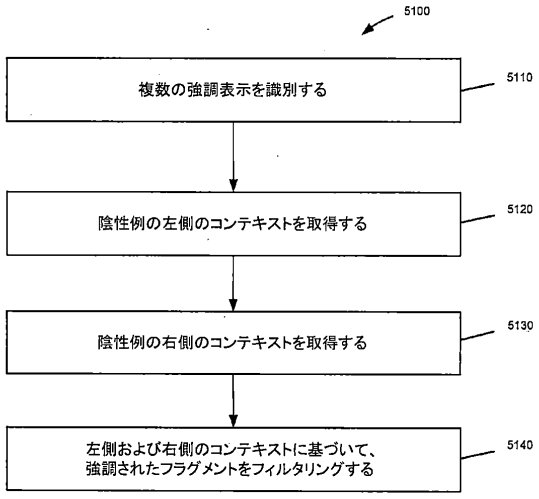



FIG. 51

【図５２】

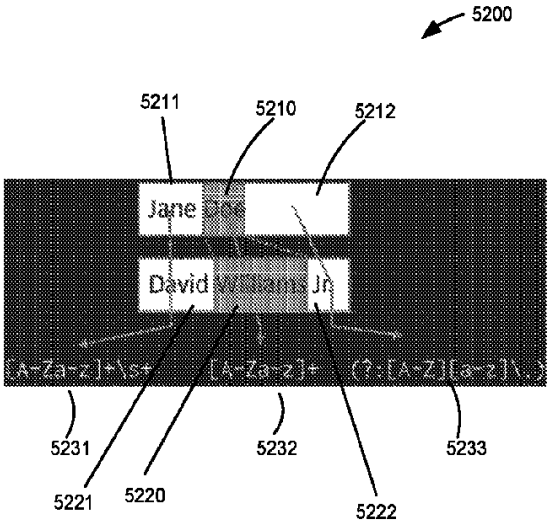



FIG. 52

【図５３】

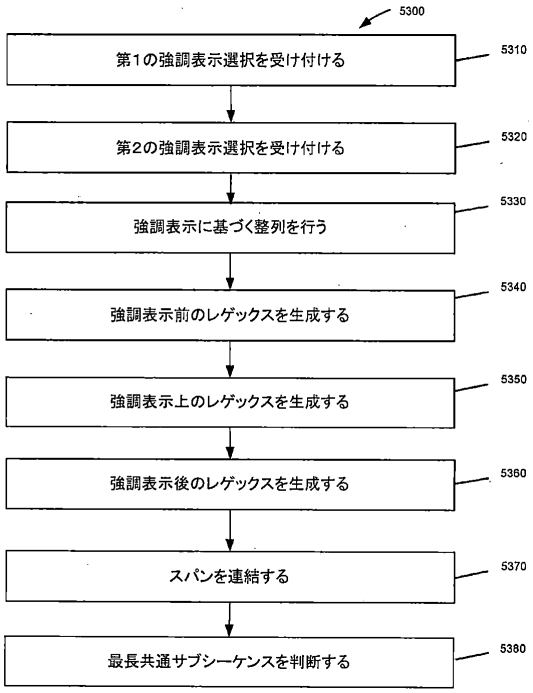



FIG. 53

【図５４】

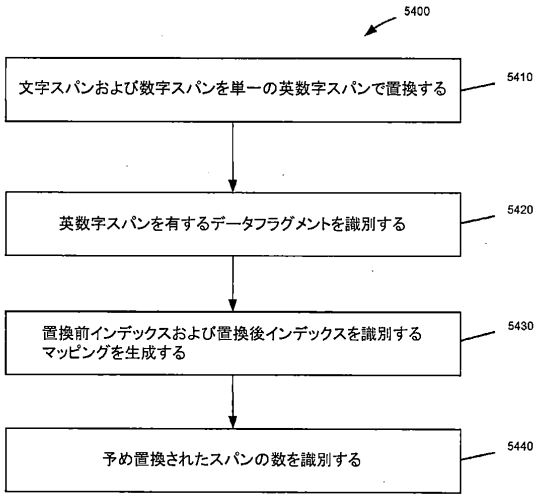



FIG. 54

【図５５】

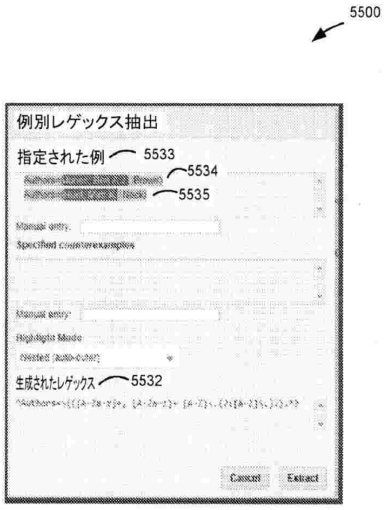



FIG. 55

フロントページの続き

(33)優先権主張国・地域又は機関
米国(US)
ルニア州、レッドウッド・シティー、オラクル・パークウェイ、５００
(72)発明者　クライダー，マーク・エル
アメリカ合衆国、９４０６５ カリフォルニア州、レッドウッド・シティー、オラクル・パークウェイ、５００
審査官　齊藤 貴孝
(56)参考文献　米国特許出願公開第２００７／０１９８５６５（ＵＳ，Ａ１）
米国特許出願公開第２０１７／０２２０２１８（ＵＳ，Ａ１）
特開２０１０－２５０４５０（ＪＰ，Ａ）
特開２０１９－０４９８２３（ＪＰ，Ａ）
(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１６／００－１６／９５８