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

特許第4872079号
(P4872079)

(45)発行日　平成24年2月8日(2012.2.8)　(24)登録日　平成23年12月2日(2011.12.2)

(51)Int.Cl.		F I		
G06F 17/24	(2006.01)	G06F	17/24	554N
G06F 17/21	(2006.01)	G06F	17/21	550A
G06F 17/30	(2006.01)	G06F	17/30	220Z
		G06F	17/30	240A

請求項の数 8　(全 22 頁)

(21)出願番号　特願2006-140850 (P2006-140850)
(22)出願日　平成18年5月19日 (2006.5.19)
(65)公開番号　特開2007-310746 (P2007-310746A)
(43)公開日　平成19年11月29日 (2007.11.29)
審査請求日　平成21年3月24日 (2009.3.24)

(73)特許権者　304021288
国立大学法人長岡技術科学大学
新潟県長岡市上富岡町１６０３－１
(74)代理人　100080089
弁理士　牛木　護
(74)代理人　100119312
弁理士　清水　栄松
(74)代理人　100119334
弁理士　外山　邦昭
(74)代理人　100137800
弁理士　吉田　正義
(72)発明者　高橋　正幸
新潟県長岡市上富岡町１６０３－１　国立大学法人長岡技術科学大学内

最終頁に続く

(54)【発明の名称】文章更新量評価プログラム

(57)【特許請求の範囲】
【請求項１】
コンピュータに、比較する更新前後のテキストから任意の閾値以上の長さを有する共通部分文字列を抽出する抽出ステップと、当該共通部分文字列の抽出数から１を減算して編集点数を求めるステップと、前記共通部分文字列を抽出元の前記各テキストにおける出現順を保持した状態で並べて分割列をそれぞれ作成するステップと、各分割列を対比して一の分割列を他の分割列と一致させるために必要な前記共通部分文字列間の最小置換回数を計算して文脈編集距離を求めるステップと、前記テキストから当該共通部分文字列を取り除いた残余の文字列の合計長さが当該テキストの全長に対して占める割合を計算して新規創作率を求めるステップと、前記各テキストから当該共通部分文字列を取り除いた残余の文字列をそれぞれ集めて非共通部分文字列集合を作成するステップと、数式１で表される
【数１】

$$DO = 1 - \frac{\left|\Gamma_1 \cap \Gamma_2\right|}{\left|\Gamma_1\right|}$$

（Γ_1，Γ_2は前記各非共通部分文字列集合から作成されるＮグラム集合、$|\Gamma_1 \cap \Gamma_2|$はＮグラム集合Γ_1及びΓ_2に共通して現れる共通要素数、$|\Gamma_1|$はＮグラム集合Γ_1に含まれるＮグラムの総数）
前記各非共通部分文字列集合間における前記閾値未満の長さによるＮグラムの不一致率である新規創作分新規度ＤＯを求める新規度評価ステップとを実行させ、

各ステップにより求められた編集点数と文脈編集距離と新規創作率と新規創作分新規度とを用いて、
評価式：ａ・ＥＰ＋ｂ・ＣＥＤ＋ＮＣＰ・ＤＯ・Ｌ
（ＥＰは編集点数，ＣＥＤは文脈編集距離，ＮＣＰは新規創作率，ＤＯは新規創作分新規度，Ｌは更新後のテキストの全長，ａ及びｂは任意の係数）
により文章更新量を算出するステップを実行させるための文章更新量評価プログラム。

【請求項２】

前記共通要素数は、前記更新前後のテキストから作成された前記各非共通部分文字列集合にそれぞれ対応する第１の文字列と第２の文字列とを終端記号で連結し、当該連結後の連結文字列に対して接尾辞配列を生成して高さ配列計算をすると共に、前記接尾辞配列作成時に接尾辞のインデックスの数値と、前記連結文字列の前方となる前記第１の文字列の文字列長との比較を行い、接尾辞の開始部分が前記第１の文字列から始まるグループと、接尾辞の開始部分が前記第２の文字列から始まるグループとに分類し、隣接する接尾辞配列が別のグループかつ所定のＮ値以上の高さを持つ位置に対応する隣接接尾辞配列間で前方一致する部分文字列の出現回数を数え上げることにより求められるものであることを特徴とする請求項１に記載の文章更新量評価プログラム。

【請求項３】

コンピュータに、前記編集点数の大小関係から各テキストの更新順序を決定するステップを実行させる請求項１記載の文章更新量評価プログラム。

【請求項４】

コンピュータに、前記文脈編集距離の大小関係から各テキストの更新順序を決定するステップを実行させるための請求項１記載の文章更新量評価プログラム。

【請求項５】

前記抽出ステップは、前記更新前後の各テキストにそれぞれ対応する第１の文字列と第２の文字列とを終端記号で連結し、当該連結後の連結文字列に対して接尾辞配列を生成して高さ配列計算をすると共に、前記接尾辞配列作成時に接尾辞のインデックスの数値と、前記連結文字列の前方となる前記第１の文字列の文字列長との比較を行い、接尾辞の開始部分が前記第１の文字列から始まるグループと、接尾辞の開始部分が前記第２の文字列から始まるグループとに分類し、隣接する接尾辞配列が別のグループかつ最大の高さを持つ位置に対応する隣接接尾辞配列間で前方一致する文字列を最長共通部分文字列として前記テキストから抽出し、残余の文字列から前記最長共通部分文字列の長さが前記閾値以下となるまで最長共通部分文字列の抽出を繰り返すものであることを特徴とする請求項１～４のいずれか１つに記載の文章更新量評価プログラム。

【請求項６】

前記抽出ステップは、前記共通部分文字列を抽出する際にそれぞれ別の特殊文字に置換するものであることを特徴とする請求項１～５のいずれか１つに記載の文章更新量評価プログラム。

【請求項７】

前記抽出ステップは、前記各テキストを対比して作成されたドットマトリックス上にプロットされたドットにおける前記ドットマトリックスの中心からのオフセットが、前記閾値以上の回数にわたって連続で一定の値として出現する箇所に対応する文字列を共通部分文字列として前記テキストから抽出するものであることを特徴とする請求項１～６のいずれか１つに記載の文章更新量評価プログラム。

【請求項８】

コンピュータに、前記共通部分文字列を抽出元の前記各テキストにおける出現順を保持した状態で並べて分割列をそれぞれ作成するステップと、前記各テキストから作成された前記各分割列を対比して、前記各分割列に含まれる前記共通部分文字列単位の一致における共通部分文字列を構成する各文字をドットとしてプロットすることにより作成されたドットマトリックスを表示させるステップとを実行させるための請求項１～７のいずれか１つ記載の文章更新量評価プログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、複数の文字列データを比較し、一の文字列データに対する他の文字列データの更新量を評価する文章更新量評価プログラムに関する。
【背景技術】
【０００２】
文章の更新量を計測する、という必要はさまざまな局面で発生する。例えば、翻訳テキストの添削を行なう校正者の作業量の評価、剽窃が疑われる文書間における創作量や模倣量の評価、継続的に更新の加えられている文書に関する更新量の定量的評価を含む履歴管理、バージョンの順序関係が不明となった文書間の変更履歴復元、自動生成されるＷｅｂページ内の新規記事部分の抽出などである。これらに共通するのは、文章の編集，更新過程における知的な作業量を定量的に把握するという課題であり、こうした作業量は単純に作業時間やファイルサイズの変化量で評価することは出来ない。
【０００３】
文章の変更量を評価する指標として、別の文章に変更するための最小操作回数を表す「レーベンシュタイン距離」（Levenstein Distance）があり、単に「編集距離」（Edit Distance）とも呼ばれ、文章間の評価指標に留まらず、近年急速な発展を遂げるバイオインフォマティクス分野におけるＤＮＡ配列間の類似性評価にまで応用されている。例えば、特許文献１では、このレーベンシュタイン距離を用いて２つの文字列の近似度を判定している。
【特許文献１】特開平６－８３８７１号公報
【発明の開示】
【発明が解決しようとする課題】
【０００４】
しかし、文章の編集，更新量を評価するという目的に照らすとき、レーベンシュタイン距離にも不満足な点が残る。例えば、レーベンシュタイン距離を用いたものでは、１文字の綴り間違いを100箇所修正する作業と、100文字からなる新規の部分を創作する作業とは全く同等に評価されるが、知的な作業としてはこれらを区別して扱いたい。また、単字や単語の修正と文脈の変更まで伴う編集とは区別して評価することが必要である。いわば、文章編集，更新という人間の知的な作業を、極力その作業類型に応じた作業量の集積として評価する方法が求められているといえる。
【０００５】
そこで本発明は上記問題点に鑑み、文章の編集・更新過程における知的作業量の定量的把握を可能とした文章更新量評価プログラムを提供することを目的とする。
【課題を解決するための手段】
【０００６】
本発明における請求項１の文章更新量評価プログラムでは、コンピュータに、比較する更新前後のテキストから任意の閾値以上の長さを有する共通部分文字列を抽出する抽出ステップと、当該共通部分文字列の抽出数から１を減算して編集点数を求めるステップと、前記共通部分文字列を抽出元の前記各テキストにおける出現順を保持した状態で並べて分割列をそれぞれ作成するステップと、各分割列を対比して一の分割列を他の分割列と一致させるために必要な前記共通部分文字列間の最小置換回数を計算して文脈編集距離を求めるステップと、前記テキストから当該共通部分文字列を取り除いた残余の文字列の合計長さが当該テキストの全長に対して占める割合を計算して新規創作率を求めるステップと、前記各テキストから当該共通部分文字列を取り除いた残余の文字列をそれぞれ集めて非共通部分文字列集合を作成するステップと、数式２で表される

【数２】

$$DO = 1 - \frac{\left|\Gamma_1 \cap \Gamma_2\right|}{\left|\Gamma_1\right|}$$

（Γ$_1$，Γ$_2$は前記各非共通部分文字列集合から作成されるＮグラム集合、｜Γ$_1$∩Γ$_2$｜はＮグラム集合Γ$_1$及びΓ$_2$に共通して現れる共通要素数、｜Γ$_1$｜はＮグラム集合Γ$_1$に含まれるＮグラムの総数）

前記各非共通部分文字列集合間における前記閾値未満の長さによるＮグラムの不一致率である新規創作分新規度ＤＯを求める新規度評価ステップとを実行させ、

各ステップにより求められた編集点数と文脈編集距離と新規創作率と新規創作分新規度とを用いて、

評価式：ａ・ＥＰ＋ｂ・ＣＥＤ＋ＮＣＰ・ＤＯ・Ｌ（ＥＰは編集点数，ＣＥＤは文脈編集距離，ＮＣＰは新規創作率，ＤＯは新規創作分新規度，Ｌは更新後のテキストの全長，ａ及びｂは任意の係数）

により文章更新量を算出するステップを実行させる。

【０００７】

新規創作，削除，置換といった編集作業が行われると元の文章から継承している文字列が分断されることに着目し、編集が行われていない部分で挟まれた、例えば、単字・単語等の比較的短い文字の挿入・削除，誤字訂正，「てにをは」の修正，単語の置き換えなどの細かい編集作業が行われた箇所を一纏まりとして、編集が行われた箇所である編集点の数を求めることができる。また、文章全域にわたる論旨の組替え，順序変更などの文脈編集では、更新前後に共通する文章に対してその順序を組替える置換操作のみが行われることに着目し、計算対象を当該置換操作が行なわれる共通部分文字列に限定することで、文脈編集以外の例えば文字の新規挿入や削除などの編集作業を除外して、文脈編集についての編集距離のみを計算することができる。また、更新後の文章の全長に対してどの程度新規な文章が創作・追加されているかを評価することができる。そして、更新後の文章のうち更新によって追加された部分の新規性を評価することができる。例えば、ア）単字・単語等の比較的短い文字の挿入・削除，誤字訂正，「てにをは」の修正，単語の置き換え、イ）文章全域にわたる論旨の組替え，順序変更，文脈編集、ウ）新規創作・追加、という作業類型に対して、編集点数と文脈編集距離と新規創作率と新規創作分新規度という評価項目で評価することにより、文章編集・更新という知的作業を、その作業類型に応じた作業量の集積として評価することができる。

【０００８】

【０００９】

【００１０】

【００１１】

【００１２】

【００１３】

【００１４】

【００１５】

本発明における請求項２の文章更新量評価プログラムでは、前記共通要素数は、前記更新前後のテキストから作成された前記各非共通部分文字列集合にそれぞれ対応する第１の文字列と第２の文字列とを終端記号で連結し、当該連結後の連結文字列に対して接尾辞配

列を生成して高さ配列計算をすると共に、前記接尾辞配列作成時に接尾辞のインデックスの数値と、前記連結文字列の前方となる前記第１の文字列の文字列長との比較を行い、接尾辞の開始部分が前記第１の文字列から始まるグループと、接尾辞の開始部分が前記第２の文字列から始まるグループとに分類し、隣接する接尾辞配列が別のグループかつ所定のＮ値以上の高さを持つ位置に対応する隣接接尾辞配列間で前方一致する部分文字列の出現回数を数え上げることにより求められるものであることを特徴とする。

【００１６】

このようにすると、一致するパターンが別々のテキストに含まれるパターンなのか同一テキストに含まれるパターンなのかを判別して、各非共通部分文字列集合で共通する部分文字列を探索することができる。

【００１７】

【００１８】

【００１９】

本発明における請求項３の文章更新量評価プログラムでは、コンピュータに、前記編集点数の大小関係から各テキストの更新順序を決定するステップを実行させる。

【００２０】

このようにすると、複数のテキスト間について求められた編集点数から更新順序を決定することにより、文章更新量を評価すると共に改定履歴の復元を行うことができる。

【００２１】

本発明における請求項４の文章更新量評価プログラムでは、コンピュータに、前記文脈編集距離の大小関係から各テキストの更新順序を決定するステップを実行させる。

【００２２】

このようにすると、複数のテキスト間について求められた文脈編集距離から更新順序を決定することにより、文章更新量を評価すると共に改定履歴の復元を行うことができる。

【００２３】

本発明における請求項５の文章更新量評価プログラムでは、前記抽出ステップは、前記更新前後の各テキストにそれぞれ対応する第１の文字列と第２の文字列とを終端記号で連結し、当該連結後の連結文字列に対して接尾辞配列を生成して高さ配列計算をすると共に、前記接尾辞配列作成時に接尾辞のインデックスの数値と、前記連結文字列の前方となる前記第１の文字列の文字列長との比較を行い、接尾辞の開始部分が前記第１の文字列から始まるグループと、接尾辞の開始部分が前記第２の文字列から始まるグループとに分類し、隣接する接尾辞配列が別のグループかつ最大の高さを持つ位置に対応する隣接接尾辞配列間で前方一致する文字列を最長共通部分文字列として前記テキストから抽出し、残余の文字列から前記最長共通部分文字列の長さが前記閾値以下となるまで最長共通部分文字列の抽出を繰り返すものであることを特徴とする。

【００２４】

このようにすると、探索された最長共通部分文字列が別々のテキストに含まれる文字列なのか同一テキストに含まれる文字列なのかを判別して、各テキストで共通する最長共通部分文字列を探索することができる。

【００２５】

本発明における請求項６の文章更新量評価プログラムでは、前記抽出ステップは、前記共通部分文字列を抽出する際にそれぞれ別の特殊文字に置換するものであることを特徴とする。

【００２６】

このようにすると、共通部分文字列を抽出した後の残余の文字列中に、所定以上の長さを有する共通部分文字列が偶発的に発生することを防止することができる。

【００２７】

本発明における請求項７の文章更新量評価プログラムでは、前記抽出ステップは、前記

各テキストを対比して作成されたドットマトリックス上にプロットされたドットにおける前記ドットマトリックスの中心からのオフセットが、前記閾値以上の回数にわたって連続で一定の値として出現する箇所に対応する文字列を共通部分文字列として前記テキストから抽出するものであることを特徴とする。

【００２８】

このようにすると、ドットマトリックスを利用して共通部分文字列の探索を行うことができる。

【００２９】

本発明における請求項８の文章更新量評価プログラムでは、前記共通部分文字列を抽出元の前記各テキストにおける出現順を保持した状態で並べて分割列をそれぞれ作成するステップと、前記各テキストから作成された前記各分割列を対比して、前記各分割列に含まれる前記共通部分文字列単位の一致における共通文字列を構成する各文字をドットとしてプロットすることにより作成されたドットマトリックスを表示させる。

【００３０】

このようにすると、文章更新量を評価するにあたり、文字列を構成する各文字ではなく、共通部分文字列をドットの要素としたドットマトリックスを作成することにより、更新前後のテキスト間に共通する文章に行われた編集作業による文章形態の変化の程度のみがドットとしてプロットされ、文章全体の編集作業の傾向を視覚的に表示させることができる。

【発明の効果】

【００３１】

本発明の請求項１によると、文章更新量の大局評価に関して、人間が編集後の文章を見た時に感じる「文章内の大よそ何箇所に手が加わっている」という感覚に近い評価が期待できる。また、文章更新量を文脈編集、新規創作、新規創作された部分についての更新前のテキストに対する新規性という観点から評価することができ、文章の編集・更新過程における知的作業量全般にわたる定量的把握が可能となる。

【００３２】

【００３３】

【００３４】

【００３５】

本発明の請求項２によると、Ｎグラム集合における共通要素数を計算する際の計算量を減らしてコンピュータ上で動かす場合の実現可能性と実行速度を向上させることができる。

【００３６】

【００３７】

本発明の請求項３によると、複数のテキストの中から所望のバージョンのテキストを選び出すことが出来る。

【００３８】

本発明の請求項４によると、複数のテキストの中から所望のバージョンのテキストを選び出すことが出来る。

【００３９】

本発明の請求項５によると、テキスト間での文字列照合を行う際の計算量を減らしてコンピュータ上で動かす場合の実現可能性と実行速度を向上させることができる。

【００４０】

本発明の請求項６によると、より正確に共通部分文字列を抽出することができる。

【００４１】

本発明の請求項７によると、共通部分文字列の探索を容易に行うことができる。

【００４２】

本発明の請求項８によると、文章更新量の視覚的把握が可能となる。

【発明を実施するための最良の形態】

【００４３】

以下、添付図面を参照しながら、本発明における文章更新量評価プログラムの好ましい実施例を説明する。

【００４４】

図１は、本文章更新量評価プログラムをインストールしたコンピュータの概略構成を示したものである。同図において、コンピュータ１は一般的なハードウェア構成を備えており、少なくとも、例えばポインティングデバイス，キーボードなどの入力部２と、中央演算処理装置であるＣＰＵ３と、例えばハードディスクなどの記憶装置４と、例えばディスプレイ装置などの表示部５とを具備する。ＣＰＵ３は、これらのハードウェア構成を有機的に結合し、文章更新量評価プログラム10に実装された文章更新量評価アルゴリズムにおける一連の情報処理を実行する。文章更新量評価プログラム10は、他のデータと共に記憶装置４に格納されており、ＣＰＵ３が入力部２の操作入力に従って文章更新量評価プログラム10の処理を適宜実行し、その処理結果を表示部５にて表示する。なお、同図では、コンピュータ１に文章更新量評価プログラム10をインストールした例を示したが、例えばＷｅｂサーバやＡＳＰサーバなどのサーバにより文章更新量評価プログラム10の処理結果のみがインターネット等のネットワークを通じてクライアントとなるコンピュータ１へ提供されるよう構成してもよい。

【００４５】

以下、文章更新量評価プログラム10に実装された文章更新量評価アルゴリズムについて詳述する。

【００４６】

本文章更新量評価アルゴリズムにおいては、人間の実際の文章編集作業に即した新規の評価モデルに基づき、文字列照合で生成した部分文字列を多段階的に分析することにより総合的に評価する。当該文章更新量評価のモデルを策定するに当たり、与えられた２つの文章間の更新量を評価するモデルとして人間の実際の文章編集作業を念頭におき、本文章更新量評価アルゴリズムでは、ア）単字・単語等の比較的短い文字の挿入・削除，誤字訂正，「てにをは」の修正，単語の置き換え、イ）文章全域にわたる論旨の組替え，順序変更，文脈編集、ウ）新規創作・追加、という以上の３項目に則した評価を行なうことから最終的な文章更新量を評価する。これらの編集作業の評価を行うには、いずれにしても文章内の編集されていない共通部分を見つけることが基本である。

【００４７】

上記評価モデルの項目ア）比較的短い置換・削除・挿入に対応する評価項目が「編集点数」である。この「編集点数」は、編集が行われた箇所である編集点の数を表し、共通部分文字列集合の要素数により決定される。新規創作，削除，置換といった編集作業が行われると元の文章から継承している文字列が分断されるために結果として編集点の数が増える。後述する閾値ｔの働きにより、連続的に発生する編集に対しては非共通部分文字列により１箇所にまとめられるため、人間が編集後の文章を見た時に感じる「文章内の大よそ何箇所に手が加わっている」という感覚に近い評価が期待できる。

【００４８】

上記評価モデルの項目イ）文脈編集に対応する評価項目が「文脈編集距離」である。この「文脈編集距離」は、文章全域にわたる論旨の組換え、順序変更のように文脈編集が行われた度合いを示す評価項目であり、編集前後の各文章において前記編集点で分割して得られた各共通部分文字列の出現順を保持した並びを分割列と呼び、当該分割列間の編集距離が文脈編集距離となる。

【００４９】

上記評価モデルの項目ウ）新規創作に対応する評価項目が「新規創作率」及び「新規創

作分新規度」である。「新規創作率」は、更新後の文章の全長に対する非共通部分文字列の合計長の割合を表し、「新規創作分新規度」は、各非共通部分文字列集合間に含まれる長さＮ（０＜Ｎ＜ｔ、ｔは任意の閾値）のN-gram（Ｎグラム）の不一致率であり、更新によって追加された部分の新規性を表す。

【００５０】

これらの評価項目「編集点数」，「文脈編集距離」，「新規創作率」，及び「新規創作分新規度」の詳しい算出方法については後述するが、当該評価項目を対比するテキスト（文字列データ）間に適用することにより、文章の編集・更新過程における知的作業量が定量的に表された文章更新量を求めることができる。

【００５１】

図２は、各部分文字列集合と評価項目との関係を示した概念図である。この図２を参照しながら、文章更新量の計算方法について概略的に説明する。なお、同図においては、比較するテキスト間に存在する閾値ｔ以上の長さで一致する文字列をｃで表記し、比較するテキストから共通部分文字列を削除して出来る残余文字列のうち、更新前のテキストから作成されたものをα、更新後のテキストから作成されたものをβで表記している。

【００５２】

文章更新量の計算方法として、比較するテキスト間に存在する、共通部分文字列の遷移状態の評価と、非共通部分文字列における連続するＮ文字の出現頻度を扱うN-gramモデルを用いた評価を併せたものを提案する。共通部分文字列および非共通部分文字列の定義については後述する。文章更新量の計算に際し、初めに比較するテキストをある閾値ｔ以上の長さで完全一致する共通部分文字列c_0，c_1，c_2，c_3とそれ以外の非共通部分文字列α_0，β_0，β_1とに分類する。この作業は、比較するテキストを何らかの文章変更が行われた「編集点」によって部分文字列に分解し、更にこれらの部分文字列を編集前後において共通している共通部分文字列c_0，c_1，c_2，c_3と、削除された文字列α_0と、新規創作された文字列β_0，β_1とに分割することと同義である。

【００５３】

次に求めた共通部分文字列c_0，c_1，c_2，c_3の割合と出現パターンから元のテキストからの編集点数及び文脈編集距離を計算する。これらの結果から更新作業の大局を評価する。共通部分文字列集合Ｃに包含される要素数から編集点数が求まり、共通部分文字列c_0，c_1，c_2，c_3の出現順の変化から文脈編集距離が求まる。他方、非共通部分文字列α_0，β_0，β_1に対しては新規創作率の計算とN-gramによる分析を行う。これにより、編集後のテキスト中に含まれるおおむね新規創作分での更新作業の質を表す新規創作分新規度を評価する。非共通部分文字列集合Ｂに包含される要素の合計長（合計文字数）から新規創作率が求まり、非共通部分文字列集合Ａと非共通部分文字列集合Ｂに包含される要素間の非部分一致率から新規創作分新規度が求まる。最後に、共通部分文字列集合Ｃと非共通部分文字列集合Ａ，Ｂでの計算結果を統合し、文章更新量としての総合評価を計算する。

【００５４】

図３乃至図６は本文章更新量評価アルゴリズムにより実行される一連の文章更新量評価処理の流れを示すフロー図である。処理全体の流れを示す図３において、上記概説した通り、比較するテキストに対して共通部分文字列の抽出処理を行い（ステップＳ１）、抽出した共通部分文字列集合の分析を行うことにより編集点数及び文脈編集距離の評価を行い（ステップＳ２）、共通部分文字列集合を抽出した残余の非共通部分文字列集合の分析を行うことにより新規創作率及び新規創作分新規度の評価を行い（ステップＳ３）、求めた編集点数，文脈編集距離，新規創作率，及び新規創作分新規度から文章更新量の計算を行う（ステップＳ４）。

【００５５】

ステップＳ１及びステップＳ２で順次実行される共通部分文字列集合及び非共通部分文字列集合の作成方法について、図４及び図５をも参照しながら詳しく説明する。ステップＳ１の共通部分文字列抽出処理では、比較するテキストに共通に含まれる部分文字列のうち最大の長さを持つ最長共通部分文字列（以下、ＬＣＳと言う）を探索し、得られたＬＣ

Ｓをテキストから削除した残余文字列から再びＬＣＳを探索する、という手順をＬＣＳの長さが閾値ｔ以下になるまで繰り返すことにより、共通部分文字列集合を求める。ｔは何文字以上の一致をもって「共通文字列」と判定するのかを決定する閾値であり、正の整数である。まず、比較するテキストをそれぞれｘ，ｙとし、x_0，y_0に初期値として比較するテキストｘ，ｙを与える（ステップＳ10，Ｓ11）。次に共通する最長共通部分文字列c_0を求め、最長共通部分文字列の文字数を示す$|c_0|$がｔ以上であればc_0をx_0，y_0から削除し、削除後に残った文字列をx_1，y_1とする（ステップＳ12～Ｓ14）。この時、c_0がx_0，y_0の中に複数回現れる場合には、先頭から検索して最初に一致した部分文字列を削除する。以後同様にして、$|c_n|$が閾値ｔ未満になるまでc_n，x_n，y_nを求める。

【００５６】

続いて、ステップＳ２の共通部分文字列集合の分析では、ステップ１で求められた部分文字列の集合$\{c_0, c_1, c_2, \cdots, c_n\}$をｘ，ｙ間の共通部分文字列集合Ｃと定義する（ステップＳ20）。テキストｘ，ｙから閾値ｔ以上の全ての共通部分文字列c_nを取り除いた残余テキストは多数の文字列断片からなり、ｘの残余テキストを$\{\alpha_0, \alpha_1, \cdots, \alpha_n\}$、ｙの残余テキストを$\{\beta_0, \beta_1, \cdots, \beta_m\}$と表す（ステップＳ21）。$\{\alpha_n\}$，$\{\beta_m\}$の両集合の要素間には閾値ｔ以上の長さを持つ共通文字列は存在しないので、これをテキストｘ，ｙ間の非共通部分文字列集合と呼び、それぞれＡ，Ｂで表す（ステップＳ22）。以上により、テキストｘ，ｙ間の共通部分文字列，非共通部分文字列の集合Ｃ，Ａ，Ｂが求められた。

【００５７】

図７に共通部分文字列が２つ存在する場合のx_1，y_1の作成例を示す。また図８に図７の作成例により生成される集合Ｃ，Ａ，Ｂを示す。なお、両図では理解を容易とするために最初から共通部分文字列と非共通部分文字列の区別がなされているが、実際には閾値ｔを満たす全ての共通部分文字列の抽出が終わった時点で非共通部分文字列が確定されることに注意されたい。テキストｘ，ｙの部分文字列への分割については様々な分割方法があるが、上記のようにして求めた共通部分文字列集合Ｃ，非共通部分文字列集合Ａ，Ｂの要素に分割した場合の部分文字列集合をＸ，Ｙと表記する。このとき、Ｘ，ＹはＸ＝Ｃ∪Ａ、Ｙ＝Ｃ∪Ｂと表される。テキストｘは部分文字列集合Ｘ、テキストｙは部分文字列集合Ｙの全要素の連結によって構成される。図８の例におけるｘ，ｙはα，β，ｃを用いて、$x = \alpha_0 c_1 c_0 \alpha_1$、$y = \beta_0 c_0 \beta_1 c_1 \beta_2$、と表すことができる。なお、この際、αとβは添え字の順で現れるが、c_nは必ずしも添え字の順に現れるわけではないことに留意されたい。

【００５８】

図５のステップＳ23で実行される編集点数の評価について説明する。編集点とは編集が行われた箇所を表すものである。ここでは共通部分文字列を基準に文章更新を評価するため、共通部分文字列でない部分は何らかの編集が行われていると捉える。よって、編集点の総数は共通部分文字列集合Ｃの要素数$|C|$によって決定され、編集点数をＥＰと表記すると、ＥＰ＝$|C|$－１、と表すことが出来る。同一の文章同士の比較であれば共通部分文字列集合Ｃの要素数$|C|$は１であるので編集点数ＥＰ＝１－１＝０となる。新規創作・削除・置換といった編集作業が行われると元の文章から文字列が分断されるために共通部分文字列の数が増加し、結果として編集点の数が増える。また、閾値ｔの働きによりｔ以下の間隔で連続的に発生する編集作業に対しては１つの非共通部分文字列として事前に処理されるため、人間が編集後の文章を見た時に感じる「文章内の大よそ何箇所に手が加わっている」という感覚に近い評価が編集点には期待出来る。

【００５９】

図５のステップＳ24で実行される文脈編集距離の評価について説明する。文脈編集距離は文章全域にわたる論旨の組替え，順序変更のように文脈編集が行われた度合いを示す評価項目であり、ＣＥＤと表記する。文脈編集距離の計算に当たっては、まず、テキストｘ，ｙに含まれる共通部分文字列を先頭から出現順に取り出し、これらを順に並べる。このように作成された並び順を保持した部分文字列の並びを分割列と呼ぶことにする。図９に

分割列の作成例を示す。同図においては、テキストｘ，ｙから出現順序順を保持した状態で共通部分文字列c_nを抽出し、文字数が|Ｃ|の分割列を生成することにより、編集距離計算用の分割列となる分割列$c_0c_1c_2c_3$と分割列$c_0c_1c_3c_2$とが生成される。ここで作成された２つの分割列について編集距離（レーベンシュタイン距離）を計算する。編集距離は文字列に対して挿入・削除・置換を行い、別の文字列と一致させるために必要な最低操作数を表したものである。

【００６０】

図10に編集距離の計算方法を、図11に編集距離の計算例を示す。図10において、比較するテキストのｎ番目とｍ番目の文字が一致する場合はｒ＝０、それ以外はｒ＝１を設定し、３地点からの移動でコストが最小となるものを取得し、以上の処理を繰り返すことにより２つの分割列間の編集距離を求める。なお、S_1，S_2，S_3は各配列要素が持つコストである。本文章更新量評価アルゴリズムにおいては、共通部分文字列に対して編集距離を行っているので、その評価に関して考慮すべき編集作業は文脈編集、すなわち分割列における置換のみであり、ここでは置換のコストを１として計算している。具体的には、図11に示すように、元のテキストの分割列$c_0c_1c_2c_3$に含まれる要素数＋１＝５の大きさとなる行と、編集後の文章の分割列$c_0c_1c_3c_2$に含まれる要素数＋１＝５の大きさとなる列とを持つ配列Ｍを用意し、この配列Ｍの配列要素Ｍ〔０，０〕の値を０とした後、第０列目の配列要素Ｍ〔０，ｍ〕（０＜ｍ＜４）についてそれぞれＭ〔０，ｍ－１〕の値に置換のコスト１を加算した値を設定する一方で、第０行目の配列要素Ｍ〔ｎ，０〕（０＜ｎ＜４）についてＭ〔ｎ－１，０〕の値に置換のコスト１を加算した値を設定することにより、図11の初期設定がされた配列Ｍとなる。編集後の文章に対応する分割列$c_0c_1c_3c_2$の第ｎ番目の共通部分文字列と、元のテキストに対応する分割列$c_0c_1c_2c_3$の第ｍ番目の共通部分文字列とが等しいか否かが判定され、配列要素Ｍ〔ｎ－１，ｍ－１〕の値Ｓ１にｒを加算した値、配列要素Ｍ〔ｎ，ｍ－１〕の値S_2に置換のコスト１を加算した値、配列要素Ｍ〔ｎ－１，ｍ〕の値S_3に置換のコスト１を加算した値の中から最小値が選び出されて配列要素Ｍ〔ｎ，ｍ〕に格納される。このようにして配列要素の値をそれぞれ求めた後の配列Ｍが図11に示されており、この場合の編集距離は２になる。

【００６１】

通常、文字列の編集においては文字の新規挿入や削除が行われるため、編集作業の複雑さとは別の比較元の文字列からの文字数の差が広がるほど編集距離が大きくなる問題があるが、本文章更新量評価アルゴリズムでは計算対象を共通部分文字列に限定することで同じ文字数での編集距離の計算を可能とした。これは同じアルファベット集合と同じ文字数からなる文字列間の編集距離の計算となるので、求められたスコアは共通部分文字列間の最小置換回数を表すものとして捉えることが可能である。この数値が意味することは、分割列間での出現順の変更回数であり、文脈編集の度合いを表すものである。このように文脈編集距離は共通部分文字列の出現順の変更回数を反映したものである。そのため、文章への追加・削除により編集点の数が増加しても共通部分文字列の出現順に変更が起こらない限りは文脈編集距離の値には影響がない。

【００６２】

再度、図３に戻って、ステップＳ３の非共通部分文字列集合の分析では、非共通部分文字列集合Ａ，Ｂから新規創作量の評価と新規性の評価を行う。当該分析処理のフローを詳しく示したものが図６である。当該分析処理では開始直後に新規創作率の評価を行う（ステップＳ30）。新規創作率とはテキストの全長に対する非共通部分文字列の長さの割合であり、テキストｙを基準に考えた時の新規創作率NCP_yは数式３、

【００６３】

【数３】

$$NCP_y = \frac{\sum|\alpha_n|}{|x|}$$

テキストｘを基準に考えた時の新規創作率NCP_xは数式４によって表される。

【００６４】
【数４】

$$NCP_x = \frac{\sum|\beta_n|}{|y|}$$

この数値の最大値は１であり、１に近づくほどテキスト内で新規創作部分が占める割合が高いことを示す。
【００６５】
新規創作分新規度は非共通部分文字列集合間におけるｔ未満の長さによるN-gramの不一致率のことであり、更新によって追加された部分の新規性を表す。長さＬの文字列から作成される長さＮのN-gram集合をΓとおく。この時、N-gramの総数は数式|Γ|＝Ｌ－Ｎ＋１によって決定される。非共通部分文字列におけるN-gramを用いた分析では、まず非共通文字列集合Ａ，Ｂのα_n，β_mを並び順を保持した状態で繋ぎ、それぞれ分割列γ，δとおく（ステップＳ31）。次にγ，δに対して閾値ｔ未満の長さの一致を調べるために０＜Ｎ＜ｔの範囲でN-gram集合Γ_A，Γ_Bを作成する（ステップＳ32）。なお、閾値ｔ以上の長さでの一致については共通部分文字列集合に振り分けられているので、ここでは対象にしない。このように、γ，δから作成したN-gram集合Γ_A，Γ_Bについて共通して現れる要素を取得し、Ａ，Ｂの要素数に対する一致率を除いた割合を計算する（ステップＳ33）。以下に示す数式５はテキストｙを基準に考えた時の新規創作分新規度$ＤＯ_y$であり、
【００６６】
【数５】

$$DO_y = 1 - \frac{|\Gamma_A \cap \Gamma_B|}{|\Gamma_A|}$$

数式６はテキストｘを基準に考えた時の新規創作分新規度$ＤＯ_x$である。
【００６７】
【数６】

$$DO_x = 1 - \frac{|\Gamma_A \cap \Gamma_B|}{|\Gamma_B|}$$

これらの値は非共通部分文字列集合Ａ，Ｂ間での部分一致を除いた割合であり、Ａ，Ｂ間に表現の流用を多く含むほど小さな値をとる。なお、Γ_A，Γ_Bの各要素数については、式$|\Gamma_A|=|A|-N+1$、及び式$|\Gamma_B|=|B|-N+1$によって表現可能である。
【００６８】
図３のステップＳ４では、これまでのステップＳ２，Ｓ３で行った共通部分文字列集合Ｃと非共通部分文字列集合Ａ，Ｂについての評価結果を統合し、文章更新量を計算する。文章更新量の評価式は、ａ・ＥＰ＋ｂ・ＣＥＤ＋ＮＣＰ・ＤＯ・Ｌと表すことが出来、ＥＰは編集点数，ＣＥＤは文脈編集距離，ＮＣＰは新規創作率，ＤＯは新規創作分新規度，Ｌは更新後の文章長であり、各項の係数ａ，ｂは各評価項目により評価される編集作業を重み付けして評価するために任意に設定された係数であり、新規創作分文字数に合わせて各評価値を定量的な把握が可能な文字数へ変換する。
【００６９】
上記説明した一連の処理は、10^2文字程度の短いテキストであれば総当りで簡単にコンピュータに計算させることによっても短時間で処理することが可能である。しかし、実際にアプリケーションとしてコンピュータ上で動かす場合には10^3から10^5文字程度のテキスト間での文字列照合を行う必要があり、実現可能性と実行速度の面においても計算量を減らす工夫が必要である。その一例として、接尾辞配列を用いた共通部分文字列の探索方法とN-gramの計算方法について解説する。
【００７０】
テキスト間での共通部分文字列の探索、一般には最長一致部分の探索には様々な方法が

存在するが、ここでは接尾辞木をコンパクトにまとめた接尾辞配列と呼ばれるデータ構造を利用した探索を採用した。この方法は共通部分文字列が探索出来るのは勿論のこと、非共通部分文字列の分析の際に利用するN-gramの生成においても同じ接尾辞配列のデータ構造を利用して計算を行うため、プログラムを簡略化することが出来るというメリットもある。一般的な接尾辞配列は、同一テキスト内の部分文字列一致について利用されるが、本文章更新量評価アルゴリズムで計算が必要となる共通部分文字列は２テキスト間で最長一致する部分文字列の探索である。そこで、接尾辞木を複数の文字列へ対応させた一般化接尾辞木と呼ばれるデータ構造を利用し、共通部分文字列を求める。この方法では終端記号＄で連結した文字列に対し、接尾辞配列を生成し、高さ配列計算をすれば良いのだが、一致するパターンが別々のテキストに含まれるパターンなのか同一テキストに含まれるパターンなのかを判別する必要がある。

【００７１】

本実施例では、接尾辞から開始位置にあるテキストを判別するためにテキストグループ配列Ｇを追加のデータ構造として作成し、これに対処する。図12に共通部分文字列の計算例を示す。図12を参照しながら当該計算方法を説明する。テキストｘとテキストｙとを結合した文字列から全ての接尾辞を作成し、それを配列Ｓに格納すると共に、接尾辞をソートし、ソート前の接尾辞の位置に対応するインデックスを格納した接尾辞配列ＳＡを作成する。文字列の最後についている「＄」は終端記号であり、そこが文章の終わりを表す記号である。同じインデックスの接尾辞配列ＳＡに対応する文字列と、次のインデックスの接尾辞に対応する文字列との前方一致した長さを高さ配列Ｈに格納する。高さ配列の値の最も大きな場所を調べれば、文章内に存在する最も長い共通文字列を調べることが可能である。接尾辞配列作成時に接尾辞のインデックスの数値と連結した最初のテキスト、つまりテキストｘの文字列長との比較を行い、接尾辞の開始部分がテキストｘから始まるもの（接尾辞のインデックスの数値＜テキストｘの文字列長＝７）をテキストグループ０，ｙから始まるもの（接尾辞のインデックスの数値≧テキストｘの文字列長＝７）をテキストグループ１として配列Ｇを作成する。隣のＳＡが別のテキストグループかつ最大の高さを持つ位置が求める最長一致部分に相当する。

【００７２】

一般化接尾辞木を用いた共通部分文字列の探索では、テキスト間に存在する最長の共通部分文字列を探すことは容易であるが、本文章更新量評価アルゴリズムで必要となる２番目以降の共通部分文字列においては取得済みの共通部分文字列の存在を把握した上で探索を行わなければならない。具体例として、テキストx_0：ＡＢＣＸＸＸＸＤＥ＄、y_0：ＡＢＣＤＥＸＸＸＸ＄を考える。この場合、両テキストに含まれる最も長い共通部分文字列c_0は「ＸＸＸＸ」で明らかであるが、これを単純に削除してx_1，y_1を生成すると、テキストx_1：ＡＢＣＤＥ＄、テキストy_1：ＡＢＣＤＥ＄となる。元のテキストx_0には「ＡＢＣＤＥ＄」という部分文字列はないにも関わらず、テキストx_1とテキストy_1との間にはc_0の長さを越える共通部分文字列が生じてしまう。この問題を回避するためには、２回目以降の共通部分文字列の計算においては「Ｃ」と「Ｄ」の間に既に取得済み（削除済み）であることを表すギャップがあることを考慮する必要がある。具体的には、各テキスト内に存在する計算済みの共通部分文字列をそれぞれ別の特殊文字に置換すればよい。

【００７３】

N-gramの計算については、既に接尾辞配列と高さ配列が計算されている状態であれば、Ｎ以上の長さを持つ接尾辞の高さ配列の値を参考に、長さ３の部分文字列の出現回数を順に数え上げていけば良い。なお、非共通部分文字列集合への分析に使うN-gramは、共通部分文字列集合の計算に接尾辞配列を使った時とは違い一度作成すると後はそれを再利用して使用することが可能である。

【００７４】

ところで、文章更新量を評価するにあたり、比較する文章間ではどのような違いや変化があるのかを視覚的に把握するためのツールとしてドットマトリックスを利用する。ドットマトリックスとは配列間の類似度が高い部分を連続した対角成分として容易に視認する

ことが可能なグラフであり、これを表示部５に表示させる。なお、本実施例では接尾辞配列を用いた方法で共通部分文字列の探索する手法について説明したが、同様の計算はドットマトリックスを利用しても可能である。図13にドットマトリックスの計算例を示す。ドットマトリックスは図14の典型例のように共通部分文字列の出現傾向を見る上で極めて有益である。特に、ドットの横軸での座標位置と縦軸での座標位置の差をとったoffset（オフセット）の値を簡単に確認出来る利点がある。そもそもoffsetの計算はドットマトリックスの中心からoffsetが連続で一定の値として出現する箇所が共通の文字列と判定するための手法であるが、この値は文章間での共通部分文字列の移動距離としても観察可能である。すなわち、offsetの値を、ドットマトリックス上においては共通部分文字列の移動量、つまり編集による文章形態の変化の程度を表す結果として捉えることが可能である。なお、本来のドットマトリックスでは１文字単位の一致から全てドットとして表示されるが、ここで取り扱うものでは共通部分文字列をドットとしてプロットしていることに注意されたい。これは言い換えると、点や線としてドットマトリックスに現れているものは全て共通部分文字列であり、閾値ｔの長さ未満の一致については表示されていない。

【００７５】

文章更新量評価プログラム10による評価結果として、同一筆者による推敲の結果、順次改定された改定履歴１～５に対するドットマトリックスによる表示結果の一覧を図15に示す。同図から、全テキストにおいて新規創作率が低く、更新前の文章を多く継承していることがわかる。また、ドットマトリックスの変化を見ても「改定履歴１」から「改定履歴５」まで時系列に沿って更新が行われていることがわかる。また、「改定履歴３」と「改定履歴４」の間ではほとんど更新が行われていないことも確認できる。これら改定履歴１～５の文章更新量評価結果一覧を表１に示す。

【００７６】

【表１】

更新前テキスト	更新後テキスト	更新後の文章長（Ｌ）	編集点数（ＥＰ）	文脈編集距離（ＣＥＤ）	新規創作率（ＮＣＰ）	新規創作分新規度（ＤＯ）	文章更新量
改定履歴１	改定履歴２	93088	22	2	0.046	0.957	**4866**
改定履歴２	改定履歴３	99709	82	28	0.080	0.949	**12426**
改定履歴３	改定履歴４	100215	8	0	0.006	0.955	**805**
改定履歴４	改定履歴５	103600	4	0	0.033	0.998	**3492**

表１では、共通部分文字列の判定閾値ｔ＝25、N-gramの長さＮ＝5、文章更新量の評価式中に現れる各項の係数ａ＝25，ｂ＝100として計算を行った。

【００７７】

ここで、共通部分文字列の判定閾値ｔと新規創作分新規度計算に用いるＮの最適化について説明する。N-gramという長さＮの連続文字の出現頻度を扱う言語モデルを用いた分析を行い、同一文章内で一致する部分文字列の割合（N-gram重複率）と、異なる文章間で一致する部分文字列の割合（N-gram一致率）を調べることにより、閾値ｔの最適化を図った。同一言語である限り、ランダムに取得した文章においても、短い部分文字列においては頻出の熟語表現や言い回しを含むが、一定以上の長さにおいては一致が見られなくなる。図16に示すグラフの例では日本語の場合は長さ11、英語の場合は22を超えると異なる文章間での一致がなくなる。しかしながら、更新文章間における共通部分文字列はこれより長い場合がほとんどで、今回使用した分析テキストでは数十から数万程度の長さでの一致が起こる。そこで、共通部分文字列の判定閾値として、同一言語内で偶然の一致が発生しないと考えられるｔ＝25を最適閾値と定めた。これは言語における大よそ最小と考えられるアルファベット集合を持つ英語に対応する長さであり、これは当然日本語にも対応する。また、新規創作分新規度計算に用いるＮの最適値についても閾値ｔと同様であり、短いと言語固有の表現による重複を含み、長すぎると一致が見られなくなる問題がある。そこで、N-gram重複率とN-gram一致率の差をとり、重複する部分文字列を最大で探知できるＮの

値を調べた。その結果を示したものが図17のグラフである。日本語と英語とではピークが違うが、日本語と英語がアルファベット集合の最大と最小と仮定すると、ほぼ全ての言語がこの範囲に入ると考えられるため、両者にとって最大の差となるＮ＝５を最適値と定めた。

【００７８】

本文章更新量評価アルゴリズムを応用して、ドキュメント（テキスト）間の更新履歴を復元することもできる。当該応用例について、上述の改定履歴１～５に実施した場合を例に説明する。改定履歴１～５について編集点数の評価結果をまとめた表を表２に示す。

【００７９】

【表２】

	改定履歴1	改定履歴2	改定履歴3	改定履歴4	改定履歴5
改定履歴1		22	98	106	106
改定履歴2			82	90	90
改定履歴3				8	12
改定履歴4					4
改定履歴5					

表の左側のラベルがテキストｘに対応する項目であり、表上部のラベルがテキストｙに対応する。編集点はｘ，ｙに関して対象になるため、片側のみ結果を表示している。同一筆者による推敲のケースでは時系列に沿って編集点数が増加していることが確認できることから、通常、文章更新によって編集点数が増えることがあっても、減らすには共通部分文字列と認識される文字列を閾値ｔ以下の長さになるまで削除をする編集作業を行わない限り減ることはない。そのため、編集点数の大小関係から各テキストの順序を決定して改定履歴の復元が可能であると考えられるが、実際の改定履歴の復元においては後述の文脈編集距離の評価と併せた判定を行うことが望ましい。

【００８０】

改定履歴１～５について文脈編集距離の評価結果をまとめた表を表３に示す。

【００８１】

【表３】

	改定履歴1	改定履歴2	改定履歴3	改定履歴4	改定履歴5
改定履歴1		2	30	30	30
改定履歴2			28	28	28
改定履歴3				0	0
改定履歴4					0
改定履歴5					

改定履歴３～５の間ではそれぞれ文脈編集距離が０であり、これらの編集は時系列で見た時に近い位置関係にあることがわかる。また、改定履歴１と改定履歴２から改定履歴３を見た時の文脈編集距離がそれぞれ30と28であり、改定履歴２の方が改定履歴３に近いことがわかる。なお、改定履歴１から改定履歴２を見た時の文脈編集距離は２である。

【００８２】

これらの文脈編集距離による評価と編集点数による評価から文章更新における位置関係を図示すると図18のようになる。同図においては文脈編集距離（括弧なし数字で表示）と編集点数（括弧付き数字で表示）とにより各テキストの順序を決定している。文脈編集距離だけでは改定履歴３～５の順序関係を決定することは出来ないが、改定履歴２からの編集点数を参照すると改定履歴３～５の順番で更新されている可能性が高いことがわかる。これは文脈編集距離による変化をテキストのバージョン・レベル１、編集点数による変化をバージョン・レベル２として考えると、改定履歴３～５では文脈変更として探知される

程の更新は起きていないが、いくらかの追加による改定が２回行われたものとして捉えることが出来る。以上のようにして更新履歴を復元することができる。

【００８３】

以上のように本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストｘ，ｙから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、当該共通部分文字列c_nの抽出数に相当する共通部分文字列集合Ｃの要素数|Ｃ|から１を減算して編集点数ＥＰを求めるステップとを実行させる。

【００８４】

新規創作，削除，置換といった編集作業が行われると元の文章から継承している文字列が分断されることに着目し、編集が行われていない部分で挟まれた、例えば、単字・単語等の比較的短い文字の挿入・削除，誤字訂正，「てにをは」の修正，単語の置き換えなどの細かい編集作業が行われた箇所を一纏まりとして、編集が行われた箇所である編集点の数を求めることができる。以上により、文章更新量の大局評価に関して、人間が編集後の文章を見た時に感じる「文章内の大よそ何箇所に手が加わっている」という感覚に近い評価が期待できる。

【００８５】

また本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストｘ，ｙから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、当該共通部分文字列c_nを抽出元の前記各テキストｘ，ｙにおける出現順を保持した状態で並べて分割列$c_0c_1c_2c_3$，$c_0c_1c_3c_2$をそれぞれ作成するステップと、各分割列を対比して一の分割列$c_0c_1c_2c_3$を他の分割列$c_0c_1c_3c_2$と一致させるために必要な前記共通部分文字列c_n間の最小置換回数を計算して文脈編集距離ＣＥＤを求めるステップとを実行させる。

【００８６】

例えば、文章全域にわたる論旨の組替え，順序変更などの文脈編集では、更新前後に共通する文章に対してその順序を組替える置換操作のみが行われることに着目し、計算対象を当該置換操作が行なわれる共通部分文字列c_nに限定することで、文脈編集以外の例えば文字の新規挿入や削除などの編集作業を除外して、文脈編集についての編集距離のみを計算することができる。以上により、文章更新量を文脈編集という観点から評価することができる。

【００８７】

さらに本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストｘ，ｙから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、前記テキストｘ，ｙから当該共通部分文字列c_nを取り除いた残余の文字列の合計長さ$\Sigma\alpha_n$，$\Sigma\beta_n$が当該テキストｘ，ｙの全長に対して占める割合を計算して新規創作率ＮＣＰを求めるステップとを実行させる。

【００８８】

このようにすると、更新後の文章の全長に対してどの程度新規な文章が創作・追加されているかを評価することができる。以上により、文章更新量を新規創作という観点から評価することができる。

【００８９】

また本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストｘ，ｙから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、前記各テキストｘ，ｙから当該共通部分文字列c_nを取り除いた残余の文字列α_n，β_mをそれぞれ集めて非共通部分文字列集合Ａ，Ｂを作成するステップと、数式７で表される

【００９０】

【数７】

$$DO = 1 - \frac{|\Gamma_1 \cap \Gamma_2|}{|\Gamma_1|}$$

（Γ_1，Γ_2は前記各非共通部分文字列集合から作成されるＮグラム集合、$|\Gamma_1 \cap \Gamma_2|$はＮグラム集合Γ_1及びΓ_2に共通して現れる前記閾値未満の長さによるＮグラムの要素数である共通要素数、$|\Gamma_1|$はＮグラム集合Γ_1に含まれるＮグラムの総数）
前記各非共通部分文字列集合Ａ，Ｂ間における前記閾値ｔ未満の長さによるＮグラムの不一致率である新規創作分新規度ＤＯを求める新規度評価ステップとを実行させる。

【００９１】

このようにすると、更新後の文章のうち更新によって追加された部分の新規性を評価することができる。以上により、文章更新量を、新規創作された部分についての更新前のテキストに対する新規性という観点から評価することができる。

【００９２】

さらに本実施例の文章更新量評価プログラム10では、前記共通要素数$|\Gamma_1 \cap \Gamma_2|$は、前記更新前後のテキストｘ，ｙから作成された前記各非共通部分文字列集合Ａ，Ｂにそれぞれ対応する第１の文字列と第２の文字列とを終端記号＄で連結し、当該連結後の連結文字列に対して接尾辞配列を生成して高さ配列計算をすると共に、前記接尾辞配列作成時に接尾辞のインデックスの数値と、前記連結文字列の前方となる前記第１の文字列の文字列長との比較を行い、接尾辞の開始部分が前記第１の文字列から始まるグループと、接尾辞の開始部分が前記第２の文字列から始まるグループとに分類し、隣接する接尾辞配列が別のグループかつ所定のＮ値以上の高さを持つ位置に対応する隣接接尾辞配列間で前方一致する部分文字列の出現回数を数え上げることにより求められるものであることを特徴とする。

【００９３】

このようにすると、一致するパターンが別々のテキストに含まれるパターンなのか同一テキストに含まれるパターンなのかを判別して、各非共通部分文字列集合で共通する部分文字列を探索することができる。以上により、Ｎグラム集合における共通要素数を計算する際の計算量を減らしてコンピュータ１上で動かす場合の実現可能性と実行速度を向上させることができる。

【００９４】

また本実施例の文章更新量評価プログラム10では、コンピュータ１に、各ステップにより求められた編集点数と文脈編集距離と新規創作率と新規創作分新規度とを用いて、評価式：ａ・ＥＰ＋ｂ・ＣＥＤ＋ＮＣＰ・ＤＯ・Ｌ（ＥＰは編集点数，ＣＥＤは文脈編集距離，ＮＣＰは新規創作率，ＤＯは新規創作分新規度，Ｌは更新後のテキストの全長，ａ及びｂは任意の係数）により文章更新量を算出するステップを実行させる。

【００９５】

このようにすると、例えば、ア）単字・単語等の比較的短い文字の挿入・削除，誤字訂正，「てにをは」の修正，単語の置き換え、イ）文章全域にわたる論旨の組替え，順序変更，文脈編集、ウ）新規創作・追加、という作業類型に対して、編集点数と文脈編集距離と新規創作率と新規創作分新規度という評価項目で評価することにより、文章編集・更新という知的作業を、その作業類型に応じた作業量の集積として評価することができる。以上により、文章の編集・更新過程における知的作業量全般にわたる定量的把握が可能となる。

【００９６】

さらに本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、当該共通部分文字列c_nの抽出数から１を減算して編集点数ＥＰを求めるステップと、前記編集点数ＥＰの大小関係から各テキストの更新順序を決定するステップとを実行させる。

【００９７】
このようにすると、複数のテキスト間について求められた編集点数から更新順序を決定することにより、文章更新量を評価すると共に改定履歴の復元を行うことができる。以上により、複数のテキストの中から所望のバージョンのテキストを選び出すことが出来る。
【００９８】
また本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、当該共通部分文字列c_nを抽出元の前記各テキストにおける出現順を保持した状態で並べて分割列をそれぞれ作成するステップと、各分割列を対比して一の分割列を他の分割列と一致させるために必要な前記共通部分文字列c_n間の最小置換回数を計算して文脈編集距離ＣＥＤを求めるステップと、前記文脈編集距離ＣＥＤの大小関係から各テキストの更新順序を決定するステップとを実行させる。
【００９９】
このようにすると、複数のテキスト間について求められた文脈編集距離から更新順序を決定することにより、文章更新量を評価すると共に改定履歴の復元を行うことができる。以上により、複数のテキストの中から所望のバージョンのテキストを選び出すことが出来る。
【０１００】
さらに本実施例の文章更新量評価プログラム10では、前記抽出ステップは、前記更新前後の各テキストｘ，ｙにそれぞれ対応する第１の文字列と第２の文字列とを終端記号＄で連結し、当該連結後の連結文字列に対して接尾辞配列ＳＡを生成して高さ配列Ｈの計算をすると共に、前記接尾辞配列ＳＡ作成時に接尾辞のインデックスの数値と、前記連結文字列の前方となる前記第１の文字列の文字列長との比較を行い、接尾辞の開始部分が前記第１の文字列から始まるグループと、接尾辞の開始部分が前記第２の文字列から始まるグループとに分類し、隣接する接尾辞配列が別のグループかつ最大の高さを持つ位置に対応する隣接接尾辞配列ＳＡ間で前方一致する文字列を最長共通部分文字列c_0として前記テキストｘ，ｙから抽出し、残余の文字列から前記最大の高さが前記閾値ｔ未満となるまで最長共通部分文字列$c_{1\sim n}$の抽出を繰り返すものであることを特徴とする。
【０１０１】
このようにすると、探索された最長共通部分文字列c_nが別々のテキストに含まれる文字列なのか同一テキストに含まれる文字列なのかを判別して、各テキストで共通する最長共通部分文字列を探索することができる。以上により、テキスト間での文字列照合を行う際の計算量を減らしてコンピュータ１上で動かす場合の実現可能性と実行速度を向上させることができる。
【０１０２】
また本実施例の文章更新量評価プログラム10では、前記抽出ステップは、前記共通部分文字列c_nを抽出する際にそれぞれ別の特殊文字に置換するものであることを特徴とする。
【０１０３】
このようにすると、共通部分文字列c_nを抽出した後の残余の文字列中に、所定以上の長さを有する共通部分文字列が偶発的に発生することを防止することができる。以上により、より正確に共通部分文字列を抽出することができる。
【０１０４】
さらに本実施例の文章更新量評価プログラム10では、前記抽出ステップは、前記各テキストｘ，ｙを対比して作成されたドットマトリックス上にプロットされたドットにおける前記ドットマトリックスの中心からのオフセットが、前記閾値ｔ以上の回数にわたって連続で一定の値として出現する箇所に対応する文字列を共通部分文字列c_nとして前記テキストｘ，ｙから抽出するものであることを特徴とする。
【０１０５】
このようにすると、ドットマトリックスを利用して共通部分文字列c_nの探索を行うこ

とができる。以上により、共通部分文字列c_nの探索を容易に行うことができる。

【０１０６】

また本実施例の文章更新量評価プログラム10では、コンピュータ１に、比較する更新前後のテキストｘ，ｙから任意の閾値ｔ以上の長さを有する共通部分文字列c_nを抽出する抽出ステップと、当該共通部分文字列c_nを抽出元の前記各テキストｘ，ｙにおける出現順を保持した状態で並べて分割列をそれぞれ作成するステップと、前記各テキストｘ，ｙから作成された前記各分割列を対比して、前記各分割列に含まれる前記共通部分文字列c_n単位の一致における共通部分文字列c_nを構成する各文字をドットとしてプロットすることにより作成されたドットマトリックスを表示させる。

【０１０７】

このようにすると、文章更新量を評価するにあたり、文字列を構成する各文字ではなく、共通部分文字列c_nをドットの要素としたドットマトリックスを作成することにより、更新前後のテキストｘ，ｙ間に共通する文章に行われた編集作業による文章形態の変化の程度のみがドットとしてプロットされ、文章全体の編集作業の傾向を視覚的に表示させることができる。以上により、文章更新量の視覚的把握が可能となる。

【０１０８】

なお、本発明は、上記実施例に限定されるものではなく、本発明の趣旨を逸脱しない範囲で変更可能である。本発明はあらゆる言語、データ形式で作成されたテキストに適用可能であることは言うまでもない。テキストから共通部分文字列を抽出する手段や各非共通部分文字列集合間における共通要素を取得する手段についても特に限定されるものではない。

【図面の簡単な説明】

【０１０９】

【図１】本発明における文章更新量評価プログラムをインストールしたコンピュータの基本構成を示すブロック図である。

【図２】各部分文字列集合と評価項目の関係を示す説明図である。

【図３】本発明における文章更新量評価プログラムにより行われる文章更新量評価処理の流れ全体を示すフロー図である。

【図４】同上、文章更新量評価処理の一部を成す共通部分文字列抽出処理の流れを示すフロー図である。

【図５】同上、文章更新量評価処理の一部を成す共通部分文字列集合の分析処理の流れを示すフロー図である。

【図６】同上、文章更新量評価処理の一部を成す非共通部分文字列集合の分析処理の流れを示すフロー図である。

【図７】共通文字列が２つ存在する場合のテキストx_1，y_1の作成例を示す説明図である。

【図８】図７の条件下で生成される集合Ｃ，Ａ，Ｂの例を示す説明図である。

【図９】分割列の作成例を示す説明図である。

【図１０】編集距離の計算方法を示す説明図である。

【図１１】編集距離の計算例を示す説明図である。

【図１２】共通部分文字列の計算例を示す説明図である。

【図１３】ドットマトリックスの計算例を示す説明図である。

【図１４】ドットマトリックスに見る文章更新の典型例を示す説明図である。

【図１５】「同一筆者による推敲」のドットマトリックス一覧を示す説明図である。

【図１６】N-gram重複率と主題の類似しない同一言語のテキストによるN-gram一致率を示すグラフである。

【図１７】N-gram重複率と主題の類似しない同一言語のテキストによるN-gram一致率の差を示すグラフである。

【図１８】「同一筆者による推敲」での文脈編集距離と編集点数による位置関係を示す説明図である。

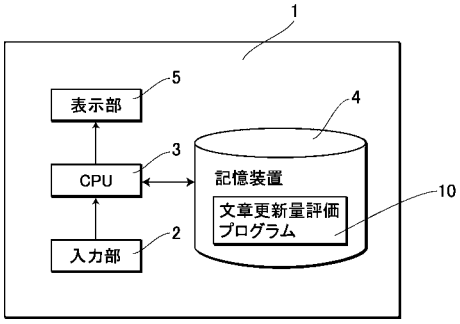
【符号の説明】

【０１１０】

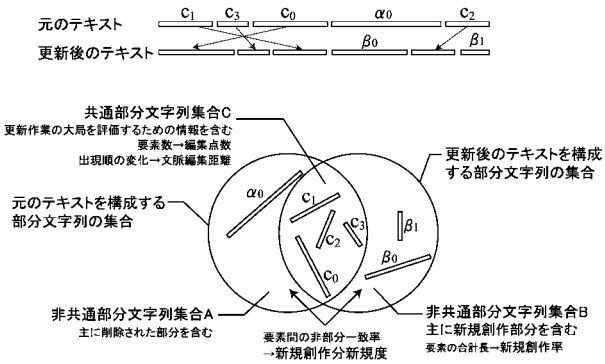
1 コンピュータ

10 文章更新量評価プログラム

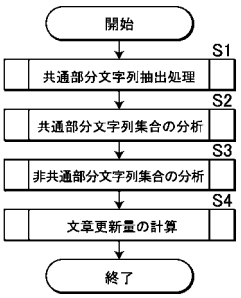
【図１】



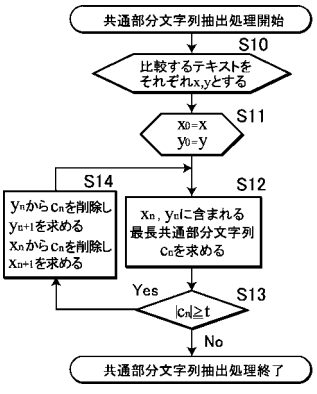
【図２】



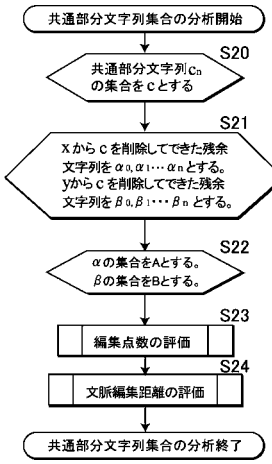
【図３】



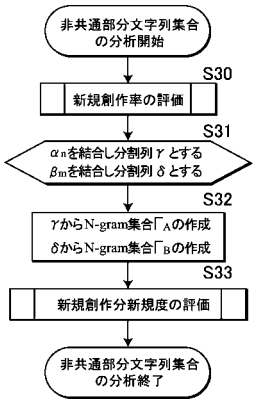
【図４】



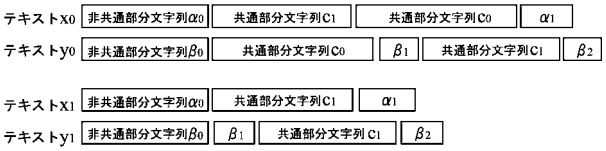
【図５】



【図６】

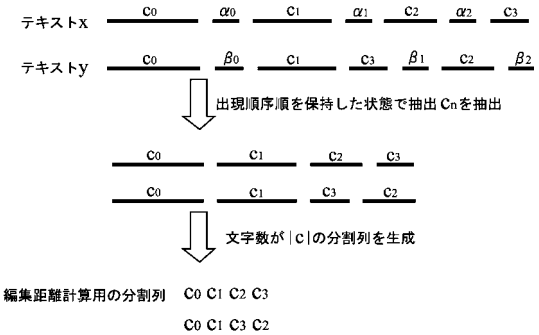


【図７】

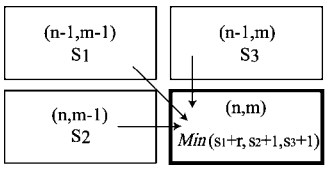


【図８】

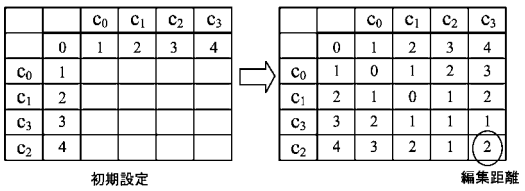
【図９】



【図１０】



【図１１】



		c0	c1	c2	c3
	0	1	2	3	4
c0	1				
c1	2				
c3	3				
c2	4				

		c0	c1	c2	c3
	0	1	2	3	4
c0	1	0	1	2	3
c1	2	1	0	1	2
c3	3	2	1	1	1
c2	4	3	2	1	2

【図１２】

テキスト x: ABCABC$

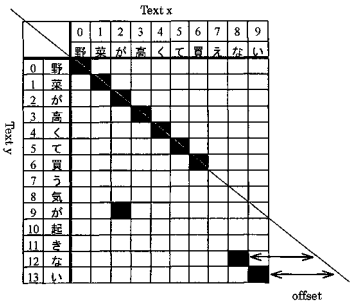
テキスト y: BCZZ$

S[0]:	ABCABC$BCZZ$	SA[0]:	$	H[0]:	1
S[1]:	BCABC$BCZZ$	SA[1]:	$BCZZ$	H[1]:	0
S[2]:	CABC$BCZZ$	SA[2]:	ABC$BCZZ$	H[2]:	3
S[3]:	ABC$BCZZ$	SA[3]:	ABCABC$BCZZ$	H[3]:	3
S[4]:	BC$BCZZ$	SA[4]:	BC$BCZZ$	H[4]:	2
S[5]:	C$BCZZ$	SA[5]:	BCABC$BCZZ$	H[5]:	2
S[6]:	$BCZZ$	SA[6]:	BCZZ$	H[6]:	0
S[7]:	BCZZ$	SA[7]:	C$BCZZ$	H[7]:	1
S[8]:	CZZ$	SA[8]:	CABC$BCZZ$	H[8]:	1
S[9]:	ZZ$	SA[9]:	CZZ$	H[9]:	0
S[10]:	Z$	SA[10]:	Z$	H[10]:	1
S[11]:	$	SA[11]:	ZZ$	H[11]:	0

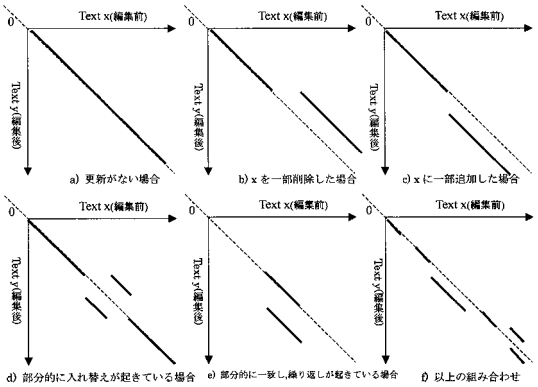
G[0]:	1
G[1]:	0
G[2]:	0
G[3]:	0
G[4]:	0
G[5]:	0
G[6]:	1
G[7]:	0
G[8]:	0
G[9]:	1
G[10]:	1
G[11]:	1

隣の SA が別のグループかつ最大の高さを持つ位置

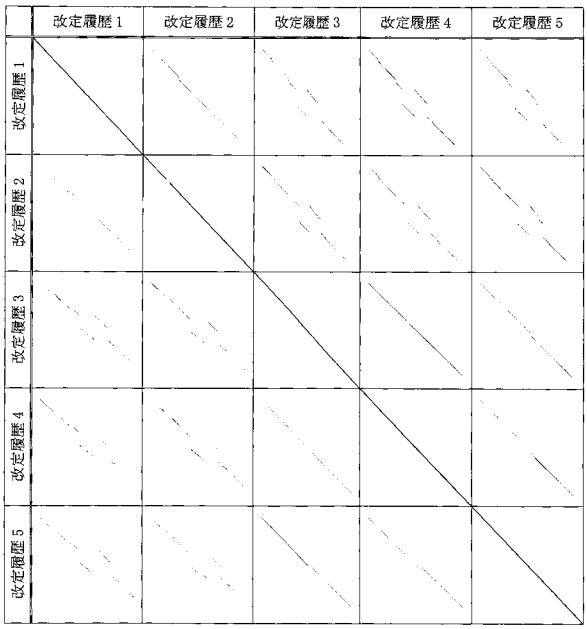
【図１３】



【図１４】

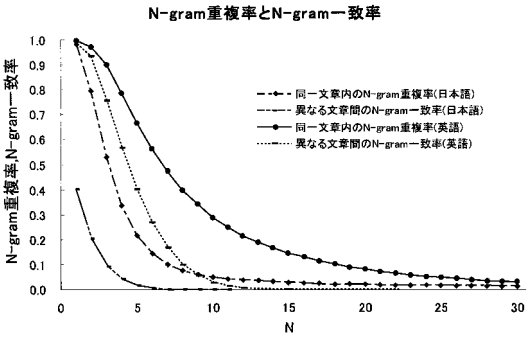


【図１５】

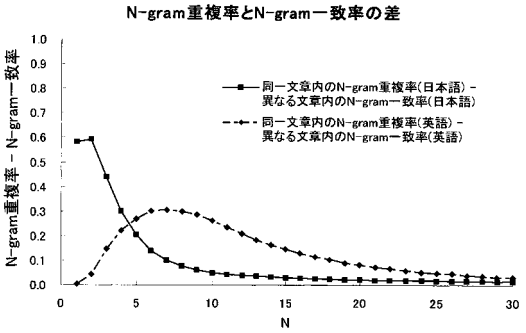


閾値 t=10，各画像の１辺は 10 万文字に相当（1pixel で 200 文字の一致）

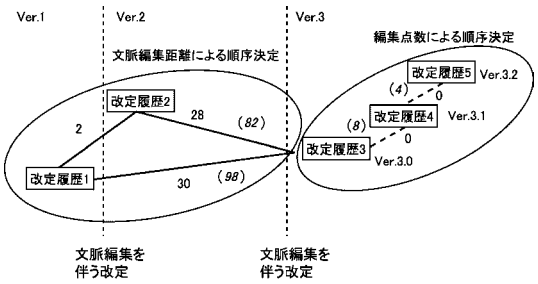
【図１６】



【図１７】



【図１８】



フロントページの続き

(72)発明者　三上　喜貴
新潟県長岡市上富岡町１６０３－１　国立大学法人長岡技術科学大学内
(72)発明者　中平　勝子
新潟県長岡市上富岡町１６０３－１　国立大学法人長岡技術科学大学内

審査官　辻本　泰隆

(56)参考文献　国際公開第２００４／０３４２８２（WO，Ａ１）
特開平０８－２９７６７５（ＪＰ，Ａ）
特開２００５－０９２７０７（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　１７／２０－１７／２６
Ｇ０６Ｆ　１７／３０