(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2025-34515
(P2025-34515A)
(43)公開日　令和7年3月13日(2025.3.13)

(51)Int.Cl.		F I		テーマコード（参考）
G16B 30/10	(2019.01)	G16B 30/10		5B056
G06F 17/16	(2006.01)	G06F 17/16	K	
G06F 7/50	(2006.01)	G06F 7/50		
G06F 7/24	(2006.01)	G06F 7/24	B	

審査請求 未請求　請求項の数 12　ＯＬ　（全 36 頁）

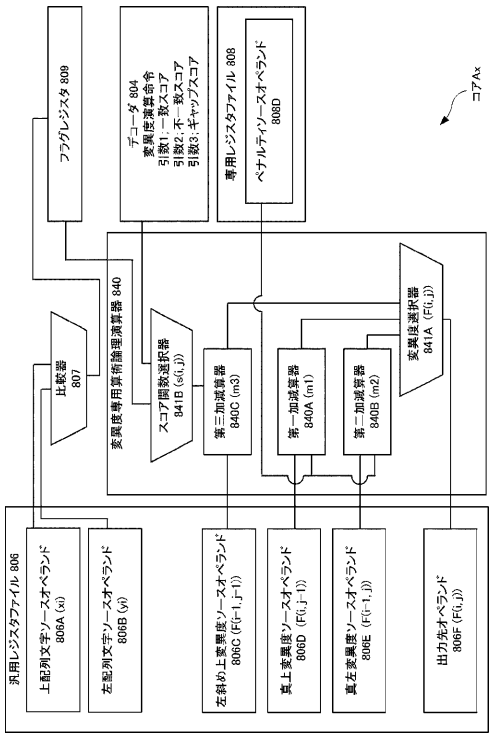
(21)出願番号　特願2023-140934(P2023-140934)
(22)出願日　令和5年8月31日(2023.8.31)

(71)出願人　720008140
先端加速システムズ株式会社
神奈川県横浜市金沢区泥亀一丁目２８番Ｂ－１３１２号
(71)出願人　501293666
株式会社ダナフォーム
神奈川県横浜市鶴見区鶴見中央２丁目６番２９号
(74)代理人　100112689
弁理士　佐原　雅史
(74)代理人　100128934
弁理士　横田　一樹

最終頁に続く

(54)【発明の名称】プロセッサ、アライメント処理用コンピュータシステム

(57)【要約】
【課題】アライメント処理に好適なプロセッサを提供する。
【解決手段】　プロセッサ（コアＡ３）は、第一ソースオペランドと、第二ソースオペランドと、第三ソースオペランドと、第四ソースオペランドと、第五ソースオペランドと、第六ソースオペランドを備える。またプロセッサ（コアＡ３）は、第一ソースオペランドの要素と第二ソースオペランドの要素の第一中間和を演算する第一加減算器と、第三ソースオペランドの要素と第四ソースオペランドの要素の第二中間和を演算する第二加減算器と、第五ソースオペランドの要素と、第六ソースオペランドの要素に基づいて得られる第六起因データの第三中間和を演算する第三加減算器とを備える。更にプロセッサ（コアＡ３）は、少なくとも第一中間和、第二中間和、及び、第三中間和の中から、最大値及び／又は最小値を選択する選択器を備える。
【選択図】図２１



【特許請求の範囲】
【請求項１】
第一ソースオペランドと、
第二ソースオペランドと、
第三ソースオペランドと、
第四ソースオペランドと、
第五ソースオペランドと、
第六ソースオペランドと、
前記第一ソースオペランドの要素と前記第二ソースオペランドの要素の第一中間和を演算する第一加減算器と、
前記第三ソースオペランドの要素と前記第四ソースオペランドの要素の第二中間和を演算する第二加減算器と、
前記第五ソースオペランドの要素と、前記第六ソースオペランドの要素に基づいて得られる第六起因データの第三中間和を演算する第三加減算器と、
少なくとも前記第一中間和、前記第二中間和、及び、前記第三中間和の中から、最大値及び／又は最小値を選択する選択器と、
を備えることを特徴とするプロセッサ。
【請求項２】
前記第一ソースオペランド、前記第三ソースオペランド、前記第五ソースオペランドは汎用レジスタファイルに格納され、
前記第二ソースオペランド、前記第四ソースオペランド、前記第六ソースオペランドは専用レジスタファイルに格納されることを特徴とする、
請求項１に記載のプロセッサ。
【請求項３】
第七ソースオペランドと、
第二選択ユニットと、を更に備え、
前記第六起因データは、少なくとも前記第六ソースオペランドの要素と第七ソースオペランドの中から前記第二選択器が選択したデータであることを特徴とする、
請求項１に記載のプロセッサ。
【請求項４】
第八ソースオペランドを更に備え、
前記選択器は、少なくとも前記第一中間和、前記第二中間和、前記第三中間和、及び、前記第八ソースオペランドの要素の中から、最大値及び／又は最小値を選択することを特徴とする、
請求項１に記載のプロセッサ。
【請求項５】
命令をデコードする命令デコーダを更に備え、
前記命令デコーダでデコードされる単一の命令により、前記第一加減算器による前記第一中間和の演算、前記第二加減算器による前記第二中間和の演算、前記第三加減算器による前記第三中間和の演算、及び、前記選択器による前記最大値及び／又は前記最小値の選択が実行されることを特徴とする、
請求項１～５のいずれか一項に記載のプロセッサ。
【請求項６】
第一文字列と第二文字列のアライメント処理で用いられるアライメント行列の現在位置を基準とした左斜め上の変異度を要素とする左斜め上変異度ソースオペランドと、
前記アライメント行列の現在位置を基準とした真左の変異度を要素とする真左変異度ソースオペランドと、
前記アライメント処理の線形ギャップペナルティ値を要素とするペナルティソースオペランドと、
前記アライメント行列の現在位置を基準とした変異度を要素とする真上変異度ソースオ

ペランドと、
前記真上変異度ソースオペランドの要素、及び、前記ペナルティソースオペランドの要素、の第一中間和を演算する第一加減算器と、
前記真左変異度ソースオペランドの要素、及び、前記ペナルティソースオペランドの要素、の第二中間和を演算する第二加減算器と、
前記斜め上変異度ソースオペランドの要素、及び、前記現在位置に対応する前記第一文字列の要素と前記第二文字列の要素の一致・不一致に基づくスコア関数値、の第三中間和を演算する第三加減算器と、
少なくとも前記第一中間和、前記第二中間和、及び、前記第三中間和の中から、最大値及び／又は最小値を選択する変異度選択器と、
を備えることを特徴とするプロセッサ。
【請求項７】
一致スコアソースオペランドと、
不一致スコアソースオペランドと、
現在位置に対応する第一文字列上の文字を要素とする第一列文字ソースオペランドと、
現在位置に対応する第二文字列上の文字を要素とする第二列文字ソースオペランドと、
前記第一列文字ソースオペランドの要素と前記第二列文字ソースオペランドの要素が一致する場合に前記一致スコアソースオペランドの要素を選択し、前記第一列文字ソースオペランドの要素と前記第二列文字ソースオペランドの要素が一致しない場合に、前記不一致スコアソースオペランドの要素を選択して前記スコア関数値として出力するスコア関数選択器と、
を備えることを特徴とする請求項６に記載のプロセッサ。
【請求項８】
スコア関数ソースオペランドを備え、
前記スコア関数値は、前記スコアソースオペランドに格納されることを特徴とする、
請求項６に記載のプロセッサ。
【請求項９】
前記左斜め上変異度ソースオペランド、前記真上変異度ソースオペランド、前記真左変異度ソースオペランドは汎用レジスタファイルに格納され、
前記ギャップソースオペランドは専用レジスタファイルに格納される、
ことを特徴とする請求項６に記載のプロセッサ。
【請求項１０】
制限定数を要素とする制限定数ソースオペランドを更に備え、
前記変異度選択器は、少なくとも、前記第一中間和、前記第二中間和、前記第三中間和、及び、前記制限定数ソースオペランドの要素の中から、最大値及び／又は最小値を選択する、
ことを特徴とする請求項６に記載のプロセッサ。
【請求項１１】
命令をデコードする命令デコーダを更に備え、
前記命令デコーダによる単一の変異度演算命令により、前記第一加減算器による前記第一中間和の演算、前記第二加減算器による前記第二中間和の演算、前記第三加減算器による前記第三中間和の演算、及び、前記変異度選択器による前記最大値及び／又は前記最小値の選択がまとめて実行されることを特徴とする、
請求項６～１０のいずれか一項に記載のプロセッサ。
【請求項１２】
第一文字列と第二文字列が保存される記憶装置、及び、プロセッサを有しており、前記第一文字列と前記第二文字列のアライメント行列を利用したアライメント処理を実行するアライメント処理用コンピュータ装置であって、
前記プロセッサは、請求項１～１０のいずれか一項に記載のものであり、
前記プロセッサは、前記記憶装置に記憶されるアライメントプログラムに基づいて、前

記アライメント行列の現在位置の変異度を算出することを特徴とする、
アライメント処理用コンピュータシステム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、アライメント処理用コンピュータシステム等に適用されるプロセッサに関する。
【背景技術】
【０００２】
ヒトゲノムは、人が持つ遺伝情報のセットであり、これを担っている物質が、約３０億対の塩基が連なったＤＮＡ（デオキシリボ核酸）である。塩基は、アデニン（Ａ）、グアニン（Ｇ）、シトシン（Ｃ）、及びチミン（Ｔ）がある。すなわち、人の遺伝情報は、これらの塩基の並び（配列）によって決定される。
【０００３】
ヒトゲノムの読み取りにはシークエンサと称される装置が用いられる。シークエンサは、サンプルとなるヒトゲノムを読み取って、これを所定の上限値（数百塩基対程度）に細断して増幅し、数億個となる遺伝情報のデータ片からなる膨大なデータ配列として出力する。現行のシークエンサは、一人分のヒトゲノムを１時間ほどで読み出すことができる。
【０００４】
シークエンサにより読み出されるばらばらのデータ片は、人の標準的なゲノム配列として定められたヒトゲノム参照配列と比較されることによって、元の長さのヒトゲノム配列に再構築され解析される。例えば、各データ片が、ヒトゲノム参照配列との比較において、どの位置にあるかが調べられ（マッピング）る。また、マッピング完了後は、どのような変異があるかといった解析がなされる。
【０００５】
なお、データ片のマッピングには、例えばＢＷＡ（Burrow-Wheeler Aligner）といったプログラムツールが用いられる。ＢＷＡは、３つのアルゴリズム、ＢＷＡ－ｂａｃｋｔｒａｃｋ、ＢＷＡ－ＳＷ、及びＢＷＡ－ＭＥＭから構成される。このうち、ＢＷＡ－ＭＥＭは、Indel（挿入欠失）に対応した高速アルゴリズムとして広く利用されている。ＢＷＡ－ＭＥＭは、読み出したデータ片に基づくクエリ文字列のうち、ヒトゲノム参照配列に繰り返し現れる部分に対して、接尾辞配列を用いてインデックスを作成し、マッピングを行うアルゴリズムである。
【０００６】
マッピングされたデータ片（照合文字列）は、ヒトゲノム参照配列における同位置の部分配列（被照合文字列）と照合され、どのような変異があるかが解析される。ヒトゲノム参照配列の部分配列とデータ片との照合には、例えば、アラインメントアルゴリズムが用いられる。アラインメントアルゴリズムでは、動的計画法に従って、アラインメント表と称される配列の各要素について変異度（類似度）を算出しながら、近似文字列を同定する。アラインメントアルゴリズムは、例えば、Ｓｍｉｔｈ－Ｗａｔｅｒｍａｎアルゴリズム、Ｓｍｉｔｈ－Ｗａｔｅｒｍａｎ－Ｇｏｔｏｈアルゴリズム、Ｎｅｅｄｌｅｍａｎ－Ｗｕｎｓｃｈアルゴリズム等が存在する（例えば、特許文献１参照）。
【先行技術文献】
【特許文献】
【０００７】
【特許文献１】米国特許第１１５５０５８４号
【発明の概要】
【発明が解決しようとする課題】
【０００８】
上述したアラインメントアルゴリズムが事項される計算機のプロセッサでは、アラインメント表（アライメント配列）の各要素（セル）の変異度（類似度）を計算する際に、複

数の命令が実行されるため、その分だけ処理速度が低下する。更に、複数の命令が実行される際、外部メモリへのアクセスが必要となり、プロセッサの処理速度がメモリへのアクセス時間により律速されてしまう。

【０００９】

本発明の一つの目的は、アライメント処理に好適なプロセッサを提供することである。また、本発明の一つの目的は、例えば、アライメント処理を高速及び／又は効率的に行う技術を提供することである。

【課題を解決するための手段】

【００１０】

上記目的に関連する本発明は、第一ソースオペランドと、第二ソースオペランドと、第三ソースオペランドと、第四ソースオペランドと、第五ソースオペランドと、第六ソースオペランドと、前記第一ソースオペランドの要素と前記第二ソースオペランドの要素の第一中間和を演算する第一加減算器と、前記第三ソースオペランドの要素と前記第四ソースオペランドの要素の第二中間和を演算する第二加減算器と、前記第五ソースオペランドの要素と、前記第六ソースオペランドの要素に基づいて得られる第六起因データの第三中間和を演算する第三加減算器と、少なくとも前記第一中間和、前記第二中間和、及び、前記第三中間和の中から、最大値及び／又は最小値を選択する選択器と、を備えることを特徴とするプロセッサである。

【００１１】

上記プロセッサは、前記第一ソースオペランド、前記第三ソースオペランド、前記第五ソースオペランドは汎用レジスタファイルに格納され、前記第二ソースオペランド、前記第四ソースオペランド、前記第六ソースオペランドは専用レジスタファイルに格納されることを特徴としてもよい。

【００１２】

上記プロセッサは、第七ソースオペランドと、第二選択ユニットと、を更に備え、前記第六起因データは、少なくとも前記第六ソースオペランドの要素と第七ソースオペランドの中から前記第二選択器が選択したデータであることを特徴としてもよい。

【００１３】

上記プロセッサは、第八ソースオペランドを更に備え、前記選択器は、少なくとも前記第一中間和、前記第二中間和、前記第三中間和、及び、前記第八ソースオペランドの要素の中から、最大値及び／又は最小値を選択することを特徴としてもよい。

【００１４】

上記プロセッサは、命令をデコードする命令デコーダを更に備え、前記命令デコーダでデコードされる単一の命令により、前記第一加減算器による前記第一中間和の演算、前記第二加減算器による前記第二中間和の演算、前記第三加減算器による前記第三中間和の演算、及び、前記選択器による前記最大値及び／又は前記最小値の選択が実行されることを特徴としてもよい。

【００１５】

上記目的に関連する本発明は、第一文字列と第二文字列のアライメント処理で用いられるアライメント行列の現在位置を基準とした左斜め上の変異度を要素とする左斜め上変異度ソースオペランドと、前記アライメント行列の現在位置を基準とした真左の変異度を要素とする真左変異度ソースオペランドと、前記アライメント処理の線形ギャップペナルティ値を要素とするペナルティソースオペランドと、前記アライメント行列の現在位置を基準とした変異度を要素とする真上変異度ソースオペランドと、前記真上変異度ソースオペランドの要素、及び、前記ペナルティソースオペランドの要素、の第一中間和を演算する第一加減算器と、前記真左変異度ソースオペランドの要素、及び、前記ペナルティソースオペランドの要素、の第二中間和を演算する第二加減算器と、前記斜め上変異度ソースオペランドの要素、及び、前記現在位置に対応する前記第一文字列の要素と前記第二文字列の要素の一致・不一致に基づくスコア関数値、の第三中間和を演算する第三加減算器と、少なくとも前記第一中間和、前記第二中間和、及び、前記第三中間和の中から、最大値及

び／又は最小値を選択する変異度選択器と、を備えることを特徴とするプロセッサである。

【００１６】

上記プロセッサは、一致スコアソースオペランドと、不一致スコアソースオペランドと、現在位置に対応する第一文字列上の文字を要素とする第一列文字ソースオペランドと、現在位置に対応する第二文字列上の文字を要素とする第二列文字ソースオペランドと、前記第一列文字ソースオペランドの要素と前記第二列文字ソースオペランドの要素が一致する場合に前記一致スコアソースオペランドの要素を選択し、前記第一列文字ソースオペランドの要素と前記第二列文字ソースオペランドの要素が一致しない場合に、前記不一致スコアソースオペランドの要素を選択して前記スコア関数値として出力するスコア関数選択器と、を備えることを特徴としてもよい。

【００１７】

上記プロセッサは、スコア関数ソースオペランドを備え、前記スコア関数値は、前記スコアソースオペランドに格納されることを特徴としてもよい。

【００１８】

上記プロセッサにおいて、前記左斜め上変異度ソースオペランド、前記真上変異度ソースオペランド、前記真左変異度ソースオペランドは汎用レジスタファイルに格納され、前記ギャップソースオペランドは専用レジスタファイルに格納される、ことを特徴としてもよい。

【００１９】

上記プロセッサは、制限定数を要素とする制限定数ソースオペランドを更に備え、前記変異度選択器は、少なくとも、前記第一中間和、前記第二中間和、前記第三中間和、及び、前記制限定数ソースオペランドの要素の中から、最大値及び／又は最小値を選択する、ことを特徴としてもよい。

【００２０】

上記プロセッサは、命令をデコードする命令デコーダを更に備え、前記命令デコーダによる単一の変異度演算命令により、前記第一加減算器による前記第一中間和の演算、前記第二加減算器による前記第二中間和の演算、前記第三加減算器による前記第三中間和の演算、及び、前記変異度選択器による前記最大値及び／又は前記最小値の選択がまとめて実行されることを特徴としてもよい。

【００２１】

上記目的に関連する本発明は、第一文字列と第二文字列が保存される記憶装置、及び、プロセッサを有しており、前記第一文字列と前記第二文字列のアライメント行列を利用したアライメント処理を実行するアライメント処理用コンピュータ装置であって、前記プロセッサは、上記のいずれかに記載のものであり、前記プロセッサは、前記記憶装置に記憶されるアライメントプログラムに基づいて、前記アライメント行列の現在位置の変異度を算出することを特徴とする、アライメント処理用コンピュータシステムである。

【発明の効果】

【００２２】

本発明によれば、アライメント処理に好適なプロセッサによって、アライメント処理を高速及び／又は効率的に行うことができる。

【図面の簡単な説明】

【００２３】

【図１】本発明の実施形態に係るコンピュータシステムの概略的構成の一例を示すブロックダイアグラムである。

【図２】同コンピュータシステムによる近似文字列照合処理の概略の一例を説明するフローチャートである。

【図３】同コンピュータシステムによるアライメント処理手順の一例を説明するフローチャートである。

【図４】同コンピュータシステムによって実現されるアライメント処理用コンピュータシ

ステムの機能ブロックの一例を説明するブロック図である。
【図５】同コンピュータシステムのアライメント処理で用いるアライメント行列のデータ構造を示す図である。
【図６】同アライメント行列の区画分割及び各区画の変異度の計算順序を示す図である。
【図７】同アライメント行列の区画分割及び各区画の変異度の計算順序の変形例を示す図である。
【図８】同アライメント行列の区画分割及び各区画の変異度の計算順序の変形例を示す図である。
【図９】同アライメント行列のデータセルにおける変異度及び帰趨経路フラグの計算概念を示す図である。
【図１０】同アライメント行列におけるバックトラックデータを示す図である。
【図１１】同アライメント行列による変異度計算の具体例を示す図である。
【図１２】同アライメント行列による変異度計算の具体例を示す図である。
【図１３】同アライメント行列によるバックトラックデータの具体例を示す図である。
【図１４】同コンピュータシステムのアライメント処理で用いるスコア行列のデータ構造を示す図である。
【図１５】同コンピュータシステムのハードウェア構成の一例を示す図である。
【図１６】同コンピュータシステムのハードウェア構成のプロセッサモジュールの一例を示すブロック図である。
【図１７】同プロセッサモジュールのコアの内部構成を示すブロック図である。
【図１８】同コアの汎用レジスタファイル、専用レジスタファイル、変異度専用算術論理演算器の内部構成を示すブロック図である。
【図１９】同コアの内部構成の変形例を示すブロック図である。
【図２０】同コアの内部構成の変形例を示すブロック図である。
【図２１】同コアの内部構成の変形例を示すブロック図である。
【図２２】同コアの内部構成の変形例を示すブロック図である。
【図２３】同コアの内部構成の変形例を示すブロック図である。
【図２４】同コアの内部構成の変形例を示すブロック図である。
【図２５】同コアの内部構成の変形例を示すブロック図である。
【発明を実施するための形態】
【００２４】
以下、図面を参照して本発明の実施の形態を説明する。ただし、以下に説明する実施形態は、あくまでも例示であり、以下に明示しない種々の変形や技術の適用を排除する意図はない。本発明は、その趣旨を逸脱しない範囲で種々変形（例えば各実施形態を組み合わせる等）して実施することができる。また、以下の図面の記載において、同一又は類似の部分には同一又は類似の符号を付して表している。図面は模式的なものであり、必ずしも実際の寸法や比率等とは一致しない。図面相互間においても互いの寸法の関係や比率が異なる部分が含まれていることがある。
【００２５】
図１は、本発明の実施形態に係るコンピュータシステムの概略的構成の一例を示すブロックダイアグラムである。同図に示すように、コンピュータシステム１は、例えば、上位コンピュータ１０と、複数の下位コンピュータ２０（１）～２０（ｎ）と、データベース３０と含み構成される。上位コンピュータ１０と下位コンピュータ２０（１）～２０（ｎ）とは、所定のインタフェースを介して通信可能に接続される。本開示では、コンピュータシステム１は、上位コンピュータ１０が複数の下位コンピュータ２０（１）～２０（ｎ）を統括的に制御する中央集権型コンピュータシステムとして構成されるが、これに限られず、例えば分散型コンピュータシステムとして構成されても良い。分散型コンピュータシステムにおいては、各コンピュータが並列分散処理により協調的に動作し得るが、特定の処理に関しては、代表する一のコンピュータのみが該処理を実行する場合があっても良い。

【００２６】
上位コンピュータ１０及び下位コンピュータ２０で共通する計算ハードウェアは、図１５に例示されるコンピューティングデバイス１００となる。コンピューティングデバイス１００は、ＤＲＡＭ等の主記憶装置となるメモリモジュール１０２、中央計算機となるプロセッサモジュール１０４、プロセッサモジュール１０４と外部機器の間のブリッジとなるチップセット１０６、ＨＤＤやＳＤＤ等の内部ストレージデバイス１０８、Ｉ／Ｏコントローラ１１０、出力インタフェース１１２、入出力インタフェース１１４、通信インタフェース１１６、外付けＨＤＤ等の外部ストレージデバイス１１８を有する。
【００２７】
更に、図１６に示すように、プロセッサモジュール１０４は、互いに独立した計算処理を実行する複数のコアＡ，Ｂ，Ｃ，Ｄと、各コアＡ，Ｂ，Ｃ，Ｄに対応して設けられる内部記憶機能の第一キャッシュメモリ（Ｌ１キャッシュメモリ）Ｌ１ａ，Ｌ１ｂ，Ｌ１ｃ，Ｌ１ｄ及び第二キャッシュメモリ（Ｌ２キャッシュメモリ）Ｌ２ａ，Ｌ２ｂ，Ｌ２ｃ，Ｌ２ｄと、全コアＡ，Ｂ，Ｃ，Ｄで共有される内部記憶機能の第三キャッシュメモリ（Ｌ３キャッシュメモリ）Ｌ３ｎを有する。コアＡ，Ｂ，Ｃ，Ｄによるアクセス待ち時間は、Ｌ１キャッシュメモリＬ１ａ，Ｌ１ｂ，Ｌ１ｃ，Ｌ１ｄが最も短く、次いでＬ２キャッシュメモリＬ２ａ，Ｌ２ｂ，Ｌ２ｃ，Ｌ２ｄが短く、最も長いのがＬ３キャッシュメモリＬ３ｎとなる。一方で、Ｌ１，Ｌ２，Ｌ３キャッシュメモリのアクセス待ち時間は、メモリモジュール（メインメモリ）１０２よりも大幅に短い。これらの構造により、プロセッサモジュール１０４内の複数のコアＡ，Ｂ，Ｃ，Ｄ同士では、並列的にタスクを処理することができる。例えば、複数のコアＡ，Ｂ，Ｃ，Ｄのそれぞれは、与えられた個々のクエリ文字列に基づいて参照文字列との比較において解析処理を行うことができる。なお、ここではキャッシュメモリが、３段階に分割されている場合を例示しているが、本発明はこれに限定されず、１段階または２段階であってもよく、４段階以上であってもよい。以降において、第一キャッシュメモリ（Ｌ１キャッシュメモリ）Ｌ１ａ，Ｌ１ｂ，Ｌ１ｃ，Ｌ１ｄ、第二キャッシュメモリ（Ｌ２キャッシュメモリ）Ｌ２ａ，Ｌ２ｂ，Ｌ２ｃ，Ｌ２ｄ、第三キャッシュメモリ（Ｌ３キャッシュメモリ）Ｌ３ｎは、これらを総称して「キャッシュメモリ」と称することができる。本発明では、プロセッサモジュール１０４におけるコア数も特に限定されない。コア数は数個から数万個まで、さまざまに選択できる。
【００２８】
更に、図１５及び図１６に示す構成は、汎用的なハード構造を例示したが、本発明はこれに限定されない。またこれらのハード構造の全部または一部は、ＦＰＧＡ等によって個別にカスタマイズ（プログラム）されたハードウェアを採用してもよい。更に、以降において、データの記憶できるハードウェアとなる、メモリモジュール１０２、内部ストレージデバイス１０８、外部ストレージデバイス１１８、第一キャッシュメモリ（Ｌ１キャッシュメモリ）Ｌ１ａ，Ｌ１ｂ，Ｌ１ｃ，Ｌ１ｄ、第二キャッシュメモリ（Ｌ２キャッシュメモリ）Ｌ２ａ，Ｌ２ｂ，Ｌ２ｃ，Ｌ２ｄ、第三キャッシュメモリ（Ｌ３キャッシュメモリ）Ｌ３ｎは、これらを総称して「記憶装置」と称することができる。
【００２９】
図１に戻って、本開示では、下位コンピュータ２０は、上位コンピュータ１０の制御の下、並列的にタスクを処理する。例えば、複数の下位コンピュータ２０のそれぞれは、与えられた個々のクエリ文字列に基づいて参照文字列との比較において解析処理を行う。以下では、下位コンピュータ２０（１）～２０（ｎ）について、それらを特に区別する必要がない限り、単に、「下位コンピュータ２０」と表記することがある。
【００３０】
データベース３０は、上位コンピュータ１０の制御の下、各種のデータ、例えば参照文字列を格納する。一例として、データベース３０は、ヒトゲノム参照配列を格納する。ヒトゲノム参照配列は、人の標準的なゲノム配列として定められた塩基対の配列を示すデータである。また、データベース３０は、参照文字列に基づいて作成されたインデックスを格納する。インデックスは、参照文字列を探索するために用いられるある種のデータ配列

構造である。なお、図中、データベース３０は、上位コンピュータ１０にのみアクセス可能に接続される構成となっているが、これに限られず、下位コンピュータ２０もまたアクセス可能に接続される構成であっても良い。また、下位コンピュータ２０も同様のデータベースを配置してもよい。ここでは、ヒトゲノム参照配列を利用する場合を例示するが、複数の生物（動物）、植物等を一まとめにしたメタゲノム参照配列を利用することもできる。

【００３１】

図２は、本発明の一実施形態に係るコンピュータシステム１による近似文字列照合処理の概略の一例を説明するフローチャートである。かかる処理は、例えば、上位コンピュータ１０及び複数の下位コンピュータ２０が、プロセッサモジュール１０４において近似文字列照合プログラムを実行することにより、他のハードウェアコンポーネントと協働し、実現される。これにより、コンピュータシステム１、または、上位コンピュータ１０、あるいは複数の下位コンピュータ２０は、図４に示す近似文字列照合用コンピュータシステム２００として機能する。また、近似文字列照合プログラムは、複数の処理ブロック（サブルーチン）を有しており、その結果として、近似文字列照合用コンピュータシステム２００は、更に詳細には、インデックス作成用コンピュータシステム２０１と、マッピング用コンピュータシステム２０２と、文字列ペア作成用コンピュータシステム２０３と、アライメント処理用コンピュータシステム２０４を内在する。

【００３２】

図２に示すように、まず、上位コンピュータ１０は、データベース３０に格納された参照文字列を読み出し、読み出した参照文字列に基づいて、インデックスを作成する（インデックス作成処理：Ｓ２０１）。具体的には、上位コンピュータ１０は、参照文字列における部分文字列を所定の条件に従って拡張し及び／又はソートして配列化することにより、所定のデータ配列構造を有するインデックスを作成する。データ配列構造は、階層的なツリー構造からなる。本開示では、このようなインデックスを階層的インデックスと称するものとする。上位コンピュータ１０は、作成した参照文字列に基づくインデックスをデータベース３０に格納する。参照文字列に基づく階層的インデックスの作成処理の詳細は後述する。なお、参照文字列に基づく階層的インデックスが既に作成されており、データベース３０に格納されている場合には、インデックス作成処理Ｓ２０１は省略され得る。これにより、コンピュータシステム１は、インデックス作成用コンピュータシステム２０１として機能する。

【００３３】

続いて、上位コンピュータ１０は、図示しない外部装置からクエリ文字列を受信し、受信したクエリ文字列に基づいて階層的インデックスを参照し、クエリ文字列の参照文字列へのマッピング処理を実行する（マッピング処理：Ｓ２０２）。一例では、外部装置は、ヒトゲノムを読み出すシークエンサであり、クエリ文字列は、シークエンサから出力される例えば１５０～２００塩基程度の塩基対のデータ片である。クエリ文字列は、一旦、データベース３０に格納されてもよい。なお、クエリ文字列の文字数は、５０以上であることが好ましく、更に好ましくは１００以上とする。一方、クエリ文字列の文字数は、１０００以下であることが好ましく、望ましくは３００以下、更に望ましくは２００以下とする。本開示では、下位コンピュータ２０もまた、上位コンピュータ１０の制御の下、割り当てられたクエリ文字列に基づいて、階層的インデックスを参照して、マッピングを行う。これにより、コンピュータシステム１は、マッピング用コンピュータシステム２０２として機能する。

【００３４】

マッピングは、クエリ文字列の少なくとも一部と一致する参照文字列における部分文字列（一致文字列）を同定する処理である。つまり、マッピングにより、参照文字列におけるクエリ文字列の少なくとも一部と一致する部分文字列の出現開始位置及び長さ（文字数）が同定される。本開示に係るマッピングでは、クエリ文字列について、階層的インデックスを検索ないしは探索することにより、参照文字列におけるクエリ文字列の出現開始位

置が同定される。なお、処理の高速化のため、作成された階層的インデックスは、メモリモジュール１０２又はプロセッサモジュール１０４内のキャッシュメモリ等の高速メモリに展開されることが好ましい。

【００３５】

なお、上記の例では、上位コンピュータ１０及び下位コンピュータ２０が、それぞれ、割り当てられたクエリ文字列に従ってマッピングを行っているが、これに限られず、例えば、上位コンピュータ１０のみが、割り当てられたクエリ文字列に基づいて、階層的インデックスを参照して、探索を行っても良い。

【００３６】

次に、上位コンピュータ１０及び下位コンピュータ２０は、それぞれ、後述する近似文字列照合のための文字列ペアを作成する（文字列ペア作成処理：Ｓ２０３）。文字列ペアは、マッピングにより同定される一致文字列を含む、参照文字列における被照合文字列とクエリ文字列における照合文字列とからなる。これにより、コンピュータシステム１は、文字列ペア作成用コンピュータシステム２０３として機能する。

【００３７】

具体的には、上位コンピュータ１０及び下位コンピュータ２０は、マッピングにより同定された一致文字列の先頭及び末尾のそれぞれに、参照文字列における対応する所定長の文字列を追加することにより、被照合文字列を作成する。つまり、被照合文字列は、参照文字列において同定された一致文字列を含む該一致文字列近傍の文字列である。例えば、参照文字列が「ＣＣＧＡＴＣＴＧＴＡＴＡＣＣＣＴＡＣＧＡ」であって、一致文字列が「ＴＡＣＣ」である場合に、例えば前後２文字ずつ追加した文字列「ＴＡＴＡＣＣＣＴ」が被照合文字列となる。ヒトゲノムの解析の場合、参照文字列について、一致文字列の先頭及び末尾に追加する塩基の長さは、それぞれ、例えば５０塩基程度であり得る。

【００３８】

また、上位コンピュータ１０は、一致文字列の末尾にクエリ文字列における対応する所定長の文字列を追加することにより、照合文字列を作成する。ヒトゲノムの解析の場合、クエリ文字列について、一致文字列の末尾に追加する文字列の長さは、例えば５０塩基程度である。

【００３９】

なお、本開示では、参照文字列について、一致文字列の先頭及び末尾に所定長の文字列が追加されるものとしたが、これに限られず、例えば、先頭又は末尾の一方にのみ所定長の文字列が追加されても良い。また、クエリ文字列について、一致文字列の先頭及び末尾に、それぞれ、所定長の文字列を追加するようにしても良いし、或いは、文字列を追加せずに、一致文字列そのものを照合文字列として扱っても良い。

【００４０】

次に、上位コンピュータ１０及び下位コンピュータ２０は、参照文字列に基づく被照合文字列とクエリ文字列に基づく照合文字列とからなる文字列ペアに基づいて少なくとも１つの近似文字列を導出する（アライメント処理：Ｓ２０４）。近似文字列の導出には、動的計画法を用いた所定のアラインメントが適用される。アラインメントは、２つの配列（文字列）をその要素どうしで置換、挿入及び欠損を許容しつつ比較して、定義されたスコア／ペナルティに従って変異度（類似度）を算出する手法である。アラインメントを実現するアルゴリズムとしては、Ｓｍｉｔｈ－ＷａｔｅｒｍａｎアルゴリズムやＮｅｅｄｌｅｍａｎ－Ｗｕｎｓｃｈアルゴリズムが知られている。また、動的計画法とは、ある段階で得られた最適解に基づいて次の段階の最適解を算出する手法である。つまり、変異度の算出では、動的計画法に従って、配列状のアラインメント表の各要素に対してスコアが算出され、最大スコアを持つ要素が決定され、これにより、少なくとも１つ以上の近似文字列が導出される。これにより、コンピュータシステム１は、アライメント処理用コンピュータシステム２０４として機能する。

【００４１】

以上のように、本実施形態の近似文字列照合処理では、参照文字列に基づく階層的イン

デックスが作成された後、与えられたクエリ文字列に従って、該階層的インデックスを探索することにより一致文字列（及びその長さ）が同定され、同定された一致文字列に基づく被照合文字列と照合文字列とからなる文字列ペアに対して近似文字列照合がなされることにより、近似文字列が導出される。換言すると、本実施形態の近似文字列照合プログラムは、参照文字列に基づくインデックス作成用プログラムと、クエリ文字列に従って階層的インデックスを探索するマッピング用プログラムと、被照合文字列と照合文字列とからなる文字列ペアを作成するプログラムと、文字列ペアを利用して近似文字列照合がなされることにより、近似文字列を導出するアライメント用プログラムを有している。なお、インデックス作成用プログラム、マッピング用プログラム、及び文字列ペアを作成するプログラムは、公知のものを採用することができるので、これ以降の詳細説明を省略する。

【００４２】

＜アライメント処理＞

まず、アライメント処理について説明する。本実施形態では、動的計画法を用いたアラインメント処理を行う。アラインメント処理とは、参照文字列における同定した出現開始位置近傍の文字列（被照合文字列）とクエリ文字列（照合文字列）の２つの配列（文字列）を、その要素どうしで置換、挿入及び欠損を許容しつつ比較し、定義されたスコア／ペナルティに従って変異度（類似度）を算出する手法である。例えば、被照合文字列Ｘ（すなわち、参照文字列に基づく被照合文字列）と文字列Ｙ（すなわち、クエリ文字列に基づく照合文字列）との比較において、照合文字列Ｙの一部の文字が置換され、挿入され、又は欠損する場合に、照合文字列Ｙは被照合文字列Ｘに一致しないと判断される。つまり、被照合文字列Ｘ及び照合文字列Ｙのそれぞれにおける各位置の文字どうしの関係は、一致する場合（一致）、一致しない場合（不一致）、及び一方の文字が存在しない場合（ギャップ）のいずれかであるといえる。本開示では、変異度の算出のために、グローバルアラインメントの一例であるＮｅｅｄｌｅｍａｎ－Ｗｕｎｓｃｈアルゴリズムを説明する。また、理解容易のため、ここでは、極めて短い８文字の被照合文字列Ｘと照合文字列Ｙとの間でのアラインメント処理を実行する場合を例示するが、実際の文字列の数は１５０～２００程度、またはそれ以上であることに留意されたい。

【００４３】

アライメント処理では、被照合文字列Ｘ及び照合文字列Ｙによって、図５に示すマトリクス状（格子状）のアライメント行列Ｖ（アラインメント表）のデータ群を生成する。表中、「φ」は空文字を意味する。ここでは、被照合文字列Ｘを、Ｘ＝ｘ１ｘ２.. ｘiと定義し、照合文字列Ｙを、Ｙ＝ｙ１ｙ２.. ｙｊと定義し、アライメント行列Ｖ内の特定の位置（ｉ，ｊ）を、データ格納セルｖ（ｉ，ｊ）と定義する。

【００４４】

＜アライメント用プロセッサ（コア）＞

次に、図１７を参照して、上位コンピュータ１０及び下位コンピュータ２０において、アライメント処理で用いる各コアＡ，Ｂ，Ｃ，Ｄの内部構成について説明する。なお、並列処理される各コアＡ，Ｂ，Ｃ，Ｄの内部構成は互いに同じであることから、ここではコアＡについて説明し、他のコアＢ，Ｃ，Ｄの説明を省略する。

【００４５】

コアＡは、プログラムカウンタ８００、デコーダ８０４、汎用レジスタファイル８０６、専用レジスタファイル８０８、実行ユニット８３０を有する。プログラムカウンタ８００は、コアＡで現在実行中の命令のアドレスを保持するレジスタとなる。コアＡは、このレジスタの値を用いて記憶装置を参照し、該当するアドレスに格納された命令を取得する。命令の実行が完了すると、実行結果に応じて、次に実行すべき命令のアドレスがプログラムカウンタにセットされる。

【００４６】

デコーダ８０４は、命令のビット列を見て、命令の種類や演算に必要なオペランドを解析する。解析結果に応じて、必要なオペランドを、汎用レジスタファイル８０６及び／又は専用レジスタファイル８０８から読み出す。またデコーダ８０４は、実行ユニット８３

０内の適切な演算器を選択し、汎用レジスタファイル８０６及び／又は専用レジスタファイル８０８から読み出された信号が、選択された演算器へ送られるように、データパスを切り替える。汎用レジスタファイル８０６及び専用レジスタファイル８０８は、演算に必要なデータを一時的に格納する領域となる。汎用レジスタファイル８０６及び専用レジスタファイル８０８は、複数のレジスタから成りたっており、１つのデータは基本的に１つのレジスタに格納される。なお、本実施形態の専用レジスタファイル８０８は、システム制御用のレジスタファイルとは異なり、演算に必要なデータの中でも、特定データのみを専用に格納するファイルであることを意味している。記憶装置のデータは、ロード命令が実行されることにより、汎用レジスタファイル８０６及び専用レジスタファイル８０８に書き込まれる。また、ストア命令によって主記憶装置に書き戻される。

【００４７】

実行ユニット８３０は、様々な演算器を内蔵しており、例えば、加減算器８３２、乗算器８３４、浮動小数点演算器８３６等を有する。特に本実施形態の実行ユニット８３０は、アライメント処理において変異度を計算する為に選択される変異度専用算術論理演算器８４０を有する。なお、実行ユニット８３０の演算結果は、指定される汎用レジスタファイル８０６及び／又は専用レジスタファイル８０８へ書き込まれる。

【００４８】

なお、これらのコアＡ，Ｂ，Ｃ，Ｄは、アライメント処理用カーネルによって制御される。アライメント処理用カーネルは、後述するアライメント用プログラムの一部に含まれる。

【００４９】

＜レジスタファイル＞

図１８に、コアＡの汎用レジスタファイル８０６、専用レジスタファイル８０８の詳細構成を示す。汎用レジスタファイル８０６は、上配列文字ソースオペランド８０６Ａ、左配列文字ソースオペランド８０６Ｂ、左斜め上変異度ソースオペランド８０６Ｃ、真上変異度ソースオペランド８０６Ｄ、真左変異度ソースオペランド８０６Ｅ、出力先オペランド８０６Ｆを格納する。上配列文字ソースオペランド８０６Ａは、データ格納セルｖ（ｉ，ｊ）に対応する被照合文字列ｘｉが取り込まれる。左配列文字ソースオペランド８０６Ｂは、データ格納セルｖ（ｉ，ｊ）に対応する照合文字列ｙｊが取り込まれる。左斜め上変異度ソースオペランド８０６Ｃは、データ格納セルｖ（ｉ，ｊ）の左斜め上のセルｖ（ｉ－１，ｊ－１）の変異度（スコア）Ｆ（ｉ－１，ｊ－１）が取り込まれる。真上変異度ソースオペランド８０６Ｄは、データ格納セルｖ（ｉ，ｊ）の真上のセルｖ（ｉ，ｊ－１）の変異度（スコア）が取り込まれる。真左変異度ソースオペランド８０６Ｅは、データ格納セルｖ（ｉ，ｊ）の真左のセルｖ（ｉ－１，ｊ）の変異度（スコア）が取り込まれる。出力先オペランド８０６Ｆは、後述する変異度専用算術論理演算器８４０の出力データが取り込まれる。

【００５０】

専用レジスタファイル８０８は、一致スコアソースオペランド８０８Ａ、不一致スコアソースオペランド８０８Ｂ、ギャップスコアソースオペランド８０８Ｃ、ペナルティソースオペランド８０８Ｄを格納する。一致スコアソースオペランド８０８Ａは、被照合文字列Ｘと照合文字列Ｙから選択される一対の文字ペア（ｘｊ，ｙｊ）の一致スコア（例えば「＋２」）が取り込まれる。不一致スコアソースオペランド８０８Ｂは、一対の文字ペア（ｘｊ，ｙｊ）の不一致スコア（例えば「－１」）が取り込まれる。ギャップスコアソースオペランド８０８Ｃは、一対の文字ペア（ｘｊ，ｙｊ）のギャップスコア（例えば「－２」）が取り込まれる。ペナルティソースオペランド８０８Ｄは、ギャップペナルティの値（例えば「－２」）が取り込まれる。

【００５１】

＜変異度専用算術論理演算器８４０＞

変異度専用算術論理演算器８４０は、スコア関数選択器８４１Ｂ、第一加減算器８４０Ａ、第二加減算器８４０Ｂ、第三加減算器８４０Ｃ、変異度選択器８４１Ａを有する。スコ

ア関数選択器８４１Ｂは、上配列文字ソースオペランド８０６Ａ、左配列文字ソースオペランド８０６Ｂ、一致スコアソースオペランド８０８Ａ、不一致スコアソースオペランド８０８Ｂ、ギャップスコアソースオペランド８０８Ｃの各要素を参照して、スコア関数ｓ（後述）に基づいてスコアを出力するマルチプレクサとなる。第三加減算器８４０Ｃは、左斜め上変異度ソースオペランド８０６Ｃの要素とスコア関数選択器８４１Ｂの出力要素とを加算する。第一加減算器８４０Ａは、真上変異度ソースオペランド８０６Ｄの要素とペナルティソースオペランド８０８Ｄの要素とを加算する。第二加減算器８４０Ｂは、真左変異度ソースオペランド８０６Ｅの要素とペナルティソースオペランド８０８Ｄの要素とを加算する。変異度選択器８４１Ａは、第一加減算器８４０Ａの出力要素と、第二加減算器８４０Ｂの出力要素と、第三加減算器８４０Ｃの出力要素の最大値又は最小値（ここでは最大値）を選択するマルチプレクサとなる。なお、変異度選択器８４１Ａは、詳細には、一対の要素における大きいほう及び／又は小さいほうの値を選択する個別選択器（マルチプレクサ）を、多段に接続することで、３要素以上の最大値及び／又は最小値を選択する構成を採用できる。

【００５２】

＜アライメント用プログラム＞

次に、アライメント用プログラムについて説明する。アライメント用プログラムは、動的計画法を用いたアラインメント処理を実行する。アライメント用プログラムにより、上位コンピュータ１０及び下位コンピュータ２０は、参照文字列における同定した出現開始位置近傍の文字列（被照合文字列）とクエリ文字列（照合文字列）の文字列ペアが、互いにどれくらい変異しているか（変異度）を推定する。

【００５３】

図３は、本実施形態に係るコンピュータシステムによるアライメント処理手順の一例を説明するフローチャートである。すなわち、図３は、図２に示した近似文字列照合処理におけるアライメント処理（Ｓ２０４）の詳細を示している。また、図４には、アライメント処理用コンピュータシステム２０４の機能ブロック構成が示される。アライメント処理用コンピュータシステム２０４は、アライメント行列作成手段６０２、定数設定手段６０４、充填領域選択手段６０６、充填ルート設定手段６０８、充填セル選択手段６１０、変異度・経路算出手段６１２、バックトラック処理手段６２０、近似文字列導出手段６２２を有する。

【００５４】

（アライメント行列作成手段６０２／アライメント行列作成ステップＳ６０２）

アライメント行列作成ステップＳ６０２において、アライメント行列作成手段６０２は、被照合文字列Ｘ及び照合文字列Ｙによって、図５に示す（ｍ＋１）×（ｎ＋１）のマトリクス状のアライメント行列Ｖ（アラインメント表）のデータ群を生成する。表中、「φ」は空文字を意味する。ここでは、被照合文字列Ｘを、Ｘ＝ｘ１ｘ２…ｘｉ…ｘｍと定義し、照合文字列Ｙを、Ｙ＝ｙ１ｙ２.. .ｙｊ…ｎと定義し、アライメント行列Ｖ内の特定の位置（ｉ，ｊ）を、データ格納セルｖ（ｉ，ｊ）と定義する。なお、アライメント行列作成手段６０２は、アライメント処理を実行する被照合文字列Ｘと照合文字列Ｙの文字列ペアを選択する（記憶装置の所定アドレスに格納する）処理と同義である。

【００５５】

（定数設定手段６０４／定数設定ステップＳ６０４）

定数設定ステップＳ６０４において。定数設定手段６０４は、アライメント処理で必要となる定数を設定する。具体的には、被照合文字列Ｘと照合文字列Ｙから選択される一対の文字ペア（ｘｊ，ｙｊ）の一致・不一致・ギャップのスコア、及び、セルの上下間及び左右間のギャップペナルティを設定する。本実施形態では、文字ペアが一致する場合のスコアを例えば「＋２」、不一致の場合のスコアを例えば「－１」、及びギャップの場合のスコアを例えば「－２」に設定し、ギャップペナルティを例えば「－２」に設定する。この定数設定手段６０４に基づいて、アライメント処理用カーネルは、これらの値を、専用レジスタファイル８０８の一致スコアソースオペランド８０８Ａ、不一致スコアソースオペ

ランド８０８Ｂ、ギャップスコアソースオペランド８０８Ｃ、ペナルティソースオペランド８０８Ｄに取り込む。

【００５６】

（充填領域選択手段６０６／充填領域選択ステップＳ６０６）

充填領域選択ステップＳ６０６において、充填領域選択手段６０６は、例えば図６に示すように、アライメント行列Ｖを複数（ｐ個）の区画に分割することで、複数の配列領域Ｖ１Ｖ２…Ｖｋ…Ｖｐを生成し、その配列順に沿って、順番に、計算を行う配列領域Ｖｋを選択する。なお、図６では、アライメント行列Ｖを縦方向に複数に分割することで、配列領域Ｖｋを左右方向に連続するセル群としている。分割方法は図６以外にも様々に存在し、例えば図７では、アライメント行列Ｖを横方向に複数に分割することで、配列領域Ｖｋを上下方向に連続するセル群としている。例えば図８では、アライメント行列Ｖを格子状に複数に分割することで、配列領域Ｖｋを上下及び左右方向に連続する格子状のセル群としている。複数の配列領域Ｖ１Ｖ２…Ｖｋ…Ｖｐの順番は、演算対象となるデータ格納セルｖ（ｉ，ｊ）に対して、左斜め上のセルｖ（ｉ－１，ｊ－１）、真上のセルｖ（ｉ，ｊ－１）、真左のセルｖ（ｉ－１，ｊ）の計算が必ず先に実行される順序となるように設定する。

【００５７】

具体的に配列領域Ｖｋを決定するために、被照合文字列Ｘと照合文字列Ｙの文字列ペアＸ・Ｙの中から、被照合文字列Ｘの部分文字列｛ｘｓ…ｘｉ…ｘｅ｝と、照合文字列Ｙの部分文字列｛ｙｓ…ｙｉ…ｙｅ｝との部分文字列ペアを抽出して、これを配列領域Ｖｋとして記憶装置の所定アドレスに格納する。なお、（ｘｓ，ｙｓ）は部分文字列ペアの開始文字を意味し、（ｘｅ，ｙｅ）は部分文字列ペアの終了文字を意味する。被照合文字列Ｘにおける部分文字列の選択は、左側から右側に向かって順番に行い、照合文字列Ｙの部分文字列の選択は、上側から下側に向かって順番に行う必要がある。配列領域Ｖｋのサイズはキャッシュメモリに収まるサイズが好ましい。本実施形態では、Ｌ１キャッシュメモリに収まるサイズが好ましく、望ましくはＬ２キャッシュメモリに収まるサイズが好ましく、更に好ましくはＬ３キャッシュメモリに収まるサイズとする。なお、ここではアライメント行列Ｖを複数（ｐ個）の区画に分割する場合を例示したが、本発明はこれに限定されず、充填領域選択手段６０６は、アライメント行列Ｖの範囲内から一部となる一つの配列領域を抽出して（換言すると、余分な配列を除外して）、その抽出範囲内に限って変異度を演算するようにしてもよい。

【００５８】

（充填ルート設定手段６０８／充填ルート設定ステップＳ６０８）

充填ルート設定ステップＳ６０８において、充填ルート設定手段６０８は、配列領域Ｖｋにおいて、データ格納セルｖ（ｉ，ｊ）の計算順序（計算ルート）Ｒｋを定義する。この計算順序Ｒｋの設定は、充填領域選択ステップＳ６０６配列領域Ｖｋの選択順序を加味し、例えば、図６の矢印に示すように、演算対象となるデータ格納セルｖ（ｉ，ｊ）に対して、左斜め上のセルｖ（ｉ－１，ｊ－１）、真上のセルｖ（ｉ，ｊ－１）、真左のセルｖ（ｉ－１，ｊ）の計算が必ず先に実行される順序を設定する。ここでは左から右に移動する計算順序を選択しているが、例えば図７のように、上から下に移動する計算順序Ｒｋを選択してもよく、また図８のような斜行移動する計算順序Ｒｋを選択することもできる。

【００５９】

（充填セル選択手段６１０／充填セル選択ステップＳ６１０）

充填セル選択ステップＳ６１０において、充填セル選択手段６１０は、図９に示すように、充填ルート設定手段６０８で設定された順番に沿って演算対象となるデータ格納セルｖ（ｉ，ｊ）を選択する。この充填セル選択手段６１０に基づいて、アライメント処理用カーネルは、汎用レジスタファイル８０６の上配列文字ソースオペランド８０６Ａ及び左配列文字ソースオペランド８０６Ｂに、データ格納セルｖ（ｉ，ｊ）に対応する文字セット（ｘｉ、ｙｊ）を取り込む。更に、アライメント処理用カーネルは、左斜め上変異度ソースオペランド８０６Ｃに、左斜め上のセルｖ（ｉ－１，ｊ－１）で予め演算済（記憶装置

に保存済み）の変異度（スコア）Ｆ（ｉ－１，ｊ－１）を取り込み、真上変異度ソースオペランド８０６Ｄに、真上のセルｖ（ｉ，ｊ－１）で予め演算済（記憶装置に保存済み）の変異度（スコア）Ｆ（ｉ，ｊ－１）を取り込み、真左変異度ソースオペランド８０６Ｅに、真左のセルｖ（ｉ－１，ｊ）に予め演算済（記憶装置に保存済み）の変異度（スコア）Ｆ（ｉ－１，ｊ）を取り込む。

【００６０】

（変異度・経路算出手段６１２／変異度・経路算出ステップＳ６１２）

変異度・経路算出ステップＳ６１２において、変異度・経路算出手段６１２は、演算対象となるデータ格納セルｖ（ｉ，ｊ）の変異度Ｆ（ｉ，ｊ）を算出する。具体的に、例えば、文字列Ｘ＝ｘ１ｘ２…ｘｉ…ｙｍと文字列Ｙ＝ｙ１ｙ２…ｙｊ…ｙｎとの比較において、文字ｘｉと文字ｙｊとの変異度Ｆ（ｉ，ｊ）は以下式１のように定義される。

【数１】

$$F(i,j)=max\begin{cases} m1=F(i,j-1)+d \\ m2=F(i-1,j)+d \\ m3=F(i-1,j-1)+s(i,j) \end{cases}$$

ここで、ｍａｘは与えられた式の値（ｍ１，ｍ２，ｍ３）の中から最大値を出力する関数、ｓはスコア関数（一致：ｓ＝２、不一致：ｓ＝－１、ギャップ：ｓ＝－２）、ｄはギャップによるペナルティ（ｄ＝２）である。

【００６１】

具体的に、式１の演算は、変異度・経路算出手段６１２に基づくアライメント処理用カーネルによって実行される。アライメント処理用カーネルは、変異度Ｆ（ｉ，ｊ）を演算するための単一の専用命令（変異度演算命令）を、デコーダ８０４に取り込ませる。その結果、デコーダ８０４は、変異度演算命令で必要なオペランドを、汎用レジスタファイル８０６及び／又は専用レジスタファイル８０８から読み出す。更にデコーダ８０４は、これらのオペランドの要素が、変異度専用算術論理演算器８４０に送られるようにデータパスを切り替える。

【００６２】

（変異度演算命令）

変異度専用算術論理演算器８４０では、スコア関数選択器８４１Ｂにおいて、上配列文字ソースオペランド８０６Ａの要素（文字ｘｉ）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）が一致する場合、一致スコアソースオペランド８０８Ａの値（「＋２」）を出力し、上配列文字ソースオペランド８０６Ａの要素（文字ｘｉ）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）が一致しない場合、不一致スコアソースオペランド８０８Ｂの値（「－１」）を出力し、上配列文字ソースオペランド８０６Ａの要素（文字ｘｉ）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）の少なくとも一方が欠損する場合、ギャップスコアソースオペランド８０８Ｃの値（「－２」）を出力する。この出力は、上述のスコア関数ｓ（ｉ，ｊ）の演算に相当する。

【００６３】

更に変異度専用算術論理演算器８４０では、第一加減算器８４０Ａにおいて、真上変異度ソースオペランド８０６Ｄの要素（Ｆ（ｉ，ｊ－１））とペナルティソースオペランド８０８Ｄの要素（「－２」）とを加算し、第二加減算器８４０Ｂにおいて、真左変異度ソースオペランド８０６Ｅの要素（Ｆ（ｉ－１，ｊ））とペナルティソースオペランド８０８Ｄの要素（「－２」）とを加算し、第三加減算器８４０Ｃにおいて、左斜め上変異度ソースオペランド８０６Ｃの要素（Ｆ（ｉ－１，ｊ－１））とスコア関数選択器８４１Ｂの出力要素とを加算する。これらの出力は、式１のｍ１、ｍ２、ｍ３の演算に相当する。

【００６４】

更にまた、変異度専用算術論理演算器８４０では、変異度選択器８４１Ａにおいて、第一加減算器８４０Ａの出力要素と、第二加減算器８４０Ｂの出力要素と、第三加減算器８

４０Ｃの出力要素の最大値又は最小値（ここでは最大値）を選択する。この出力は、式１のｍａｘ関数の演算、すなわち、変異度Ｆ（ｉ，ｊ）の結果データに相当する。この出力（変異度Ｆ（ｉ，ｊ））は、出力先オペランド８０６Ｆに記録される。

【００６５】

変異度・経路算出手段６１２は、変異度演算命令が終了した後に、演算対象となるデータ格納セルｖ（ｉ，ｊ）の帰趨経路フラグＢ（ｉ，ｊ）を算出する。帰趨経路フラグＢは、式１において、データ格納セルｖ（ｉ，ｊ）を、最大値として選択されたｍ１、ｍ２、ｍ３に対応する上流側のデータ格納セルに紐づける経路情報となる。例えば、最大値としてｍ１が選択された場合、データ格納セルｖ（ｉ，ｊ）と真上のセルｖ（ｉ，ｊ－１）が紐づけられる。例えば、最大値としてｍ２が選択された場合、データ格納セルｖ（ｉ，ｊ）と真左のセルｖ（ｉ－１，ｊ）が紐づけられる。例えば、最大値としてｍ３が選択された場合、データ格納セルｖ（ｉ，ｊ）と左斜め上のセルｖ（ｉ－１，ｊ－１）が紐づけられる。つまり、帰趨経路フラグＢ（ｉ，ｊ）は、真上に向かう経路、真左に向かう経路、左斜め上に向かう経路、の３種類の情報の組み合わせとなる。例えば、本実施形態における帰趨経路フラグＢ（ｉ，ｊ）として、ｍ１、ｍ２、ｍ３の順番に、最大値として選択された場合を１、選択されなかった場合を０と定義する。結果、帰趨経路フラグＢ（ｉ，ｊ）＝（[１又は０]，[１又は０]，[１又は０]）となる。

【００６６】

（残存充填セルの判定ステップＳ６１４）

変異度・経路算出ステップＳ６１２が完了すると、充填セルの判定ステップＳ６１４に進み、計算順序Ｒに沿って、未演算のデータ格納セルｖ（ｉ，ｊ）が残存するか否かを判定する。残存する場合は、充填セル選択ステップＳ６１０に戻って、充填セル選択手段６１０が次の演算対象となるデータ格納セルｖ（ｉ，ｊ）を選択する。結果、変異度・経路算出ステップＳ６１２において、次のデータ格納セルｖ（ｉ，ｊ）の変異度Ｆ（ｉ，ｊ）が出力される。一方、充填セルの判定ステップＳ６１４において、未演算のデータ格納セルｖ（ｉ，ｊ）が残存しない場合は、次いで、未処理の充填領域の判定ステップＳ６１６に進む。

【００６７】

（未処理の充填領域の判定ステップＳ６１６）

未処理の充填領域の判定ステップＳ６１６では、未処理の配列領域Ｖｎが残存するか否かを判定する。残存する場合は、充填領域選択ステップＳ６０６に戻って、充填領域選択手段６０６が、次の演算対象となる配列領域Ｖｎを選択する。その後は、上述の充填ルート設定ステップＳ６０８、充填セル選択ステップＳ６１０、変異度・経路算出ステップＳ６１２、残存充填セルの判定ステップＳ６１４が繰り返される。一方、未処理の充填領域の判定ステップＳ６１６において、未処理の配列領域Ｖｎが残存しない場合は、アラインメント行列Ｖに対する変異度Ｆの充填が完了したことになり、バックトラック処理Ｓ６２０に進む。

【００６８】

（バックトラック処理Ｓ６２０）

バックトラック処理Ｓ６２０において、バックトラック処理手段６２０は、図１０に示すように、変異度が最大となるデータ格納セルｖを起点セルｖｓとして選択し、帰趨経路フラグＢ（ｉ，ｊ）で上流側に紐づいている全てのデータ格納セルｖ（ｉ，ｊ）の中から、変異度Ｆ（ｉ，ｊ）が最も大きいものを選択していく。これにより、例えば、要素位置（ｉ，ｊ）のいずれか一方が０となるデータ格納セル（終点セルｖｅ）まで遡ることができる。起点セルｖｓの要素位置（ｉｓ，ｊｓ）から終点セルｖｅまでの要素位置（ｉｅ，ｊｅ）なでの経路となる要素位置群（要素位置配列）の情報を、ここではバックトラックデータＢＴと定義する。バックトラックデータＢＴが生成されたら、近似文字列導出処理Ｓ６２２に進む。

【００６９】

（近似文字列導出処理Ｓ６２２）

近似文字列導出処理Ｓ６２２において、近似文字列導出手段６２２は、バックトラックデータＢＴを、終点位置（ｉｅ，ｊｅ）から起点位置（ｉｓ，ｊｓ）に向かって参照することで、近似文字列を導出する。参照経路において、右斜め下への移動を示す場合は「移動先位置における被照合文字ｘｉと照合文字ｙｉの双方存在（一致又は置換）」を意味し、真右方向への移動を示す場合は「移動先位置における被照合文字ｘｉに対する照合文字ｙｉの欠損」を意味し、真下方向への移動を示す場合は「移動先位置における被照合文字ｘｉに対する照合文字ｙｉの挿入」を意味する。なお、「移動先位置における被照合文字ｘｉと照合文字ｙｉの双方存在（一致又は置換）」を意味する経路では、その移動先位置の被照合文字ｘｉと照合文字ｙｉが一致する場合と、不一致の場合がある。不一致の場合は、被照合文字ｘｉが照合文字ｙｉに「置換」されたことを意味する。

【００７０】

例えば図１０のバックトラックデータＢＴの場合、近似文字列は、以下の通りとなる。

Ｘ：（ｘ１）（ｘ２）（ｘ３）　－　（ｘ４）＜ｘ５＞（ｙ６）（ｙ７）

Ｙ：（ｙ１）（ｙ２）（ｙ３）＜ｙ４＞（ｙ５）　－　（ｙ６）（ｙ７）

なお、記号「（　）」は双方存在を示しており、そのなかでも被照合文字ｘｉと照合文字ｙｉが同じ場合は一致、異なる場合は置換となる。また、記号「＜　＞」は文字の挿入を示す。記号「－」は文字の欠損を示す。つまり、近似文字列は、被照合文字列Ｘに対する照合文字列Ｙの相関を示すデータセットとなる。

【００７１】

なお、図１０の点線に示すように、バックトラックデータＢＴが途中で分岐することで、複数のバックトラックデータＢＴ２が生成されたり、変異度が最大となるデータ格納セルｖ（起点セルｖｓ）が複数存在することで他のバックトラックデータＢＴ３が生成されたりする場合も同様である。この場合は、各バックトラックデータに対応する近似文字列が複数存在することになる。以上の全てのステップを経て、Ｎｅｅｄｌｅｍａｎ－Ｗｕｎｓｃｈアルゴリズムによる近似文字列を導出が完了する。

【００７２】

＜アライメント処理の具体例＞

次に具体例として、被照合文字列Ｘ「ＧＣＣＴＣＧＣＴ」と照合文字列Ｙ「ＧＣＣＡＴＴＧＡ」との間でのアラインメント処理を行う場合を説明する。アライメント行列作成ステップＳ６０２では、図１１のアライメント行列Ｖを生成する。アライメント行列Ｖにおいて、表中の「φ」は空文字であり、Ｆ（０，０）＝０と定義する。また、式１において、変異度Ｆ（ｉ，ｊ）を決定する引数（ｉ，ｊ）の少なくとも一方が負の値になる場合、演算の範囲外であるため、計算を省略する（出力Ｎｕｌｌとする）。アラインメント行列Ｖは、ある種の格子状のデータ構造として、記憶装置上に展開され、プロセッサの利用に供される。

【００７３】

定数設定ステップＳ６０４では、文字ペアが一致する場合のスコアを「＋２」、不一致の場合のスコアを「－１」、及びギャップの場合のスコアを「－２」に設定し、ギャップペナルティを「－２」に設定する。充填領域選択ステップＳ６０６では、最初に計算を行う配列領域Ｖ１を選択する。充填ルート設定ステップＳ６０８では、配列領域Ｖ１においてルートＲ１を定義する。

【００７４】

充填セル選択ステップＳ６１０では、最初に、演算対象となるデータ格納セルｖ（０，０）を選択するが、その変異度Ｆ（０，０）はあらかじめ０と定義されているためにその後の演算を省略し、次のデータ格納セルはｖ（１，０）を選択する。

【００７５】

変異度・経路算出ステップＳ６１において、変異度Ｆ（１，０）を演算する式１のｍａｘ関数内のｍ１，ｍ２，ｍ３は、以下となる。

ｍ１＝Ｆ（１，－１）－ｄ＝Ｎｕｌｌ

ｍ２＝Ｆ（０，０）－ｄ＝０－２＝－２

ｍ３＝Ｆ（０，－１）＋ｓ（ｘ１，ｙ０）＝Ｎｕｌｌ
結果、ｍａｘ関数はｍ２を選択し、Ｆ（１，０）＝－２となる。そして、帰趨経路フラグＢ（１，０）は以下となる。
Ｂ（１，０）＝（０，１，０）（つまり「真左」のみ）
【００７６】
変異度・経路算出ステップＳ６１２が完了すると、充填セルの判定ステップＳ６１４に進み、計算順序Ｒ１に沿って、未演算のデータ格納セルｖ（ｉ，ｊ）が残存するか否かを判定する。残存するので、充填セル選択ステップＳ６１０に戻って、充填セル選択手段６１０が次の演算対象となるデータ格納セルｖ（２，０）を選択する。次いで、変異度・経路算出ステップＳ６１２において、以下の通りＦ（２，０）及びＢ（２，０）が計算される。
Ｆ（２，０）＝Ｆ（１，０）－ｄ＝－４
Ｂ（２，０）＝（０，１，０）（つまり「真左」のみ）
【００７７】
その後、充填セルの判定ステップＳ６１４及び充填セル選択ステップＳ６１０を経て、次の演算対象となるデータ格納セルｖ（３，０）を選択し、変異度・経路算出ステップＳ６１２において、以下の通りＦ（３，０）及びＢ（３，０）が計算される。
Ｆ（３，０）＝Ｆ（２，０）－ｄ＝－４
Ｂ（３，０））＝（０，１，０）（つまり「真左」のみ）
【００７８】
これらのステップをｎ回繰り返すと、データ格納セルｖ（ｎ，０）において、以下の通りＦ（ｎ，０）及びＢ（ｎ，０）が計算される。
Ｆ（ｎ，０）＝－ｎｄ
Ｂ（ｎ，０））＝（０，１，０）（つまり「真左」のみ）
【００７９】
以上の結果、配列領域Ｖ１の変異度Ｆ及び帰趨経路フラグＢは図１１に示すようになる。充填セルの判定ステップＳ６１４において、計算順序Ｒ１に沿って、未演算のデータ格納セルｖ（ｉ，ｊ）が残存しないことになり、次いで、未処理の充填領域の判定ステップＳ６１６に進み、未処理の配列領域Ｖｋが残存するか否かを判定する。未処理の配列領域Ｖｋが残存するので、充填領域選択ステップＳ６０６に戻って、充填領域選択手段６０６が次の演算対象となる配列領域Ｖ２を選択する。
【００８０】
配列領域Ｖ２における最初のデータ格納セルｖ（０，１）における式１のｍａｘ関数内のｍ１，ｍ２，ｍ３は以下となる。
ｍ１＝Ｆ（０，０）－ｄ＝０－２＝－２
ｍ２＝Ｆ（－１，１）－ｄ＝Ｎｕｌｌ
ｍ３＝Ｆ（－１，０）＋ｓ（ｘ０，ｙ１）＝Ｎｕｌｌ
結果、ｍａｘ関数はｍ１を選択して、変異度Ｆ（０，１）＝－２となる。そして、帰趨経路フラグＢ（０，１）は以下となる。
Ｂ（１，０）＝（１，０，０）（つまり「真上」のみ）
【００８１】
次に選択される次のデータ格納セルｖ（１，１）における式１のｍａｘ関数内のｍ１，ｍ２，ｍ３は以下となる。
ｍ１＝Ｆ（１，０）－ｄ＝－２－２＝－４
ｍ２＝Ｆ（０，１）－ｄ＝－２－２＝－４
ｍ３＝Ｆ（０，０）＋ｓ（ｘ１，ｙ１）＝０＋２＝２
結果、ｍａｘ関数はｍ３を選択して、変異度Ｆ（１，１）＝２となる。そして、帰趨経路フラグＢ（１，１）は以下となる。
Ｂ（１，１）＝（０，０，１）（つまり「左斜め上」のみ）
【００８２】

次に選択されるデータ格納セルｖ（２，１）における式１のｍａｘ関数内のｍ１，ｍ２，ｍ３は以下となる。
ｍ１＝Ｆ（２，０）－ｄ＝－４－２＝－６
ｍ２＝Ｆ（１，１）－ｄ＝２－２＝０
ｍ３＝Ｆ（１，０）＋ｓ（ｘ２，ｙ１）＝－２－１＝－３
結果、ｍａｘ関数はｍ２を選択して、変異度Ｆ（２，１）＝０となる。そして、帰趨経路フラグＢ（２，１）は以下となる。
Ｂ（２，１）＝（０，１，０）（つまり「真左」のみ）

【００８３】

次に選択されるデータ格納セルｖ（３，１）における式１のｍａｘ関数内のｍ１，ｍ２，ｍ３は以下となる。
ｍ１＝Ｆ（３，０）－ｄ＝－６－２＝－８
ｍ２＝Ｆ（２，１）－ｄ＝０－２＝－２
ｍ３＝Ｆ（２，０）＋ｓ（ｘ３，ｙ１）＝－４－１＝－５
結果、ｍａｘ関数はｍ２を選択して、変異度Ｆ（３，１）＝－２となる。そして、帰趨経路フラグＢ（３，１）は以下となる。
Ｂ（３，１）＝（０，１，０）（つまり「真左」のみ）

【００８４】

その後の演算の一部説明を省略し、データ格納セルｖ（６，１）における式１のｍａｘ関数内のｍ１，ｍ２，ｍ３は以下となる。
ｍ１＝Ｆ（６，０）－ｄ＝－１２－２＝－１４
ｍ２＝Ｆ（５，１）－ｄ＝－６－２＝－８
ｍ３＝Ｆ（５，０）＋ｓ（ｘ６，ｙ１）＝－１０＋２＝－８
結果、ｍａｘ関数は、ｍ２又はｍ３を選択して、変異度Ｆ（６，１）＝－８となる。そして、帰趨経路フラグＢ（６，１）は以下となる。
Ｂ（６，１）＝（０，１，１）（つまり「真左」と「左斜め上」との組み合わせ）
この帰趨経路フラグＢ（６，１）のように、帰趨経路フラグＢは、複数経路を選択する場合もある。

【００８５】

以上の繰り返し演算を、配列領域Ｖ２で行うと図１２のようになる。同様に、すべての配列領域Ｖ１～Ｖ９まで順番に行うと、図１３のアライメント行列Ｖが完成する。未処理の充填領域の判定ステップＳ６１６では、未処理の配列領域が残存しなくなるので、バックトラック処理Ｓ６２０に進む。

【００８６】

バックトラック処理Ｓ６２０では、データ格納セルｖ（６，７）の変異度Ｆ（６，７）が最大値「７」を有しているので、これを起点セルｖｓに選択する。従って、データ格納セルｖ（６，７）からデータ格納セルｖ（０，０）まで、帰趨経路フラグＢを示す矢印で紐づいている経路の中から、変異度Ｆが最大のルートを選択する。このバックトラックを行うことにより、灰色で着色されたバックトラックデータＢＴ｛（６，７）→（５，６）→（４，５）→（３，４）→（３，３）→（２，２）→（１，１）→（０，０）｝が選択される。バックトラックデータＢＴの取得が完了すると、近似文字列導出処理Ｓ６２２に進む。

【００８７】

近似文字列導出処理Ｓ６２２では、バックトラックデータＢＴを、終点位置（０，０）から起点位置（６，７）に向かって参照して、近似文字列を導出する。ここでは（０，０）→（１，１）＝「Ｇ（一致）」、（１，１）→（２，２）＝「Ｃ（一致）」、（２，２）→（３，３）＝「Ｃ（一致）」、（３，３）→（３，４）＝「Ａ（挿入）」、（３，４）→（４，５）＝「Ｔ（一致）」、（４，５）→（５，６）＝「Ｔ（置換）」、（５，６）→（６，７）＝「Ｔ（一致）」となる。

【００８８】

これにより、以下の近似文字列が導出される。
X：「ＧＣＣ　－　Ｔ　Ｃ　Ｇ」
Ｙ：「ＧＣＣ＜Ａ＞Ｔ［Ｔ］Ｇ」
ただし、記号「＜　＞」は文字間への挿入を表し、記号「－」は欠損を示し、記号「［　］」は置換を示す。つまり、本事例では、被参照文字列Ｘが「ＧＣＣＴＣＧ」であるところ、被参照文字列Ｘの「Ｃ」と「Ｔ」の間に「Ａ」が挿入され、被参照文字列の「ＴＣＧ」におおける「Ｃ」が「Ｔ」に置換された相関を示している。

【００８９】

以上のように、Ｎｅｅｄｌｅｍａｎ－Ｗｕｎｓｃｈアルゴリズムに従って、アラインメント配列における変異度が最大値である要素位置を特定することにより、そこから近似文字列を導出することができる。

【００９０】

（第一変形例の紹介）

図１８に示すコアＡでは、変異度専用算術論理演算器８４０がスコア関数選択器８４１Ｂを備えるようにし、単一命令となる変異度演算命令において、スコア関数ｓの計算も同時に行う場合を例示したが、本発明はこれに限定されない。例えば、図１９に示すコアＡ１ように、変異度専用算術論理演算器８４０から独立した命令で実行されるスコア関数専用算術論理演算器８４２にスコア関数選択器８４１Ｂを配置して、スコア関数ｓの演算を実行させることができる。この場合、汎用レジスタファイル８０６には、スコアソースオペランド８０６Ｓを格納することで、スコア関数専用算術論理演算器８４２（スコア関数選択器８４１Ｂ）の出力データを、スコアソースオペランド８０６Ｓの取り込むことができる。変異度専用算術論理演算器８４０の第三加減算器８４０Ｃでは、スコアソースオペランド８０６Ｓの要素と左斜め上変異度ソースオペランド８０６Ｃの要素を加算すればよい。

【００９１】

更にこの図１８では、スコアソースオペランド８０６Ｓが、スコア関数専用算術論理演算器８４２の出力データを直接取り込む場合を例示しているが、本発明はこれに限定されない。アライメント用プログラムでは、スコア関数専用算術論理演算器８４２を利用して、例えば図１４に示すように、被照合文字列Ｘ及び照合文字列Ｙによる（ｍ＋１）×（ｎ＋１）のマトリクス状のスコア行列Ｓ（スコア表）のデータ群を生成し、まとめて、スコア値を算出して記憶装置に保存してもよい。この場合、スコアソースオペランド８０６Ｓは、記憶装置に保存されているスコア行列Ｓから、所望のスコア値ｓ（ｉ，ｊ）を取り込むことが好ましい。

【００９２】

（第二変形例の紹介）

図１８及び図１９に示すコアＡ、Ａ１では、変異度専用算術論理演算器８４０の変異度選択器８４１Ａが、第一加減算器８４０Ａの出力要素と、第二加減算器８４０Ｂの出力要素と、第三加減算器８４０Ｃの出力要素の最大値又は最小値（ここでは最大値）を選択する場合を例示した。これは、Ｎｅｅｄｌｅｍａｎ－Ｗｕｎｓｃｈアルゴリズム等のいわゆるグローバルアライメント処理に好適である。一方、本発明はこれに限定されず、いわゆるローカルアライメント処理に適用することができる。この場合、例えば図１９の第一変形例を更に変形させた図２０の第二変形例のコアＡ２のように、専用レジスタファイル８０８が、更に、制限定数オペランド８０８Ｅを格納するようにし、変異度選択器８４１Ａが、第一加減算器８４０Ａの出力要素と、第二加減算器８４０Ｂの出力要素と、第三加減算器８４０Ｃの出力要素と、制限定数オペランド８０８Ｅの要素の最大値又は最小値（ここでは最大値）を選択するようにしてもよい。制限定数オペランド８０８Ｅは、変異度選択器８４１Ａによって選択される変異度Ｆ（ｉ，ｊ）値が、所定値よりも小さくならないように制限するためのデータ（例えば「０」）を保持する。

【００９３】

例えば、Ｓｍｉｔｈ－Ｗａｔｅｒｍａｎアルゴリズムによるローカルアライメント処理では、文字列Ｘ＝ｘ１ｘ２…ｘi…ｙｍと文字列Ｙ＝ｙ１ｙ２…ｙｊ…ｙｎとの比較

において、文字ｘiと文字ｙｊとの変異度Ｆ（ｉ，ｊ）を、以下式２のように定義する。
【数２】

$$F(i,j)=max\begin{cases} m1=F(i,j-1)+d \\ m2=F(i-1,j)+d \\ m3=F(i-1,j-1)+s(i,j) \\ z \end{cases}$$

ここで、ｍａｘは与えられた式の値（ｍ１，ｍ２，ｍ３，ｚ）の中から最大値を出力する関数、ｓはスコア関数（一致：ｓ＝２、不一致：ｓ＝－１、ギャップ：ｓ＝－２）、ｄはギャップによるペナルティ（ｄ＝２）、ｚは、変異度が所定値よりも小さくならないように制限を加えるための制限定数（ここでは「０」）である。
【００９４】
（第三変形例の紹介）
図１８～図２０に示すコアＡ、Ａ１、Ａ２では、専用レジスタファイル８０８が、一致スコアソースオペランド８０８Ａ、不一致スコアソースオペランド８０８Ｂ、ギャップスコアソースオペランド８０８Ｃを格納し、スコア関数選択器８４１Ｂによって、これらの要素の値を選択的に出力する場合を例示したが、本発明はこれに限定されない。
【００９５】
例えば図２１に示すコアＡｘのように、デコーダ８０４で処理される変異度演算命令自体に、引数として、一致スコア、不一致スコア、ギャップスコアを格納しておくことができる。この場合、予め、汎用又は専用の比較器８０７において、上配列文字ソースオペランド８０６Ａの要素（文字ｘi）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）の一致・不一致を比較して、その一致・不一致判定結果を、フラグレジスタ８０９に記録する。スコア関数選択器８４１Ｂでは、フラグレジスタ８０９に記録されている判定結果を参照することで、上配列文字ソースオペランド８０６Ａの要素（文字ｘi）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）が一致する場合、変異度演算命令をマスクレジスタによってマスクして引数：一致スコア（「＋２」）を抽出・出力し、上配列文字ソースオペランド８０６Ａの要素（文字ｘi）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）が一致しない場合、変異度演算命令をマスクレジスタによってマスクして引数：不一致スコア（「－１」）を抽出・出力すればよい。これにより、オペランドの数を削減することができる。換言すると、デコーダ８０４等の命令専用レジスタが、その引数を保持することによって、専用レジスタファイル８０８（専用オペランド）を兼ねることができる。なお、上配列文字ソースオペランド８０６Ａの要素（文字ｘi）と左配列文字ソースオペランド８０６Ｂの要素（文字ｙｊ）の少なくとも一方が欠損するか否かの欠損判定については、上記比較器８０７に再ロードして欠損判定を行い、その欠損判定結果を、フラグレジスタ８０９に記録することで、スコア関数選択器８４１Ｂで、変異度演算命令をマスクレジスタによってマスクして引数：ギャップスコア（「－２」）を抽出・出力できる。勿論、比較器８０７やフラグレジスタ８０９を２つ以上用意しておくことで、一致・不一致判定と、欠損判定を同時に行う構成にしても良い。
【００９６】
（第四変形例：汎用化事例の紹介）
【００９７】
なお、図１８～図２１に示すコアＡ、Ａ１、Ａ２、Ａｘは、汎用レジスタファイル８０６と専用レジスタファイル８０８を備えることで、アライメント処理のカスタマイズされる事例を例示したが、本発明はこれに限定されず、全てのオペランド群を汎用レジスタファイル８０６に格納してもよい。図１８に示すコアＡを、汎用化した第三変形例に係るコアＡ３を図２２に示す。
【００９８】

コアＡ３は、汎用レジスタファイル８０６－Ｈ３、専用レジスタファイル８０８－Ｈ３、汎用算術論理演算器８４０－Ｈ３を備える。汎用レジスタファイル８０６－Ｈ３は、第九ソースオペランド８０６Ａ－Ｈ３、第十ソースオペランド８０６Ｂ－Ｈ３、第五ソースオペランド８０６Ｃ－Ｈ３、第一ソースオペランド８０６Ｄ－Ｈ３、第三ソースオペランド８０６Ｅ－Ｈ３、出力先オペランド８０６Ｆ－Ｈ３を格納する。

【００９９】

専用レジスタファイル８０８－Ｈ３は、第六ソースオペランド８０８Ａ－Ｈ３、第七ソースオペランド８０８Ｂ－Ｈ３、第八ソースオペランド８０８Ｃ－Ｈ３、第二ソースオペランド８０８Ｄ－Ｈ３ａ、第四ソースオペランド８０８Ｄ－Ｈ３ｂを格納する。なお、第二ソースオペランド８０８Ｄ－Ｈ３ａと第四ソースオペランド８０８Ｄ－Ｈ３ｂは、図１８におけるペナルティソースオペランド８０８Ｄを、２つのオペランドに分けて独立させて汎用化したものであり、第一加減算器８４０Ａ－Ｈ３と、第二加減算器８４０Ｂ－Ｈ３に、別々の要素を加算できるようにしている。

【０１００】

汎用算術論理演算器８４０－Ｈ３は、第二選択器８４１Ｂ－Ｈ３、第一加減算器８４０Ａ－Ｈ３、第二加減算器８４０Ｂ－Ｈ３、第三加減算器８４０Ｃ－Ｈ３、第一選択器８４１Ａ－Ｈ３を有する。第二選択器８４１Ｂ－Ｈ３は、第九ソースオペランド８０６Ａ－Ｈ３、第十ソースオペランド８０６Ｂ－Ｈ３、第六ソースオペランド８０８Ａ－Ｈ３、第七ソースオペランド８０８Ｂ－Ｈ３、第八ソースオペランド８０８Ｃ－Ｈ３の各要素を参照して、スコア関数ｓに基づいてスコアを出力するマルチプレクサとなる。第三加減算器８４０Ｃ－Ｈ３は、第五ソースオペランド８０６Ｃ－Ｈ３の要素と第二選択器８４１Ｂ－Ｈ３の出力要素とを加算する。第一加減算器８４０Ａ－Ｈ３は、第一ソースオペランド８０６Ｄ－Ｈ３の要素と第二ソースオペランド８０８Ｄ－Ｈ３ａの要素とを加算する。第二加減算器８４０Ｂ－Ｈ３は、第三ソースオペランド８０６Ｅ－Ｈ３の要素と第四ソースオペランド８０８Ｄ－Ｈ３ｂの要素とを加算する。第一選択器８４１Ａ－Ｈ３は、第一加減算器８４０Ａ－Ｈ３の出力要素と、第二加減算器８４０Ｂ－Ｈ３の出力要素と、第三加減算器８４０Ｃ－Ｈ３の出力要素の最大値又は最小値（ここでは最大値）を選択するマルチプレクサとなる。なお、第一選択器８４１Ａ－Ｈ３は、、例えば、これらの三要素の中の二つ（例えば第一加減算器８４０Ａ－Ｈ３の出力要素と、第二加減算器８４０Ｂ－Ｈ３の出力要素）を比較する第一比較器と、この第一比較器の出力と残りの一つ（例えば第三加減算器８４０Ｃ－Ｈ３の出力要素）を比較する第二比較器の組み合わせによって、汎用的に実現できることは言うまでもない。

【０１０１】

図２２のコアＡ３において汎用化された部品・部材に付す符号は、図１８に示すコアＡの専用部品・部材に付した符号に「－Ｈ３」を付加したものを採用している。この符号対応関係によって、図２２のコアＡ３の詳細機能の説明を省略する。

【０１０２】

以上の通り、図２１のコアＡ３は、第一ソースオペランド８０６Ｄ－Ｈ３と、第二ソースオペランド８０８Ｄ－Ｈ３ａと、第三ソースオペランド８０６Ｅ－Ｈ３と、第四ソースオペランド８０８Ｄ－Ｈ３ｂと、第五ソースオペランド８０６Ｃ－Ｈ３と、第六ソースオペランド８０８Ａ－Ｈ３と、第一ソースオペランド８０６Ｄ－Ｈ３の要素と第二ソースオペランド８０８Ｄ－Ｈ３ａの要素の第一中間和を演算する第一加減算器８４０Ａ－Ｈ３と、第三ソースオペランド８０６Ｅ－Ｈ３の要素と第四ソースオペランド８０８Ｄ－Ｈ３ｂの要素の第二中間和を演算する第二加減算器８４０Ｂ－Ｈ３と、第五ソースオペランド８０６Ｃ－Ｈ３の要素と、第六ソースオペランド８０８Ａ－Ｈ３の要素に基づいて得られる第六起因データ（ここでは第二選択器８４１Ｂ－Ｈ３の出力要素）の第三中間和を演算する第三加減算器８４０Ｃ－Ｈ３と、少なくとも第一中間和、第二中間和、及び、第三中間和の中から、最大値及び／又は最小値を選択する第一選択器８４１Ａ－Ｈ３と、を備える。

【０１０３】

このコアＡ３において単一命令で実行される機能を汎用式化すると以下式３となる。

【数３】

$$L = max \begin{cases} a + b \\ c + d \\ e + f \end{cases}$$

ここで、Ｌは第一選択器８４１Ａ－Ｈ３の出力要素、ａは第一ソースオペランド８０６Ｄ－Ｈ３の要素、ｂは第二ソースオペランド８０８Ｄ－Ｈ３ａの要素、ｃは第三ソースオペランド８０６Ｅ－Ｈ３の要素、ｄは第四ソースオペランド８０８Ｄ－Ｈ３ｂの要素、ｅは第五ソースオペランド８０６Ｃ－Ｈ３の要素、ｆは第六ソースオペランド８０８Ａ－Ｈ３の要素に基づいて得られる第六起因データとなる。

【０１０４】

図２２のコアＡ３は、図１８のコアＡと同様のアライメント処理に好適に用いることができるが、その他の汎用用途で使用することもできる。なお、ここでは汎用レジスタファイル８０６－Ｈ３と専用レジスタファイル８０８－Ｈ３に分ける場合を例示しているが、すべてのオペランドを汎用レジスタファイル８０６－Ｈ３に格納することで、更に、汎用性を高めるようにしてもよい。

【０１０５】

（第五変形例：汎用化事例の紹介）

図１９に示す第一変形例のコアＡ１を汎用化した、第五変形例に係るコアＡ４を図２３に示す。コアＡ４は、汎用レジスタファイル８０６－Ｈ４、専用レジスタファイル８０８－Ｈ４、第一汎用算術論理演算器８４０－Ｈ４、第二汎用算術論理演算器８４２－Ｈ４、を備える。汎用レジスタファイル８０６－Ｈ４は、第十ソースオペランド８０６Ａ－Ｈ４、第十一ソースオペランド８０６Ｂ－Ｈ４、第六ソースオペランド８０６Ｓ－Ｈ４、第五ソースオペランド８０６Ｃ－Ｈ４、第一ソースオペランド８０６Ｄ－Ｈ４、第三ソースオペランド８０６Ｅ－Ｈ４、出力先オペランド８０６Ｆ－Ｈ４を格納する。

【０１０６】

専用レジスタファイル８０８－Ｈ４は、第七ソースオペランド８０８Ａ－Ｈ４、第八ソースオペランド８０８Ｂ－Ｈ４、第九ソースオペランド８０８Ｃ－Ｈ４、第二ソースオペランド８０８Ｄ－Ｈ４ａ、第四ソースオペランド８０８Ｄ－Ｈ４ｂを格納する。なお、第二ソースオペランド８０８Ｄ－Ｈ４ａと第四ソースオペランド８０８Ｄ－Ｈ４ｂは、図１９におけるペナルティソースオペランド８０８Ｄを、２つのオペランドに分けて独立させて汎用化したものであり、第一加減算器８４０Ａ－Ｈ４と、第二加減算器８４０Ｂ－Ｈ４に、別々の要素を加算できるようにしている。

【０１０７】

第二汎用算術論理演算器８４２－Ｈ４は、第二選択器８４１Ｂ－Ｈ４を備える。第二選択器８４１Ｂ－Ｈ４は、第十ソースオペランド８０６Ａ－Ｈ４、第十一ソースオペランド８０６Ｂ－Ｈ４、第七ソースオペランド８０８Ａ－Ｈ４、第八ソースオペランド８０８Ｂ－Ｈ４、第九ソースオペランド８０８Ｃ－Ｈ４の各要素を参照して、スコア関数ｓに基づいてスコアを出力するマルチプレクサとなる。第二選択器８４１Ｂ－Ｈ４の出力データは、第六ソースオペランド８０６Ｓ－Ｈ４に取り込むことができる。

【０１０８】

第一汎用算術論理演算器８４０－Ｈ４は、第一加減算器８４０Ａ－Ｈ４、第二加減算器８４０Ｂ－Ｈ４、第三加減算器８４０Ｃ－Ｈ４、第一選択器８４１Ａ－Ｈ４を有する。第三加減算器８４０Ｃ－Ｈ４は、第五ソースオペランド８０６Ｃ－Ｈ４の要素と第六ソースオペランド８０６Ｓ－Ｈ４の要素とを加算する。第一加減算器８４０Ａ－Ｈ４は、第一ソースオペランド８０６Ｄ－Ｈ４の要素と第二ソースオペランド８０８Ｄ－Ｈ４ａの要素とを加算する。第二加減算器８４０Ｂ－Ｈ４は、第三ソースオペランド８０６Ｅ－Ｈ４の要素と第四ソースオペランド８０８Ｄ－Ｈ４ｂの要素とを加算する。第一選択器８４１Ａ－

Ｈ４は、第一加減算器８４０Ａ－Ｈ４の出力要素と、第二加減算器８４０Ｂ－Ｈ４の出力要素と、第三加減算器８４０Ｃ－Ｈ４の出力要素の最大値又は最小値（ここでは最大値）を選択するマルチプレクサとなる。

【０１０９】

図２３のコアＡ４において汎用化された部品・部材に付す符号は、図１９に示すコアＡ１の専用部品・部材に付した符号に「－Ｈ４」を付加したものを採用している。この符号対応関係によって、図２３のコアＡ４の詳細機能の説明を省略する。

【０１１０】

以上の通り、図２３のコアＡ４は、第一ソースオペランド８０６Ｄ－Ｈ４と、第二ソースオペランド８０８Ｄ－Ｈ４ａと、第三ソースオペランド８０６Ｅ－Ｈ４と、第四ソースオペランド８０８Ｄ－Ｈ４ｂと、第五ソースオペランド８０６Ｃ－Ｈ４と、第六ソースオペランド８０６Ｓ－Ｈ４と、第一ソースオペランド８０６Ｄ－Ｈ４の要素と第二ソースオペランド８０８Ｄ－Ｈ４ａの要素の第一中間和を演算する第一加減算器８４０Ａ－Ｈ４と、第三ソースオペランド８０６Ｅ－Ｈ４の要素と第四ソースオペランド８０８Ｄ－Ｈ４ｂの要素の第二中間和を演算する第二加減算器８４０Ｂ－Ｈ４と、第五ソースオペランド８０６Ｃ－Ｈ４の要素と第六ソースオペランド８０６Ｓ－Ｈ４の要素に基づいて得られる第六起因データ（ここでは第六ソースオペランド８０６Ｓ－Ｈ４の要素そのもの）の第三中間和を演算する第三加減算器８４０Ｃ－Ｈ４と、少なくとも第一中間和、第二中間和、及び、第三中間和の中から、最大値及び／又は最小値を選択する第一選択器８４１Ａ－Ｈ４と、を備える。

【０１１１】

このコアＡ４において単一命令で実行される機能を汎用式化すると以下式４となる。

【数４】

$$L=\max\begin{cases} a+b \\ c+d \\ e+f' \end{cases}$$

ここで、Ｌは第一選択器８４１Ａ－Ｈ４の出力要素、ａは第一ソースオペランド８０６Ｄ－Ｈ４の要素、ｂは第二ソースオペランド８０８Ｄ－Ｈ４ａの要素、ｃは第三ソースオペランド８０６Ｅ－Ｈ４の要素、ｄは第四ソースオペランド８０８Ｄ－Ｈ４ｂの要素、ｅは第五ソースオペランド８０６Ｃ－Ｈ４の要素、ｆ'は第六ソースオペランド８０６Ｓ－Ｈ４の要素となる。

【０１１２】

図２３のコアＡ４は、図１９のコアＡ１と同様のアライメント処理に好適に用いることができるが、その他の汎用用途で使用することもできる。なお、ここでは汎用レジスタファイル８０６－Ｈ４と専用レジスタファイル８０８－Ｈ４に分ける場合を例示しているが、すべてのオペランドを汎用レジスタファイル８０６－Ｈ４に格納することで、更に、汎用性を高めるようにしてもよい。

【０１１３】

（第六変形例：汎用化事例の紹介）

図２０に示す第二変形例のコアＡ２を汎用化した、第六変形例に係るコアＡ５を図２４に示す。

【０１１４】

コアＡ５は、汎用レジスタファイル８０６－Ｈ５、専用レジスタファイル８０８－Ｈ５、第一汎用算術論理演算器８４０－Ｈ５、第二汎用算術論理演算器８４２－Ｈ５を備える。汎用レジスタファイル８０６－Ｈ５は、第十二ソースオペランド８０６Ａ－Ｈ５、第十三ソースオペランド８０６Ｂ－Ｈ５、第六ソースオペランド８０６Ｓ－Ｈ５、第五ソースオペランド８０６Ｃ－Ｈ５、第一ソースオペランド８０６Ｄ－Ｈ５、第三ソースオペランド８０６Ｅ－Ｈ５、出力先オペランド８０６Ｆ－Ｈ５を格納する。

【０１１５】
専用レジスタファイル８０８－Ｈ５は、第九ソースオペランド８０８Ａ－Ｈ５、第十ソースオペランド８０８Ｂ－Ｈ５、第十一ソースオペランド８０８Ｃ－Ｈ５、第二ソースオペランド８０８Ｄ－Ｈ５ａ、第四ソースオペランド８０８Ｄ－Ｈ５ｂ、第八ソースオペランド８０８Ｅ－Ｈ５を格納する。なお、第二ソースオペランド８０８Ｄ－Ｈ５ａと第四ソースオペランド８０８Ｄ－Ｈ５ｂは、図２０におけるペナルティソースオペランド８０８Ｄを、２つのオペランドに分けて独立させて汎用化したものであり、第一加減算器８４０Ａ－Ｈ５と、第二加減算器８４０Ｂ－Ｈ５に、別々の要素を加算できるようにしている。
【０１１６】
第二汎用算術論理演算器８４２－Ｈ５は、第二選択器８４１Ｂ－Ｈ５を備える。第二選択器８４１Ｂ－Ｈ５は、第十二ソースオペランド８０６Ａ－Ｈ５、第十三ソースオペランド８０６Ｂ－Ｈ５、第九ソースオペランド８０８Ａ－Ｈ５、第十ソースオペランド８０８Ｂ－Ｈ５、第十一ソースオペランド８０８Ｃ－Ｈ５の各要素を参照して、スコア関数ｓに基づいてスコアを出力するマルチプレクサとなる。第二選択器８４１Ｂ－Ｈ５の出力データは、第六ソースオペランド８０６Ｓ－Ｈ５に取り込むことができる。
【０１１７】
第一汎用算術論理演算器８４０－Ｈ５は、第一加減算器８４０Ａ－Ｈ５、第二加減算器８４０Ｂ－Ｈ５、第三加減算器８４０Ｃ－Ｈ５、第一選択器８４１Ａ－Ｈ５を有する。第三加減算器８４０Ｃ－Ｈ５は、第五ソースオペランド８０６Ｃ－Ｈ５の要素と第六ソースオペランド８０６Ｓ－Ｈ５の要素とを加算する。第一加減算器８４０Ａ－Ｈ５は、第一ソースオペランド８０６Ｄ－Ｈ５の要素と第二ソースオペランド８０８Ｄ－Ｈ５ａの要素とを加算する。第二加減算器８４０Ｂ－Ｈ５は、第三ソースオペランド８０６Ｅ－Ｈ５の要素と第四ソースオペランド８０８Ｄ－Ｈ５ｂの要素とを加算する。第一選択器８４１Ａ－Ｈ５は、第一加減算器８４０Ａ－Ｈ５の出力要素と、第二加減算器８４０Ｂ－Ｈ５の出力要素と、第三加減算器８４０Ｃ－Ｈ５の出力要素と、第八ソースオペランド８０８Ｅ－Ｈ５の要素の最大値又は最小値（ここでは最大値）を選択するマルチプレクサとなる。
【０１１８】
図２４のコアＡ５において汎用化された部品・部材に付す符号は、図２０に示すコアＡ２の専用部品・部材に付した符号に「－Ｈ５」を付加したものを採用している。この符号対応関係によって、図２４のコアＡ５の詳細機能の説明を省略する。
【０１１９】
以上の通り、図２４のコアＡ５は、第一ソースオペランド８０６Ｄ－Ｈ５と、第二ソースオペランド８０８Ｄ－Ｈ５ａと、第三ソースオペランド８０６Ｅ－Ｈ５と、第四ソースオペランド８０８Ｄ－Ｈ５ｂと、第五ソースオペランド８０６Ｃ－Ｈ５と、第六ソースオペランド８０６Ｓ－Ｈ５と、第八ソースオペランド８０８Ｅ－Ｈ５と、第一ソースオペランド８０６Ｄ－Ｈ５の要素と第二ソースオペランド８０８Ｄ－Ｈ５ａの要素の第一中間和を演算する第一加減算器８４０Ａ－Ｈ５と、第三ソースオペランド８０６Ｅ－Ｈ５の要素と第四ソースオペランド８０８Ｄ－Ｈ５ｂの要素の第二中間和を演算する第二加減算器８４０Ｂ－Ｈ５と、第五ソースオペランド８０６Ｃ－Ｈ５の要素と第六ソースオペランド８０６Ｓ－Ｈ５の要素に基づいて得られる第六起因データ（ここでは第六ソースオペランド８０６Ｓ－Ｈ５の要素そのもの）の第三中間和を演算する第三加減算器８４０Ｃ－Ｈ５と、少なくとも第一中間和、第二中間和、及び、第三中間和の、第八ソースオペランド８０８Ｅ－Ｈ５の要素の中から、最大値及び／又は最小値を選択する第一選択器８４１Ａ－Ｈ５と、を備える。
【０１２０】
このコアＡ５において単一命令で実行される機能を汎用式化すると以下式５となる。

【数５】

$$L=max\begin{cases} a+b \\ c+d \\ e+f' \\ g \end{cases}$$

ここで、Ｌは第一選択器８４１Ａ－Ｈ５の出力要素、ａは第一ソースオペランド８０６Ｄ－Ｈ５の要素、ｂは第二ソースオペランド８０８Ｄ－Ｈ５ａの要素、ｃは第三ソースオペランド８０６Ｅ－Ｈ５の要素、ｄは第四ソースオペランド８０８Ｄ－Ｈ５ｂの要素、ｅは第五ソースオペランド８０６Ｃ－Ｈ５の要素、ｆ'は第六ソースオペランド８０６Ｓ－Ｈ５の要素、ｇは第八ソースオペランド８０８Ｅ－Ｈ５の要素となる。

【０１２１】

図２４のコアＡ５は、図２０のコアＡ２と同様のアライメント処理に好適に用いることができるが、その他の汎用用途で使用することもできる。なお、ここでは汎用レジスタファイル８０６－Ｈ５と専用レジスタファイル８０８－Ｈ５に分ける場合を例示しているが、すべてのオペランドを汎用レジスタファイル８０６－Ｈ５に格納することで、更に、汎用性を高めるようにしてもよい。

【０１２２】

（第七変形例：汎用化事例の紹介）

図２２に示す汎用的なコアＡ３を更に簡素化した、第七変形例に係るコアＡ６を図２５に示す。このコアＡ６は、コアＡ３の汎用レジスタファイル８０６－Ｈ３に格納される第六ソースオペランド８０８Ａ－Ｈ３、第七ソースオペランド８０８Ｂ－Ｈ３、第八ソースオペランド８０８Ｃ－Ｈ３を省略している。代わりとして、コアＡ６は、これらの要素（第六ソースｒ１、第七ソースｒ２、第八ソースｒ３）の値を、デコーダ８０４－Ｈ６で処理される汎用演算命令の引数側に格納する。更にコアＡ６は、比較器８０７－Ｈ６及びフラグレシスタ８０９－Ｈ６を備えることができる。比較器８０７－Ｈ６では、第九ソースオペランド８０６Ａ－Ｈ６と第十ソースオペランド８０６Ｂ－Ｈ６の一致・不一致・欠損を比較演算し、その比較結果をフラグレジスタ８０９－Ｈ６に記録する。第二選択器８４１Ｂ－Ｈ６は、フラグレジスタ８０９－Ｈ６に記録される結果に基づいて、引数となる第六ソースｒ１、第七ソースｒ２、第八ソースｒ３のいずれかを、マスクレジスタを用いて抽出・出力するマルチプレクサとなる。換言すると、デコーダ８０４等の命令専用レジスタが、その引数を保持することによって、専用レジスタファイル８０８－Ｈ６や汎用レジスタファイル８０６－Ｈ６に格納されるオペランドの一部を兼ねることができる。コアＡ６において、図２２のコアＡ３の専用部品・部材に類似する機器の符号には、コアＡ３の符号の「－Ｈ３」を「－Ｈ６」に変更したものを採用することで、コアＡ６の詳細機能の説明を省略する。

【０１２３】

このコアＡ６において単一命令で実行される機能を汎用式化すると以下式６となる。

【数６】

$$L=max\begin{cases} a+b \\ c+d \\ e+(r1\ or\ r2\ or\ r3) \end{cases}$$

ここで、Ｌは第一選択器８４１Ａ－Ｈ３の出力要素、ａは第一ソースオペランド８０６Ｄ－Ｈ３の要素、ｂは第二ソースオペランド８０８Ｄ－Ｈ３ａの要素、ｃは第三ソースオペランド８０６Ｅ－Ｈ３の要素、ｄは第四ソースオペランド８０８Ｄ－Ｈ３ｂの要素、ｅは第五ソースオペランド８０６Ｃ－Ｈ３の要素、ｒ１、ｒ２、ｒ３は命令レジスタ（デコー

ダ８０４）に引数として格納される第六ソース、第七ソース、第八ソースとなる。

【０１２４】

なお、汎用化事例となるコアＡ４、Ａ５、Ａ６、Ａ７では、第二ソースオペランド８０８Ｄ－Ｈ３ａの要素ｂ、第四ソースオペランド８０８Ｄ－Ｈ３ｂの要素ｄの双方を用いる場合を例示したが、必要に応じて、要素ｂ、要素ｄの一方を省略しても良く、Ｓｍｉｔｈ－Ｗａｔｅｒｍａｎ－Ｇｏｔｏｈアルゴリズム等に好適となる。

【０１２５】

本実施形態のアライメント処理用コンピュータシステム２０４では、プロセッサモジュール１０４がコアＡ～コアＤを備えているため、コアＡ～Ｄに対して、単一命令で、アライメント行列Ｖの現在位置の変異度Ｆ（ｉ，ｊ）を算出することができる。これにより、メモリモジュール（メインメモリ）１０２へのアクセス回数を削減できると共に、Ｌ１～Ｌ３キャッシュメモリのヒット率が高くなるので、極めて高速な変異度演算が実現される。結果、被照合文字列と照合文字列とからなる文字列ペアに対して、短時間で、近似文字列を導出できる。更にアライメント処理用コンピュータシステム２０４では、定数設定手段６０４が。ギャップペナルティ等の定数データを、専用レジスタファイル８０８に保存することで、変異度・経路算出手段６１２における複数回の変異度の演算で共用する。これによっても、メモリモジュール（メインメモリ）１０２へのアクセス回数を削減できるので、演算速度を高めることができる。

【０１２６】

また、アライメント処理用コンピュータシステム２０４では、充填領域選択手段６０６が、アライメント行列Ｖの全体に対して一部となる配列領域Ｖ１Ｖ２…Ｖｋ…Ｖｐを抽出して、その範囲内で、変異度・経路算出手段６１２が変異度を演算するようにしているので、Ｌ１～Ｌ３キャッシュメモリのヒット率が一層高くなるので、演算速度を高めることができる。

【０１２７】

上記各実施形態は、本発明を説明するための例示であり、本発明をこれらの実施形態にのみ限定する趣旨ではない。本発明は、その要旨を逸脱しない限り、さまざまな形態で実施することができる。

【０１２８】

例えば、上記実施形態では、コアＡが、被照合文字列と照合文字列とからなる文字列ペアのアライメント処理を実行する場合を例示したが、複数のコアＡ～Ｄを並列的に利用して、共通の文字列ペアのアライメント処理を実行してもよい。この場合、アライメント処理用カーネルが、コアＡ～Ｄの演算順序を管理するために、例えば単一命令複数データ（Single Instruction, Multiple Data）方式を採用することができ、更に、パイプライン処理方式を採用してもよい。更にアライメント処理用カーネルは、単一命令複数スレッド（Single Instruction, Multiple Threads）方式を採用してもよい。また、ここでは４個のコアＡ～Ｄを例示したが、コアの数は特に限定されるものではない。

【０１２９】

例えば、本明細書に開示される方法においては、その結果に矛盾が生じない限り、ステップ、動作又は機能を並行して又は異なる順に実施しても良い。説明されたステップ、動作及び機能は、単なる例として提供されており、ステップ、動作及び機能のうちのいくつかは、発明の要旨を逸脱しない範囲で、省略でき、また、互いに結合させることで一つのものとしてもよく、また、他のステップ、動作又は機能を追加してもよい。

【０１３０】

上記各実施形態は、本発明を説明するための例示であり、本発明をこれらの実施形態にのみ限定する趣旨ではない。本発明は、その要旨を逸脱しない限り、さまざまな形態で実施することができる。例えば、本実施形態では、遺伝子情報となる参照文字列について、アライメント処理を行う場合を例示したが、本発明はこれに限定されず、遺伝子情報以外の文字列ペアをアライメント処理する場合に適用することができる。

【０１３１】

更に上記実施形態では、コンピューティングデバイスのハードウェア構成が、汎用的な構造である場合を例示したが、本発明はこれに限定されない。例えば、ＦＰＧＡ（Field Programmable Gate Array）等によって個別にカスタマイズされたハードウェアを採用してもよい。

【０１３２】

また、本明細書では、さまざまな実施形態が開示されているが、一の実施形態における特定のフィーチャ（技術的事項）を、適宜改良しながら、他の実施形態に追加し、又は該他の実施形態における特定のフィーチャと置換することができ、そのような形態も本発明の要旨に含まれる。

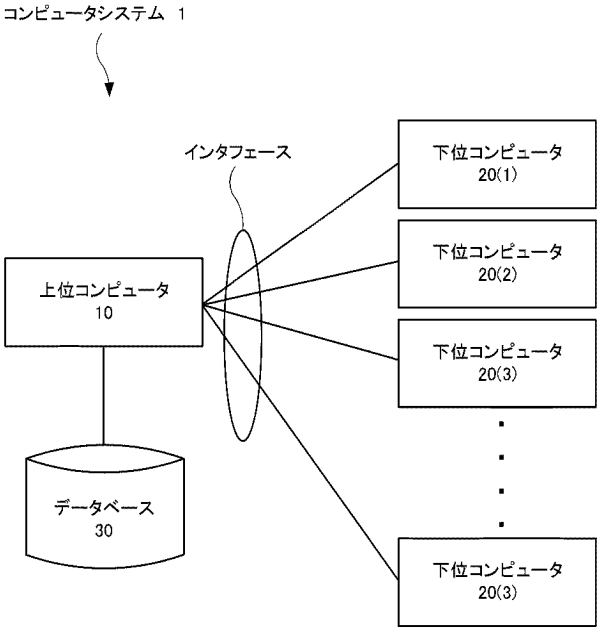
【符号の説明】

【０１３３】

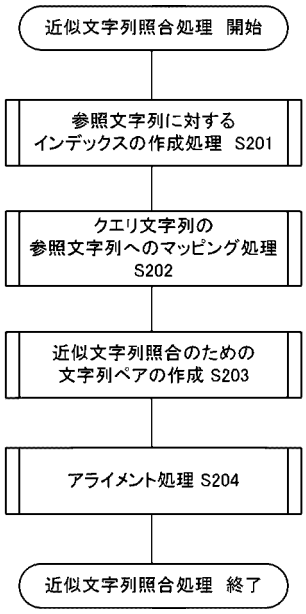
Ａ，Ｂ，Ｃ，Ｄ　　コア
１　コンピュータシステム
１０　上位コンピュータ
２０　下位コンピュータ
３０　データベース
１００　コンピューティングデバイス
１０２　メモリモジュール
１０４　プロセッサモジュール
１０６　チップセット
１０８　内部ストレージデバイス
１１２　出力インタフェース
１１４　入出力インタフェース
１１６　通信インタフェース
１１８　外部ストレージデバイス
２００　近似文字列照合用コンピュータシステム
２０１　インデックス作成用コンピュータシステム
２０２　マッピング用コンピュータシステム
２０３　文字列ペア作成用コンピュータシステム
２０４　アライメント処理用コンピュータシステム
６０２　アライメント行列作成手段
６０４　定数設定手段
６０６　充填領域選択手段
６０８　充填ルート設定手段
６１０　充填セル選択手段
６１２　変異度・経路算出手段
６２０　バックトラック処理手段
６２２　近似文字列導出手段
８００　プログラムカウンタ
８０４　デコーダ
８０６　汎用レジスタファイル
８０６Ａ　上配列文字ソースオペランド
８０６Ｂ　左配列文字ソースオペランド
８０６Ｃ　上変異度ソースオペランド
８０６Ｄ　真上変異度ソースオペランド
８０６Ｅ　真左変異度ソースオペランド
８０６Ｆ　出力先オペランド
８０６Ｓ　スコアソースオペランド
８０８　専用レジスタファイル
８０８Ａ　一致スコアソースオペランド

８０８Ｂ　不一致スコアソースオペランド
８０８Ｃ　ギャップスコアソースオペランド
８０８Ｄ　ペナルティソースオペランド
８０８Ｅ　制限定数オペランド
８３０　実行ユニット
８３２　加減算器
８３４　乗算器
８３６　浮動小数点演算器
８４０　変異度専用算術論理演算器
８４０Ａ　第一加減算器
８４０Ｂ　第二加減算器
８４０Ｃ　第三加減算器
８４１Ａ　変異度選択器
８４１Ｂ　スコア関数選択器
８４１Ｂ　スコア関数専用算術論理演算器
８４２　スコア関数専用算術論理演算器

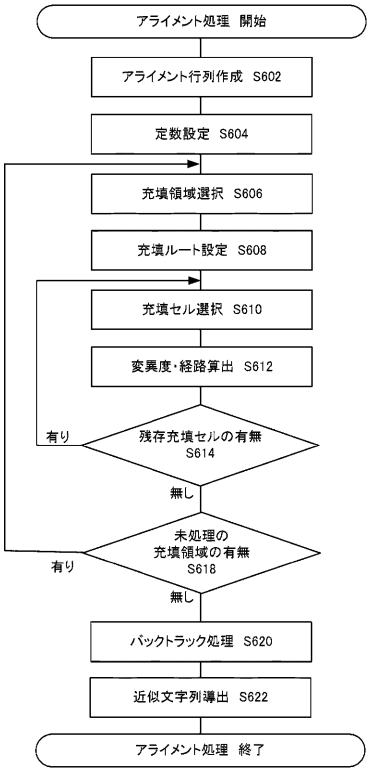
【図１】



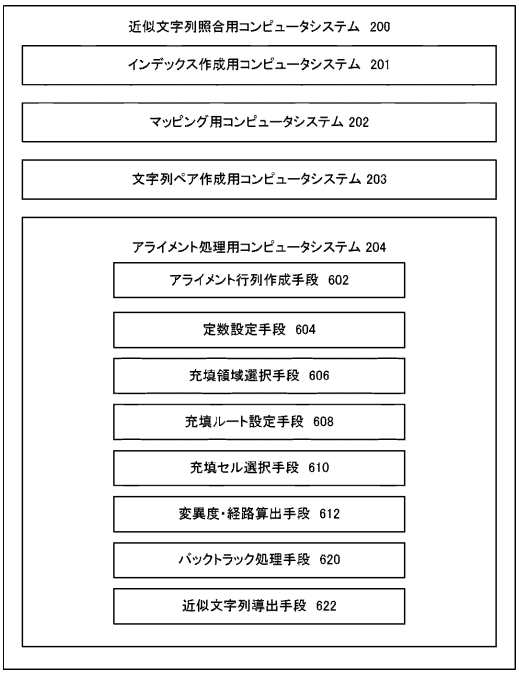
【図２】



【図３】



【図４】



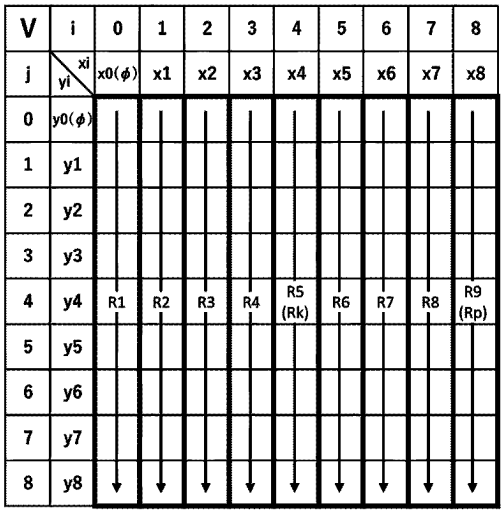
【図５】

V	i	0	1	2	3	4	5	6	7	8(m)
j	xi / yi	x0(ϕ)	x1	x2	x3	x4	x5	x6	x7	x8
0	y0(ϕ)									
1	y1									
2	y2									
3	y3									
4	y4									
5	y5									
6	y6									
7	y7									
8(n)	y8									

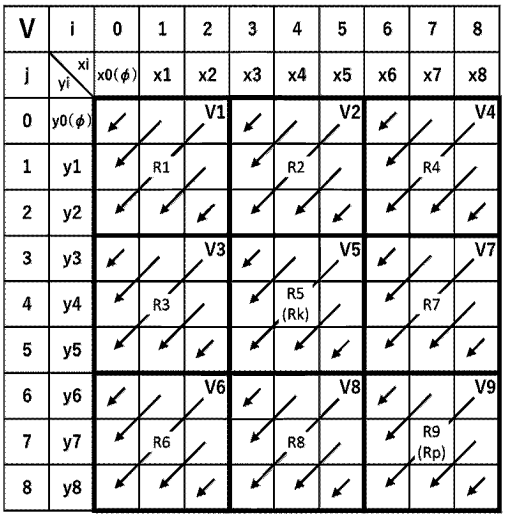
【図６】

V	i	0	1	2	3	4	5	6	7	8(m)	
j	xi / yi	x0(ϕ)	x1	x2	x3	x4	x5	x6	x7	x8	
0	y0(ϕ)	→				R1				→	V1
1	y1	→				R2				→	V2
2	y2	→				R3				→	V3
3	y3	→				R4				→	V4
4	y4	xs / ys				R5(Rk)				xe / ye	V5(Vk)
5	y5	→				R6				→	V6
6	y6	→				R7				→	V7
7	y7	→				R8				→	V8
8(n)	y8	→				R9(Rp)				→	V9(Vp)

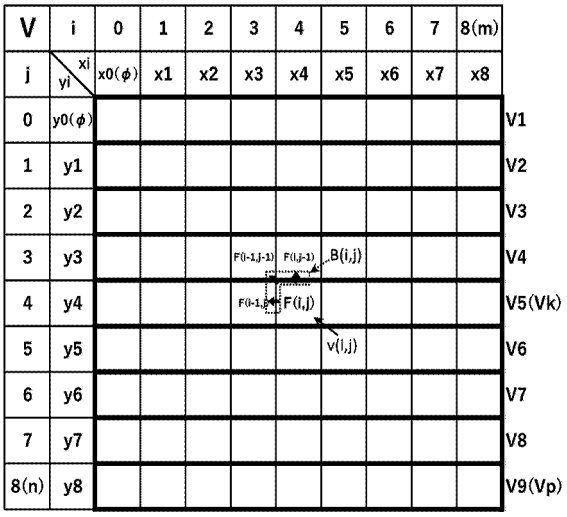
【図７】


V
i
j
xi
yi
0
1
2
3
4
5
6
7
8
x0(ϕ)
x1
x2
x3
x4
x5
x6
x7
x8
y0(ϕ)
y1
y2
y3
y4
y5
y6
y7
y8
R1
R2
R3
R4
R5
(Rk)
R6
R7
R8
R9
(Rp)

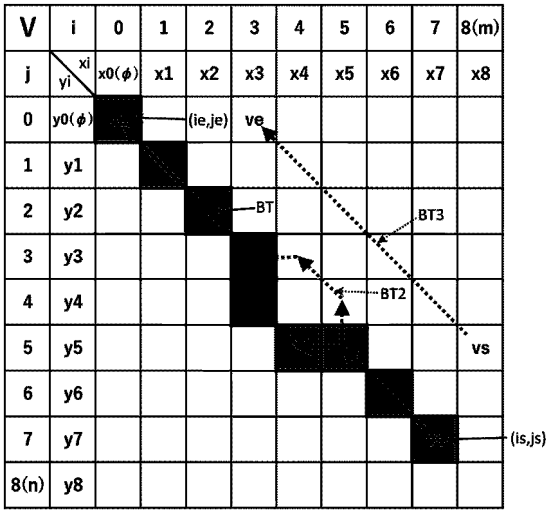
【図８】


V
i
j
xi
yi
0
1
2
3
4
5
6
7
8
x0(ϕ)
x1
x2
x3
x4
x5
x6
x7
x8
y0(ϕ)
y1
y2
y3
y4
y5
y6
y7
y8
V1
V2
V3
V4
V5
V6
V7
V8
V9
R1
R2
R3
R4
R5
(Rk)
R6
R7
R8
R9
(Rp)

【図９】


V
i
j
xi
yi
0
1
2
3
4
5
6
7
8(m)
8(n)
x0(ϕ)
x1
x2
x3
x4
x5
x6
x7
x8
y0(ϕ)
y1
y2
y3
y4
y5
y6
y7
y8
V1
V2
V3
V4
V5(Vk)
V6
V7
V8
V9(Vp)
F(i-1,j-1)
F(i,j-1)
B(i,j)
F(i-1,j)
F(i,j)
v(i,j)

【図１０】


V
i
j
xi
yi
0
1
2
3
4
5
6
7
8(m)
8(n)
x0(ϕ)
x1
x2
x3
x4
x5
x6
x7
x8
y0(ϕ)
y1
y2
y3
y4
y5
y6
y7
y8
(ie,je)
ve
BT
BT3
BT2
vs
(is,js)

【図１１】

V	i	0	1	2	3	4	5	6	7	8	
j	xi / yi	ϕ	G	C	C	T	C	G	C	T	
0	ϕ	0	-2	-4	-6	-8	-10	-12	-14	-16	V1
1	G										V2
2	C										V3
3	C										V4
4	A										V5
5	T										V6
6	T										V7
7	G										V8
8	A										V9

【図１２】

V	i	0	1	2	3	4	5	6	7	8	
j	xi / yi	ϕ	G	C	C	T	C	G	C	T	
0	ϕ	0	-2	-4	-6	-8	-10	-12	-14	-16	V1
1	G	-2	2	0	-2	-4	-6	-8	-10	-12	V2
2	C										V3
3	C										V4
4	A										V5
5	T										V6
6	T										V7
7	G										V8
8	A										V9

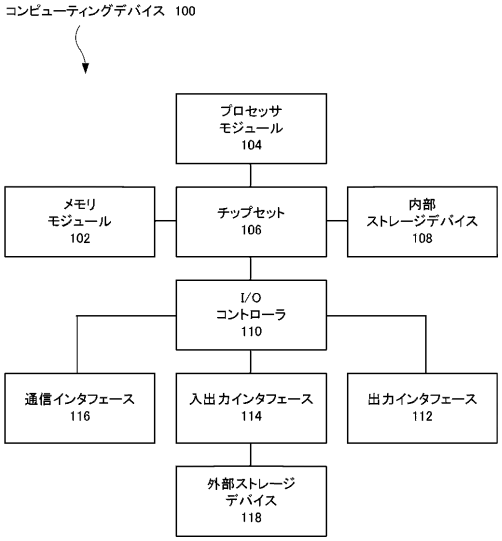
【図１３】

V	i	0	1	2	3	4	5	6	7	8	
j	xi / yi	ϕ	G	C	C	T	C	G	C	T	
0	ϕ	0	-2	-4	-6	-8	-10	-12	-14	-16	
1	G	-2	2	0	-2	-4	-6	-8	-10	-12	
2	C	-4	0	4	2	0	-2	-4	-6	-8	
3	C	-6	-2	2	6	4	2	0	2	4	
4	A	-8	-4	0	4	5	3	1	-1	-3	
5	T	-10	-6	-2	2	6	4	2	0	1	
6	T	-12	-8	-4	0	4	5	3	1	2	
7	G	-14	-10	-6	-2	2	3	7	5	3	BT
8	A	-16	-12	-8	-4	0	1	5	6	4	

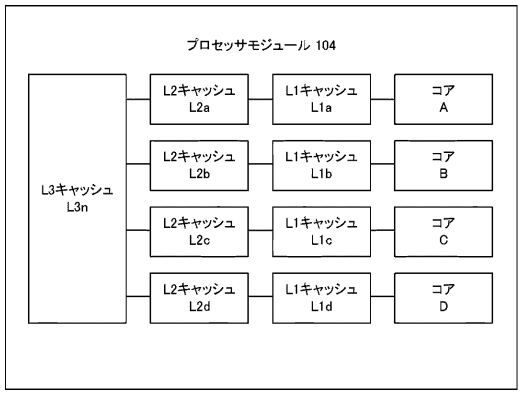
【図１４】

S	i	1	2	3	4	5	6	7	8
j	xi / yi	G	C	C	T	C	G	C	T
1	G	2	-1	-1	-1	-1	2	-1	-1
2	C	-1	2	2	-1	2	-1	2	-1
3	C	-1	2	2	-1	2	-1	2	-1
4	A	-1	-1	-1	-1	-1	-1	-1	-1
5	T	-1	-1	-1	2	-1	-1	-1	2
6	T	-1	-1	-1	2	-1	-1	-1	2
7	G	2	-1	-1	-1	-1	2	-1	-1
8	A	-1	-1	-1	-1	-1	-1	-1	-1

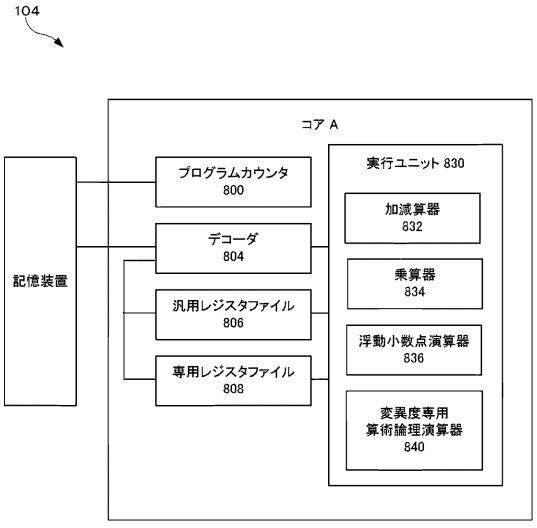
【図１５】



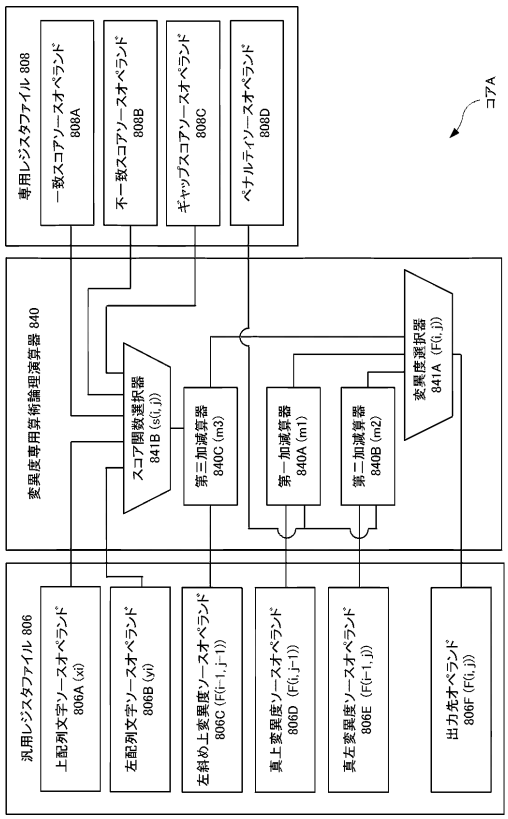
【図１６】



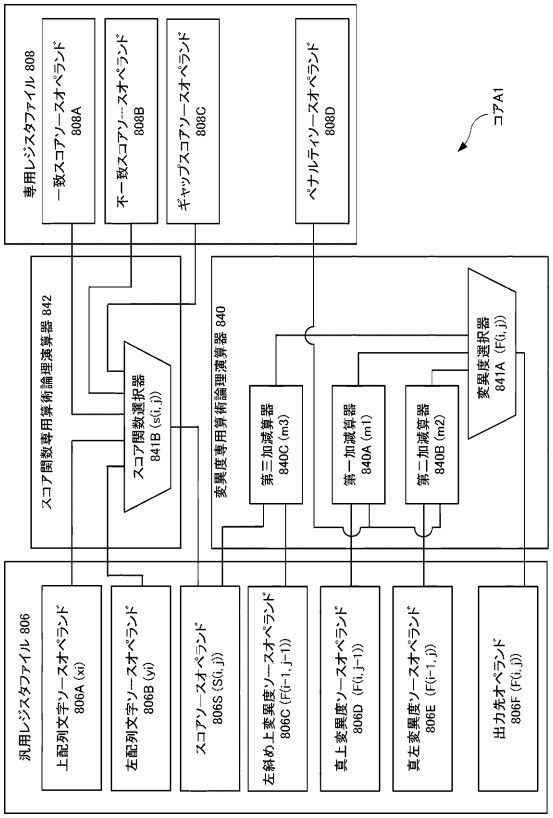
【図１７】



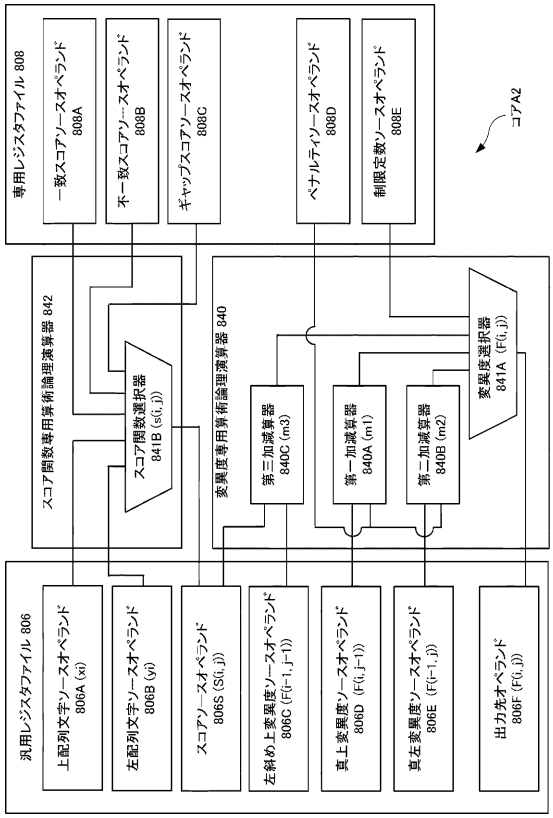
【図１８】



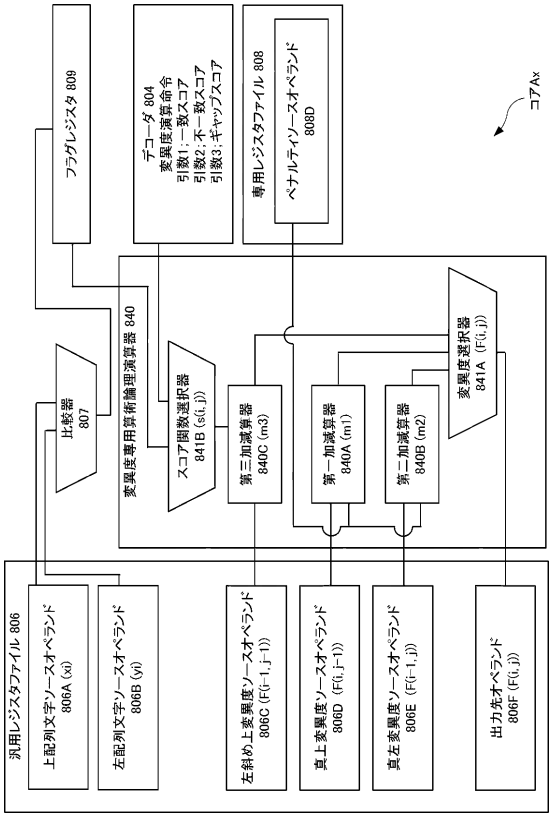
【図１９】



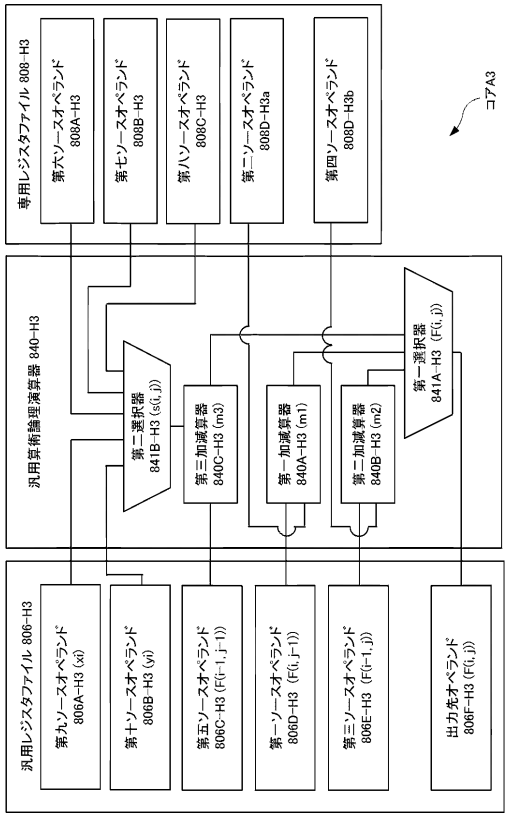
【図２０】



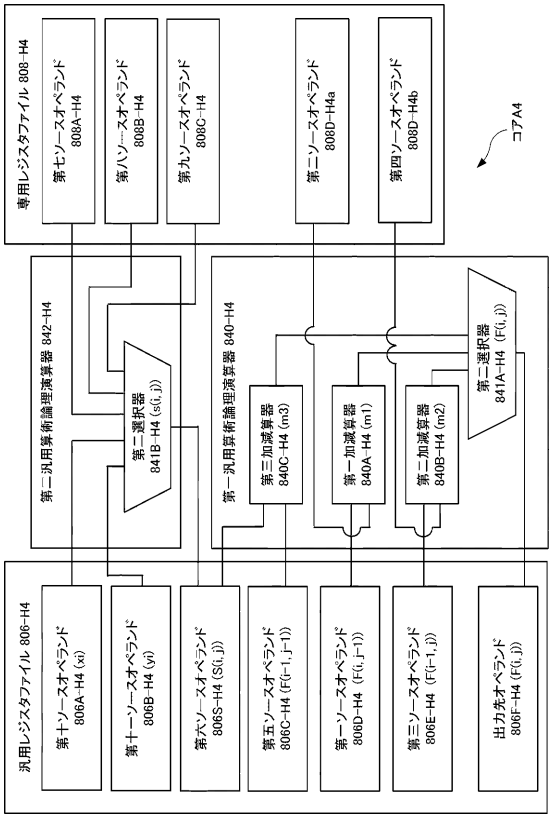
【図２１】



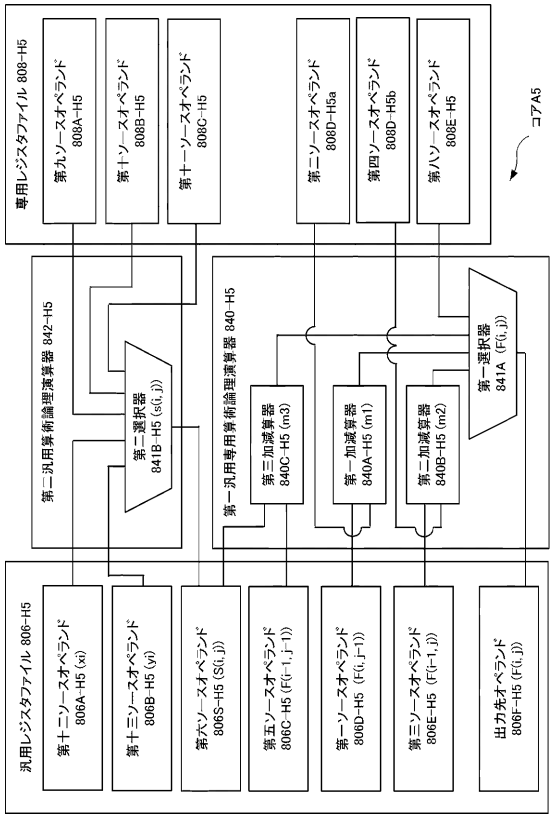
【図２２】



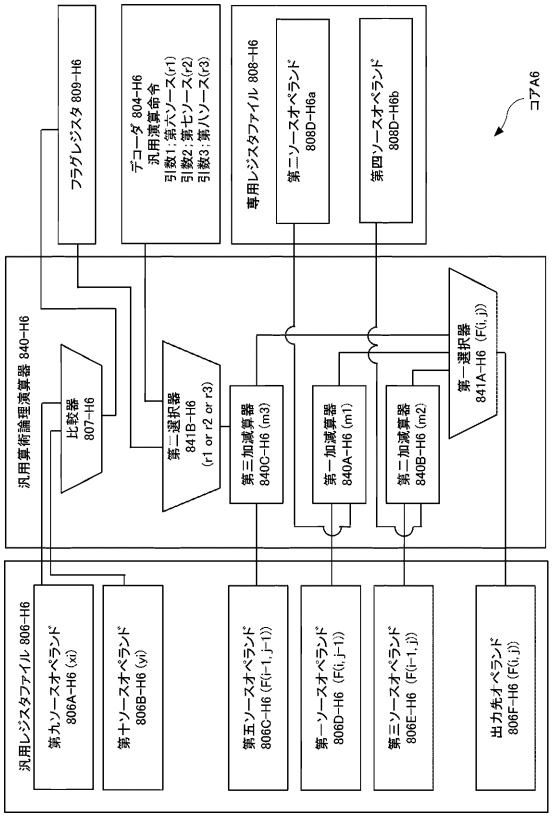
【図２３】


専用レジスタファイル 808-H4
第七ソースオペランド 808A-H4
第八ソースオペランド 808B-H4
第九ソースオペランド 808C-H4
第二ソースオペランド 808D-H4a
第四ソースオペランド 808D-H4b
コアA4
第二汎用算術論理演算器 842-H4
第二選択器 841B-H4 (s(i, j))
第一汎用算術論理演算器 840-H4
第三加減算器 840C-H4 (m3)
第一加減算器 840A-H4 (m1)
第二加減算器 840B-H4 (m2)
第一選択器 841A-H4 (F(i, j))
汎用レジスタファイル 806-H4
第十ソースオペランド 806A-H4 (xi)
第十一ソースオペランド 806B-H4 (yi)
第六ソースオペランド 806S-H4 (S(i, j))
第五ソースオペランド 806C-H4 (F(i-1, j-1))
第一ソースオペランド 806D-H4 (F(i, j-1))
第三ソースオペランド 806E-H4 (F(i-1, j))
出力先オペランド 806F-H4 (F(i, j))

【図２４】


専用レジスタファイル 808-H5
第九ソースオペランド 808A-H5
第十ソースオペランド 808B-H5
第十一ソースオペランド 808C-H5
第二ソースオペランド 808D-H5a
第四ソースオペランド 808D-H5b
第八ソースオペランド 808E-H5
コアA5
第二汎用算術論理演算器 842-H5
第二選択器 841B-H5 (s(i, j))
第一汎用専用算術論理演算器 840-H5
第三加減算器 840C-H5 (m3)
第一加減算器 840A-H5 (m1)
第二加減算器 840B-H5 (m2)
第一選択器 841A (F(i, j))
汎用レジスタファイル 806-H5
第十二ソースオペランド 806A-H5 (xi)
第十三ソースオペランド 806B-H5 (yi)
第六ソースオペランド 806S-H5 (S(i, j))
第五ソースオペランド 806C-H5 (F(i-1, j-1))
第一ソースオペランド 806D-H5 (F(i, j-1))
第三ソースオペランド 806E-H5 (F(i-1, j))
出力先オペランド 806F-H5 (F(i, j))

【図２５】


フラグレジスタ 809-H6
デコーダ 804-H6
汎用演算命令
引数1:第六ソース(r1)
引数2:第七ソース(r2)
引数3:第八ソース(r3)
専用レジスタファイル 808-H6
第二ソースオペランド 808D-H6a
第四ソースオペランド 808D-H6b
コアA6
汎用算術論理演算器 840-H6
比較器 807-H6
第二選択器 841B-H6 (r1 or r2 or r3)
第三加減算器 840C-H6 (m3)
第一加減算器 840A-H6 (m1)
第二加減算器 840B-H6 (m2)
第一選択器 841A-H6 (F(i, j))
汎用レジスタファイル 806-H6
第九ソースオペランド 806A-H6 (xi)
第十ソースオペランド 806B-H6 (yi)
第五ソースオペランド 806C-H6 (F(i-1, j-1))
第一ソースオペランド 806D-H6 (F(i, j-1))
第三ソースオペランド 806E-H6 (F(i-1, j))
出力先オペランド 806F-H6 (F(i, j))

フロントページの続き

(72)発明者　牧野　淳一郎
神奈川県横浜市金沢区泥亀一丁目２８番Ｂ－１３１２　先端加速システムズ株式会社内

Ｆターム(参考) 5B056 BB42