(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2007-122398
(P2007-122398A)
(43)公開日　平成19年5月17日(2007.5.17)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 17/30	(2006.01)	G06F	17/30	350C	5B075
G06F 13/00	(2006.01)	G06F	17/30	140	
		G06F	13/00	540F	

審査請求 未請求　請求項の数 5　OL　（全 16 頁）

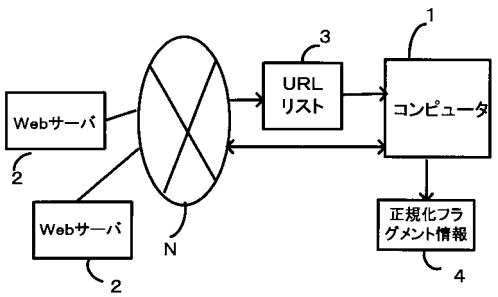
(21)出願番号　特願2005-313518 (P2005-313518)
(22)出願日　平成17年10月27日 (2005.10.27)

(71)出願人　591089084
中村　健一
東京都足立区谷中５－２－１７－２０７
(74)代理人　100120916
弁理士　佐藤　壽見子
(72)発明者　中村　健一
東京都足立区谷中５－２－１７－２０７

Fターム(参考)　5B075 ND03　ND36

(54)【発明の名称】フラグメントの同一性判定方法およびコンピュータプログラム

(57)【要約】
【課題】ＵＲＬが異なるＷｅｂページについて、ページ内容の全部又は一部が同一か否かを判定する手段としてフラグメントという概念を導入し、このフラグメントの同一性を判定する方法を提供する。
【解決手段】内容の異同を判定したいＷｅｂページのＵＲＬリストを入力情報とし、各ＵＲＬを指定して該当ページのＨＴＭＬ文書を取得し、各ＨＴＭＬ文書から抽出したフラグメントのそれぞれを３つのサブフラグメントに分割し、この３サブフラグメントとＵＲＬ文字列とを判定要素とすることによって、異なるフラグメント同士の同一性の有無を判定する。判定結果に基づいて、正規化されたフラグメント情報を生成する。この正規化フラグメント情報は、Ｗｅｂページの全部又は一部の内容の異同を推定するために利用することができる。
【選択図】図１



【特許請求の範囲】
【請求項１】
インターネットと接続可能なコンピュータが、
インターネット上で公開されているＷｅｂページの任意個数のＵＲＬを入力情報とするステップと、
前記の各ＵＲＬに対応するＷｅｂページのＨＴＭＬ文書を取得するステップと、
取得した各ＨＴＭＬ文書から、リンク要素周辺の文字情報であるフラグメントを抽出するステップと、
抽出した各フラグメントを、リンク要素直前の前フラグメント、リンク要素内部の中フラグメント、およびリンク要素直後の後フラグメントに３分割するとともに、これらの分割された３種類のサブフラグメントに、該フラグメントを含むＨＴＭＬ文書に対応するＵＲＬの文字列を加えた４つを判定要素とするステップと、
或るフラグメントの４つの判定要素のそれぞれについて、他のフラグメントの対応する判定要素との類似度を算出するステップと
算出された４つの類似度を総合的に評価した結果に基づいて、フラグメント同士の同一性の有無を判定するステップと、
前記同一性判定結果に基づいて、正規化されたフラグメント情報を生成し、出力するステップ、
とからなることを特徴とするフラグメントの同一性判定方法。
【請求項２】
請求項１に記載のフラグメントの同一性判定方法において、
前フラグメントと後フラグメントのそれぞれを構成する要素については、リンク要素からの距離に応じて重み付けを行い、類似度の評価をすることを特徴とするもの。
【請求項３】
請求項１に記載のフラグメントの同一性判定方法において、
３種類の各サブフラグメントの一致・不一致を評価するためのしきい値の決定に際し、比較される各フラグメントのＵＲＬ文字列同士の類似度を利用することを特徴とするもの。
【請求項４】
請求項３に記載のフラグメントの同一性判定方法において、
前フラグメント、中フラグメントおよび後フラグメントの３種類のサブフラグメントのうち、少なくとも２種類以上が一致すると判定された場合に、同一性のあるフラグメントであると判定することを特徴とするもの。
【請求項５】
請求項１～４のいずれか１に記載の方法を、コンピュータに実行させるためのコンピュータプログラム。

【発明の詳細な説明】
【技術分野】
【０００１】
ＵＲＬが異なるＷｅｂページについて、ページ内容の全部又は一部が同一か否かを判定するための方法に関する。
【背景技術】
【０００２】
インターネット上には、大量のＷｅｂページが存在し、あるＷｅｂページから他のＷｅｂページへのリンクを張ることが広く行われている。あるＷｅｂページにアクセスした閲覧者は、そのＷｅｂページに張られたリンクをたどって別のＷｅｂページへアクセスし、そのアクセス先のＷｅｂページもまた、別のＷｅｂページへのリンクを張っていると、そのＷｅｂページへアクセスするというように、次々とＷｅｂページを閲覧していくことができる。
【０００３】

また、Ｗｅｂブラウザとサーバとの通信プロトコルであるＨＴＴＰには、リファラーと呼ばれる情報が規定されている。このリファラー情報を利用すると、リンク先ページへのリンクを張っているリンク元ページのＵＲＬ（Ｕｎｉｆｏｒｍ　Ｒｅｓｏｕｒｃｅ　Ｌｏｃａｔｏｒ）を認識することが可能である。このリファラー情報を利用して、自己をリンク先とするリンク元一覧の表示を行うＷｅｂページも出現し、この一覧から適当なＵＲＬを指定して、そのリンク元のページを閲覧することもできる。

なお、本出願人が、ＵＲＬ：ｈｔｔｐ：／／ｓｏｕｒｃｅｆｏｒｇｅ．ｊｐ／ｐｒｏｊｅｃｔｓ／ｒｅｆｅｒｅｒｈｏｕｎｄ／にて公開しているプログラムも、このリファラー情報を利用している。

【発明の開示】

【発明が解決しようとする課題】

【０００４】

このようなリンク元一覧を表示することは、たいへん便利なものであり、共通の関心を持つ者同士の交流手段や、広告宣伝手段としてのインターネットの可能性を広げるものである。

しかし、現行のリンク元一覧表示には、いくつかの問題がある。この問題について、図１４、図１５を参照しながら、説明する。

【０００５】

問題点の第一は、リンク元のＵＲＬを偽ることが可能であるということである。

例えば、ある邪悪な意図をもったクライアント１０３ａが、スパムを使って宣伝しているようなサイト１０２ａをリンク元としてＷｅｂサイト１０１にアクセスしてきたように装うことが可能である。リンク先サイト１０１では、このリンク元サイト１０２ａのＵＲＬをリンク元一覧に掲載することもありうる（図１５のＵＲＬ１０４）。その場合、このような事情を知らずに当該サイト１０１にアクセスしてきたクライアント１０３ｂは、リンク元一覧にあるサイト１０２ａへアクセスしてしまう、といったおそれがある。もし、そのサイト１０２ａが、例えば、非合法なドラッグを販売するようなサイトであったならば、リンク先サイト１０１の評判まで落としかねない。

【０００６】

第二の問題点は、通常、一つのＷｅｂページは複数のＵＲＬ、すなわち別名を持つということである。

たとえば、Ｗｅｂブラウザ上で、次の５つのＵＲＬのどの１つを指定しても、サイト１０２ｂの同じＷｅｂページが得られるものとする。

ｈｔｔｐ：／／ｗｗｗ．ｗｅｂ＿２ｂ．ｃｏｍ
ｈｔｔｐ：／／ｗｅｂ＿２ｂ．ｃｏｍ
ｈｔｔｐ：／／ｗｗｗ．ｗｅｂ＿２ｂ．ｃｏｍ／ｉｎｄｅｘ．ｈｔｍｌ
ｈｔｔｐ：／／ｗｅｂ＿２ｂ．ｃｏｍ／ｉｎｄｅｘ．ｈｔｍｌ
ｈｔｔｐ：／／２１４．＊．＊．＊（サイト１０２ｂのＩＰアドレス）

閲覧者が、このリンク元サイト１０２ｂの同一ページを介してサイト１０１にアクセスしてきた場合、上記のリファラー情報を利用すると、リンク先１０１は、すべて異なるＷｅｂページであると認識し、重複してリンク元一覧に載せてしまいかねない（図１５のＵＲＬ１０５）。そのため、リンク先１０１にアクセスしてきた閲覧者は、リンク元一覧に掲載されているのは、それぞれ異なるＷｅｂページであると勘違いをした結果、同じ内容のＷｅｂページに何度もアクセスすることもありうる。

また、事実上同一のＷｅｂページが重複して掲載されてしまうと、リンク元一覧が膨大となって極めて見にくいＷｅｂページとなり、閲覧者の不便を招くことになる。

【０００７】

この別名の問題点は、目下、大流行の兆しが見えるＷｅｂログ（以下「ブログ」という）において特に重大である。このブログでは、複数の異なるページに同一の記事が含まれ、それらの複数ページは、みな異なるＵＲＬをもつからである。

【０００８】

第１の問題を解決するためには、リンク元１０２ａのＷｅｂページに、自己のサイト１０１に含まれるページへのリンクが存在するか否かを確認すればよい。もし存在しないならば、そのリファラー情報は虚偽であるので、そのリンク元１０２ａのＵＲＬは、リンク元一覧に載せないようにすることで問題の解決が図られる。

【０００９】

第２の問題、すなわち、同一Ｗｅｂページが複数のＵＲＬを持つという問題を解決するためには、リンク元のＵＲＬを指定することによりリンク元ページのＨＴＭＬ文書を入手し、このＨＴＭＬ文書の内容から、すでにリンク元一覧に掲載済みのＷｅｂページと同一内容のＷｅｂページか否かを判定すればよい。そして、ＵＲＬが異なっていても、ページ内容が同一であれば、掲載済みと判定し、リンク元一覧には載せないようにすることで問題が解決できる。

【００１０】

本発明は、上記の問題解決のためにフラグメントという概念を導入し、このフラグメントの同一性判定結果を、比較されるＷｅｂページ同士の全部又は一部の異同の推定に役立てようとするものである。この点では、ＵＲＬ：ｈｔｔｐ：／／ｓｏｕｒｃｅｆｏｒｇｅ．ｊｐ／ｐｒｏｊｅｃｔｓ／ｒｅｆｅｒｅｒｈｏｕｎｄ／にて公開しているプログラムも同様である。

しかし、インターネット上には膨大な量のサイトが、相互にリンクを張り巡らしている現状を考えると、このようなフラグメントの異同判定処理に伴う計算量は無視できない。

そのため、本発明は、この計算量を削減し、上記の問題解決手段を実用に耐え得るものとすることも目的とする。

【００１１】

また、計算量を削減しようとすると精度が低下しがちである。そのため、異なるフラグメントを同一と判定したり（第一種の過誤）、同一フラグメントを異なるものと判定したり（第二種の過誤）することを極力回避する手段を講じることも目的とする。

なお、本発明は、第二の問題を解決する手法の提供が主眼であるが、判定処理の過程から、第一の問題の解決手段も提供している。

【課題を解決するための手段】

【００１２】

このような目的を達成するために、請求項１に記載の発明の方法は、インターネットと接続可能なコンピュータが、インターネット上で公開されているＷｅｂページの任意個数のＵＲＬを入力情報とするステップと、前記の各ＵＲＬに対応するＷｅｂページのＨＴＭＬ文書を取得するステップと、取得した各ＨＴＭＬ文書から、リンク要素周辺の文字情報であるフラグメントを抽出するステップと、抽出した各フラグメントを、リンク要素直前の前フラグメント、リンク要素内部の中フラグメント、およびリンク要素直後の後フラグメントに３分割するとともに、これらの分割された３種類のサブフラグメントに、該フラグメントを含むＨＴＭＬ文書に対応するＵＲＬの文字列を加えた４つを判定要素とするステップと、或るフラグメントの４つの判定要素のそれぞれについて、他のフラグメントの対応する判定要素との類似度を算出するステップと算出された４つの類似度を総合的に評価した結果に基づいて、フラグメント同士の同一性の有無を判定するステップと、前記同一性判定結果に基づいて、正規化されたフラグメント情報を生成し、出力するステップ、とからなることを特徴とする。

【００１３】

「リンク元Ｗｅｂページの ＵＲＬを取得」するために、通信プロトコルＨＴＴＰに規定されているリファラー情報を利用する。この情報によって、コンピュータは、リンク元のＵＲＬを認識できる。ただし、本発明にとって不可欠であるのは、リファラー情報ではなく、リンク元Ｗｅｂページを探し出す手段である。このような手段のひとつがリファラー情報であるが、他にも、現在広く普及している検索エンジンには、リンク元ページを探し出す機能を有するものもあり、これを利用してもよい。

【００１４】

「ＨＴＭＬ文書」とは、Ｗｅｂページを記述したファイルであって、Ｗｅｂブラウザによって読み込まれ、画面にＷｅｂページとして表示されるものをいう。このようなファイルは、ＨＴＭＬで記述されているものが多いので、「ＨＴＭＬ文書」と表現する。
「リンク要素」とは、Ｗｅｂページ上にリンクを作り出す要素をいう。代表的なリンク要素としては、ＨＴＭＬ文書において、対をなす＜Ａ＞タグで始まり、＜／Ａ＞タグで終了する文字列があるが、ＪａｖａＳｃｒｉｐｔ（登録商標）を用いてリンクされているものもリンク要素として判別される。

【００１５】

「フラグメント」とは、リンク要素周辺の文字情報を単純な文字列として抽出して得た一つのまとまりであって、抽出すべき全体の文字数や、タグの前後の文字数は任意に決めればよい。
本発明では、フラグメントを、＜Ａ＞タグの前か、タグの内側か、＜／Ａ＞タグの後かによって、前フラグメント、中フラグメントおよび後フラグメントの３つのサブフラグメントに分割する。
なお、以下の説明において、サブフラグメントに分割する前のフラグメント、及びサブフラグメントの両者を区別しないで、単に「フラグメント」ということもある。

【００１６】

フラグメントは、本発明の基本となる概念である。このフラグメントを利用した判定方法の特徴は、意味を解釈するのではなく、単なる文字列の一致・不一致を判定するという点にある。たとえば、「明るい」と「あかるい」は同一でないと判定する。これは、字面を見るだけであって、文字列の意味まで考慮していないからである。
なお、リンク要素は、一つのＨＴＭＬ文書に複数ありうる。或る一つのリンク先に着目し、このリンク先にリンクを張っているリンク元を抽出したい場合は、自己（＝リンク先）のＵＲＬ文字列を、ＵＲＬの値として持つリンク要素のみを本発明の処理対象としてもよい。しかし、特定のリンク先に着目することなく、インターネット上で張り巡らされている複数のリンク元・リンク先についてのフラグメント情報を得ようとするのが目的であれば、一つのＨＴＭＬ文書内のすべてのリンク要素を処理対象とすればよい。

【００１７】

「正規化されたフラグメント情報」とは、次の２つの条件（条件１）と（条件２）を満たしているフラグメント情報のことである。すなわち、（条件１）は、フラグメントの同一性を判定した結果、互いに一致するフラグメントが重複出現しないこと、（条件２）はこれらの重複のないフラグメントが、それを含むＷｅｂページのＵＲＬと多対多で対応していることである。

【００１８】

このように、請求項１に記載の発明によれば、ＨＴＭＬ文書の中からフラグメントを取り出し、他のＨＴＭＬ文書中のフラグメントとの異同を比較し、このフラグメントの比較結果によって、Ｗｅｂページの全部又は一部の異同を推定できる。ＨＴＭＬ文書全体ではなく、フラグメントの同一性の有無のみを判定すればよいので、計算量が削減される。しかも、フラグメントを３つに分け、比較的文字数の少ない文字列同士について別個に類似度の評価をするので、一層計算量が削減される。また、ＵＲＬ文字列同士の類似性も加味することで、判定の精度も向上する。
なお、ｈｔｔｐ：／／ｓｏｕｒｃｅｆｏｒｇｅ．ｊｐ／ｐｒｏｊｅｃｔｓ／ｒｅｆｅｒｅｒｈｏｕｎｄ／において公開されているプログラムも、本発明と同様、リンク要素周辺の文字情報を単純な文字列として抽出し、その同一性の評価によってリンク元の重複を防いでいる。しかし、このプログラムにおいては、リンク要素周辺の文字情報はサブフラグメントに分割されず、一続きのままで同一性を評価している。このため、本発明の方法と同程度の精度で同一性評価を行うには、本発明の方法に比べて１４０～３００％の計算量を要する。

【００１９】

また、上記した目的を達成するために、前フラグメントと後フラグメントのそれぞれを構

成する要素については、リンク要素からの距離に応じて重み付けを行い、類似度の評価をすることとしてもよい。

これにより、類似度評価の精度が向上する。なぜならば、リンク要素から離れるに従って、それがノイズ成分である可能性が高く、重み付けによって、ノイズ成分を低く評価できるからである。なお、「ノイズ成分」とは、リンク要素と意味的なつながりのない文字列のことをいう。

フラグメントを構成する「要素」とは、文字あるいは部分文字列をいう。

【００２０】

さらに、３種類の各サブフラグメントの一致・不一致を評価するためのしきい値の決定に際し、比較される各フラグメントのＵＲＬ文字列同士の類似度を利用するようにしてもよい。この場合、前フラグメント、中フラグメントおよび後フラグメントの３種類のサブフラグメントのうち、少なくとも２種類以上が一致すると判定された場合に、同一性のあるフラグメントであると判定するようにしてもよい。

【００２１】

これにより、サブフラグメントだけでなくＵＲＬ文字列も含む４つの要素についての類似度を総合的に評価するので、フラグメント同士の同一性評価の精度が向上する。

【００２２】

上述のフラグメント同士の同一性判定機能を、コンピュータに実現させるためのコンピュータプログラムも本発明である。

【発明の効果】

【００２３】

アクセス回数の多いＷｅｂページには大量のリンク元が存在する。これらのリンク元の重複チェックのためには、リンク元ＷｅｂページのＨＴＭＬ文書を既知のリンク元ＨＴＭＬ文書と比較しなければならない。本発明は、Ｗｅｂページ全体を比較することなく、ＨＴＭＬ文書から抽出したフラグメントの同一性を判定することにより、この比較のための計算量を大幅に削減し、重複チェックを実用に耐えるものとすることができる。

一般的に、計算量の削減と精度とはトレードオフの関係にあるが、本発明は、精度の低下を抑制できる。

【発明を実施するための最良の形態】

【００２４】

（１）本発明の実施形態であるシステム例の構成

図１に従い、本発明を実施するシステム例を説明する。

コンピュータ１は、インターネットＮに接続可能であり、インターネットＮに接続するＷｅｂサーバ２等の他のコンピュータと情報の送受信を行うものである。

【００２５】

コンピュータ１は、インターネットＮ上に張り巡らされたリンクのリンク元及びリンク先のＵＲＬリスト３を入手できる。この入手方法は、リファラー情報の利用によるものでも、現在普及している各種検索エンジンから提供を受けるものでも何でもよい。コンピュータ１は、前記のＵＲＬリスト３を入力とし、フラグメントの異同判定処理を行った結果を、正規化フラグメント情報４として出力する。

なお、リンク先およびリンク元のＷｅｂサイトには、個人が公開するブログなども含まれる。

【００２６】

コンピュータ１は、図２に示すように、処理部５、記憶部６、インタフェース部７を備え、他に図示しない入力部や出力部も備えている。

【００２７】

記憶部６は、本発明をコンピュータ１に実施させるためのプログラムＰＲＯＧを記憶したプログラム格納部８、ＵＲＬリスト格納部９、抽出フラグメントテーブル格納部１０、正規化フラグメント情報格納部１１を備え、他に各種パラメータなども記憶する。

ＵＲＬリスト格納部９は、入力されたＵＲＬリスト３を記憶しておくものであり、フラグ

メント同一性判定処理の間のみ記憶するものであってもよい。
抽出フラグメントテーブル格納部１０は、ＨＴＭＬ文書から抽出したフラグメントを３分割したサブフラグメント及びＵＲＬを記憶するものであり、フラグメント同一性判定処理の間のみ記憶するものであってもよい。
正規化フラグメント情報格納部１１は、出力結果である正規化フラグメント情報４を記憶するものである。なお、記憶部６は、各種補助記憶装置やＲＯＭ，ＲＡＭなどによって実装されている。
【００２８】
処理部５は、プログラム格納部８からプログラムＰＲＯＧをメモリ上に読み込み、このプログラムＰＲＯＧの命令コードを実行する。処理部５は、図示しないＣＰＵで実現される。
インタフェース部７は、外部との信号の送受信を行う。
【００２９】
（２）本発明による判定処理の概要
上記のシステム構成のもと、コンピュータ１によって、比較対象となるフラグメントの同一性の有無が、どのように判定されるかについて、図３の処理フローに従い、説明する。
コンピュータ１に０個以上のリンク先のＵＲＬと、１個以上のリンク元のＵＲＬが列挙されたＵＲＬリスト３が入力される（ステップＳ１）。コンピュータ１は、ＵＲＬリスト３を参照し１個ずつリンク元のＵＲＬを取り出し、当該ＵＲＬを指定して、Ｗｅｂページ閲覧要求をし、当該ページのＨＴＭＬ文書を入手する（ステップＳ２）。
【００３０】
このＨＴＭＬ文書からリンク先のＵＲＬ文字列を検索する（ステップＳ３）。
具体的には、ＨＴＭＬ文書中に、リンク先のＵＲＬ文字列が、リンク要素のＵＲＬの値として出現するか否かを検索する。
リンク先のＵＲＬ文字列が見つからなければ（ステップＳ４でＮｏ）、このＵＲＬに対応するサイトは虚偽のリンク元であると判断して、このＵＲＬ文字列についてのフラグメント抽出処理をスキップする。リンク先とは無関係のＷｅｂページを対象に、フラグメントの正規化を行っても意味がないからである。
【００３１】
リンク先のＵＲＬ文字列が見つかれば（ステップＳ４でＹｅｓ）、そのリンク要素周辺の文字情報を取り出す（ステップＳ５）。取り出された文字情報が、本発明のフラグメントである。取り出されたフラグメントを３種類のサブフラグメントに分割し、これらのサブフラグメントをＵＲＬ文字列とともに抽出フラグメントテーブル格納部１０に登録する（ステップＳ６）。図４に、このテーブルへの登録例を示す。
このようにして、すべてのリンク元ＵＲＬに対応するＨＴＭＬ文書から、リンク先ＵＲＬを含むフラグメントをすべて取り出す。
【００３２】
次に、抽出された各フラグメントについてステップＳ７からＳ８の処理を行う。ある一つのフラグメントに着目し、比較対象となるフラグメントを取り出す（ステップＳ７）。
比較対象となるのは、異なるＵＲＬに対応したＨＴＭＬ文書に含まれ、かつ、同一のリンク先を持ち、しかも、未だ同一性判定がなされていないフラグメントである。
比較対象たりうるフラグメントが取り出されると、同一性判定処理を行う（ステップＳ８）。この処理の内容については、後に詳しく説明する。
【００３３】
上記の同一性判定結果に基づき、フラグメント情報の正規化を行う（ステップＳ９）。この正規化の処理は、ステップＳ９のように、全フラグメントについての同一性判定を終了してから行ってもよいが、ステップＳ６からＳ７のループの中で、逐次行ってもよい。
なお、これを後の処理等で活用するために、正規化フラグメント情報格納部１１に記憶する。
【００３４】

ここで、ステップＳ９のフラグメントの正規化について、具体例を挙げて説明する。
図５に示すように、リンク元のＵＲＬとして、ＵＲＬ＿Ａ、ＵＲＬ＿Ｂ及びＵＲＬ＿Ｃの３つがあり、ＵＲＬ＿Ａに対応するＨＴＭＬ文書には、リンク先ＵＲＬを含むフラグメントＡ１、Ａ２，Ａ３の３つがあり、ＵＲＬ＿Ｂに対応するＨＴＭＬ文書には、フラグメントＢ１，Ｂ２，Ｂ３の３つがあり、ＵＲＬ＿Ｃに対応するＨＴＭＬ文書には、フラグメントＣ１、Ｃ２の２つがあるものとする。図５の同一性判定結果欄の記号が同じものは、互いに同一性があると判定されたフラグメントである。
この例では、図６に示すように、互いに同一性のないフラグメントのグループが４つある。
第１のグループを、フラグメントＡ１で代表させるならば、フラグメントＢ２は不要の情報となる。同様に、第２のグループを、フラグメントＡ２で代表させると、フラグメントＢ１とフラグメントＣ２は不要の情報となり、第３のグループを、フラグメントＢ３で代表させると、フラグメントＣ１は不要の情報となる。
【００３５】
なお、どのフラグメントで代表させるかについての基準として、例えば、次の（ａ）、（ｂ）が考えられる。すなわち、（ａ）そのフラグメントに対応するリンク元からの訪問者数が所定時間（例えば、過去２４時間）内で、一番多いフラグメント、（ｂ）最後に存在を確認してからの時間がもっとも短いフラグメントである。
（ａ）の基準で一つに決まれば、そのフラグメントを残すことになるが、もし、決まらなければ、（ｂ）の基準で決めることになる。
【００３６】
このような、正規化処理の結果を図７に示す。この例からもわかるように、正規化処理の結果は、フラグメントとＵＲＬとの対応は一般に多対多となる。なお、１対１、１対多、多対１対応を含むことは言うまでもない。
図７で例示するような多対多の対応を、正規化フラグメント情報格納部１１に格納する際のデータ構造は、実装レベルの問題であり、特に限定しない。ただし、フラグメントには、図７の括弧内に示すように、代表されるフラグメントが含まれていたＨＴＭＬ文書のＵＲＬも併せて登録する。その理由は、フラグメント間の同一性判定には、後述するように、ＵＲＬ文字列の類似度を組み込んでいるからである。もし、正規化したときに、ＵＲＬも登録しておかないと、いったん正規化した後に見つかったフラグメントとの間で、ＵＲＬ文字列を含めた判定ができなくなる。
【００３７】
以上の処理の結果、あるＷｅｂページをリンク先とするリンク元のフラグメントの同一性が判定できた。同一性あるフラグメントを有するＷｅｂページ同士は、全部または一部の内容が一致すると推定できる。そのため、正規化フラグメント情報を参照することによって、リンク元一覧には、内容が重複するＷｅｂページのＵＲＬを重複掲載しないようにすることができる。図７の例では、ＵＲＬ＿ＢとＵＲＬ＿ＣのＷｅｂページは、ＵＲＬ＿ＡのＷｅｂページと内容的な重複があると推定されるので、ＵＲＬ＿Ａだけをリンク元一覧に掲載すればよい。
なお、Ｗｅｂページでは、リンク元一覧に、ユーザに示すためのテキストを記述することがある。このテキストは、本発明のフラグメントとは無関係である。フラグメントは、類似性評価用の情報にすぎず、ユーザにどのようなテキストを示すかは、別の問題である。
【００３８】
以上、本発明によるフラグメントの同一性判定方法の概要を説明した。
しかし、以上の説明は、あくまで例示である。ＵＲＬリストを入力とし、正規化フラグメント情報を出力する処理を、少ない計算量で、かつ、必要十分な精度で行うことが本質であって、この処理のための具体的なシステム構成等の実装は限定されるものではない。
以下、フラグメントの抽出と、類似度評価の手法について詳しく説明する。
【００３９】
（３）本発明による判定処理の詳細

（３－１）フラグメントの抽出
自己のＵＲＬ文字列を、ＵＲＬの値としてもつリンク要素の周辺から取り出した文字情報をフラグメントとすることは既に述べたが、具体例をあげて詳細に説明する。
文字情報１２として、図８のように、
ａｂｃｄｅｆｇｈｉ
が取り出せたとする。ＨＴＭＬのタグ部分１３を除いた“ａｂｃｄｅｆｇｈｉ”をフラグメントとするのも一つの方法である。あるいは、タグ部分１３の内部の属性値１４を取り出し、“ａｂｃ_blankｄｅｆｇｈｉ”をフラグメントとするのもよい。
【００４０】
ここで重要なのは、リンク元のＷｅｂページ全体（ＨＴＭＬ文書全体）の同一性を判別するのではなく、フラグメントの同一性を判別するだけであるという点である。２つのＷｅｂページのそれぞれを記述するＨＴＭＬ文書には、同一性あるフラグメントが含まれているならば、これらのＷｅｂページの全部あるいは一部が一致すると判断しうる。このように、ＨＴＭＬ文書全体を比較することなく、リンク要素周辺から抽出したフラグメント同士の比較だけでよい。これだけでも、Ｗｅｂページの全部又は一部の異同判定に要する計算量は削減されるのであるが、次に示すようなフラグメントの３分割によって、更なる計算量の削減が図られる。
【００４１】
図９は、文字情報１２を３つのサブフラグメントに分割する例を示すものである。文字情報１２は、図８と同一である。
リンク要素周辺の文字情報１２の属性内容と要素内容を、その順番を保持して抽出し、リンク要素の前・中・後を区別して保存する。
文字情報１２の開始タグの直前の文字列ａｂｃを前フラグメント１５とし、終了タグの直後の文字列ｇｈｉを後フラグメント１７とする。前フラグメント１５と後フラグメント１７で挟まれた部分
ｄｅｆ
からタグの内部の属性値_blankとタグ外部の文字列ｄｅｆを取り出し、_blankＤＥＦを中フラグメント１６として保存する。
【００４２】
このようにして、３種類のサブフラグメントが抽出され、これらをフラグメントの同一性判定のための判定要素とする。
さらに、リンク元のＵＲＬ文字列をも判定要素の一つに加える。
なお、４つの判定要素の一つにＵＲＬ文字列を含めたのは、同じ内容を含むＷｅｂページは似通った文字列からなるＵＲＬを持つことが多いので、ＵＲＬ文字列の類似度を前・中・後フラグメントの一致・不一致判定のためのしきい値決定に利用するためである。この点については、後に詳しく説明する。
【００４３】
（３－２）文字列同士の類似性
文字列同士を比較する場合、各文字列を構成する個々の文字ごとに比較してもよい。しかし、計算量削減という観点からは、次に述べるように、元の文字列を部分文字列群に分割することが好ましい。
また、類似度の算出は、各部分文字列同士を比較して差分を求め、差分の長さと分割前の原文字列の長さとの比に基づいて行うとよい。ここで、「差分の長さ」とは、完全一致しない部分文字列の文字数をいう。「差分の長さと原文字列の長さとの比」とは、文字列同士の不一致度であり、この不一致度を１から引いたものが類似度である。
なお、差分を求めるためのアルゴリズムは、すでに公知のものがあるので、これを利用する。〔参考文献：Ｊａｍｅｓ Ｗ．Ｈｕｎｔ ａｎｄ Ｔｈｏｍａｓ Ｇ．Ｓｚｙｍａｎｓｋｉ：Ａ Ｆａｓｔ Ａｌｇｏｒｉｔｈｍ ｆｏｒ Ｃｏｍｐｕｔｉｎｇ Ｌｏｎｇｅｓｔ Ｃｏｍｍｏｎ Ｓｕｂｓｅｑｕｅｎｃｅｓ、Ｃｏｍｍｕｎｉｃａｔｉｏｎｓ ｏｆ ｔｈｅ ＡＣＭ、ｖｏｌ．２０、ｎｏ．５、ｐｐ．３５０－３５３、Ｍａｙ １９７７

〕
【００４４】
（３－２－１）剰余分割
図１０に従い、類似性評価の基本的な手法について説明する。
フラグメントｆｒ１は、Ｃ１、Ｃ２、・・・、Ｃ１０の１０個の文字が順に並んだ文字列であり、フラグメントｆｒ２は、Ｄ１、Ｄ２、・・・、Ｄ１１の１１個の文字が順に並んだ文字列であるとする。
　各文字列を、文字コードが5で割り切れる文字のところで分割し、部分文字列のリストに変換する。なお、この分割の仕方を、以下「剰余分割」という。
フラグメントｆｒ１は、○印で囲んだＣ５，Ｃ１０を剰余が０になる文字とすると、部分文字列Ｇ１，Ｇ２に分割できる。このような剰余分割をフラグメントｆｒ２についても行うと、部分文字列Ｈ１，Ｈ２とＨ３に分割できる。
【００４５】
異同の判定は、部分文字列のリスト{Ｇ１，Ｇ２}と{Ｈ１，Ｈ２、Ｈ３}同士でおこなえばよい。このときの比較演算回数は、両フラグメントの部分文字列の個数の積（＝２×３）である。文字単位で比較する場合は、両フラグメントの文字数の積（＝１０×１１）だけ、比較演算をしなければならないことを考えると、計算量が大幅に削減される。
【００４６】
剰余分割のメリットは、計算量の削減だけではない。剰余分割によれば、文字の挿入・離脱があっても、剰余がゼロになる文字は変わらない、という利点もある。もし、所定個数ずつ原文字列を部分文字列に分割するとしたならば、１文字の挿入あるいは離脱があるだけで、以後の部分文字列がすべて異なってしまう。
【００４７】
ところで、５で割ることは本質的ではない。この値が大きいほど部分文字列の個数が減り、計算量が削減されるが、類似度評価の精度が低下する。本実施形態では、５という値が適切なトレードオフとなったにすぎない。要するに、自然言語処理のように文字列の持つ意味を考えた、おおがかりな処理をすることなく、機械的に部分文字列に分割すればよいので、計算量が削減できるという点が重要なのである。
【００４８】
　ただし、部分文字列同士の比較は、１文字が異なるだけで異なる部分文字列と判定されてしまい、文字同士の比較の場合よりも精度が粗くなる。しかし、その代償として、計算量の削減というメリットが得られる。
計算量は、小規模な運用では問題にならないが、大規模な運用ではサーバ運用費にかかわる重大な要因となる。
【００４９】
（３－２－２）文字列同士の類似度の算出（基本）
　図１１に従い、評価の基本的な手法を説明する。
　フラグメントｆｒ３の原文字列ｓは、剰余分割により部分文字列ｓ１、・・・、ｓ５に分割され、フラグメントｆｒ４の原文字列ｔは、剰余分割により部分文字列ｔ１、・・・、ｔ６に分割されるものとする。
文字列ｓｔｒの長さを、ｌｅｎ（ｓｔｒ）と表記し、図１１の例では、ｌｅｎ（ｓ）＜ｌｅｎ（ｔ）とする。
　図１１に示す、単に実線で結ばれているｓ１とｔ１、ｓ２とｔ２、ｓ４とｔ５およびｓ５とｔ６は互いに同一の部分文字列であり、破線で結ばれているｓ３とｔ３、ｓ３とｔ４は異なる部分文字列であるとする。なお、部分文字列が異なるとは、一文字でも異なる場合をいい、部分文字列の全文字が一致するときは同一とする。
【００５０】
上記の例では、フラグメントｆｒ３のｓ３、フラグメントｆｒ４のｔ３とｔ４が差分として取り出される。
両フラグメントの不一致度Ｄは次の式で得られる。

Ｄ＝（ｌｅｎ（ｔ３）＋ｌｅｎ（ｔ４））／ｌｅｎ（ｔ）
この式からわかるように、フラグメントを構成する文字の個数が異なる場合は、文字数の多いフラグメントを基準に考える。図１１の例では、フラグメントｆｒ４が基準となる。なお、類似度（Ｅ）は、Ｅ＝１－Ｄである。
【００５１】
（３－２－３）文字列同士の類似度の算出（重み付けの考慮）
図１２に示すように、リンク要素からの距離によって重み付けを行う。ここで、図１２は、図１１に、重み付け（ｗ１，・・・、ｗ６）を追加したにすぎない。
同一フラグメントを構成する文字列であっても、リンク要素に近い部分は同一性成分が並び、リンク要素から遠い部分はノイズ成分が並ぶことが多いと経験的に推測できる。そのため、リンク要素からの距離を考慮して部分文字列に重み付けをする。
図１２の例では、フラグメントｆｒ４の部分文字列ｔ１の重み付け係数をｗ１、ｔ２の重み付け係数をｗ２などとする。
この場合の両フラグメントの不一致度Ｄは次の式で得られる。
Ｄ＝（ｗ３＊ｌｅｎ（ｔ３）＋ｗ４＊ｌｅｎ（ｔ４））／ｌｅｎ（ｔ）
このように重み付けをすることで、計算量削減に伴う精度の低下を抑えることができる。
【００５２】
ここで、重み付け係数は1より小さい値で、リンク要素からの距離が大きいほど小さくなる。ノイズ成分を低く評価するためである。
ただし、類似度の算出にあたって、重み付けを考慮するのは、前フラグメントと後フラグメントのみである。中フラグメントは、リンク要素の内側にあり、ノイズ成分の考慮は不要のため、重み付けはしない。
ところで、重み付け係数をいくつにするかは、実装レベルの問題にすぎない。プログラムＰＲＯＧに定数として記述してもよく、記憶部６に、パラメータとして記憶させてもよい。運用状況によって、適当な値を設定すればよい。
【００５３】
なお、上記の式において、分母をｌｅｎ（ｔ）の代わりに、
ｗ１＊ｌｅｎ（ｔ１）＋ｗ２＊ｌｅｎ（ｔ２）＋ｗ３＊ｌｅｎ（ｔ３）＋ｗ４＊ｌｅｎ（ｔ４）＋ｗ５＊ｌｅｎ（ｔ５）＋ｗ６＊ｌｅｎ（ｔ６）
としてもよい。
しかし、実用上、重み付けをしないｌｅｎ（ｔ）で十分である。
その理由は、たとえリンク要素から遠いところであっても一致しているなら、それはフラグメントが類似していることを強く示唆するからである。
分母をｌｅｎ（ｔ）とすると、Ｄの値が小さくなる、すなわち、一致していない場合の不一致評価が軽くなる。これは、類似度Ｅが大きく評価されることを意味し、同一内容を含むＷｅｂページをリンク元として重複掲載したくない、という本発明の目的にかなうものである。
【００５４】
（３－３）フラグメントの同一性の判定
図１３に従い、３つのサブフラグメントとＵＲＬ文字列の類似度を総合的に評価してフラグメントの同一性を判定する方法について説明する。
比較対象となる３つのサブフラグメント同士、およびＵＲＬ文字列同士の４種類の類似度を算出する。
予め、ＵＲＬ文字列同士の類似度に応じて、サブフラグメントの一致・不一致を判定するためのしきい値を定めておく。
図１３に示す例では、ＵＲＬ文字列の類似度が０．８を超えるときは、例えば、しきい値を０．７にセットし、０．８以下のときは、例えば、しきい値を０．９にセットする。このように、類似度が高いほど、低いしきい値をセットするのは、ＵＲＬ文字列が似通っているほどフラグメントが一致する可能性が高いからである。
なお、しきい値は、実験段階では、学習データを集めて、適宜設定する。運用段階では、

日々データが集積していくであろうから、この大量なデータに基づいて、しきい値を更新していくとよい。

【００５５】

図１３の（ケースＡ）では、ＵＲＬ文字列の類似度が０．９なので、しきい値は０．７である。前フラグメントと中フラグメントの類似度は、それぞれ０．７を上回っているので、一致しているものとする。したがって、３種類のフラグメントのうち、少なくとも２種類が一致しているので、比較対象のフラグメントは同一性があると判定される。

図１３の（ケースＢ）では、ＵＲＬ文字列の類似度が０．４なので、しきい値は０．９である。しきい値０．９を上回っているのは、中フラグメントだけである。したがって、３種類のサブフラグメントのうち、少なくとも２種類が一致という条件を満たしていないので、比較対象のフラグメントは同一性がないと判定する。

【００５６】

このように、フラグメントの同一性判定のために、４つの判定要素の類似度を別個に算定し、これらを総合的に評価した点が、本発明の大きな特徴である。各判定要素は、比較的短い文字列であり、これを部分文字列に分割して差分を求めるので、計算量削減効果がある。

また、ＵＲＬ文字列の類似度をしきい値決定のために利用することは、判定の精度を高めるものである。

【００５７】

（４）計算量削減のための他の手法

同一性あるフラグメントを効率よく見つけるためには、図３のステップＳ７からＳ８の処理において、総当り法によることなく、剰余分割された部分文字列を、検索のインデックスとして利用することで、効率よく処理ができる。これについて、以下に説明する。

【００５８】

互いに類似性の低いフラグメントの集合があるとき、その集合に含まれる任意のフラグメントは、他のどのフラグメントにもない部分文字列を持つ可能性が高い。こうした部分文字列を「ユニークな部分文字列」と呼ぶことにする。

既知のフラグメントの集合について、ユニークな部分文字列と、その部分文字列を含む既知のフラグメントへのポインタ（上記の実施形態では、抽出フラグメントテーブルのアドレス）を登録した辞書（以下、「指紋辞書」と呼ぶ）をあらかじめ作成しておく。

未知のフラグメントが発見されて、それに類似した既知のフラグメントを探すとき、部分文字列について指紋辞書をあたることで、類似性が高いと見込まれるフラグメントを、抽出フラグメントテーブルから迅速に検索することができる。

【００５９】

因みに、既存の自然言語処理では意味的な観点からの利用を前提とするため、インデックスを作成するための分割処理に、形態素解析などの計算量の多い処理を用いている。しかし、本発明では、文字情報の持つ意味的な類似性はまったく考慮しないので、機械的な剰余分割によって得られた部分文字列が、直ちにテーブル検索のためのインデックスとして利用できる。また、同様の機械的な分割であるＮグラムに比べても、少ない計算量で目的を達せられる。

【００６０】

（５）その他

本発明の同一性判定プログラムは、どのようなコンピュータ言語で記述されていてもよい。Ｗｅｂページを記述したＨＴＭＬ文書中で、当該プログラムが引用されるようになっていてもよい。

【００６１】

また、本発明の方法を実施するコンピュータは、１又は複数のリンク先サイトを管理するＷｅｂサーバであってもよいが、サイトとは無関係に、フラグメントの正規化処理サービスを行うものであってもよい。

【００６２】

さらに、入力ＵＲＬリストは、特にリンク先、リンク元という別は必須ではない。リンク先、リンク元を区別することなく、インターネット上に張り巡らされたサイト群から抽出したフラグメントの正規化処理を行う場合もあるからである。その場合、図３のステップＳ３とＳ４の処理は不要なことは言うまでもない。

【００６３】

コンピュータ１へのＵＲＬリスト３の入力の仕方は、インターネットＮを介して受信しても、管理者等によるキーボード入力等でも、ＵＲＬ情報を格納した記憶媒体から読み込むものであっても、何でもよい。

【００６４】

出力される正規化フラグメント情報４は、コンピュータ１の補助記憶装置に格納するものに限らず、コンピュータ１とは別のデータベースサーバなどに格納してもよい。あるいは、本発明のコンピュータが、他のコンピュータ（ＰＤＡや携帯電話も含む）からの依頼により、フラグメント同一性判定サービスを行うならば、自分に接続している記憶装置に格納しなくてもよい。代わりに、サービス依頼元のコンピュータに、正規化フラグメント情報４を送信したり、プリンタ出力したりしてもよい。

【００６５】

なお、本発明の、フラグメントという概念を利用した同一性判定の手法は、リンク元一覧表示という利用の仕方に限定されない。創意工夫次第で、インターネット上のさまざまな場面で活用される可能性を秘めている。

【００６６】

要は、以上のように開示された実施の形態はすべての点で例示であって、制限的なものではないということである。したがって、種々の変形が可能である。しかし、その変形が特許請求の範囲に記載された技術思想に基づくものである限り、その変形は本発明の技術的範囲に含まれる。

【図面の簡単な説明】

【００６７】

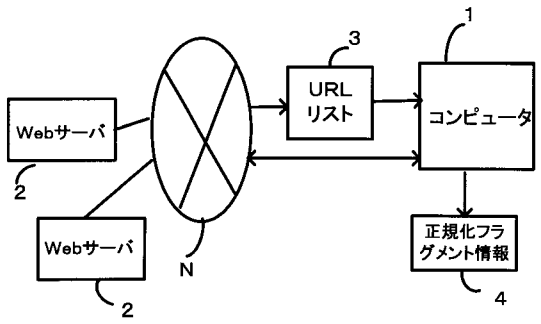
【図１】本発明の実施形態のシステム構成例を示す図である。
【図２】本発明の実施形態で用いるコンピュータのブロック図である。
【図３】本発明の実施形態の処理概要を示す流れ図である。
【図４】抽出されたフラグメントを分割して格納したテーブル構造を例示する図である。
【図５】フラグメントの正規化を説明するための図である。
【図６】フラグメントの正規化を説明するための図である。
【図７】フラグメントの正規化を説明するための図である。
【図８】フラグメントの抽出を説明するための図である。
【図９】３種類のサブフラグメントの抽出を説明するための図である。
【図１０】文字列の剰余分割を説明するための図である。
【図１１】文字列を部分文字列のリストに変換した後、差分を抽出することを説明するための図である。
【図１２】文字列を部分文字列のリストに変換した後、重み付けを考慮して差分を抽出することを説明するための図である。
【図１３】フラグメントの異同の判定の仕方を説明するための図である。
【図１４】従来技術の説明のためのシステム構成例を示す図である。
【図１５】従来技術の説明のためのリンク元一覧表示例を示す図である。_

【符号の説明】

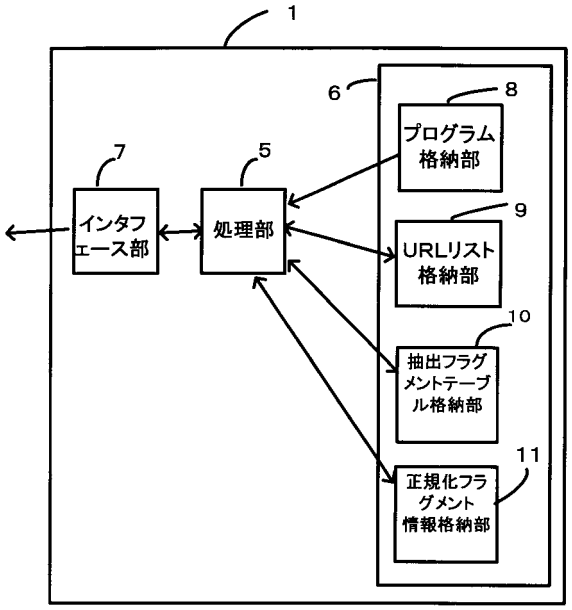
【００６８】

１　コンピュータ
３　ＵＲＬリスト
４　正規化フラグメント情報
Ｎ　インターネット

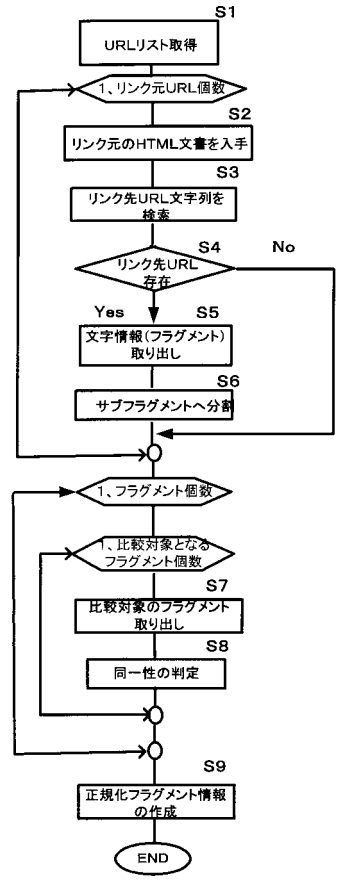
【図１】


Webサーバ
2
Webサーバ
2
N
URL
リスト
3
コンピュータ
1
正規化フラグメント情報
4

【図２】


1
6
8
プログラム格納部
7
インタフェース部
5
処理部
9
URLリスト格納部
10
抽出フラグメントテーブル格納部
11
正規化フラグメント情報格納部

【図３】



【図４】

レコードNo	前フラグメント	中フラグメント	後フラグメント	URL文字列
1	abc	_blankDEF	ghi	Web_2b.com

【図５】

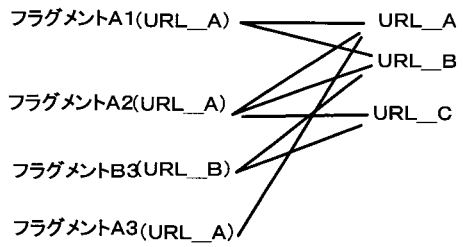
Webページのアドレス	HTML文書に含まれるフラグメント	同一性判定結果〔注〕
URL__A	フラグメントA1	○
	フラグメントA2	△
	フラグメントA3	―
URL__B	フラグメントB1	△
	フラグメントB2	○
	フラグメントB3	◇
URL__C	フラグメントC1	◇
	フラグメントC2	△

〔注〕同じ符号は、同一性があると判定されたフラグメントであることを意味する。

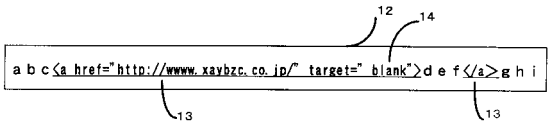
【図６】

グループNo.	同一性判定結果	フラグメント	URL
1	○	フラグメントA1	URL__A
		フラグメントB2	URL__B
2	△	フラグメントA2	URL__A
		フラグメントB1	URL__B
		フラグメントC2	URL__C
3	◇	フラグメントB3	URL__B
		フラグメントC1	URL__C
4	―	フラグメントA3	URL__A

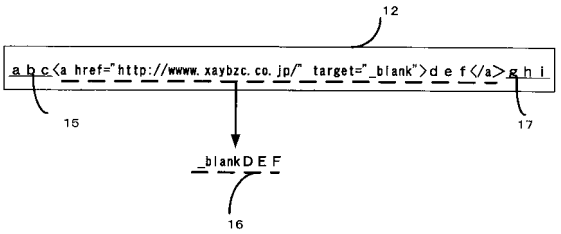
【図７】



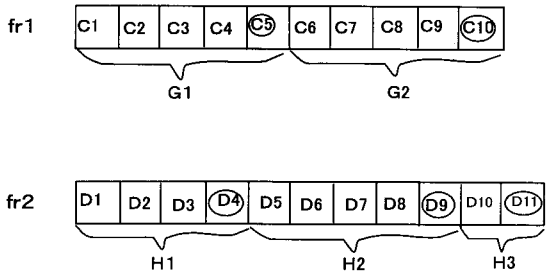
【図８】



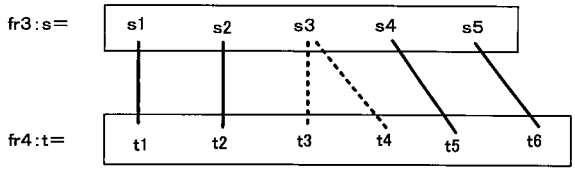
【図９】



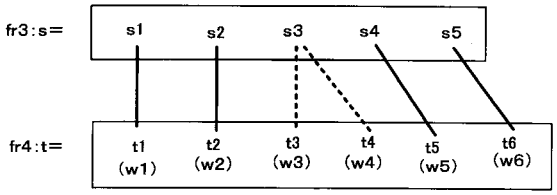
【図１０】



【図１１】



【図１２】



【図１３】

（類似度のしきい値）

URL文字列の類似度 ＞0．8 ならば、0．7

URL文字列の類似度 ＜=0．8 ならば、0．9

（ケースA）

判定要素	類似度	一致(○)、不一致(×)
URL文字列	0．9	しきい値は、0．7
前フラグメント	0．75	○
中フラグメント	0．8	○
後フラグメント	0．5	×

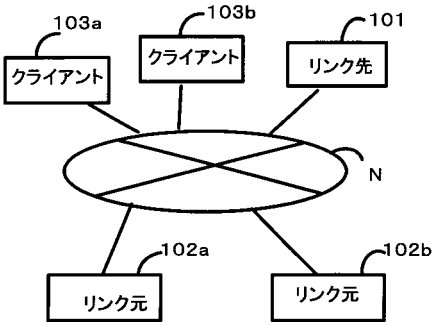
⇒ 判定結果：同一性あるフラグメント

（ケースB）

判定要素	類似度	一致(○)、不一致(×)
URL文字列	0．4	しきい値は、0．9
前フラグメント	0．8	×
中フラグメント	0．95	○
後フラグメント	0．7	×

⇒ 判定結果：同一性のないフラグメント

【図１４】



【図１５】