WWW文書における属性情報抽出の試み

清水　奨，神林　隆，佐藤　進也，ポールフランシス
NTTソフトウェア研究所

1996年10月25日

WWW上で提供される膨大な情報に対して、検索支援のニーズが高まっている。しかし、現在の検索システムの多くはそれぞれが収集した情報を個別にデータベース化しており、互換性は考えられていない。このため検索システムの数だけ収集ロボットが作られるといった非効率性が指摘されている。複数の検索システムの協調動作を実現するためには、収集した情報が持つさまざまな属性情報(文書タイプ、言語その他)を取り出し、共通に利用できるようにすることが重要である。本稿では、WWWで提供される文書を対象とし、属性情報を抽出するためのフレームワークについて述べる。属性の抽出を文書タイプの識別、言語の識別をはじめとする幾つかの工程にわけ、著者らが開発中の検索システムIngridにおける実装について述べる。また各々の工程における技術的な問題点と解決のためのアプローチを示す。

1　はじめに

WWW上に存在する多種多様な情報に対する検索のニーズが高まっている。すでに従来のデータベース技術を応用したさまざまな検索システムが稼働しており[1]、広くサービスを提供している。

一方、現状の検索システムには課題も多く残されている。検索の品質やユーザインターフェースに関する課題の他、最も大きな課題の一つは、それぞれの検索システム間に互換性がなく、何種類もの情報収集ロボットが同じ様な目的のために動作している点である。

WWW上に存在する情報の内容は非常に多岐に渡るため、これらを直接扱う場合、各検索システムのインデクシング手法に深く結びついた処理にならざるを得ない。そこで、元の情報を直接扱う代わりに最低限必要な情報だけを抽出し、検索対象とすれば互換性を持たせる事ができる。

このような抽出情報(以下では属性情報と呼ぶ[1])のフォーマットとしては様々なものが提案されており[2][3][4][5]、例えば異なる検索システムの協調を念頭においたSTAIRS[6]ではZ39.50[7]で定義された属性情報の一部をSOIF[5]形式で取り扱う事を提案している。

しかしWWW文書を対象とした属性情報の標準化の動きは始まったばかりで、自動的に属性情報を取り出すためのソフトウェアは少ない[8]。

本論文では、属性情報を自動的に抽出するための基本的なフレームワークについて述べ、著者らが開発中の検索システムIngrid[9]における実装例に基づき、段階を追って自動抽出の手法を議論する。

以下二章では既存のフレームワークと提案するフレームワークについて述べる。三章では提案フレームワークを細分化し幾つかの工程に分類し、実装例にもとづく議論を行う。四章で各工程における技術的な問題点と解決のための現状のアプローチについて述べ、五章で結びとする。

Experimental Report of automatic meta-info extraction from WWW documents,Susumu Shimizu, Takashi Kambayashi, Shin-ya Sato and Paul Francis.

NTT Software Laboratories,
M9-311A 3-9-11 Midoricho Musashino TOKYO Japan
Email: shimizu@ntt-20.ntt.jp

[1]通常metadataと呼ばれるが、metadataは画像や物理実験データなどに対しても使われるので、ここでは属性情報という言葉を使った。

2 属性情報抽出フレームワーク

インタネット上の情報を対象にした既存の情報抽出システムとしては、米国コロラド大の Harvest[10] で SOIF を自動生成するのに用いられた Essence[8] が代表的なものである。

図 1[8] に、Essence のフレームワークを示す。

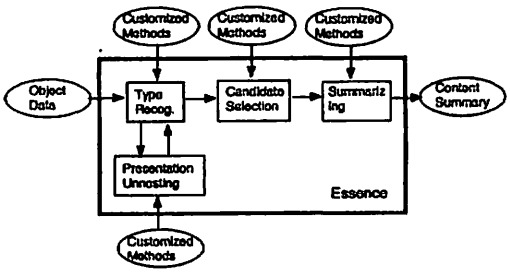

図 1: Essence におけるデータの流れ

Essence のフレームワークは、各フェーズにおけるカスタマイズモジュールを容易に組み込めることを特徴としており、文書タイプの認識 (Type Recognition)、テキストなどのインデクス可能なファイルの選択 (Candidate Selection)、属性情報の抽出 (Summarizing) の三段階にわかれている。図の Presentation Unnesting は tar アーカイブやシェルアーカイブ、ディレクトリ、圧縮ファイルなどをフラットなファイル集合に展開する操作を行う。

Essence のフレームワークはシンプルなため、実装を行う場合の参照モデルとしては不足が見受けられる。例えば、エンコーディングや言語の検出をどの段階でやるか、HTML 化されたオンラインマニュアルのように、複数の小さなファイルが全体として構造を持っている場合はどうするかなどである。これらの機能を図 1 の Customized Methods で対処すると、さまざまな実装が現われて効率的でない。

著者らが提案するフレームワークを図 2 に示す。従来との違いは、次の二点である。

- 言語検出部 (Language Detector) の導入
  言語の検出と言語依存部への振り分け
- フラグメント検出部 (Fragment Construction) の導入
  複数ファイルの関係情報の抽出

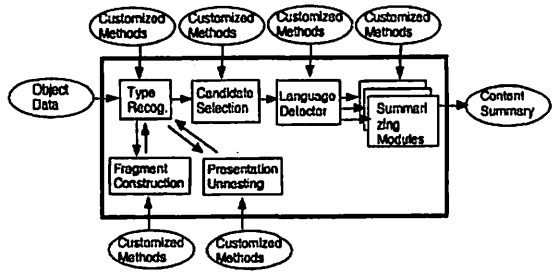

図 2: 提案フレームワーク

これらの部分を導入することにより、次第に流通量が増えつつある多言語で記述された情報や、WWW 上の情報に特徴的な細分化されたファイル群をまとめて扱うことが可能となる。以降、このフレームワークに基づく工程を順に述べる。

3 属性情報の抽出工程

本章では著者らが開発中の属性情報抽出プログラム Ingrid Publisher[11] を実装例とし、属性情報の抽出を工程順に議論する。

3.1 文書タイプ認識

文書タイプ認識 (Type Recognition) は Essence と同様、原則としてファイル拡張子の名前づけ規則と Unix の file コマンドと同様な特定のテキストパターンに基づく推測を用いる。例えば、.txt であれば可読テキストであるとか、From ではじまり Received: や Date: が行頭にあればメールであるといった推測である。Ingrid Publisher が認識できる文書タイプは現在 12 種類であり、Essence のようにモジュールを追加することで増やすことができる。

ただし、このような単純な手法ではうまくいかない例として以下のものがある。

- HTML だが、タグを取ると別のタイプ (例えばメール) になるもの。
- メールだが、メッセージボディは別のタイプ (例えば HTML) である。

現在の実装では、前者であれば HTML、後者であればメールと判断されるが、WWW ロボットが収集する

情報には、前者に属するものが多い。このためHTMLについては、タグを取った結果を再度タイプ認識プロセスに渡し、必要に応じて属性情報を抽出する必要がある。また後者についてはMIME[12]ヘッダのエントリを工夫する案[13]がある。

アーカイブ処理 (Presentation Unnesting)

この工程はEssenceと同様、compressやgzipで圧縮されたtarアーカイブを展開し、インデクス可能なテキストファイルやPSファイルだけを抜き出す。抜き出されたファイルから後の工程で抽出された属性情報は最終的にマージされる。ただし、この処理によってアーカイブを普通のテキストファイルと同列に扱う場合、閲覧時のインターフェイスを考慮する必要がある。閲覧時に属性情報だけを見せる枠組みがないと、オリジナルの閲覧をする度にアーカイブを転送しなくてはならなくなる。即ち、HTMLファイルを主体とする場合など、属性情報よりもオリジナルの閲覧が重視されるような場合はアーカイブをそのまま処理するのは適切でない。

フラグメント検出 (Fragment Construction)

WWW上の文書では、多数の小ファイルが一群をなして何らかの情報を伝えることが多い。オンラインマニュアルやHTML化された教科書、メーリングリストのアーカイブ公開、など多くの例がある。こうしたファイル群は、個々のファイルを扱う他に、一群としても扱えることが望ましい。Ingrid Publisherでは、HTMLのリンク元をParentLinkという属性で保存している[2]。

例えば双方向リンクや、上位レベルへ戻るリンクを削除して図3のような関係があったとする。ParentLinkの属性があれば、あるキーワードで検索したときに図のAからDの4つがヒットしても、(ParentLinkのない)Aから読み進めることができる。また、検索結果の表示などに際して、Aだけを表示するような制御ができる。

3.2 言語検出

WWW上にはさまざまな言語の情報が存在するので、理想的には、これらを自動的に検出できる仕組みが望まれる。TITAN[14]など自動検出の試みもあるが、まだ課題も多い。工程としては次のようになる。

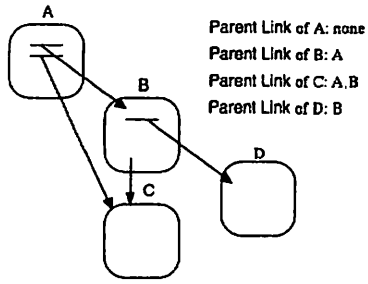

図3: フラグメントの検出

エンコーディング検出

言語検出の前半はエンコーディングの検出である。ただしあらかじめ候補を絞り込まない限り、この段階で決定的に検出できるエンコーディングは少ない。この場合は後半の言語検出部の助けが必要である。例えばJIS X 0208-1990のEUC圧縮フォーマット(主に日本語)とKS C 5601-1992の8bitコード(主に韓国語)とGB 2312-80の8bitコード(主に中国語)はどれもA1-FEの領域のデータストリームであり、ISO-2022のようなユニークなエスケープシーケンスに頼った自動判別などはできない[3]。この場合、この段階ではA1-FEの領域にあるデータの比率からEUCエンコーディングであることが推測できるにすぎず、次の段階で文字パターンの比較をし、たとえば「、(A1A2)」や「。(A1A3)」が多いから日本語であろうと判断することになる。

なおIngrid Publisherの現在の実装では、エンコーディングの検出は外部コンバータ[4]に頼っている。

言語検出

エンコーディングの検出が終わると、言語検出を行う。例えば、仮にエンコーディング検出部でISO-8859-1 Latin1の文字セットがそのままのコードで使われていると判断できたとしても、デンマーク語からスウェーデン語まで14ヵ国で使われているため検出が必須である。また、ISO-2022-JPエンコーディン

[2]なおversion0.1ではこの属性は人間が指定する必要がある

[3]各国のエンコーディングについては[15]に詳しい。

[4]現在はMule付属のcocoを用いている

グで JIS X 0208-1983 文字セットを使って中国語や、ロシア語を記述する例もある[5]。

Ingrid Publisher で用いている言語検出ルーチンは図 4に示す単純な方法である。

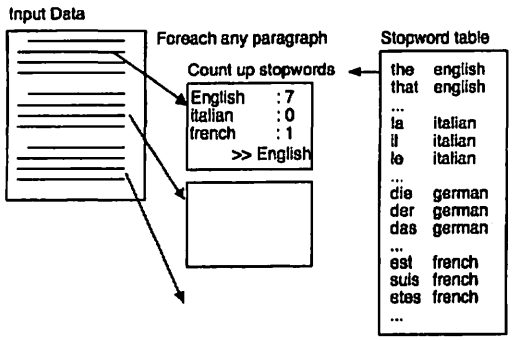

図 4: 言語の検出

即ち、あらかじめ言語特有のストップワード (最頻出語) 数十語からなるリストを作っておき、段落ごとに処理してマッチしたストップワード数を計数する。ストップワードが一つも現われなかった場合は、UNKNOWN として判断を保留し次の段落に進む。全体を通じて判断できなかった場合はエラーとする。

実際の評価はまだ行なっていないが、エンコーディング候補を決めコーパス (品質のそろった文書集合) を用いて言語検出を行う方法や、N-gram に基づく手法 (各単語の先頭数文字、もしくは末尾数文字の出現頻度をコーパスにより統計データ化し、判別に利用) に比べると精度が落ちることが指摘されている [14]。しかし、ある言語の Native Speaker であれば数十語からなるストップワードのリストを作ることはコーパスを作るよりも容易であるため、拡張性を重視する立場からストップワードによる実装を行った。

3.3 言語依存の処理

言語を検出したあとは、各々の言語に応じた処理を行う。言語に応じた処理は属性情報全体に関わるが、ここで説明するのは主にキーワードの処理である。

[5]こうした例は今後多言語環境が普及すればなくなるものと思われる

キーワード抽出

Ingrid Publisher では、キーワード抽出に Salton[16] の tf.idf 法を使っている。まず適当な文書集合を使って各単語の document frequency(df) を求めておく。入力文書は単語に分解 (日本語については JUMAN[17] を使用) され、その文書中での頻度 (Term Frequency, tf) が計算される。その単語の df がもしあれば、その逆数を tf にかけて重みとする。tf.idf 法自体は言語に独立であるが、単語分解の手法と df を作るための文書集合が言語依存であるため、キーワード抽出部は言語ごとにモジュール化されている。このようにして計算された重みの大きいものから 100 程度のキーワードを抽出しておく。

揺らぎ処理

情報検索における検索漏れを少なくする (recall 率をあげる) ために、得られたキーワードに対しさまざまな処理を加える。我々の実装では、英語では stemming の処理 [16] と、キャピタライズの対応を行い、日本語では全角／半角の吸収と活用語尾の吸収を行っている。

最後に、Ingrid Publisher が処理した結果 (取り出した属性情報) の例を以下に挙げる。また、各項目の説明を付録に記述した。

```
Ig-pubVers: NTT-SOFTLAB-1.0ALPHA
Ig-pubBy: shimizu@ingrid.org
Ig-pubDate: Wed, 18 Sep 1996 06:49:55 GMT
Ig-resTitle: Ingrid Global Web Discovery (Japanese)
Ig-resDate: Fri, 6 Sep 1996 06:01:41 GMT
Ig-resLanguage: Japanese
Ig-resType: text/html
Ig-resParentLink: http://www.ingrid.org/
Ig-resTerms: 182
Ig-resURL: http://www.ingrid.org/index-j.html
Ig-resBestTerms: 20
Ig-resTermComb:
 { http://www.ingrid.org/index-j.html:url:ingrid }
 { ingrid.org:dns:ingrid } { www.ingrid.org:dns:ingrid }
 { ingrid || Ingrid }
 { 情報検索システム || 情報検索システム }
 { プロジェクト || プロジェクト } { 広域 || 広域 }
 { web || web } { スケーラビリティ || スケーラビリティ }
 { japanese || Japanese }
 { インフラストラクチャ || インフラストラクチャ }
 { global || Global } { discovery || Discovery }
 { 基礎 || 基礎 } { 運用 || 運用 } { 分散 || 分散 }
 { 情報 || 情報 } { 可能 || 可能 || 可能な }
 { 目標 || 目標 } { 環境 || 環境 } { 研究 || 研究 }
 { サービス || サービス }
 { アプリケーション || アプリケーション } { 検索 || 検索 }
 { メイリングリスト || メイリングリスト }
 { 研究プロジェクト || 研究プロジェクト }
 { ソフトウェア || ソフトウェア } { 開発 || 開発 }
 { リソース || リソース }
 { フリーソフトウェア || フリーソフトウェア }
 { ホワイト || ホワイト } { コミュニティ || コミュニティ }
 { ntt || NTT } { ゲーム || ゲーム }
 { アナウンス || アナウンス } { english || English }
```

```
{ イエローページ || イエローページ }
{ インタフェース || インタフェース }
{ ライブラリ || ライブラリ }
{ フィルタリング || フィルタリング } { api || API }
{ ダウンロード || ダウンロード } { mud || MUD }
{ ユーザ || ユーザ } { プロトコル || プロトコル }
{ ページ || ページ }
{ ソフトウェア研究所 || ソフトウェア研究所 }
{ トピック || トピック } { 検索エンジン || 検索エンジン }
{ 要旨 || 要旨 } { 実験 || 実験 } { 公開 || 公開 }
{ 詳 || 詳しい || 詳しく } { 潜在 || 潜在 }
{ 追加 || 追加 } { 忠告 || 忠告 } { 英語 || 英語 }
{ 項目 || 項目 } { 利用 || 利用 } { 質問 || 質問 }
{ 会話 || 会話 } { 参加 || 参加 } { 設計 || 設計 }
{ 成長 || 成長 } { 自身 || 自身 } { 広告 || 広告 }
{ 文書 || 文書 }
Ig-resInitText: Ingrid Global Web Discovery
(Japanese). Ingrid: 広域情報検索システム
NTT ソフトウェア研究所
Ingrid研究プロジェクト　英語 (English) のページ
```

Ingridプロジェクトは、情報が広域に分散している環境下で、スケーラビリティを損なわずに運用可能な、情報検索システムのインフラストラクチャを作り上げることを目標とした基礎研究プロジェクトです。

私たちは、皆さんが持っているwebリソースを、Ingridのインフラストラクチャに追加して検索できるようにする、フリーソフトウェアを設計、研究、開発しています。このソフトウェアは、プロトコル・インタフェース、APIおよびそのライブラリが公開されているので、Ingridのインフラストラクチャを利用したアプ...

4 問題点と現状のアプローチ

前章の各フェーズにおける問題点と解決のためのアプローチについて述べる。言語依存の処理における諸問題は自然言語処理の分野で研究が続けられているものと共通であるのでここでは触れない。

文書タイプ認識における問題

文書タイプの認識における問題点は、ある文書タイプのフォーマットが他の文書タイプを内包する場合の検出である。

現在のアプローチには、MIMEのContent-typeヘッダやMultipartがある。しかしMIMEヘッダは一つのファイルに複数のタイプを含めることはできても、一つのタイプが実は他の情報を包むのに使われている場合には対処できない。

したがって、3章で例外にあげたような、HTMLでマークアップされたメールメッセージなどについてはMIMEだけに頼らず、独自に対応する必要がある。例えば入力ストリームの文書タイプ候補をヒント情報としてあらかじめ設定ファイルに持たせる手法や、HTMLのコメント部にマークアップする前の情報を置き、属性情報はそこから取り出すといった手法がある[6]。

[6]MIMEのレベルでEmbedded-typeとかReal-Content-typeの様なヘッダを定義できれば、すっきりと解決できる可能性がある。

フラグメント検出における問題

検索サービスの結果のうち最も好ましくないものの一つは、同じタイトルですこしづつ内容が違うものが並ぶような場合である。このためフラグメントの問題は重要である。しかし、あるファイル群をまとめて扱うべきかどうかは内容に絡む問題であるため、完全に自動化することは難しい。

現状のアプローチは、以下のような判断基準を導入してフラグメント検出を行うというものである。

- 同一サイト内のリンク
- URLの深さの差が1レベル以内
- 先頭の数単語が共通

この判断基準は、経験的にオンラインマニュアルによく用いられるフォーマットに基づいている。例えばLaTeX2HTML[18]で変換されたファイル群に適用できる。

言語検出における問題

言語検出における問題は、特に短い文書において自動検出が難しい点である。このため現状では、入力ストリームにおける言語の候補を絞る手法[14]をとるのが一般的である。

WWW上の文書を対象とした言語検出のためのアプローチは、プロトコルとHTMLの両面から進められている。HTTP1.1[19]ではContent-Languageヘッダを使ってサーバ側で言語の指示ができるようになり、Accept-Languageリクエストヘッダで特定の言語を指定してサーバにリクエストできるようになった。

またHTML-I18NのdraftではクライアントHTML-I18N のdraft[20]では、クライアント側の参照モデルに触れている。このモデルでは、サーバがHTTP1.1のContent-Languageヘッダをつけない場合、ファイル中の<HEAD></HEAD>内に記述する<META HTTP-EQUIV = "Content-Type" CONTENT = "text/html ; charset = ISO-2022-JP">という記述でこれを補い、これもなければ、クライアントが表示に使っているデフォルトのエンコーディングと言語を用いる提案がされている。これらの提案の実装と普及が待たれる。なお実装例は[21]にある。

5 終わりに

本論文では、WWW で提供される情報を対象とする検索システムの間で共通に利用可能な属性情報抽出を目的として、抽出のためのフレームワークと実装例について述べた。また、属性情報の抽出における現状の問題点とアプローチを整理した。

本稿で述べたようなソフトウェアは、多くのコンポーネントからなる複雑なものになりがちである。紹介した実装では、各工程それぞれにおいて改良すべき点が数多く残されているが、言語に独立な部分と言語に依存する部分を分離し、他の言語モジュールを容易に追加できるようにしたことが大きな特徴となっている。

今後は 4章にあげた問題点の解決をめざし実装を続けていく。なお本論文で紹介したソフトウェアは http://www.ingrid.org/publisher/download.html からダウンロードすることができる。

参考文献

[1] 清水, 日本の *Search Engine* のリスト, http://www.ingrid.org/w3conf-bof/search.html

[2] Stuart Weibel, et al., *OCLC/NCSA Metadata Workshop Report*, The March 1995 Metadata Workshop, available at http://www.oclc.org:5046/ conferences/metadata/dublin_ core_report.html

[3] R.V.Guha, *Meta-Content Format*, Apple Computer Technical Draft, http://www. atg.apple.com/go/ProjectX/mcf.html

[4] C. Weider, J. Fullton, S. Spero., *Architecture of the Whois++ Index Service*, RFC1913, available at ftp://ds.internic.net/rfc/rfc1913.txt

[5] Harvest User's Manual,*The Summary Object Interchange Format (SOIF)*, http://harvest.cs.colorado.edu/harvest/user-manual/node151.html

[6] Kevin Chang, Hector Garcia-Molina, et al., *STAIRS ver1.0*, http://www-diglib.stanford.edu/diglib/WP/PUBLIC/DOC82.html

[7] ANSI/NISO, *Z39.50-1995 Information Retrieval*

[8] D. R. Hardy, M. F. Schwartz. *Customized Information Extraction as a Basis for Resource Discovery.*, ACM Trans. on Computer Systems(TOCS) Vol.14 Number.2

[9] ポール フランシス et, al. 次世代情報検索インフラストラクチャ *Ingrid*, NTT R&D Vol.45 No.2 1996 pp159–166

[10] C. Mic Bowman, et,al.,*The Harvest Information Discovery and Access System*, in Proc. of the 2nd Intl. WWW Conf. pp763-771, Chicago 1994

[11] Ingrid Research Group, *Publisher User's Manual*, http://www.ingrid.org/publisher/manual/index.html

[12] N. Freed, et,al.,*MIME(Multipurpose Internet Mail Extensions) Part One*, RFC1521, will be obsoleted by InternetDraft draft-ietf-822ext-mime-imb-07.txt both available at ftp://ds.internic.net/

[13] J. Palme,*MIME E-mail Encapsulation of Aggregate HTML Documents(MHTML)*, InternetDraft, draft-ietf-mhtml-info-03.txt available at ftp://ds.internic.net/

[14] 菊井他, インターネット情報ナビゲーションにおける多言語機能, 自然言語処理の応用に関するシンポジウム,1995 情報処理学会

[15] Ken Lunde,*Understanding Japanese Information Processing*, O'Reilly & Associates, Inc. 1995(邦訳「日本語情報処理」ソフトバンク刊)

[16] G. Salton, *The Smart Retrieval System — Experiments in Automatic Document Processing*, Prentice-Hall, Inc. 1971

[17] Matsumoto,Y. *Japanese Morphological Analysis Sytem JUMAN Manual, Ver1.0*, Nara Institute of Science and Technology, 1993

[18] N. Drakos., *From Text to Hypertext: A Post-Hoc Rationalisation of LaTeX2HTML*, in Proc. of the 1st Intl. WWW Conf. in Geneva,CERN, May 1994

[19] R. Fielding, J. Gettys, et,al.,*Hypertext Transfer Protocol – HTTP/ 1.1*, IETF Internet-Draft, available at ftp://ds.internic.net/internet-drafts/ draft-ietf-http-v11-spec-07.txt

[20] F. Yergeau, et,al., *Internationalization of the Hypertext Markup Language.*,IETF Proposed Standard,available at ftp://ds.internic.net/internet-drafts/ draft-ietf-html-i18n-05.txt

[21] Vancouver Webpages, *Sample Pages for Various Character Sets.*, http://vancouver-webpages.com/multilingual/

付録:Ingrid Resource Profile Entries

本論文で述べた属性情報を Resource Profile(RP) と呼んでいる。ここでは、その項目について述べる。なおデータのエンコーディングには Mule 内部コードを採用している。

Ig-pubVers	RP を作ったソフトウェアの ID
Ig-pubBy	RP を作った人の連絡先 (メールアドレスなど)
Ig-pubDate	RP を作った日付: 書式は RFC822 とほぼ同じだが、西暦に 4 桁使い、時間帯は GMT に統一する。
Ig-resTitle	リソースのタイトル
Ig-resDate	リソースの日付: 書式は pubDate と同じ。
Ig-resLanguage	リソースの言語: ISO639 の英語による言語表現の正式名
Ig-resType	リソースのタイプ。表現は MIME 準拠
Ig-resParentLink	HTML の場合のみ使用、リンク元の URL を保持
Ig-resTerms	リソースに含まれる単語数
Ig-resURL	リソースの URL
Ig-resBestTerms	特に重みの大きい単語の数
Ig-resTermComb	抽出した単語: 書式は {term1 || term2} のように、単語の揺らぎをひとまとまりにする。
Ig-resInitText	リソースの出だし、上限 1KB