SlothLib: Webサーチ研究のためのプログラミングライブラリ

SlothLib: A Programming Library for Research on Web Search

大島 裕明† 中村 聡史‡
田中 克己‡

Hiroaki OHSHIMA Satoshi NAKAMURA
Katsumi TANAKA

本論文では，Webサーチ研究におけるソフトウェア試作時のコストを低減するためのライブラリSlothLibについて述べる．Webサーチに関するソフトウェアの試作を行う際には，主たるアイデアの実装以外に，Web検索APIサービス，Web上の情報取得と文字コード判別，既存のクラスタリングアルゴリズム，自然言語処理ツールなどの利用方法を習得する必要がある．このような機能を部品として簡単に再利用できるようにし，Webサーチに関するソフトウェア試作を迅速化することがSlothLibの目的である．さらに，既存のWeb検索エンジンを自身の目的にカスタマイズすることもSlothLibの目的の一つである．SlothLibでは，各種機能やアルゴリズムのうち，類似のものには極力共通インターフェースを持たせて実装しており，あとで機能を入れ替えることが容易である．また，SlothLibで実装された機能をGUI上で利用できる「ブロックプログラミング環境」を開発した．これは，ビジュアルインターフェース上でブロックを接続することで，ユーザがアイデアを試せるシステムである．

We developed a programming library, called SlothLib, for research especially on Web search. The purpose of SlothLib is to achieve a rapid prototyping for realizing ideas on Web search and for customizing conventional Web search softwares. It reduces costs for prototype implementation. Conventionally, researchers need to study several software issues, such as Web search APIs or national language processing tools before implementing their own ideas. SlothLib provides common interfaces for similar services. It is easy for researchers or developers to modify or change implementations for some functions. We also developed a visual programming environment for SlothLib. Each function of SlothLib is represented as a programming block on the environment, in which users can select and connect those programming blocks for implementing their prototypes. This leads to a visual and rapid prototyping environment, by which researchers or developers can easily test their ideas.

1. はじめに

Web サーチが利用される機会は増える一方であり，Web サーチの研究がますます重要になってきている．Web サーチの研究における試作システムの実装では，Web 検索エンジンベンダーである Google，Yahoo!，MSN などが各々提供する API や，各種の自然言語処理のツール，クラスタリング手法などにおいては既存のアルゴリズムを利用することが多い．これらの API やツールの利用法の習得，既存のアルゴリズムの実装にかかるコストは無視できないほど大きい．また，システムでそれらを利用した部分がうまく分離されていない場合には，システムの改良時に別ツールや別アルゴリズムを試すためにはその変更に大きなコストをかけることになる．

本論文で述べる SlothLib は，Web サーチ研究におけるソフトウェア試作時のコストを低減させることを目的とするプログラミングライブラリである．SlothLib は新しい機能をライブラリ化して提供するものというよりも，むしろ，現在提供されている様々なツールや既存のアルゴリズムを網羅的に利用可能な環境を提供するためのものである．Web 検索 API サービス，Web 上の情報取得とそれに伴う文字コード判別，既存のクラスタリングアルゴリズム，自然言語処理ツールなどが部品として利用可能である．各種機能やアルゴリズムのうち，類似のものには極力共通インターフェースを持たせて実装しており，試作システムの変更時などに機能を入れ替えることが容易である．また，用意された部品を組み合わせることによって既存の Web 検索エンジンを自身の目的にカスタマイズすることも SlothLib の目的の一つである．

さらに，SlothLib で利用可能な機能を GUI 上で組み合わせて利用できる「ブロックプログラミング環境」を作成した．Web 検索や特徴ベクトル生成といった機能がブロックとして表現され，それらを接続することで簡便なプログラミングを行うことができ，アイデアをすぐに試すことが可能である．

SlothLib は現在，C#で実装されており，Windows の.NET Framework 環境において利用することができる．

以下，2 節で関連研究について，3 節で SlothLib が持つ機能について，4 節でブロックプログラミング環境について，5 節でまとめと今後の課題について述べる．

2. 関連研究

2.1 基盤ライブラリと Web 検索を横断的に利用する環境

プログラミングライブラリは数多いが，その多くはある特定の機能を利用してもらうためのものである．SlothLib は，むしろ，メタレベルのライブラリであり，他のライブラリを統合的に利用するためのものである．ここでは，共通基盤としてのライブラリや，数多く存在する Web 検索を横断的に利用するための環境を提供する研究について紹介する．

阿部ら[1]は様々なメディアに対するアノテーション処理を統合的に扱うためのライブラリを構築している．アノテーション処理に特化しているライブラリと見ることができるが，文書要約や文書の重要語抽出など，自然言語処理の基本的な処理を行うことができ，自然言語処理を専門としないものにとっては自然言語処理のライブラリとしての利用価値も高い．Mendelzon[2]らは，Web 検索エンジンを横断的に検索することができる WebSQL という言語を開発した．言語を提供することによって，異なるインターフェースを持つ検索エンジンを統一的に扱うことができるようにしようとしている．Kistler[3]らは，WebL という言語を開発した．これは，Web 文書の取得，サービスの統合，データの抽出，Web 文書の作成など，様々な Web における行動を記述することができ，Web 行動そのものをプログラミングしようとしている．

† 学生会員 京都大学大学院情報学研究科社会情報学専攻
ohshima@dl.kuis.kyoto-u.ac.jp
‡ 正会員 京都大学大学院情報学研究科社会情報学専攻
{nakamura, tanaka}@dl.kuis.kyoto-u.ac.jp

2.2 視覚的プログラミング環境

視覚的なプログラミング環境は，現在，様々なものが開発されている．いくつかについて紹介する．

Microsoft の SQL Server 2005 には，BI Development Studio[4]という ETL 処理を視覚的なインターフェースで行う環境がある．Ito ら[5]は，Web アプリケーションに対してラッパーを作成し，それらを組み合わせたサービスを作成できるような環境を開発した．Tanaka ら[6]は，パッドというデータや機能を格納した部品を視覚環境上に配置して合成することによって新たな機能を実現するアプリケーションを開発している．松村ら[7]は 2006 年度上記の IPA 情報処理推進機構の未踏ソフトウェア創造事業において，セマンティック Web サービスの連携を視覚的に行うための PatchService というサービスの開発を行っている．現在数多くあるセマンティック Web サービスのそれぞれに，ラッパーを作成して連携可能にし，視覚的な環境において組み合わせることができる．Microsoft Research ASIA の Ma ら[8]は Web Studio という環境を作成しており，そこでは，Web 検索機能などを利用したアプリケーションを視覚的に作成することができる．Max[9]は音楽信号処理の視覚的プログラミング環境である．音楽信号処理では各種処理を並行的に行うことが多く，また，処理の流れとデータの流れがほぼ等しいため，視覚的環境との親和性が高く，広く利用されている．

3. SlothLibの機能

3.1 Web 検索サービス

現在,様々な検索エンジンの API が公開されている.また，Blog 検索エンジンでは，RSS が提供されるものがあり，それらもプログラムから利用することができる．それぞれの検索エンジンは SlothLib 上でクラスとして表現され，それらは全て共通のインターフェースを実装している．

現在 SlothLib では，下記のように一般的な Web ページ検索，Blog 検索，動画検索，質問回答システム検索，商品検索など，多岐にわたる Web 検索エンジンが利用可能である．

- **Web 検索**（Google，Yahoo!，MSN）
- **Blog 検索**（goo，livedoor）
- **動画検索**（Yahoo!，Youtube）
- **質問回答システム検索**（はてな，goo，Yahoo!）
- **その他**（価格.com 商品検索）

これらのいくつかでは共通インターフェースで扱う以上の情報が提供されている．例えば，Youtube API では，検索された動画に付加されているタグや，これまで何回閲覧されたかという情報があるが，これらを含む全ての情報はそれぞれの結果のクラスに保持されており，ユーザがこれらの情報を利用したい場合には利用可能であるようにしている．

いくつかの API や Web サービスには，利用時にクレジット表示を義務づけるものがある．例えば，Yahoo!を利用したプログラムにはクレジット表示が義務づけられている．そこで，Yahoo!のクレジット表示が行えるコントロールを検索のためのクラスとは別に提供している．

3.2 Web 情報収集

Web から多くの文書や画像データなどを取得することも多い．多くのファイルを Web から収集する必要がある場合は，たいていの場合はマルチスレッドを利用した方が速い．SlothLib では，URL を与えるとローカルコンピュータにファイルとして保存する機能をもつクラスとして，シングルスレッドのものとマルチスレッドのものの 2 つを提供している．

3.3 各種文書読み込みと日本語文字コード判別

文書には，テキストファイル，PDF，MS-Word，Power Point，一太郎，HTML など，様々な形式が存在する．このような様々なフォーマットのファイルを読み込むためには専用の API やツールを利用する必要がある．このようなことを行うツールはいくつか存在し，例えば，xdoc2txt は非常に多くのフォーマットの文書から，その内容のテキストだけを抽出するツールであり，Namazu や Meadow2 といったソフトウェアでテキスト形式以外のファイルを扱うために利用されている．SlothLib では xdoc2txt を利用するクラスを用意して，様々なファイルの内容を読み込むことができるようにしている．

日本語のテキストファイルを読み込む場合には，文字コードの自動判別も必要となってくる．文字列コードを判定するものとしては，TxtEnc や NKF 等を利用することが考えられる．現時点では TxtEnc を利用した文字コードの判別のためのクラスが用意されている．これらのツールを組み合わせたクラスも用意されており，SlothLib ユーザは様々な種類の文書ファイルから，文書タイプや文字コードなどを考えずに文書読み込みを行うことができる．

3.4 自然言語処理関連

ベクトル空間モデルでは，文書とクエリを特徴ベクトルとして表現し，ベクトルどうしの類似度を計算することによって文書検索を行う．システムとしては SMART[10]が著名である．日本語の文書を特徴ベクトルで表現するためには，文章を形態素に分解することがしばしば行われる．

また，英語の場合でも形態素解析を行ったり，形態素解析は行わないまでもステミングという語幹の抽出を行ったりすることがある．そのような，各種の自然言語処理のためのツールは様々なものが公開されている．SlothLib では日本語形態素解析器の茶筌[11]や MeCab[12]などをクラスで表現し，それらの結果に対して，大文字小文字や半角全角をそろえる，ストップワードを排除する，特定の品詞の語のみを抜き出す，英単語に Porter Stemmer によるステミングを行う，といったフィルタをかけることができ，特に特徴ベクトル生成において必要な機能が多数用意されている．

3.5 特徴ベクトル

文書の特徴ベクトルの生成には各種手法が存在する．文書の特徴ベクトルを作成する際には一般的に，局所的重み，大局的重み，正規化の 3 つについて考える必要がある．局所的重みとは，ある文書そのものにどのような語がどの程度出現しているかという特徴のことである．文書に語が出現するか否かによる 2 進重み，文書に語が出現する回数による Term Frequency(TF)，TF の対数，すなわち log(TF)などがよく利用される．大局的重みとは，文書集合全体においてある語がどの程度の重要度を持っているかということである．大局的重みについては考慮しない場合もあるが，Inverse Document Frequency(IDF)がしばしば用いられる．他にも大局的重みの計算手法は考えられる．正規化とは，局所的重みや大局的重みを考慮して作成された特徴ベクトルの長さを調整するものといえる．正規化を行わない場合，もとの文書の量によって作成されるベクトルの長さが不均一になり，後の処理において不都合が起こる場合がある．正規化としてよく行われるのが，全てのベクトルの長さを 1 にするコサイン正規化である．コサイン正規化されたベクトルどうしの内積は，2 つのベクトルが成す角のコサイン値になる．

このように，文書の特徴ベクトルを生成するのにも，様々なバリエーションが考えられ，場合によっては，TF によるベクトルを生成した後，各要素で値の対数をとる，IDF の重み

と掛け合わせる，正規化を行う，といったことを行う必要がある．SlothLib では，デザインパターンにおけるデコレーションパターンを利用したクラスが多数用意されており，ベクトルに対して修飾を行うことで実現する．

まず，3.4 節で述べた形態素解析器の結果から TF によるベクトルを作成する．このベクトルを tf とすると，例えば，TF の対数によるベクトルを作成するには，

```
IVector<string> logtf = new LogVectorSa<string>(tf);
```

というように，LogVectorSa というクラスのコンストラクタで修飾されるベクトルを与える．他の機能も同様に実現され，例えば，先ほどの logtf にコサイン正規化を行う場合には，

```
IVector<string> costf = new CosNormVectorSa<string>(logtf);
```

とする．これらの修飾のためのクラスを含む全てのクラスは共通のインターフェースを実装しており，ユーザは求めたいベクトルを簡単に作成することができる．

作成したベクトルに対して距離や類似度の計算が行われることが多いが，その手法には，ユークリッド距離，マンハッタン距離，内積，コサイン，Jaccard 係数，Dice 係数など，数多く存在する．現時点の SlothLib では内積，コサイン，ユークリッド距離のみが計算可能であるが，今後さらに多くの手法をサポートする予定である．

3.6 クラスタリング

特徴ベクトルで表されたアイテムが複数存在するとき，それらを自動的に分類するためにクラスタリングが利用される．クラスタリング手法には，階層型クラスタリングと非階層型クラスタリングが存在する．階層型クラスタリングは，初期状態として各アイテムをそれぞれ 1 つのクラスタとみなし，ある判定によって最も近いと判定されるクラスタを結合することを繰り返し，最終的には全体が 1 つのクラスタになるような手法である．クラスタ数や距離・類似度の閾値によって途中で停止させることで複数のクラスタを得ることができる．非階層型クラスタリングは階層型クラスタリングではないもので，代表的なものには K-means 法がある．K-means 法では最終的に作成するクラスタの数をあらかじめ決めておき，得られる結果のクラスタは階層的なものではない．

SlothLib では現在，階層型クラスタリングのうち，下記の 3 種について実装しており，今後さらに増やす予定である．

- 最短距離法
- 最長距離法
- 群平均法

クラスタリングされるアイテムが 3.5 節で述べた特徴ベクトルのクラスで表されている場合は，特徴ベクトルの配列をクラスタリングのクラスに与えるだけで，クラスタリングを行うことが可能である．

3.7 その他の機能

上記以外にも，行列の計算，統計ツールの利用，日本語構文解析や係り受け解析器などのより高度な自然言語処理ツールの利用などを行えるよう，SlothLib の機能拡張を続けている．また，ユーザインターフェースの部品化も行っている．その 1 つにばねモデルによるグラフインターフェースがある．

図 1 は Web 情報からある語の同位語や話題語を求め，その結果をばねモデルのグラフで表示するアプリケーションである．アイデアはユーザに与えられた語に助詞を付加して Web 検索を行い，その検索結果のみを解析することによって知識を抽出するというものであるが，SlothLib から利用できる Web 検索 API サービス，自然言語処理，特徴ベクトル機能を利用することにより，内部機能についてはほぼアイデアの実装のみに注力可能であった．さらに，ばねモデルインターフェースを利用することによって，視覚的にもわかりやすいアプリケーションを作成することが可能であった．

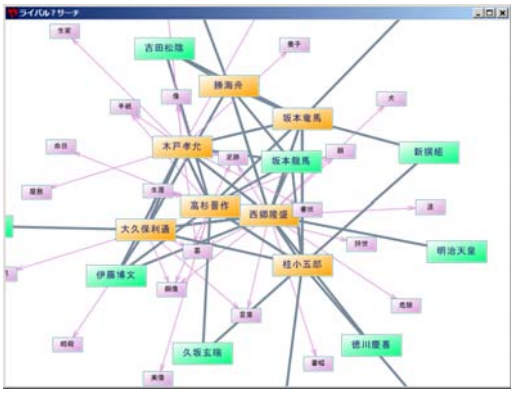

図 1 SlothLib を利用したアプリケーション例
Fig.1 An example application using SlothLib

4. ブロックプログラミング環境

アイデアを完全に実装する前には，予備実験がしばしば行われる．予備実験では大規模な実装は行わないまでも，多少のデータでアイデアを試すことになる．そのような，アイデアをテストする環境として，視覚的にプログラミングを行えるブロックプログラミング環境を作成した．SlothLib で実装されている Web 検索や特徴ベクトル生成などの機能が，GUI 上にブロックとして存在している．各ブロックはデータの入出力ポートを持ち，それらを接続していくことによってプログラムの流れを作ることができる．

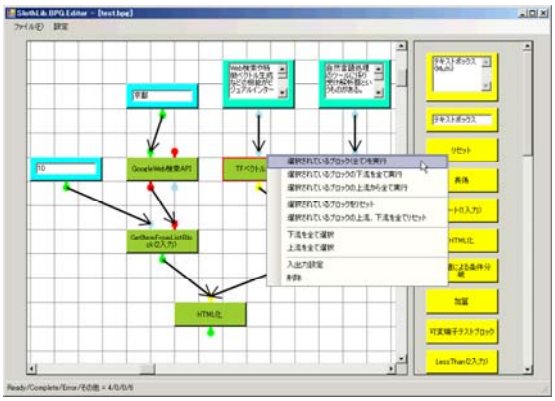

図 2 BlockProGramming の実行例
Fig.2 An example of BlockProGramming

図 2 はブロックプログラミング環境におけるプログラムの実行例である．右側には利用可能なブロックが並べられており，ユーザはブロックをドラッグして左側のスペースに配置する．各ブロックはデータの入出力を行うポートを持ち，各ポートはあらかじめ決められた型のデータを扱うようになっている．ある出力ポートからマウスをドラッグして別のブロックの入力ポートに矢印を接続することでデータの流れを決めることができる．その際には接続可能な入力ポートに対して点線が表れ，どのポートに接続できるかユーザが簡単に分かるようになっている．また，一部のポートは入出力形式を設定画面上から変更できるようになっており，例えば，ユーザが入力可能なテキストボックスのブロックの出力は，整数，浮動小数，文字列のいずれかに設定可能である．ブロックの配置と入出力の接続によって作成されたプログラム

は，保存することも可能である．

ブロックは，入力ポートに必要な情報が与えられている場合にのみ実行可能となる．ブロックは必要な入力があれば任意のタイミングで実行可能で，実行終了すると出力ポートからデータが出力される．ブロックの実行方法は複数用意されており，特定のブロックのみを実行する，あるブロックから下流に向かって連続的に実行する，あるブロックを実行するために必要なブロックを上流にさかのぼって全て実行するといったことが可能である．

ブロックを自作することも容易であるよう設計されている．全てのブロックは IBlock という共通インターフェースを実装しており，ある程度の処理がすでに記述された AbstractBlock 抽象クラスを継承することによって，ユーザは機能のみを実装することによってブロックが作成可能である．GUI 上のブロック間でやりとりされるデータには，数値から検索結果まで数多く存在するが，ブロックの自作と同様に，データについてもユーザが容易に作成可能である．作成したブロックやデータはDLL化してプラグインディレクトリに配置するだけで自動的に利用可能となる．ブロックプログラミング環境は，プログラミング初心者向けというよりもむしろ，ある程度プログラミングができる人が，システムの実装前にアイデアを試すためのものである．例えば，実装しようと考えるシステムの一部をブロックプログラミング環境で試すことが考えられる．ブロックプログラミング環境では，入出力ポートにマウスカーソルを近づけるとデータの内容が表示されたりするなど，アイデアを試す際には優秀なデバッグ環境としても利用できる．

今後，ブロックプログラミング環境で作成したプログラムを C#のコードに変換する機構を作成する予定である．コードへの変換が可能になれば，テストから実際のシステムの作成にあたって，コストを大きく減らすことができるようになる．

5. まとめと今後の課題

本論文では，Web サーチに関する研究や，既存の Web サーチソフトウェアをカスタマイズするためのプログラミングライブラリ SlothLib の概要について述べた．既存のツールやアルゴリズムの類似機能について共通インターフェースを提供しており，研究者はアイデアの実装により注力できるようになる．また，SlothLib の視覚的プログラミング環境についても述べた．

SlothLib は.Net 環境で利用可能であるが，今後 Java への移植を検討している．また，それ以降の展開としては，Ruby への移植や SlothLib の機能の Web サービス化などの要望が寄せられており，それらについても検討する予定である．

[謝辞]

本研究の一部は，文部科学省 21 世紀 COE 拠点形成プログラム「知識社会基盤構築のための情報学拠点形成」（リーダー：田中克己，平成 14～18 年度），文部科学省科学研究費補助金特定領域研究「情報爆発時代に向けた新しい IT 基盤技術の研究」計画研究「情報爆発に対応する新 IT 基盤研究支援プラットフォームの構築」(研究代表者:安達淳, Y00-01, 課題番号 18049073）ならびに計画研究「情報爆発時代に対応するコンテンツ融合と操作環境融合に関する研究」（研究代表者：田中克己，A01-00-02，課題番号 18049041），および，文部科学省研究委託事業「知的資産の電子的な保存・活用を支援するソフトウェア技術基盤の構築」，異メディア・アーカイブの横断的検索・統合ソフトウェア開発（研究代表者：田中克己）によるものです．ここに記して謝意を表します．また，SlothLib の開発には，稲川雅之氏，河重貴洋氏，木村塁氏，郡宏志氏，小谷彬氏，小西信次氏，近藤浩之氏，白砂健一氏，山口雅史氏，山本岳洋氏，吉田大我氏らの多大なる貢献を頂きました．ここに記して謝意を表します．

[文献]

[1] 阿部裕行, 伊藤一成, M. J. Durst, 「アノテーション処理のための基盤ライブラリの構築」, 電子情報通信学会技術研究報告, 第 17 回データ工学ワークショップ (DEWS 2006), 1A-i5 (2006).

[2] A. Mendelzon, G. Mihaila, T. Milo: “Querying the World Wide Web,” Proc. of the 4th International Conference on Parallel and Distributed Information Systems (PDIS '96), pp.80-91 (1996).

[3] T. Kistler, H. Marais: “WebL: A Programming Language for the Web,” Proc. of the 7th International World Wide Web Conference (WWW '98), pp.259-270 (1998).

[4] “Business Intelligence Development Studio," http://msdn2.microsoft.com/ja-jp/library/ms173767.aspx

[5] K. Ito, Y. Tanaka: “A visual environment for dynamic web application composition,” Proc. of the fourteenth ACM conference on Hypertext and hypermedia (Hypertext 2003), pp.84-193 (2003).

[6] Y. Tanaka: “Meme Media and Meme Market Architectures for the Reediting and Redistribution of Knowledge Resources,” Proc. of the International Conference on Multimedia Modeling (MMM '98), pp.1-10 (1998).

[7] 松村郁生, 宮浦宏暢, 尾曽越雅文, 「PatchService」 http://www.patchservice.net/

[8] W. Ma: “Building Infrastructure to Support Web-scale Data Mining for Search,” データベースと Web 情報システムに関するシンポジウム(DBWeb 2006)内講演 (2006).

[9] Max/MSP, http://www.cycling74.com/products/maxmsp

[10] G. Salton, M. McGill: “Introduction to Modern Information Retrieval,” McGraw-Hill (1983).

[11] 形態素解析器茶筌, http://chasen-legacy.sourceforge.jp/

[12] 形態素解析器 MeCab, http://mecab.sourceforge.net/

大島 裕明 Hiroaki OHSHIMA

2007 年京都大学大学院情報学研究科博士後期課程修了．博士（情報学）．主に Web 検索，Web からの知識抽出の研究に従事．情報処理学会，電子情報通信学会，ACM 各会員．現在は，京都大学大学院情報学研究科社会情報学専攻特任助教．

中村 聡史 Satoshi NAKAMURA

京都大学大学院情報学研究科社会情報学専攻特任助教. 2004 年大阪大学大学院情報学研究科博士後期課程修了．博士（工学）．主にヒューマンコンピュータインタラクション，ウェブ検索の研究に従事．情報処理学会，日本データベース学会会員．

田中 克己 Katsumi TANAKA

京都大学大学院情報学研究科社会情報学専攻教授．1976 年京都大学大学院修士課程修了．博士（工学）．主にデータベース，マルチメディアコンテンツ処理の研究に従事．IEEE Computer Society，ACM，人工知能学会，日本ソフトウェア科学会，情報処理学会，日本データベース学会等各会員．