アプリケーションのための日本語形態素解析システム

颯々野 学　斎藤 由香梨　松井 くにお
富士通研究所

1 はじめに

さまざまな自然言語処理の応用を考える上で、日本語の形態素解析の技術は最も基本的なものである。そのため、機械翻訳を始めとして、キーワード抽出や校正支援などさまざまな応用の場面が考えられる。最近の商用の機械翻訳のシステムを見てみても技術的には十分実用レベルにある。しかし、実際にさまざまな応用システムの中で使っていくには、高い性能を達成するだけではなく、初期開発コストや保守や運用のコストを低くすることも必要である。

これに対し我々は研究目的だけでなく、さまざまなアプリケーションへの利用を考慮に入れた日本語形態素解析システム (Breakfast と呼んでいる) を設計、開発している [1]。更に実際にその応用も進めている。

本稿では、この日本語形態素解析システム Breakfast での設計上の留意点、システムの概要、実際の適用例などを述べる。最後に、システムの設計、開発、応用の流れの全体を振り返り、それらの有効性を検討する。

2 システムの設計

まず従来のシステムの問題点を考察し、それを元にシステムの設計の方針を述べる。

2.1 従来システムの問題点

設計を始めた時点でも、富士通内では既に形態素解析を利用したシステムが存在した。多数の用途が考えられるにも関わらず、さまざまなアプリケーションで利用されているものはなかった。いくつかの原因が考えられるが、大きな理由は特定のアプリケーションに依存し過ぎていることだと考えた。それらの内訳を簡単に整理すると以下のようになる。

システムのモジュラリティ 例えば、機械翻訳専用の形態素解析システムでは、単純に形態素解析だけの部分を切り出すのが難しい。辞書も含めてコンパクトにライブラリ化がされていないと、他のアプリケーションから利用しにくい。また、特定のプラットフォームへの依存も問題になる。

辞書と形態素文法の保守性 アプリケーションには独自の単語の設定基準や品詞体系がある。それらは、基本部分とアプリケーション固有の部分が分かれていないため、辞書の再利用や、形態素文法の修正が難しい。

資源の共有 複数の部署で辞書の開発が行なわれている。しかし、辞書の独立性が低く独自に修正されているため、簡単に辞書データをマージすることができない。また、共有していくための組織がない。

2.2 設計の方針

2.1 節で示した考察を踏まえて、形態素解析システムの設計と開発を進めていった。我々が目標としたのは、図 1 に示すサイクルを確立し、自然言語処理を利用したアプリケーションを効率良く開発しつつ、日本語形態素解析システム自身の完成度を高めていくことである。本節では、設計の方針を述べる。

モジュール化の徹底 アプリケーションへの組み込みと、将来の部分的な改良への対応のために、システム内のモジュール化を計った。動的計画法に基づく解探索、コスト計算、辞書アクセス、メモリ管理、表示などのモジュールが極力独立になるように作成した。プラットフォーム依存の部分も一箇所にまとめた。

コストと連接属性の語彙からの分離 辞書の保守を容易にするためと、辞書を交換し易くするために、原則として辞書中の各形態素には優先度や形態素のコストと連接の属性は入れないようにした。連接の属性は、形態素文法をコンパイルすることで与えられる。基本的には表記と品詞でのみ管理されている。

分かり易い品詞体系 品詞体系があまり馴染みの無い独自の体系では、利用者の学習コストがかかる。そこで、品詞体系は学校文法をベースにした。学校文法は広く知られており、各アプリケーションの開発者や利用者が辞書を利用する際の学習コストは低く抑えられる。他の辞書データの取り込みもやり易くなる。

連絡先: 颯々野 学, 富士通研究所, 〒 211-88 川崎市中原区上小田中 4-1-1, Email: sassano@flab.fujitsu.co.jp

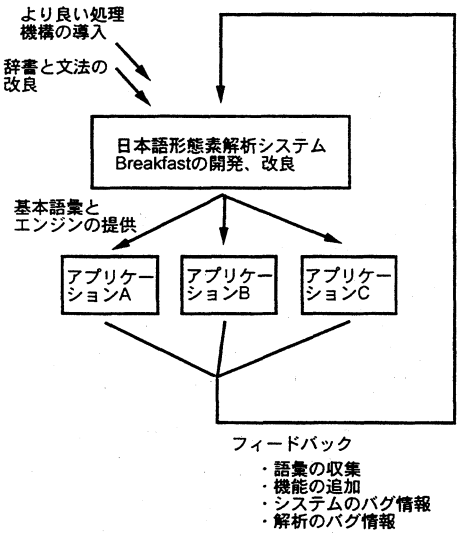

図 1: システム開発のサイクル

アプリケーションに依存しない語彙選定 形態素解析用辞書の語彙は一般的なものとし、特定の目的に依存し過ぎないように注意した。各アプリケーションでは、それぞれに必要な語を独自に追加して使用することを前提にする。

3 システムの概要

2.2 節の方針に基づいて実装した。C 言語で記述し、辞書アクセスライブラリを含めておよそ 52,000 行 (コメント含む) ある。実際の開発にあたっては、まず過去に作成した機械翻訳用の辞書をベースに、SunOS 4.1.x 上で作業を進めた。その後、他のプラットフォーム (Solaris 2.x, Windows 95, Windows NT) で動作させることや、連接属性と語彙の分離などを徐々に行なった。辞書の整備、形態素文法の開発作業も並行して進めた。

以下では、システムの主な構成要素と解析精度について述べる。なお、辞書アクセスライブラリや、解析速度については文献 [1] で報告している。

3.1 解析エンジン

形態素解析の基本となるアルゴリズムには動的計画法を用いたものを採用した。隣接する形態素が連接可能かどうか検査するために接続表を使う。コスト関数を使って解を順序付けする。コスト関数の枠組は [2] に準じ、式 (1) で表される。

$$(\text{接続コスト}) = (\text{形態素コスト}) + (\text{連接コスト}) \quad (1)$$

(形態素コスト) は品詞などに基づき決定されるコスト、(連接コスト) は隣接する形態素の連接コストである。

この解析エンジンを用いるアプリケーションでは、式 (1) を必要に応じて変更すればよい。動的計画法を用いて最適解を得る部分はそのまま利用できる。

3.2 品詞体系

基本となる語の品詞体系は学校文法をベースとした。我々の品詞体系の大まかな構成を表 1 に示す。用言の活用の扱いは活用語尾分離方式 (動詞の語幹と活用語尾を分割して辞書登録する) とし、活用語尾にも独立のタグを割り当てているためタグの数が多くなっている。

表 1: 品詞体系

品詞	名詞	動詞	形容詞	形容動詞
タグ数	12	15	2	3

副詞	連体詞	感動詞	接続詞	助詞
1	1	1	3	14

助動詞	接辞	活用語尾	その他	合計
98	12	125	9	295

3.3 形態素文法

隣接するカテゴリ間の接続とその連接コストを連接規則として記し、その集合を形態素文法とした。図 2 に連接規則の例を、表 2 に形態素文法の構成を示す。

```
;; 普通名詞 N と格助詞 JS_KAKU がコスト
;; 5 で接続する。
(defconnect
  '((N))
  '((JS_KAKU)) 5)
```

図 2: 連接規則記述の例

活用語尾分離方式を取っているため、活用関係の規則が多くなっている。また、辞書中の各形態素には優先度などを持たせていないため、それを補うための個別の形態素に関する例外的な規則が多くなっている。

3.4 辞書

実際の開発では、まずいろいろな所で利用されていた形態素解析用辞書の最大公約数となる語彙を収集し、それをベースに整備を行なった。

表 2: 形態素文法の構成

規則の種類	規則数
文節間の規則	47
品詞間の接続の規則	317
活用の規則	236
活用の例外規則	25
語構成の規則	41
個別の形態素に関する規則	314
合計	980

現在、一般的な固有名詞類を含む基本的な語彙 (約 105,000 語) を整備している。その他に専門用語など約 250,000 語を集めた辞書も用意している。

3.5 解析精度

まず、精度の向上のために行なった手順を述べ、次に我々が定期的に調べている解析精度について報告する。

実際の調節にあたっては、まず基準となる例文 (約 2,500 文) を用意し、それを元に文法の核の部分を調節した。その後、実際のテキストでのテスト、利用者からのバグ情報などを元に修正を行なっている。辞書への形態素の追加、品詞体系の見直しなども適宜行なっている。

表 3: 解析精度の測定結果

セット A	Jul/96	Sep/96	Feb/97
品詞誤り数	52	22	24
分割誤り数	11	6	5
出力形態素数	8246	8280	8263
正解率	98.78%	99.66%	99.65%

セット B	Oct/96	Nov/96	Feb/97
品詞誤り数	69	38	39
分割誤り数	54	21	19
出力形態素数	10,095	10,067	10,069
正解率	98.78%	99.41%	99.42%

表 3 に解析精度の測定結果を示す。基本となる語彙を集めた辞書を用いて測定した。ここで、セット A、B はともに無作為に選んだ新聞の社説記事である。異なる社説がそれぞれ 7 記事 (303 文)、11 記事 (340 文) 含まれている。

なお、正解率と正解の判定基準は次のように定義した。

$$\text{正解率} = \frac{\text{正解形態素の数}}{\text{出力された全形態素の数}}$$

- 登録語は、形態素の分割位置、形態素の品詞タグがともに正しいものを正解とする。
- 未登録語は、分割位置が正しいものを正解とする。

測定時期により品詞体系や辞書を変更しているので、単純にそれぞれの精度を比較することはできない (例えば、品詞分類が緩くなれば精度の値は向上する) が、十分実用的なレベルで安定している。

4 アプリケーションへの適用

Breakfast の改良と並行して、アプリケーションへの適用を進めた。それぞれのアプリケーションの開発担当者を含む Breakfast 利用者のメーリングリストを作成し、情報交換を行なって開発の効率化に努めた。

以下では、主な実際のアプリケーションへの適用例を紹介する (技術的な詳細についてはそれぞれの参考文献を参照されたい)。開発担当者に質問をして、Breakfast の利用状況を調査した。概要を表 4 に示す。アプリの欄の番号は以下の節と対応している。また、Breakfast の利用の程度を知る目安として、追加修正したコードの量や、元の Breakfast のコードがどの程度含まれているかも記す。

表 4: アプリケーションでの利用状況

アプリ	利用形態	解析エンジン	辞書
4.1	コマンド	そのまま	そのまま
4.2	コマンド	そのまま	追加
4.3	ライブラリ	コスト関数変更	独自
4.4	ライブラリ	コスト関数変更	調節

4.1 情報検索への応用

全文検索を行なう情報検索システム [3, 4] では、インデクス作成時と、検索時の日本語検索キーの解析に Breakfast を利用している。表示部の変更と、形態素解析サーバとしての機能の強化、プロトコルの追加が行なわれている。この形態素解析サーバのコードの殆どは Breakfast のものである。修正箇所は数百行程度。

4.2 障害事例検索への応用

交換機ソフトの障害事例検索システム [5] では、キーワードの切り出しに Breakfast を使っている。このシステムは、各事例をキーワードベクトルモデルで表現し、過去の事例の中から現在の障害状況と似ているものを選び出すものである。分野固有の単語や検索のための同義語は利用者が定義して使っている (追加は 1,000 語以下)。Breakfast 自身は変更していない。

4.3 校正支援への応用

仮名漢字変換の変換誤りを重点的に校正するシステム [7] では、コスト関数、辞書を置き換えて Breakfast を使っている。辞書アクセスルーチンも修正されている。このシステムはワードプロセッサに組み込まれている。追加コードは約 2,500 行で、このシステムのコードの約 95% は Breakfast のものである。

4.4 文字認識後処理への応用

文字認識後の候補文字を全て考慮した形で形態素解析を行なって、誤りを訂正するシステム [6] で Breakfast を使っている。文字認識システムの一部として組み込まれている。このシステムでは、辞書アクセスルーチン、コスト関数、形態素文法を変更している。候補文字を考慮した辞書引きを行なえるように辞書アクセスルーチンは改造されている。追加コードは約 5,000 行で、このシステムのコードの約 90% は Breakfast のものである。

5 評価と考察

2.2 節で掲げた目標が達成できたかどうかを客観的に評価するのは難しいが、アプリケーションでの利用実績など踏まえて、我々が採った方針が有効であったかどうかを考察し、評価を試みる。

全体的なことについて 短期間の間に複数の実用的なシステムで利用でき、基本的な目標は達成できたと言える。システムのモジュール、解析エンジンと辞書、文法などあらゆる箇所で依存関係を極力減らしたことが成功した。メーリングリストもバグ情報の迅速な通知や修正に役立った。

性能について 各単語に優先度を持たせないなど保守性優先の設計を行なった。しかし、実際のアプリケーションで解析精度が問題になることは殆どなく、十分実用的な精度が得られた。

開発コストの削減について まず、形態素解析のエンジンを複数のアプリケーションで共有できた点で初期開発コストの削減ができた。また、個々のアプリケーションで追加した機能 (複数解出力、他の OS への移植など) を元の Breakfast に反映させたので、その点でも開発の省力化ができた。

システムのバージョンアップについて Breakfast は 3-6 カ月おきに新版をリリースしている。しかし、アプリケーションの側では必ずしも利用できていない。理由は、製品計画やアプリケーション固有の改造箇所があるためである。これは、システム内のモジュールの再構成をしつつ、徐々にバージョンアップの負荷を下げていくしかない。

6 おわりに

本稿では、広くアプリケーションで使うための日本語形態素解析システムの設計と開発、実際の応用例について述べた。複数のアプリケーションで利用されていることを報告し、採用した方針が十分有効であることを確認した。

ただ、ミクロに見れば形態素解析システム自身にはまだまだ改良の余地がある。今後は、今までに確立した開発サイクルの中で形態素解析システムを改良し、アプリケーションへの応用を更に効率良く進めていきたい。

付録

我々は日本語形態素解析システム Breakfast を無償公開している。本稿で説明したものと基本的には同じである (機能制限あり)。詳しくは下記の URL を参照して頂きたい。

http://www.fujitsu.co.jp/hypertext/free/breakfast/

参考文献

[1] 颯々野 学, 難波 功:“利用者による調節が可能な高速日本語形態素解析”, 情報処理学会第 52 回全国大会論文集, 5B-4 (1996).

[2] Toru Hisamitsu and Yoshihiko Nitta:“A Uniform Treatment of Heuristic Methods for Morphological Analysis of Written Japanese”, *Proc. of 2nd Japan-Australia Joint Workshop on NLP* (1991).

[3] “ウェイズ (WAIS) サーバー” (http://www.fujitsu.co.jp/hypertext/wais/).

[4] “InfoNavigator Home Page” (http://infonavi.infoweb.or.jp/).

[5] 岡本 青史, 他:“類似事例検索システム —通信ソフト故障診断問題への適用—”, 情報処理学会第 52 回全国大会論文集, 7J-4 (1995).

[6] 小川 知也, 他:“文書認識における言語情報の活用 (1)”, 情報処理学会第 52 回全国大会論文集, 2J-8 (1996).

[7] 伊吹 潤, 他:“校正支援システム Joyner における表記誤りの自動訂正方式”, 自然言語処理研究会, 情報処理学会, NL 117-22 (1997).