情 報 学 基 礎 51－16
自 然 言 語 処 理 127－16
（1998. 9. 18）

日本語情報抽出システムの開発と評価

野畑 周
東京大学大学院
理学系研究科 情報科学専攻
〒113-0024 文京区本郷 7-3-1
理学部 7 号館内
nova@is.s.u-tokyo.ac.jp

関根 聡
Computer Science Department
New York University
715 Broadway, Room 709
New York, NY 10003, USA
sekine@cs.nyu.edu

概要

情報抽出システムの性能を向上させ、また複数の分野について情報抽出を行えるようにするためには、システムのもつ知識ベースの作成・更新をユーザが容易に行える必要がある。我々は、ニューヨーク大学が作成した情報抽出システム、そしてそのシステムの性能向上のためのカスタマイズツールの日本語化を行なった。さらに、Message Understanding Conference-6(MUC-6) で行われた人事異動に関する記事からの情報抽出タスクに基づいて日本語情報抽出システムの評価を行い、オリジナルのシステムとほぼ同等の性能を出せることを示した。

キーワード：情報抽出、パターンマッチング、カスタマイズ、MUC

Development and Evaluation of Japanese Information Extraction System

Chikashi NOBATA
Department of Information Science
University of Tokyo
Science Building 7. Hongo 7-3-1
Bunkyo-ku, Tokyo 113-0024 JAPAN
nova@is.s.u-tokyo.ac.jp

Satoshi SEKINE
Computer Science Department
New York University
715 Broadway, Room 709
New York, NY 10003, USA
sekine@cs.nyu.edu

Abstract

For improving Information Extraction systems and porting the systems to various domains, knowledge base in the systems should be easily created and customized. We developed the Japanese Information Extraction system and a customization tool based on the English system developed at New York University. We evaluated the system on an executive succession event, which is the scenario of Sixth Message Understanding Conference. The results demonstrated that the performance was comparable to that of the original English system.

Keywords: Information Extraction, Pattern Matching, Customization, MUC

1 はじめに

情報抽出システムの開発についての関心が近年高まっている。本論文でいう情報抽出システムの目的は、自然言語テキストに記述されている特定のイベントや関係の種類を認識することである。つまり、テキストからある情報を取り出し、それをデータベースなど特定の構造で記述することが目的であり、より一般的な text understanding に比べれば単純で扱いやすい問題である。実際、情報抽出システムの性能評価について客観的な基準 (F-measure [1]) が定められ、それを用いた情報抽出システムのコンテスト — Message Understanding Conference(MUC) — [1] が行われてきている。これまでの MUC プロジェクトの結果から言えることは、情報抽出という分野は将来有用であると考えられるが、まだ研究されるべきことが多いということである。まず向上すべきなのは抽出された情報の精度である。MUC-6(1995[2])、MUC-7(1998[3]) においては、参加したシステムの評価は最高でも F-measure で 60 を越えない。

また、情報抽出システムを別の分野に適用するのにかかる時間と労力にはっきりした上限がないことも問題である。システムのカスタマイズには何ヶ月も時間がかかり、またシステム内部の知識と対象分野の知識の両方が必要とされる。本論文では、英語向けに開発された情報抽出システムに基づいた日本語情報抽出システムとそのカスタマイズツールについてその構成と評価とを述べる。

2 情報抽出

本節では、情報抽出の定義を例を挙げながら詳しく述べる。まず、本論文で情報抽出について言及するときに用いる MUC プロジェクトでの用語を紹介する。

domain とは抽出対象とするテキストの分野を示す。例えば「経済記事」は domain の一つである。

scenario は抽出されるべきイベントの集合を指す。つまり、ある domain に含まれるテキストから抽出されるべき情報の集合である。MUC-6 で抽出されるべき scenario は「人事異動」であり、情報抽出システムは人事異動のイベント、つまりある会社の代表者が退任したり、新たな役職に就いたりといったことを抽出する。

人事異動のイベントには、人名、会社名、地名などが現われる。テキストに含まれる、scenario において重要なこれらの名詞句 (Named Entity) を認識するタスクは、Named Entity Task と呼ばれる。Named Entity Task は情報抽出のサブタスクとして MUC-6 で定義されている。

MUC-6 では、人事異動のイベントを抽出する作業を Scenario Template Task と呼ぶ。人事異動のイベントは、ある会社のある役職に対する人の就任、退任である。これに付随して取り出すべき情報として異動の理由、異動前または後の会社名などがある。情報抽出システムは、これらの情報を予め定義された型式 (Template) に従って出力する。

MUC-6 で定義されている型式は、複数の基礎的な Template とそれらを結ぶポインタとで表わされるが、これを表によって簡潔に示すと表 1のようになる。各行が一つのイベントに対応し、各列がイベントのスロット、つまりイベントの記述に必要な各情報に対応する。我々の情報抽出システムはテキストを段階的に解析し、表の中の空欄を充たす適切な文字列をテキストから取り出していく。次節では解析の各過程について述べる。

3 日本語 Proteus システム

日本語 Proteus システムの構成を図 1に示す。オリジナルの Proteus システムについては [4] を参照されたい。システムは階層的モジュールと、各モジュールに付随する知識ベースとから成っており、各モジュールは入力テキストを変換して次のモジュールへ渡していく。

最初のモジュールはテキストを一文ずつに切り分け、さらに各文を形態素解析して品詞情報を付与する。ここでは JUMAN[5] を用いている。

二番目のモジュールは Named Entity の抽出を行う。組織名、人名、地名、時間表現、数値表現

[1] F-measure とは、再現率 (Recall) と適合率 (Precision) とから計算される値で、特に重みづけをしない場合以下のように定義される:

$$F = \frac{2PR}{P+R}$$

表 1: 抽出されるイベントの例

例文：

北海道テレビ放送（札幌市）は二十四日開いた株主総会後の取締役会で、滝井禧夫専務（５９）の社長昇格と、畠山武社長（６９）の代表権のない会長就任を決めた。

結果：

役職名	会社名	地名	人名	Status	On the Job	Vacancy Reason
専務	北海道テレビ放送	札幌市	滝井禧夫	OUT	NO	REASSIGNMENT
社長	北海道テレビ放送	札幌市	滝井禧夫	IN	YES	REASSIGNMENT
社長	北海道テレビ放送	札幌市	畠山武	OUT	NO	REASSIGNMENT
会長	北海道テレビ放送	札幌市	畠山武	IN	YES	REASSIGNMENT

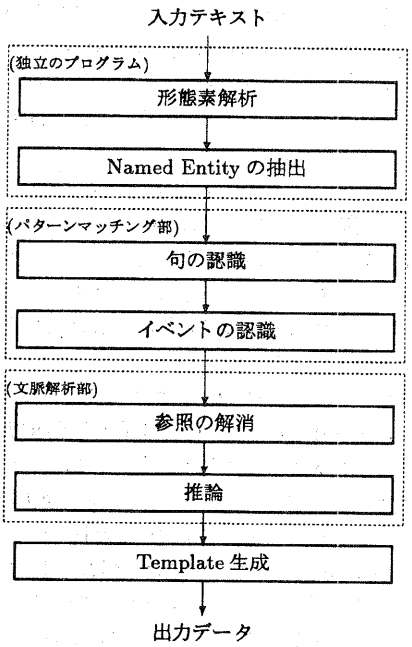

図 1: 日本語 Proteus システムの構成

が認識され、対応する SGML タグが入力テキストに付与される。ここまでのモジュールは、決定木手法を用いた一つの独立したプログラム [6][7] になっている。

次の二つのモジュールは、単純なパターンから複雑なパターンまで段階的にパターンマッチングを行う。パターンは、あるパターン記述言語に従って書かれた正規表現であり、マッチしたときに行われるべき操作と結びつけられている。また、パターンは高速化のために予め展開して保存されている。また、これらのモジュールはパターンマッチングに用いられる概念階層 (conceptual hierarchy) をもっている。概念階層は Named Entity に含まれる名詞句や、パターン記述に必要な動詞句について上位下位関係を定義している。例えば、「組織」という概念の下位概念には「会社」や「政府団体」があり、「会社」の下位概念には「銀行」や「新聞社」など、さらに細かい概念が定義されている。

パターンマッチングの最初の過程では、名詞句や動詞句といった、部分的な構文単位が認識される。句が認識されるたびに、その句に対応する意味情報が付与される。例えば、表 1の情報を抽出するのに必要なパターンの一つに、名詞句を認識する以下のようなパターンがある:

NP(組織) ' (' NP(場所) ') '

NP() は、それぞれ対応する概念に含まれる名詞を主辞とする名詞句にマッチする。パターンがマッチした後に行われる操作によって、マッチした表現全体が組織名を主辞とする一つの句と見なされ、地名の情報は組織名の意味情報に追加される。

パターンマッチングの二番目の過程では、domain や scenario に固有のパターンを適用し、より範囲の広い構文単位を認識する。各パターンについてマッチしたときに行われる操作によって、マッチした表現がもつイベントの情報が logical form として格納される。例えば、以下のようなパターンを考える:

NP(人名) ' が' NP(役職名) ' に' VP(昇格)

```
(defpattern jpn-appoint2
 "j-person-to-change jcasemarker-agent
  j-at-date?
  np-sem(C-position) 'に'
  jnoun-appoint jv(C-サ変動詞)?:
  verb-span=7.span,
  position-at=4.attributes |
  np-sem(C-position) 'に'
  j-person-to-change jcasemarker-agent
  j-at-date?
  jnoun-appoint jv(C-サ変動詞)?:
  verb-span=7.span,
  position-at=1.attributes")
```

図 2: システム内のパターン記述の例

このパターンがマッチすると、人名、役職名、述語「昇格」のそれぞれの意味情報から異動のイベントを表わす logical form が作成される。実際のパターン記述の例を図 2に示す。

パターンマッチング以降のモジュールは、できあがった logical form について操作を行う。参照の解消モジュールでは、相互参照の解消を行い、テキスト中に複数回現われる同じ Named Entity の意味情報を一つにまとめる。推論モジュールでは、空いているイベントのスロットに適切な意味情報を入れたり、複数の文にまたがって記述されているイベントを一つの logical form にまとめあげたりする。最後に Template 生成モジュールでは、出来上がった logical form を指定された型式、例えば表 1のような型式に沿って出力する。

4 カスタマイズツール

4.1 情報抽出システムの問題点

情報抽出システムを新たな scenario や domain に適用するときのカスタマイズの困難さについては先にふれた。ここでは、システムをカスタマイズするときに、システムのどの部分を再構成するべきかという点について検討していく。

前節でモジュール化されたシステムの概要を説明したが、このシステムでは各モジュールの実行部分はほぼ汎用的であり、新たな domain にもそのまま対応できる。つまり、カスタマイズするのは知識ベースの部分に限られる。知識ベースのカスタマイズの中では、パターンの作成がもっとも複雑で、労力を必要とする部分である。

語彙項目や概念階層も domain に固有の要素を追加する必要があるが、これらは簡単なツールを用意すれば比較的容易に編集することができる。Template の出力型式についても、出力モジュールがきちんと作られているならば、出力の変換は、モジュールに対する命令や関係代数の演算で表現することができる。domain や scenario によっては、文脈処理のためにもっと複雑な推論モジュールが必要になる可能性がある。例えば、時制の処理である。現在の情報抽出システムでは、複雑な時制を扱うことは想定されていないので、今後この種の処理を行うために拡張される必要があるかもしれない。もっとも、時制の解析自体は汎用的であるから、いったん拡張されれば新たな scenario に適用する際のカスタマイズに要する労力は少なくてすむと思われる。

パターンのカスタマイズはこれらのカスタマイズとは性質を異にする。scenario や domain に固有の情報から成っており、そのためシステムの性能をある程度の水準まで引き上げるためにはパターン全体を視野に入れながら作成と評価を何度も繰り返す必要がある。

パターンの作成・更新にかかる時間と労力の軽減は、情報抽出の分野においては重大な問題である。この問題を解決する方法として、システムの内部に精通した人間でなくてもパターンの変更を行なえるようにしようという研究が行われてきている。Massachusetts 大学では教師付き学習を用いたシステムを開発し、MUC のトレーニングコーパスからパターンを学習する研究を行っている [8]。トレーニングコーパスとは、人手で作成された Template のことで、MUC で情報抽出の結果を評価するときに正解として用いられるものである。情報抽出のタスクの中では、教師付き学習が効果を示しているものもある。Named Entity に限れば、BBN は最近同様の手法を用いて人手による最も性能の良いシステムにほとんど劣らない結果を出している [9]。しかしながら、教師付き学習の性能は、人手で作成された充分な量のトレーニングデータが手に入るかどうかにかかっている。Massachusetts 大学のシステムは、人手で生成されたトレーニングデータが数千記事用意されたときには他のシス

テムと同程度の性能を示したが、MUC-6で用意された100記事のトレーニングデータでは獲得された知識は充分ではなく、性能は低下している[10]。

4.2 Proteusシステムの解決法

前節で述べた問題に対してProteusシステムが与える解決方法は、パターンをいくつかのレベルに分けて、それぞれを別々に扱うことである。最も低レベルのパターンはシステムの中心部分に組み込む。さらに対象とするdomainに応じて、ある程度のパターンをパターンライブラリとして予め提供する。

一方、各scenarioに固有なパターンは、ユーザの入力からシステムが直接獲得していくことができる。ユーザはこの作業を対話的なグラフィカルインタフェースによって行い、内部の動作はユーザには見えないようになっている。パターンを作っていく過程では、ユーザはその正規表現にのみ注意を払い、パターンを直接手でコーディングすることはない。ユーザの行う作業は、以下の二つである:

- テキストから対象としているイベント記述の例を抜き出すこと
- 対応する出力の構造を選択すること

これらの情報から、Proteusシステムはユーザが指定したテキストから、ユーザが指定した構造を取り出すパターンを自動的に作成する。オリジナルである英語用のProteusシステムでは、さらに生成されたパターンの適用範囲を大きくするために、パターンの統語的一般化も行うことができる。

4.3 パターン獲得の手法

この節では、カスタマイズツールによるパターン獲得の詳細を述べる。最初にシステムが有するパターン群は、組み込みのパターンや対象domainに対応するパターンライブラリである。これらのパターンは、新しいパターンが獲得されるときにそのパターンの構成要素として用いられる。パターン獲得のサイクルは、ユーザの観点から見ると、パターンの獲得を繰り返してシステムのもつパターン群を拡張していくことである。図3に示すウインドウでパターン獲得が行われる。ユーザは、ウインドウの一番上のボックスに例文を入力する。二番目のボックスにはパターンの各構成要素が表示される。ユーザはこれらを一定の制限のもとで編集することができる。その下の大きなボックスでは、各要素のもつ構造がグラフ表示されている。ウインドウの下部では、パターンがマッチしたときに作成されるべきイベントが定義される。

パターンの獲得は、以下のような段階を経て行われる。

(1) 例文を与える
ユーザは一番上のボックスに例文を打ち込むか、または入力テキストが表示されているウインドウから文字列をコピーする。

(2) イベントを選ぶ
ウインドウの左下にあるメニューから、例文に対応するイベントを選ぶ。メニューにあるイベントは、スロットの情報も含めてあらかじめscenarioの定義の一部として与えられている。

(3) パターンマッチングを行う
ユーザがmatchボタンを押すと、システムは現在あるパターン全てを用いて例文に対してパターンマッチングを行う。マッチングの結果がパターンを表示するボックスに示される。

(4) パターンの要素を修正する
マウス操作によって、パターン表示ボックス内の各要素を修正する。行える修正操作には以下のものがある:

- generalize: 概念階層に従って、要素の属する概念をより上位の概念に一般化する。
- optional: 疑問符(?)を後ろに付加し、その要素がoptionalであることを示す
- literal: パターンマッチングのときに、その要素が字面そのままで現われる必要があることを示す
- remove: パターンに必要ない要素を取り除く

(5) イベントのスロットに要素を入れる
ユーザは、パターン表示ボックス内のどの要

素がイベントのどのスロットを充たすのに使われるのかを指定することができる。要素をクリックすると、その要素のもつ logical form がグラフ表示される。その logical form の親ノードをスロットに drag-and-drop することでスロットに入れる要素を指定することができる。

(6) パターンを作成する
ユーザが acquire を選択すると、システムは新しいパターンを作成し、またそのパターンがマッチしたときにイベントのスロットを埋める操作を作成してパターンと対応させる。パターンはシステムのもつパターン群に追加される。

(7) 新しいパターンをテストする
ユーザは、新しく作成されたパターンをウインドウ上で元の例文に対して、また新しい文に対して適用することができる。またこのパターンを他のトレーニングコーパスについても適用してみて、さらに改良を加えることもできる。

対話的に獲得されたパターンは、パターンライブラリに追加して再利用することができる。また、節を含まないパターンも同様にこのツールで獲得することができる。

5 システムの評価

MUC-6 での対象 scenario であった人事異動について、日本語 Proteus システムの性能評価を行った。結果を表 2に示す。

英語のコーパスは Wall Street Journal(1993、1994年)、日本語のコーパスは日本経済新聞 (1993、1994年) からとられている。オリジナルの Proteus システムは MUC-6 で最も良い成績を挙げたシステムの一つである。ここに掲げたものはその MUC-6 における評価である。日本語 Proteus システムもオリジナルの Proteus システムに見劣りしない性能を上げていることが分かる。テストコーパスに対する性能において日本語 Proteus システムの方が英語よりもよい理由としては、以下のことが考えられる。

- 日本語の新聞記事の方が表現が定型的である：
例えば、人事異動のイベントが受動態で記述される頻度は英語に比べて日本語の方が低い。つまり日本語の場合少ないパターンでより多くの記事からイベントを抽出することができると考えられる。

- パターン作成にかけた期間が長い：
英語のパターン作成は MUC-6 の運営方針により、1ヶ月で行われた。一方日本語のパターン作成には2ヶ月強かかっている。ただし、英語のシステムには過去のパターンの蓄積があるのに対し、日本語のパターンは一から作成する必要があったので、かけた時間だけより適用範囲の広いパターンライブラリが作成できたとは必ずしもいえない。

- Named Entity Task が独立したプログラムで行われている：
オリジナルの Proteus システムでは Named Entity Task もパターンマッチングを用いて行っているが、日本語 Proteus システムでは Named Entity Task は決定木手法を用いた別のプログラム [6][7] によって行い、その出力を利用してイベントの抽出を行っている。これによりパターンマッチングだけでは認識することが難しい、あるいは認識するのに多くのパターンを書く必要があるような Named Entity がパターンマッチングを行う前に予め認識されており、パターンマッチングの性能の向上とパターン作成に必要な労力の軽減の双方に効果があったと考えられる。

6 結論

本研究では、英語向けの情報抽出システムに基づいた日本語情報抽出システムを開発し、オリジナルと同程度の性能が発揮できることを示した。また英語と同様に知識ベースの編集が行えるカスタマイズツールを提供した。情報抽出において、カスタマイズは重要な問題である。カスタマイズのしやすさは情報抽出のあらゆる面における進歩に影響を及ぼす。高精度の情報抽出が行えると主張するためには、広範囲の domain について実際に高精度の情報抽出が行えねばならない。カスタ

表 2: 日英両 Proteus システムの性能評価

言語	コーパスの種類	記事数	recall	precision	F-measure
英語	トレーニング	100 記事	61	74	67.01
〃	テスト	51 記事	47	70	56.40
日本語	トレーニング	87 記事	64	71	67.32
〃	テスト	90 記事	57	70	62.61

マイズにかかるコストは開発時間だけでなく、カスタマイズを行うのに必要な知識の程度にも比例する。我々の提供する手法は、ある domain の専門家が、システムに関する知識がなくても、情報抽出システムをその domain に適応させることができるようにすることで、カスタマイズのコストを小さくするものである。

謝辞

本論文は、International Workshop on Lexically Driven Information Extraction で発表された、Roman Yangarber と Ralph Grishman による論文に基づいている。日本語用情報抽出システムとカスタマイズツールは、英語用のシステムに基づいて開発された。英語用の情報抽出システムは Ralph Grishman 教授によって、英語用のカスタマイズツールは Roman Yangarber によって開発された。彼らはまた日本語のシステムを我々が開発するのに協力してくれた。彼らに深い感謝の意を表したい。この研究は富士通研究所の援助を受けている。

参考文献

[1] Ralph Grishman and Beth Sundheim. Message Understanding Conference-6: A Brief History. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING 96)*, pp. 466–471, Copenhagen, August 1996.

[2] DARPA. *Proceedings of the Sixth Message Understanding Conference(MUC-6)*, Columbia, MD, USA, November 1995. Morgan Kaufmann.

[3] DARPA. *Proceedings of the Seventh Message Understanding Conference(MUC-7)*, Fairfax, VA, USA, May 1998.

[4] Ralph Grishman. The NYU system for MUC-6 or Where's the Syntax? In *Proceedings of the Sixth Message Understanding Conference(MUC-6)*, pp. 167–175, Columbia, MD, USA, November 1995. Morgan Kaufmann.

[5] 松本裕治, 黒橋禎夫, 山地治, 妙木裕, 長尾真. 日本語形態素解析システム JUMAN 使用説明書 version 3.3. 京都大学工学部, 奈良先端科学技術大学院大学, 1997.

[6] Satoshi Sekine. NYU:Description of the Japanese NE System Used for MET-2. In *Proceedings of the Seventh Message Understanding Conference(MUC-7)*, Fairfax, Virginia, USA, May 1998.

[7] Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A Decision Tree Method for Finding and Classifying Names in Japanese Texts. In *Proceedings of the Sixth Workshop on Very Large Corpora*, Montreal, Canada, August 1998.

[8] Stephen Soderland, David Fisher, Jonathan Aseltine, and Wendy Lenhert. CRYSTAL:Inducing a Conceptual Hierarchy. In *Proceedings of the International Joint Conference Artificial Intelligence(IJCAI-95)*, pp. 1314–1319, Montreal, Canada, 1995.

[9] Daniel Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a

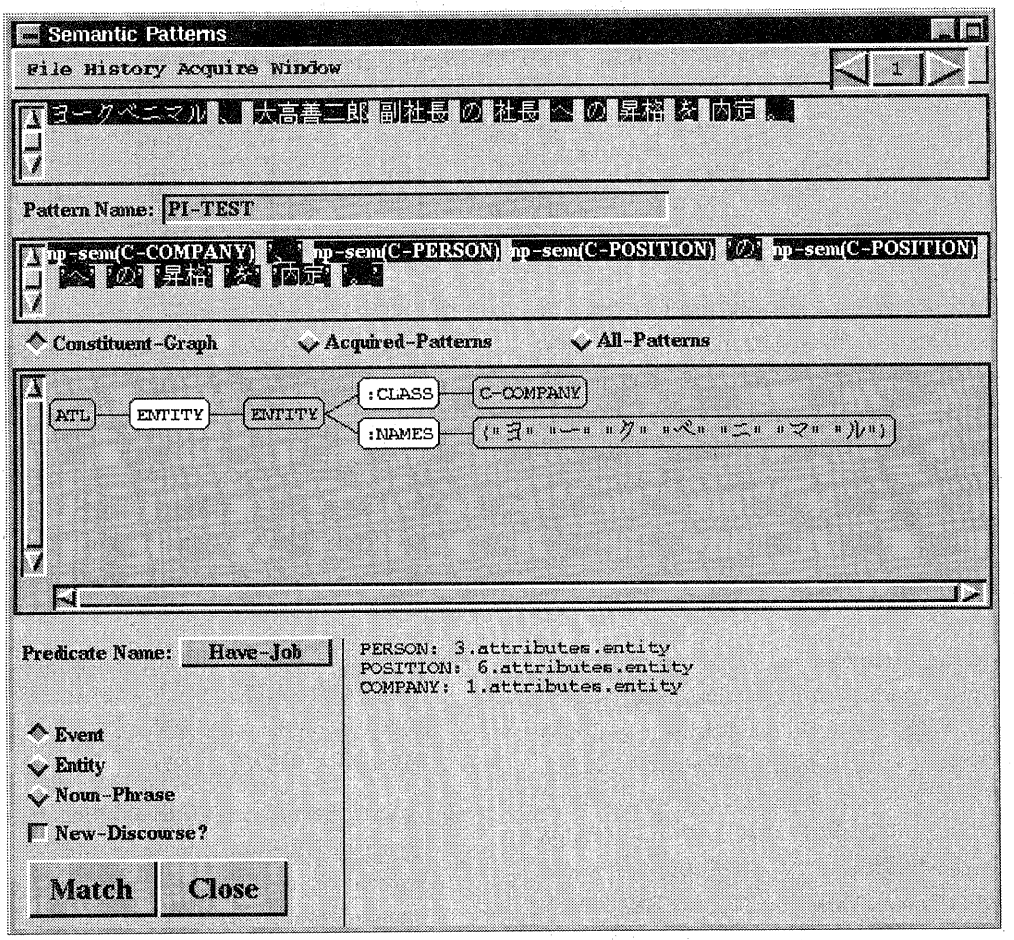

図 3: パターン獲得ウインドウ

High-Performance Learning Name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pp. 194–201, Washington, DC, USA, April 1997. Association for Computational Linguistics.

[10] David Fisher, Stephen Soderland, Joseph McCarthy, Fangfang Feng, and Wendy Lehnert. Description of the UMass System as Used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference(MUC-6)*, pp. 127–140, Columbia, MD, USA, November 1995. Morgan Kaufmann.

[11] Roman Yangarber and Ralph Grishman. Customization of Information Extraction Systems. In *Proceedings of the International Workshop on Lexically Driven Information Extraction*, pp. 1–11, Frascati, Italy, 1997.