修士論文

国語辞典からの類義表現抽出と SYNGRAPH構造による柔軟マッチング

指導教員　　黒橋　禎夫　助教授

東京大学大学院 情報理工学系研究科　電子情報学専攻

46403　　大西　貴士

平成18年2月3日作成

内 容 梗 概

自然言語は同じ内容を表現する場合でも様々な表現を使用することができる．このような様々な表現のずれをいかにして吸収するかは自然言語処理における重要な問題である．この問題は，国語辞書やシソーラスなどから得られる同義関係や上位・下位関係を統合的に参照することで解決できると考えられる．そこで本論文では，国語辞典から自動で抽出した同義関係や上位下位関係を用い，文を SYNGRAPH というデータ構造で扱うことで，様々な表現のずれを吸収する柔軟なマッチング手法を提案する．機械翻訳と情報検索のタスクで実験を行ったところ，提案手法の有効性を確かめることができた．

目 次

第 1章： 序論

自然言語は自由度が高いため，同じ内容を表現するにしても様々な表現を使用することができる．そのような様々な表現のずれをいかにして吸収するかが自然言語処理における重要な課題である．

例えば，用例ベースの機械翻訳を行うときに，入力文として

ホテルに一番近い駅はどこですか

が与えられたとする．このとき，用例に

旅館の最寄りの駅はどこですか
↔ Where's the nearest station from the hotel?

があったとしても，単純な完全マッチングを行うだけではこの用例を翻訳に使うことはできない．こういった問題は，機械翻訳だけでなく，情報検索や質問応答など様々なタスクでみられる．

このような問題を克服するには，表現のずれを吸収する柔軟なマッチングが必要であり，そのためには，同義・類義表現の知識の獲得と，それらを柔軟に統合して利用する枠組みの 2 つが必要となる．

本研究では，これらの 2 つの問題を次のように解決し，柔軟なマッチングを実現する手法を提案する．

1. 国語辞典から全自動で同義関係や上位下位関係の知識を獲得する．
2. 表現のずれをパックした SYNGRAPH というデータ構造を導入することにより，表現のずれの組み合わせを効率的に扱う．

1 については，シソーラスから同義関係や上位下位関係を獲ることができる．しかし，既存のシソーラスをそのまま柔軟なマッチングに利用するのは適当ではない．理由の 1 つとして，既存のシソーラスは広い意味での同義関係でしかまとめられておらず，本当に意味が近い同義関係かどうかを区別することができない．これは，機械翻訳での用例検索のような入力文と非常に近い用例を探す場合に特に問題となる．他の理由として，既存のシソーラスには句レベルの同義関係がほとんど含まれていないこともあげられる．

このような問題点を解決するため本研究では，国語辞典から，常識的，基本的な同義・類義表現を網羅的に，完全に自動的に獲得する．これにより,「夕食」と「食事」のような上位関係,「旅館」と「ホテル」のような単純な同義語関係に加えて,「一番」と「もっとも」などの副詞の同義語や,「最寄り」と「一番近い」のような語と句の同義関係など，広

い範囲の類義関係を扱うことができる．また，新聞記事の括弧表現から,「ＨＤＴＶ」＝「ハイビジョン」のような専門用語や固有表現の同義関係を獲得した．

2について，これまで，様々な表現の組み合わせを扱って効果が確かめられたのは，情報検索でのクエリ拡張だけであった．しかしそれは，Bag-of-words のアプローチであり，係り受け関係を含めた文レベルの同義関係までは扱ってはいない．このように文レベルの同義関係を詳しく扱うことは，機械翻訳において重要であり，高性能な情報検索や質問応答を実現するためにもやはり必要である．

本研究では文レベルの同義関係を扱うために SYNGRAPH というデータ構造を導入した．類義関係を単純に展開すると組み合わせ数が爆発してしまうが，それぞれの同義関係に ID を与え，様々な表現のな組み合わせをパックした SYNGRAPH 構造で文を表現することで組み合わせ爆発の問題を解決した．

本論文の構成は以下のとおりである．まず，2 章で表現のずれについて説明し，3 章で関連研究をあげる．4 章で国語辞典や新聞記事からの知識獲得手法について説明する．5 章で SYNGRAPH 構造について説明し，6 章で SYNGRAPH を用いた用例検索，情報検索手法を説明する．7 章で実験結果を示し，最後に 8 章で結論を述べる．

第 2章 :　様々な表現のずれ

この章では，まず表現のずれの定義を説明し，次に表現のずれのもとになる基本要素を分類し，基本要素が複数組み合わさった複雑な表現のずれについて紹介する．最後に本研究で扱う基本要素について述べ，どういった範囲の表現のずれを吸収するかについて述べる．

2.1　表現のずれ

表現のずれとは，表層的には異なるが意味的には近い内容を持つ表現のことをいう．表現のずれは，語，句，文などの様々なレベルにおいて見られるが，それらは，表現のずれの元になる基本要素が複合したものとみなすことができる．

2.2　基本要素の分類

本研究では，表現のずれの基本要素として，(1) 同義関係，(2) 上位下位関係，(3) 機能的表現の有無，(4) 構文レベルの言い換えの 4 つに分類した．

2.2.1　同義関係

語や句レベルの同義関係によるもの．語と語の同義関係と語と句の同義関係に分けられる．本研究では，語を句で言い換えたものを同義句と呼んでいる．

語と語の同義関係

地震 ＝ じしん ＝ 地しん （かな漢字）
コンピュータ ＝ コンピューター （表記ゆれ）
発生 ＝ 起こる （同義語）
帰り ＝ 帰る （体言と用言）

語と句の同義関係

発災 ＝ 災害が発生する （同義句）

2.2.2　上位下位関係

語と語の間の上位下位の関係によるもの．図 2.1 のようにツリー状に表現される．

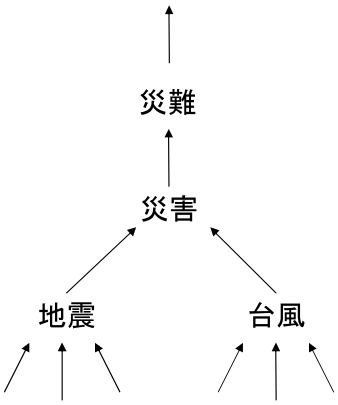

図 2.1: 上位下位関係

2.2.3 機能的表現の有無・交代

助詞や接尾辞などの機能的な表現の有無，交替によるもの．助詞の有無・交代，反意語の否定などがある．

助詞の有無・交代

日本車 = 日本の車
東京転勤 = 東京への転勤
問題を検討 = 問題について検討
酒に酔う = 酒で酔う

反意語の否定

明るい = 暗くない
不完全だ = 完全でない

その他

明けかかる = 明けはじめる（アスペクト）
飲んだはずだ = 飲んだに違いない（モダリティ）
読める = 読むことができる（可能）
赤 = まっ赤（強調）

2.2.4 構文レベルの言い換え

構文レベルの言い換えによるもの．以下のように，格要素の入れ替わり，受身・使役，対議，対称，単文の構造，文の接続表現，冗長な表現などがある．

格要素の入れ替わり

駅で切符を買う = 切符を駅で買う

受身・使役

AがBを捕まえる ＝ AにBが捕まえられる
AがBを捕まえる ＝ AにBを捕まえさせる

対義

AがBにCを預ける ＝ AからBがCを預かる
AがBにCを売る ＝ AからBがCを買う

対称

AがBと会う ＝ BがAと会う ＝ AとBが会う

単文の構造

太郎が怒った ＝ 怒った太郎 ＝ 怒ったのは太郎だ

文の接続表現

転んだ ので 泣いた ＝ 転んだ ために 泣いた
＝ 転ん で 泣いた
逃げる なら 撃つ ＝ 逃げ れば 撃つ

冗長な（無くても推測できる）表現

雷 が鳴って 驚いた ＝ 雷で驚いた
学校の 中の プール ＝ 学校のプール

2.3 複雑な表現のずれ

前節で挙げたような基本要素が複数組み合わさることで，さらに複雑な表現のずれが生まれる．例えば，

打鍵ミス
キーの打ち間違い

の例では，図 2.2 のように「打鍵」＝「鍵盤をたたく」といった同義関係や「鍵盤」＝「キー」のような上位下位関係などの複数の基本要素によって表現のずれが生み出されている．

このような複雑な表現のずれを吸収するためには，まず表現のずれの基本要素を吸収し，さらに，それを組み合わせ的に扱うことのできる枠組が必要となる．本研究では，同義関係にある語や句を ID で代表させ，SYNGRAPH というデータ構造を用いたマッチングを行うことで，複雑な表現のずれを吸収する．

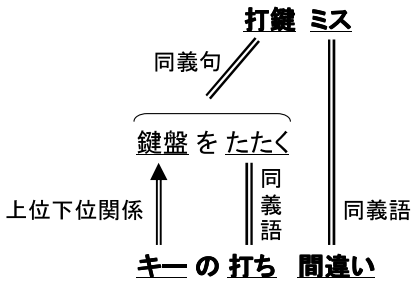

図 2.2: 「打鍵ミス」と「キーの打ち違い」の関係

2.4 本研究で扱う基本要素

本研究で主に扱う基本要素は同義関係と上位下位関係である．同義関係の中のかな漢字と表記ゆれについては，形態素解析器の辞書を整備し，各語を代表表記で表すことによってこれらの表現のずれを吸収する．また，同義語，同義句，上位下位関係については，国語辞典や新聞の括弧表現からこれらの関係を自動で獲得したものを知識として利用する．さらに，SYNGRAPH データ構造を用いることで，基本要素が組み合わさった複雑な表現のずれを吸収する．

そのほかの基本要素について，助詞の有無・交代は，助詞の違いを見ないことによってある程度解決できる．また,「不適合」や「適合しない」といった反意語の否定については，否定表現を含むか含まないかの否定フラグで扱うことで吸収できる．

構文レベルの言い換えについては，本研究ではほとんど扱っていないが，SYNGRAPH は依存構造木をベースにした構造であるため,「どこで切符を買うのですか」と「切符をどこで買うのですか」のような格要素の入れ替わりについては表現のずれを吸収できる．

第 3章 :　関連研究

関連研究として,

- 情報検索に関する研究
- 知識獲得に関する研究
- 柔軟なマッチングに関する研究

をあげる.

3.1　情報検索に関する研究

情報検索は，大量の文書集合の中から与えられた質問文に適合する文書を抽出するタスクであり，質問文と文書との間のマッチングの性能が直接的に反映されるタスクである．ここでは，情報検索で良い性能を示すとして知られており，本研究でも利用した Okapi のモデルを紹介する．さらに，情報検索の評価型ワークショップである IREX について説明する．

3.1.1　Okapi

Okapi は S. E. Robertson らによってに開発された検索システムで，確率型検索モデルに基づいた重要度計算を行っている．ここで紹介するのは，Okapi で用いられている BM25 という計算式である．これは，単純な tf・idf にくらべて良い性能を示すため，多くの検索システムで用いられている手法である．

Okapi の確率型検索モデルは，クエリが与えられたときの各文書の適合確率で文書を降順にランキングしたものが検索結果として最適になるという PRP 原理（Probability Ranking Principle）を出発点として定式化している [1]．さらに，2-Poisson モデル [2] を用いて tf や df，文書長の概念を組込み，大幅な近似をすることで，下のような BM25 と呼ばれる式が得られる [3]．この手法では，クエリ q が与えられると，文書 d_i ごとにスコア $S(q, d_i)$ を計算し，ランキングする．

$$
\begin{aligned}
S(q,d_i) &= \sum_{q\text{ 中の単語}_j} (s_1 s_3 \times \frac{tf_{ij}}{K+tf_{ij}} \times \omega_j \times \frac{tf_{qj}}{k_3+tf_{qj}}) + k_2 \times dl_q \frac{avdl-dl_i}{avdl+dl_i} \\
\omega_j &= \log \frac{N-df_j+0.5}{df_j+0.5} \\
K &= k_1((1-b)+b\frac{dl_i}{avdl})
\end{aligned}
\tag{3.1}
$$

ここで，N は文書数，tf_{ij} は 文書$_i$ 中の 単語$_j$ の出現回数，tf_{qj} はクエリ中の 単語$_j$ の出現回数，df_j は 単語$_j$ の出現文書数，dl_i は文書長（dl_i の単語数），$avdl$ は平均文書長である．s_i は，k_i に関する係数で，$s_i = k_i + 1$ を用いる．

このように BM25 は，k_1，k_2，k_3，b の 4 つのパラメータを持つ式であり，$k_1 = 2.0$，$k_2 = 0.0$，$k_3 = \infty$，$b = 0.75$ と設定した式がよく使われている．

$$
S(q,d_i) = \sum_{q\text{ 中の単語}_j} \left(tf_{qj} \times \frac{3.0tf_{ij}}{(0.5+1.5dl_i/avdl)+tf_{ij}} \times \log \frac{N-df_j+0.5}{df_j+0.5} \right) \tag{3.2}
$$

3.1.2 IREX

IREX は，情報検索や情報抽出の評価型ワークショップで，情報検索タスクと，固有表現抽出タスクがある．IREX の情報検索タスクは，図 3.1 のような検索要求に適合する内容の文書（新聞記事）を 2 年分の新聞記事（毎日新聞 94，95 年：約 20 万記事）から検索するタスクである．

```
<TOPIC>
<TOPIC-ID>1007</TOPIC-ID>
<DESCRIPTION>駅伝の結果</DESCRIPTION>
<NARRATIVE>スポーツとしての駅伝の結果が報道されている記事。
競歩駅伝、クロスカントリー駅伝、予戦、海外での駅伝なども含
む。<NEG>結果の報道でなかったり、前年の結果が参照されてい
るだけのような物は含まない。</NEG></NARRATIVE>
</TOPIC>
```

図 3.1: 検索要求の例

検索要求は，DESCRIPTION（検索要求の簡潔な表現）と NARRATIVE（人間が見て可能な限り曖昧なく判断できる程度に詳細な記述．否定的な表現には¡NEG¿タグが付けられている.）の 2 種類の情報からなり，どの情報を使用してもよい．検索要求は予備試験用に 6 個，本試験用に 30 個ある．回答は，検索要求に適合る文書の ID を確信度の高い順に 300 件まで回答することができる．

適合文書の判定データは人手によって作成され，A（適合），B（一部適合），C（不適合）の 3 段階に分けられている．適合判定は，IREX の全参加者が提出した記事を集め（プーリング），そのすべての記事に対して行われている．検索要求にもよるが，1 つの検索要求あたり約 1300〜3000 記事に対して判定データがある．

3.2 知識獲得に関する研究

表現のずれの吸収には 2.2 章であげたような基本要素となる知識の獲得が不可欠である．ここでは，知識獲得に関する研究を紹介する．

3.2.1 シソーラス

同義関係や上位下位関係をまとめた言語リソースとしてまず，人手で構築されたシソーラスが考えられる．シソーラスは，語を意味によって分類し体系化したものであり，自然言語処理の様々な場面で使われている．代表的なシソーラスとして英語では WordNet[4]，日本語では日本語語彙体系 [5] がある．

WordNet はプリンストン大学で開発している英語のシソーラスである．品詞ごとに同義の語を Synset としてまとめ，さらに，上位下位関係，全体部分関係が定義されている．WordNet には全部で約 16 万語が登録されており，Synset は 12 万個である．日本語語彙体系は，NTT で開発している日本語のシソーラスで，意味体系として約 3000 のカテゴリが木構造で最大 12 段階の深さまで定義されている．単語数は全部で約 30 万語が登録されている．

これらのシソーラスを用いて表現のずれを吸収することも考えられるが，シソーラスで想定している類義のレベルが不十分であることが問題となる．例えば，日本語語彙体系では,「部屋」という意味カテゴリに「リビング」,「トイレ」,「便所」などといった語が分類されている.「トイレ」と「便所」であれば同義関係としてもよいが,「リビング」と「トイレ」では同義関係にすることはできない．

シソーラスを人手で構築するにしても，作成や保守のコストを考えると大規模なものを構築するのは無理がある．そこで，国語辞典や大規模なコーパスから上位・下位関係や同義関係を獲得することが必要となる．

3.2.2 国語辞典からの関連語の獲得

国語辞典の定義文は，見出し語の意味を言語で記述したものである．多くの場合，見出し語の上位語が用いられている場合が多い．そこで，Nakamura らは，ロングマン現代英英辞典から，定義文のパターンに注目することで上位語を抽出し，シソーラスの自動構築を行った [6]．また，鶴丸らは，新明解国語辞典から，上位下位関係，同義関係，全体部分関係，集合要素関係，関連関係などの関係を抽出した．さらに，それらの逆関係

を作成し，見出し語からその関連語を検索できるシステムを構築した [7].

3.2.3 大規模なコーパスからの言い換え表現の獲得

最近では，新聞記事や Web データなどの大規模テキストを用いて言い換え表現を自動獲得する研究が様々に行われている．

表現が違うが，内容は同じであるような文書からなるコーパスをコンパラブルコーパスと呼ぶ．コンパラブルコーパスは，同じニュースを報道している異なる新聞社の記事や，同じ物語の異なる翻訳などを集めることで得られる．Barzilay らは，コンパラブルコーパスを，Multiple-Sequence Alignment（MSA）というアルゴリズムを用いて図 3.2 のような 1 つの単語ラティスで表現し，言い換え表現を獲得した [8].

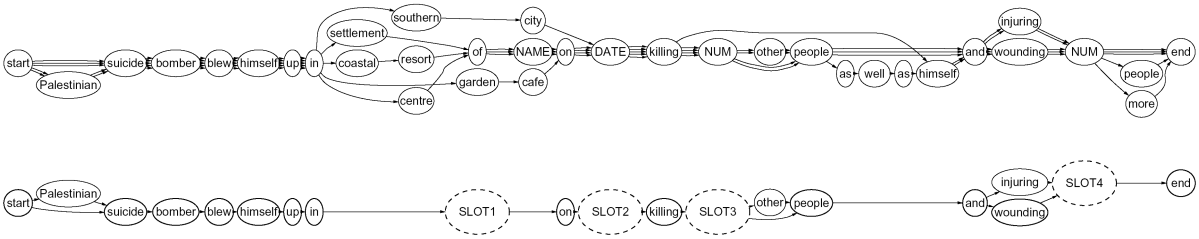

図 3.2: 単語ラティス

一方，コンパラブルでないコーパスから言い換え表現を獲得する研究もある．Lin らは，大規模コーパス中の動詞句の主語と目的語に現れる単語の分布の相互情報量を用いて，類似した動詞句を収集した [9]．Jacquemin らは，複合語に注目してコーパス中に有意に共起する内容語を含むような名詞句を抽出し，複合語の言い換えルールを獲得した [10].

こういったコーパスベースの手法では，言い換え表現を網羅的に集めることはできない．大規模なコンパラブルコーパスを収集することは簡単ではないし，頻度や分布を用いる手法は，特定の種類の言い換え表現しか得られないからである．

3.3 知識の利用に関する研究

ここでは，前節で獲得したような知識を利用し，柔軟なマッチングを行う研究について紹介する．

3.3.1 シソーラスを利用したクエリ拡張

Voorhees らは，WordNet を用いて検索語と同じ Synset に属する語や上位，下位の Synset に属する語を検索クエリに追加して検索を行った [11]．他にも既存のシソーラス

を用いて単純にクエリを拡張する手法が試されたが，検索精度はほとんど改善しなかった．その理由として，既存のシソーラスは同義関係の近さが均一でないことや，これらの研究は Bag-of-words の検索モデルであるため，どこまで展開してよいかの制御が難しいことがあげられる．

3.3.2 知識の事前展開

清田らは，図 3.3 のように知識ベースを同義表現辞書と上位・下位辞書を用いて再帰的に展開しておくことで表現のずれを吸収する質問応答システムを構築した．マッチングは係り受け関係も考慮することによって，フレーズレベルの同義関係も扱っている [12].

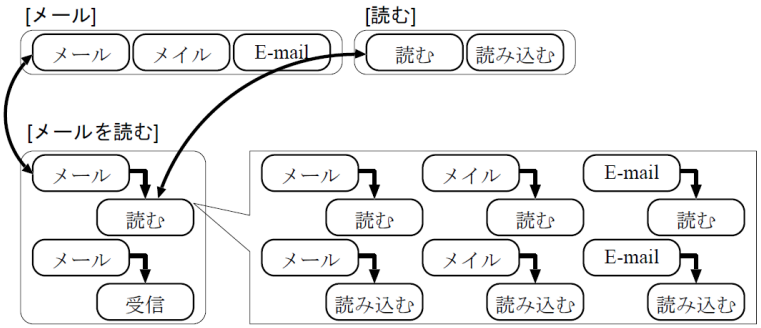

図 3.3: 知識の展開

しかし，知識を単純に展開する手法は，知識がドメインに依存した小規模なものであれば有効だが，一般的な類義関係も含むような大規模な知識に対しては，組み合わせ爆発を起こす問題がある．

3.3.3 言い換えシステムを利用

Takahashi らは，質問応答システムにおいて，図 3.4 のように質問文と検索対象を言い換えエンジン [13] を用いて相互に言い換えていき，文と文とのマッチングは Collins の Tree Kernel[14] を拡張した構文的マッチングを行うことで回答を探索する手法を提案した [15].

しかし，検索時にダイナミックに言い換えを行い，マッチングするため，計算時間の面で探索空間を絞らざるをえず，また，構文的制約が厳しすぎるために，言い換えや構文的マッチングの効果はほとんど得られなかった．

3.3.4 様々なマッチングプログラムの統合

黒橋らは，類義表現の基本要素のマッチングを行うプログラムを共有メモリをもちいて統合し，ボトムアップにマッチングを繰り返すことによって柔軟な類義性判定システ

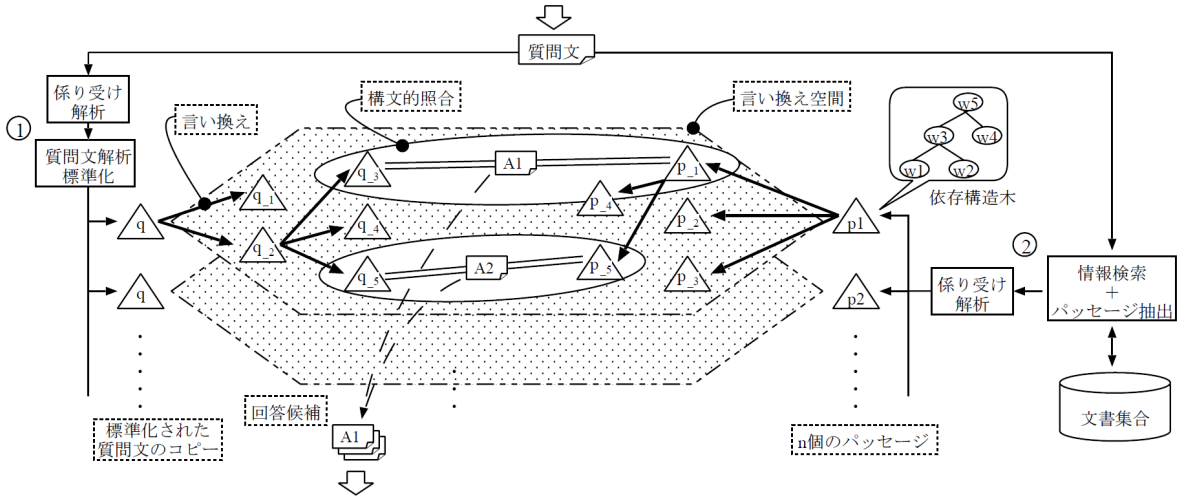

図 3.4: 言い換えによる探索

ムを構築した．しかし，情報検索や質問応答などのアプリケーションでの評価は行っていない [16].

第 4章： 類義表現データベース

2.2 章であげた同義関係や上位下位関係による表現のずれを吸収するためには，なんらかの形で同義関係や上位下位関係の知識を獲得し，利用する必要がある．同義関係や上位下位関係の知識源として既存のシソーラスを利用することが考えられるが，日本語語彙体系や分類語彙表といった既存のシソーラスは同義関係であったとしても必ずしも似た意味であるわけではなく，そのまま同義関係として利用するには適さない．また，シソーラスを手作業で構築することも考えられるが，大規模なシソーラスの構築には莫大なコストがかかり，現実的ではない．

そこで，本研究では，国語辞典や新聞記事から同義関係や上位下位関係を自動で抽出し，類義表現データベースとしてまとめる．柔軟なマッチングは，類義表現データベースを利用して行うことになる．類義表現データベースは図 4.1 のようなもので，同義関係となる語や句を集めた同義グループと，同義グループ間の上位下位関係からなる．柔軟なマッチングのために，各同義グループには ID を与え，これを SYNID と呼ぶ．本論文中では，SYNID を同義グループの一表現を＜ ＞で囲ったもので表現する．

この章では，類義表現データベースの構築手順を説明する．

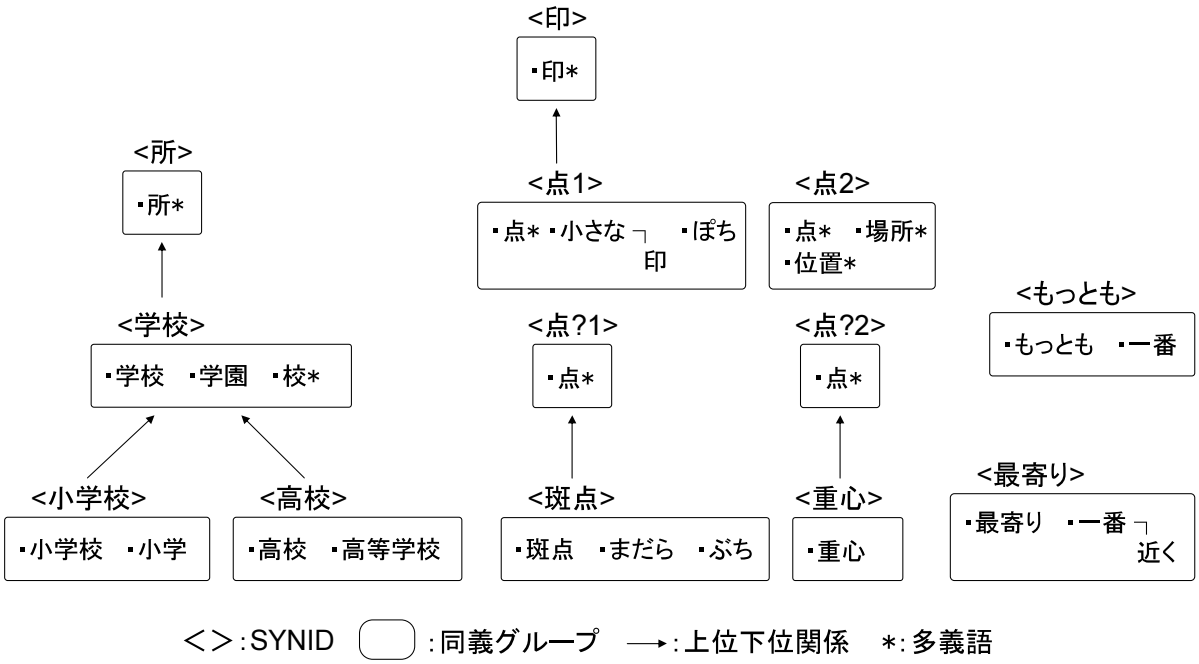

図 4.1: 類義表現データベースの例

4.1 国語辞典からの知識の抽出

国語辞典は，見出語の意味を自然言語で説明したものである．そのため見出語と定義文との関係をみることで，同義関係や上位下位関係などの様々な知識を得ることができる．本研究では見出語の品詞が代名詞を除く名詞の場合と副詞の場合に分けて国語辞典からの知識の抽出を行う．代名詞を除く名詞の場合は，4.1.1〜4.1.3 節で，副詞の場合は，4.1.4 節で説明する．

4.1.1 定義文 1 文目からの上位語，同義語，下位語の抽出

定義文から取り出すことができる最も基本的な関係として，一般的には定義文の主辞が見出し語の上位語であるという関係がある．例えば，

夕食 ： 夕方の**食事**。
重心 ： 重さがつりあって中心となる**点**。

の例では，見出語「夕食」の定義文「夕方の食事」の主辞は太線で表した「食事」であり,「重心」の場合は「点」が定義文の主辞である．このとき,「食事」が「夕食」の上位語であり,「点」が「重心」の上位語となっていることが分かる．

しかし，場合によっては文の主辞以外の語が上位語であったり，同義語，下位語が示される場合もある．これらは次のような文末パターンによって判別することができる．ここでは，下線部が上位語，同義語，下位語を判別するパターンである．

上位語

土星 ： **わく星** の一つ。
とび ： **タカ** の一種。

同義語

アイス ： 「**アイスクリーム**」の略。
大韓民国 ： 略して「**韓国**」とも言う。
朝食： **あさごはん**。（※ 定義文が一語の場合）

下位語

硬筆 ： 筆に対して、**えんぴつ**や、**ペン・ボールペン** などのこと。

このようにして，定義文 1 文目から上位語，同義語，下位語を抽出する．

4.1.2 定義文 1 文目からの同義句の抽出

辞書の定義文は見出し語の説明であり，その全体が見出し語と同義関係にあると考えられる．特に，以下の例のように，定義文が比較的短い場合には，見出し語と定義文が言い換え可能な関係にある．

夕食 ： **夕方の食事**。
最寄り ： **一番近い**ところ。

そこで，定義文が 3 文節以下の長さであれば，見出し語に対して定義文全体を同義句として抽出する．本研究では，同義関係になる句のことを同義語と区別して同義句外呼ぶ．ここで,「最寄り」の例のように定義文末尾が「こと」「ところ」などの抽象度の高い語である場合は，言い換え表現としては不要であるので削除する．こうすることによって,「最寄り（の）駅」と「一番近い駅」といった表現のずれを吸収することができる．

4.1.3 定義文 2 文目以降からの同義語の抽出

日本語の国語辞典では，定義文 2 文目以降に，同義語が示されることが多い．このような同義語は，定義文 2 文目以降で 1 語，または，複合語のものを取り出せばよい．
以上のことをまとめると，以下の「夕食」の定義文からは以下のような関係が抽出される．

夕食 ： 夕方の食事。夕飯。
夕食 → 食事 （上位下位関係）
夕食 ＝ 夕方の食事 （同義句）
夕食 ＝ 夕飯 （同義語）

4.1.4 副詞からの同義関係の抽出

見出語の品詞が副詞のとき，定義文は以下のようになっている．

もっとも ： いちばん。何よりも。
ほんのり ： ほのかにあらわれる ようす。かすかに。
こそこそ ： 人に見つからないように、かくれてする ようす。

このように，副詞の定義文は，一言で言い換えているか，文末が「…ようす」のパターンで書かれていることが多い．そこで，文末が「…ようす」である定義文以外を同義関係として抽出する．副詞に関しては上位下位関係は抽出しない．

4.1.5 類義表現データベースの構築

前節までで説明した方法で取り出される 2 項関係をまとめることにより，類義表現データーベースを得ることができる．
ここでは多義語の扱いに注意する必要がある．下のように国語辞典で複数の項目が与えられている場合，その語は多義語であると考える．

点 ： 1. 小さなしるし。ぽち。
2. 場所。位置。
…

A＝B，B＝C という関係がある場合，B に意味の曖昧性がなければ A＝B＝C という関係にまとめる．しかし，B が多義語で意味の曖昧性のある場合には，意味の違う B を通して不適切なまとまりを作る場合があるので，組み合わせ処理は行わない．同様に上位下位関係においても，A→B，B→C や A→B，C→B のときに B に意味の曖昧性があればこれらの組み合わせは行わない．

以下の例の場合，

斑点：ところどころに散らばっている**点**。まだら。ぶち。
重心：重さがつりあって中心となる**点**。

「斑点」,「重心」の上位語はどちらも「点」であるが,「点」は多義であり意味の曖昧性がある．そのため＜斑点＞→＜点＞と＜重心＞→＜点＞の関係をまとめることは行わない．マッチング対象文中に「点」という表現があれば，各同義グループが別々に与えられることになる．国語辞典の定義文中の語の多義性解消や，マッチングを行う文中の語の多義性解消は今後の課題である．

このように，多義語の場合を除いて 2 項関係の組み合わせを行うと，まず，同義語，同義句関係をまとめた同義グループができる．さらに，同義グループ間の上位下位関係をつなげることで，類義表現データーベースが作られる．

4.2 新聞記事の括弧表現からの同義関係の抽出

新聞記事中で括弧表現があった場合，括弧の前の表現と，括弧内の表現は同義関係にあることが多い．また，括弧で表現されているような同義関係は国語辞書に書かれることの少ない専門用語や省略表記であることが多い．そこで，新聞記事の括弧表現から同義関係を自動で抽出し類義表現データベースに追加する．

新聞記事の括弧表現とは以下のようなものである．

ASEAN（東南アジア諸国連合）
二酸化炭素（ＣＯ２）
時短（労働時間短縮）
企業の合併・買収（M&A）
ＨＤＴＶ（ハイビジョン）

括弧の前の表現（下線部）と括弧内の表現は，略語などの同義関係であることが多い．しかし,「一等（１０００万円)」や「ワシントン・ポスト（米)」など数値や所属を表すこともあり，括弧の前と中の表現をそのまま同義関係として抜き出すわけにはいかない．

そこで，括弧の前と中の表現を文字種ごとに分類し，頻度や双方向に参照し合っているかなどの情報でふるいにかけることで同義関係を抽出する．ここで抽出した同義関係も 2 項関係であるから，前節と同様に同義グループを作り，類義表現データベースに追加する．

括弧表現からの同義関係を追加することで，国語辞典から得られる常識的な類義関係に加えて，専門用語や省略表記なども柔軟にマッチングすることが可能となる．

第 5章： SYNGRAPH

ある文について，前節で抽出した類義表現を組み合わせ的に使えば，様々な別の，類義の文を作り出すことができる．例えば，

ホテル ＝ 旅館
最寄り ＝ 一番近い
一番 ＝ もっとも

という類義表現を,「ホテルに一番近い駅はどこですか」という文に適用すると，

ホテルに一番近い駅はどこですか
＝ ホテルにもっとも近い駅はどこですか
＝ ホテルの最寄りの駅はどこですか
＝ 旅館に一番近い駅はどこですか
＝ 旅館にもっとも近い駅はどこですか
＝ 旅館の最寄りの駅はどこですか

のように 6 通りの文をつくり出すことができる．助詞の有無・交代を考えるとさらに多くのバリエーションが生まれる．

柔軟なマッチングで必要なことは，元の文から派生する様々な表現を吸収することである．しかし，これを上の例のように事前に展開しておく方法は，類義表現の数が大規模になると組み合わせ爆発を起こしてしまい現実的ではない．また，マッチングを行う時にダイナミックに類義表現を探索していく方法も考えられるが，これも計算時間が爆発してしまうと予想される．

大規模な類義表現を扱うためには，文を効率的なデータ構造で扱う必要がある．本研究では，効率的なデータ構造として SYNGRAPH というデータ構造を提案する．

この章では，SYNGRAPH データ構造について説明し，SYNGRAPH どうしのマッチングについて説明する．

5.1 SYNGRAPHデータ構造

SYNGRAPH とは文の様々な類義表現をパックしたデータ構造である．文を SYNGRAPH で扱うことによって様々な類義表現を生成することができる．これを用いてマッチングを行うことによって柔軟なマッチングを実現する．図 5.1 は「ホテルに一番近い駅はどこですか」の文の SYNGRAPH である．

SYNGRAPH は図 5.1 で四角で表したノードとノード間の係り受け関係を表すリンクからなるグラフ構造である．図 5.1 では係り受け関係を矢印で表現している．

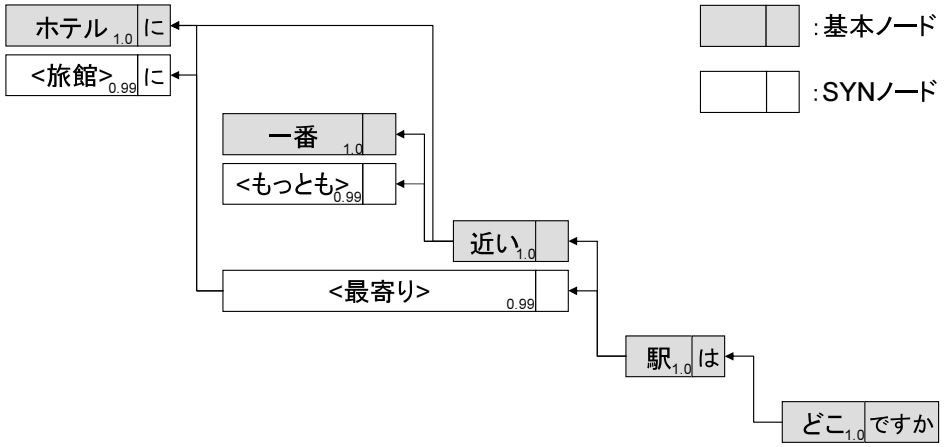

図 5.1: SYNGRAPH の例

5.1.1 基本ノードと SYN ノード

SYNGRAPH のベースとなるのは，もとの文の依存構造木であり，これは，1 つの自立語と 0 個以上の付属語からなる基本句をもとにした木構造である．SYNGRAPH もこの基本句を基本のノードとしており，これを基本ノードと呼ぶ．そして，基本ノードの自立語がある同義グループに属していれば，その同義グループの SYNID を新しい別のノードとして付与する．これを SYN ノードと呼ぶ．図 5.1 では色のついたノードが基本ノード，それ以外のノードが SYN ノードである．基本ノードを同義表現の SYN ノードと区別するのは，完全マッチと同義グループでのマッチとを区別するためである．

さらに，複数のノードに対応する表現に同義グループがあれば，その SYNID のノードも加える．図 5.1 の＜最寄り＞がこのような SYN ノードである．そしてさらに，各 SYN ノードに対して，類義表現データベースにおいて上位の同義グループがあれば，その SYN ノードを加える．このように，SYNGRAPH は元の文の依存構造木に類義表現の SYN ノードを付与して行くことで様々な表現をパックしていく．

5.1.2 ノードが持つ要素

各ノードは，以下のような要素を持つ．基本ノードと SYN ノードはでき方の違いから異なる名前で呼んでいるが，どちらも共通した要素を持ち，同じノードとしてのオブジェクトである．

ID

基本ノードの場合は基本句の自立語.
SYN ノードの場合は SYNID.
ノード間のマッチングは ID が一致するかどうかで決定する．

ノードスコア（NS: Node Score）

SYNID と元の語／句の表現とのずれに応じたスコア.
ただし，基本ノードのノードスコアは 1 である.
0〜1 までの実数値をとる.
ノードスコアの計算方法は 6.1 節で説明する.

基本ノード数
ノードがいくつの基本ノードにまたがって付与されているかの個数.

子供へのポインタ
係り受けの子供のノード（基本句）へのポインタ

付属語
基本句の付属語.
基本的には SYN ノードの付属語は基本ノードの付属語が継承される.

否定フラグ
そのノードが否定表現を含んでいるかどうか.

上位フラグ
ある SYNID の上位概念として付与された SYN ノードであるかどうか.

5.1.3 SYNGRAPH の実装

SYNGRAPH はノードの 2 重配列として実装した．まず，構文解析の最小単位である基本句レベルの配列があり，それがノードの配列となる．1 つの基本句には最低 1 つの基本ノードがあり，類義表現によって様々な SYN ノードが追加される．複数の基本ノードにまたがって付与される SYN ノードはヘッドの基本句の配列に追加した．こうすることによって，子供のノードへのポインタは，各々のノードへのポインタを持つ必要が無くなり，子供の基本句へのポインタだけですむようになる．このため，ノード追加時の手間を省くことができる.

図 5.2 では，丸い四角で囲ったものが基本句であり，基本句の中に基本ノード（灰色）や SYN ノード（白色）がある．係り受けは，各ノードから基本句へのリンクとして表現される．図 5.1 では，係り受けはノードからノードへのリンクであるので，＜最寄り＞ノードは,「ホテル」ノードと＜旅館＞ノードへの 2 つのポインタを持つ．もしこれに，＜旅館＞の上位の＜建物＞ノードを付与する場合を考えると，＜最寄り＞ノードや「近い」ノードも＜建物＞ノードへのポインタを追加しなければならない．しかし，図 5.2 では，係り受けはノードから基本句へのリンクであるので，＜建物＞ノードを「ホテル」の基本句に追加するだけでよい.

5.2 SYNGRAPH マッチング

ここでは，SYNGRAPH どうしのマッチングについて説明する.

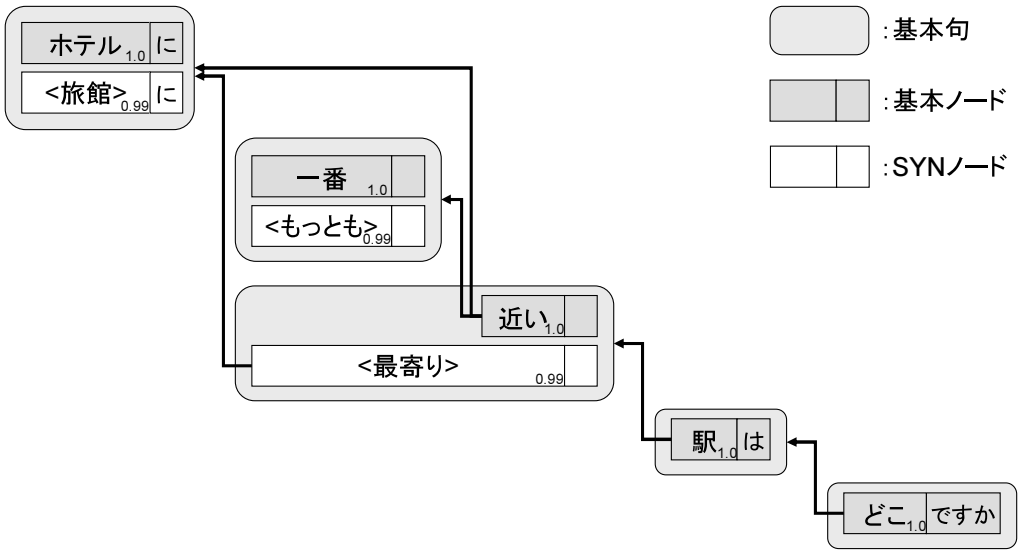

図 5.2: SYNGRAPH における係り受け関係の表現

5.2.1 ノードどうしのマッチング

SYNGRAPH どうしのマッチングの説明の前に，まず，2 つのノード間のマッチングとそのスコアを定義する．

2 つのノードが同じ ID を持ち，両方に上位フラグが立っている場合を除いて，その 2 つのノードはマッチしたと定義する．

各基本ノードには一段階上位の SYN ノードまで付与されるため，これを介して，上位下位一段階の表現のずれまでがマッチすることになる．しかし，上位フラグが両方に立っている場合，つまり，2 つのノードがシソーラス上で兄弟の関係にあるときはマッチさせない．なぜなら，兄弟の関係は意味的に近くない場合が多く，マッチさせることの副作用が大きいと考えられるからである．これについては，実験の章で考察する．

図 5.3 に「災害」とマッチする SYNID の範囲を示す．＜災難＞を通して上位概念の「災難」とマッチし，＜災害＞を通して下位概念の「地震」や「台風」とマッチする.「地震」と「台風」については，＜災害＞を共通して持つが，どちらも上位フラグ（図では下線で表した）が立っているのでマッチさせない．

2 つのノード間のマッチングのスコア（NMS: Node Match Score）は，マッチするノードのノードスコア，NS_1，NS_2 をもとに次のように計算する．

$$NMS = NS_1 \times NS_2 \times \text{付属語不一致ペナルティ} \times \text{否定不一致ペナルティ} \quad (5.1)$$

ここで，付属語不一致ペナルティは付属語に不一致があれば 0.9，そうでなければ 1 とする．ただし，マッチする SYNGRAPH のヘッド同士では常に 0 とする．つまり，ヘッド同士のマッチングは，必ず付属語が一致していなければならない．否定不一致ペナルティは，否定フラグが不一致であれば 0.6，そうでなければ 1 とする．NMS は，0 以上 1

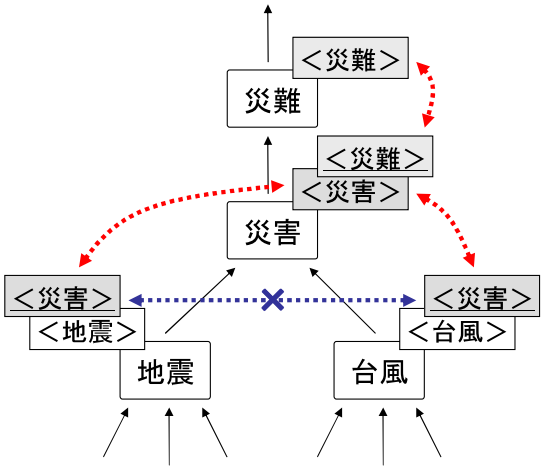

図 5.3: マッチングの範囲

以下の実数となりノード間の意味の近さを表す．2 つの基本ノードが完全一致する場合に NMS = 1 となる．

図 5.4 の例では，左側の「ホテル」ノードと右側の「ホテル」ノードの NMS は付属語の不一致があるので 0.9（= 1.0 × 1.0 × 0.9），＜最寄り＞ノードの NMS は 0.98 （= 0.99 × 0.99 × 1.0）,「駅」ノードの NMS は 1.0 となる．

5.2.2 SYNGRAPHどうしのマッチング

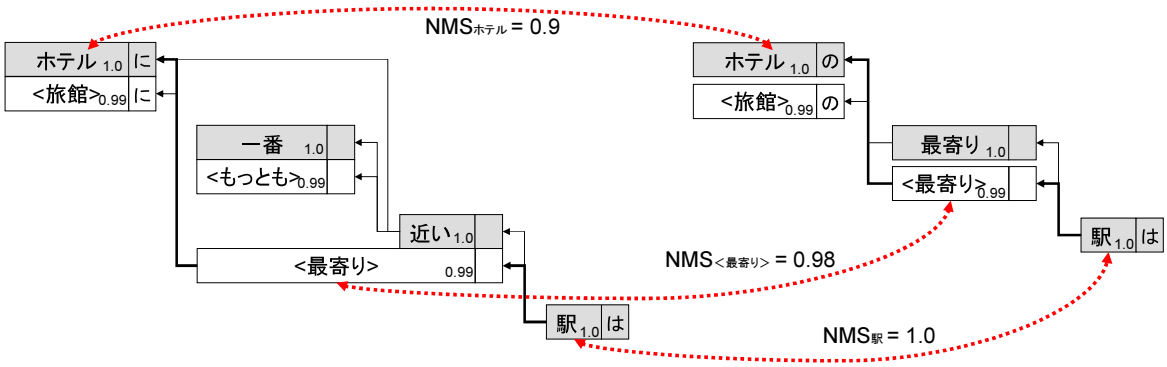

図 5.4: SYNGRAPH 全体マッチ

2 つの SYNGRAPH は，図 5.4 のように，もとの文を過不足なくカバーする同一のノード群が，同一の係り受け関係をもつ場合にマッチすると考える．このことを，SYNGRAPH 全体マッチと呼ぶことにする．さらに，SYNGRAPH_1 の部分木と別の SYNGRAPH_2 が SYNGRAPH 全体マッチしていることを，SYNGRAPH_1 に SYNGRAPH_2 が SYNGRAPH マッチしていると呼ぶことにする．

まとめると，SYNGRAPH マッチは SYNGRAPH の部分と全体とのマッチであり，SYNGRAPH 全体マッチは，相互に SYNGRAPH マッチ場合である．

SYNGRAPH マッチのスコア，SMS（SYNGRAPH Match Score）は，それぞれの NMS を部分木側の基本ノード数で重みづけ平均したもので，SYNGRAPH_1 に SYNGRAPH_2 が SYNGRAPH マッチしているときのスコア，SMS_{12} は以下のように定義する．

$$SMS_{12} = \frac{\sum(NMS \times \text{基本ノード数}_1)}{\sum \text{基本ノード数}_1} \tag{5.2}$$

図 5.4 の例では,「ホテル」，＜最寄り＞,「駅」のノードが共通しており，同一の係り受けをもつ．さらに,「ホテル」，＜最寄り＞,「駅」が元の文をすべてカバーしているので，2 つの SYNGRAPH は SYNGRAPH 全体マッチしている．このときの SMS は，$SMS_{\text{左右}} = \frac{0.9\times1+0.98\times2+1.0\times1}{1+2+1} = 0.965$，$SMS_{\text{右左}} = \frac{0.9\times1+0.98\times1+1.0\times1}{1+1+1} = 0.96$ となる．

5.2.3 SYNGRAPH マッチングのアルゴリズム

SYNGRAPH マッチするかどうかは，それぞれの SYNGRAPH のヘッドから，マッチするノードの係り受け関係を順にたどっていくことで調べられる．また，SYNGRAPH マッチには，図 5.4 で「ホテル」ノードではなく，上位の＜旅館＞ノードを対応付ける場合のように複数のノードの対応付けが考えられる場合があるが，その中から最も SMS が大きい対応付けを選択する．これは単純には組み合わせ爆発を起こすが，実際には head からノードのマッチを調べていく際に基本句ごとにスコアが最大になるものを選択していくことによって効率的に探索することができる（図 5.5）．

この探索では，ある基本句でスコアが最大でないものを選ぶことによって，最終的に全体がマッチすると分かる場合がないわけではないが，そのような例はまれである．

```
# SYNGRAPH A の TA 番目の基本句をヘッドとする部分に
# SYNGRAPH B の TB 番目の基本句をヘッドとする部分がマッチするかを調べる関数
sub match {
   引数は, (A, TA, B, TB);
   # 一致する SYNID があるか探す
   if (NA.ID = NB.ID ただし NA, NB は A[TA], B[TB] に含まれるノード) {
      NMS(= NA.NS * NB.NS * ペナルティ) が最大になる NA, NB を探す;
      基本ノード数 = NA. 基本ノード数;
      if (NA に子供が存在 and NB に子供が存在) {
         # 子供がマッチするか調べていく
         for 基本句番号 TBC (NB の子供) {
            for 基本句番号 TAC (NA の子供) {
               match(A, TAC, B, TBC);
               if (マッチした) {
                  SMS = (NMS*基本ノード数 + マッチ.SMS*マッチ. 基本ノード数)
                        / (基本ノード数 + マッチ. 基本ノード数);
                  基本ノード数 = 基本ノード数 + マッチ. 基本ノード数;
               }
               else { return マッチしない; }
            }
         }
         return マッチした;
      }
      if (B に子供が存在しない) { return マッチした; }
   }
   return マッチしない;
}
```

図 5.5: SYNGRAPH マッチのアルゴリズム

第 6章： SYNGRAPHによる柔軟マッチング

SYNGRAPH を使った柔軟マッチングによる情報検索や用例検索は図 6.1 の手順で行う．まず，類義表現データベースを SYNGRAPH に変換する．ここでは，先に述べた SYNGRAPH マッチングを繰り返して適用することにより，類義表現データベース内の相互関係を明確化する．次に，SYNGRAPH 化した類義表現データベースと検索対象との SYNGRAPH マッチングによって検索対象を SYNGRAPH に変換する．最後に，やはり類義表現データベースとの SYNGRAPH マッチングによって入力を SYNGRAPH に変換する．そして，機械翻訳の用例検索においては，さらに入力と検索対象（用例集合）の SYNGRAPH マッチングを行うことによって入力と部分一致する用例を検出する．IR においては，入力と検索対象（文書集合）の SYNGRAPH の類似度計算に基づいて文書のランキングを行う．

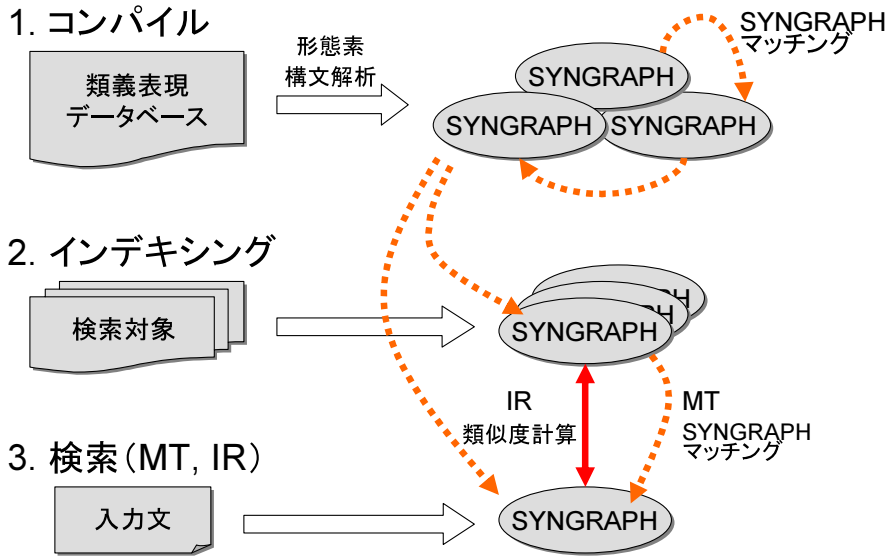

図 6.1: SYNGRAPH を使った情報検索／用例検索

6.1 コンパイル：類義表現データベースのSYNGRAPH化

類義表現データベースを SYNGRAPH 化は，以下の手順でおこなう．

ステップ 1：依存構造木への変換

まず，形態素解析，構文解析をおこなって各表現を依存構造木に変換する．形態素解析器として JUMAN[17] を，構文解析器 [18] として KNP を用いる．次に，依存構造木の基本句を SYNGRAPH の基本ノードとして登録する．これが，その表現のもっとも単純な SYNGRAPH となる．

「コンピュータ」「コンピューター」,「リンゴ」「りんご」「林檎」,「表わす」「表す」などの表記バリエーションについては，JUMAN がそれぞれの語の代表表記を出力することで吸収される．

基本ノードの ID は基本句の自立語の代表表記を用いる．付属語は基本句の助詞，助動詞，付属語の原形を用いる．ただし,「個」や「歳」などの単位になるような接尾辞は付属語にはせず，ID に含める．また，接頭辞の「不〜」や「無〜」,「〜ない（接尾辞）」や「〜ぬ（助動詞）」は否定フラグをたて，ID には含めない．

ひらがなで「かぜ」と表記したとき,「風」なのか「風邪」なのか曖昧性が出てくるが，JUMAN では曖昧性のある形態素があるときは，曖昧性の候補を出力できる．このように曖昧性のある形態素がある場合は，その候補すべてを別々の基本ノードとして登録する.「かぜ」の場合は,「風」と「風邪」の両方を基本ノードとして登録することになり,「風」にも「風邪」にもマッチすることができる．ただし，類義表現データベースでは，各表現が曖昧性のない表記で書かれているとし，1 番目の候補だけを基本ノードとする．

ステップ 2：SYN ノードの付与

類義表現データベース中の各表現どうしで SYNGRAPH マッチングを行う．SYNGRAPH マッチし，SYNGRAPH 全体マッチしない場合に，部分木側のマッチした部分のヘッドに全体がマッチした側の SYNID を ID とする SYN ノードを付与する．また，その SYNID に上位の SYNID が定義されていれば，それも SYN ノードとして付与する．

図 6.3 の例では，たとえば＜潜水＞の「水の中に潜る」の部分木「水の中」が，＜水中＞の「水の中」とマッチする．そのため,「水の中に潜る」の「中」の部分に＜水中＞ノード，さらにその上位の＜中＞ノードが与えられる．

新しく付与される SYN ノードのノードスコア（NS）は，SYNGRAPH マッチのスコア，SMS に，関係に応じたペナルティをかけたものとする．ペナルティは同義関係の場合は 0.99，上位関係の場合は 0.7 とする．先の「水の中」の例では，SMS は完全一致のため 1 となり，＜水中＞ノードの NS は 0.99，＜中＞ノードの NS は 0.7 となる．同義関係でもスコアに差をつけることで，完全一致によるマッチングと同義関係によるマッチングとを区別することができる．

上位下位関係を通してありとあらゆる語がマッチすることを避けるため，新しく付与する SYN ノードのノードスコアが 0.5 未満のものは SYN ノードを付与しない．上位下位関係のペナルティが 0.7 であるため，最高でも 1 段階上位の SYN ノードまでしか付与されない．

ステップ 2 のマッチングにおいて，SYNGRAPH 全体マッチする場合に SYN ノードを

付与しないのは，全体マッチの場合を許すと多義性の問題に注意して区別しておいた同義表現グループ間に次々と SYN ノードが芋づる式に付与されてしまい，不適切な同義関係が導かれてしまうからである．

例えば，図 6.2 で考えると，同義グループに＜軽い＞「軽い」＝「ライト」，＜光＞「光」＝「ライト」がある場合，双方の「ライト」は全体がマッチする．このとき，(1)＜軽い＞の「ライト」に＜光＞の SYN ノードを付与してしまうと，別のところで「光」を含む表現があったとき，(2)「光」に＜光＞の SYN ノードが付与されるのは良いが，＜光＞を通して＜軽い＞の「ライト」ともマッチすることになり，結局，(3)「光」に＜軽い＞の SYN ノードが付与されてしまう．つまり,「光」と「軽い」がマッチしてしまうことになる．このように，芋づる式に同義語がマッチしてしまうのを避けるために SYNGRAPH の全体どうしがマッチする場合は SYN ノードを付与しない．

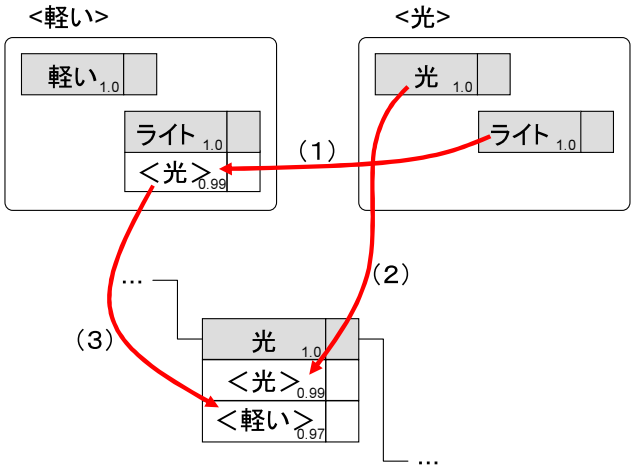

図 6.2: 芋づる式のマッチング

ステップ 3：ステップ 2 の繰り返し

ステップ 2 を繰り返し，どの同義表現にもそれ以上新たに SYN ノードを付与できなくなるまで続ける．

ある SYNGRAPH に新たに同義 SYNID が付与されていくことによって，別の SYNGRAPH ともマッチする可能性が出てくる．そのため，ここでの処理は繰り返し行う必要がある．図 6.3 の例では，まず 1 回目の繰り返しで，＜潜水＞の「水の中」の部分と，＜ダイビング＞の「水中」の部分に＜水中＞が付与される．そうすると，次の 2 回目の繰り返しで，＜ダイビング＞の「水中に潜る」と＜潜水＞の「水の中に潜る」がマッチするようになり，＜ダイビング＞の「水中に潜る」の部分に＜潜水＞ノードが付与される．

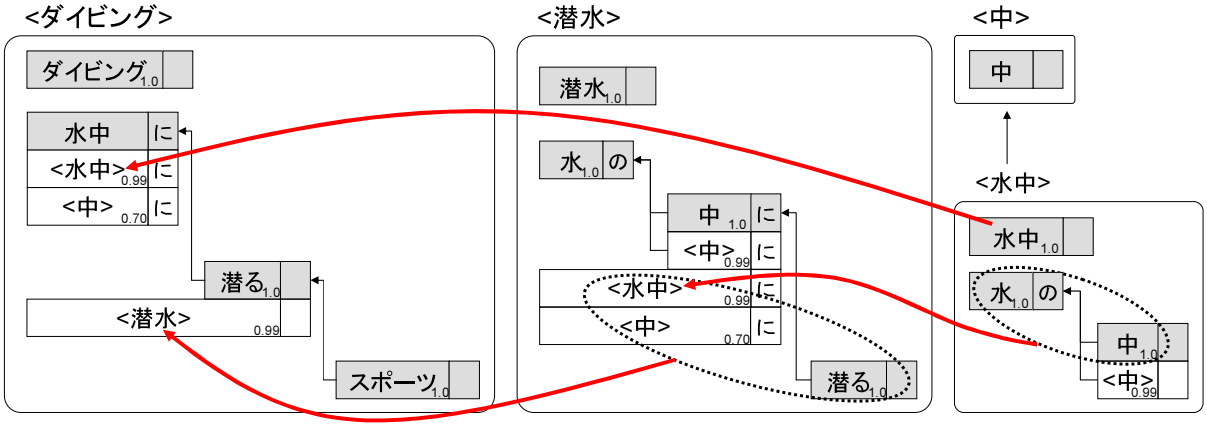

図 6.3: 類義表現データベースの SYNGRAPH 化

6.2 インデキシング：検索対象の SYNGRAPH 化

SYNGRAPH 化した類義表現データベースを参照して，検索対象の各文の SYNGRAPH 化を行う．SYNGRAPH 化された検索対象文はデータベースに保存し，検索のためのインデックスもここで作成する．

検索対象の SYNGRAPH 化は前節の類義表現データベースの SYNGRAPH 化とほぼ同じである．すなわち，まずステップ 1 で検索対象文を基本的な SYNGRAPH に変換する．次にステップ 2 で，この SYNGRAPH のあらゆる部分と，類義表現データベースの各表現の SYNGRAPH の全体との SYNGRAPH マッチングを行い，マッチすれば SYN ノードを付与し，さらに上位の同義グループがあれば上位の SYN ノードも付与する．

ステップ 1 で，曖昧性のある形態素がある場合は，その候補を全て別々の基本ノードとして登録する．こうすることによって，全ての候補とマッチできるようになる．

ステップ 2 では，類義表現データベースの各表現が SYNGRAPH 化されているので，表現の組み合わせを考える必要はなく，類義表現を効率的に見つけることができる．例えば，図 6.4 に示すように，検索対象文の「潜水するスポーツ」の部分に対して，＜ダイビング＞の SYNGRAPH とのマッチングを行うだけで＜ダイビング＞の SYN ノードを与えることができる．この時に＜水中＞や＜潜水＞にある SYNGRAPH を参照する必要はない，ただし，小さい部分のマッチが先に行われて,「潜水」に＜潜水＞ノードが与えられている必要がある．

また，インデキシングのステップ 2 では，図 6.2 のような芋づる式のマッチングが起こりえないので，SYNGRAPH 全体マッチした場合も SYN ノードを付与する．

なお，前節の処理とは異なり，ステップ 3 の繰り返しは必要ない．類義表現データベースの SYNGRAPH 化では同義表現の SYNGRAPH に相互の変化があるために繰り返しが必要であったが，ここでは，ある検索対象文への SYN ノード付与が他の検索対象文とのマッチング処理に影響しないからである．

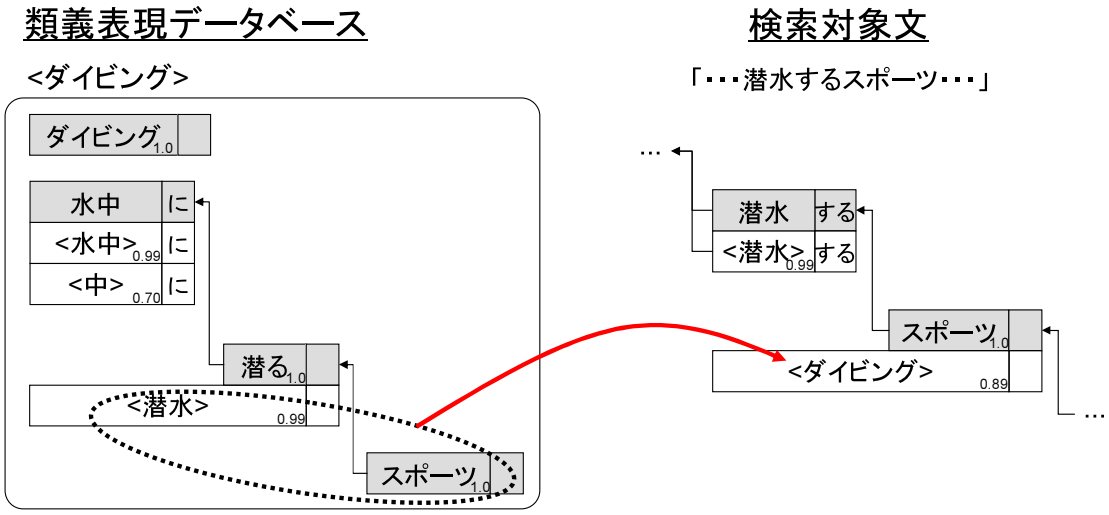

図 6.4: 検索対象の SYNGRAPH 化

6.3 機械翻訳における用例検索

用例ベースの機械翻訳では，2 言語間で対応の取れている対訳コーパスから用例を取り出し，入力文とマッチする用例を組み合わせることで翻訳を行う．入力文と用例とが柔軟にマッチすることによって，翻訳に使う用例の候補を増やすことができる．

6.3.1 MT の全体の流れ

まず，用例ベース翻訳の概要について説明する．図 6.5 に用例ベース翻訳流れを示す．用例ベース翻訳は大きく分けてアライメントと翻訳の 2 つの部分に分かれる．

アライメントは，与えられた対訳コーパスを解析し対訳間での対応関係を見つけ，対応の取れたものを用例としてデータベースに保存する作業である．対訳辞書を用いて単語ごとの対応をとり，単語ごとの対応をマージしながらフレーズ単位の対応関係を見つけていく．用例データベースには単語単位からフレーズ単位，文全体といった様々な大きさの用例が登録される．

翻訳は，入力文に近い用例を探し出し，それらを張り合わせて翻訳文を生成する作業である．まず，入力文と用例のマッチングを行い，翻訳に使う用例の候補を集める（用例検索）．次に，マッチした部分の周辺の類似度を考慮し，実際の翻訳に使われる用例を決定する（用例選択）．最後に用例を張り合わせて翻訳文を生成する（翻訳文生成）．

本研究では，用例検索における入力文と用例とのマッチングを SYNGRAPH マッチングで行う．

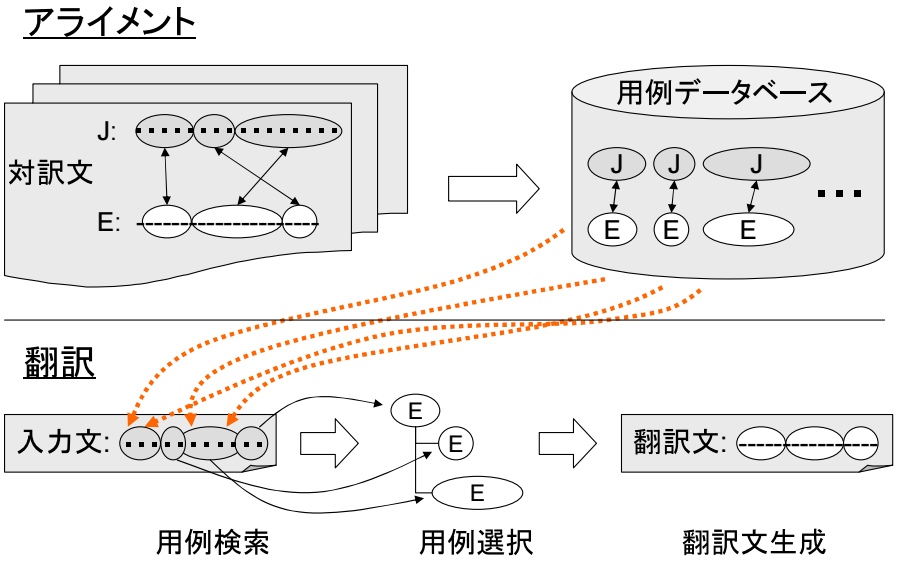

図 6.5: 用例ベース翻訳システム

6.3.2 用例検索の手順

アライメント

用例検索で SYNGRAPH マッチングを行うには，あらかじめ用例を SYNGRAPH に変換しておく必要がある．そこで，アライメントの実行時に対訳文を SYNGRAPH に変換し，用例を SYNGRAPH の形でデータベースに登録する．

用例で「2 月」や「3 時」といった数字を含む表現は,「<num> 月」,「<num> 時」として汎化して扱う．SYNGRAPH でも，元の表現からなるノードに加えて，<num> で汎化したノードも付与する．<num> は翻訳文生成の時に適切な表現に書き換えられる．

用例は，1 つの対訳文に対して単語単位から文全体までの様々な大きさをもつ部分木で，それぞれ別個のものとしてデータベースに登録される．しかし，これを 1 つ 1 つ SYNGRAPH に変換するのは同じ構造を重複して持つことになり効率が悪くなってしまう．

そこで用例データベースを図6.6のように, SYNGRAPH の実体を登録する SYNGRAPH データベースと，用例の情報を登録する用例情報データベースの 2 つに分けて扱う．SYNGRAPH データベースには対訳文の日本語側を SYNGRAPH に変換したものを登録する．用例情報データベースは，対訳文のどの部分の用例であるかの情報だけを登録する．この 2 つのデータベースを参照することによって，仮想的に部分木の SYNGRAPH をつくり出すことができる．

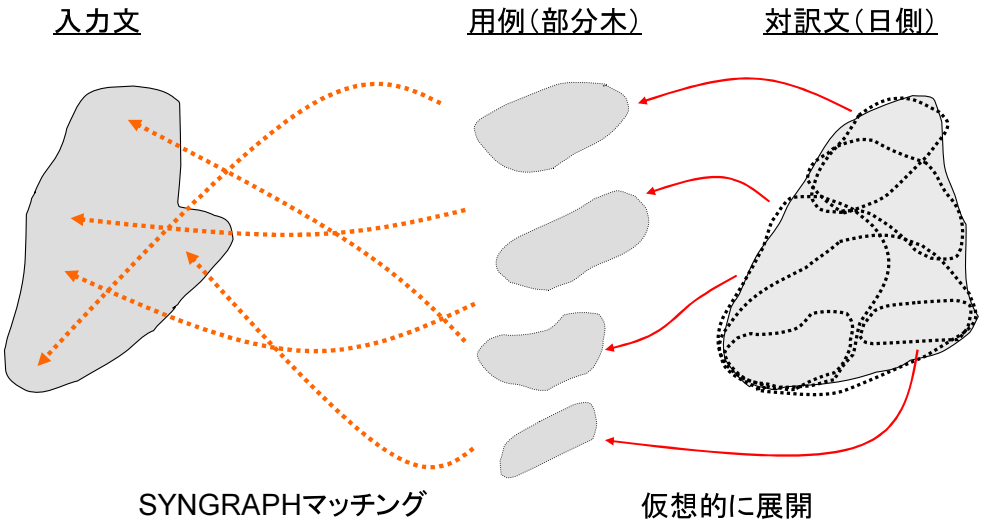

図 6.6: MT における用例の扱い

用例検索

用例検索は，入力文の SYNGRAPH のあらゆる部分と，用例集合の各 SYNGRAPH の全体とのマッチングを行い，マッチする部分を探し出す作業である．

まず入力文を SYNGRAPH 化するが，これは 6.2 節で示した検索対象の SYNGRAPH 化と同一の処理である．また，入力文の SYNGRAPH と用例の SYNGRAPH のマッチングも，5.2.3 節で説明した SYNGRAPH マッチングによって実現される．

6.2 節で説明した検索対象の SYNGRAPH 化は，検索対象の SYNGRAPH の部分と類義表現データベースとで SYNGRAPH マッチングを行い，マッチしたら SYN ノードを付与した．用例検索も同様で，入力文の SYNGRAPH の部分と用例集合とで SYNGRAPH マッチングを行い，マッチしたら用例を表すノード（用例ノード）を入力文の SYNGRAPH に付与する．用例選択では，付与された用例ノードを参照しながら翻訳に使う用例を選択する．このように SYNGRAPH を用いた用例検索は，検索対象の SYNGRAPH 化と同様に SYNGRAPH マッチングによって効率的に実現することができる．

用例検索をまとめると図 6.7 のようになる．対訳文の対応関係から用例が取り出される．用例は，対訳文の部分木であり，これは，前節で説明したように対訳文を仮想的に展開したものである．そして，入力文が与えられると，入力文と用例との SYNGRAPH マッチングを行い，マッチすれば入力文の SYNGRAPH に用例 ID が付与される．つまり，用例検索では，用例の仮想化と SYNGRAPH マッチングによって，入力文と対訳文で部分と部分のマッチングを行っていることになる．

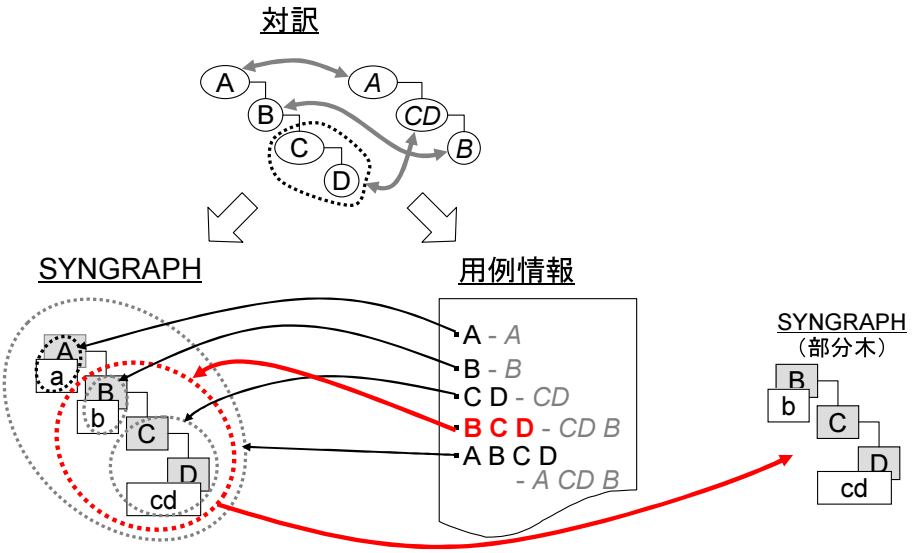

図 6.7: 入力文と対訳文とのマッチング

6.3.3 上位下位関係の利用

機械翻訳で利用する用例は入力文と非常に近い表現である必要がある．そうでなければ，用例の翻訳を入力の翻訳として用いることが不適切になる可能性があるからである．このことから，用例検索では上位下位関係のずれは用いないほうがよいと考えられる．これについては実験の節で議論する．

6.4 情報検索における文書ランキング

情報検索では，必ずしも検索文（入力文）と同義／類義の表現を探す必要はなく，検索文と近い表現を含む文書を検索できればよい．従来の IR では，単語の重要度をベースに文書のスコアを計算していたが，本研究ではこれを SYNID に拡張する．

6.4.1 単語／SYNID の重要度

文書$_i$ における 単語$_j$ または SYNID$_j$ の重要度，W_{ij} は，3.1.1 節で紹介した Okapi の BM25 での語の重要度を拡張して以下の式で計算する．

$$W_{ij} = \frac{3.0}{(0.5 + 1.5dl_i/avdl) + tf_{ij}} \times \log \frac{N - df_j + 0.5}{df_j + 0.5} \tag{6.1}$$

ただし，$W_{ij} < 0$ のとき $W_{ij} = 0$

ここで，dl_i は文書長（基本ノード数），$avdl$ は平均文書長，N は全文書数，df_j は単語$_j$ または $SYNID_j$ の出現文書数である．W の計算に用いる df，dl，N はインデキシング時にあらかじめ計算しておく．

BM25 との一番の違いは，tf の部分である．BM25 のように単語のみを考慮する場合には，それがそこに明確に存在しているので，tf を文書中の出現回数とし，単語の各々の出現を同等に扱っていた．しかし，SYNID は単語を類義表現に拡張したものであり，その存在の確からしさは各 SYN ノードによって異なり，それが 6.1 節で導入したノードスコアである．そのため，tf を出現回数とはせず，ある SYNID がその文書中で出現したときのノードスコアの総和とした．

$$tf_{ij} = \sum_{\text{文書}_i\text{中の単語}_j\text{または } SYNID_j\text{の出現 } k} NS_k \tag{6.2}$$

式 3.2 と違い，第 1 項目の分子には tf がない．これは W が，ある文書について，ある単語または SYNID1 個あたりの重要度を計算しているためである．検索文と文書間での類似度計算のときに，tf 倍分の W が足しあわせられるため，最終的なスコアは BM25 と同等となる．

6.4.2 情報検索の手順

情報検索は，次のような手順で行う．

1. 検索文の SYNGRAPH 化
2. 粗い類似度計算による絞りこみ
3. マッチング
4. ランキング

検索文の SYNGRAPH 化

まず，検索文を SYNGRAPH に変換する．これは，6.2 節で示した検索対象の SYNGRAPH 化と同一の処理である．これによって，検索文と検索対象文書がともに SYNGRAPH で表現されることになる．

粗い類似度計算による絞りこみ

情報検索では，検索文の SYNGRAPH と検索対象文書の SYNGRAPH との間で類似度計算を行うことになるが，検索対象文書すべてと類似度計算を行うのは計算時間の面から現実的ではない．そこで，BM25 と同様の計算で粗い類似度計算を行い，その後のマッチング処理のための絞りこみを行う．

検索文と 文書$_i$ との 粗い類似度$_i$ は以下の式で計算した.

$$\text{粗い類似度}_i = \sum_{\text{検索文の基本句}_t} \max_{t\text{にあるノード}_n\text{について}} \left(W_{in} \times tf_{in} \times NS_n \times \text{基本ノード数}_n\right) \quad (6.3)$$

この計算は，あらかじめ，単語／ SYNID から文書への転置インデックスを作成し，$W_{in} \times tf_{in}$ を計算しておくことで高速に計算できる．同じ基本句で同義関係や上位下位関係などで複数のノードがマッチする場合は，スコアが最大になるものだけを加算する．これは，多義語などのような基本ノードに加えて多数の SYN ノードを持つ語に多くのスコアを与えてしまうことを防ぐためである．また，検索文のノードの基本ノード数をかけているのは，複数ノードにまたがって付与される SYN ノードに大きな類似度を与えるためである．SYNID による拡張がない場合は，$NS = 1$ となり，tf が 文書$_i$ での出現回数となるから，これは BM25 と同じ類似度となる．

マッチング

検索文と検索対象文書とのマッチングを行い，類似度を計算する．マッチングは文書に含まれる文ごとに行い，文類似度を計算する．文類似度を合計して検索文と検索対象文書との類似度を計算する．

文類似度は，マッチするノードペアすべてについて，NMS に W やクエリ側の基本ノードをかけ合わせたスコア（IRMS）を計算し，合計したものをベースとする．さらに，マッチするノード間に係り受け関係がある場合には，IRMS をもとにした加点を行う．

情報検索は機械翻訳の用例検索と違い，質問文の表現と一致する表現を探す必要はない．そのため，ここでの NMS の付属語不一致ペナルティを 1 にして付属語の不一致は無視した．

$$\begin{aligned} \text{類似度} &= \sum_{\text{文書に含まれる文}} \text{文類似度} \\ \text{文類似度} &= \sum_{\text{マッチするノードペア } k} IRMS_k \\ &\quad + K_{\text{係}} \sum_{\substack{\text{マッチし，係り受け} \\ \text{関係にあるノードペ} \\ \text{ア } h \text{ と } m}} (IRMS_h + IRMS_m) \end{aligned} \quad (6.4)$$

$$IRMS_k = NMS_k \times W_k \times \text{クエリ側の基本ノード数}$$

ここで，$K_{\text{係}}$ は係り受け関係のスコアに対する変数である．

粗い類似度計算の時と同様に，ある基本句において複数のノードがマッチする場合には，それらの中からもっとも類似度が高くなるノードを選択して加算する．どのノード

が一番スコアが大きくなるかは係り受けの一致にも係わってくるので，図 6.9 のように再起的なアルゴリズムで計算する．

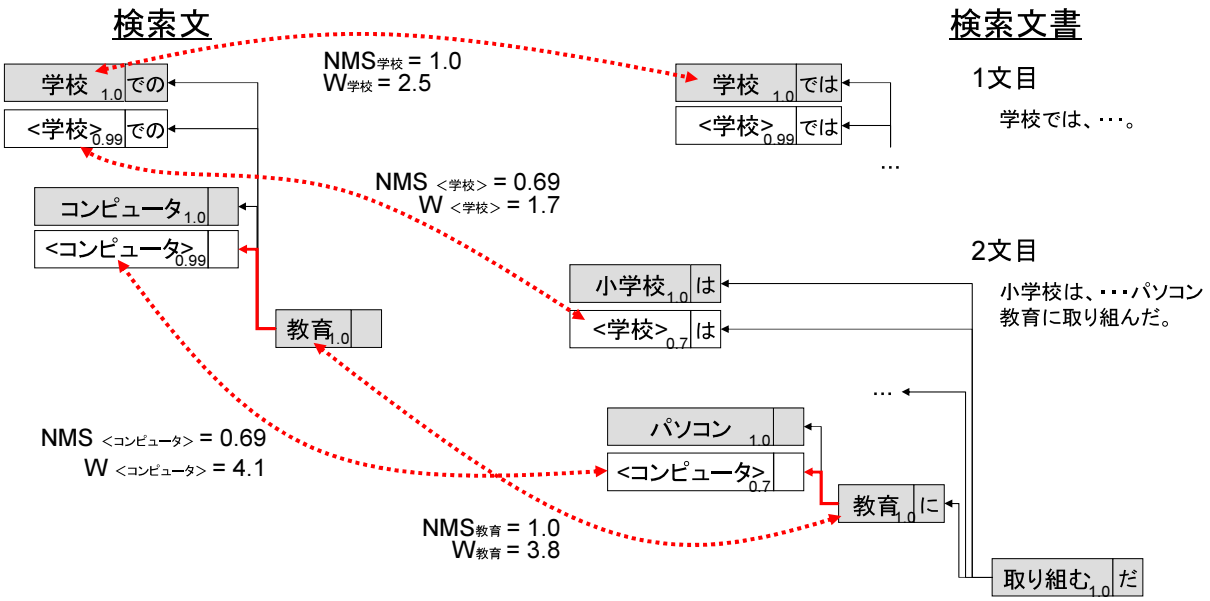

図 6.8: IR におけるマッチング

図6.8の例では，1 文目は「学校」と<学校>がマッチするが，$IRMS_{学校} = 1.0 \times 2.5 \times 1 = 2.50$，$IRMS_{<学校>} = 0.98 \times 1.7 \times 1 = 1.67$ となり，「学校」のほうが類似度が大きくなるので「学校」が選ばれる．2 文目は，<学校>と<コンピュータ>–「教育」の部分がマッチする．このとき，$IRMS_{<学校>} = 0.69 \times 1.7 \times 1 = 1.17$，$IRMS_{<コンピュータ>} = 0.69 \times 4.1 \times 1 = 2.83$，$IRMS_{教育} = 1.0 \times 3.8 \times 1 = 3.80$ となる．$K_{係} = 0.5$ とすると，この文書の類似度は，

$$\begin{aligned} 類似度 =& IRMS_{学校} + IRMS_{<学校>} + IRMS_{<コンピュータ>} + IRMS_{教育} \\ &+ 0.5 \times (IRMS_{<コンピュータ>} + IRMS_{教育}) \\ =& 13.6 \end{aligned} \tag{6.5}$$

となる．

ランキング

マッチングで計算した類似度に基づいて文書をランキングし，上位の文書を検索結果として出力する．

```
# 検索文 Q の qt 番目の基本句をヘッドとする部分と文 S との類似度
sub matching {
   引数は，Q，qt，S，係り先のスコア Kscore;

   foreach qn (Q[qt]) {
      if (qn とマッチするノードが S に存在) {
         foreach sn (S の qn とマッチするノード) {
            IRMS = NMS(qn, sn) * W * qn.基本ノード数;
            類似度 = IRMS * tf(S, sn) / sn.NS;
            if (係り受けもマッチ) {
               類似度 += K 係 * (Kscore + IRMS);
            }
            if (qn の子供がある) {
               foreach qnc (qn の子供) {
                  類似度 += matching(Q, qnc, S, IRMS);
               }
            }
         }
         類似度が一番良いノードを残す;
      }
      # マッチするノードがない場合
      else {
         if (qn の子供がある) {
            foreach qnc (qn の子供) {
               類似度 = matching(Q, qnc, S, 0);
            }
         }
      }
   }
   return 類似度が一番良いもの;
}
```

図 6.9: IR におけるマッチングのアルゴリズム

第 7章： 実験と考察

SYNGRAPH を用いたマッチングの有効性を示すために，マッチングのための知識を国語辞典と新聞記事から獲得し，機械翻訳と情報検索の 2 つのタスクで実験を行った．

7.1 知識の獲得

国語辞典（例解小学国語辞典：見出し語，約 3 万語）[19] と，新聞記事（毎日新聞 12 年分，読売新聞 14 年分：約 2,500 万文）から類義表現データベースを構築した．表 7.1 に示すように大規模な類義表現データベースが得られた．

表 7.1: 構築した類義表現データベース

	国語辞典	新聞記事
同義グループ数	15,988	1,405
1 同義グループ内の平均表現数	1.8	2.4
同義グループ間の上位下位関係数	11,446	–

この類義表現データベースのコンパイルにかかった時間は，約 30 分程であった．コンパイルによって，1 つの基本ノードについて平均 1.6 個の SYN ノードが付与されていた．

7.1.1 考察

国語辞典からの知識獲得，類義表現データベースのコンパイルについて以下のような問題点があった．

巨大な同義グループ

4.1.5 節で説明したように，多義語を介して同義でない語が同じ同義グループにまとめられてしまうことはない．しかし，別見出しではなく同じ見出しに同義語が連続して書かれているものによって，同義とはいえないような多くの語が 1 つの同義グループにまとめられてしまう例があった．

例えば国語辞典で，以下のように定義されていると，

> 手筈 ： 準備。手順。
> 手順 ： 段取り。

寸法 ： 1. 長さ。
2. 計画。段取り。

「手筈」や「寸法$_2$」は，定義文で同義語が連続して書かれている.「寸法」は多義語であるので,「長さ」と「計画」は別々の同義グループに分かれるが，それ以外の「手筈」,「準備」,「手順」,「段取り」,「寸法$_2$」,「計画」,「段取り」は 1 つの同義グループにまとまる.

このようにして，非常に巨大な同義グループができる例があり，最大のものは 50 個が 1 つの同義グループとなり表 7.2 のような語が含まれていた．実験に使用した類義表現データベースには，アドホックではあるが，表 7.2 の同義グループは含めなかった.

表 7.2: 巨大な同義グループ

アイデア	プラン	案	意見	意向	意思	意図	奥底	下拵え
階	企て	企画	筋道	具合	系統	計画	見込み	御膳立て
工夫	巧み	考え	催し	思い	思い付き	思考	思惑	支度
次第	手順	手配	手筈	準備	順序	所見	心	寸法
正気	正体	声	段取り	都合	念頭	備え	腹	本意
本心	目算	目論み	用意	略				

これは,「手筈：準備。手順。」や「寸法$_2$：計画。段取り。」のように定義文で同義語が連続してあげられているときに，それらを同義として扱ったことが原因となっている.「手筈」＝「準備」はよいが,「準備」＝「手順」とは必ずしも言えない.

これを解決するには，定義文から得られる同義関係を上位下位関係のように方向つきで考える必要があるが，これについては，今後の課題である．方向つきで考えることで,「手筈」⇒「準備」,「手筈」⇒「手順」となり,「準備」と「手順」のマッチングには 2 重のペナルティがかかるようになる.

ひらがなの解析

小学生用の国語辞典を用いたため定義文がひらがなで書かれていることが多い．例えば,「腹立ち：おこる（こと）。」からは,「腹立ち」＝「おこる」の同義関係が得られるが，形態素解析で「おこる」が「起こる」になり,「腹立ち」＝「起こる」となる．ひらがなの曖昧性ついては，定義文の文脈を見ることによってある程度解消できると考える．また，どうしても曖昧性解消ができないものについては，人手で漢字を与えることで解決する.

本手法は，辞書の解析で形態素・構文解析を行い，さらにコンパイルでも形態素・構文解析を行うため，2 度手間な部分がある．ひらがなの曖昧性解消を行うためにも，辞書の解析からコンパイルを一貫して扱う必要がある.

括弧表現の同義グループ

新聞の括弧表現から抽出された同義関係は，単純にマージして同義グループを作ったが,「ＣＤ＝コンパクトディスク」,「ＣＤ＝キャッシュディスペンサー」のように英数字による略字によって全く違うものが 1 つの同義グループにまとめられてしまった例があった．

2 項関係から同義グループにまとめるのは注意して行う必要がある．この場合も，共起情報などを利用して似たような文脈で使われるものだけをまとめる必要がある．

7.2 機械翻訳での評価

7.2.1 IWSLT

機械翻訳の評価型ワークショップである IWSLT'05[20] の日英翻訳タスクを用いて，本研究室で開発している用例翻訳システムをベースとして実験を行った [21]．IWSLT では旅行ドメインを対象とし，トレーニングセットとして 20,000 文ペアの対訳コーパス，テストセットとして 506 文の日本語文が与えられる．

評価は，各テスト文に対する 16 個の英訳文に基づく NIST[22] と BLEU[23] を用いた．NIST は英訳文との n-gram の適合率の相加平均，BLEU は英訳文との n-gram の適合率の相乗平均である．NIST は単語訳の正確さを，BLEU は語順の正しさを重視したスコアである．

実験は，類義表現データベースの違いによって，

同義×上下×：空の類義表現データベースを使用（ベースライン）
同義○上下×：同義関係のみ使用
同義○上下○：同義関係に加えて上位下位関係も使用

の 3 通りの条件で行った．

7.2.2 実験の結果

表 7.3: IWSLT 結果

類義表現 DB	NIST	BLEU
同義×上下×	8.630	0.422
同義○上下×	8.734	0.428
同義○上下○	8.689	0.425

実験結果を表 7.3 に示す．同義関係のみを使用する場合が最もスコアが高く，ベースラインとの相対スコアで NICT が 1.2%，BLEU が 1.4%の，統計的に有意な向上がみられ

た（$p < 0.05$）．この時に利用された完全一致でない用例の例を表 7.4 に示す．国語辞典から常識的な同義関係を抽出できたために，完全一致では見つけることのできなかった多くの用例を検索することができた．また，SYNGRAPH が依存構造木をもとにした構造であるため，図 7.1 のような構文レベルの表現のずれも吸収することができた．

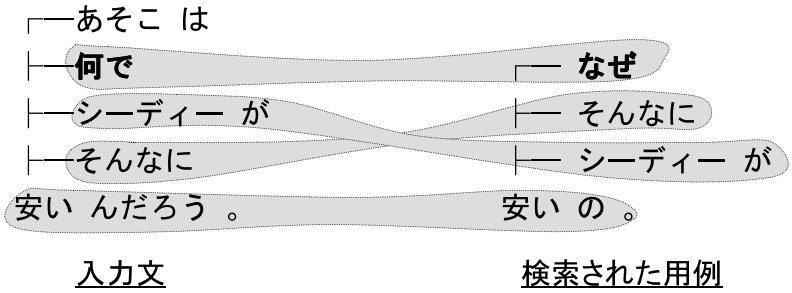

図 7.1: 構文レベルのマッチングの例

一方，上位下位関係も使用した場合は，同義関係のみを使用した場合に比べてスコアが下がった．これは，翻訳においては上位下位関係の用例の訳を利用することは必ずしも妥当ではないことを示している．例えば,「デパート」に対して，その上位語「店」の翻訳 “store” が使われるなどしたことが，スコア低下の原因となっていた．

表 7.4: 用例検索で柔軟にマッチできた例

入力文	使用された用例
婦人 用トイレは…	**女子** 用トイレは…
この**近く**に…	この**辺り**に…
いいえ，**特に**ありません	いいえ，**何も**ありません
日本語の**案内書**は…	日本語の**ガイドブック**は…
何でシーディーがそんなに安いんだろう	**なぜ**そんなにシーディーが安いの
簡単にいえば，…	**つまり**，…

7.2.3 考察

翻訳スコアがベースラインよりも悪くなった例について分析すると，以下のような失敗原因があった．

翻訳システム側の問題

用例検索で柔軟なマッチングができていても，それ以外のアライメント，用例選択，翻訳文作成の部分で失敗した例があった．これは，翻訳システム側の問題である．

アライメントでは，対訳辞書を用いて対応する部分を探していく．現在は，辞書でのマッチングは完全一致で行っているが，これも SYNGRAPH を用いて柔軟なマッチングを行うことで，より正確なアライメントが得られる可能性がある．用例選択では，現在でも用例の周辺の類似度を考慮して使用する用例を選んでいるが，さらに，全体的な文脈を考慮する必要がある．また，翻訳候補の確率を導入して最適な用例を選択するという手法も考えられる．[24]

多義語による誤り

SYGRAPH は，意味の曖昧性の解消は行っておらず，曖昧性のある語や多義語は，候補全てについてノードを付与する．これによって，様々な表現とマッチできるようになる．しかし，用例選択では，多義語が本来の意味と違う表現とマッチし，それによって翻訳結果が間違ってしまう例が見られた．

完全一致と同義グループと通してのマッチングでは，完全一致のほうがスコアが大きくなり優先されるが，用例の大きさや周りの状況によっては用例選択で完全一致が負ける場合もある．

例えば,「車を手配してドアの所につけましょう」という入力文の「所」の部分が,「日本大使館の住所を調べて下さい」の「住所」とマッチする．マッチした部分だけを見ると，必ずしも間違っているとはいえないが，この文脈で「住所」の訳を使うのはおかしい．

これらの解決のためには，形態素解析，構文解析での曖昧性解消や，文脈を考慮した用例選択を行う必要がある．

7.3 情報検索での評価

7.3.1 IREX

情報検索の評価型ワークショップである IREX のデータセットを用いて実験を行った．

3.1.2 節で説明したように，IREX のそれぞれの検索要求には，DESCRIPTION（短い検索文）と，NARRATIVE（長い検索文）の 2 つの検索文があるが，今回はその両方を用いて実験を行った．ただし，DESCRIPTION は NARRATIVE の 2 倍の重みを与えた．

正解データは適合の度合によって A（適合），B（部分的に適合），C（不適合）の 3 段階に判定されているが，今回は A と判定されたものだけを正解とし，評価は R-precision で行った．R-precision は，ある検索課題において，関連する記事の数が R だとすると，検索結果の上位 R 件の中にどれだけ正解が含まれているかという割合である．

実験は，類義表現データベースの違いと係り受け関係の有無によって，

係受×同義×上下×：BM25（ベースライン）
係受○同義×上下×：空の類義表現データベースを使用，$K_{係} = 0.5$
係受○同義○上下×：同義関係のみ使用，$K_{係} = 0.5$
係受○同義○上下○：同義関係に加えて上位下位関係も使用，$K_{係} = 0.5$

の 4 通りの条件で行った.

7.3.2 実験の結果

表 7.5: IREX 結果

手法	利用する情報	R-precision
IREX の 1 位	–	0.493
BM25	係受×同義×上下×	0.474
提案手法	係受○同義×上下×	0.492
	係受○同義○上下×	0.509
	係受○同義○上下○	0.514

実験結果を表 7.5 に示す．参考として IREX において 1 位の成績のシステムのスコアを示した．BM25 のベースラインに対して，係り受け関係を用いることでスコアが改善し，類義表現を利用するとさらに改善がみられた．情報検索においては，機械翻訳タスクの場合とは異なり，同義関係に加えて上位下位関係を利用する場合が一番スコアが高く，ベースラインと比べて相対スコアで 8.2%の統計的有意な向上がみられた（$p < 0.05$）.

このタスクにおいて類似度計算に寄与した類義表現の例を表 7.6 に示す．IREX は新聞記事を検索対象としているため,「労働時間短縮」＝「時短」のような括弧表現からの同義関係が有効であった.

表 7.6: 情報検索で柔軟にマッチできた例

検索文	文書中の表現
学校 に **コンピュータ** を導入	**小学校** に **パソコン** 導入…
偽札 事件	大量の **ニセ札**，…
労働時間短縮	…の**時短** 運動を展開
米国三大自動車メーカー…	**ビッグスリー**…
… **自動車** 販売…	… **乗用車** を販売…

7.3.3 考察

正解データ

IREX の正解判定データはプーリングされた文書にしかない．IREX の参加システムがまったく抽出することができなかった文書は，いくら検索要求に適合した文書であっても正解判定がないため，不適合文書としてカウントされる.

図 7.2 は，そのような未判定文書の例である.「偽札」と「ニセ札」の単純な表記ゆれだが，未判定文書であった.

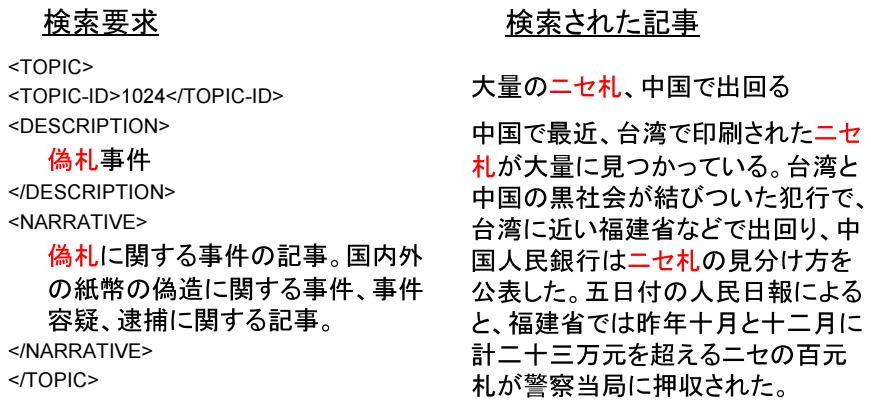

図 7.2: 未判定文書の例

係り受け関係をみる効果

係受〇同義〇上下〇のシステムについて，$K_{係}$ の値を変えて実験を行ったところ，図 7.3 のようになった．$K_{係} = 0.5$ のときが最も精度がよく，係り受け関係をみることが検索精度の向上に有効であることがわかる．$K_{係} > 0.5$ で精度が下がっているのは，適合文書が必ずしも検索要求の表現と同じ係り受け関係を持っていないために，係り受けの得点を得られない正解文書のラインキングが相対的に下がってしまうためだと思われる.

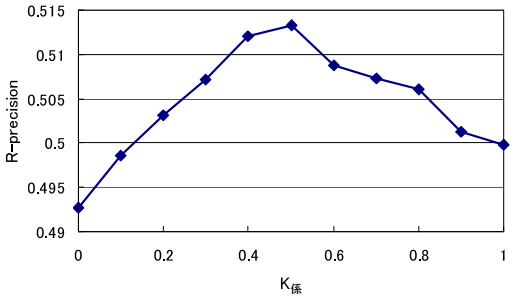

図 7.3: 係り受け関係の効果

兄弟関係のマッチング

5.2.1 節でのべたように，本研究では，シソーラスで兄弟の関係にある語はマッチしないようにした．兄弟関係のマッチを許した場合,「駅伝」と「競馬」が「競走」の下位概念になっているため「駅伝の結果」という検索要求に対して「競馬の結果」がマッチす

ることになる．兄弟関係のマッチは上位下位関係のペナルティが 2 重にかかるため NMS は小さくなるが，この例では，係り受け関係もマッチするため係り受けの得点も加算され，上位にランクされる．

兄弟関係でもシソーラスの深さによってマッチしても良いと考えられるものもあるが，副作用の方が大きかった．これは，上位下位関係のペナルティを 0.7 と固定しているからである．意味の近さや遠さを反映したペナルティを設定することで適切な類似度を計算できるようになると考える．

計算時間

1 つの検索要求に対して，平均すると約 1 分程度の時間がかかる．これは，マッチングのために SYNGRAPH をデータベースから取得し，類似度を計算しているためである．

しかし，情報検索の類似度計算は係り受け関係いがいは BM25 と同等で，係り受け関係も 2 ノード間の関係しかみていない．個々の文の SYNGRAPH を取得して類似度を計算する必要はなく，各ノードやノード間の係り受けを展開して転置インデックスを作成しておけばよい．係り受けを展開するためデータベースサイズは大きくなるが，高速に検索できるようになると考える．

第 8章： 結論

本論文では，自然言語の重要な特徴である表現のずれを取り扱うために，柔軟なマッチングを実現する方法を提案した．柔軟マッチングのためには，知識源と枠組みの問題がある．前者については国語辞典や新聞記事からの自動抽出を行い，網羅性のある類義表現を獲得した．後者については SYNGRAPH というデータ構造を導入し，SYNGRAPH による柔軟なマッチング手法を提案した．機械翻訳における用例検索と，情報検索における類似文書ランキングにこの手法を適用し，実験を行ったところ，提案手法の有効性を確かめることができた．

今後の展望として，知識獲得に関しては，辞書からの関係抽出において語の曖昧性解消の導入や，同義関係や上位下位関係間への距離尺度の導入がある．また，マッチングに関しては，機能的表現のずれや構文的なずれのある類義表現の取り扱いなどがある．

謝 辞

本研究を行うにあたり，熱心に御指導下さいました黒橋禎夫助教授に深く感謝いたします．
学術研究指導員の河原大輔氏には，様々な御意見，御指導をいただきありがとうございました．学術研究指導員の岡本雅史氏には研究会などで様々なアドバイスをいただきありがとうございました．
また，研究室の皆様には，研究をすすめるにあたって様々な協力をいただきました．辰巳正治君には，国語辞典から抽出した同義関係，上位下位関係データを提供していただきました．笹野遼平君には，新聞記事から抽出した同義関係データを提供していただきました．中澤敏明君には，機械翻訳システムについて教えていただきました．本当にありがとうございました．

参 考 文 献

[1] S. E. Robertson, C. J. van Rijsbergen, and M. F. Porter. Probabilistic models of indexing and searching. In *SIGIR '80: Proceedings of the 3rd annual ACM conference on Research and development in information retrieval*, pp. 35–56, 1981.

[2] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 232–241, 1994.

[3] S. E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *the third Text REtrieval Conference (TREC-3)*, 1994.

[4] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, , and Katherine J. Miller. Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography (special issue)*, Vol. 3, No. 4, pp. 235–312, 1990.

[5] 池原悟, 宮崎正弘, 白井諭, 横尾昭男, 中岩浩巳, 小倉健太郎, 大山芳史, 林良彦. 日本語語彙大系 CD － ROM 版. 岩波書店, 1999.

[6] Junichi Nakamura and Makoto Nagao. Extraction of semantic information from an ordinary english dictionary and its evaluation. In *Proc. of the 12th COLING*, pp. 459–464, 1988.

[7] Hiroaki Tsurumaru, Toru Hitaka, and Sho Yoshida. An attempt to automatic thesaurus construction from an ordinary japanese language dictionary. In *Proc. of the 11th COLING*, pp. 445–447, 1986.

[8] Regina Barzilay and Lillian Lee. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL 2003*, pp. 16–23, 2003.

[9] Dekang Lin and Patrick Pantel. Discovery of inference rules for question answering. *Natural Language Engineering*, Vol. 7, No. 4, pp. 343–360, 2001.

[10] Christian Jacquemin, Judith L. Klavans, and Evelyne Tzoukermann. Expansion of multi-word terms for indexing and retrieval using morphology and syntax. In *35th Annual Meeting of the Association for Computational Linguistics*, pp. 24–31, 1997.

[11] Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *SIGIR*, pp. 61–69, 1994.

[12] 清田陽司, 黒橋禎夫, 木戸冬子. 大規模テキスト知識ベースに基づく自動質問応答 - ダイアログナビ-. 自然言語処理, Vol. 10, No. 4, pp. 145–176, 7 2003.

[13] TAKAHASHI Tetsuro, IWAKURA Tomoya, IIDA Ryu, FUJITA Atsushi, and INUI Kentaro. Kura: A transfer-based lexico-structural paraphrasing engine. In *The 6th Natural Language Processing Pacific Rim Symposium, Workshop on Automatic Paraphrasing: Theories and Applications*, pp. 37–46, 2001.

[14] M. Collins and N. Duffy. Convolution kernels for natural language. In *Neural Information Processing Systems*, pp. 625–632, 2001.

[15] Tetsuro Takahashi, Kozo Nawata, Shinya Kouda, and Kentaro Inui. Seeking answers by structural matching and paraphrasing. In *NTCIR Workshop 3 Question Answering Challenge (QAC)*, 2002.

[16] 黒橋禎夫, 酒井康行. 日本語表現の柔軟な照合. 言語処理学会 第 7 回年次大会 発表論文集, pp. 343–346, 2001.

[17] 黒橋禎夫, 河原大輔. 日本語形態素解析システム JUMAN version 4.0 使用説明書. 東京大学大学院情報理工学系研究科, 2003.

[18] 黒橋禎夫. 日本語構文解析システム KNP version 2.0 b6 使用説明書. 京都大学大学院情報学研究科, 1998.

[19] 田近洵一. 例解小学国語辞典. 三省堂, 1997.

[20] IWSLT. http://www.is.cs.cmu.edu/iwslt2005/, 2005.

[21] Sadao Kurohashi, Toshiaki Nakazawa, Kauffmann Alexis, and Daisuke Kawahara. Example-based machine translation pursuing fully structural nlp. In *International Workshop on Spoken Language Translation (IWSLT'05)*, pp. 207–212, 2005.

[22] G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *HLT*, pp. 257–258, 2002.

[23] K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL 2002*, pp. 311–318, 2002.

[24] Eiji Aramaki, Sadao Kurohashi, Hideki Kashioka, and Hideki Tanaka. Probabilistic model for example-based machine translation. In *MT Summit X*, pp. 219–226, 2005.