日本語の技術文書における技術用語に着目したダイナミックプログラミングでの検索方法

山本英子[†] 山本幹雄[††]
梅村恭司[†] Kenneth W.Church[†††]

本稿では,技術文書でよく用いられる「機械翻訳システム」のような技術用語を念頭に置き,ダイナミックプログラミングに基づく新しい類似度を提案する.また,提案する方法を辞書を用いる方法(BD法)とngramを用いる方法(BN法)の二つの基本システムと比較する.これらの二つの基本システムはどちらとも標準的な頻度と情報量の内積の類似度を利用するが,対象となる語彙が異なる.BD法は日本語のテキストを既存の辞書に基づくプログラムを使って分割した単語を用いるのに対し,BN法はすべての質問の部分文字列を用いる.本研究では,技術的な日本語アブストラクトのテストコレクションにおいて,提案したダイナミックプログラミングを用いる方法(DP法)がBN法とBD法と比べ,最も効果的であることを報告する.また,質問が一つの長い技術用語を含み,その用語を構成する単語がどれも特によいキーワードでないとき,DP法が比較的に強いことを述べる.

Dynamic Programming: A Method for Taking Advantage of Technical Terminology in Japanese Documents

Eiko Yamamoto,[†] Mikio Yamamoto,[††] Kyoji Umemura[†]
and Kenneth W.Church[†††]

We introduce a new similarity measure based on dynamic programming, intended for technical terms such as *machine translation system*, which are quite common in technical writing. We compare our proposal with two baseline systems: baseline-dict and baseline-ngram. Both baselines use standard tf/IDF cosine similarity, but on different vocabularies. Baseline-dict uses an off-the-shelf dictionary-based program to segment the Japanese text into words, whereas baseline-ngram uses all of the substrings of the query. For our test collection of technical Japanese abstracts, we find that the proposed dynamic programming solution works better than baseline-dict which works better than baseline-ngram. The dynamic programming method is relatively strong when the query contains a single long technical term, and none of the words in the term are particularly good keywords.

1. はじめに

情報検索において最も効果的な単位は何であろうか.現在の研究では,英文については単語であるという傾向にある.Damashek[12]のような例外もあるが,英文では単語の部分文字列のようなより小さな単位はほとんど用いられない.しかし,文字に基づくシステムはアジアの言語においてよく用いられる.[15)~18)].一方,句や単語の列のようなより大きな単位も注目されている.Faganの初期の研究成果[3]は見込みのあるものだった.しかしながら,Stzalkowski[13]などの多くの研究者たちによる過去十年にも渡る努力も空しく,句に関する研究は未だ効果を実証するに至っていない.また,Mitra *et al.*[22]は特に高いランクで,句はあまり役に立たないということを示唆している.

もう一方,Mitra *et al.*[22]は,句が中間のランクでは最も役立ちそうだということを発見した.中間のランクとは明らかに正解(高いランク)でも,明らかに不正解(低いランク)でもないドキュメントのことである.本稿でも,実験結果から,特に質問とドキュメントの両方に技術用語,"Computer Sciences","Information Retrieval","machine translation"のような句が多く使われるとき,句が中間のランクで最も役立つことを発見した.

「技術用語」という用語を定義することは難しい.その言語を使う人は誰でも知っているというような「一般

† 豊橋技術科学大学 情報工学系
Department of Information and Computer Sciences, Toyohashi University of Technology
†† 筑波大学 電子・情報工学系
Institution of Computer Sciences and Electronics, University of Tsukuba
††† AT & T Labs – Research

語彙」とは違い，「技術用語」は比較的小さな分野の専門家に使われている．一般語彙と技術用語の区別は翻訳者にとって特に重要である．なぜなら，翻訳されるドキュメントの著者の分野と翻訳したドキュメントの読者の分野のどちらともよく知っている翻訳者はほとんどいないからである．一般的に，翻訳されるドキュメントにある用語に適した正しい訳は文脈を考慮すると一つしかないので，作文的に用語を翻訳することは危険である．例えば"Computer Science" を日本語に翻訳するとき，"Computer Science" を文字通り「計算機科学」とは訳されない．

Sager の技術用語学の教科書の導入部分[6] と，理工学分野の技術用語の翻訳に関する Hann の議論[7] をみると，用語はあるジャンル，技術作文では非常に問題となるが，他のジャンル，例えば，新聞雑誌や小説ではあまり問題にならないとされている．

我々は一般語彙と技術用語の区別もまた情報検索において重要であると考えている．技術用語を利用できるとき，その用語は特に関連性についての強い手がかりとなるだろう．また，ほとんどの技術用語は一つの単語より長く，句の形をしており，複数の単語からなる用語，マルチワード用語が技術文書ではありふれたものである．本稿の実験は日本語の大きな技術的なアブストラクトのコーパス[20),21)] に基づいている．以前の句に関する研究もさまざまな異なったジャンルを扱ってきたが，技術的なジャンルを扱った研究は少ない．

多くの検索方法，特に単語に基づく方法はこれらのマルチワード用語の利点を十分に得ることができない．句に基づく方法は技術用語の利点を多く得ることができるだろうと期待されるが，自然言語処理をこれらの多くのシステムは処理をしすぎて，性能を低下させている場合もある．文献[13] の第 3 節では，例えば，英語において，「情報検索 (information retrieval)」，「情報の検索 (retrieval of information)」，「より多くの情報を検索する (retrieve more information)」のようなさまざまな表面的な形式を，「検索 (retrieve)」は見出し (head) または操作であり，「情報 (information)」は引数である「検索＋情報」のような一つの固定した表現に正則化する複雑な自然言語処理の技術をいくつか議論している．最先端の自然言語処理を用いても，大きな制限されないテキストのコーパスでこの種の正則化を確実に実行することは非常に難しいだけでなく，そうすることが性能低下の原因となる場合もある．

一般語彙と違って，技術用語は非常に固定されている．それは「用語」と用語が表す「概念」の一対一の関係をできるだけ保つスタイルは技術作文でのよいスタイルと考えられているからである．同じことに二つの用語を使うこと (類義語) または，二つのことに一つの用語を使うこと (多義語) があることは混乱のもとである．このことについて，Sager[6] は *British Standards Guide* を参照しながら，技術用語のような専門用語を次のように定義させるべきであると提案している．

> 専門用語：正確に定義される明瞭な用語が本質的である．一つの基準を持つ専門用語は，同じオブジェクトまたは概念が，常に同じ用語によって記述または表現され，類義語を使って記述または表現されないように，無矛盾であるべきである．...

この提案は翻訳者たちに専門家たちのように正しく，無矛盾に，一つの専門用語に対して全員が同じ訳を適用することを期待している．この正当性と無矛盾性は翻訳を難しくするが，情報検索のやりやすいものである．

意味に関する変形だけでなく，文法の変形も避けるべきである．例えば，「情報工学 (Computer Sciences)」，「情報検索 (information retrieval)」という用語はときどき固定した表現と呼ばれる．この用語の変形はかなり珍しい．もし "Computer Sciences", "the science of computation", "information retrieving", "retrieval of information" のような変形を許して，概念を参照することはまったく非文法的とはいわないが，悪いスタイルになるだろう．したがって，変形の可能性は限定されているので，質問とドキュメントを表層的に用語をマッチングする最も簡単な方法でさえ，大がかりな自然言語処理技術より効果的に実行できる．

しかしながら，いくつかのとても限られた変形が可能である．Sager[6] は，英文でも長い複雑な名詞列にある単語をときどき省略できると示している．

1の例はあるマニュアルから選ばれたもので，その名詞列は指示を与える役目があるので，Sager が指摘するように，文体の変形はとても限定されている．

日本語にも同じようなパターンがある．技術用語はとても正確に使われる．英語と同じように，一つの用語が二つの物を参照すること (多義語) または二つの用語が同じ物を参照すること (同義語) があれば，混乱してしまう．また，文体の変形がとても限定されたものしかないので，同じ用語の二つの変形例は文字でマッチングがとれるようである．

しかしながら，本研究では，上で述べた変形例をテストコレクション[20),21)] に発見することができた．テストコレクションは 30 個の質問の正解判定のほかに，約 33 万個の技術的なアブストラクト (200MB) を含んでいる．30 個の質問のほとんどが技術用語を含む．これら

dynamo trap clamp bolt → dynamo clamp bolt → clamp bolt
gearbox end cover gasket → end cover gasket → gasket
exhaust valve lifter cable → exhaust lifter cable → cable

図1 マニュアルからの変形例
Fig. 1 Examples of Variations from a Manual

```
統計的 手法(statistical method)
        統計 的な 手法
        (STATISTICS TEKINA method)
自律 移動 ロボット(autonomous mobile robot)
        自律 的に 移動 する ロボット
        (autonomous TEKINI mobile SURU robots)
        自律 的な 移動 ロボット
        (autonomous TEKINA mobile robot)
画像 理解(image understanding)
        画像 の 理解
        (image NO understanding)
情報 検索 システム(information retrieval system)
        地理 情報 の 検索 システム
        (GEOGRAPHICAL information NO retrieval system)
機械 翻訳 システム(machine translation system)
        機械 翻訳 試験 システム
        (machine translation EXPERIMENTAL system)
```

図2 変形例
Fig. 2 Examples of Variations

の技術用語のうちの五つを変形例とともに, 図2に示す. 図2に示すように, 変形はかなり限られ, 「的な」, 「する」, 「の」のような平仮名文字の追加がよく見られる. これらの文字は英文にある機能単語や形態素の接辞に似ている.

最後の二つの例はSagerの英文の例によく似ている. 例えば, 「機械翻訳システム」と「機械翻訳の実験システム」は両方ともたいてい同じ物を参照するために使われる. これは長い句にある単語を追加または削除することで同じ物を表現する場合があることを示している. 特に句がかなり長くなるとき, 変形に単語の追加または削除が見られる. 例えば, 「地理情報の検索するシステム」は漢字文字「地理」だけでなく, 平仮名文字「の」も追加されている.

本稿では, ある用語にいくつかの文字を挿入または削除することによってできるこの種の変形に強いダイナミックプログラミングを用いる情報検索方法 (DP 法) を提案する. 用語に含まれる文字のほとんどが損なわれないこの種の変形では, めったに文字の順序が入れ替わることはない. このことから, 我々はダイナミックプログラミングを用いて変形をモデル化する方法のほうが, 完全に順序情報を無視し, 文字列を単語または文字の袋として考える方法より効果的に情報を検索するのではないかと考えた.

2. ベースラインシステム

我々の提案する方法 (DP 法) を詳しく紹介する前に, 三つのベースラインシステムを紹介する. 本稿では, ベースラインシステムとして, 単語に基づくシステムとngramに基づくシステムを考慮している. まず, 単語に基づくシステムをBD法 (baseline-dict) とする. このシステムを実行するために, 既存の日本語形態素解析プログラム「茶筌」[23] を使って質問とドキュメントを単語に区切り, Salton[4] によって提案される $tf \cdot IDF$ の重みを用いる内積スコアリング関数を使う. 「茶筌」は大きな日本語の単語辞典を使って文字の順列を単語に区切る. また, 「茶筌」は品詞を割り当てる. 本研究では, 名詞, 動詞, 未定義語を用語として使い, 他の品詞の単語はストップワードと考えた. 次にBD法で用いた類似度の関数を示す.

定義 2.1 t は名詞, 動詞, 未定義語として質問とドキュメントの両方に現われる単語, $tf(t)$ はそのドキュメントの単語 t の出現頻度 (term frequency), $df(t)$ は単語 t が出現するドキュメントの数 (document frequency), N はドキュメントの総数であるとする.

$$SIM_{dict} = \sum_{t} tf(t) \cdot (-log_2(df(t)/N))$$

文字に基づくシステムをBN法 (baseline-ngram) とする. ほとんどの文字に基づくシステムは質問とド

キュメントを短く，できるだけ重ねながら，バイグラム (bigram) またはトライグラム (trigram) のようなngram に区切る．日本語では，漢字，片仮名，平仮名の三種類の文字が使われる．漢字は昔中国から来た文字で数千文字，片仮名は外来語を表現するために使われる文字で 50 文字，そして平仮名は日本語の機能単語を表現するために使われる文字で 50 文字ある．片仮名は漢字よりかなり少ないので，片仮名単語は漢字単語より長い傾向がある．例えば，図 2では，漢字単語「機械」や「翻訳」はどちらとも二文字であるのに対し，片仮名単語「システム」は四文字である．Fujii and Croft[8] が示したように，短い ngram は漢字にはかなり効果的であるが，片仮名には効果的でないと推定できる．このことから，短い ngram は中国語にはもっと効果的であると推定できる．なぜなら，中国語では使われる文字がすべて漢字だからである．実際に，中国語のための短い ngram を用いたシステムが議論され[16),17)]，韓国語のためのシステムも議論されている[15)]．このように、アジアの言語において，短い ngram である bigram で情報検索を行なう手法がよく知られている[18)]．

上のような結果があるからといって，単に短い部分文字列だけに注目することはない．suffix array と PAT-tree[10),11),24),26)] などの複雑なデータ構造を使えば，非常に長い ngram を用いることができる．しかし実際に，ngram と bigram のどちらが効果的であるかは明らかでない．そこで本研究では，もう一つのベースラインシステムとして，bigram を用いたシステム，BB 法 (baseline-bigram) を考慮した．このシステムは基本的に BN 法と同じであるが，対象とする文字列の長さが 2 以下に限られているものである．実際に，表 2の最後の行に示されるように，この実験では長い ngram は効果があることを示すデータが得られた．

本研究では，BN 法は質問にあるすべての部分文字列を考えるものとした．それぞれの部分文字列について，標準的な内積尺度を使って，$tf \cdot IDF$ の重みとスコアを計算する．

次に BN 法で用いた類似度の関数を定義する．スコア関数は一致した部分文字列の重みと考えられる．本研究では，典型的な IDF に基づく関数を使う．$df(\xi)$ は部分文字列 ξ の出現ドキュメントの数である．通常，df は単語を対象とするが，ここでは部分文字列を対象とする．

定義 2.2 α, β, ξ, η を文字列とする．α_{ik} を i 番目の文字から $i+k-1$ 番目の文字までの α の部分文字列とし，β_{jk} を j 番目の文字から $j+k-1$ 番目の文字までの β の部分文字列とする．また，$Score$ を後述する文字列から正の実数値を求める関数とする．

$$SIM_{ngram} = \sum_{i,j,k} Comp(\alpha_{ik}, \beta_{jk})$$

但し，$Comp(\xi, \eta)$ は次のように定義される．

- $\xi = \eta$ ならば，$Score(\xi)$
- $\xi \neq \eta$ ならば，0.0

同様に，BB 法で用いた類似度の関数を定義する．

定義 2.3 α, β, ξ, η を文字列とする．α_{i1} を α の i 番目の文字とし，β_{j1} を β の j 番目の文字とする．α_{i2} を i 番目の文字から次の文字までの α の部分文字列とし，β_{j1} を j 番目の文字から次の文字までの β の部分文字列とする．また，$Score$ を後述する文字列から正の実数値を求める関数とする．

$$SIM_{bigram} = \sum_{i,j} Comp(\alpha_{i1}, \beta_{j1}) + Comp(\alpha_{i2}, \beta_{j2})$$

次にスコア関数を示す．スコア関数は，より長い部分文字列を短い部分文字列より優先させるために，IDF(Inverse Document Frequency) に部分文字列の長さをかけたものとした．

- 空文字ならば，
  $$Score("") = 0.0$$
- それ以外ならば，
  $$Score(\xi) = -L \cdot \log_2(df(\xi)/N)$$

本研究では，この三つのベースラインシステムをチューニングしておらず，その結果，効率的または効果的ではないが，DP 法との整合性があるように設計した．おそらく我々の提案する方法 (DP 法) も同様に原理に忠実な定義である．特に，一つの長い技術用語によって主に構成される質問に対しては性能が高いことが予想される．図 3にテストコレクションにある 24 番の質問を示す．この質問は「機械翻訳システム」を含んでいて，我々の検索システムではこの用語を利用して，効果的に検索することができるが，ベースラインシステムではどちらとも効果的に検索することができない．

BD 法は「機械翻訳システム」を「機械」，「翻訳」，「システム」の三つの日本語単語に区切る．しかし，三つの単語がどれもあまりよいキーワードではないので，BD 法では効果的に検索することができない．実際に，「システム」は特に役に立たない．この単語は 33 万個のドキュメントの約半分に現われるので，その IDF 重みの寄与は無視できる．しかし，質問の最後の文に解説されるように，「機械翻訳」については書かれているが，「システム」については書かれていないドキュメントは不適切だとしているので，「システム」はその質問には必要不可欠な単語である．

BN 法は「機械翻訳システム」という用語を含む問題について DP 法の次に効果的である．なぜなら，この用

<検索要求>機械翻訳システムについて</検索要求><検索要求説明>開発した、あるいは、開発中の機械翻訳システムについて、その構成、特徴、機能、性能などについて述べている文献がほしい。機械翻訳技術を他の何かに応用したもの、あるいは、機械翻訳の要素技術や辞書の作成などについての記述だけでは不充分である。</検索要求説明>

<TITLE>about machine translation system</TITLE>
<EXPLANATION>Documents that explain the features, the functions and the performance of a machine translation system either developed or under development. It is not enough if the document only describes the application of machine translation systems, or related technology such as compilation of dictionary.</EXPLANATION>

図3 質問24とその英訳
Fig. 3 Query #24 and Its Translation

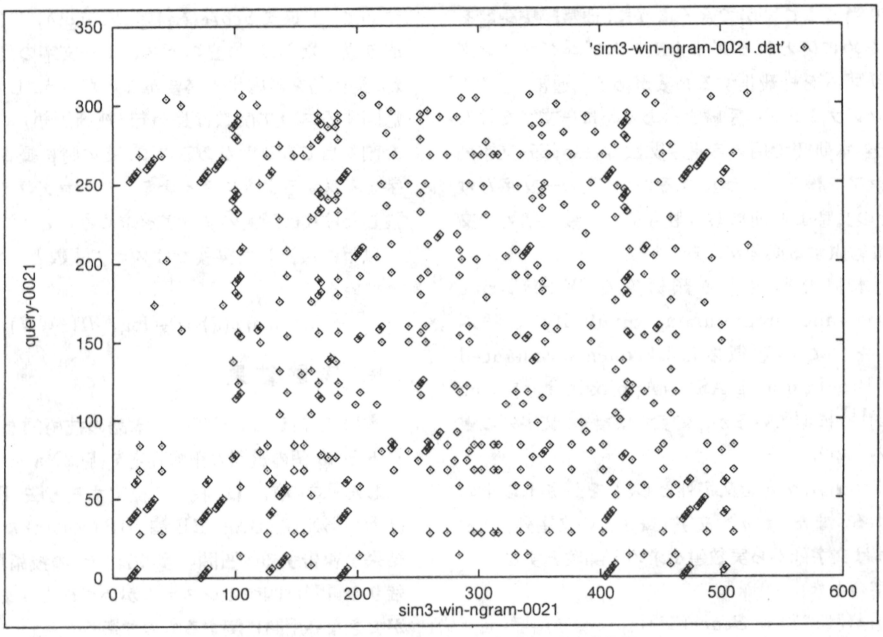

図4 質問とドキュメント間の一致状況を表す形状図
Fig. 4 Dotplot between Query and A Document

語の全体を含んでいるドキュメントには非常に高いスコアを与え，この用語の部分しか含んでいないドキュメントにはより低いスコアを与えるからである．BN法はとBB法はBD法よりこの質問について効果的であるが，図2に示した変形を捕らえることができない．実際に，テスト集合にある多くの質問について，BN法とBB法は効果的でなく，BD法よりもかなり悪い結果であった．

図4は「機械翻訳システム」という句を含んでいる質問と「機械翻訳実験システム」という句を含んでいるドキュメントの間の一致状況を表すdotplots[9]と呼ばれている形状図である．この図は質問のi番目の文字がドキュメントのj番目の文字と一致するとき，i-jの位置に点が描かれる．dotplotsは，生物学で長いDNAの列をマッチングするために使われている．この図に現われる断続的な斜線によって「機械翻訳システム」と「機械翻訳実験システム」の対応をはっきりと見ることができる．

3. 編集距離と提案したシステム

編集距離は「機械翻訳システム」と「機械翻訳実験システム」間の数回の挿入と削除操作による変形について考える自然な方法である．編集距離については文献[19]

の付録に記述されている．編集距離は一方の文字列をもう一方の文字列に写像するために必要な挿入と削除操作などの最小編集操作数である．編集距離はスペル修正で広く使われている．本研究では，この編集距離または，これに密接に関係している類似尺度の情報検索への利用を試みた．図4に示したように，我々が提案する方法はできるだけ多くの点を通る左下の角から右上の角までのパスを見つける．このアプローチは完全に文字列を単語の袋として見る考えを捨て，順序情報をもとらえる．

しかしながら，ある程度の性能を得るためには，まだ多少の改良が必要である．例えば，すべての点を平等に考慮しないように，部分文字列のIDF重みを導入する必要がある．これは関数 *score* で実行される．その他に，重み関数が長い部分文字列（技術用語）に非常に高いスコアを割り当てることができるように，通常編集距離を計算するために使われるダイナミックプログラミングのアルゴリズムを一般化する必要がある．通常，ダイナミックプログラミングの手続きはある時に一文字だけを考慮するが，本研究で用いる重み関数は長い部分文字列は高いスコアを得ることができるので，次の一文字だけでなく，その文字より前にある最も長い一致文字列の文字をすべて考慮する必要がある．

我々は，本稿で提案する類似度をSWS (String Weight dynamic programming Similarity) と呼ぶことにした．この類似度はUkkonen's Enhanced Dynamic Programming ASM (Approximate String Matching)[14] に似ているが，文字ではなく，文字列の重みに基づいている．

定義 3.1 $\alpha, \beta, \gamma, \delta$ を文字列とし，ξ を長さ1以上の文字列とする．また，x, y を文字（長さ1の文字列）とする．$Score$ は文字列から実数値を求める関数とする．

- 空文字のとき，
  $SIM_{DP}(\text{""},\text{""}) = Score(\text{""})$
- それ以外
  $SIM_{DP}(\alpha,\beta) =$
  $\quad MAX(SIM_{DPs}(\alpha,\beta), SIM_{DPg}(\alpha,\beta))$
  但し，SIM_{DPs} は次のように定義する．
  - 前に最大一致文字列 ξ が存在するとき，
    $SIM_{DPs}(\xi\alpha,\xi\beta) =$
    $\quad MAX(Score(\gamma) + SIM_{DP}(\delta\alpha,\delta\beta))$
    $\quad$ for all γ; for all δ; such that $\xi = \gamma\delta$
  - そのような文字列が存在しないとき，
    $SIM_{DPs}(\alpha,\beta) = 0.0$

  SIM_{DPg} は次のように定義する．
  - $SIM_{DPg}(x\alpha, y\beta) =$
    $\quad MAX(SIM_{DP}(\alpha, y\beta),$
    $\quad SIM_{DP}(x\alpha, \beta),$
    $\quad SIM_{DP}(\alpha,\beta))$

次の例では，スコア関数が単にその引数の長さを返す場合の SIM_{DP} を示す．このケースでは，SIM_{DP} はできるだけ多くの「点」を通るようなパスを発見し，この最良のパスにある点の数を返す．

例 3.2 スコア関数が単に引数の長さを返す場合，
$SIM_{DP}(ABCD, ABXCD) = 4$
$SIM_{DP}(ABCD, ABXDC) = 3$
$SIM_{DP}(ABCD, DCXBA) = 1$

編集距離を計算する方法である文字ごとの重みを計算するダイナミックプログラミング法は長いマッチが指数的に高いスコアを得ることができないと仮定している．言い換えれば，もしスコア関数がすべての部分文字列について，不等式 $Score(\delta\gamma) \leq Score(\delta) + Score(\gamma)$ が成り立つならば，最良のパスは単一文字の列から構成され，長い句を考慮する必要がないだろう．しかし，我々が意図するスコア関数は長い句（技術用語）に高いスコアを割り当てることがあるので，その時に長さ1以上の文字を考慮するようにダイナミックプログラミング法を拡張しなければ最大のスコアを求めることができない．

以下に報告した実験ではスコア関数としてIDFを用いている．

$$Score(\xi) = -\log_2(df(\xi)/N)$$

4. 実験結果

本研究では，33万個の日本語の技術的なアブストラクトと30個の質問の正解判定を持つNacsis[20),21)] を利用した．この節では，我々の提案する方法（DP），BD法（BD），BN法（BN），BB法（BB）の四つのシステムの結果を報告する．質問の多くは一つの技術用語を含み，残りの質問は四つのシステムがいづれもうまく扱うことができない用語に関するものである．

表2は，DP法はBD法より効果的で，BD法はBN法とBB法より効果的であることを示している．また，BN法はBB法より効果的であることも示している．この表はそれぞれのシステムを二つずつ各質問について，表1に示す11点平均精度（11 point average）を使って比較し，すべての質問について数値で判定した結果である．

表3は，30個の質問のうち四つの質問について，BD法，BN法，DP法を詳しく分析するものの説明である．ここでは，BB法はBN法とほとんど同じ振舞いをするので，省略した．この表では，“○” は性能が高いことを表し，“×” は性能が低いことを表している．また，かっこの中の数字はその用語の変形を数えず，用語がそのまま

表1　それぞれのシステムの質問ごとの11点平均精度の適合率
Table 1　11 point average precision, broken down by method and query.

質問番号	BB	BD	BN	DP	質問番号	BB	BD	BN	DP
1	0.1359	0.2462	0.2216	0.3006	16	0.0093	0.0180	0.3275	0.8263
2	0.3130	0.2099	0.5022	0.6704	17	0.0000	0.0471	0.0011	0.0407
3	0.0045	0.0257	0.0028	0.0148	18	0.0142	0.0788	0.0051	0.0980
4	0.1227	0.1048	0.1815	0.3136	19	0.3028	0.2091	0.4901	0.6890
5	0.0046	0.1046	0.0008	0.0013	20	0.4642	0.5660	0.5705	0.7775
6	0.0277	0.0580	0.0221	0.1890	21	0.0404	0.0881	0.0713	0.1583
7	0.0008	0.0005	0.0010	0.0024	22	0.0459	0.0516	0.0703	0.2493
8	0.0000	0.0836	0.0086	0.2119	23	0.1222	0.2003	0.1475	0.3481
9	0.0260	0.0157	0.0372	0.2402	24	0.2320	0.1915	0.2887	0.3234
10	0.2417	0.3751	0.2991	0.3618	25	0.0324	0.6076	0.2291	0.4877
11	0.0553	0.2533	0.0303	0.2638	26	0.0732	0.1087	0.4665	0.5136
12	0.0498	0.1008	0.1687	0.1566	27	0.0348	0.0659	0.0696	0.2556
13	0.0103	0.0707	0.0150	0.0655	28	0.0395	0.0072	0.0435	0.0589
14	0.2081	0.4107	0.3051	0.4257	29	0.0536	0.1543	0.0882	0.1151
15	0.1401	0.1538	0.1436	0.2116	30	0.0033	0.0094	0.0142	0.0527

表2　DP法はBD法(BN法とBB法より効果的)より効果的である.
Table 2　DP is better than BD, which is better than BN and BB

X vs Y	X win	Y win
DP vs BD	23	7
DP vs BN	29	1
DP vs BB	29	1
BD vs BN	15	15
BD vs BB	23	7
BN vs BB	25	5

表3　質問に技術用語が含まれ,その用語を構成する単語が低いIDF重みを持つとき,DP法は効果的に働く.
Table 3　DP works well when the query mentions a technical term, and each of the words in the term have little IDF weight.

質問番号	BD	BN	DP	用語	注
1	○	○	○	自律移動ロボット (124)	典型的な例
12	×	○	○	データマイニング (74)	形態素解析に難ある例
24	×	×	○	機械翻訳システム (244)	語順が問題となる例
29	○	×	×	位置計測 (176)	対応するものが用語でない例

現われるドキュメントの数を表している.質問1は,用語と用語を構成する単語の多くがそれらのIDF重みによって示されるようなよいキーワードである用語を含む質問であり,すべてのシステムにとって簡単な質問であることを示している.質問12では,変形のない用語を含む質問はBN法でもDP法でも効果的に検索することができることを示している.質問24では,DP法の強さが観測できる.これは,用語がその用語を構成する単語よりもかなり強い手がかりとなる質問である.質問29では,BD法の強さが観測できる.この場合,位置計測という用語,およびその変形はドキュメントにはなく,この質問だけに使われる複合名詞であった.

これらの四つの質問の再現率を図5-8に示す.図6では,性能に差があるとき,その差はグレイ領域でもっとも大きいことがわかる.また,Mitra *et al.*[22]も言うように,トップにある少数のドキュメントについては句によって違いが現われないことがわかった.

スコアの合算法で,最大の一致するパスを求めることを説明した.検索に効果のある文字列は,ある意味での単語または句である.したがって,DP法は検索のための合算と同時に,検索に特化した単語または句の抽出も行なっていると考えることができる.これは,単語の区切りに関する情報を人間の直感によって与えられたBD法よりも,効果的に区切っていると解釈できる.

BN法,BB法,BD法ともに,パラメータを付け加えてチューニングして,性能を向上させることができるが,同様のチューニングはDP法にも施すことができる.情報の合算方式の比較という観点からでは,すべて,頻度

と情報量をパラメータなしで合算しており，意味のある比較となっていると判断できる．

5. 実行速度

すべての文字列の情報量を求めることは，コストのかかる処理である．しかしながら，実行できない処理ではない．山本ら[26]の方法によって，通常のコンピュータで実現できる処理である．具体的にはsuffix arrayを用いて，すべての部分文字列を同じ出現パターンを持つグループに分類し，そのグループに対して出現の情報量を求めるという処理を行なった．部分文字列の総数は$n(n-1)/2$であるが，グループの総数は多くても$2n$であることが証明できるので，あらかじめそのグループに情報量の表を作っておくことができる．

実用的な情報検索システムを作成するには，さらに工夫が必要であるが，実行できない速度ではない．具体的には，PentiumII 400Mhz，メモリ512Mのコンピュータの上で，30個の質問に関して10時間で情報検索を行なっている．これは，総当たりで比較した結果である．

6. おわりに

我々は文字列重みを使用するダイナミックプログラミング法を提案した．そして，その方法が，特に質問が「機械翻訳システム」というような一つの技術用語から構成され，用語の中の単語が低いIDF重みを持つ単語であるとき，効果的であることを示した．今後の課題として，我々は提案した方法がどのような質問に対して効果的に働くかを明示することができるので，新しいダイナミックプログラミング法とより伝統的な方法を組み合わせ，両者の利点を得ることができる混合システムを作ることを考えている．

謝辞 本研究は住友電工との共同研究の成果を利用させて頂きました．深く感謝いたします．

参考文献

1) G.Salton and M. E. Lesk, Computer Evaluation of Indexing and Text Processing, *journal of the ACM*, Vol.15, No.1, pp.8–36, January 1968.

2) G. Salton and M.J. McGill, The SMART and SIRE Experimental Retrieval Systems, pp.118–155, New York: McGraw-Hill, 1983.

3) J. L. Fagan, Experiments in Automatic Phrase Indexing For Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods,it PhD thesis, Department of Computer Science, Cornell University, Ithaca, NY, 1987. Available as CUCS Technical Report TH87-868.

4) Gerard Salton and Christopher Buckley, Term-Weighting Approaches in Automatic Text Retrieval, *Information Proceeding and Management*, 24, pp.513–523, 1988.

5) V. Raghavan, G.S. Jung, and P. Bollmann, A Critical Investigation of Recall and Precision as Measures of System Performance, *ACM Transactions on Information Systems*, Vol.7, No.3, pp.205–229, July 1989.

6) Juan C. Sager, A Practical Course in Terminology Processing, John Benjamins Publishing Company, Amsterdam/Philadelphia, 1990.

7) Michael Hann, The Key to Technical Translation, John Benjamins Publishing Company, Amsterdam/Philadelphia, 1992.

8) Hideo Fujii and W. Bruce Croft, A Comparison of Indexing Techniques for Japanese Text Retrieval, *In proceeding of SIGIR'93*, Pittsburgh PA, USA, pp.237–246, 1993.

9) Kenneth Church and Jon Helfman, Dotplot: a Program for Exploring Self-Similarity in Millions of Lines of Text and Code, *The Journal of Computational and Graphical Statistics*, 2:2, pp. 153-174, 1993.

10) Udi Manber and E. Myers, Suffix array: A new method for on-line string searches, *SIAM Journal on Computing*, 22:5, pp. 935 - 948, 1993. http://glimpse.cs.arizona.edu/udi.html.

11) M. Nagao and S. Mori, A new method of n-gram statistics for large number of n and automatic extraction of words and phrases from large text data of Japanese, *Coling-94*, pp.611-615, 1994.

12) Damashek, Marc. Gauging Similarity with n-grams: Language Independent Categorization of Text," *Science*, Vol 267, 10 Feb 1995, pp 843-848.

13) T. Strzalkowski, L. Guthrie, J. Karlgren, J. Leistensnider, F. Lin, J. Perez-Carballo, T. Straszheim, J. Wang, J. Wilding., Natural Language Information Retrieval: Trec-5 Report, *in E. M. Voorhees and D. K. Harman (eds.), The Fifth Text REtrival Conference (TREC-5)*, pp. 291-314,1996. http://trec.nist.gov/

14) H. Berghel and D. Roach, An Extension of Ukkonen's Enhanced Dynamic Programming ASM Algorithm, *ACM Transactions on Information Systems*, Vol.14, No.1, pp.94–106, January 1996.

15) Joon Ho Lee and Jeong Soo Ahn, Using n–Grams for Korean Text Retrieval, *In proceeding of SIGIR'96*, Zurich, Switzerland, pp.216–224, 1996.

16) K.L.Kwok, Comparing Representations in Chinese Information Retrieval, *In proceeding of SIGIR'97*, Philadelphia PA, USA, pp.34–41, 1997.
17) Aitao Chen, Jianzhang He, Liangjie Xu, Fredric C. Gey, and Jason Meggs, Chinese Text retrieval Without Using a Dictionary, *In proceeding of SIGIR'97*, Philadelphia PA, USA, pp.42–49, 1997.
18) Yasushi Ogawa and Toru Matsuda, Overlapping statistical word indexing: A new indexing method for Japanese text, *In proceeding of SIGIR'97*, Philadelphia PA, USA, pp.226–234, 1997.
19) Robert R. Korfhage, Information Storage and Retrievel, WILEY COMPUTER PUBLISHING, Johon Wiley & Sons, Inc., Printed in USA, 1997.
20) Kando, N. et al., NTCIR:NACSIS Test Collection Project, *20th Annual Colloquium of BCSIRSG*, Autrans, France, March 25-27, 1997.
21) Kageura, K. et al., NACSIS Corpus Project for IR and Terminological Research, *Natural Language Proceeding Pacific Rim Symposium'97*, Phuket, Thailand, pp.493-496, December 2-5, 1997.
22) M. Mitra, C. Buckley, A. Singhal, C. Cardie, An Analysis of Statistical and Syntactic Phrases, *RIAO-97*, pp. 200-214.
23) Yuji Matsumoto, Akira Kitauchi, Tatsuo Yamashita, Yoshitaka Hirano, Osamu Imaichi, and Tomoaki Imamura, Japanese Morphological analysis System ChaSen Manual, *NAIST Technical Report, NAIST-IS-TR97007*, February 1997, http: //cactus.aist-nara.ac.jp/ lab/ nlt/ chasen.html.
24) Lee-Feng Chien, PAT-Tree-Based Keyword Extraction for Chinese Information Retrieval, in proceeding of SIGIR'97 Philadelphia PA, USA, pp.50–58, 1997.
25) Justin Zobel and Alistair Moffat, Exploring the Similarity Space, *In proceeding of SIGIR FORUM*, Vol.32, No.1, pp.18–34, 1998.
26) Mikio Yamamoto and Kenneth W. Church, Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus, *In proceeding of 6th Workshop on Very Large Corpora*, Ed. Eugene Charniak, Montreal, pp.28–37, 1998.

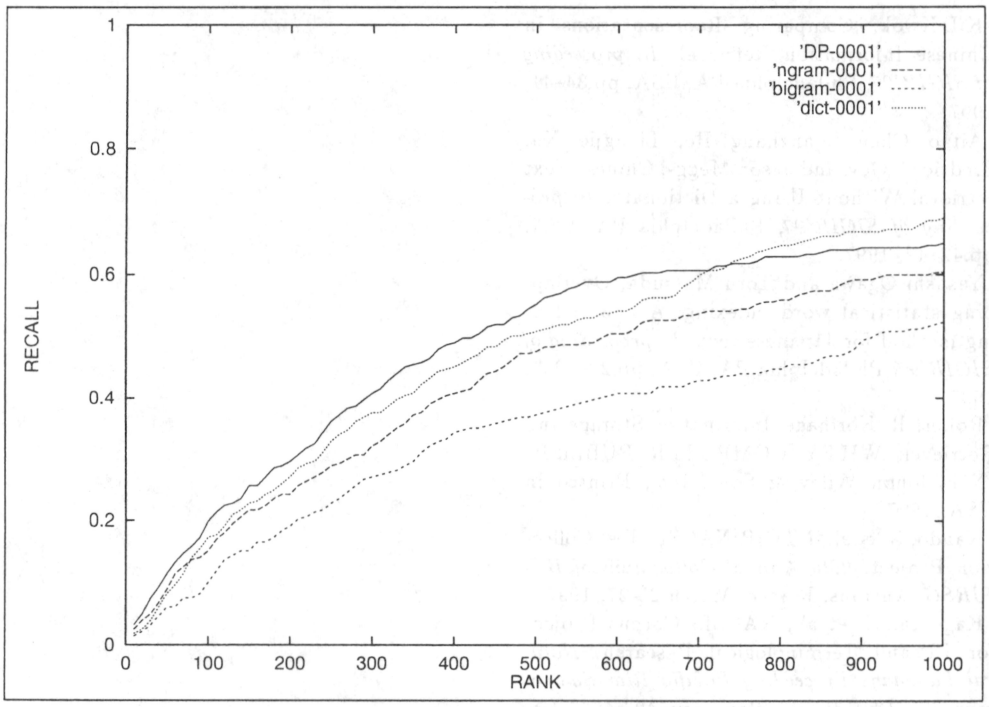

図5　質問1,「自律移動ロボット」: 簡単な質問

Fig. 5　Query #1, *autonomous mobile robot*: easy for all three systems.

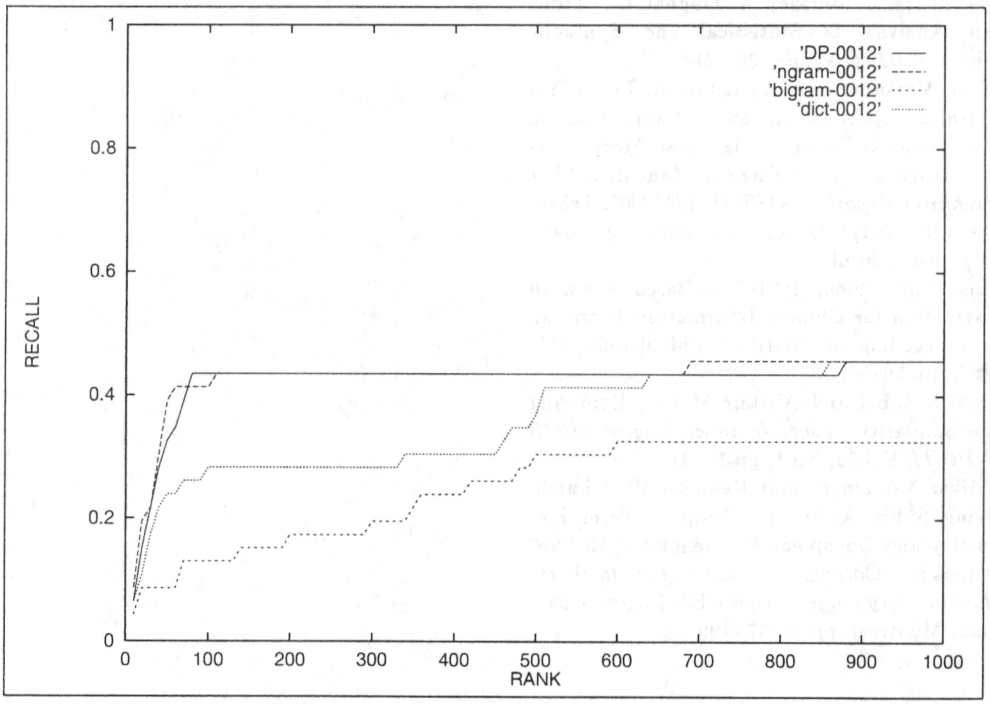

図6　質問12,「データマイニング」: 用語を正しく単語に分割できないため, BD 法には難しい質問.

Fig. 6　Query #12, *data mining*: hard for baseline-dict, because the word segmentator split *data* inappropriately.

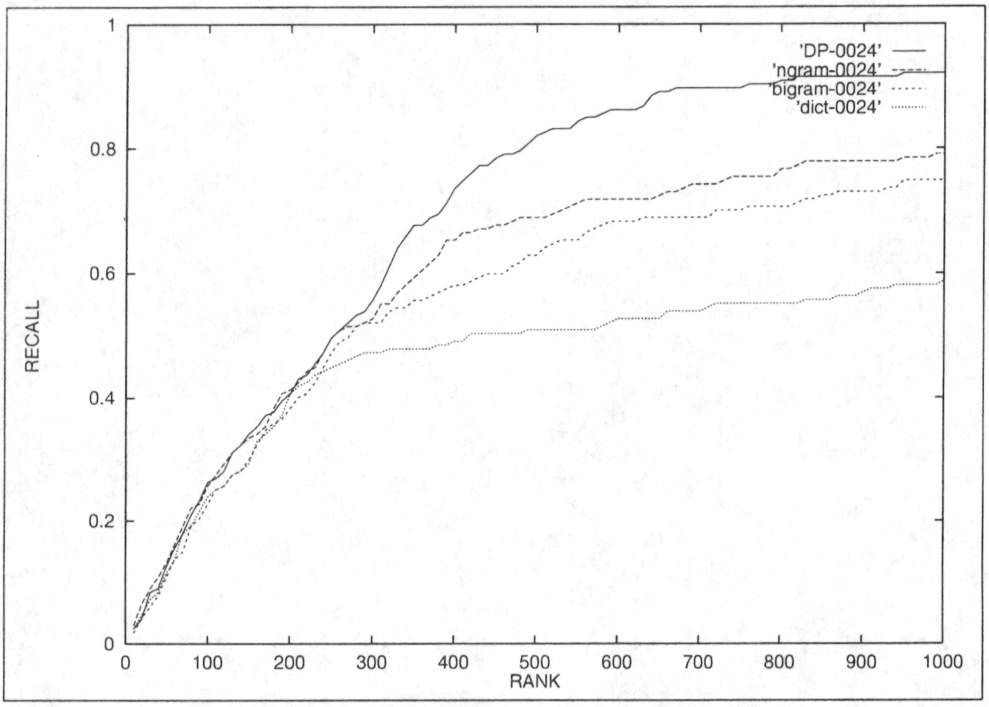

図7 質問 24,「機械翻訳システム」: DP 法が得意とする質問

Fig. 7 Query #24, *machine translation system*: ideal for dynamic programming (DP)

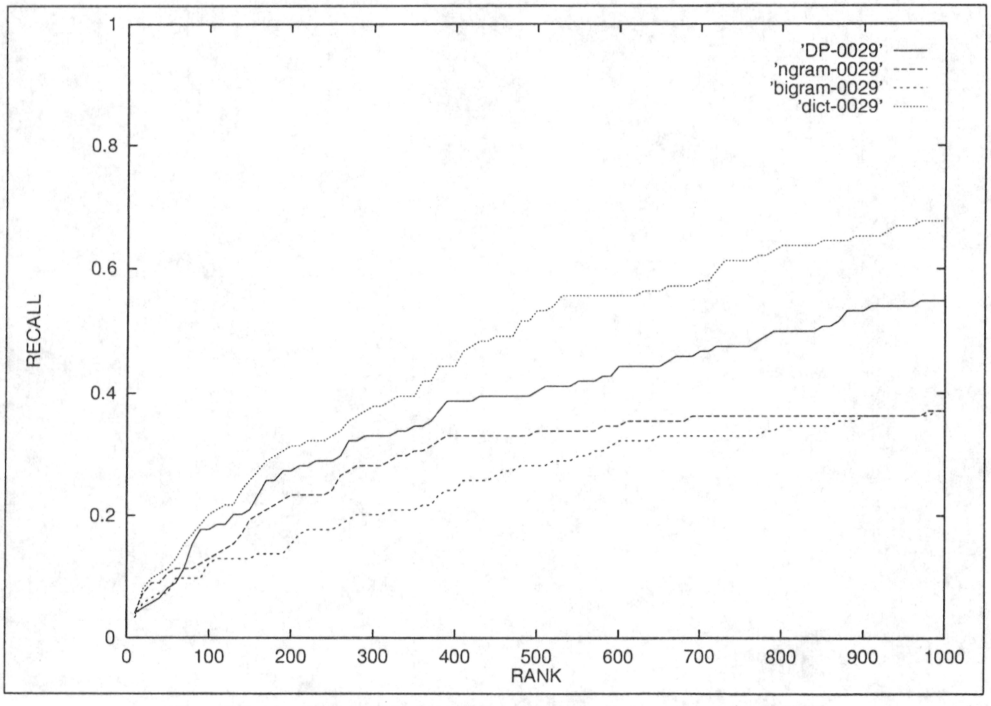

図8 質問 29,「位置 + 計測」: 技術用語が存在しないため,DP 法には難しい質問.

Fig. 8 Query #29, *position + measurement*: hard for dynamic programming (DP) because there is no technical term.