情報処理学会論文誌　データベース　Vol. 1　No. 1　36–48 (June 2008)

ブートストラップ式同位語辞書構築における検索効率の向上

河合英紀[†1]　水口弘紀[†2]　土田正明[†2]
國枝和雄[†1]　山田敬嗣[†1]

本稿では，同一の意味的階層に属する単語集合（同位語辞書）を高い検索効率で Web から抽出する方法を提案する．近年，Web 上に散在した知識を収集するアプローチの 1 つとしてブートストラップ式同位語辞書構築手法が注目を集めている．サーチエンジンが提供する検索 API は知識収集の強力なツールである．しかし，検索 API を通じて大量の検索を行うことは，サーチエンジンに過剰な負荷をかけてしまうことになる．そのため，サーチエンジン側でも検索 API を通じた検索回数を制限している．そこで本稿では，検索回数をコストとして考慮し，なるべく少ない検索回数でより多くの同位語を収集可能な検索戦略を求めることを目標とする．実験の結果，合計 2,000 回の検索で 10 万語以上のキーワードを適合率 0.9 以上で抽出することができた．

Cost-effective Search Strategy for Bootstrapping Lexicon Acquisition

Hideki Kawai,[†1] Hironori Mizuguchi,[†2]
Masaaki Tsuchida,[†2] Kazuo Kunieda[†1]
and Keiji Yamada[†1]

In this paper, we propose a cost-effective search strategy framework to extract keywords in the same semantic class from the Web. Constructing a dictionary based on the bootstrapping technique is one of the promising approaches to harnessing knowledge scattered around the Web. Open web application programming interfaces (APIs) are powerful tools for the knowledge-gathering process. However, we have to consider the cost of API calls because too many queries can overload the search engines, and they also limited the number of API calls. Our goal is to optimize a search strategy that can collect as many new words as possible with the least API calls. Our results shows that the optimized search strategy can extract more than 100,000 words with a precision of 0.90 by with only 2,000 search API calls.

1. はじめに

インターネットの出現から現在まで，Web 上には莫大な量の知識が蓄積されてきた．情報抽出は，Web 上に分散したヘテロな知識を活用するための重要な技術である．情報抽出における中心的な要素技術として，文書中に出現する人名，地名，組織名などの固有表現（Named Entity）を抽出するタスクがあげられる[1]．固有表現抽出手法には，大きく，（A）人手によるルール記述[2]，（B）注釈付きデータによる機械学習[3)–5)]，（C）教師なし学習によるブートストラップ[6)–8)]，の 3 種類のアプローチが存在する．

言語的な専門知識が必要なルール記述や，注釈付きデータの作成が大変な機械学習のアプローチに比べ，ブートストラップでは，注釈なしコーパスと数語〜数十語程度の小規模な固有表現の例（シード語集合）を準備するだけでよく，人的コストが低い手法として最近注目が集まっている．

ブートストラップ式の辞書構築手法は，主に，抽出パターン構築と，新語抽出の繰返しにより構成される．抽出パターン構築では，コーパス中でシード語の周辺文字列の出現頻度を計数し，より多くの文書に共通する定型表現をグローバルパターンとして構築する．たとえば，俳優名の辞書を構築する場合，「俳優の〜さん」などの定型表現がグローバルパターンとなる．新語抽出では，グローバルパターンがマッチする新語を抽出することによって辞書を増殖させる．

特に近年では，Web 2.0[9] の隆盛にともない，サーチエンジンの検索 API を利用して，膨大な Web ページ群を注釈なしコーパスとして辞書構築に活用できるようになっている．たとえば，代表的なサーチエンジン活用型のブートストラップ式同位語辞書構築手法として，KnowItAll[7] があげられる．

しかし，従来のサーチエンジン活用手法では，非常に多くの検索回数を必要としていた．これは，従来手法がグローバルパターンを利用していることが原因である．なぜなら，グローバルパターン構築のためには，1 語 1 語のシード語について大量の検索結果文書を照合する必要があるからである．

†1 NEC C&C イノベーション研究所
NEC C&C Innovation Research Laboratories
†2 NEC 共通基盤ソフトウェア研究所
NEC Common Platform Software Research Laboratories

短時間に大量の検索を行うとサーチエンジンの負荷を高めてしまうため，KnowItAll では，クエリを複数のサーチエンジンに分散して発行するなどの対策を行っているが，根本的な問題解決にはなっていない．一方，サーチエンジンの側でも，過剰な負荷を避けるために，検索 API の呼び出し回数や検索結果の最大件数を制限している．これらの制限を遵守しながら効率良く辞書構築を行うためには，少ない検索回数でなるべく多くの新語を抽出できる方法が必須である．

そこで本稿では，より効率の高い検索戦略を備えたブートストラップ式辞書構築手法として，Cost-Effective Search Strategy（CESS）フレームワークを提案する．本フレームワークの主な特徴は，複数のシード語を組み合わせて AND 検索するマルチシードクエリと，後述する使い捨てのローカルパターンの 2 点である．

CESS フレームワークにおける辞書構築の基本コンセプトを示すために，俳優名の辞書を構築する場合を例として説明する．たとえば，シード語集合として 10 人の俳優名リストが与えられ，マルチシードクエリとしてシード語集合から 5 人の俳優名をランダムに組み合わせて AND 検索を行ったとする．この場合，検索結果には様々な Web ページが含まれるが，もし，クエリに使った 5 人の俳優名がすべて ⟨OPTION⟩ タグで囲まれていた場合，その Web ページには俳優名をメニューとするフォームが記述されており，同じフォームの ⟨OPTION⟩ タグの間にはシード語に未登録の別の俳優名も記述されているだろうと期待できる．つまり，単一ページ内における一貫性の高い周辺文字列をローカルパターンとして利用すれば，そのページから複数の新語を抽出できる．ただし，ローカルパターンは単一ページ内だけに通用するフォーマットであるために使い捨てである．

このように，従来手法ではグローバルパターンの構築に大量の検索結果文書の照合が必要なのに対し，提案手法では単一ページ内の照合だけでローカルパターンを構築可能である．また，「俳優の〜さん」のようなグローバルパターンがマッチする新語は 1 文書あたりたかだか数語程度であるのに対し，⟨OPTION⟩ タグのようなローカルパターンにマッチする新語は 1 文書あたり数十語になることも多い．さらに，マルチシードクエリでは，より多くのシード語で検索するほど一貫性の高いフォーマットで記述された文書がヒットする可能性が高まる．以上より，従来手法に比べて提案手法の方が検索効率が高くなると期待できる．

また，CESS フレームワークの別の特徴としては，文字列ベースのローカルパターンを採用しているため，日本語や英語などの言語，および HTML や XML などのマークアップ方式に依存しないという利点がある．また，異なるフォーマットで記述された複数のページから抽出された新語ほど同位語としての信頼度が高いと見なすシードスコアを定義することによって，Web のヘテロ性を辞書の適合率向上に積極的に活用している．さらに，マルチシードクエリとローカルパターンの採用により，誤抽出されたノイズ語に対する高いロバスト性を維持できるという副次的な利点もある．評価実験では，検索効率と辞書の適合率を指標として，提案手法の有効性を検証する．

2. 関 連 研 究

ブートストラップ式辞書構築手法の従来技術の例としては，単語の配列をパターンとして利用する研究[6)–8)] や，HTML のタグ構造をパターンとして利用する研究[10)] があげられる．これらは，いずれもコーパス内に共通して出現するシード語の周辺情報をグローバルパターンとして利用する点では同じである．グローバルパターンを構築するためには，シード語が出現する文書を網羅的に照合する必要があり，処理時間や検索回数を増大させる原因となっている．

たとえば，Brin [11)] は，2,700 万件の Web ページをコーパスとして固有表現の 2 項関係をブートストラップ式に求めているが，文書の照合に時間がかかりすぎるために実験を途中で断念している．また，KnowItAll [7)] は，各シード語に対して再帰的に検索を繰り返し，膨大な件数の検索結果文書を取得している．また，KnowItAll では，抽出された新語をグローバルパターンと組み合わせた検索を行うことによって，新語の信頼性を判定している[12)]．たとえば，固有表現が地名で，新語が「Chicago」だった場合，「Chicago」単独の検索結果と，「the cities such as Chicago」という表現での検索結果の比をもとに，「Chicago」の地名らしさを判定する．彼らの方法によれば，m 語の新語と n 個のグローバルパターンが存在した場合，信頼性判定に $m \times n$ 回の検索が必要になる．このように，従来手法においては，非常に貪欲な文書照合処理と検索戦略が用いられてきた．

一方，筆者らは，検索 API の呼び出しを，メモリ管理における二次記憶へのアクセスと同様にコストとして考慮し，検索効率を考慮したブートストラップ式辞書構築手法を研究している[13)–16)]．CESS フレームワークでは，複数のシード語を組み合わせたマルチシードクエリを検索に利用する点と，使い捨てのローカルパターンによって新語を抽出する点の 2 点が大きな特徴である．これまで，文献 13) において，マルチシードクエリの基本的なアイデアを検討し，文献 15) で検索効率と辞書構築過程の詳細な分析を行ってきた．また，文献 14)，16) では，ローカルパターンの種類の分類と，日・英の複数のドメインに対する言語非依存性を評価した．本稿ではさらに，マルチシードクエリやローカルパターン，および検索戦略の有効性についての定量的な検証を行い，従来技術が採用してきたシングルシード

クエリとグローバルパターンとはまったく異なるアプローチによって，より検索効率と精度の高い辞書構築手法を実現していることを実証する．

その他の関連技術としては，HTML で記述されたテーブル構造やリスト構造など，Web ページの局所的な構造情報をパターンとして利用する Web ラッパ生成技術[17)–19)] があげられる．特に，SEAL[20)] は，文字列ベースのローカルパターンを利用している点で提案手法と類似している．しかし，Web ラッパ生成技術では，検索回数を節約するための検索戦略については考慮されていない．また，基本的には同一サイトもしくは同一ページ内での一貫性のあるフォーマットに注目した抽出を行っているだけで，提案手法が活用している異なるサイトや異なるページにおけるヘテロ性は考慮されていない．一方，サーチエンジンとクエリログを活用した同位語発見手法[21)] も存在するが，提案手法に比べると，得られる辞書の規模や精度の面でまだ課題がある．

3. CESS フレームワーク

図 1 に，CESS フレームワークの概要を示す．本フレームワークは，主に検索部と抽出部の 2 つのモジュールから構成されている．検索部は Query Builder と Crawler の 2 つのプロセスから構成されており，初期の入力としてシード語集合 $S = \{s_1, s_2, \ldots, s_n\}$ を受け付ける．Query Builder は，3.1 節で述べる検索戦略に従って ω 個の単語を S から選び出し，AND 検索用のクエリ Q を作成する．次に，Crawler が，WWW を介して検索 API に対してクエリ Q を発行し，検索結果の文書本体をクロールして文書群 D を作成し，抽出部の Pattern Generator に渡す．

1 回の検索で得られる文書群 D の件数は検索 API の仕様にもよるが，通常は 10〜100 件に制限されている．KnowItAll[7)] では，あるクエリ Q に対する検索結果のヒット件数が多

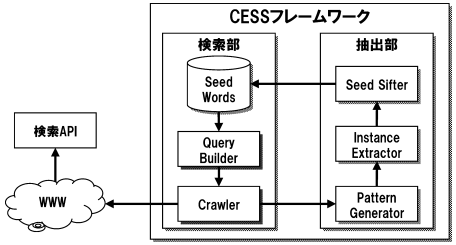

図 1　CESS フレームワークの概要図
Fig. 1　An overview of CESS framework.

数ある場合は，クエリ拡張などの手法を使って再帰的に繰り返し検索を行い，なるべく多くの検索結果を取得する戦略をとっている．一方，本フレームワークでは，1 つのクエリ Q に対する検索は 1 回しか行わない．その理由は，検索効率を高めるためである．たとえば，1 回の検索で返される URL リストの数が 100 件，検索回数が 3 回に制限されていたとしよう．この場合，クエリ $Q_1 = \{s_1 \cap s_2 \cap s_3\}$ で検索を 3 回行って Q_1 に関する検索結果を 300 件集めるのと，3 つのクエリ $Q_1 = \{s_1 \cap s_2 \cap s_3\}$，$Q_2 = \{s_4 \cap s_5 \cap s_6\}$，$Q_3 = \{s_7 \cap s_8 \cap s_9\}$ に関する検索結果をそれぞれ 100 件ずつ集めるのと，どちらがより多くの新語を集められるだろうか．直感的には，後者の方が同じ新語が重複して抽出される確率が少なく，より多くの新語が集まると期待できる．つまり，1 つのクエリ Q に対する検索は 1 回にしておき，より多くのクエリに対する検索結果を集める方が，検索効率が高いはずである．

また，Query Builder と Crawler はそれぞれ履歴情報を保持しており，重複したクエリの発行や，1 度取得した文書を再度取得することがないよう，効率化対策が行われている．

抽出部は，Pattern Generator，Instance Extractor，Seed Sifter の 3 つのプロセスから構成されている．抽出部の役割は，一貫性の高いフォーマットで記述されている構造化文書を見つけ，そこからなるべく多くの新語を抽出し，抽出パターンの多様性を基準にシード語として追加すべきか否かを判断することである．

Pattern Generator は，Crawler から渡された文書群 D 内の各文書 d について，各シード語の前後の一貫性の高い周辺文字列を 3.2 節で述べる方法によってパターン p として検出し，パターン集合 P に格納する．次に，Instance Extractor は，パターン集合 P 中の個別のパターン p について文書 d を照合し，パターン p で囲まれたシード語集合 S に未登録の新語 i を抽出する．Seed Sifter は，新語 i がどれだけ異なるパターン p から抽出されたかを異なり数 v として計数し，v が閾値 θ 以上であれば，新語 i をシード語集合 S に追加する．一般に，Web は分散・ヘテロな情報源であるため，異なるサイトの Web ページは異なるフォーマットで記述されているはずである．そのため，抽出パターンの異なり数 v が多いほど，新語 i は様々なサイトでシード語と同じ意味的階層に属する単語として記述されていると判断できる．本稿では，閾値 θ を**ヘテロ性判定パラメータ**と呼ぶ．

3.1 マルチシードクエリ

ここでは，マルチシードクエリとしてシード語から複数の語を選択する際の Query Builder の検索戦略について詳細に説明する．最も単純な検索戦略は，一様にランダムに ω 語を選ぶ方法である．一般的に，ω の値が大きくなるほど，シード語がリストやテーブルで構造化されたページが検索されやすい．しかし，ω が大きすぎると，検索結果は 0 件になってしま

```
QueryBuilder(S){
  SS ← ϕ
  while |SS| < ω
    j ← rand * |S|
    k ← rand
    if k < Score(S[j])^γ then
      remove S[j] from S and add to SS
    endif
  endfor
  Q ← combine SS into an AND query
  return(Q);
}
```

図 2　Query Builder の検索戦略アルゴリズム
Fig. 2　Algorithm of the search strategy in Query Builder.

うため，適切な ω の値を求める必要がある．本稿では，ω を**シードパラメータ**と呼ぶ．

一方，適切な ω を選んだとしても，単純な一様乱数に基づく検索戦略では，シード語集合が大規模になるとシード語どうしの低共起性が問題になってくる．たとえば，シード語が 1 万人の俳優名の辞書に成長した場合，1 万人の俳優名リストからランダムに 5 人の俳優名を選ぶとすると，多くのクエリに出現頻度の低い俳優名や，共起頻度の低い俳優名の組合せが含まれることになり，検索結果が 0 件になってしまう．また，低頻度語はシード語としての信頼性も低く，間違ったパターンが構築されてしまうというリスクもある．そこで本稿では，シードスコアのべき乗分布に従う乱数による検索戦略によって，高頻出・高信頼度のシード語を優先して選択する方法を提案する．あるシード語 s_j のシードスコア $Score(s_j)$ は，以下のように定義できる．

$$Score(s_j) = v_j / v_{\max} \tag{1}$$

ここで，v_j はシード語 s_j の抽出に使われたパターンの異なり数，$v_{\max}$ はシード語全体での抽出パターンの異なり数の最大値である．シードスコアの値の範囲は，$0 < Score(s_j) \leq 1$ であり，シードスコアが高いほど，シード語 s_j がより多くのページでより多様なフォーマットで記述されていることを意味する．つまり，シードスコアは近似的にシード語 s_j の出現頻度と信頼度を表している．

図 2 に，Query Builder による検索戦略アルゴリズムの詳細を示す．図 2 において，`rand` は $[0, 1)$ なる一様乱数を生成する関数とする．また，`^`はべき乗を表す演算子であるとする．

図 2 を見ると，Query Builder は，入力引数としてシード語集合 S を受け取り，初期化処理としてサンプルシード集合 SS を空集合に設定する．次に，シード語を選択するために，2 つの一様乱数 j と k を生成する．もし，$k < Score(s_j)^\gamma$ であれば，実際に j 番目のシード語 s_j をシード語集合 S からサンプルシード集合 SS に移動する．そうでなければ，また 2 つの一様乱数 j と k を生成して，シード語の選択に戻る．シードパラメータ ω 語分がサンプルシード集合 SS に選択されたら，SS 中の語を AND 演算子で結んでクエリ Q とする．

図 2 のアルゴリズムでは，j 番目のシード語が実際にクエリとして選ばれる確率は，シードスコアのべき乗に比例する．指数 γ を**検索戦略パラメータ**と呼ぶ．検索戦略パラメータ γ を大きくするほど，高頻出・高信頼度の語を優先してクエリを作成できる．以下では，検索戦略パラメータ γ によって，マルチシードクエリを構成する語のシードスコアの分布がどのように変化するかを理論的に考察する．

たとえば，1 万語のシード語があり，そのうちシードスコア 1.0 の語が 10 語，シードスコア 0.1 の語が 1,000 語あったとする．仮に $\gamma = 1$ と設定した場合，シードスコア 1.0 の語がクエリとして選択される確率 $Pr(1.0)$ は，$(10/10,000) \times 1.0^1 = 0.001$ である．一方，シードスコア 0.1 の語が選択される確率 $Pr(0.1)$ は，$(1,000/10,000) \times 0.1^1 = 0.01$ である．$Pr(1.0) < Pr(0.1)$ であるから，クエリ Q には，多くの低頻出語が混入するはずである．一方，$\gamma = 2$ と設定した場合，$Pr(1.0)$ は，$(10/10,000) \times 1.0^2 = 0.001$，$Pr(0.1)$ は，$(1,000/10,000) \times 0.1^2 = 0.0001$ となる．この場合，$Pr(1.0) > Pr(0.1)$ であるから，クエリ Q として高頻出語が優先して選択されるはずである．

上記の議論を定式化すると以下のようになる．シードスコアが t であるシード語の語数 $w(t)$ の分布が，一般的な単語の頻度分布と同様にべき乗則に従うと仮定すると，$w(t)$ は次式で表すことができる．

$$w(t) = at^{-b} \tag{2}$$

ここで，a，b は正の定数である．

さらに，シードスコア t のシード語がクエリに選ばれる確率 $Pr(t)$ は，上記の例を一般化して次式で表現できる．

$$Pr(t) = c \times \frac{w(t)}{|S|} \times t^\gamma \tag{3}$$

ここで，c は正規化定数であり，$\int_0^1 Pr(t)dt = 1$ となるように決定する．

今，シードスコア t の語がクエリとして選択される確率が t に比例するための検索戦略パラメータ γ の値を求める．この場合，$u = kt$ なる任意のシードスコア u に対して，$Pr(u) = kPr(t)$ となるため，以下の方程式が成立する．

$$c \times \frac{a(kt)^{-b}}{|S|} \times (kt)^{\gamma} = k \times c \times \frac{at^{-b}}{|S|} \times t^{\gamma} \quad (4)$$

式 (4) を解くと，γ は以下のように求まる．

$$\gamma = b + 1 \quad (5)$$

したがって，検索戦略パラメータが $\gamma = b + 1$ を満たす場合に，クエリに選ばれるシード語は，シードスコア t に比例した分布となる．また，$\gamma > b + 1$ とした場合は，より高頻度・高信頼度のシード語に偏ったクエリとなる．さらに，$\gamma = 0$ の場合は，一様乱数 j で指定されるシード語 s_j が必ず選ばれるので，本節の冒頭で述べた一様にランダムに ω 語を選ぶ検索戦略と一致する．

3.2　ローカルパターン

CESS フレームワークでは，ローカルパターンとして HTML タグベースではなく，文字列ベースの周辺文字列を採用している．これは，シード語が記述されている一貫したフォーマットとは，必ずしも HTML タグのテーブル構造やリスト構造だけではないからである．たとえば，映画のタイトルが年代とともに，「- モダンタイムス (1936)⟨BR⟩」，「- フランケンシュタインの花嫁 (1935)⟨BR⟩」，「- カサブランカ (1944)⟨BR⟩」のように列挙されている場合を考える．この場合，映画のタイトルだけを抽出するためには，HTML タグベースのパターンでは不十分である．一方，文字列ベースの左側パターンとして「- 」，右側パターンとして「(19」を使えば，上記の例における映画のタイトルを抽出できる．もちろん，2000 年代の映画が記述されていれば右側パターンとしては「(20」を使う必要もありうる．このように，1 つのページ内に出現する一貫したフォーマットは，必ずしも 1 つだけとは限らず，抽出のために複数のパターンを構築する必要もある．

Pattern Generator のアルゴリズムの詳細を**図 3** に示す．本アルゴリズムは，大きく，Pattern Detection と，Coherence Validation の 2 つの処理から構成されている．Pattern Generator は入力引数として単一の文書 d と，シード語集合 S を受け取り，最初に左側パターン集合 LP と，右側パターン集合 RP，および，抽出パターン集合 P をそれぞれ空集合として初期化する．また，シード語カウンタ $matchCnt$ を 0 に初期化する．

Pattern Detection 処理では，シード語集合 S に含まれるシード語 s について，文書 d 内の出現位置を照合し，出現していればシード語カウンタ $matchCnt$ を 1 つ増加させる．同時に，シード語 s より左側の 2〜50 文字の suffix を左側パターン集合 LP に追加し，シード語 s より右側の 2〜50 文字の prefix を右側パターン集合 RP に追加する．これを，全シード語の全出現位置について繰り返す．

```
PatternGenerator(d,S){
  LP ← ϕ
  RP ← ϕ
  P ← ϕ
  matchCnt ← 0

  // Pattern Detection
  for each s in S
    for each occurrence of s in d
      matchCnt ← matchCnt + 1
      for k = 2 to 50
        add k-letter suffix followed by s to LP
        add k-letter prefix following s to RP
      endfor
    endfor
  endfor

  // Coherence Validation
  remove all lp s.t. coherence(lp) < μ from LP
  remove all rp s.t. coherence(rp) < μ from RP

  add all LP-RP pairs <lp, rp> to P

  return P
}
```

図 3　Pattern Generator のローカルパターン生成アルゴリズム
Fig. 3　Algorithm of local pattern generation in Pattern Generator.

Coherence Validation 処理では，周辺文字列 xp の一貫性 $Coherence(xp)$ を次式で定義する．

$$Coherence(xp) = \frac{Occurence(xp)}{matchCnt} \quad (6)$$

ここで，周辺文字列 xp としては，左側パターン lp，右側パターン rp の両方があてはまる．また，$Occurence(xp)$ は，文書 d 内でシード語 s が周辺文字列 xp に隣接する回数である．

式 (6) は，$Coherence(xp)$ が高いほど，文書 d 内においてシード語 s の周辺文字列 xp が一貫したフォーマットで記述されていることを意味している．一方，シード語 s が構造化されていない自然文中に現れている場合は，多様な周辺文字列が存在するため，ある周辺文字列 xp の一貫性 $Coherence(xp)$ は低くなる．したがって，Coherence Validation 処理では，$Coherence(xp)$ が閾値 μ 以上の場合にのみ，周辺文字列 xp を抽出パターンとする．閾値 μ を**一貫性判定パラメータ**と呼ぶ．

最後に，Pattern Generator は，左側パターン集合 LP と右側パターン集合 RP に含まれる周辺文字列の組合せ $< lp, rp >$ を抽出パターン集合 P に格納して Instance Extractor に渡す．抽出パターン集合 P は処理対象となる文書 d に強く依存しているため，新語を抽出するために P を適用するのはそれが由来する文書 d 内だけで使い捨てとなる．

3.3　ノイズ語に対するロバスト性

ここでは，CESS フレームワークの，ノイズ語に対するロバスト性について議論する．ブートストラップ式辞書構築では，パターン構築と新語抽出の繰返しの途中で，ターゲットとする辞書のドメインとは関係ないノイズ語が誤抽出され，シード語集合に混入するおそれがある．ノイズ語を使って構築した抽出パターンは新たなノイズ語を生み出すため，連鎖的に辞書の品質が悪化してしまう．CESS フレームワークでは，Seed Sifter におけるヘテロ性判定パラメータ θ が，ノイズ語のシード語集合への混入を防ぐ．また，運悪くノイズ語がシード語集合に混入してしまったとしても，マルチシードクエリとローカルパターンによって，ノイズ語の悪影響を抑える工夫がなされている．

CESS フレームワークにおいては，シード語集合に混入したノイズ語が悪影響を及ぼす可能性があるのは，（A）クエリ汚染，（B）抽出パターン汚染のケースである．

（A）クエリ汚染とは，検索クエリとしてノイズ語が使われてしまうケースである．たとえば，シード語集合のうち 90%が正しい固有表現，10%がノイズ語であったとする．従来手法のように，単一シード語をクエリとして選択する場合，ノイズ語をクエリとして，ドメインに関係ない検索結果を得る確率は 0.1 である．一方，CESS フレームワークにおいて，たとえば 3 語をマルチシードクエリとして利用する場合，3 語すべてがノイズ語になってしまう確率は $0.1^3 = 0.001$ しかない．また，3 語のうち 2 語がノイズ語である確率は ${}_3C_2 \times 0.1^2 \times 0.9 = 0.027$ であり，依然として小さい．一方，3 語のうち 1 語だけがノイズ語である確率は ${}_3C_1 \times 0.1 \times 0.9^2 = 0.243$ であるが，クエリの半数以上が正しい固有表現であるため，検索結果はある程度ドメインに関係するはずである．一般に，シードパラメータ ω を大きくするほど，ノイズ語によるクエリ汚染の悪影響は小さくなる．

（B）抽出パターン汚染とは，シード語集合に含まれるノイズ語によって，間違った抽出パターンが構築されてしまうケースである．この場合，一貫性判定パラメータ μ が，間違った抽出パターンが構築されるのを防ぐ．たとえば，文書 d が，日記のような自然文で記述されている非構造化文書であった場合は，シード語集合にノイズ語が混入されていても，もともと一貫性判定パラメータ μ を超えるような一貫性を持つ抽出パターンは存在しないため，ノイズ語の悪影響は小さい．また，文書 d が，リストページのように構造化された文書であった場合は，ノイズ語の周辺文字列によって正しい抽出パターンの一貫性が低く評価され，一貫性判定パラメータ μ を下回る可能性はあるが，それによって間違った抽出パターンが構築されてしまう可能性は低い．また仮に，間違った抽出パターンが構築されてしまったとしても，従来手法と違って使い捨てなので，悪影響の範囲はパターンが構築された文書内だけにとどめられる．

4.　評価実験と考察

CESS フレームワークにおけるブートストラップ式辞書構築アルゴリズムを Perl で実装し，評価実験を行った．辞書構築のターゲットとした言語は英語と日本語，ドメインは `Actor`，`Film`，`Location`，`Company`，`School` の 5 種類とした．シード語の初期集合の選択方法の恣意性に関する議論を避けるため，本稿では，筆者らが意図的に初期集合を構築するのではなく，すでに存在する固有表現のランキングを利用した．

英語の `Actor` ドメインでは，2004 年〜2006 年のアカデミー賞を受賞した 10 人の俳優名[⋆1]を，日本語の `Actor` ドメインでは，2005 年〜2007 年に日本アカデミー賞を受賞した 10 人の俳優名[⋆2]をそれぞれ初期シード語集合とした．英語の `Film` ドメインでは，1997 年〜2006 年にアカデミー監督賞を受賞した 10 作品のタイトルを，日本語の `Film` ドメインでは，1998 年〜2007 年に日本アカデミー賞最優秀作品賞を受賞した 10 作品のタイトルをそれぞれ用いた．英語の `Location` ドメインでは，2007 年に発表された世界の最も生活費が高い都市の上位 10 都市[⋆3]を，日本語の `Location` ドメインでは，英語と同じシード語集合を日本語表記にしたものを用いた．英語の `Company` ドメインでは，2007 年に発表された Fortune 500 の上位 10 社[⋆4] を，日本語の `Company` ドメインでは，日経ナビ 2008 で発表された人気企業ランキングの上位 10 社[⋆5]を用いた．英語の `School` ドメインでは，2006 年に発表された「The Top American Research Universities」のランキング上位 10 大学[⋆6]を，日本語の `School` ドメインでは，DBWeb2006 [⋆7]の発表者の所属大学をプログラムの先頭から 10 大学選んで初期シード語集合とした．

また，Web サーチエンジンとしては，Yahoo! Web API [⋆8]を用いた．検索 API のパラメータとしては，基本的な機能のみを利用した．具体的には，1 回の検索で返される文書の最大件数 *result* を 50，検索結果の先頭位置 *start* を 0，検索対象とするファイルの種類

⋆1 http://www.oscar.com/
⋆2 http://www.japan-academy-prize.jp/
⋆3 http://www.mercer.com/costofliving/
⋆4 http://money.cnn.com/magazines/fortune/fortune500/
⋆5 http://job.nikkei.co.jp/2008/
⋆6 http://mup.asu.edu/research2006.pdf
⋆7 http://db-www.naist.jp/dbweb2006/
⋆8 http://developer.yahoo.co.jp/

表 1　抽出パターンの例
Table 1　Examples of local patterns.

	パタン例
非タグ (49.9%)	(例1) , Keanu Reeves, (例2) title="View all posts filed under Keanu Reeves">< (例3) alt="Keanu Reeves" style="text-decoration;">
両側タグ (34.6%)	(例4) .htm">Keanu Reeves</A><IMG SRC= (例5) ">Keanu Reeves</option> <option value=" (例6) /">Keanu Reeves <a href="
一部タグ (15.5%)	(例7) Starring Keanu Reeves (例8) - Keanu Reeves (例9) 　 Keanu Reeves (19

format を html，検索対象の言語 *language* は辞書のターゲット言語が英語の場合は en，日本語の場合は jp を設定した．

4.1　抽出部のパラメータ決定

同一条件で様々な検索戦略を比較するため，最初に抽出部における，一貫性判定パラメータ μ，およびヘテロ性判定パラメータ θ の決定を行った．予備実験として，日本語の `Actor` ドメインの初期シード語集合から 3 語ずつランダムに選んで 20 種類のクエリを作成し，それぞれ 1 回ずつ検索を行って 1,000 件の Web ページを得た．この 1,000 ページに対して，各パラメータの値を変化させながらシード語として追加される新語の数と正解率を測定した．その結果，正解率 90%以上を達成し，新語の数が最も多くなるパラメータの組合せとして $(\mu, \theta) = (0.5, 5)$ を得た．以下，これら 2 つのパラメータを固定して実験を行った．

英語の `Actor` ドメインにおける抽出パターンの例を**表 1** に示す．抽出パターンのうち，新語が HTML タグに挟まれていない「非タグ」のケースはほぼ半数を占めた．具体例としては，「,」やセミコロン「;」やストローク「|」のような区切り文字によるパターン（例 1）や，リンクの title 属性や画像の alt 属性中に記述されるパターン（例 2，3）などがあった．一方，新語が HTML タグに正確に挟まれている「両側タグ」のケースは，全抽出パターンの 34.6%であった．具体例としては，リンクを指定する ⟨A⟩ タグ（例 4）やオプションを指定する ⟨OPTION⟩ タグ（例 5）に挟まれたケースが大半であった．一方，リストを指定する ⟨LI⟩ タグやテーブルの項目を指定する ⟨TD⟩ タグを含む抽出パターン（例 6）は抽出パターン全体の 7%しか存在しなかった．また，パターンの一部にタグを含む「一部タグ」のケースは全体の 15.5%であった．具体例としては，俳優名の前に「Starring」という文字列が付与されていたり（例 7），ハイフン「-」で列挙されていたり（例 8），俳優名の後に年度を表す「(19」という文字列が付与されているパターン（例 9）があった．以上より，HTML タグだけでは見つけられないパターンも検出可能であることから，文字列ベースによる抽出パターンの方がより多くの新語を抽出できることが分かる．

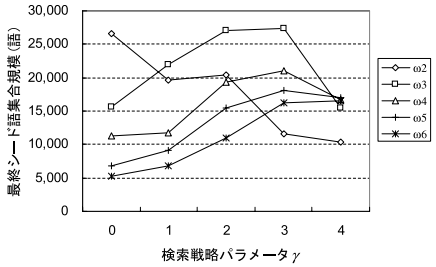

図 4　検索戦略による最終シード語集合の規模の違い
Fig. 4　Final size of seed sets based on different search strategies.

4.2　辞書構築の基本結果

検索戦略の評価のために，検索回数を 200 回に限定して実験を行った．ここでは，辞書構築のプロセスの概要を理解するために，英語の `Actor` ドメインにおける結果について，詳細に述べる．

図 4 に，シードパラメータ ω と検索戦略パラメータ γ を変化させた場合の，最終シード語集合の規模の違いを示す．検索戦略パラメータ $\gamma = 0$ は，一様なランダムサンプリングに基づくマルチシードクエリである．この場合，シードパラメータ ω を 2 とした場合に最大規模の 26,564 語が得られ，ω が大きくなるにつれて最終シード語集合は小さくなった．これは，3.1 節で述べたように，ω が大きいと低共起性の問題が大きく，ほとんどのクエリで検索結果が 0 件になってしまうからである．

検索戦略パラメータが $1 \leq \gamma \leq 3$ の範囲では，$\omega \geq 3$ の最終シード語集合は増加している．これは，検索戦略パラメータ γ の増加にともなって，高頻度・高信頼度語が優先的にクエリに使われ，検索結果が増えたからである．ところが，シードパラメータ $\omega = 2$ の場合は，逆に最終シード語集合の規模は減少している．この理由は以下のように考えられる．前述のように，2 語のマルチシードクエリでは $\gamma = 0$ でも検索結果が十分存在していたため，高頻度語を優先する効果は小さい．一方で，同じ高頻度語を含む類似したクエリで何度も検索を行う結果，抽出される語の重複が増えて新語の発見効率が落ちる．トータルで見ると，$\omega = 2$ の場合は，重複によるネガティブな効果の方が大きく，結果として最終シード語集合の規模が減少する．

表 2　異なる言語・ドメインにおける結果の概要
Table 2　Overview of experimental result in different languages and domains.

	英語		日本語	
	最終シード語	精度	最終シード語	精度
Actor	27,344	0.93	10,944	0.85
Film	17,592	0.97	12,011	0.98
Location	6,496	0.92	2,646	0.87
Company	3,422	0.96	10,620	0.68
School	9,788	0.91	1,865	0.84
合計	64,642	0.94	38,086	0.84

検索戦略パラメータが $\gamma = 4$ になると，$\omega \geq 3$ でも最終シード語集合が減少した．これは，$\omega = 2$ の場合と同様に，高頻度語の優先による検索結果増加の効果が飽和する一方，重複によるネガティブな効果が支配的になるからである．最終的に，$(\omega, \gamma) = (3, 3)$ の場合に，最終シード語集合 27,344 語と最大になった．

最終シード語集合のサイズでは，$(\omega, \gamma) = (2, 0)$ と $(3, 3)$ の場合はほぼ同程度の規模の辞書が得られているが，精度の比較をすると，$(\omega, \gamma) = (2, 0)$ では適合率が 0.81 であったのに対して，$(\omega, \gamma) = (3, 3)$ の場合は適合率は 0.93 であった．したがって，本ドメインにおいては，最終シード語の規模の面でも精度の面でも $(\omega, \gamma) = (3, 3)$ の結果が優れている．

4.3　言語・ドメイン別辞書構築結果

異なる言語，および異なるドメインにおける結果の概要を**表 2** に示す．本実験では，前節での実験結果をふまえてシードパラメータ ω と検索戦略パラメータ γ を，$(\omega, \gamma) = (3, 3)$ に固定して 200 回の検索を行った．また，最終的に得られたシード語集合の規模と，ランダムサンプリングした 500 語について適合率を評価した．

英語の `Actor` ドメインでの結果は，前節で詳しく述べたとおりである．日本語の `Actor` ドメインでは，最終的に 10,944 語が抽出された．Amazon および Google を使って人手チェックを行い，実際に俳優として出演している映画もしくはテレビドラマが確認できた語を正解としたところ，適合率は 0.85 であった．典型的な不正解の例としては，映画の監督名，原作者名，映画タイトル，ミュージシャン名，政治家名などがあった．政治家名など，映画に関係ない不正解語が抽出されたページについて詳細に調べてみると，有名人の誕生日のデータベースのようなサイトも存在したが，多くは，検索エンジン最適化のために過剰に人名を埋め込んだページであった．

英語の `Film` ドメインでは，最終的に 17,592 語が抽出された．IMDb および Google を使って人手チェックを行い，実際に映画もしくはテレビドラマ作品であると確認できた語を正解としたところ，適合率は 0.97 であった．典型的な不正解の例は，タイプミスであった．また，日本語の `Film` ドメインでは，最終的に 12,011 語が抽出された．Amazon および Google を使って英語と同じ判定基準で人手チェックを行ったところ，適合率は 0.98 であった．典型的な不正解の例としては，俳優名やミュージシャン名であった．

英語の `Location` ドメインでは，最終的に 6,496 語が抽出された．Google を使って人手チェックを行い，実在する都市名もしくは観光地名であると確認できた語を正解としたところ，適合率は 0.92 であった．典型的な不正解の例としては，「Africa」などの広域な地域名，「Poland」などの国名であった．一方，日本語の `Location` ドメインでは，最終的に 2,646 語が抽出された．Google を使って英語と同じ判定基準で人手チェックを行ったところ，適合率は 0.87 であった．典型的な不正解の例としては，英語の場合と同じく，「南アメリカ」などの広域な地域名，「オランダ」などの国名などがあったが，そのほかに，日本語に特有のノイズ語として「カンタス航空」などの航空会社名が含まれていた．これは，海外の地名が構造化されて記載されている日本語ページは，旅行会社の航空チケット販売のページであるケースが多いからであると考えられる．

英語の `Company` ドメインでは，最終的に 3,422 語が抽出された．Google を使って人手チェックを行い，実在する企業を確認できた語を正解としたところ，適合率は 0.96 であった．典型的な不正解としては，国名や「Software」「Network」のような一般名詞であった．日本語の `Company` ドメインでは，最終的に 10,620 語が抽出された．Google を使って英語と同じ判定基準で人手チェックを行ったところ，適合率は 0.68 と，他のドメインに比べて著しく低い値となった．典型的な不正解の例としては，株式コードと思われる数字，「食品系メーカー」などの業界を表す表現，「警備員」などの職種を表す表現，「広島」などの地名，「薬」などの一般名詞であった．業界や職種を表す表現は，日本語に特有の不正解語であったため，これらが出現しているページを詳細に調べたところ，ほとんどが検索エンジン最適化のために過剰に就職・転職関連のリンクを埋め込んだページであった．

英語の `School` ドメインでは，最終的に 9,788 語が抽出された．Google を使って人手チェックを行い，実際に大学または教育機関であると確認できた語を正解としたところ，適合率は 0.91 であった．典型的な不正解の例は，国名や州名であった．日本語の `School` ドメインでは，最終的に 1,865 語が抽出された．Google を使って英語と同じ判定基準で人手チェックを行ったところ，適合率は 0.84 であった．典型的な不正解の例としては，「岡山」などの地名や「基礎工学部」などの学部名であった．

以上の結果をまとめると，英語に関しては，合計で 64,642 語を平均 0.94 の適合率で抽出できた．200 回の繰返し処理後にもかかわらずこれほど高い適合率を達成できたのは，3.3 節で述べたロバスト性が主な要因である．一方，Riloff ら[6] の実験では，50 回の繰返し処理で辞書の適合率が 0.46〜0.76 程度に低下している．また，KnowItAll[7] も同じく `Actor` および `Film` ドメインでの辞書構築を試みているが，最終的な辞書の全単語を対象とした適合率はそれぞれ 0.7，0.5 である．これらの比較からも，提案手法は高いロバスト性を有しているといえる．なお，検索技術の分野では再現率も重要な指標であるが，あるドメインに対して辞書に登録すべき真の単語集合を求めることは困難であり，真の再現率を評価することは不可能である．KnowItAll[7] の評価実験では，最終的な辞書の全単語を暫定的に再現率 1.0 と見なして適合率–再現率曲線を求めている．一方，本研究では，辞書の規模を真の再現率の目安と見なし，適合率とあわせて評価基準とする立場をとっている．

日本語に関しては，合計で 38,086 語を平均 0.84 の適合率で抽出できた．英語と比較して語数が少ないのは，英語ページが WWW 全体の 60%近く存在するのに対して，日本語ページの占める割合は 3.5%しかない[22] ことが原因として考えられる．適合率についても，日本語の方が英語よりも低かった．不正解となったノイズ語のうち，日本語に特有のノイズ語が含まれるページとしては，`Company` ドメインでの例のように，過剰に検索エンジン最適化を施した Web ページが目立っていた．このような過剰な検索エンジン最適化ページに対する対策は，今後の課題である．また，今回の実験結果から，提案手法は Web ページ内にリストとして記述される可能性の高い固有名詞については高い検索効率で収集できることが分かったが，一方で，一般名詞を収集してシソーラスを構築するなどの用途には向いていない可能性がある．

最終的に，日・英あわせて 10 ドメインについて，合計 2,000 回の検索で 102,728 語の辞書を適合率 0.90 で構築できた．このとき，クロールした全ページ数は 50,340 件であったので，1 ページあたり約 2 語の新語を抽出できたことになる．一方，KnowItAll[7] は，正確な検索回数は公表していないが，54,753 語の辞書を構築するために，313,349 ページをクロールしており，1 ページあたりの新語抽出件数は 0.2 語にも満たない．対象としているドメインが若干異なるために単純な比較はできないが，本事実も提案手法の有効性を示しているといえる．

4.4　マルチシードクエリの有効性

図 3 に示した Pattern Generator に対する入力であるシード語集合 S は，クエリに使われたシード語だけでなく，全シード語を含む集合である．そのため，1 語のみからなるシングルシードクエリを用いて検索結果文書 d を取得し，ローカルパターンを生成することも原理的には可能である．そこで，マルチシードクエリの有効性検証のため，前節にあげた言語・ドメインにおいてシードパラメータ $\omega = 1$ とした実験も行った．その結果，シングルシードクエリでは，英語の `Location` ドメインで 386 語，日本語の `Film` ドメインで 1 語だけシード語が増加したのみで，他のすべてのドメインにおいてシード語は 1 語も増加しなかった．

この理由は，シングルシードクエリに対する検索結果には，クエリに使われたシード語単独のトピックに関連する公式ページや詳細情報のページが大量に含まれているからである．単独トピックに関連する公式ページには，他のシード語のリストはほとんど含まれていないため，ローカルパターンは生成されず，その結果シード語は増加しない．英語の各ドメインにおいて実際に使われたシングルシードクエリの例と，それに対する検索結果上位 3 件の文書のタイトルと URL を**表 3** に示す．

`Actor` ドメインでは，クエリに使われた俳優の公式ページや，映画データベースにおける俳優個人の情報のページが検索結果上位を占めていることが分かる．`Film` ドメインでは，クエリに使われた映画の公式ページやパロディのページ，また，Wikipedia などの詳細情報

表 3　シングルシードクエリに対する検索結果の例
Table 3　Example of search result for single seed queries.

Actor "Jamie Foxx"	The Official Jamie Foxx Website http://www.jamiefoxx.com/
	Jamie Foxx http://www.imdb.com/name/nm0004937/
	The Official Jamie Foxx Website http://www.jamiefoxx.com/music
Film "Titanic"	The official Titanic site by Paramount Pictures and Twentieth Century Fox. http://www.titanicmovie.com/
	RMS Titanic - Wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/RMS_Titanic
	Titanic in 30 seconds with bunnies. http://www.angryalien.com/0604/titanicbunnies.html
Location "Tel-Aviv"	Welcome to Tel-Aviv/Yafo New Site http://www.tel-aviv.gov.il/english/home.asp
	Tel Aviv - Wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/Tel_Aviv
	Welcome to Tel-Aviv/Yafo New Site http://www.tel-aviv.gov.il/English/Tourism/Sites/Index.htm
Company "Bank of America"	Bank of America | Home | Personal http://www.bankofamerica.com/index.cfm
	Bank of America | Home | About Bank of America http://www.bankofamerica.com/index.cfm?page=about
	Bank of America | Newsroom http://newsroom.bankofamerica.com/
School "Stanford University"	Stanford University http://www.stanford.edu/
	Department of Mathematics - Stanford University http://math.stanford.edu/
	Stanford University | Chemistry Department http://www.stanford.edu/dept/chemistry/

が検索結果上位を占めていた．また，Location ドメインでは，クエリに使われた地名に関する自治体や観光情報のページが，Company ドメインでは，クエリに使われた企業のサイトやその企業のサブドメインのサイトが検索上位を占めていた．さらに，School ドメインでは，クエリに使われた大学のホームページや，その大学における各学部のホームページが検索結果の上位を占めていた．

この傾向は，他のシングルシードクエリでも同様であり，また，上位 3 件以降の検索結果についても同様であった．したがって，シード語がリストとして記述されているページの割合はきわめて少なく，その結果，ローカルパターンもほとんど形成されないままに辞書構築プロセスが終了してしまうのである．以上のことから，CESS フレームワークにおけるマルチシードクエリの有効性が検証できたといえる．

4.5　ローカルパターンの有効性

CESS フレームワークでは，ローカルパターンの適用範囲をそれが生成されたページ内だけに限定し，使い捨てにすることによって，新語の抽出精度を保っている．この使い捨て効果の有効性を検証するため，ローカルパターンを使い捨てにせず，グローバルパターンとして全文書集合に適用した場合の抽出精度を評価した．

あるローカルパターン lpt の信頼度 $Conf(lpt)$ を，そのパターンが抽出した新語の数 N_{ext} に対して，実際に最終シード語集合に含まれた語の数 N_{seed} の比と定義すると，次式で表すことができる．

$$Conf(lpt) = \frac{N_{seed}}{N_{ext}} \tag{7}$$

4.2 節で実験した，英語の Actor ドメインで $(\omega, \gamma) = (3, 3)$ の場合に生成されたローカルパターン lpt のうち，1,000 語以上抽出したパターンを信頼度 $Conf(lpt)$ の順に並べると，上位 10 件のローカルパターンは**表 4** のようになる．たとえば，パターン「”⟨br⟩[新語] -」は 1,550 語の新語を抽出しており，そのうち 1,527 語が最終シード語集合に含まれており，1,527/1,550 = 0.99 という信頼度を持っていることが分かる．

表 4 をさらに詳しく見ると，信頼度の高いローカルパターンには「 page1.html¨⟩[新語]⟨/a⟩⟨br /⟩⟨a class=”AAJ”」のように，特定フォーマットのページに依存した特殊パターンと，「”⟨br⟩[新語] -」や「/”⟩[新語]⟨/a⟩⟨」のように，どのページでもマッチする可能性のある汎用パターンの 2 種類があることが分かる．

前者のような特殊パターンをグローバルパターンとして使った場合は，ローカルパターンと同様に，特定フォーマットのページからしか新語が抽出されないと推測される．一方，後者のような汎用パターンをグローバルパターンとして使ってしまうと，新語として抽出したい部分以外にもパターンがマッチしてしまい，大量のノイズ語を発生させることになってしまう．実際に，表 4 に示す 10 件のパターンをグローバルパターンとして全文書集合に適用し，新語として抽出されたページ数が多い上位 200 語について，実際に俳優名か否かを判定したところ，適合率は 30%にとどまった．グローバルパターンの適用範囲を全文書集合ではなく，「少なくとも 1 語以上のシード語が抽出された文書」に限定しても，適合率は 52%であった．典型的なノイズ語としては，「Home」や「Contact Us」のようにナビゲーションリンクとしてよく用いられる表現や，「Celebrity News」や「AOL Music」のような，俳優名がよく出現するサイトの名前が大半であった．

表 4　高信頼ローカルパターンの上位 10 件
Table 4　Top 10 local patterns with higher conficence.

ローカルパタン lpt	N_{ext}	N_{seed}	$Conf(lpt)$
” [新語] -	1550	1527	0.99
/”>[新語]<	2016	1958	0.97
/>[新語]</td> <	1270	1239	0.98
¨page1.html”>[新語] <a class=”AAJ”	3837	3710	0.97
.shtml”>[新語]<	1495	1441	0.96
”>[新語] <a href=”//www.welove	4571	4349	0.95
>[新語]　　(	1288	1232	0.96
</td></tr><tr><td>[新語]</td><td>ACT</td>	1162	1114	0.96
”>[新語]</option> <option value=”	8316	7848	0.94
/” title=””>[新語] (	2183	2054	0.94

以上のことから，提案フレームワークで生成される抽出パターンを使い捨てにせずグローバルパターンとして使ってしまうと，汎用パターンがノイズ語を多く抽出してしまい，再現率の向上よりも適合率の低下の効果が顕著になってしまうことが分かる．また，逆に，CESS フレームワークにおいては，汎用パターンであっても，それが生成されたページ内だけで使い捨てにすることによって，信頼性の高いパターンとして利用できていることが示された．

4.6　検索戦略の有効性

検索戦略パラメータの効果をさらに詳しく検証するために，4.2 節で実験した英語の Actor ドメインのほかに，Film，Location，Company，School ドメインについても，シードパラメータを $2 \geq \omega \geq 4$ の範囲で，検索戦略パラメータを $0 \geq \gamma \geq 4$ の範囲で変化させながら，それぞれ 200 回の検索に限定して辞書構築を行った．各ドメインにおいて最大規模の最終シード語が得られたときのシードパラメータ ω^* と検索戦略パラメータ γ^* の組合せを**表 5** に示す．

表 5 を見ると，いずれのドメインにおいても，最大規模の最終シード語が得られたシー

表 5　各ドメインにおける最適パラメータ
Table 5　Optimal parameters in each domain.

	γ^*	ω^*	最終シード語	精度	b
Actor	3	3	27,344	0.93	2.3
Film	3	3	17,592	0.97	2.8
Location	0	3	37,317	0.98	2.9
Company	1	3	4,984	0.93	2.4
School	2	3	20,335	0.93	1.7

ドパラメータ ω^* の値は 3 であった．これは，3 語のシード語からなるクエリでは，検索結果が 0 件になるケースが比較的低いのに加え，リストページが検索結果に含まれやすいからだと考えることができる．

一方，最大規模の最終シード語が得られた検索戦略パラメータ γ^* の値はドメインによってばらつきがあり，Actor と Film ドメインでは $\gamma^* = 3$ であったが，Location ドメインでは $\gamma^* = 0$，Company ドメインでは $\gamma^* = 1$，School ドメインでは $\gamma^* = 2$ であった．ただ，これら 5 つのドメインのうち，単純な一様乱数によるランダムサンプリングが最適であったのは Location ドメインだけであり，他の 4 つのドメインでは $\gamma^* \geq 1$ であった．このことから，シード語を選択する際に，ランダムに 2 語選ぶよりも，3 語以上を高頻度語を優先して選択する検索戦略の有効性が確認できた．なお，本稿では，べき乗分布以外の高頻度語優先方式の比較は行っていないため，べき乗分布が最適であるとは限らない．したがって，高頻度語を優先するための最適な関数やそのパラメータは今後の課題である．

表 5 では，最適パラメータの組合せ (ω^*, γ^*) における精度の評価結果も示している．4.2 節でも指摘したが，Actor ドメインにおける $(\omega, \gamma) = (2, 0)$ と $(\omega^*, \gamma^*) = (3, 3)$ の適合率に差があったほか，Company ドメインにおいて，$(\omega, \gamma) = (2, 0)$ での適合率が 0.80 であったのに対して，$(\omega^*, \gamma^*) = (3, 1)$ での適合率は 0.93 と高い値を示した．しかし，他のドメインでは，最適パラメータにおける精度と，他のパラメータの結果と比較してもあまり大きな差は見られなかった．

また，表 5 では，最終シード語集合におけるシードスコア t を持つ語数 $w(t)$ の分布を，べき乗関数 $w(t) = at^{-b}$ でフィッティングした結果得られた b の値も示している．これを見ると，b の値はどのドメインにおいても 2〜3 程度の値で一定であり，最適な検索戦略パラメータ γ^* との明確な相関関係はあまり見られなかった．しかしながら，表 5 に示した最適パラメータによる最終シード語集合の合計は 107,572 語であり，4.3 節で求めた最終シード語集合 64,642 の 1.7 倍であるので，ドメインに依存した最適検索戦略パラメータ γ^* を効率良く決定することは重要であり，提案フレームワークにおける今後の課題である．

5. 結　　論

本稿では，検索回数を節約できる検索戦略を備えたブートストラップ式辞書構築手法として，CESS フレームワークを提案した．提案手法の特徴は，複数のシード語を組み合わせて AND 検索するマルチシードクエリと，使い捨てのローカルパターンの 2 点である．マルチシードクエリについては，シードスコアのべき乗分布に従う検索戦略について議論した．ローカルパターンについては，ページ内のフォーマットの一貫性の計算方法を定義し，ノイズ語に対するロバスト性について考察した．実験では，日・英それぞれ 5 種類のドメインについて，200 回の検索で構築される辞書の規模と適合率を評価した．その結果，合計 2,000 回の検索で，10 万語以上の辞書を適合率 0.9 以上で構築できた．また，提案手法の特徴である，マルチシードクエリ，使い捨てのローカルパターン，および，べき乗分布に従う検索戦略について，定量的に有効性を検証した．その結果，従来技術が採用してきたシングルシードクエリやグローバルパターンとはまったく異なるアプローチによって，より検索効率と精度の高い辞書構築手法を実現できていることを実証した．最適パラメータの選定によって，さらに大規模な辞書を効率良く構築できる可能性があるため，検索戦略パラメータの決定方法を工夫することが今後の課題である．

参 考 文 献

1) Grishman, R. and Sundheim, B.: Message Understanding Conference – 6: A Brief History, *The 16th International Conference on Computational Linguistics* (1996).
2) 竹元義美，福島俊一，山田洋志：辞書およびパターンマッチルールの増強と品質強化に基づく日本語固有表現抽出，情報処理学会論文誌，Vol.42, No.6, pp.1580–1591 (2001).
3) Riloff, E.: Automatically constructing a dictionary for information extraction tasks, *The 11th National Conference on Artificial Intelligence*, AAAI, pp.811–816, AAAI Press/The MIT Press (1993).
4) Soderland, S., Fisher, D., Aseltine, J. and Lehnert, W.W.: Crystal: Inducing a conceptual dictionary, *The 14th International Joint Conference on Artificial Intelligence*, pp.1314–1319 (1995).
5) Califf, M.E. and Mooney, R.J.: Relational learning of pattern-match rules for information extraction, *AAAI Spring Symposium on Applying Machine Learning to Discourse Processing* (1998).

6) Riloff, E. and Jones, R.: Learning dictionaries for information extraction using multi-level bootstrapping, *The 16th National Conference on Artificial Intelligence*, AAAI, pp.1044–1049, AAAI Press/The MIT Press (1999).
7) Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.-M., Shaked, T., Soderland, S., Weld, D.S. and Yates, A.: Web-scale information extraction in KnowItAll, *The 13th International Conference on World Wide Web* (2004).
8) Pasca, M., Lin, D., Bigham, J., Lifchits, A. and Jain, A.: Organizing and searching the World Wide Web of facts – step one: the one-million fact extraction challenge, *The 21st National Conference on Artificial Intelligence*, AAAI, pp.1400–1405, AAAI Press/The MIT Press (2006).
9) O'Reilly, T.: What is Web 2.0: Design patterns and business models for the next generation of software. http://www.oreillynet.com/lpt/a/6228
10) 楠村幸貴，土方嘉徳，西田正吾：テンプレートの交叉と DOM 構造の解析による情報抽出手法の提案，第 17 回データ工学ワークショップ（DEWS）(2006).
11) Brin, S.: Extracting patterns and relations from the World Wide Web, *The International Workshop on the World Wide Web and Databases*, pp.172–183 (1998).
12) Soderland, S., Etzioni, O., Shaked, T. and Weld, D.: The use of Web-based statistics to validate information extraction, *AAAI workshop on Adaptive Text Extraction and Mining* (2004).
13) 水口弘紀，河合英紀，土田正明：Web 知識を利用したブートストラップによる辞書増殖手法，第 18 回データ工学ワークショップ（DEWS）(2007).
14) 河合英紀，水口弘紀，土田正明：ブートストラップ式辞書構築における検索効率の向上，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）(2007).
15) Kawai, H., Mizuguchi, H. and Tsuchida, M.: Cost-Effective Search Strategy Framework for Automatic Dictionary Construction, *The International Conference on Database Systems for Advanced Applications*（*DASFAA*）(2008).
16) 河合英紀，水口弘紀，土田正明，國枝和雄，山田敬嗣：ブートストラップ式同位語増殖における検索戦略の研究，*DBSJ Letters*, Vol.6, No.4 (2008).
17) Kushmerick, N.: Wrapper induction: efficiency and expressiveness, *AAAI Workshop on AI and Information Integration* (1998).
18) 野口正人，廣川佐千男：Web からの同系統単語知識獲得についての実験，情報処理学会第 65 回全国大会講演論文集，pp.223–226 (2003).
19) Chuang, S.-L., Chang, K.C.-C. and Zhai, C.: Context-Aware wrapping: synchronized data extraction, *The 33rd Very Large Data Bases Conference* (2007).
20) Wang, R. and Cohen, W.: Language-independent set expansion of named entities using the Web, *IEEE International Conference on Data Mining* (2007).
21) 大島裕明，山口雅史，小山　聡，田中克己：Web 検索エンジンのインデックスとクエリログを用いた同位語発見，情報処理学会データベースと Web 情報システムに関するシンポジウム（DBWeb2006）論文集，pp.305–312 (2006).
22) 池内　淳：ウェブ統計調査におけるサーチエンジンの有効性，2004 年度日本図書館情報学会春季研究集会発表要綱 (2004).

(平成 19 年 12 月 20 日受付)
(平成 20 年 4 月 14 日採録)

(担当編集委員　有次 正義，DBWeb2007 推薦論文)

河合　英紀（正会員）

昭和 49 年生．平成 10 年慶應義塾大学大学院理工学研究科計測工学専攻修士課程修了．同年日本電気株式会社入社．平成 17 年から平成 18 年にかけて米国 Stanford 大学客員研究員．情報統合に関する研究に従事．現在，NEC C&C イノベーション研究所主任．情報検索，Web マイニングに関する研究に従事．平成 13 年情報処理学会全国大会奨励賞，平成 16 年データ工学ワークショップ優秀論文賞受賞．IEEE，ACM，日本データベース学会各会員．

水口　弘紀（正会員）

昭和 51 年生．平成 12 年筑波大学大学院工学研究科電子・情報工学専攻修士課程修了．同年日本電気株式会社入社．現在，同サービスプラットフォーム研究所主任．テキストマイニング，データベース統合技術の研究開発に従事．日本データベース学会会員．

土田　正明（正会員）

昭和 55 年生．平成 17 年東京理科大学大学院理工学研究科経営工学専攻修士課程修了．同年日本電気株式会社入社．現在，NEC サービスプラットフォーム研究所にて情報抽出，情報検索に関する研究に従事．言語処理学会，日本データベース学会各会員．

國枝　和雄（正会員）

昭和 39 年生．平成元年京都大学大学院工学研究科情報工学専攻修士課程修了．平成 4 年同博士後期課程単位取得認定退学．同年日本電気株式会社入社．グラフィカルユーザインタフェース，航空ステレオ写真測量の研究開発に従事．現在，C&C イノベーション研究所研究部長．知識コンピューティングの研究に従事．京都大学博士（工学）．映像情報メディア学会，日本バーチャルリアリティ学会各会員．

山田　敬嗣（正会員）

昭和 33 年生．昭和 62 年京都大学大学院工学研究科情報工学専攻博士後期課程修了（工学博士）．同年 NEC C&C 情報研究所入社．平成 2 年から平成 3 年までカリフォルニア大学サンディエゴ校客員研究員．現在，NEC C&C イノベーション研究所研究所長．パターン認識，人工神経回路網，機械学習，ユビキタス情報システムの研究に従事．電子情報通信学会，IEEE 等各会員．当学会坂井記念特別賞，学術奨励賞等を受賞．