第 3 章

意味の数学モデルによる高速連想検索アルゴリズム

3.1 意味的連想検索の高速化アルゴリズム

意味の数学モデルにおける意味的連想検索とは，文脈を表す単語列 (文脈語群と呼ぶ) によって選ばれた意味空間の中で，ある単語 (キーワードと呼ぶ) に最も近い意味の単語を単語群 (検索対象語群と呼ぶ) の中から選ぶことである．

ある文脈語群が与えられ，その文脈において，検索キーワードに最も意味の近い検索対象語を求めるとき，全ての検索対象語との距離を計算したのでは利用者に対し高速な応答を行うことができない．そこで，次の順序に従って距離計算を行うことにより，全ての検索対象語を対象とした距離計算を行うことなく，検索キーワードに最も意味の近い検索対象語を探すことができる．

予め，各意味素において，その要素の大きさに応じて，検索対象語がソートされているものとする．

検索キーワード p のベクトルを

$$\mathbf{x} := (x_1, x_2, \cdots, x_\nu) \ , \ \mathbf{x} \in \mathcal{I}$$

．とし，検索対象語のベクトル $\mathbf{y}_i$ を

$$\mathbf{y}_i := (y_{i1}, y_{i2}, \cdots, y_{i\nu}) \ , \ \mathbf{y}_i \in \mathcal{I}$$
$$, i = 1, 2, \cdots, m$$

とする．

(1) 文脈に応じた部分空間 $P_{\varepsilon_s}(s_\ell)\mathcal{I}$ を形成する意味素の中から，意味重心 $\mathrm{G}^+(s_l)$ に最も関連が強い意味素 $\mathbf{q}_j$ を選ぶ．

(2) 範囲変数 Δ_h を ∞ に初期化する．検索キーワードとの距離が最も近い検索対象語の候補 z を $NULL$ に初期化する．検索対象語のベクトルの集合 $\mathcal{Y}$ を全検索対象語のベクトルに初期化する．

(3) 検索対象語のベクトルの集合 $\mathcal{Y}$ から，$x_j \pm \Delta_h$ の範囲外にある検索対象語を除く．(図 1-(c), 図 1-(d) : 一番最初に，このステップが実行される時は $\Delta_h = \infty$ なので何も除かれない．)

(4) $\mathcal{Y}$ が空集合ならば，検索対象語 z を解とし，終了する．

(5) 集合 $\mathcal{Y}$ の要素から，意味素 $\mathbf{q}_j$ 上において，検索キーワードのフーリエ係数 x_j に最も近いフーリエ係数 y_{kj} $(1 \leq k \leq m)$ を持つ検索対象語 $\mathrm{y}_k \in \mathcal{Y}$ を探す (図 1-(a)). (この検索は，検索対象語を意味素 $\mathbf{q}_j$ 上における値の大きさの順にソートしていることにより，高速に実行できる．) y_k を 集合 $\mathcal{Y}$ から取り除く．

(6) 文脈に応じた部分空間上で，検索キーワードのベクトル x と検索対象語のベクトル y_k との距離 $\rho(\mathrm{x}, \mathrm{y}_k)$ を求める (図 1-(b)).

もし，範囲変数 Δ_h より距離 $\rho(\mathrm{x}, \mathrm{y}_k)$ の方が大きいならば，(4) に行く．

さもなければ，つまり，範囲変数 Δ_h より距離 $\rho(\mathrm{x}, \mathrm{y}_k)$ の方が，小さいならば，範囲変数 Δ_h を $\rho(\mathrm{x}, \mathrm{y}_k)$ とし，ベクトル y_k を検索キーワードとの距離が最も近い検索対象語の候補 z をとする．そして，(3) に行く．

(3) の処理によって，連想検索時における検索対象語の数を大幅に減らすことができる．

3.2 意味的連想検索の高速化アルゴリズムの拡張

パターン・マッチングによる連想検索では，ユーザが与えたパターンと同じ文字列を持つ情報を解とする．一方，意味の数学モデルによる意味的連想検索では，意味の近いもの解とする．そのため，ユーザーが与えた検索キーワードに意味的に最も近い単語だけでなく，2番目,3番目に近い単語も，ユーザーにとっては有用であることが多い．

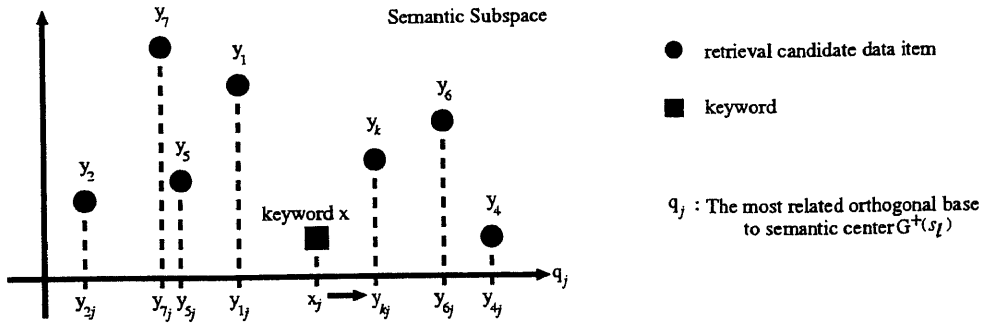

図 1-(a): 検索キーワードに最も近い検索対象語を高速に求めるアルゴリズム

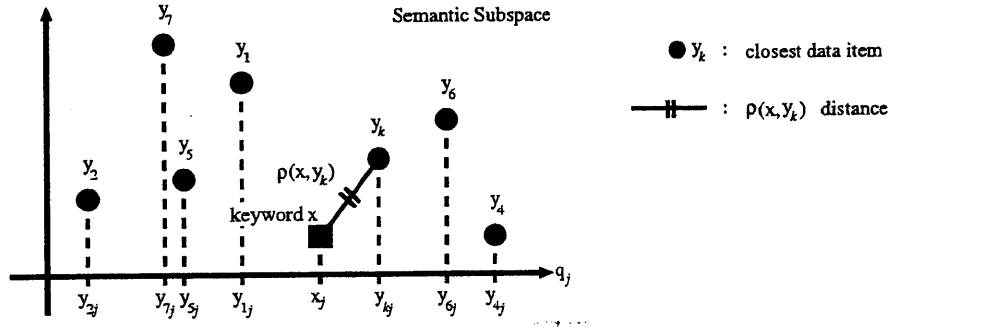

図 1-(b): 検索キーワードに最も近い検索対象語を高速に求めるアルゴリズム

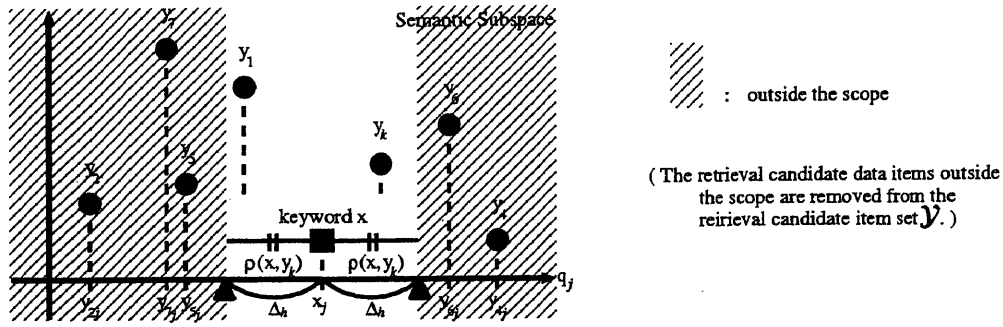

図 1-(c): 検索キーワードに最も近い検索対象語を高速に求めるアルゴリズム

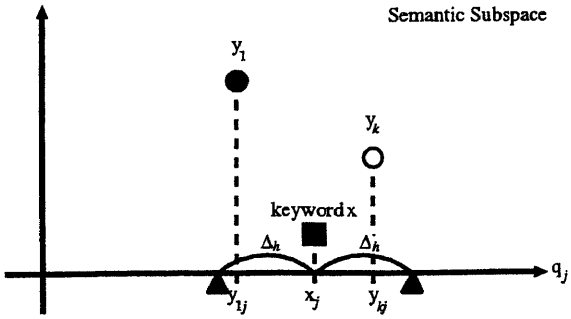

図 1-(d): 検索キーワードに最も近い検索対象語を高速に求めるアルゴリズム

検索キーワードに最も近い単語を高速に求めるアルゴリズムは，解を求めるのに必要のない距離計算を省くので，高速な検索が可能である．しかし，最も近い単語を求める過程で，本来2番目に近い単語の距離計算を省く場合があり得るので，検索キーワードからの距離が最も近い単語の次の候補であった単語が，検索キーワードから2番目に近いとは限らなくなる．

検索キーワードに2番目に近い単語を求める方法はとしては，検索キーワードとの距離が最も小さい検索対象語を検索対象から除き，先のアルゴリズムを繰り返す方法が考えられる．しかし，この方法では，同一の検索対象語に対して，同一の距離計算を何度も行う可能性があるので，検索キーワードと全検索対象語との距離を素直に計算する方法よりも，計算回数が多くなってしまう可能性が考えられる．

そこで，検索キーワードに最も近い検索対象語を高速に求めるアルゴリズムを拡張し，検索キーワードからの距離が次に近い単語を求めるアルゴリズムを提示する．

まず，高速な連想検索のアルゴリズムによって，検索キーワードから最も近い単語の検索を求めた直後の状態を考える．図 3.2 は，$\mathrm{y}_1, \mathrm{y}_2, \mathrm{y}_3$ の順に距離計算を行い，検索キーワードのベクトル x に最も近い検索対象語 y_3 を求めた直後の状態である．

(1) 検索キーワードのベクトル x に2番目に近い単語を求めるにあたって，先ほど求めた解である y_3 を候補から外す．

(2) 現在の時点で距離が分かっている検索対象語のベクトル $\mathrm{y}_1, \mathrm{y}_2$ だけで考えると，検索キーワードのベクトル x からの距離が最も小さいベクトルは y_2 であることがわかる．これにより，図 3.2 中の斜線内の領域には，検索キーワードのベクトル x からの距離が y_2 より小さい検索対象語は存在しないことが分かる．

(3) 検索キーワードのベクトル x と検索対象語のベクトル y_2 との距離 $\rho(\mathrm{x}, \mathrm{y}_2)$ を考える (この距離は，既に求まっている)．範囲 $x_j \pm \rho(\mathrm{x}, \mathrm{y}_2)$ の外にある検索対象語は，必ず y_2 より離れた位置にあるので，この領域にも，検索キーワードのベクトル x からの距離が y_2 より小さい検索対象語は存在しないことが分かる (図 3.3)．

したがって，残った領域 (斜線領域に挟まれた領域) において，y_2 を検索キーワードからの距離が最も近い検索対象語の候補として，検索キーワードからの距離が最も近い検索対象語を高速に求めるアルゴリズムを適用すれば，検索キーワードから 2 番目に近い検索対象語を求めることが可能である．この例では，解は y_4 である．

(3) の特別な場合として，他に 2 番目に近い検索対象語の候補が存在する可能性のある領域（斜線領域に挟まれた領域）が無い場合がある（ 図 3.4 ）．この場合は，距離計算を全く行わなくとも，現在の候補が解であることが分かる．

また，2 番目に近い検索対象語の候補が，そもそも存在しない場合も考えられる（ 図 3.5 ）．この場合は，まだ距離計算を行っていない領域において，検索キーワードに最も近い検索対象語を高速に求めるアルゴリズムを適用する．

任意の解を求めた直後の状態は，この 3 種類通りのいずれかである．また，本アルゴリズムを適用した直後に，再び，本アルゴリズムを適用すると，任意の個数の解を，検索キーワードからの距離が近い順に求めることが可能である．

また，次のアルゴリズムは，検索キーワードに最も近い検索対象語を求めるアルゴリズムを拡張し，任意の個数の解を高速に求めるものである．

予め，各意味素において，その要素の大きさに応じて，検索対象語がソートされているものとする．

検索キーワード p のベクトルを

$$\mathbf{x} := (x_1, x_2, \cdots, x_\nu) \ , \ \mathbf{x} \in \mathcal{I}$$

．とし，検索対象語のベクトル $\mathbf{y}_i$ を

$$\mathbf{y}_i := (y_{i1}, y_{i2}, \cdots, y_{i\nu}) \ , \ \mathbf{y}_i \in \mathcal{I}$$
$$, i = 1, 2, \cdots, m$$

とする．

(1) 文脈に応じた部分空間 $P_{e_s}(s_l)\mathcal{I}$ を形成する意味素の中から，意味重心 $\mathbf{G}^+(s_l)$ に最も関連が強い意味素 $\mathbf{q}_j$ を選ぶ．

(2) 検索キーワードからの距離の小さい順に並べられた解の候補リスト $\mathcal{Z}$ を $NULL$ に初期化する．検索対象語のベクトルの集合 $\mathcal{Y}$ を，全検索対象語に初期化する．

(3) 解の候補リスト $\mathcal{Z}$ の先頭要素を，解の候補 $\mathbf{z}$ とする．

(4) 解の候補 $\mathbf{z}$ が $NULL$ ならば，範囲変数 Δ_h を ∞ に初期化する．解の候補 $\mathbf{z}$ が $NULL$ でなければ，範囲変数 Δ_h を，検索キーワードと解の候補 $\mathbf{z}$ との距離 $\rho(\mathbf{x}, \mathbf{y}_z)$ とする．

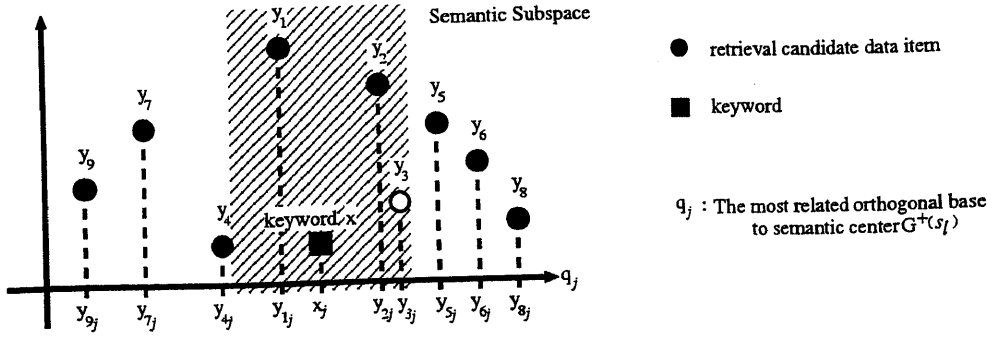

図 3.2: キーワードに 2 番目に近い単語を探す例 -1 : 検索終了直後の状態

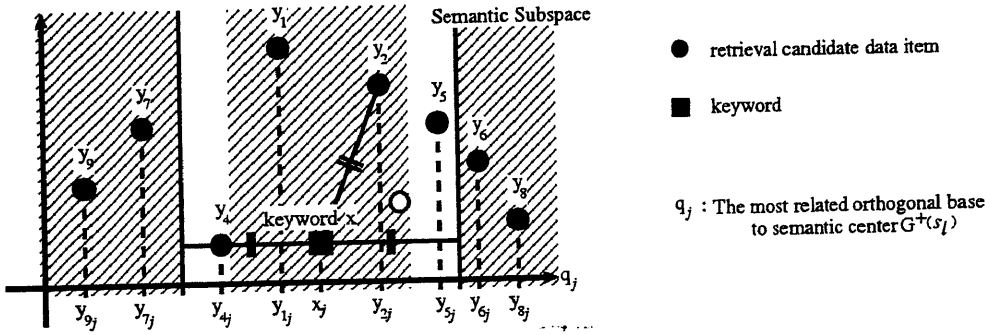

図 3.3: キーワードに 2 番目に近い単語を探す例 -1 : 続き

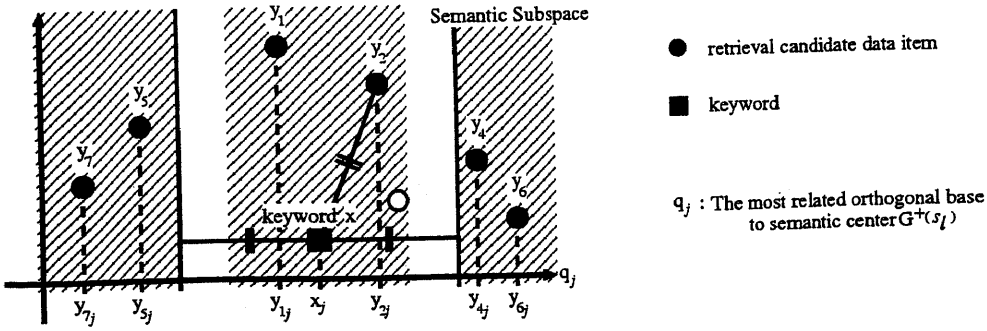

図 3.4: キーワードに 2 番目に近い単語を探す例 -2

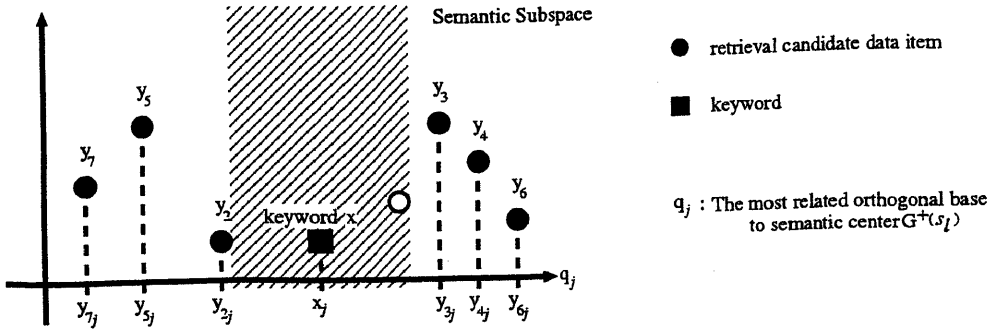

図 3.5: キーワードに 2 番目に近い単語を探す例 -3

(5) 集合 $\mathcal{Y}$ の要素から，$x_j \pm \Delta_h$ の範囲内にあり，かつ，意味素 $\mathbf{q}_j$ 上において，検索キーワードのフーリエ係数 x_j に最も近いフーリエ係数 y_{kj} $(1 \leq k \leq m)$ を持つ検索対象語のベクトル $\mathbf{y}_k \in \mathcal{Y}$ を，を探す．（この検索は，検索対象語を意味素 $\mathbf{q}_j$ 上における値の大きさの順にソートしていることにより，高速に実行できる．）$\mathbf{y}_k$ を 集合 $\mathcal{Y}$ から取り除く．

(6) もし，$\mathbf{y}_k$ が存在しなければ，解の候補リスト $\mathcal{Z}$ から $\mathbf{z}$ を取り除き，$\mathbf{z}$ を解とする．解の個数が指定数未満ならば，(3) に行く．指定個数の解が得られたならば，終了する．

(7) 文脈に応じた部分空間上で，検索キーワードのベクトル $\mathbf{x}$ と 検索対象語のベクトル $\mathbf{y}_k$ との距離 $\rho(\mathbf{x}, \mathbf{y}_k)$ を求める．解の候補リスト $\mathcal{Z}$ に，(求めた距離 $\rho(\mathbf{x}, \mathbf{y}_k)$ を含めて) $\mathbf{y}_k$ を追加する．

距離 $\rho(\mathbf{x}, \mathbf{y}_k)$ が，範囲変数 Δ_h より大きければ，(5) に行く．

距離 $\rho(\mathbf{x}, \mathbf{y}_k)$ が，範囲変数 Δ_h より小さければ，ベクトル $\mathbf{y}_k$ を解の候補 $\mathbf{z}$ とする．そして，(4) に行く．

本アルゴリズムを適用することにより，検索キーワードと全ての検索対象語との距離を計算することなく，検索キーワードからの距離が近い順に，指定個数の解を得ることが可能となる．

3.3 英英辞典を対象とした実験

高速な意味的連想検索アルゴリズムの有効性の検証のために，全ての検索パターンを試行し，検索時の距離計算回数の測定を行うことが考えられる．しかし，意味の数学モデルによる連想検索では，キーワードと文脈の組合せにより，無限に近いパターンの検索を行うことが可能である．したがって，高速な意味的連想検索アルゴリズムの有効性を検証するにあたって，全ての検索パターンを試行するのは不可能である．そこで，Longman Dictionary of Contemporary English において基本語とされている英単語 2328 語を対象として，それぞれの単語が 1 回ずつ解となることを想定した実験を行った．

3.3.1 実験環境

実験では，Longman Dictionary of Contemporary English において基本語とされている英単語 2328 語を，The General Basic English Dictionary の定義を用いて定義し，2328×874 の行列を作成し，イメージ空間を構成した．検索対象単語群として，上述の基本英単語 2328 語を用いた．検索キーワードとなる単語群は，上の行列において，同じ見出し語を持つ単語群の定義を合成した行列を使用した．

実験に使用した計算機は Sun4/ELC．OS は SunOS 4.1.4 である．

3.3.2 実験方法

実験を次の方法によって行った．

(1) 検索対象語群から，1 単語を選び出し，それを $\mathbf{d}_i$ とする．

(2) $\mathbf{d}_i$ の定義に使用されている語句を，文脈語群とする．

(3) $\mathbf{d}_i$ と同じ綴の単語を，検索キーワードとする．

(4) 検索対象語群の中から，与えた文脈において，検索キーワードに近い順に，10 単語検索する．

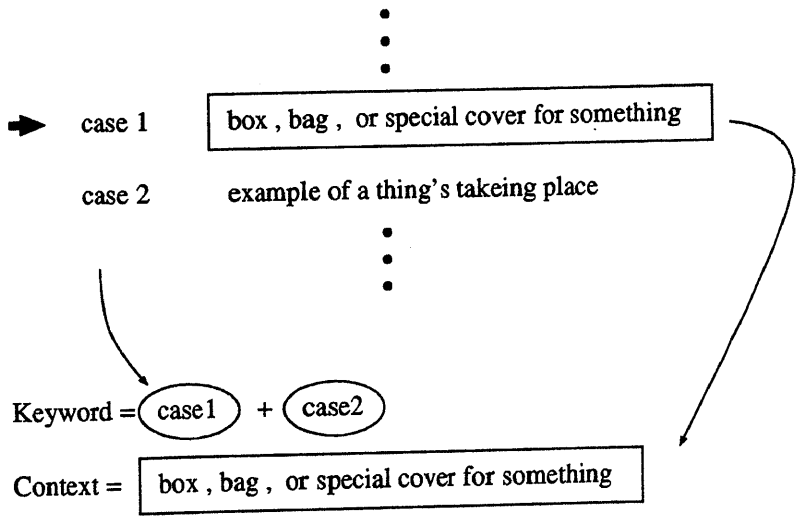

図 3.6: 英英辞典を対象とした実験の例

図 3.6 は，本実験に使用するパラメータの一部を示したものである．この例において，case には「箱」の意味の case1 と，「場合」の意味の case2 が存在する．こ

の図では，case1 を解と想定している．この時，検索キーワードには，「箱」の意味の case1 と，「場合」の意味の case2 のベクトルを合成したベクトルを与える．また，文脈語群には case1 に使用されている単語群を与える．検索対象語 2328 単語の中から，検索キーワードに近い順に 10 単語を検索し，その時に行った距離計算を測定する．

2328 語の各々について，意味的連想検索を行い，その処理に必要な平均距離計算回数を求めた．また，意味空間のしきい値 ε_s を，0.0 から 0.9 まで 0.1 きざみで変化させ，それによる影響を調べた．

3.3.3 実験結果

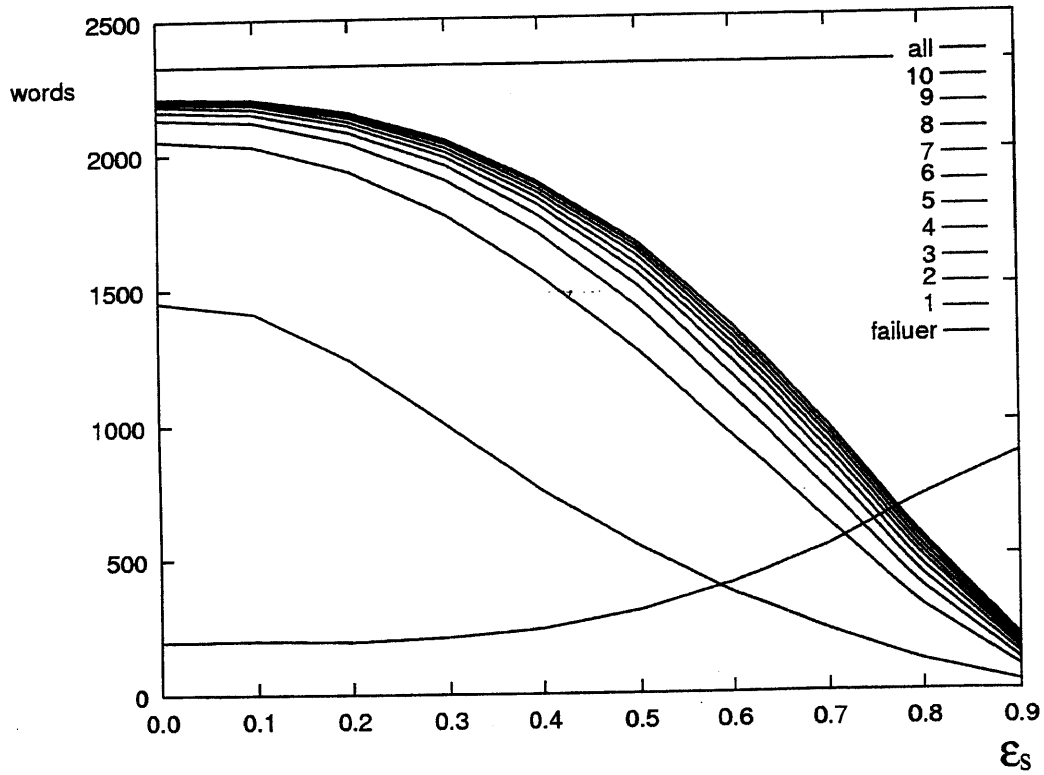

図 3.7: 英英辞典を対象とした実験の結果

実験結果を図 3.7 に示す．最上位のの直線は，キーワードと全検索対象語との距離を計算した時の計算回数である．上に凸になっているグラフの中で，一番下のグラフは，検索キーワードに最も近い検索対象語だけを求めた時の，平均計算回数である．その 1 つ上のグラフは，拡張された高速化のアルゴリズムを使用し，検索キーワードから 2 番目に近い検索対象語までを求めた場合の，平均計算回数である．検索キーワードに最も近い検索対象語だけを求めた時の計算回数と比較すると，多くの計算が必要であっ

たことがわかるが，キーワードと全検索対象語との距離を計算した時よりも，高速であることがわかる．さらに，その 1 つ上のグラフは，検索キーワードから 3 番目に近い検索対象語までを求めた時の平均計算回数である．検索キーワードから 2 番目に近い検索対象語までを求めた時の計算回数と比較すると，より多くの計算が必要であったことがわかる．しかし，キーワードと全検索対象語との距離を計算する場合よりも，距離計算回数は少ない．

また，意味空間のしきい値 ε_s による，高速な意味的連想検索アルゴリズムへの影響については，ε_s の増加に伴い，アルゴリズムの有効性が，より強く現れる．これは，意味空間のしきい値 ε_s の増加に伴い，距離計算を行う空間の次元数が減少していることによると思われる．

しかし，ε_s の増加に伴い，想定した解と異なる解が選択される回数（下に凸のグラフ）も増加を始めるので，ε_s に関しては，0.5 以下程度に設定するのが望ましいと思われる．