Multi-level prefix-filter を用いた高速重複文書照合

Fast Duplicated Documents Detection with Multi-level Prefix-filter

立石 健二♥ 久寿居 大♠
Kenji TATEISHI Dai KUSUI

重複文書照合は，大量の文書集合から類似度閾値以上の文書ペアを漏れなく高速に検出する問題であり，顧客 DB の名寄せや，コンタクトセンターの障害事例データの傾向分析等に適用できる．近年提案された重複文書照合方式に prefix-filter がある．これは，文書ペアで共通しない語の数に基づいて類似度閾値以上にならないことが確実な文書ペアを類似度計算前に検出し，filtering する手法である．しかしこの手法は，類似度閾値未満の文書ペアも filtering されずに多く残ってしまうため，類似度計算の回数が十分に削減できない問題があった．そこで，本稿では，異なる複数の prefix-filter を組み合わせることにより，prefix-filter の利点(厳密解，パラメータ調整不要)を維持しつつ類似度計算の回数を削減する multi-level prefix-filter を提案する．

Duplicated document detection is a problem of finding all document-pairs rapidly whose similarities are equal to or more than a given threshold, and can be used for data cleaning of customer databases or trend analysis of failure case databases on contact center. There is a method proposed recently called prefix-filter, which finds document-pairs whose similarities never reach the threshold based on the number of uncommon words/characters in a document-pair, and removes them before similarity calculation. However, prefix-filter cannot decrease the number of similarity calculations sufficiently because it leaves many document-pairs whose similarities are less than the threshold. In this paper, we propose multi-level prefix-filter, which reduces the number of similarity calculations more efficiently as maintains the advantage of prefix-filter (exact solution, no extra parameter) by applying multiple different prefix-filters.

1. はじめに

重複文書照合は，大量の文書集合から類似度閾値以上の文書ペアを漏れなく高速に検出する問題である．重複文書照合は，例えば，異なる人/場所/方法によって管理された顧客DB名寄せ(クリーニング)[1,2,3,4,5,6,7]やコンタクトセンターの障害事例の傾向分析[8]に活用されており，さらに，掲示板やblog文書の重複を検出することにより，スパムフィルタリングにも適用可能である．重複文書照合システムは，利用者から照合対象の文書集合と類似度閾値を受け取ると，類似度閾値以上となるすべての文書ペア，もしくはそれらをグループにまとめたものを提示する．さらに名寄せのようなデータクリーニングを目的とする場合，利用者は，システムが出力した重複文書のそれぞれが真に重複であるかを最終的に確認する．

この問題を最もnaiveに実装すると，全ての異なる文書ペアで類似度計算を行うことになるが，文書数の増加につれ組み合わせ爆発が起こり，膨大な時間がかかってしまう．従来の重複文書方式は，処理の軽い照合により文書ペアの候補を荒く絞りこんでから，それらに対して処理の重い類似度計算を行っていた[1,2,3,4,5,6,7]. その中で, prefix-filter[6,7]は，文書ペアで共通しない語の数に基づいて類似度閾値以上にならないことが確実な文書を類似度計算前にfilteringする手法である．例えば，全ての文書が10語から構成されており，類似度が80%とは10語のうち8語が共通することを意味するとする．この時，文書ペアの類似度が80%以上ならば，一方の文書から3語を選択すればそのうち必ず1語は他方の文書に含まれなければならない．したがって，類似度閾値が80%のとき，文書から選択した3語を一つも含まない文書ペアは，類似度が80%にならないことが確実なのでfilteringできる．prefix-filterは，その他の方式[1,2,3,4,5]と比較して，荒い絞込みのための新たなパラメータ調整が必要ない点，照合もれを発生しない厳密解を得られる点で優れている．

しかし，prefix-filterは類似度閾値未満になる文書ペアもfilteringされずに多く残ってしまうため，実際には，類似度計算の回数が十分に削減できない問題があった. prefix-filterは，各文書から選択する語の組み合わせ(上記の例では，3語をどのように選ぶか)によってfilteringできる文書ペアが変わるため，極端な場合，1語しか共通しない文書ペアもfilteringされずに残る可能性がある．重複文書照合では，類似度計算の回数と各々の類似度計算にかかる時間の積を照合時間と近似できる．文書ペアの同一性を正確に判定するためには，表記のゆれや同義語展開を考慮した処理の重い類似度計算が必要となる．そのため，できる限り類似度計算の回数を削減できることが望ましい．

そこで本稿では，prefix-filterの特徴(完全解，パラメータ調整不要)を維持しながら，類似度計算の回数を削減するmulti-level prefix-filterを提案する．提案方式は，異なるprefix-filterを複数回適用して文書ペアを絞り込む．20万件の顧客DBを対象として，類似度関数として編集距離を用いて評価実験を行ったところ，multi-level prefix-filterは従来型prefix-filterよりも照合回数を最大で約1/4に削減できることがわかった．

2. prefix-filter を用いた重複文書照合

prefix-filter は，類似度閾値以上にならないことが確実な文書ペアを filtering して，類似度計算の回数を削減する方法である．下記の 4 つのステップで処理される．

(1) 類似度が ST(0≦ST≦1)以上となる文書ペアで共通する語の割合の最小値 x(0≦x ≦1)を定める．
(2) 照合対象の文書群で語の優先順位を決定する．
(3) 各文書から 1-x の割合を超える最小の語群を，(2)で決定した優先順位にしたがって選択する．
(4) (3)で選択した語群が一つも共通しない文書ペアを filtering して,残りの文書ペアに対し類似度計算を行う．

♥ 会員 NEC インターネットシステム研究所
k-tateishi@bq.jp.nec.com
♠ 非会員 NEC インターネットシステム研究所
kusui@ct.jp.nec.com

例えば，類似度関数として編集距離による類似度[1]を用い，類似度閾値は ST=0.6 として，図 1 の 6 文書を重複照合する．まず，編集距離の場合，類似度が ST=0.6 以上となる文書ペアで共通する語の割合の最小値は x=0.6 となる．これは，共通する語の割合が 0.6 未満の文書ペアは類似度が決して 0.6 以上にならないことを意味する．x は類似度関数から求めることが可能であり，導出方法は，[7]に詳しい．類似度関数としては，編集距離の他，cosine 類似度，Jaccard 係数等の x を求めることが可能である．なお，語とは文字又は単語を表すが，編集距離に基づく類似度関数の場合は文字を表す．

次に，照合対象の文書群で語の優先順位を決定する．図 1 の(a)では，6 文書で出現する全ての語に関して，出現文書数が少ない文字に高い優先順位を与えている(出現文書数が同じときはアルファベット順とした)．どのような方法で優先順位を決めても，filtering する文書ペアは必ず類似度閾値未満となるが，filtering できる文書群は異なる．経験的には出現文書数が少ないものから順番に選ぶと，filtering の効果が大きいことが知られている[7]．出現文書数が少ない語を選択したほうが，文書ペアで語が共通する可能性が低くなり，filtering の効果が高くなるからである．

次に，各文書から 1-x=0.4 の割合を超える最小の語群を図 1 の(a)の優先順位で選択する．例えば，文書 1 の 9 語から 4 語を選ぶと割合が 0.4 を超える．4 語を(a)の優先順位で選択すると，{A,B,C,I}となる．図 1 の(b)では，各々の文書から選択した単語を太字で背景色を付けて表示している．

最後に，各文書から選択した語群が一つも共通しない文書ペアを filtering し，残りの文書ペアに関して類似度計算を行う．選択した語群が一つも共通しない文書ペアは，共通する語の割合が 0.6 未満となるため，類似度は決して 0.6 以上にはならない．filtering は索引ファイルを利用すれば高速に実装できる．索引ファイルには，各文書から選択した語とその文書 ID の対応関係を登録する．例えば，文書 1 の類似度計算の対象となる文書群を探す場合は，文書 1 から選択された{A,B,C,I}のいずれかが選択された文書群を索引ファイルから検索する．その結果，文書 1 に対しては文書 3 と文書 5 が類似度計算の対象となる．同様に，文書 2 については文書 4，文書 3 については文書 5，文書 4 については文書 6 となる．最終的な類似度計算の回数は，6 文書の総当りでは(6*5)/2=15 回の計算が必要なところが，(c)のように 5 回に削減できる．

3. multi-level prefix-filter

prefix-filter の問題は，類似度閾値未満になる文書ペアも filtering されずに多く残ってしまうため，類似度計算の回数が十分に削減できないことである．prefix-filter は，各文書から選択する語の組み合わせによって filtering できる文書ペアが変わるため，極端な場合，1 語しか共通しない文書ペアも残る可能性がある．例えば図 1 では，(a)の優先順位に基いて語を選択すると，文書 4 と文書 6 は共通する語が K のみであるのに filtering されずに残ってしまう．文書ペアの同一

(a) A>G>I>K>L>M>B>C>D>F>E>H>J

(b)

文書1	A	B	C	D	E	F		H	I	J		
文書2					E		G	H		J		L
文書3		B	C	D	E	F		H	I			
文書4					E		G	H		J	K	L
文書5	A	B	C	D		F				J		M
文書6											K	M

(c)
edit_sim(文書1, 文書3) = 1-(2/9) = 0.78
edit_sim(文書1, 文書5) = 1-(4/9) = 0.45
edit_sim(文書2, 文書4) = 1-(1/6) = 0.83
edit_sim(文書3, 文書5) = 1-(4/7) = 0.43
edit_sim(文書4, 文書6) = 1-(5/6) = 0.17

図 1 Prefix-filter
Fig 1 Prefix-filter

性を正確に判定するためには，表記のゆれを考慮した編集距離や類似度計算の際に同義語展開を伴うといった処理の重い類似度関数が必要となってくる．そのため，類似度計算の回数はできる限り削減できることが望ましい．

そこで改良手法として異なる prefix-filter を複数回適用して，類似度計算の対象となる文書ペアを絞り込む multi-level prefix-filter を提案する．各々の prefix-filter は，異なる基準で定めた優先順位に従い文書から語を選択する．そのため，各々の prefix-filter で filtering できる文書ペアは異なる．そして，全ての prefix-filter を通して残った文書ペアのみを類似度計算する．これにより，従来の prefix-filter よりも多くの文書ペアを照合漏れを発生せずに filtering できる．図 2 に，prefix-filter を 2 回適用した multi-level prefix-filter の例を示す．1 回目と 2 回目で，語の優先順位を変更すると，各文書から選択される語が変わり，さらに filtering される文書ペアも変わることがわかる．1 回目と 2 回目の prefix-filter で残った文書ペアの積集合を求めると，類似度計算の回数を 3 回に削減できることがわかる．

各々の prefix-filter の語の優先順位は次のよう定める．n 回目の prefix-filter に関する語 w の優先度のスコア Score(n,w)を下記の式で求め，スコアが小さい方が優先順位が高いとする．

$$Score(n,w)=\begin{cases} df(w) & n=1 \\ 0.1*df(w)+\sum_{i=1}^{n-1} sdf(i,w) & n\geq 2 \end{cases}$$

ここで，df(w)は w の出現文書数を，sdf(i,w)は i 回目の prefix-filter で w が選択された文書数を表す．基本的な方針は，出現文書数が少ない語に高い優先順位を与えることである．前述したように，出現文書数が少ない語を選択したほうが，文書ペアで語が共通する可能性が低くなり，filtering の効果が高くなるからである．一方，multi-level prefix-filter では各々の prefix-filter で異なる文書が filtering できるほど，類似度計算回数の削減効果が高いと考えられる．そこで，各々の prefix-filter でなるべく異なる語が選択できるように，2 回目以降の prefix-filter では，出現文書数が低く(第 1 項)，かつ，それまでに選択されなかった語(第 2 項)に高い優先順位を与えるようにした．

図 3 に multi-level prefix-filter のアルゴリズムを示す．提案方式も 2 節と同様に索引ファイルを用いて実装できる．索

[1] 文書 d1，文書 d2 の編集距離による類似度 edit_sim(d1,d2)は下記の式で表せる。

$$\text{edit\_sim}(d1,d2)=1-\frac{\text{edit\_distance}(d1,d2)}{\max(|d1|,|d2|)}$$

ここで、|d1|、|d2|は d1、d2 に含まれる語(文字)数、edit_distance(d1,d2)は、d1 から d2 に変換するのに必要な最小の編集(挿入/削除/置換)回数である。例えば、図 1 で edit_distance(文書 1,文書 5)は、文書 1 を文書 5 に変換するためには、E と H と I を削除し、M を挿入すればよいから 4 となる。

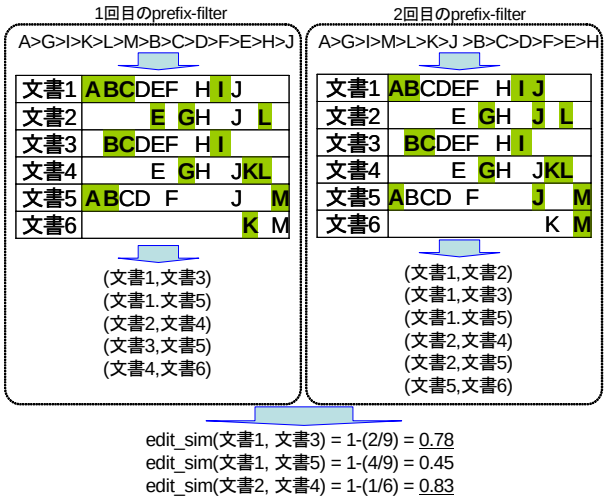


図 2 Multi-level prefix-filter
Fig 2 Multi-level prefix-filter

引ファイル作成処理で prefix-filter の適用回数分の個数の索引ファイル D_INDEX，W_INDEX を作成し，それらを用いて filtering 処理で類似度計算対象となる文書を絞り込む．複数のプロセッサを搭載したマシンを用いる場合は，filtering 処理を並列化することも可能である．

提案方式は，Score(n,w)に基づいて n 回目の prefix-filter の語 w の優先順位を照合対象の文書集合に対して一意に定めている．一方，Score(n,w,d)を定義して，n 回目の prefix-filter の語 w の優先順位を文書 d 毎に変え，n-1 回目までの prefix-filter で filtering した文書群の分布に基づいて n 回目の prefix-filter の語 w の優先順位を決定する方法もある．しかしながら，この方法は，より高い filtering 効果を得られる可能性があるが，n 回目の prefix-filter のために，n-1 回目の prefix-filter の結果を待つ必要があり並列化に適していないこと，個々の Score の計算処理が重くなる上に Score 計算回数が多くなることから採用しなかった．

4. 評価実験

4.1 実験方法

multi-level prefix-filter の類似度計算回数の削減効果を，従来型 prefix-filter と比較する．評価に用いたデータは，名寄せを目的として実際の実業務で使用された 20 万件の顧客 DB である．それぞれのデータは組織名と住所の 2 つのフィールドを持つ．各フィールドの語数(文字数)の平均は組織名で 11 文字，住所で 18 文字である．また，類似度 80%以上を重複とみなした場合に一つ以上の文書と重複となる文書数は，組織名に関しては 86031 件で 43%，住所に関しては 120368 件で 60%と比較的重複の多いデータである．使用したマシンの OS は Windows2000，CPU は Pentium Xeon 2.6GHz，Memory は 3.4GByte を搭載している．顧客 DB の組織名と住所は表記のゆれを多く含むため類似度関数は編集距離による類似度を用いた．類似度閾値は 80%とした．

4.2 実験結果

図 4 の(a)に，prefix-filter の適用回数を変化させたときの，類似度計算の回数の変化を示す．prefix-filter の適用回数 n=1 回が従来型の prefix-filter を意味する．類似度の計算回数は，組織名と住所のどちらを照合対象とした場合でも，適用回数が 2 回のときに最も削減の割合が大きく，適用回数が 10 回でほぼ収束する．適用回数が 1 回と 10 回を比較すると，

```
Nはprefix-filterの適用回数、 Dは照合対象の文書集合を表す

索引ファイル作成処理:
- すべての語wに対してScore(1回目,語w)を計算
for(i=1; i≦N; i++) {
    for(d∈D) {
        - Score(i回目,語w)に従いdから1-xの割合を超える最小の語群Wを選択
        - 文書dをキーとするD_INDEX( i回目,文書d)にWを追加
        - それぞれのw∈Wをキーとするw_INDEX( i回目,語w)にdを追加
    }
    -すべての語wに対して Score(i+1回目,語w)を計算
}

照合処理:
for(d∈D) {
  filtering処理:
    for(i=1; i≦N; i++) {
        - W_INDEX( i回目,文書d)を参照してdで選択された語群Wを検索
        - D_INDEX( i回目,語w)を参照してw∈Wを含む文書集合のうち、
          文書番号がdより大きい文書集合DSSwを検索
        - {DSSw | w∈W}の和集合の文書群DSSiを求める
    }
    - {DSSi | 1≦i≦N}の積集合の文書群DSを求める

    類似度計算処理:
    for(ds∈DS) {
        - sim(文書d,文書ds)を計算、類似度閾値以上の場合は出力
    }
}
```

図 3 multi-level prefix-filter のアルゴリズム
Fig 3 Algorithm of multi-level prefix-filter

multi-level prefix-filter は従来型 prefix-filter と比較して，組織名で約 1/4(削減率 77%)，住所で約 1/3(削減率 69%)に削減できた．

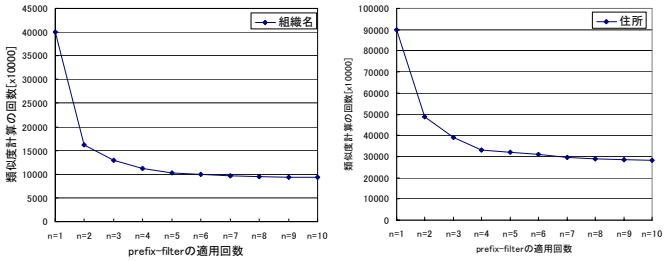
図 4 の(b)に，prexfix-filter の適用回数を変化させたときの，照合時間の変化を示す．照合時間は，prefix-filter の適用時間と類似度計算の時間の和となる．prefix-filter の適用回数が増えるほど，照合回数が削減できるため後者の時間を短縮できるが，前者の時間が増加する．なお，本実験では filtering 処理の並列化は行っていない．組織名，住所共に prefix-filter の適用回数が 4 回のときが最も照合時間の短縮率が大きく，組織名では約 57%，住所では約 51%の短縮を実現できた．

図 4 の(c)に，prefix-filter の適用回数を 4 回，照合対象文書を変化させた場合の類似度計算回数と照合時間の削減率を示す．削減率は，従来技術である適用回数が 1 回の prefix-filter と，提案方式である適用回数が 4 回の multi-level prefix-filter を比較した．この結果から，文書数が増えても提案方式の効果は変わらないことがわかる．

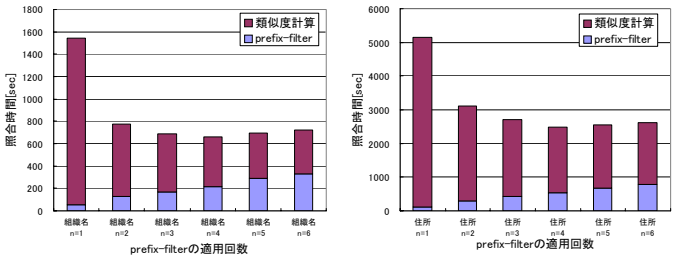
4.3 考察

以上の実験より，顧客 DB において提案方式である multi-level prefix-filter は，類似度計算回数を最大 1/4 に削減でき，その効果は DB の規模に依存しないことがわかった．また，組織名のほうが住所の方が効果が大きいことが分かった．前述したように，住所の方が組織名よりも平均語(文字)数が長く，含まれる重複の割合も大きい．したがって，提案手法は，例えば顧客 DB のような含まれる文字数が比較的少ない DB で，完全一致や空白の有無のような機械的に検出できる重複は削除するといった重複の割合を少なくした DB に対して，残りの重複を編集距離や同義語辞書に基づく処理の重い類似度関数を用いて検出する場合に特に効果が高いと予想する．

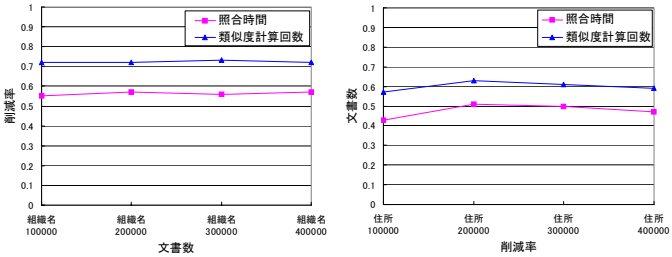
提案方式は，prefix-filter を複数回適用して，すべての prefix-filter を通して残った文書ペアのみを類似度計算の対象とする．したがって，どのような文書集合に対しても，従来方式と比較して少なくとも類似度計算回数が増えることはない．一方，照合回数の削減の程度は，文書集合の特徴に



(a) 類似度計算の回数



(b) 照合時間



(c) 文書数による変化

図 4 実験結果
Fig 4 Experimental Result

依存する．例えば，前述した文書集合に含まれる語の数や，文書集合に含まれる重複の数である．これらが照合回数の削減にどの程度影響するかの調査は今後の課題としたい．

5. 関連研究

重複文書照合は，文書ペアを処理の軽い方法で大まかにし絞りこむ filtering 処理(ブロッキング処理)と，残った文書ペアを編集距離等の類似度関数を用いて詳細な絞りこみを行う類似度計算処理の 2 つから構成される[10,11]．従来の filtering 処理は，類似度計算処理の類似度関数とは独立した方法を用いていた[1,2,3,4,5]．例えば，[1]は，filtering 処理で Standard blocking と呼ばれるデータの先頭から n 文字が一致するレコードのブロックを大量につくり，類似度計算処理同じブロック内のレコード間で類似度を計算する．

しかし，これらの方法は，n 文字の長さ等のブロック作成のためのパラメータをユーザが試行錯誤する作業が必要であった．また，改良手法は提案されているものの，それでも本来は重複となるべき文書ペアが filtering 処理で別ブロックに分かれてしまい，照合漏れが残る問題があった．さらに，このような照合漏れが実際にどのぐらいの規模になるのか，理論上予測できず，照合対象の文書集合に依存する問題があった．prefix-filter[6,7]による filtering 処理は，これらの問題を解決した方式であり，新たなパラメータ調整が必要ない点，照合もれを発生しない厳密解を得られる点で優れている．提案手法である multi-level prefix-filter は，prefix-filter の改良手法であり，prefix-filter の利点を失わずに，prefix-filter より高い filtering 効果を持つことを前節までに述べた．

6. おわりに

本稿では，重複文書照合において，異なる優先順位で語を選択する複数の prefix-filter を用いて類似度計算の対象となる文書集合を絞り込む multi-level prefix-filter を提案した．20 万件の企業名の顧客 DB を編集距離を用いて重複照合したところ，multi-level prefix-filter は従来型 prefix-filter よりも類似度計算回数を最大で約 1/4 にできることがわかった．

[文献]

[1] Jaro, M. A.: Advances in Record Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida, *Journal of the American Statistical Society, 84 (406)*, pp.414–420 (1989).
[2] Hernandez, M. A. and Stolfo, S. J.: The Merge/Purge Problem for Large Databases,*Proceedings of ACM SIGMOD*, pp.127–138 (1995).
[3] 相澤彰子，大山敬三，高須淳宏，安達淳: 複数書誌データベース統合における重複エントリーの高速検出法，情報処理学会研究報告 2004-FI-75, pp.111-118(2004).
[4] McCallum, A., Nigam, K. and Ungar, L.H.: Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching, *Proceedings of ACM SIGKDD*, pp.169–178 (2000).
[5] 立石健二，久寿居大：重複レコード照合における分割統合照合方式の提案と有効性評価，情報科学技術レターズ,LD–002，pp.49–50 (2006).
[6] Sarawagi, S. and Kirpal, A.: Efficient set joins on similarity predicates, *Proceedings of the 2004 ACM SIGMOD*, pp.743–754 (2004).
[7] Chaudhuri, S., Ganti, V. and Kaushik, R.: A Primitive Operator for Similarity Joins in Data Cleaning, *Proceedings of ICDE*, pp.5-16 (2006).
[8] 難波巧，柳瀬隆史：擬似事例頻度を利用した質問回答検索，電子情報通信学会技術研究報告言語理解とコミュニケーション研究会，pp.101–108(2002).
[9] Gravano, L., Ipeirotis, P., Jagadish, H., Koudas, N., Muthukrishnan, S. and Srivastava, D.: Approximate string joins in a database (almost) for free, *Proceedings of VLDB* (2001).
[10] 相澤彰子，高須淳宏，大山敬三，　安達淳：異種データベース間でのレコード照合に関する研究動向，*NII Journal No.8*, pp.43–51 (2004).
[11] Elmagarmid, A., Ipeirotis, P. and Verykios, V.: Duplicate Record Detection (2006).

立石 健二　Kenji TATEISHI
1999 年九州大学大学院システム情報科学研究科知能システム学専攻修士課程修了．同年日本電気(株)入社．現在，NEC インターネットシステム研究所主任．情報抽出，情報検索に関する研究に従事．情報処理学会，日本データベース学会，各会員．

久寿居 大　Dai KUSUI
1992 年京都大学大学院工学研究科修士課程修了. 現在, NEC インターネットシステム研究所主任研究員．情報分析・知識管理システムの研究・開発に従事．情報処理学会会員．