社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－ＮＬ－185 (4)
2008－SLP－ 71 (4)
2008／5／22

文書要約の最大充足化問題によるモデル化

高村 大也　　奥村 学
東京工業大学 精密工学研究所
〒226-8503 横浜市緑区長津田町 4259
{takamura,oku}@pi.titech.ac.jp

文書要約を最大被覆問題として解く．組み合わせ最適化の視点から，適切なアルゴリズムをこの問題に適用し，包括的な比較実験を行う．具体的には，性能保証付貪欲アルゴリズム，乱択アルゴリズム，分枝限定法を本課題に適用する．さらにこれらを含めた数種類のアルゴリズムの比較実験を行う．また，比較実験の結果を参考に，文書クラスタへの関連性を考慮したモデルへの拡張を行う．

キーワード：自動要約，最大被覆問題，貪欲アルゴリズム，乱択アルゴリズム，整数計画法

Modelling Text Summarization with Optimization Problem

Hiroya Takamura　　Manabu Okumura
Tokyo Institute of Technology, Precision and Intelligence Laboratory
4259 Nagatsuta Midori-ku Yokohama, JAPAN, 226-8503

We discuss text summarization in terms of maximum coverage problem. We use appropriate algorithms to solve the optimization problem and conduct comprehensive experiments. Namely, we use a greedy algorithm with performance guarantee, a randomized algorithm, and a branch-and-bound method. We also conduct comparative experiments of some methods including these three, followed by extension of summarization model that takes into account the relevance to the document cluster.

Keywords: text summarization, greedy algorithm, randomized algorithm, integer programming

1　序論

本稿では，文書の自動要約手法を扱う．自動要約とは，与えられた単数あるいは複数の文書から，その内容を簡潔に表した短い文書 (要約) を生成する研究課題である (Mani, 2001)．良い要約を作るためには，いかにして冗長性を排除するかという点が大きな課題となる．

さて，自動要約の代表的な手法として，文選択によるものがある．これは，与えられた文書から必要な文を選択することにより要約を生成する手法である．この手法は，出力となる要約において，少なくとも文レベルでの文法性は保証されるという特長がある．本稿ではこの手法を取り扱う．これまでの文選択による要約手法の多くでは，各文を選択するか否かという二値分類問題を逐次的に解くことで要約が実現されてきた．例えば，maximal marginal relevance は，既に選択された文と似た文に対してペナルティが与えられる関数で，文の逐次選択に使われる局所的な目的関数といえる (Carbonell and Goldstein, 1998; Goldstein et al., 2000)．しかし，この種の逐次的な方法では，作成された要約が全体として良いかどうかという視点が考慮されていない．要約は全体で情報を担うものであり，全体としての良さを目標として作成されるべきである．

本稿では，文書要約を最適化問題に帰着させ，その問題を大域的に解くことを試みる．特に，我々は文書要約を最大充足化問題・最大被覆問題として捉える．このように帰着することの利点はいくつかある．まず，これらの最大化問題は，文章に表現されている事象が要約により被覆されているか否かを直接的にモデル化できる．二文間の類似度を差し引くことにより冗長性を減らすなどの場当たり的な対応が不要になる．冗長性は二文間の類似度の和で表現するのでなく，要約文章全体で測られるべきものであろう．また，解くべき問題を正確に把握することにより，組み合わせ最適化の分野で開発された様々な知見や計算手法を利用することができる．

本稿の貢献についてここでまとめておく．文書要約を最大被覆問題と見なしたのは我々が初めてではない．しかし，組み合わせ最適化の視点から，適切なアルゴリズムをこの問題に適用し，包括的な比較実験を行った研究はこれまでない．具体的には，性能保証付貪欲アルゴリズム，乱択アルゴリズム，分枝限定法を用いて最適化問題を解くことにより文書要約を実現した．さらにこれ

ら含めた数種類のアルゴリズムの比較実験を行った．また，本稿の後半では比較実験結果の観察に基づき，関連性を考慮した文書要約モデルへの拡張を行った．

2 文書要約のモデル化

本稿では，文選択による文書要約を扱う．文書要約タスクには，単文書要約タスクと複数文書要約タスクがある．前者は，一つの文書が与えられ，その文書の要約を作成するタスクであり，後者は，同じトピックについて記述した文書クラスタ (document cluster) が与えられ，その文書クラスタの要約を作成するタスクである．ここで提案する手法はどちらのタスクにも適用可能である．どちらの場合も，前処理として文書を何らかの言語単位に分割し，言語単位集合 $D = \{s_1, \cdots, s_{|D|}\}$ を作成する．ここから言語単位を選択することにより要約を作成することを考える．言語単位としては様々な種類が考えられるが，出力となる要約の文法性を保証するためにここでは文を用いることにし，以下の説明も文を用いて進める．ただし，提案する手法は言語単位には依存しない．

さらに，各文を構成する概念単位 (conceptual unit (Filatova and Hatzivassiloglou, 2004)) を考える．各文 s_i は，概念単位の集合 $\{e_{i1}, \cdots, e_{i|s_i|}\}$ で表されるとする．例えば，「男は，本を買って読んだ」という文は，“男が本を買った” という概念単位と，“男が本を読んだ” という概念単位から成ると見なすことができる．しかし，概念単位の粒度を定めるのは容易ではない．シンプルな方法では，前述の例文は，“男”，“本”，“買う”，“読む” という 4 つの概念単位から成ると考えることもできる．この概念単位の定義については，いくつかの研究があり，例えば Hovy ら (2006) は，文の依存木を切断することによって得られた部分依存木を概念単位とすることを提案し，これを basic element と呼んだ．本稿における実験では，単純に単語の語幹を概念単位とした．次節以降では，概念単位として単語を例にとって説明していく．

文書要約では，少数の文でより多くの概念単位を被覆することが目的となる．つまり，より多くの概念単位を被覆する部分集合 $S \subset D$ を選びたい．以下，そのためのモデルを紹介する．

2.1 個数制約付最大充足化問題によるモデル化

まず，選択できる文の個数が制限されている場合を考える．これは，一種の個数制約 (cardinality constraint) が与えられている場合である．選択できる文の最大数を K で表すことにする．

単語 e に対し，それを含む文が一つ以上選択されたときに，e は被覆されたとみなす．ここで，各文に一つのリテラルを対応させ，真ならばその文が選択され，偽ならば選択されないとする．単語に対し，それを含む文に対応するリテラルの選言を考えると，その真偽により e が被覆されたか否かを表現することができる．つまり，文書要約は，真が割当てられるリテラルの数が K 以下であるという制約の下で，真となる選言 (単語) の数を最大にする問題，すなわち個数制約付最大化充足問題 (maximum satisfiability problem with cardinality constraint (MAX-SAT-CC)) とみなすことができる．ただし，各選言は正のリテラルのみから成っているという点で，特殊な MAX-SAT-CC である．

後の議論のために，MAX-SAT-CC を整数計画問題の形で表現しておく．x_i は，文 s_i が選択された場合に 1 となり，それ以外の場合は 0 であるような変数とする．また，a_{ij} を，文 s_i が単語 e_j を含むときに 1 となり，それ以外の場合は 0 であるような定数とする．単語 e_j が被覆されるための必要十分条件は，$\sum_i a_{ij} x_i \geq 1$ である．よって，MAX-SAT-CC は，次の整数計画問題で表される：

$$\begin{aligned} maximize \quad & |\{j | \textstyle\sum_i a_{ij} x_i \geq 1\}| \\ s.t. \quad & \textstyle\sum_i x_i \leq K;\ \forall i, x_i \in \{0,1\}. \end{aligned}$$

しかし，この目的関数は扱いにくいので，補助的な変数 z_j を導入し，次のように同値な問題に書き換える：

$$\begin{aligned} maximize \quad & \textstyle\sum_j z_j \\ s.t. \quad & \textstyle\sum_i x_i \leq K;\ \forall j, \sum_i a_{ij} x_i \geq z_j; \\ & \forall i, x_i \in \{0,1\};\ \forall j, z_j \in \{0,1\}. \end{aligned}$$

単語 e_j が被覆される場合に z_j は 1 となり，それ以外の場合は 0 である．これは，MAX-SAT などの問題を整数計画問題で表現する標準的な方法である (Korte and Vygen, 2002)．

さて，単語のなかにも，重要なものとそうでないものがあるだろう．そこで，単語 e_j の重要度 w_j を導入する．目的は，被覆される単語の重要度の和を最大化することになる．つまり，上の整数計画問題の目的関数を $\sum_j w_j z_j$ として，重み付き MAX-SAT-CC として文書要約を捉えることができる．しかし，MAX-SAT-CC は NP 困難問題であり，厳密解を求めることは一般には難しい．

2.2 ナップサック制約付き最大被覆問題によるモデル化

ここまでは，選択される文数により制約が記述されていた．しかし，実際の応用を考えると，こ

の記述方法は一般的には好ましくない．なぜならば，文数によって制約が記述されている場合，長い文が選ばれやすくなるだろう．しかし，長い文から成る要約は，人間が読む場合にコストが高く，また限られた枠に納めることも困難になることがあるだろう．そこで，より現実的な制約の記述方法として，単語数による記述を考える．要約の共通課題を扱う DUC でも単語数による記述が採用されている (Dang, 2006)．

そこで新たに，選択された文の単語数の和が K 以下であるという制約を考える．整数計画問題で表すと，次のようになる：

$$\begin{aligned} maximize \quad & \textstyle\sum_j w_j z_j \\ s.t. \quad & \textstyle\sum_i c_i x_i \leq K;\ \forall j, \sum_i a_{ij} x_i \geq z_j; \\ & \forall i, x_i \in \{0,1\};\ \forall j, z_j \in \{0,1\}. \end{aligned}$$

ここで，c_i は文 s_i を選択するコスト，つまり文内の単語数である．

さて，これはナップサック制約付きの重み付き MAX-SAT-CC であるが，正のリテラルのみから成ることを考えると，ナップサック制約付き最大被覆問題 (maximum coverage problem with knapsack constraint (MCKP)) であることがわかる．ただし，この問題は制約がナップサック制約であるに過ぎず，いわゆるナップサック問題とは異なるものであることを強調しておく．MCKP も NP 困難問題である．文書要約が MCKP として定式化できるという事実は，Filatova ら (2004) により指摘されている．

2 節冒頭で，本稿では概念単位として単語のみを考えると述べた．しかし，単語の表現方法や，採用する単語の種類などにより，結果は変わってくる．また，単語の重み w_j の決め方も非常に重要である．本稿では，単語の表現方法として，語幹を用いた．また，内容語であること (品詞が名詞，動詞，形容詞のいずれか)，またストップワードリスト (ROUGE で用いられているもの) に記載されていないことの二つの条件を満たす単語のみ概念単位として用いた．

単語の重みについては，先行研究 (Yih et al., 2007) で提案されている 2 種類の方法を試した．一つ目は，文書クラスタでの出現頻度に比例する重みを付けたものだが，各文書の始めの部分に出てきた場合は重くカウントしている (補完重みと呼ぶことにする)．二つ目は，訓練データで学習した logistic regression を用いて，各単語が人手で作られた要約に出てくる確率を計算し，これを重みとして使った (訓練重みと呼ぶことにする)．その際に使用されている素性は，文書クラスタ内での頻度や，出現位置などである．

3 最適化問題の解法

本節では，MCKP の解法について述べる．Filatova ら (2004) が用いた貪欲アルゴリズムをまず紹介する．次に，それを修正した性能保証付貪欲アルゴリズムを紹介する．この性能保証付貪欲アルゴリズムの文書要約への適用はこれまで試みられておらず，本稿が初めてである．次に Yih ら (2007) が用いたスタック・デコーディングを紹介する．Yih らは MCKP という用語は用いてないが，使用した目的関数は MCKP と同じものである．その後，乱択アルゴリズム (randomized algorithm) による近似解法を紹介する．さらに，分枝限定法 (branch-and-bound method) による厳密解法を紹介する．

ここで紹介するアルゴリズム自体は新しいものではないが，性能保証付貪欲アルゴリズム，乱択アルゴリズム，分枝限定法を用いて MCKP を解くことにより文書要約を実現した研究はこれまでにない．また，これらの解法を文書要約問題に対して包括的な形で適用し，相互比較したのは本稿が初めてである．

これ以外の近似解法として，MAX-SAT に対しては条件付確率法 (method of conditional probability)(Hromkovič, 2003) が知られているが，ナップサック制約の場合は条件付確率の計算が困難であり，MCKP には使えない．また，pipage アプローチ (Ageev and Sviridenko, 2004) も知られているが，部分的な列挙が必要である上に，多数個の緩和問題を解く必要があり，計算量的に優位であるとはいえないので今回は触れない．

本節の最後に，スタック・デコーディングなどの既存アルゴリズムにおいては，非文法的な要約が生成されるという問題点を指摘し，それを回避する方法を考える．

3.1 貪欲アルゴリズム

Filatova ら (2004) は貪欲アルゴリズムを用いている．W_l は文 s_l が被覆する単語の重みの和を表し，W'_l は文 s_l が被覆する単語の中で，S で被覆されていないものの重みの和を表すとする．W'_l が最大である s_l を逐次的に選択していく．

Algorithm 1
$S \leftarrow \phi$
while $D \backslash S \neq \phi$
  $s_i \leftarrow \arg\max_{s_l \in D \backslash S}\ W'_l$
  insert s_i into S
end while
output S

このアルゴリズムは，各文のコスト c_i が等しい場合は性能保証が与えられるが，文書要約で必

要になる一般の場合については性能保証が与えられていないものであった[1].

3.2 性能保証付き貪欲アルゴリズム

ここではKhullerら (1999) によって提案された性能保証のある貪欲アルゴリズムを紹介する．このアルゴリズムは，W_l' とコスト c_l の比 W_l'/c_l が最大となる s_l を逐次的に選択していく．さらに，このようにして逐次的な方法で得られた要約と，単文から成る要約を比較して，目的関数値の高い方を出力とするアルゴリズムである．

Algorithm 2
$S \leftarrow \phi$
while $D \backslash S \neq \phi$
  $s_i \leftarrow \arg\max_{s_l \in D \backslash S}\ W_l'/c_l$
  insert s_i into S
end while
$s_t \leftarrow \arg\max_{s_l}\ W_l$
if $score(S) \geq W_t$, output S,
otherwise, output $\{s_t\}$

ここで，$score(S)$ は要約 S の目的関数値である．このアルゴリズムの解は，最悪でも最適解の $(1-1/e)/2$ を達成することが証明されている (Khuller et al., 1999)．彼らはさらに，より高い性能が保証されているアルゴリズムを提案しているが，部分的な列挙が必要とされ，計算コストが高いので，今回は採用しない．

3.3 スタック・デコーディング

スタック・デコーディングは，Jelinek (1969) によって提案され，Yihら (2007) により，文書要約のデコーディングに適用された．これは，K 個の優先度付きキューを用意し，k 番目の優先度付きキューに k 単語からなる要約を入れ，番号の若いキューに文を加えることにより新たな要約を作成し，番号の古いキューに保存していく，一種の動的計画法である[2]．各キューのスタックサイズを定数 ($maxsize$) に制限することにより，現実的な計算時間で近似解を得ることができる．

Algorithm 3
for $k = 0$ **to** $K-1$
  for each $S \in queues[k]$
    for each $s_l \in D$
      insert s_l into S
      insert S into $queues[\min(k+c_l, K)]$
      pop if queue-size exceeds **maxsize**
    end for
  end for
end for
return the best solution in $queues[K]$

3.4 乱択アルゴリズム

Khullerら (2006) は，MCKP に乱択アルゴリズム (randomized algorithm)(Hromkovič, 2003) を適用することを提案した．彼らの方法では，まずMCKP の整数制約である $x_i \in \{0,1\}$ 及び $z_j \in \{0,1\}$ を，それぞれ $x_i \in [0,1]$ 及び $z_j \in [0,1]$ に置き換えることにより，線形緩和問題を作成する．得られた線形緩和問題の解 x_i^* を，文 s_i が選択される確率 ($P(x_i = 1)$) とみなして，その確率に従って各文 s_i が選択されるか否かをランダムに決定する．このランダム生成を繰り返すことにより，多数の解候補を生成し，目的関数値が最も高くなるものを選ぶ．

このアルゴリズムで生成される各解候補について，その語数の期待値は K 以下であること，また目的関数値の期待値は最適解の少なくとも $(1-1/e)$ 倍であることが示せる (Khuller et al., 2006)．

3.5 分枝限定法

分枝限定法は，整数計画問題の厳密解を計算する方法である．一般には多項式時間では解けないが，サイズの小さな問題ならば，分枝限定法により実用的な時間で厳密解が求まることがある (Hromkovič, 2003)．本稿での実験では利用可能だったが，スケーラビリティに乏しいので，常に利用できるわけではなく，近似手法の併用が不可欠である．

3.6 アルゴリズムの強制約化

上で紹介したアルゴリズムのうち，分枝限定法は語数制限に厳密に従う要約を生成する．しかし，他のアルゴリズムは語数制限に常に従うとは限らない．ROUGE(Lin, 2004) などを用いた評価では，語数制限を越えた部分は切り捨てて評価値が算出されるので，語数の超過によって評価値が大きくなることはない．しかし，そもそも文選択アプローチの長所は，文レベルの文法性が保証されることであったにも関わらず，超過部分が切り捨てられることでその長所が失われてしまうことになる．特に，スタック・デコーディングは，語数制限に従わない多くの解候補の中で，切り捨てを考慮した上で最も目的関数値が高いものを選んでおり，非文法性を許すことで評価値を高めていることになり，本来の文書要約の目的にそぐわない．

[1] McDonald(2007) は，Filatovaら (2004) のアルゴリズムが性能保証を持つとして参照しているが，それは等コストの場合のみに成立し，Filatova らの要約実験の設定では性能は保証されていない．

[2] データ構造の用語としての厳密な意味でのスタックは使用されていないことに注意してほしい．

このような観点から，我々は，各アルゴリズムに対して語数制限に厳密に従う変種を用意し，これらについて議論を進めていく．これらの変種を強制約アルゴリズムと呼ぶことにする．各々の強制約アルゴリズムは次の修正により得られる：

- 貪欲アルゴリズム：制約 $c_i + \sum_{s_l \in S} c_l \leq K$ を満たす場合にのみ文の挿入を許可する．
- 性能保証付貪欲アルゴリズム：制約 $c_i + \sum_{s_l \in S} c_l \leq K$ を満たす場合にのみ文の挿入を許可する．
- スタック・デコーディング：$queues[\min(k + c_l, K)]$ を，$queues[k + c_l]$ に書き換える．
- 乱択アルゴリズム：語数制約を満たす解候補のみから最終的な出力を選ぶ．語数の期待値が K 以下であるので，充分な回数だけランダム生成を行えば，多数個のそのような解候補が生成すると期待される．

4 実験及び考察

4.1 実験設定

実験には，DUC'04(2004) のデータを用いた．DUC'04 の task 2 と同じ設定で実験を行った．これは複数文書要約タスクであり，それぞれ 10 個程度の文書から成る 50 個の文書クラスタが与えられる．各文書クラスタに対して一つの要約を作成することが求められる，要約の長さは 100 語以内とした．また，単語の重み計算に学習を用いる場合の訓練データには，DUC'03(2003) の task 2 のデータを利用した．語幹の抽出には，Porter(1980) の語幹抽出アルゴリズムを用いた．

評価には ROUGE version 1.5.5(Lin, 2004) を用いた．特に ROUGE-1 を用いて結果の分析を行った[3]．評価値の差の検定には，対応のある 2 標本のための Wilcoxon 符号順位検定を用いた．一部のアルゴリズムで，線形計画問題及び整数計画問題を解く必要があるが，これには GLPK パッケージ (Makhorin, 2006) を用いた．線形計画問題は単体法を，整数計画問題は前述のように分枝限定法を用いて解いた．

ここで比較する手法は，貪欲アルゴリズム (greedy)，性能保証付貪欲アルゴリズム (g-greedy)，乱択アルゴリズム (rand)，スタック・デコーディング (stack)，分枝限定法 (exact) の 5 種類である．リード文から要約を生成する手法などがベースラインとして通常用いられるが，先行研究 (Yih et al., 2007) において，ベースラインに対するスタック・デコーディングの優位性は示されていると考えられるので，ベースラインとの比較は省略した．

[3]用いたオプションは次の通り：
-n 4 -x -m -2 4 -u -f A -p 0.5 -l 100 -t 0 -d -s .

表 1: ROUGE 値 (補完重みの場合)

手法	1		2	SU4
	強制約	弱制約		
greedy	0.291	0.283	0.083	0.125
g-greedy	0.311	0.311	0.085	0.124
rand100k	0.312	0.321	0.082	0.123
stack30	0.322	0.318	0.084	0.125
exact	0.318	–	0.083	0.125

表 2: ROUGE 値 (訓練重みの場合)

手法	1		2	SU4
	強制約	弱制約		
greedy	0.295	0.283	0.083	0.124
g-greedy	0.320	0.331	0.080	0.119
rand100k	0.317	0.321	0.083	0.121
stack30	0.322	0.329	0.079	0.121
exact	0.325	–	0.081	0.123

4.2 実験結果

表 1，2 に実験結果を示す．表中，列 1，2，SU4 はそれぞれ，ROUGE-1，ROUGE-2，ROUGE-SU4 を表す．ROUGE-1 については，強制約と弱制約の両方の場合について値を記した．また，rand100k は，乱択アルゴリズムにおいて解候補のランダム生成を 100,000 回行った場合の結果である．stack30 は，スタック・デコーディングにおいて，スタックサイズを 30 にした場合の結果である．補完重みの場合，一つの文書クラスタから要約を生成するための平均計算時間は，greedy と g-greedy が 0.01 秒，rand100k が 3.4 秒，stack30 が 3.3 秒，exact が 4.7 秒であった．訓練重みでも同程度の計算時間であった．

補完重みの場合 (表 1) も，訓練重みの場合 (表 2) も，性能保証付貪欲アルゴリズムは貪欲アルゴリズムを有意に上回った (有意水準 $\alpha = 0.01$)．補完重みの場合は，greedy を除けば，exact と他のアルゴリズムの間に有意な差はなく，高速なアルゴリズムで厳密解と同等の性能が出ることが確認された．訓練重みの場合は，greedy と rand100k を除き，exact と他のアルゴリズムの間に有意な差はなかった．強制約より弱制約が総じて良くなっていることが多く，弱制約が非文法性を許すことで評価値を上げていることがわかる．

表 3: 様々なスタックサイズに対するスタック・デコーディング (強制約) の ROUGE 値

手法	補完重み	訓練重み
stack10	0.321	0.326
stack20	0.325	0.323
stack30	0.322	0.322
stack40	0.322	0.322
stack50	0.322	0.322
stack70	0.321	0.322
stack100	0.321	0.323
stack200	0.321	0.323

注目すべきであるのは，補完重みの場合に stack30 が exact を上回っている点である．この点を調査するために，stack のスタックサイズを変化させて性能への影響を調べた．結果は表 3 の通りである．この結果から，学習重みの場合もスタックサイズを 10 まで減少させると，ROUGE-1 の値が exact を上回ることがわかる．また，最も性能が高くなる点はスタックサイズが小さい箇所にあり，その後は ROUGE-1 値は減少傾向にある．スタックサイズが大きいほど，より多くの解候補の中から目的関数値で最適な要約を選んでいることを考えると，これは直観に反する結果である．スタック・デコーディングにより局所的な解探索を行うことが何らかの良い影響を与えていることになる．しかし，そのような局所探索が最適解を与えているとは考えにくく，より適した目的関数を設定し，それを最適化する努力をするのがよいであろう．

5 文書要約モデルの拡張

前節の考察を踏まえ，モデルの拡張を行う．現在のモデルについてもう一度考えてみる．前述のように，適切な概念単位の粒度及び定義は非常に困難であるため，我々は単語を概念単位とした．しかし，“A が B を C した” という事象のような，単語より情報量の高いものが概念単位として適切であることもあるだろう．そのような場合，“A が D を E した” という事象は異なる概念単位であるが，単語を概念単位として利用していると，二つの事象間に A という冗長な概念が生じる．つまり，要約としては簡潔であっても，単語レベルでは冗長性を有するのである．別の見方をすると，冗長な単語を有することにより，選択される文の文書クラスタへの関連性を保っていると見ることもできる．また，要約文書の結束性や一貫性を保つためには，ある程度の冗長性が必要とされるであろう．

5.1 モデルの拡張

MCKP の目的関数は被覆度合いを表す項から成っていたが，これに文書クラスタの主題への関連性 (relevance) に対応する項を加える．文 s_j の関連性は，s_j 内に出現する単語の重みの和 $\sum_j w_j a_{ij}$ で表されるとし，これを選択されたすべての文について足し合わせる：

$$\begin{aligned} maximize \quad & \textstyle\sum_j w_j z_j + \lambda \sum_i (\sum_j w_j a_{ij}) x_i \\ s.t. \quad & \textstyle\sum_i c_i x_i \le K;\ \forall j, \sum_i a_{ij} x_i \ge z_j; \\ & \forall i, x_i \in \{0,1\};\ \forall j, z_j \in \{0,1\}. \end{aligned}$$

ここで λ は定数である．関連性を考慮したモデルであるので，ここでは MCKP-Rel と呼ぶことにしておく．

McDonald(2007) のモデルは，関連性から冗長性を差し引いた目的関数を用いており，MCKP-Rel と抽象的なレベルで類似している．しかし，McDonald は冗長性を，二文から成るペアを用いて定義している．MCKP-Rel は要約全体の被覆度合いを用いており，より直感的である．

MCKP-Rel に対しても，前述の種々のアルゴリズムが適用可能であるが，乱択アルゴリズムの適用可能性については説明を加える必要がある．乱択アルゴリズムで生成される要約の語数の期待値に関しては，3.4 節と全く同じ議論が成立する．次に乱択アルゴリズムの近似性能について考える．乱択アルゴリズムを用いて 1 つの解候補をサンプリングしたとき，その解が与える目的関数値の期待値の下限を求めてみる．Khuller ら (2006) の MCKP における下限に関する議論と同様の議論が，MCKP-Rel に対しても可能である．

まず，$P(z_j = 1)$ の下限を求める：

$$\begin{aligned} P(z_j = 1) &= 1 - P(z_j = 0) \\ &= 1 - \prod_{i \in S_j} (1 - x_i^*) \\ &\ge 1 - \prod_{i \in S_j} \exp(-x_i^*) \\ &= 1 - \exp(\sum_{i \in S_j} -x_i^*) \\ &= 1 - \exp(\sum_i -a_i x_i^*) \\ &\ge 1 - \exp(-z_j^*) \\ &\ge (1 - 1/e) z_j^*. \end{aligned}$$

これを用いて，乱択アルゴリズムによってランダム生成された 1 つの要約 S に対する目的関数値の期待値を下から抑えることができる：

表 4: MCKP-Rel の ROUGE 値 (補完重み)

手法	強制約	弱制約
rand100k	0.322 (0.2)	0.335 (0.6)
stack30	0.328 (0.1)	0.326 (0.3)
exact	0.339 (0.2)	–

表 5: MCKP-Rel の ROUGE 値 (訓練重み)

手法	強制約	弱制約
rand100k	0.329 (0.6)	0.335 (0.6)
stack30	0.326 (0.2)	0.335 (0.3)
exact	0.333 (0.5)	–

$$
\begin{aligned}
& E[obj(S)] \\
\geq\ & (1-1/e)\sum_j w_j z_j^* + \lambda \sum_i (\sum_j w_j a_{ij}) x_i^* \\
=\ & (1-1/e) OPT_{LP} \\
\geq\ & (1-1/e) OPT_{original}.
\end{aligned}
$$

5.2 拡張モデルの実験

ここでは，スタック・デコーディング，乱択アルゴリズム，分枝限定法を試した．実験に用いたデータや実験設定は前節と同じである．MCKP-Rel で使われる λ の値は，DUC2003 のデータを開発データとして用いて決定した．具体的には，λ を 0.0 から 1.0 まで 0.1 刻みで変化させ，ROUGE-1 が最大となる λ を選んだ．

表 4，5 に，結果を示す．ここでは，議論の対象となる ROUGE-1 の値のみを記す．強制約の場合を見ると，exact が最も高い ROUGE 値を出している．MCKP では局所解が高い ROUGE 値を出しており，目的関数が不適切であると考えられたが，そのような問題は MCKP-Rel では軽減されていると思われる．

パラメータ λ の値を変化させたときの ROUGE-1 の変化を，図 1，2 に示す．簡単のため，強制約の場合のみを示すことにする．図の左端 ($\lambda = 0.0$) が，MCKP に対応する．図からわかるように，MCKP を MCKP-Rel が常に上回っている．また，右端の方が ROUGE 値が低くなっている場合が多く，文書クラスタへの関連性を重視しすぎると性能が下がることがわかる．つまり，被覆性はやはり重要であるといえる．

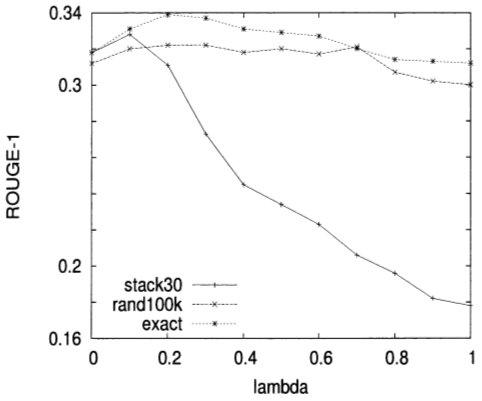

図 1: MCKP-Rel (強制約，補完重みの場合)

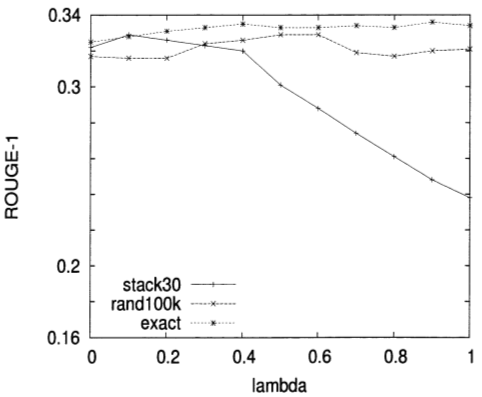

図 2: MCKP-Rel (強制約，訓練重みの場合)

6 結論

本稿では，文書要約を最大被覆問題として解いた．組み合わせ最適化の視点から，適切なアルゴリズムをこの問題に適用し，包括的な比較実験を行った．具体的には，性能保証付貪欲アルゴリズム，乱択アルゴリズム，分枝限定法を本課題に適用した．さらにこれらを含めた数種類のアルゴリズムの比較実験を行った．また，比較実験の結果を参考に，文書クラスタへの関連性を考慮したモデルへの拡張を行った．

今後の発展としては，まず概念単位の工夫が挙げられる．要約の評価のために提案されている basic element(Hovy et al., 2006) などを，概念単位に利用することが考えられる．また，Yih ら (2007) のように，短縮した文を選択候補に加えることも可能である．また，最適化問題の計算

手法についても，要約問題に特化させた形で高速化して利用するという発展も考えられる．特に，pipage アプローチ (Ageev and Sviridenko, 2004) などは有望である．

モデルとしては，文の選択だけでなく，文を適切な順序に並べ替える技術も考えている．Deshpande ら (2007) は，選択と並べ替えの手法を提案しているが，彼らは単語を選択してタイトルを生成するという，選択要素が等コストであるような課題を扱っている．一般の文書要約に適用するには新たな技術の開発が必要である．

参考文献

Alexander A. Ageev and Maxim Sviridenko. 2004. Pipage rounding: A new method of constructing algorithms with proven performance guarantee. *Journal of Combinatorial Optimization*, 8(3):307–328.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 335–336.

Hoa Trang Dang. 2006. Overview of DUC 2006. In *Proceedings of Document Understanding Conference 2006.*

Pawan Deshpande, Regina Barzilay, and David Karger. 2007. Randomized decoding for selection-and-ordering problems. In *Proceedings of the Human Language Technologies Conference and the North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL)*, pages 444–451.

DUC'03. 2003. Document Understanding Conference. In *HLT/NAACL Workshop on Text Summarization.*

DUC'04. 2004. Document Understanding Conference. In *HLT/NAACL Workshop on Text Summarization.*

Elena Filatova and Vasileios Hatzivassiloglou. 2004. A formal model for information selection in multi-sentence text extraction. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 397–403.

Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *Proceedings of ANLP/NAACL Workshop on Automatic Summarization*, pages 40–48.

Eduard Hovy, Chin-Yew Lin, Liang Zhou, and Junichi Fukumoto. 2006. Automated summarization evaluation with basic elements. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC).*

Juraj Hromkovič. 2003. *Algorithmics for Hard Problems.* Springer.

Frederick Jelinek. 1969. Fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, 13:675–685.

Samir Khuller, Anna Moss, and Joseph S. Naor. 1999. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45.

Samir Khuller, Louiqa Raschid, and Yao Wu. 2006. LP randomized rounding for maximum coverage problem and minimum set cover with threshold problem. Technical Report CS-TR-4805, The University of Maryland.

Bernhard Korte and Jens Vygen. 2002. *Combinatorial Optimization: Theory and Algorithms.* Springer.

Chin-Yew Lin. 2004. ROUGE: a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out*, pages 74–81.

Andrew Makhorin, 2006. *Reference Manual of GNU Linear Programming Kit, version 4.9.*

Inderjeet Mani. 2001. *Automatic Summarization.* John Benjamins Publisher.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, pages 557–564.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Wen-Tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. 2007. Multi-document summarization by maximizing informative content-words. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1776–1782.