木刈込みに基づく文書要約のためのZDDを用いた動的計画法

西野 正彬[1]　安田 宜仁[2]　平尾 努[1]　湊 真一[2,3]　永田 昌明[1]

[1]NTT コミュニケーション科学基礎研究所
[2] JST ERATO 湊離散構造処理系プロジェクト
[3] 北海道大学 大学院情報科学研究科
{nishino.masaaki,hirao.tsutomu,nagata.masaaki}@lab.ntt.co.jp
yasuda@erato.ist.hokudai.ac.jp　minato@ist.hokudai.ac.jp

概 要

入力文書を木とみなし，長さ制約下でその最適な部分木を求める木刈込み問題として文書要約問題を定式化して解く方法が提案されている．本稿ではこれら木刈込み問題に対する，動的計画法に基づいた解法を提案する．既存手法で用いられていた整数線形計画問題 (ILP) ソルバを用いた解法と比較すると，提案法には (1) 必ず厳密解が求まる，(2) 実行時間を入力の木のノード数から見積もることができる，(3) ILP ソルバよりも高速である，という 3 つの利点がある．

1　はじめに

組合せ最適化に基づく要約手法の一種として，入力テキストをその最小構成単位がノードに対応するような木として表現し，長さ制約を満たし，かつ与えられた目的関数の値を最大化するような部分木を求めることで要約を生成する手法が提案されている [Filippova 08, Filippova 13, Hirao 13, Kikuchi 14] ．部分木中では入力の木におけるノード間の関係性が保持されるため，これらの手法によって言語的によい性質をもった要約を得ることができる．

これら木刈込み問題に基づく手法に対する効率的な求解アルゴリズムはこれまで知られておらず，そのため既存手法では ILP ソルバを用いて解を求めていた．しかし，ILP ソルバには (1) 常に解が求まるとは限らない，(2) 解が求まるとしても実行時間を事前に見積ることができない，という問題点があった．

本稿では木刈込み問題に対する動的計画法アルゴリズムを提案する．提案法は動的計画法であるため，常に厳密解を求めることができる．また，実行時間は木のノード数 N，要約の長さ制約 L に対して $O(NL\log N)$ である．提案法のポイントは**ゼロサプレス型二分決定グラフ** *(Zero-suppressed Binary Decision Diagrams: ZDD)*[Minato 93] を用いて解候補である部分木の集合をコンパクトに表現する点にある．ZDD を用いることで，異なる木刈込み問題を統一的に解くことができ，かつ ZDD の構造から計算時間の上限を与えることができる．さらに，実験ではソルバと比較して提案法が 10 倍〜300 倍程度高速であること，問題が大規模になっても実行時間が抑えられることを確認した．

2　木刈込み問題

本稿では 3 種類の木刈込み問題を扱う．いずれの問題でも，入力の文もしくは文書を $D = \{e_1, \ldots, e_N\}$ とする．e_i は D の i 番目の構成単位とし，w_i, l_i を e_i のスコアと長さとする．スコアは各問題に対して，たとえば e_i に含まれる語の出現頻度などに基づいて適切に定める．長さ制約 L のもとでの木刈込み問題は

$$\begin{aligned} &\text{Maximize} \sum_{e_i \in T} w_i \\ &\text{Subject to } T \in \mathcal{T} \text{ and } \sum_{e_i \in T} l_i \leq L \end{aligned} \tag{1}$$

として定式化できる．ここで T は D の部分集合であり，ある部分木を表す．$\mathcal{T}$ は解として選択できる部分木の集合である．$\mathcal{T}$ の定義は問題によって異なる．

EDU 抽出　Hirao らは木刈込みに基づく単一文書要約手法を提案した [Hirao 13]．彼らの手法では，まず対象テキストに対して Elementary Discourse Unit (EDU) をノードとし，EDU 間の依存関係をエッジとするような依存構造木を作る．次にこの木に対して問題 (1) を解くことによって要約を得る．部分木の集合 $\mathcal{T}$ は入力の依存構造木の根ノードを含む任意の部分木の集合とする．

文短縮　Filippova らは単一の文を依存構造木として表現して，その最適な部分木を求めることで文短縮する手法を提案した [Filippova 08]．この手法も問題 (1) の木刈込み問題を解いているとみなすことができる．ただし，部分木の集合 $\mathcal{T}$ は EDU 抽出の場合とは異なり，根候補集合 $R \subseteq D$ のうちのいずれかを根とする任意の部分木の集合とする．

文短縮・抽出　Kikuchi らは [Hirao 13] の発展形として，木刈込みに基づいて文短縮と文抽出を同時に行う手法を提案した [Kikuchi 14]．彼らの方法は [Hirao 13] の EDU 抽出法を拡張し，文書を文をノードとする木（以下，外側の木とよぶ）として表現したのちに，各文をさらに単語をノードとする木（以下，内側の木とよぶ）として表現して入れ子の木を作り，その最適な部分木

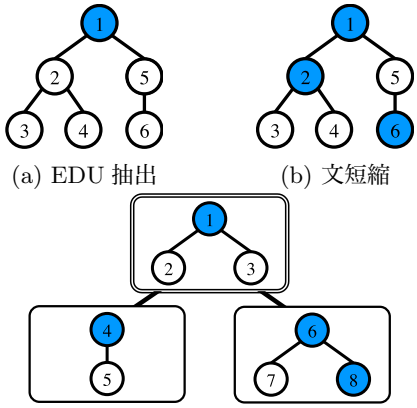


図 1: 各木刈込み問題における入力の例

表 1: 図 1 の入力に対して，木の集合 $\mathcal{T}$ に含まれる部分木，含まれない部分木の例

	含まれる	含まれない
EDU 抽出	$e_1e_2e_3$, $e_1e_2e_5$	$e_2e_3e_4$, e_6
文短縮	$e_1e_2e_5$, $e_2e_3e_4$	e_3, e_5e_6
文短縮・抽出	$e_1e_2e_4$, $e_1e_4e_5e_8$	$e_4e_5e_6$, $e_1e_2e_7$

を求める．部分木の集合 $\mathcal{T}$ に含まれる木 T は，外側の木に対しては根を含む部分木を作成し，その後外側の木の部分木に含まれる各ノードに対応する内側の木に対して，根候補集合に含まれるいずれかのノードを根とする任意の部分木を作成したものとする．

図 1 は 3 つの木刈込み問題における入力の木の例である．各ノード中の数字はノードの番号を表している．また，色が塗られたノードは根候補集合に含まれるノードである．これらの入力に対する部分木の集合 $\mathcal{T}$ に含まれる/含まれない部分木の例を表 1 に示す．なお，以下では部分木 T をそれに含まれるノードの集合を用いて表現する．

3 ZDD

提案法のポイントは部分木の集合 $\mathcal{T}$ を ZDD を用いて表現する点にある．ZDD は DAG であり，部分木の集合を少ないノード数の ZDD として表現することで効率的に解を求めることができる．図 2 (a) は集合族 $\{e_1e_2, e_1e_3, e_2e_3\}$ を表現する ZDD の例である．ZDD は終端ノードと分岐ノードの二種類のノードをもつ．1 つの ZDD は必ず $\top$ と $\bot$ の 2 つの終端ノードをもつ．図中では各終端ノードを四角で表現している．分岐ノードの出次数は必ず 2 である．ある分岐ノードを始点とする 2 つのエッジは LO エッジ，HI エッジ，また各エッジの終点であるノードを LO 子，HI 子とよぶ．分岐ノードは $e_1, \ldots, e_N$ のいずれかをラベルとしてもつ．図中では分岐ノードを円で表現しており，中の数字は対応するラベル e_i の添字 i を表している．また，LO エッジは破線，HI エッジは実線でそれぞれ表されている．ZDD には親ノードをもたないノードが必ず 1 つだけ含まれており，これを根ノードとよぶ．

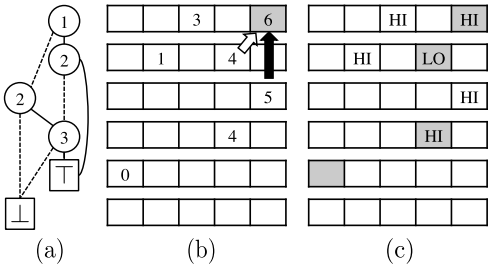


図 2: ZDD とそれを用いた動的計画法の例．(a) 集合族 $\{e_1e_2, e_1e_3, e_2e_3\}$ を表す ZDD, (b) テーブル S, (c) テーブル B.

Algorithm 1 動的計画法

Input: 集合族 $\mathcal{T}$ を表現する ZDD，長さ制約 L, 重み w_i, 長さ l_i $(1 \le i \le N)$
Output: 最適な部分木 T

Initialize $S[i][j] \leftarrow -\infty$ **for all** i, j.
$S[Z-1][0] \leftarrow 0$.
for $i = Z-2, \ldots, 1$ **do**
  for $j = 0, \ldots, L$ **do**
    if $j \ge l_{\mathrm{v}(i)}$ **and** $S[\mathrm{hi}(i)][j - l_{\mathrm{v}(i)}] + w_{\mathrm{v}(i)} > S[\mathrm{lo}(i)][j]$ **then**
      $S[i][j] \leftarrow S[\mathrm{hi}(i)][j - l_{\mathrm{v}(i)}] + w_{\mathrm{v}(i)}$
      $B[i][j] \leftarrow HI$
    else
      $S[i][j] \leftarrow S[\mathrm{lo}(i)][j]$, $B[i][j] \leftarrow LO$
$k^* \leftarrow \mathrm{argmax}_{0 \le k \le L+1} S[1][k]$
$i \leftarrow 1$, $j \leftarrow k^*$, $T \leftarrow \emptyset$
while $(i, j) \neq (Z-1, 0)$ **do**
  if $B[i][j] = HI$ **then**
    $T \leftarrow T \cup \{\mathrm{v}(i)\}$, $i \leftarrow \mathrm{hi}(i)$, $j \leftarrow j - l_{\mathrm{v}(i)}$
  else
    $i \leftarrow \mathrm{lo}(i)$
return T

ZDD 中では，根ノードから $\top$ 終端ノードまでのパスが集合族に含まれる 1 つの集合に対応する．図の ZDD に対しては 3 つの異なるパスが存在し，それが集合族に含まれる 3 つの集合にそれぞれ対応している．ZDD の各ノードを，根ノードから順に $z_1, \ldots, z_Z$ とする．ここで Z は ZDD のノード数とする．また，z_Z を $\bot$ 終端ノード，z_{Z-1} を $\top$ 終端ノードとする．ノード z_j の LO 子，HI 子，ラベルをそれぞれ $\mathrm{lo}(j)$, $\mathrm{hi}(j)$, $\mathrm{v}(j)$ と表す．なお，ZDD の分岐ノードのラベルには順序が定められているものとする．すなわち，根ノードから $\top$ 終端ノードへのいずれのパスにおいても，そのパス中に出現する分岐ノードのラベルの順序は一定であるとする．図の ZDD では $e_1 < e_2 < e_3$ の順序が用いられている．

4 動的計画法による木刈込み問題の求解

提案法は 0-1 ナップサック問題のための標準的な動的計画法と似ている．アイテム数 N，長さ制約 L のもとでの 0-1 ナップサック問題は，まず $N \times (L+1)$ 要素からなる 2 次元テーブルを用意し，その各要素を再帰的に計算することによって $O(NL)$ で解くことができる．提案法でも同様に 2 次元テーブルを用意し，それを埋めることによって問題を解く．ただし，用意す

表 2: 木刈込み問題ごとの ZDD ノード数のオーダ

問題	ZDD ノード数
EDU 抽出	$O(N)$
文短縮	$O(N \log \|R\|)$
文短縮・抽出	$O(N \log \|R^*\|)$

るテーブルは部分木の集合 $\mathcal{T}$ を表現する ZDD のサイズ Z に対して $Z \times (L+1)$ 要素をもつ．つまり ZDD の各ノードに対して $L+1$ 個の要素をもつ配列を用意することによってテーブルを準備する．このテーブルを ZDD の構造に由来する再帰式に基づいて埋めることで木刈込み問題を解くことができる．

ZDD を用いた動的計画法によっての手順をアルゴリズム 1 に示す．アルゴリズムはテーブルの要素を埋める手続き (1-9 行目) と，バックトラック手続き (10-17 行目) から構成されている．まず $N \times (L+1)$ 要素をもつ 2 次元テーブル S と B とを用意する．S は部分問題に対する最適解を保持するために用いる．具体的には，$S[i][j]$ $(1 \leq i \leq Z, 0 \leq j \leq L)$ には ZDD のノード z_i を根とする ZDD が表す集合族に含まれる，長さの合計が j であるような集合の重みのうち，最大のものを保持している．B はバックトラック時に最適解を与える遷移先を記憶するために用いられる．S を初期化したのちに (2 行目)，ZDD の終端ノードから根ノードの順にたどりながら，各 ZDD ノードに対応する S, B の要素を順に更新する (5-9 行目)．$S[i][j]$ の値は z_i の 2 つの子ノード $\mathrm{hi}(i)$ と $\mathrm{lo}(i)$ のいずれかに由来する値のうち，より大きなほうを設定する．そして，$B[i][j]$ には $S[i][j]$ をどのように設定したかを HI, LO いずれかのラベルを与えることで記憶する．提案法によって集合族 $\mathcal{T}$ に含まれる部分木のうち，長さ制約 L を満たすもの求めることができる．その実行時間はテーブル S, B のサイズに比例するため $O(ZL)$ である．

アルゴリズムの動作例を図 2 で示す．ここで $\mathcal{T} = \{e_1e_2, e_1e_3, e_2e_3\}$，長さ・重みをそれぞれ $(l_1, l_2, l_3) = (1, 1, 3)$, $(w_1, w_2, w_3) = (2, 1, 4)$，$L = 4$ とする．$\mathcal{T}$ は図 2 (a) の ZDD で表現され，また処理終了時点でのテーブル S, B はそれぞれ図 (b), (c) のようになる．いま，$S[1][4]$（図 (b) 中の右上のセル）を更新するときの処理を考える．z_1 の 2 つの子ノードのうち，HI 子由来の値は $S[2][3] + w_1 = 5$ であり，LO 子由来の値は $S[3][4] = 5$ である．そのため，HI 子由来の値 6 を $S[1][4]$ に設定し，$B[1][4]$ には HI 子由来であることを示すために HI を設定する．バックトラックでは $B[1][4]$ からスタートして，$B[1][4] \to B[2][3] \to B[4][3] \to B[5][0]$. と進む．最終的にこのパスが対応する集合 e_1e_3 を出力する．

5 ZDD のサイズ

前節で示した動的計画法の実行時間は $\mathcal{T}$ を表現する ZDD のノード数に依存する．ここでは各木刈込み問題における部分木の集合 $\mathcal{T}$ を表現する ZDD について，そのノード数の上限を与える．具体的には，木

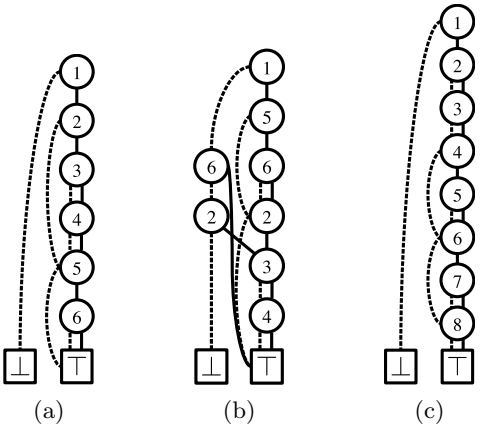


図 3: 図 1 の各入力の木に対応する部分木の集合 $\mathcal{T}$ を表現する ZDD．(a) EDU 抽出，(b) 文短縮，(c) 文短縮・抽出

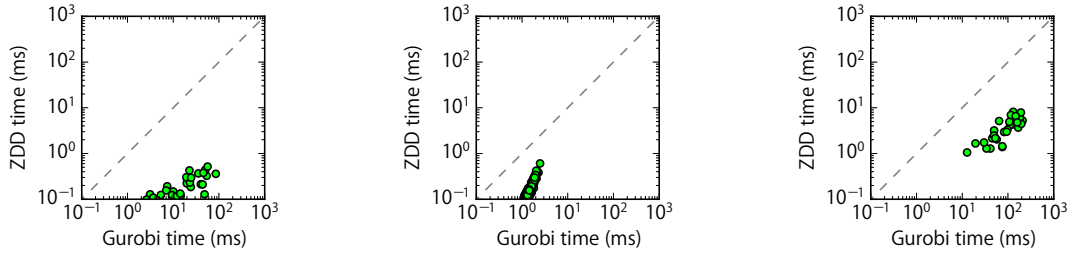
のノードに対して重みが小さい子ノードから順にたどるような深さ優先行きがけ順の順序を与えると，その順序のもとで EDU 抽出，文短縮，文短縮・抽出の各問題に対して，$\mathcal{T}$ を表現する ZDD のサイズは表 2 のようになることを示すことができる（証明略）．ここで $|R|$ は根候補集合のサイズ，$|R^*|$ は各内側の木の根候補集合のうち，要素数が最大のもののサイズである．また，木のノードの重みは，そのノードおよび子孫ノードのに含まれる根候補ノードを根とする木のサイズの最大値とする．根候補集合のサイズの最大値は N であるから $|R|, |R^*| \leq N$ であり，したがって任意の $\mathcal{T}$ に対してて ZDD のノード数は常に $O(N \log N)$ となる．この結果より，前節で説明した動的計画法は常に $O(NL \log N)$ で実行できることが分かる．図 3 は，図 1 の入力例に対応する $\mathcal{T}$ を表現する ZDD の例である．

6 ZDD の構築

提案する動的計画法を実行するためには，まず入力の木が与えられた時に $\mathcal{T}$ を表現する ZDD を構築する必要がある．詳細は省略するが，ZDD のトップダウン構築法 [湊 12] の変種を用いることで，この処理は ZDD のノード数に比例する時間で行うことができる．すなわち，ZDD の構築時間を考慮したとしても提案法は $O(NL \log N)$ で木刈込み問題を解くことができる．

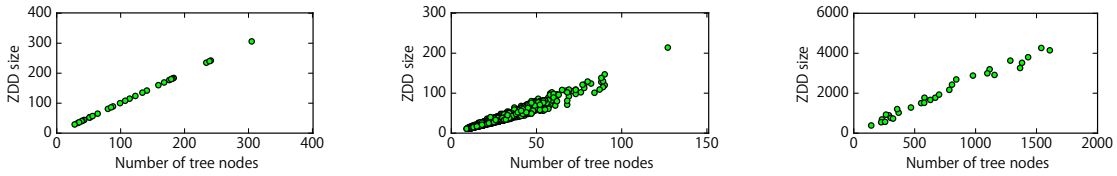
7 検証

3 種類の木刈込み問題のそれぞれについて提案法と ILP ソルバ (Gurobi 5.5.0) の性能を比較した，EDU 抽出，文短縮・抽出については，元論文で用いられていたのと同じ RST Discourse Treebank に含まれる要約用データセットを用いた．データセットは 30 文書からなる．EDU，語の重みも元論文と同じものを用いた．文短縮についても元論文で用いられた英語の分短縮用コーパス（約 1,300 文）を用いた．



(a) EDU 抽出　(b) 文短縮　(c) 文短縮・抽出

図 4: 提案法と Gurobi の速度比較



(a) EDU 抽出　(b) 文短縮　(c) 文短縮・抽出

図 5: 入力の木のノード数と ZDD のサイズの関係

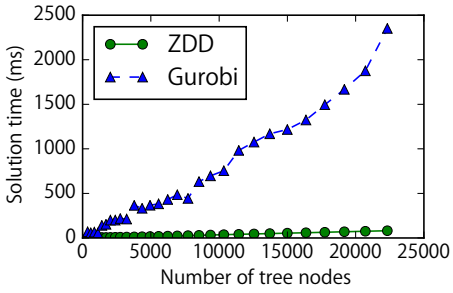


図 6: 入力の木のサイズを変化させたときの実行速度の変化

検証結果を図 4 に示す．図は提案法と Gurobi の実行時間の関係のプロットであり，1 点があるインスタンスでの提案法と Gurobi の実行時間を示す．点が点線より下にあるときに提案法が高速であったことを表す．図より，全てのインスタンスにおいて提案法がより高速であったことを示している．EDU 抽出，文短縮，文抽出と文短縮の 3 種類の問題に対し，それぞれ最大で 300, 10, 50 倍提案法のほうが高速であった．

図 5 は，それぞれ入力の木のノード数と ZDD のサイズの関係である．今回利用したデータセットでは ZDD のサイズは木のノード数に対してほぼ線形の関係にあることが分かる．EDU 抽出だけでなく文短縮および文短縮・抽出においても ZDD のサイズが線形に近い結果となったのは，入力の木のサイズが大きくなってもその木の根候補集合（文中に出現する動詞の集合）のサイズがあまり大きくならなかったためと考えられる．

図 6 は，入力の木のノード数を変化させたときの求解にかかる時間の変化である．入力の木としては，文短縮・抽出における 30 文書に対応する入力の木の根ノードをつなぐことで構築されるものを用いた．提案法では Gurobi と比べて入力の木のサイズが増加しても実行時間が抑えられていることが分かる．

8　おわりに

本稿では木刈込み問題に対する動的計画法アルゴリズムを提案した．提案法は入力の木のノード数によって定まる実行時間で厳密解を求めることができる．また，検証では提案法が実際に高速に動作することも確認した．今後は木刈込み以外の文書要約の定式化に対しても提案法の適用可能性を探る予定である．

参考文献

[Filippova 08] Filippova, K. and Strube, M.: Dependency tree based sentence compression, in *INLG'08* (2008)

[Filippova 13] Filippova, K. and Altun, Y.: Overcoming the Lack of Parallel Data in Sentence Compression, in *EMNLP'13* (2013)

[Hirao 13] Hirao, T., Yoshida, Y., Nishino, M., Yasuda, N., and Nagata, M.: Single-Document Summarization as a Tree Knapsack Problem, in *EMNLP'13* (2013)

[Kikuchi 14] Kikuchi, Y., Hirao, T., Takamura, H., Okumura, M., and Nagata, M.: Single document summarization based on nested tree structure, in *ACL'14* (2014)

[Minato 93] Minato, S.: Zero-suppressed BDDs for set manipulation in combinatorial problems, in *DAC'93* (1993)

[湊 12] 湊 真一：BDD/ZDD を用いたグラフ列挙索引化技法, オペレーションズ・リサーチ, Vol. 57, No. 11, pp. 597–603 (2012)