1B-3

頂点容量制約付き有向全域木パッキング問題に対するラグランジュ緩和に基づく列生成法

田中 勇真† 今堀 慎治‡ 柳浦 睦憲†

†名古屋大学大学院 情報科学研究科 計算機数理科学専攻 ‡名古屋大学大学院 工学研究科 計算理工学専攻

1 はじめに

本研究では, 頂点容量制約付き有向全域木パッキング問題を扱う. 入力として, 有向グラフ $G = (V, E)$, 根 $r \in V$, 各頂点 $i \in V$ に対する容量 $b_i \in \mathbb{R}_+$ ($\mathbb{R}_+$ は非負実数の集合), および2つの関数 $t : E \to \mathbb{R}_+$ と $h : E \to \mathbb{R}_+$ が与えられる. t と h はそれぞれ辺を利用したときに始点と終点に要求する消費量を表す. 便宜上, グラフ G 上の r に流入する有向全域木全ての集合を T_{all} とする. 木 $j \in T_{\mathrm{all}}$ において頂点 $i \in V$ から出る (に入る) 辺の集合を $\delta_j^+(i)$ ($\delta_j^-(i)$) と記す. すると, 木 $j \in T_{\mathrm{all}}$ の各頂点 $i \in V$ における消費量 a_{ij} は

$$a_{ij} = \sum_{e \in \delta_j^+(i)} t(e) + \sum_{e \in \delta_j^-(i)} h(e) \tag{1}$$

と定義される. (1) 式の右辺の1つ目 (2つ目) の項は, 木 j において頂点 i から出る (に入る) 全ての辺が i に要求する消費量の合計を表す. 頂点容量制約付き有向全域木パッキング問題は, $r \in V$ に流入する木の集合 $T \subseteq T_{\mathrm{all}}$ と, それぞれの木 $j \in T$ のパッキング回数 x_j を定め, 制約

$$\sum_{j \in T} a_{ij} x_j \le b_i, \quad \forall i \in V$$

の下で x_j の合計を最大化する問題である.

本問題は, NP 困難であることが証明されている [1]. 頂点が空間に埋め込まれ, 各辺 $e \in E$ の消費量 $t(e)$ と $h(e)$ が, 辺 e の両端点間のユークリッド距離のみに依存する関数に限定された場合でも NP 困難であることも証明されている.

近年, アドホックネットワークやセンサーネットワークといったネットワークが注目を集めるようになっているが, グラフパッキング問題にはそれらを背景にしたものが多数ある. 本問題も, センサーネットワークに重要な応用を持つ問題である. 関連研究としては [2] などがある.

我々は以前, 本問題に対して2段階の線形緩和に基づくアルゴリズムを提案した [3]. 1段階目にパッキングするのに必要な木の集合 T を生成し, 2段階目に1段階目に生成した各木 $j \in T$ のパッキング回数 x_j を決定する. 本研究では, ラグランジュ緩和を用いた新たな1段階目のアルゴリズムを提案する. 計算実験により, 提案アルゴリズムは以前のアルゴリズムや既存のアルゴリズムに比べて実用的に優れていることを示す.

†Yuma TANAKA and Mutsunori YAGIURA ‡Shinji IMAHORI
†Department of Computer Science and Mathematical Informatics, Graduate School of Information Science, Nagoya University
‡Department of Computational Science and Engineering, Graduate School of Engineering, Nagoya University

2 定式化

頂点容量制約付き有向全域木パッキング問題を整数計画問題として定式化すると次のようになる:

$$\begin{aligned} P(T_{\mathrm{all}}) \quad \max \quad & \sum_{j \in T_{\mathrm{all}}} x_j \\ \text{s.t.} \quad & \sum_{j \in T_{\mathrm{all}}} a_{ij} x_j \le b_i, \quad \forall i \in V \\ & x_j \ge 0, \qquad x_j \in \mathbb{Z}, \ \forall j \in T_{\mathrm{all}}. \end{aligned}$$

ここで, $\mathbb{Z}$ は整数集合を表し, a_{ij} は (1) で定義されたものである. $P(T_{\mathrm{all}})$ は全ての木 T_{all} に対する定式化であるが, T_{all} のサイズは指数的である. そこで, 一部の木集合 $T \subseteq T_{\mathrm{all}}$ に対する問題 $P(T)$ を考える. ただし, $P(T)$ は整数計画問題のままであり, 一般的に厳密な最適解を求めることは困難である. そのため, 本研究では $P(T)$ にラグランジュ緩和を適用する. $P(T)$ のラグランジュ緩和問題は次のようになる.

$$\begin{aligned} LR(T, \lambda) \quad \max \quad & \sum_{j \in T} x_j + \sum_{i \in V} \lambda_i \Big(b_i - \sum_{j \in T} a_{ij} x_j \Big) \\ & = \sum_{j \in T} c_j(\lambda) x_j + \sum_{i \in V} \lambda_i b_i, \\ \text{s.t.} \quad & x_j \in \{0, 1, \ldots, u_j\}, \quad \forall j \in T. \end{aligned}$$

ここで, $\lambda_i \ge 0$ は頂点 $i \in V$ に対するラグランジュ乗数であり, u_j は木 $j \in T$ のみをパッキングしたときの最大パッキング回数である (すなわち, $u_j = \min_{i \in V : a_{ij} > 0} \lfloor b_i / a_{ij} \rfloor$). また, 各木 $j \in T$ 対する $c_j(\lambda) = 1 - \sum_{i \in V} a_{ij} \lambda_i$ は相対コストと呼ばれるものである. 任意の $\lambda \ge 0$ 対して, 最適解 $x(\lambda)$ は次のように簡単に決定できる: 全ての木 $j \in T$ に対して, $c_j(\lambda) > 0$ ならば $x_j(\lambda) = u_j$, $c_j(\lambda) \le 0$ ならば $x_j(\lambda) = 0$ とする. また, $LR(T, \lambda)$ の最適値は $P(T)$ の上界となる.

表 1: 計算結果

Instance name	$\|V \setminus \{r\}\|$	$\|E\|$	$UB_{b.k.}$	提案アルゴリズム					以前のアルゴリズム [3]			SFIS
				$\|T\|$	UB	Obj.	Gap (%)	Time (s)	Obj.	Gap (%)	Time (s)	Obj.
rnd200-5-100000-h	200	1970	2602	1451	2603	2583	0.73	56.7	2152	17.29	149.8	1755
rnd200-5-100000-t	200	1970	2500	1291	2622	2500	0.00	52.8	1943	22.28	101.0	2302
rnd200-5-100000	200	1970	1411	1005	1412	1401	0.71	22.7	1173	16.87	70.3	1091
rnd200-50-100000-h	200	20030	2874	1525	2876	2857	0.59	76.5	2779	3.31	123.9	2374
rnd200-50-100000-t	200	20030	2867	1270	2868	2855	0.42	78.1	1920	33.03	100.9	2542
rnd200-50-100000	200	20030	1569	929	1570	1552	1.08	28.8	1299	17.21	48.2	1332

3 提案アルゴリズム

提案アルゴリズムの大まかな流れは次の通りである:

1. 初期木集合 T をランダムに生成する.
2. 現在の T に対する $LR(T, \lambda)$ の良いラグランジュ乗数 λ を得るために劣勾配法を適用する.
3. $LR(T, \lambda)$ の最適解 $x(\lambda)$ を求める.
4. 現在の T と得られた λ に対する $LR(T, \lambda)$ の最適値が改善するような木 $\tau \in T_{\mathrm{all}} \setminus T$ を生成し, 現在の T に新たに生成した木 τ を追加する.
5. 停止条件を満たしていないならば 2 に戻る.

2 行目の劣勾配法とは更新式によりラグランジュ乗数 λ を繰り返し更新する方法で, 良いラグランジュ乗数を得るための良く知られた発見的解法である. 我々は劣勾配法に対して高速化手法を提案し, 劣勾配法の 1 反復あたり約 2 から 4 倍の高速化に成功した. また, 過去の情報から劣勾配法に実際に使用する木の数を削減した.

4 行目のような木 τ の生成と追加を繰り返す方法は一般的に列生成法と呼ばれるものである. 我々は以前の研究 [3] で新たな木 τ を見つける多項式時間の手法を提案した. 本研究の提案アルゴリズムにおいてもその結果を利用している.

なお, ここでは詳しく解説しないが, 提案アルゴリズムの 2 段階目 (各木 $j \in T$ に対してパッキング回数 x_j を決定する) 部分は以前のアルゴリズムと同様の方法を用いている.

4 計算実験

問題例をランダムに生成し, 提案アルゴリズム, 以前のアルゴリズム [3], および既存のアルゴリズム [2] に対して計算実験を行った. その結果を表 1 に示す. 表 1 の最初の 3 列は, 問題名, (根を除いた) 頂点数, 辺数を表す. 列 "$UB_{b.k.}$" は既存の最良の上界を表す. 続く残りの列は, 本研究の提案アルゴリズム, 以前のアルゴリズム [3], 既存のアルゴリズム [2] を我々が新たに実装したもの (SFIS と表記する) の計算結果である. 列 "$|T|$" は提案アルゴリズムによって生成された木の数, 列 "UB" は提案アルゴリズムによって得られた最良の上界, 列 "Obj." は出力された解の目的関数値, 列 "Gap (%)" は $UB_{b.k.}$ と Obj. の間のギャップ (すなわち, $((UB_{b.k.} - Obj.)/UB_{b.k.}) \times 100\%$), 列 "Time (s)" は計算時間 (秒) を表す. なお, 比較の公平性から以前のアルゴリズムは提案アルゴリズムと同じ木の数を生成したときに停止させ, SFIS に 100 秒の時間制限を与えている.

表 1 より, 提案アルゴリズムは以前のアルゴリズムと既存のアルゴリズムよりも良い結果を得ていることが確認できる. 生成した木の数が同じであり, 計算時間が短いにも関わらず提案アルゴリズムは以前のアルゴリズムよりも良い目的関数値を得ている. また, $UB_{b.k.}$ と目的関数値のギャップは約 1%以下と非常に小さく, 1 つの問題例に対しては厳密な最適解を発見した.

5 まとめ

頂点容量制約付き有向全域木パッキング問題に対してラグランジュ緩和に基づくアルゴリズムを提案した. 計算実験により, 提案アルゴリズムは以前のアルゴリズムや既存のアルゴリズムよりも実用的に優れていることを示した.

参考文献

[1] S. Imahori, Y. Miyamoto, H. Hashimoto, Y. Kobayashi, M. Sasaki, and M. Yagiura. The complexity of the node capacitated in-tree packing problem. *Networks*, 59:13–21, 2012.

[2] M. Sasaki, T. Furuta, F. Ishizaki, and A. Suzuki. Multi-round topology construction in wireless sensor networks. In *Proceedings of the Asia-Pacific Symposium on Queueing Theory and Network Applications*, pages 377–384. 2007.

[3] Y. Tanaka, S. Imahori, M. Sasaki, and M. Yagiura. An LP-based heuristic algorithm for the node capacitated in-tree packing problem. *Computers & Operations Research*, 39:637–646, 2012.