Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT

Information Processing Letters 89 (2004) 181–185

Information Processing Letters

www.elsevier.com/locate/ipl

Partitioning of trees for minimizing height and cardinality

András Kovács [a,b], Tamás Kis [b,*]

[a] Budapest University of Technology, Magyar tudósok körútja 2/d, 1117 Budapest, Hungary
[b] Computer and Automation Research Institute, Kende utca 13–17, 1111 Budapest, Hungary

Received 11 February 2003; received in revised form 3 November 2003

Communicated by S. Albers

Keywords: Partitioning of trees; Graph algorithms; Analysis of algorithms; Computational complexity

1. Introduction

Partitioning a tree $T = (V, E)$ into q subtrees $P = \{ST_1, \dots, ST_q\}$ such that a given set of constraints is satisfied and a criterion is optimized constitutes a widely studied class of problems that has numerous applications, see, e.g., [1–5]. In this note we consider new criteria and provide polynomial time algorithms. Before presenting our results, we introduce the notation and terminology used throughout the paper.

In the sequel $T = (V, E)$ always denotes a rooted tree with *vertex-set* V, *edge-set* E, and *root* r. The *sons* of a vertex $v \in V$ will be denoted by $S(v)$, noting that $S(v) = \emptyset$ if and only if v is a leaf. Let $T(v)$ be the subtree of T rooted at v consisting of v and all vertices down to the leaves. The following definitions apply to T and also to all $T(v)$. $P = \{ST_1, \dots, ST_q\}$ is a *partitioning* of T if and only if

(a) each component ST_i is a subtree (connected subgraph) of T,
(b) the ST_i are disjoint, and
(c) the union of the vertex-sets $V(ST_i)$ of the ST_i equals V.

The *cardinality* of a partitioning P of T is defined as $q(P) = |P|$. Each ST_i is rooted at the vertex closest to r in T. The *root component* of P is the one containing r, and will be denoted by ST^r. The *root component weight* of P is the total weight of the vertices in ST^r, i.e., $rw(P) = \sum_{u \in ST^r} w(u)$. For any partitioning P of T, let T^P denote the rooted tree obtained from T by contracting each $ST_i \in P$ into a vertex. The *height $h(T)$ of a rooted tree T* is the maximum number of edges of paths having one end at the root. The *height $h(P)$ of a partitioning P* is the height of T^P. The *level $\ell_P(v)$ of a vertex $v \in V$* with respect to a partitioning P of T is the height of the *induced partitioning P'* of $T(v)$ consisting of those $ST_i \in P$ with $V(ST_i) \subseteq V(T(v))$ and also ST^v, where $ST^v = ST_j \cap T(v)$ and $v \in ST_j \in P$.

Given a non-negative weight function $w : V \to \mathbb{R}_+$ on the vertices of T and a constant W, we say that a partitioning $P = \{ST_1, \dots, ST_q\}$ of T satisfies the *knapsack constraint* if and only if

$$\sum_{v \in V(ST_i)} w(v) \leqslant W \quad \text{for every } ST_i \in P. \tag{1}$$

* Corresponding author.
E-mail addresses: akovacs@mit.bme.hu (A. Kovács), tamas.kis@sztaki.hu (T. Kis).

0020-0190/$ – see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.ipl.2003.11.004

We call a partitioning P of T *admissible* if it satisfies (1). Let $AP(T)$ denote the set of all admissible partitionings of T. We assume that $w(v) \leqslant W$ for each $v \in V$.

The problem of finding an admissible partitioning of minimum cardinality is solved by Kundu and Misra [4]. A generalization in which there are multiple weight functions on the vertices is analyzed and solved by Hamacher et al. [2]. For fixed q, the problem of minimizing (maximizing) the maximum (minimum) weight of a subtree with respect to a wide range of weight functions is solved by the shifting algorithm technique of Becker and Perl [1]. The same paper extends these results by considering additional constraints limiting the size or the height of a subtree. Maravalle et al. [5] considered problems involving dissimilarities in or between subtrees of a partitioning.

In this paper we present polynomial time algorithms for solving the following problems:

(i) *Minimum height partitioning*. Find an admissible partitioning of T of minimum height (Section 2).
(ii) *Minimum height versus minimum cardinality partitioning*. Determine the Pareto set of admissible partitionings with respect to height and cardinality (Section 3).

Notice that the second problem is of interest, since minimizing the height and the cardinality of a partitioning are conflicting objectives, as shown by the example in Fig. 1.

The above combination of optimization criteria may be of importance when designing telecommunication networks, or even when balancing assembly lines.

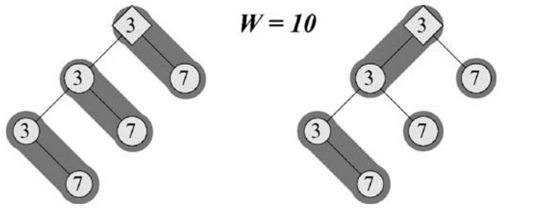

Fig. 1. A rooted tree with two partitionings: on the left one minimizing the cardinality, and on the right one minimizing the height. The weights are indicated on the vertices and the weight limit is 10.

2. The minimum height problem

In order to solve the minimum height problem, each vertex v of T will be labeled with a level $\ell(v)$ and also with the weight $rw(v)$ of the maximal subtree rooted at v spanned by those vertices $u \in T(v)$ with $\ell(u) = \ell(v)$. Namely, for each leaf v of T, let $\ell(v) = 0$ and $rw(v) = w(v)$. In the iterative step an unlabeled vertex v is chosen all of whose sons are labeled. Let $\ell_{\max} = \max_{u \in S(v)} \ell(u)$ denote the maximum $\ell(u)$ value among all sons u of v. If $w(v) + \sum_{u \in S(v),\ \ell(u)=\ell_{\max}} rw(u) \leqslant W$, then $\ell(v) = \ell_{\max}$ and $rw(v) = w(v) + \sum_{u \in S(v),\ \ell(u)=\ell_{\max}} rw(u)$. Otherwise, $\ell(v) = \ell_{\max} + 1$ and $rw(v) = w(v)$. The output of the algorithm is $P_A = \{ST_1, \ldots, ST_q\}$, where each ST_i is a maximal subtree of T with all vertices assigned the same ℓ value. The algorithm is illustrated in Fig. 2.

First note that P_A is an admissible partitioning as it satisfies the knapsack constraint (1) by construction. In order to show that P_A is of minimum height, let $\ell^*(v) = \min \ell_P(v)$ denote the minimum level of v over all $P \in AP(T)$. For each leaf v of T, $\ell^*(v) = 0$. Clearly, $\ell^*(r)$ equals the minimum height of an admissible partitioning of T. Observe that $\ell^*(u) \leqslant \ell^*(v)$ whenever $u \in S(v)$. It suffices to show the following:

Lemma 1. *$\ell(v) = \ell^*(v)$ for all $v \in V$.*

Proof. For each leaf v of T, $\ell^*(v) = 0 = \ell(v)$. Moreover, as P_A is an admissible partitioning, $\ell(v) \geqslant \ell^*(v)$, $v \in V$. Suppose there exists a non-leaf v with $\ell(v) > \ell^*(v)$ and let v be of maximum distance to r with this property. It follows that $\ell(u) = \ell^*(u)$ holds for all $u \in V(T(v)) \setminus \{v\}$. Letting $\ell_{\max} = \max_{u \in S(v)} \ell(u)$, we have $\ell_{\max} \leqslant \ell^*(v) < \ell(v)$. Consequently, $\ell_{\max} + 1 =$

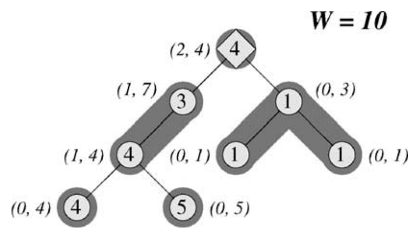

Fig. 2. A minimum-height partitioning determined by the algorithm. The labels $(\ell(u), rw(u))$ are indicated next to the vertices.

$\ell(v)$ holds, since the algorithm can assign at most $\ell_{\max}+1$ as the $\ell(v)$ value of v. This occurs if

$$w(v) + \sum_{u\in S(v),\ \ell(u)=\ell_{\max}} rw(u) > W.$$

Observe that the LHS equals $\sum_{u\in L\cup\{v\}} w(u)$, where $L=\{u\in T(v)\mid \ell(u)=\ell_{\max}\}$. Hence, there exists no admissible partitioning of T with a component containing $L\cup\{v\}$. If there were, then the component containing $L\cup\{v\}$ would have weight strictly greater than W, violating (1). To prove that $\ell^*(v) > \ell_{\max}$, consider an admissible partitioning Q of T with $\ell_Q(v)=\ell^*(v)$. Let $ST\in Q$ be the component containing v. Since Q is admissible and $v\in ST$, $L\not\subseteq ST$ holds. Hence, as $L\subseteq V(T(v))\setminus\{v\}$ by construction, there exists $u\in L\setminus ST$ such that $\ell_Q(v)>\ell_Q(u)$. Since $\ell^*(u)=\ell(u)=\ell_{\max}$ as $u\in L$, $\ell^*(v)=\ell_Q(v)>\ell_Q(u)\geqslant \ell^*(u)=\ell(u)=\ell_{\max}$. On the other hand, $\ell_{\max}\leqslant \ell^*(v)<\ell(v)=\ell_{\max}+1$ implies that $\ell^*(v)=\ell_{\max}$, a contradiction. □

Finally note that the running time of our algorithm is linear in the size of T.

3. Minimum height vs. minimum cardinality partitionings

In this section we present a polynomial time algorithm for determining the set of Pareto optimal partitionings of T with respect to three criteria: minimum height, minimum cardinality and minimum root component weight. The third criterion is necessary to make our iterative bottom-up algorithm work. Note that from such a set one can easily derive the Pareto set with respect to height and cardinality.

If P and Q are admissible partitionings of $T(v)$, we say that P *dominates* Q if and only if $q(P)\leqslant q(Q)$, $h(P)\leqslant h(Q)$ and $rw(P)\leqslant rw(Q)$. The set of Pareto optimal partitionings $PO(v)$ of a subtree $T(v)$ is a minimal subset (w.r.t. set inclusion) of $AP(T(v))$ such that each $Q\in AP(T(v))$ is dominated by some $P\in PO(v)$. Note that $T(v)$ may admit several different Pareto sets.

Clearly, for each leaf v of T, $PO(v)$ consists of only the trivial partitioning $\{v\}$. We will show that if v is not a leaf of T, $PO(v)$ can be constructed by combining partitionings chosen from the sets $PO(u)$, $u\in S(v)$. We start by a closer look to combining partitionings.

Any partitioning of $T(v)$ can be obtained by applying the following comb operator to suitable partitionings of the $T(u)$, $u\in S(v)$. Namely, let P_u be a partitioning of $T(u)$, $u\in S(v)$, and $K\subseteq S(v)$, then $P:=\mathrm{comb}(\{P_u\mid u\in S(v)\},K)$ is a partitioning of $T(v)$ consisting of all the components of all P_u except the root components of those P_u with $u\in K$, which together with v constitute the root component of P. See Fig. 3 for illustration.

The three parameters of $P=\mathrm{comb}(\{P_u\mid u\in S(v)\},K)$ can be derived as follows:

$$h(P)=\max\{\max\{h(P_u)\mid u\in K\},\ \max\{h(P_u)+1\mid u\in S(v)\setminus K\}\},\tag{2}$$

$$q(P)=\sum_{u\in S(v)} q(P_u)-|K|+1,\tag{3}$$

$$rw(P)=\sum_{u\in K} rw(P_u)+w(v).\tag{4}$$

A fundamental property of the comb operator is that it preserves dominance:

Lemma 2. *If for each $u\in S(v)$, P_u and Q_u are admissible partitionings of $T(u)$ such that P_u dominates Q_u, then $P=\mathrm{comb}(\{P_u\mid u\in S(v)\},K)$ dominates $Q=\mathrm{comb}(\{Q_u\mid u\in S(v)\},K)$, for any $K\subseteq S(v)$. Moreover, if Q is admissible, then so is P.*

Proof. Since P_u dominates Q_u for each $u\in S(v)$, we have $h(P_u)\leqslant h(Q_u)$, $q(P_u)\leqslant q(Q_u)$ and $rw(P_u)\leqslant rw(Q_u)$. Relating this to Eqs. (2)–(4), we obtain $h(P)\leqslant h(Q)$, $q(P)\leqslant q(Q)$ and $rw(P)\leqslant rw(Q)$. Hence, P dominates Q.

To finish the proof, observe that Q is admissible if and only if $rw(Q)\leqslant W$, as all Q_u are admissible by

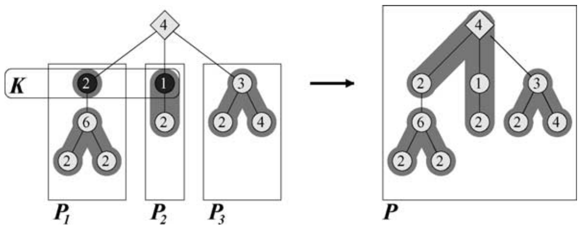

Fig. 3. The comb operator applied to selected partitionings of the sons of the root.

assumption. Since P dominates Q by the first part of the lemma, $rw(P) \leqslant rw(Q)$ holds, thus $rw(P) \leqslant W$. Hence, P is admissible too. □

Lemma 3. *Let Q be an admissible partitioning of $T(v)$. Then there exist $P_u \in PO(u)$, $\forall u \in S(v)$, and $K \subseteq S(v)$ such that $P := \mathrm{comb}(\{P_u \mid u \in S(v)\}, K)$ dominates Q.*

Proof. Letting $ST^v \in Q$ be the root component of Q, define $K := S(v) \cap ST^v$ and let Q_u be the partitioning induced on $T(u)$ by Q, $u \in S(v)$. By the definition of the sets $PO(u)$, for each Q_u there exists $P_u \in PO(u)$ such that P_u dominates Q_u. Applying Lemma 2 to $\{P_u \mid u \in S(v)\}$, $\{Q_u \mid u \in S(v)\}$ and K, we deduce that $P = \mathrm{comb}(\{P_u \mid u \in S(v)\}, K)$ is an admissible partitioning of $T(v)$ dominating Q. □

Consequently, $PO(v)$ can be constructed by finding appropriate combinations of partitionings chosen from the sets $PO(u)$, $u \in S(v)$. To facilitate the computation, let $D^h(v)$ be a minimal subset (w.r.t. set inclusion) of $AP(T(v))$ such that each $Q \in AP(T(v))$ with $h(Q) = h$ is dominated by some $P \in D^h(v)$. Notice that for each q there can be at most one $P \in D^h(v)$ with $q(P) = q$, since $P \in AP(T(v))$, $h(P) \leqslant h$ and $rw(P) = \min\{rw(Q) \mid Q \in AP(T(v)),\ q(Q) = q,\ h(Q) \leqslant h\}$ must hold. Moreover, we have the following:

Lemma 4. *Let P be arbitrary member of $D^h(v)$ and suppose $P = \mathrm{comb}(\{P_u \mid u \in S(v)\}, K)$, where $P_u \in PO(u)$, $u \in S(v)$ and $K \subseteq S(v)$. Then the following conditions hold*:

(i) *For each $u \in K$: $h(P_u) \leqslant h$, and $rw(P_u) = \min\{rw(Q_u) \mid \forall Q_u \in PO(u)$ with $h(Q_u) \leqslant h$ and $q(Q_u) = q(P_u)\}$.*
(ii) *For each $u \in S(v) \setminus K$: $h(P_u) \leqslant h - 1$ and $q(P_u) = \min\{q(Q_u) \mid Q_u \in PO(u)$ with $h(Q_u) \leqslant h - 1\}$.*

Proof. Part (i). Suppose there exists $u^* \in K$ with $h(P_{u^*}) > h$. Then, by Eq. (2), $h(P) \geqslant h(P_{u^*}) > h$, a contradiction. Now, suppose there exists $Q_{u^*} \in PO(u^*)$ such that $h(Q_{u^*}) \leqslant h$, $q(Q_{u^*}) = q(P_{u^*})$ and $rw(Q_{u^*}) < rw(P_{u^*})$, then $P' := \mathrm{comb}(\{P_u \mid u \in S(v) \setminus \{u^*\}\} \cup \{Q_{u^*}\}, K)$ strictly dominates P, contrary to the definition of $D^h(v)$.

Part (ii). Similar to part (i). □

The results above suggest the following method for constructing $D^h(v)$: define for each son $u \in S(v)$ the set $F(u)$ consisting of two types of triples:

(i) for each q the triple (if it exists) $(P, q(P) - 1, rw(P))$, where $P \in PO(u)$ satisfies $h(P) \leqslant h$, $q(P) = q$ and $rw(P)$ smallest possible, and
(ii) the triple (if it exists) $(P, q(P), 0)$, where $P \in PO(u)$ satisfies $h(P) \leqslant h - 1$ and $q(P)$ smallest possible.

Now, for each $q = 1, \ldots, |V(T(v))| - 1$ determine $c_q := \min\{\sum_{u \in S(v)} rw_u \mid \forall (P_u, q_u, rw_u) \in F(u),\ u \in S(v)$, with $\sum_{u \in S(v)} q_u = q\}$. Notice that $c_q = \infty$ if and only if some $F(u)$ is empty or there exists no selection of triples with $\sum q_u = q$. If $c_q < \infty$, any optimal solution $\{(P_u, q_u, rw_u) \mid u \in S(v)\}$ induces a partitioning $P := \mathrm{comb}(\{P_u \mid u \in S(v)\}, K)$ of $T(v)$, where $K = \{u \in S(v) \mid rw_u > 0\}$, with $h(P) \leqslant h$, $q(P) = (\sum_{u \in S(v)} q_u) + 1$ and $rw(P) = (\sum_{u \in S(v)} rw_u) + w(v)$ and satisfying parts (i) and (ii) of Lemma 4. Consequently, P is admissible if and only if $c_q \leqslant W - w(v)$. Hence, the admissible partitionings corresponding to the non-dominated (q, c_q) pairs constitute $D^h(v)$.

To compute the c_q assume that $S(v) = \{u_1, \ldots, u_d\}$ and fill in a $d \times (|V(T(v))| - 1)$ table c_{kq} as follows: if $k = 1$, let $c_{1q} = rw_1$ if there exists $(P_1, q_1, rw_1) \in F(u_1)$ with $q_1 = q$, and ∞ otherwise; if $2 \leqslant k \leqslant d$, let $c_{kq} = \min\{c_{k-1,q-q_k} + rw_k \mid (P_k, q_k, rw_k) \in F(u_k), q_k < q\}$. One may verify that $c_{dq} = c_q$ holds for each $q = 1, \ldots, |V(T(v))| - 1$.

Since the dominance relation is transitive, we have the following:

Lemma 5. *Let $D(v)$ be a minimal subset (w.r.t. set inclusion) of $\bigcup_h D^h(v)$ such that each $Q \in \bigcup_h D^h(v)$ is dominated by some $P \in D(v)$. Then $D(v)$ is a Pareto optimal set of partitionings of $T(v)$.*

Consequently, we will choose $PO(v) = D(v)$ in the following.

Now we are ready to present our algorithm for finding all the sets $PO(v)$, $v \in V(T)$. For each leaf

v of T, $PO(v)$ consists of only $\{v\}$. In the iterative step, the algorithm selects some vertex v such that $PO(v)$ is not computed yet, but $PO(u)$ is known for all $u \in S(v)$. To compute $PO(v)$, first determine the sets $D^h(v)$ for each reasonable h (see above), then drop those members of $D := \bigcup_h D^h(v)$ dominated by some other member (break ties arbitrarily).

Concerning the running time, the dynamic program has time complexity $\mathrm{O}(d_v n^2)$, where $n = |V| \geqslant |V(T(v))| \geqslant |F(u)|$, and $d_v = |S(v)|$. Thus $PO(v)$ can be determined in $\mathrm{O}(d_v n^3)$ time by varying h between 1 and $|V(T(v))|$. The entire algorithm terminates in $\mathrm{O}(n^4)$ time.

Acknowledgements

This work has been supported by the grant NRDP 2/040/2001. The authors thank an anonymous referee for comments that helped to significantly improve the presentation.

References

[1] R.I. Becker, Y. Perl, The shifting algorithm technique for the partitioning of trees, Discrete Appl. Math. 62 (1995) 15–34.
[2] A. Hamacher, W. Hochstättler, C. Moll, Tree partitioning under constraints—clustering for vehicle routing problems, Discrete Appl. Math. 99 (2000) 55–69.
[3] D.S. Johnson, K.A. Niemi, On knapsacks, partitions, and a new dynamic programming technique for trees, Math. Oper. Res. 8 (1) (1983) 1–14.
[4] S. Kundu, J. Misra, A linear tree partitioning algorithm, SIAM J. Comput. 6 (1977) 151–154.
[5] M. Maravalle, B. Simeone, R. Naldini, Clustering on trees, Comput. Statist. Data Anal. 24 (1997) 217–234.