Discrete Applied Mathematics 15 (1986) 135-144
North-Holland

GREEDY CONCEPTS FOR NETWORK FLOW PROBLEMS

Wolfgang W. BEIN* and Peter BRUCKER

Fachbereich Mathematik/Informatik, Universität Osnabrück, Postfach 4469, 4500 Osnabrück, West Germany

Received 16 April 1986

We provide a general theoretical framework to prove greedy results for the optimal cost flow problem. We also draw the border line to problems where greedy is not valid. In some cases polymatroid explanations can be given for the greedy behaviour.

1. Introduction

A directed graph $G=(V,E)$ is given by a finite vertex set V and an edge set $E\subset V\times V$. We choose two distinct vertices $s\in V$ and $t\in V$ that we consider as source and sink respectively.

A path (from s to t) is a sequence of distinct edges

$$(s,x_1),(x_1,x_2),\dots,(x_{r-1},x_r),(x_r,t) \text{ for some } r.$$

Denote by P the set of all such paths. For an edge $e\in E$ let $P_e\subset P$ be the set of paths passing through this edge.

With every edge $e\in E$ we associate a nonnegative integer capacity b_e and an arbitrary positive cost c_e. For a path $p\in P$ the capacity and cost is defined by

$$b_p=\min_{e\in p} b_e, \tag{1}$$

$$c_p=\sum_{e\in p} c_e, \tag{2}$$

respectively. Then the problem of finding a cost optimal flow can be defined as follows.

$$\begin{aligned}&\max \sum_{p\in P} c_p x_p,\\ &\text{subject to } \sum_{p\in P_e} x_p\le b_e \quad \text{all } e\in E,\\ &\qquad x_p\ge 0, \text{ integer.}\end{aligned} \tag{3}$$

In this paper series-parallel networks will play an important role. A (two terminal)

*Current address: University of California, Computer Science Div., Berkeley, CA 94720, USA.

0166-218X/86/$3.50

series-parallel graph is a graph with source and sink, which is defined as follows:

(i) A single edge (u, v) is a series-parallel graph with source u and sink v.

(ii) If S_1 and S_2 are series-parallel graphs, so is the graph obtained by either of the following operations:

(a) Parallel composition: identify the source of S_1 with the source of S_2 and the sink of S_1 with the sink of S_2.

(b) Series composition: identify the sink of S_1 with the source of S_2.

The construction process of series-parallel graphs along their recursive definition may be represented by binary trees which are called decomposition trees. In a decomposition tree the edges of the graph are represented by the leaves of the tree. The inner nodes of the tree are labelled by S indicating a series composition or P indicating a parallel composition. Furthermore each subtree in the decomposition tree corresponds to a series-parallel subgraph.

In Fig. 1 an example of a series-parallel graph together with its decomposition tree is shown. Another example for a series-parallel graph is an intree with additional edges from a source to all leaves. Its decomposition tree has the property that in each series composition at least one of the subgraphs is a single edge (see Fig. 2). A third example are outerplanar graphs, which have the property that there are single edges in parallel compositions rather than series compositions. Outerplanar graphs have been dealt with in [3].

Valdes, Tarjan and Lawler [10] gave a linear algorithm to check whether a given graph is series-parallel and to construct its decomposition tree in that case. For that reason we assume that series-parallel graphs are given by their decomposition tree.

Bein, Brucker and Tamir [2] have shown that if the graph is series parallel problem (3) can be solved by the greedy algorithm which in this case is identical with

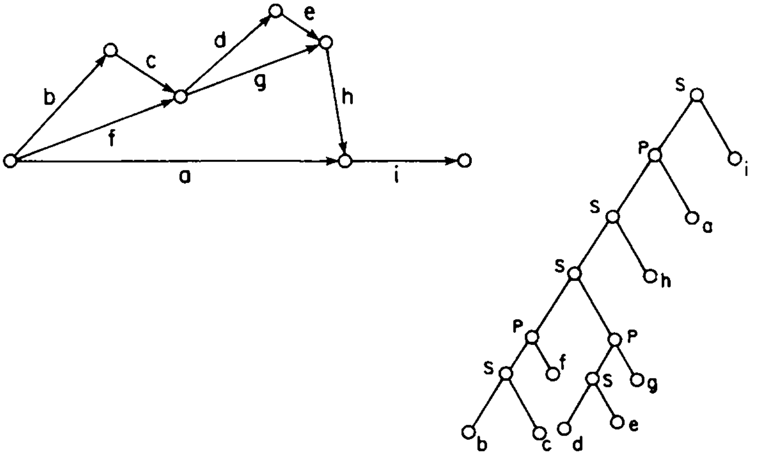

Fig. 1.

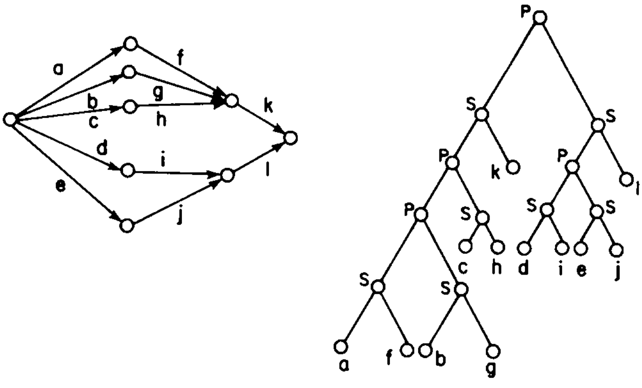

Fig. 2.

the augmenting path method. In this paper the same result will be derived by demonstrating that series composition and parallel composition preserves the property that the problem can be solved by the greedy algorithm. This leads to an algorithm which is different from the augmenting path method. The results are derived within a more general framework of linear programming interesting in its own right which will be discussed in the next section. Other linear programs for which greedy algorithms succeed are described in an excellent survey paper by Hoffman [8].

Applications of these concepts to tree structures are discussed in the last section. Furthermore we also show that these structures are polymatroidal in contrast to the series-parallel case.

2. Constructions preserving greedy

Let $B=\{1,\dots,r\}$ be a finite set. We consider the following linear program

$$\begin{aligned} &\max \sum_{i\in B} c_i x_i, \\ &\text{subject to} \sum_{i\in S} x_i \le b(S) \quad \forall S\in 2^B, \\ &\qquad\qquad x_i \ge 0, \text{ integer} \end{aligned} \tag{4}$$

where c_i are positive real numbers and b is an integer-valued set function on 2^B.

In almost all applications there will only be restrictions for a comparatively small subset system $\mathscr{A}\subset 2^B$ with $\bigcup_{T\in\mathscr{A}} T=B$. In this case it is possible to formally extend b to the entire power set. We define this 'minimal extension' by

$$\bar{b}(S)=\min\left\{ \sum_{T\in\mathscr{B}} b(T) \,\middle|\, \mathscr{B}\subset\mathscr{A}, S\subset \bigcup_{T\in\mathscr{B}} T\right\}$$

and consider (4) with right side $\bar{b}$. Denote the feasible region of this program by $\bar{F}$ and let F be the feasible region of the original program.

The following lemma shows that both problems are equivalent.

Lemma 1. $\bar{F}=F$.

Proof. Let $x \in F$ and $S \in 2^B$. Then there is a collection $\mathscr{B} \subset \mathscr{S}$ such that

$$\bar{b}(S)=\sum_{T \in \mathscr{B}} b(T) \quad \text{and} \quad S \subset \bigcup_{T \in \mathscr{B}} T=S'.$$

We have

$$\sum_{i \in S} x_i \leq \sum_{i \in S'} x_i \leq \sum_{T \in \mathscr{B}} b(T)=\bar{b}(S),$$

so $x \in \bar{F}$.

Conversely let $x \in \bar{F}$. Then let $S \in \mathscr{S}$ and we have

$$\sum_{i \in S} x_i \leq \bar{b}(S) \leq b(S)$$

and so $x \in F$. $\square$

Now we consider Algorithm 1 which is called the greedy algorithm. In this algorithm F is the feasible set of (4) and e_i denotes the i-th unit vector.

Algorithm 1

1. **For** all $i \in B$ **do** $x_i := 0$;
2. **While** there exists an $i \in B$ with $x+e_i \in F$ **do**
   Begin
3. $I := \{i \in B \mid x+e_i \in F\}$;
4. $c_j := \max\{c_i \mid i \in I\}$;
5. $\varepsilon := \max\{\alpha \mid x+\alpha e_j \in F\}$;
6. $x := x+\varepsilon e_j$
   End

Unfortunately the greedy algorithm which can be regarded as a combinatorial analogon to gradient methods in continuous optimization does not always solve (4). We are interested in constructing cases in which greedy is valid.

Consider the following two distinct problems.

$$\begin{aligned} &\max \sum_{i \in P} c_i x_i, \\ &\text{s.t.} \sum_{i \in S} x_i \leq b_1(S) \quad \forall S \in 2^P, \\ &\quad x_i \geq 0, \text{ integer} \end{aligned} \tag{5}$$

and

$$\begin{aligned} &\max \sum_{j\in Q} d_j y_j, \\ &\text{s.t.} \sum_{j\in T} y_j \le b_2(T) \quad \forall T\in 2^Q, \\ &\qquad y_j \ge 0, \text{ integer,} \end{aligned} \tag{6}$$

where $P=\{1,\dots,r\}$ and $Q=\{1,\dots,s\}$.

Furthermore, let $\bar{x}$ and $\bar{y}$ be optimal greedy solutions. We then set

$$t^* = \min\left\{ \sum_{i\in P} \bar{x}_i, \sum_{j\in Q} \bar{y}_j \right\} \tag{7}$$

and consider the new problems (8) and (9) which correspond to parallel composition and series composition in connection with network flow problems.

$$\begin{aligned} &\max \sum_{i\in P} c_i x_i + \sum_{j\in Q} d_j y_j, \\ &\text{s.t.} \sum_{i\in S} x_i \le b_1(S) \quad \forall S\in 2^P, \\ &\qquad \sum_{j\in T} y_j \le b_2(T) \quad \forall T\in 2^Q, \\ &\qquad x_i, y_j \ge 0, \text{ integer.} \end{aligned} \tag{8}$$

$$\begin{aligned} &\max \sum_{i\in P, j\in Q} (c_i + d_j) z_{ij}, \\ &\text{s.t.} \sum_{i\in S, j\in Q} z_{ij} \le b_1(S) \quad \forall S\in 2^P, \\ &\qquad \sum_{j\in T, i\in P} z_{ij} \le b_2(T) \quad \forall T\in 2^Q, \\ &\qquad \sum_{i\in P, j\in Q} z_{ij} \le t^*, \\ &\qquad z_{ij} \ge 0, \text{ integer.} \end{aligned} \tag{9}$$

Lemma 2. *Greedy is valid for* (8), *if greedy is valid for* (5) *and* (6).

Proof. Greedy solution (x^*, y^*) of (8) induces greedy solutions x^* for (5) and y^* for (6). Now assume that there exists a better solution (x, y) for (8) than (x^*, y^*). Then $\sum_{i\in P} c_i x_i > \sum_{i\in P} c_i x_i^*$ or $\sum_{j\in Q} d_j y_j > \sum_{j\in Q} d_j y_j^*$ which contradicts to the fact that x^* and y^* are greedy solutions at (5) and (6) respectively. □

Lemma 3. *Greedy is valid for* (9), *if greedy is valid for* (5) *and* (6) *with* $\sum_{i\in P} x_i \le t$ *and* $\sum_{i\in Q} y_i \le t$ *for all* $t \le t^*$.

Proof. Substitute

$$x_i = \sum_{j \in Q} z_{ij}, \qquad y_j = \sum_{i \in P} z_{ij}. \tag{10}$$

For the objective function of (9) we get

$$\begin{aligned} \sum_{i \in P, j \in Q} (c_i + d_j) z_{ij} &= \sum_{i \in P} \sum_{j \in Q} c_i z_{ij} + \sum_{j \in Q} \sum_{i \in P} d_j z_{ij} \\ &= \sum_{i \in P} c_i \sum_{j \in Q} z_{ij} + \sum_{j \in Q} d_j \sum_{i \in P} z_{ij} \\ &= \sum_{i \in P} c_i x_i + \sum_{j \in Q} d_j y_j. \end{aligned} \tag{11}$$

Thus (9) may be written

$$\begin{aligned} \max\ & \sum_{i \in P} c_i x_i + \sum_{j \in Q} d_j y_j, \\ \text{s.t.}\ & \sum_{i \in S} x_i \le b_1(S) \quad \forall S \in 2^P, \\ & \sum_{i \in P} x_i \le t, \\ & \sum_{j \in T} y_j \le b_2(T) \quad \forall T \in 2^Q, \\ & \sum_{j \in Q} y_j \le t, \\ & x_i, y_j \ge 0, \text{ integer.} \end{aligned} \tag{12}$$

Without loss of generality we assume that $c_i > c_{i+1}$ for $i = 1, \ldots, r-1$, $d_j > d_{j+1}$ for $j = 1, \ldots, s-1$ and that $c_{ij} := c_i + d_j$ are pairwise distinct. These assumptions are not essential for the proof. However, due to the fact that now the greedy solutions are unique the description of the proof is much easier.

For every solution of (12) with $\sum_{i \in P} \bar{x}_i = \sum_{j \in Q} \bar{y}_j = t$ we can find a solution $\bar{z}$ satisfying (10) by applying Algorithm 2. Note that Algorithm 2 is the greedy algorithm applied to 10.

Algorithm 2

1. $i := 1$; $j := 1$;
2. **While** $i \le r$ and $j \le s$ **do**
   Begin
3. **If** $x_i < y_j$ **then Begin**
4. $z_{ij} := x_i$;
5. $i := i + 1$;
6. $y_j := y_j - x_i$
   End
7. **Else if** $x_i > y_j$ **then Begin**
8. $z_{ij} := y_j$;
9. $j := j + 1$;

```
                $x_i := x_i - y_j$
                End
        Else if $x_i = y_j$ then Begin
                $z_{ij} := x_i$;
                $i := i+1$; $j := j+1$
                End
    End
```

Furthermore because of (11) the value of the objective functions for $\bar{z}$ and $\bar{x}$, $\bar{y}$ are identical.

Denote by x^*, y^* the greedy solution of (12). x^*, y^* solves (12) optimally and we have $\sum_{i \in P} x_i^* = \sum_{j \in Q} y_j^*$. Let z^* be the corresponding solution for (9) that is obtained by solving the transportation problem (10) with Algorithm 2. Then this solution is optimal for (9) because if there was a better solution for (9) we would have had a better solution for (12) using the substitution (10).

All we still have to show is that z^* is the solution that the greedy algorithm applied to (9) provides. Let $\bar{z}$ be the greedy solution of (9). We have to show $\bar{z} = z^*$. We show this inductively along the construction of z^* by Algorithm 2.

Assume that $z_{ij}^* = \bar{z}_{ij}$ for all $i = 1, \dots, i_0$, $j = 1, \dots, j_0$. The next index pair is either (i_0+1, j_0), (i_0, j_0+1) or (i_0+1, j_0+1). For brevity we only consider the case (i_0, j_0+1). Assume that $\bar{z}_{i_0 j_0+1} \neq z_{i_0 j_0+1}^*$. Then $\bar{z}_{i_0 j_0+1} > z_{i_0 j_0+1}^*$ because otherwise $\bar{z}$ was not constructed by greedy. Now one of the variables $x_{i_0}^*$ or $y_{j_0+1}^*$ is going to be saturated.

Case 1: *$y_{j_0+1}^*$ is being saturated.* Then we have

$$y_{j_0+1}^* = z_{i_0 j_0+1}^* < \bar{z}_{i_0 j_0+1} \leq \sum_{i \in P} \bar{z}_{i\, j_0+1} = \bar{y}_{j_0+1}.$$

This is a contradiction to the fact that y^* is a greedy solution for (12).

Case 2: *$x_{i_0}^*$ is being saturated.* Then we have $x_{i_0}^* = \sum_{j=1}^{j_0} z_{i_0 j}^* + z_{i_0 j_0+1}^*$ due to Algorithm 2 and

$$\bar{x}_{i_0} = \sum_{j=1}^{j_0} \bar{z}_{i_0 j} + \sum_{j=j_0+1}^{r} \bar{z}_{i_0 j} \geq \sum_{j=1}^{j_0} z_{i_0}^* + \bar{z}_{i_0 j_0+1}.$$

Thus $x_{i_0}^* < \bar{x}_{i_0}$ – a similar contradiction. □

If we apply Lemmas 1 and 2 to the optimal flow problem for series-parallel graphs we get

Theorem 1. *The greedy algorithm solves the optimal flow problem for series-parallel graphs.*

Proof. It is evident that greedy is correct for a single edge. Inductively this property is preserved by either parallel or series compositions due to Lemmas 2 and 3. □

The following theorem which can be proved in an analogous manner as a corresponding theorem in Bein, Brucker and Tamir [2] shows that series-parallel graphs are the most general class for which greedy is valid.

Theorem 2. *G is a series-parallel graph if and only if for every set of nonnegative costs c_e and nonnegative integer capacities b_e greedy solves the optimal flow problem.*

The greedy algorithm for series-parallel graphs can be implemented in such a way that its complexity is $O(|V|^2)$ as was shown in [2]. However the proof of Theorem 1 gives rise to a different algorithm with computational advantages. First the optimal solutions are computed for all leaves of the decomposition tree T, i.e., the single edges of the series-parallel graph. We do this by just filling them to their capacity. Then let T_1 and T_2 be two subtrees of T with common father v. Assume that optimal solutions x, y have already been computed for the corresponding subgraphs G_1 and G_2. If v is a P-node, then we get an optimal solution to the composed graph by just merging x and y according their c-values.

In the case of an S-node all we have to do is to solve the transportation problem (10) in a greedy manner.

In detail this algorithm is described in [2]. While this approach along the decomposition tree still has complexity $O(|V|^2)$ for series-parallel graphs, for trees an implementation with $O(|V| \log |V|)$ complexity is possible by the use of mergeable heaps (see [5]).

For the even more special class of problems with nested constraints a linear time algorithm is possible (see [4]).

3. Polymatroidal aspects

The feasible set of (4) is called a polymatroid if the set function $b: 2^B \to \mathbb{R}$ has the following properties:

(i) $b(\emptyset) \geq 0$ (b is nonnegative).
(ii) $S \subset T \subset B$ implies $b(S) \leq b(T)$ (b is nondecreasing). (13)
(iii) $S, T \subset B$ implies $b(S) + b(T) \geq b(S \cap T) + b(S \cup T)$ (b is submodular).

Furthermore it is well known (see e.g. [6], [9]) that the following theorem holds:

Theorem 3. *The greedy algorithm solves* (4) *correctly for arbitrary non-negative costs c_i if and only if the feasible region F of* (4) *is a polymatroid.*

However, this does not imply that the set of feasible solutions in (3) for series-parallel graphs is a polymatroid. The reason is that the vector (c_p) in (3) is of the special form

$$(c_p) = A(c_e) \tag{14}$$

where A is the path-arc incidence matrix of the graph and thus it is not necessary that for each (c_p) there is a solution (c_e) of (14). An example for such a situation is given below:

Example. Consider the graphs of Figs. 3 and 4 where all capacities are 1. The graph of Fig. 3 is series-parallel. For the graph of Fig. 4 greedy fails to generate an optimal solution if applied to the path costs induced by the edge costs given. However, the set of feasible solutions is identical for both graphs.

That the set of feasible solutions in (3) for the graph in Fig. 3 is not a polymatroid may be also derived from the fact that $p_1 + p_2 = \{e_1, e_2\} + \{e_3, e_4\}$ is a basis, but $p_3 = \{e_3, e_2\}$ cannot be augmented from $p_1 + p_2$.

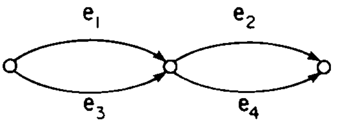

Fig. 3.

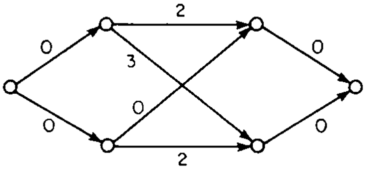

Fig. 4.

For trees, however, (14) can be solved for an arbitrary vector (c_p). Thus the set of feasible solutions in (3) for a tree is a polymatroid.

4. Concluding remarks

Our concept demonstrates that not only the set of feasible solutions determines whether greedy is valid or not but also the structure of the objective function is of importance.

This result is in line with different concepts of Barnes and Hoffman [1] and Gilmore and Gomory [7] who have identified various objective functions for which greedy is optimal in bipartite flow problems.

Note that we used these ideas in Algorithm 2.

It seems that the interrelation between objective function and the set of feasible solution is well worth for further investigation.

Acknowledgements

The authors are very grateful to Alan Hoffman for pointing out an error in the previous formulation of Lemma 3. They would also like to express their sincere thanks to the referees for their useful comments which improved the presentation of the paper.

References

[1] E.R. Barnes and A.J. Hoffman, On transportation problems with upper bounds on leading rectangles, SIAM J. Algebraic Discrete Methods 3 (1985) 487–496.

[2] W.W. Bein, P. Brucker and A. Tamir, Minimum cost flow algorithms for series-parallel networks, Discrete Appl. Math. 10 (1985) 117–124.

[3] M.W. Bern, E.L. Lawler and A.L. Wong, Linear-time computation of optimal subgraphs of decomposible graphs, University of California, Berkeley, Working paper, March 1985.

[4] P. Brucker, Network flows in trees and knapsack problems with nested constraints, in: H.J. Schneider and H. Göttler, eds., Proc. 8th Conf. Graphtheoretic Concepts in Computer Science. (Hanser, München, 1982) 25–35.

[5] P. Brucker, An $O(n \log n)$-algorithm for the minimum cost flow problem in trees, in: G. Hammer and D. Pallaschke, eds., Topics in Operations Research and Mathematical Economics (Springer, Berlin, 1984) 299–306.

[6] A. Federgruen and H. Groenevelt, The greedy procedure for resource allocation problems: necessary and sufficient conditions for optimality, Columbia University, Working paper, July 1983.

[7] P.C. Gilmore and R.E. Gomory, Sequencing a one state-variable machine: A solvable case of the traveling salesman problem, Operations Research 12 (1964) 655–679.

[8] A.J. Hoffman, On greedy algorithms that succeed, in: I. Anderson, ed., Surveys in Combinatorics 1985 (Cambridge University Press, 1985) 97–112.

[9] L. Lovász, Submodular functions and convexity, in: A. Bachem, M. Grötschel and B. Korte, eds., Mathematical Programming (Springer, Berlin, 1982) 235–257.

[10] J. Valdes, R.E. Tarjan and E.L. Lawler, The recognition of series-parallel diagraphs, SIAM J. Comput. 11 (1982) 298–313.