OR Spektrum (1994) 15:181–188

OR Spektrum

Some recent results in the analysis of greedy algorithms for assignment problems*

Ulrich Faigle

Department of Applied Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

Received 24 September 1993 / Accepted 2 November 1993

Abstract. We survey some recent developments in the analysis of greedy algorithms for assignment and transportation problems. We focus on the linear programming model for matroids and linear assignment problems with Monge property, on general linear programs, probabilistic analysis for linear assignment and makespan minimization, and on-line algorithms for linear and non-linear assignment problems.

Zusammenfassung. Einige neuere Entwicklungen auf dem Gebiet der Analyse von Greedy-Algorithmen für Transport- und Zuordnungsprobleme werden übersichtsartig dargestellt. Das Hauptinteresse gilt linearen Programmiermodellen für Matroide und lineare Zuordnungsprobleme mit Monge-Eigenschaft, allgemeinen linearen Programmen, der probabilistischen Analyse von linearen Zuordnungen und der Makespan-Minimierung sowie On-Line-Algorithmen für lineare und nichtlineare Zuordnungsprobleme.

Key words: Greedy algorithms, assignment problems, Monge property, job scheduling

Schlüsselwörter: Greedy-Algorithmus, Zuordnungsproblem, Monge-Eigenschaft, Job-Shop-Anordnung

1. Introduction

The algorithmic idea of constructing the desired object "greedily", i.e., according to local criteria of optimal improvement, is one of the oldest and most natural principles in the design of algorithms. Many heuristics are greedy algorithms and practical experience shows that they usually perform quite well. On the more theoretical side, already Fibonacci proved the optimality of the greedy algorithm for the representation of a rational number by unit fractions (cf. Erdös and Graham (1980)). Best known in discrete optimization is perhaps the optimality of the greedy algorithm for matroids, whose discovery goes back to Boruvka (1926). For the problem of finding an optimal weighted spanning tree in a graph, it also goes under the name of *Kruskal's algorithm.*

Especially the greedy algorithm for matroids has received a lot of attention after Edmonds (1970) pointed out its potential for combinatorial optimization within the setting of linear programming. Various generalizations of Edmonds' approach have been introduced (e.g., *generalized polymatroids* by Frank and Tardos (1988) or *submodular systems* by Fujishige (1990), which can be viewed as matroids on ordered ground sets (cf. Faigle (1987). Relaxation of the independence axioms for matroids has led to *greedoids* (cf. Korte et al. (1991)). We do not want to survey these developments here but refer the interested reader to the cited literature.

Far from trying to cover even approximately all algorithms that may justly be termed "greedy" and their applications in discrete optimization, we concentrate here on some recent results on some greedy algorithms for some assignment problems. The assignment problem with a linear objective function is a weighted graph matching problem. Changing the type of objective function yields the problems of *makespan minimization* and *loss minimization* respectively for job shop problems.

The *linear assignment problem* can generally not be solved by the matroid greedy algorithm. Hoffman (1963), however, exhibited special cases, where a simple direct algorithm works. Namely, when the weights have the so-called *Monge property.* Hoffman's result has been generalized in various directions. Queyranne et al. (1993) were able to exhibit a general common framework for both Edmonds' and Hoffman's greedy algorithms. We outline the framework and its connections with matroids and greedoid theory in Sect. 2 and 3. In Sect. 4, we discuss general linear programs that can be solved by the greedy algorithm. Hoffman et al. (1985) had observed a Monge-type property of the restriction matrix in the (0, 1)-case. Yet, it appears difficult to fit this property into the

* This is an expanded version of a survey lecture presented at the DGOR/NSOR Symposium, Amsterdam, 1993

aforementioned context. Indeed, the analogous property for general non-negative matrices is even less clearly coupled with Monge matrices. Combined with series-parallel decomposition, this approach yields the network greedy algorithm of Bein and Brucker (1986), for example.

Section 5 surveys some results on the probabilistic analysis of greedy algorithms for assignment problems. While most of these results refer to the estimation of the *average-case* performance, Aronson et al. (1993) are able to show that randomization of the greedy algorithm for matching problems helps on concrete graphs. We consider on-line problems in Sect. 6. On-line problems constitute a field where the use of the greedy principle lies directly at hand as the performance of an on-line algorithm is measured by its relative performance after each step. Often a greedy algorithm is very good if not optimal (e.g., loss minimization). A surprising result of Bartal et al. (1992) shows that the greedy algorithm for makespan minimization for job shop problems can measurably be beaten when one passes from pure list scheduling to *balanced* list scheduling. The situation is even worse for the greedy algorithm for the linear assignment problem, where the performance is an exponential factor off the on-line optimum (Kalyanasundaram and Pruhs (1993)).

2. Matroids and generalizations

Let $S=\{s_1,\ldots,s_n\}$ be a finite set and $c: S\to\mathbb{R}$ a weight function. c extends to all subsets A of S via

$$c(a)=\sum_{a\in A} c(A).$$

An **independence system** is a family $\mathcal{I}$ of subsets of S such that

$$\phi\in\mathcal{I}; \tag{2.0}$$

$$I\in\mathcal{I} \text{ and } K\subseteq I \text{ implies } K\in\mathcal{I}. \tag{2.1}$$

Assuming $c(s_1)\geq\ldots\geq c(s_n)\geq 0$, the *greedy algorithm* constructs a feasible solution for the optimization problem

$$\max_{I\in\mathcal{I}} c(I)$$

by starting with ϕ and then trying to add $s_1, s_2, \ldots$ in turn to the partial solution already obtained until no further feasible augmentation is possible.

Boruvka (1926) showed that the greedy algorithm is successful provided $\mathcal{I}$ is a *matroid,* i.e., satisfies in addition to (2.0) and (2.1)

$$I, K\in\mathcal{I} \text{ and } |I|<|K| \text{ imply } I\cup\{x\}\in\mathcal{I}$$
$$\text{for some suitable element } x\in K\setminus I. \tag{2.2}$$

Conversely, it is not hard to see that (2.2) is also necessary for the greedy algorithm to work relative to all possible weightings c.

It turns out that many structures in algorithmic combinatorics satisfy properties (2.0) and (2.2) but not necessarily (2.1) (e.g., the *bisimplicial elimination schemes* below). Such structures have been called *greedoids* (see Korte et al. (1991) for a comprehensive treatment). In spite of their name, however, greedoids do not necessarily guarantee the greedy algorithm to be successful unless they are "essentially matroids" (see also Helman et al. (1993)).

For independence systems that are intersections of k matroids, Korte and Hausmann (1978) proved that the greedy algorithm achieves at least $\frac{1}{k}$ times the optimum.

For $k=2$, an example is obtained from a bipartite graph $G=(V_1\dot\cup V_2, E)$ with $E\subseteq V_1\times V_2$ when we take $S=E$ and define the matroids $\mathcal{I}_i (i=1,2)$ to consist of all subsets of pairwise in V_i non-incident edges. $\mathcal{I}_1\cap\mathcal{I}_2$ consists now of all *matchings* of G, i.e., all sets of pairwise non-incident edges in G. (More generally, it is easy to verify by induction that the greedy algorithm finds a matching yielding at least half the maximum in arbitrary, not necessarily bipartite graphs.)

If the objective function c has a special structure, bipartite matching problems may be optimally solved by a greedy-type algorithm: fill the edge with currently minimal pair of indices and then remove it (and all incident edges) from the graph. This algorithm is the so-called **north-west corner rule** for assignment and transportation problems. (The next section will exhibit this rule as a kind of linear programming dual of the usual greedy algorithm.) For ease of exposition, however, we will restrict our discussion to assignment problems. With the usual reduction to assignment problems, the approach allows a straightforward extension to transportation problems.

Assignment problems are traditionally stated as minimization problems:

$$\begin{aligned}
&\min \sum_{i,j=1}^{n} c_{ij}x_{ij}\\
&\sum_{j=1}^{n} x_{ij}=1 \quad (i=1,\ldots,n)\\
&\sum_{i=1}^{n} x_{ij}=1 \quad (j=1,\ldots,n)\\
&x_{ij}\geq 0 \text{ integer.}
\end{aligned}$$

The graph-theoretic model underlying the assignment problem is the complete bipartite graph $K_{n,n}$ on $2n$ vertices and costs c_{ij} on the edges. It is well-known that any cost-minimal matching of size n in $K_{n,n}$ solves the assignment problem.

Say that the edge (s,t) in $K_{n,n}$ is *bisimplicial* if for all edges (u,v),

$$c_{st}+c_{uv}\leq c_{sv}+c_{ut}. \tag{2.3}$$

The process of *bisimplicial elimination* attempts to decompose $K_{n,n}$ by iteratively removing the vertices s and t of bisimplicial edges (s,t). The edge sets giving rise to such an

elimination scheme can be shown to form a greedoid and the process itself may be viewed as a Church-Rosser-type decomposition of $K_{n,n}$ (see Faigle et al. (1986)).

Derigs et al. (1986) observed that for any bisimplicial edge (s,t) there is a cost-minimal n-matching containing (s,t). It follows that any bisimplicial elimination scheme which completely decomposes $K_{n,n}$ also solves the assignment problem (see also Hoffman (1963)). We give an example.

The cost matrix $C=(c_{ij})$ has the *Monge property* if for all $s<u$ and $t<v$,

$$c_{st}+c_{uv}\leq c_{sv}+c_{ut}. \tag{2.4}$$

Hence, if C is Monge, $[(1,1),(2,2),\ldots,(n,n)]$ is a bisimplicial elimination sequence for $K_{n,n}$. It can be interpreted as the heuristic solution for the assignment problem constructed by the NW-corner rule.

A variant of the assignment problem is the *bottleneck assignment problem:*

$$\begin{aligned}&\min_{x_{ij}>0}\ \max\ c_{ij}\\ &\sum_{j=1}^{n} x_{ij}=1 \quad (i=1,\ldots,n)\\ &\sum_{i=1}^{n} x_{ij}=1 \quad (j=1,\ldots,n)\\ &x_{ij}\geq 0 \text{ integer.}\end{aligned}$$

Call $C=(c_{ij})$ now *bottleneck Monge* if for all $s<u$ and $t<v$,

$$\max\{c_{st},c_{uv}\}\leq\max\{c_{sv},c_{ut}\}. \tag{2.4'}$$

As before, one sees that the NW-corner rule will be optimal for bottleneck Monge problems (Burkard (1992)). We also remark that the preceding analysis not only applies to the assignment but also to the seemingly more general transportation problem.

Note that the Monge property of a matrix is not invariant under permutation of rows and columns. Some effort has therefore been spent on recognizing *permuted* Monge matrices efficiently (cf. Deineko and Filonenko (1979); Chandrasekaran (1986), and Klinz et al. (1992)). On the other hand, it is obvious that bisimplicial decomposition is invariant under permutation of rows and columns. Hence also permuted Monge matrices will allow a complete bisimplicial decomposition of $K_{n,n}$.

A bisimplicial edge can be easily found in time $O(n^4)$ by checking condition (2.4) for each edge. Hence a bisimplicial elimination sequence (a.k.a. *Monge sequence*) can be found in $O(n^5)$ steps. A more sophisticated implementation yields running time $O(n^3\log n)$ (cf. Alon et al. (1989)).

Without going into details, we mention a well-known result about traveling salesman problems, where the trivial permutation is the analogue of the NW-corner heuristic for assignment problems: if the distance matrix is Monge, the identity yields an optimal tour (cf. Lawler et al. (1985) and van der Veen (1992)).

3. Linear programs from set functions

We will now interprete some of the concepts of the previous section within the framework of linear programming. The basic idea is due to Edmonds (1970).

Consider a (finite) set E and a non-negative weighting $c:E\to\mathbb{R}_+$. Let f be a real-valued function defined on the collection of all subsets of E with the property $f(\phi)=0$. We are interested in the linear program

$$\begin{aligned}&\max\ \sum_{e\in E} c_e x_e\\ &\sum_{e\in A} x_e\leq f(A) \quad \text{for all } A\subseteq E\end{aligned} \tag{3.1}$$

and its dual

$$\begin{aligned}&\min\ \sum_{A\subseteq E} f(A)y_A\\ &\sum_{A\ni e} y_A=c_e \quad \text{for all } e\in E\\ &y_A\geq 0.\end{aligned} \tag{3.1*}$$

Let us relabel the elements, if necessary, so that $E=\{1,\ldots,n\}$, where $c_1\geq c_2\geq\ldots\geq c_n\geq 0$. For $i=1,\ldots,n$, we define

$$E_i=\{1,2,\ldots,i\}.$$

A greedy approach to (3.1) constructs the vector $\bar{\mathbf{x}}\in\mathbb{R}^n$ as follows:

$$\bar{\mathbf{x}}_i=f(E_i)-f(E_{i-1}) \quad (i=1,\ldots,n), \tag{3.2}$$

where $E_0=\phi$.

Dually, we construct y^* in a similar greedy fashion relative to the list $c_1\geq c_2\geq\ldots\geq c_n\geq 0$:

$$y_A^*=\begin{cases}c_i-c_{i+1} & \text{if } A=E_i \text{ for some } i\\ 0 & \text{otherwise,}\end{cases} \tag{3.2*}$$

where $c_{n+1}=0$.

By construction, y^* is a feasible solution for (3.1*) and satisfies

$$\sum_{A\subseteq E} f(A)y_A^*=\sum_{i\in E} c_i\bar{x}_i.$$

Linear programming duality now implies that both y^* and $\bar{x}$ are optimal for (3.1) and (3.1*) resp. – provided $\bar{x}$ is feasible for (3.1)!

A sufficient condition for $\bar{x}$ to be feasible is that f is *submodular,* i.e., for all $A,B\subseteq E$, we have

$$f(A\cup B)+f(A\cap B)\leq f(A)+f(B). \tag{3.3}$$

Submodular functions arise, for example, from matroid independence systems $\mathcal{I}$ via

$$f(A)=\max\{|A\cap I|\ \ |I\in\mathcal{I}\}. \tag{3.4}$$

In the matroid case, the vector $\bar{\mathbf{x}}$ is *exactly* the $(0,1)$-incidence vector of the solution generated by Boruvka's greedy algorithm.

Let us formally generalize our model by assuming a (partial) order $P=(E,\leq)$ on E. An *ideal* of P is a subset $A\subseteq E$ such that for all $a\in A$ and $e\in E$,

$$e\leq a \quad \text{implies} \quad e\in A. \tag{3.5}$$

Denote by A^+ the maximal elements of the ideal A in the order induced by P. Observe that the collection $\mathcal{F}=\mathcal{F}(P)$ of ideals of P is closed under unions and intersections.

We are interested in the optimization problem

$$\max \sum_{e\in E} c_e x_e$$
$$\sum_{e\in A^+} x_e \leq f(A) \quad \text{for all } A\in\mathcal{F} \tag{3.6}$$

and its LP-dual

$$\min \sum_{A\in\mathcal{F}} f(A)y_A$$
$$\sum_{A^+\ni e} y_A = c_e \quad \text{for all } e\in E. \tag{3.6*}$$
$$y_A \geq 0.$$

It is instructive to discuss in some detail the case of the complete bipartite graph $K_{n,n}$. We suppose the vertex sets $V_1=\{a_1<\ldots<a_n\}$ and $V_2=\{b_1<\ldots<b_n\}$ to be naturally ordered by their labels and take P as the induced partial order on the set $E=V_1\cup V_2$.

Each ideal A in P is determined by its set A^+ of maximal elements, which corresponds to a pair (i,j) with $0\leq i,j\leq n$. For example, $i\neq 0\neq j$ may be thought of as the edge (a_i,b_j) in $K_{n,n}$ while $(0,j)$ denotes the vertex b_j or, more precisely, the ideal consisting of the first j elements of V_2.

The non-negative weighting f' of the edges of $K_{n,n}$ can be extended to all ideals of P by

$$f_{ij}=\begin{cases} f'(a_i,b_j) & \text{if } i,j\geq 1\\ 0 & \text{if } i=j=0\\ M & \text{otherwise,}\end{cases} \tag{3.7}$$

where M is some number larger than the sum of the edge weights in $K_{n,n}$. Assume $c\equiv 1$ for simplicity.

It is now clear that (3.6*) is equivalent with the assignment problem in $K_{n,n}$. Moreover, f' has the Monge property in $K_{n,n}$ if and only if f is submodular on the set $\mathcal{F}(P)$ of ideals of P!

In order to approach (3.6) in the spirit of (3.2), we list E in the sequence $a_1,b_1,a_2,b_2,\ldots,a_n,b_n$, which we also write as $e_1,e_2,\ldots,e_{2n}$. Define again for $i=1,\ldots,2n$, the ideal

$$E_i=\{e_1,e_2,\ldots,e_i\}$$

and construct $\bar{x}$ greedily via

$$\bar{x}_i=f(E_i)-\bar{x}_{i-1}. \tag{3.8}$$

(Note that $E_i^+=\{e_{i-1},e_i\}$ for $i\geq 2$).

With $\bar{x}$ satisfying (3.8), we apparently have

$$\sum_{i=1}^{2n}\bar{x}_i=\sum_{j=1}^{n} f(E_{2j}). \tag{3.9}$$

The r.h.s. of (3.9) suggests to take as the greedy solution of (3.6*) in our case the vector y^* given by

$$y_A^*=\begin{cases}1 & \text{if } A=E_{2j} \text{ for some } 1\leq j\leq n\\ 0 & \text{otherwise,}\end{cases} \tag{3.8*}$$

which is precisely the heuristic solution according to the NW-corner rule. To prove the respective optimality of (3.8) and (3.8*) it therefore suffices to verify that $\bar{x}$ is feasible, which is a routine computation when f is submodular.

We have now seen that the NW-corner rule for assignment problems with Monge costs is just the LP-dual of a matroid-type greedy algorithm. If one takes, more generally, the partial order P to consist of k chains, one arrives at the model introduced by Queyranne et al. (1993). This model in particular comprises the setting for k-dimensional transportation problems (cf. Bein et al. (1991)). The Monge property and NW-corner rule generalize directly to this setting.

More generally, problem (3.6) with submodular costs can be solved for forests P (cf. Faigle and Kern (1993a)). For arbitrary P, the problem is open.

4. Greedy linear programs

In Sect. 3, we looked at linear programs with $(0,1)$-constraint matrices and right-hand-sides that guarantee the greedy algorithm to be successful. We now allow general non-negative constraint matrices A and modify the greedy algorithm so that it always produces some feasible solution for the linear program

$$\max cx$$
$$Ax\leq b, \tag{4.1}$$

where $b\geq 0$ and $c\geq 0$, and the coordinates are arranged so that $c=(c_1,\ldots,c_n)$ satisfies $c_1\geq c_2\geq\ldots\geq c_n\geq 0$.

We start with an approach due to Kovalev and Vasilkov (1992) and introduce a (partial) order "$\preccurlyeq$" on $\mathbb{R}^n$ which is compatible with our objective function $f(x)=cx$:

$$(x_1,\ldots,x_n)\preccurlyeq(y_1,\ldots,y_n) \text{ iff } \sum_{j=1}^{i}x_j\leq\sum_{j=1}^{i}y_j \quad \text{for } i=1,\ldots,n. \tag{4.2}$$

It follows that $c_1\geq\ldots\geq c_n$ holds if and only if $f(x)=cx$ is monotone increasing with respect to "$\preccurlyeq$". Denoting by D the (non-empty) feasibility domain of (4.1), we thus want to find the best maximal element in $(D,\preccurlyeq)$.

The greedy algorithm finds a maximal element $\bar{x}$ of $(D, \preccurlyeq)$ as follows:

for $i = 1, \ldots, n$,

$$\bar{x}_i = \max \{x \in \mathbb{R} \mid (x_1, \ldots, x_{i-1}, x, 0, \ldots, 0) \in D\}. \quad (4.3)$$

Clearly, $\bar{x}$ will be optimal if $(D, \preccurlyeq)$ has a unique maximal element. Assuming (4.1) to be non-degenerate, let B be the (row) basis of A generated by the greedy solution (4.3). Kovalev and Vasilkov then observe that $(D, \preccurlyeq)$ has a unique maximum if and only if for $i = 1, \ldots, n$,

$$c^i B^{-1} \geq 0, \quad (4.4)$$

where $c^i = (1, \ldots, 1, 0, \ldots, 0)$ is the sum of the first i unit vectors. The proof of (4.4) essentially is linear programming duality and the fact that $f(x) = cx$ is $\preccurlyeq$-monotone if and only if c lies in the cone generated by $c^1, \ldots, c^n$.

While (4.4) offers an opportunity to check whether the greedy solution is optimal, it would be useful to have a combinatorial characterization of those matrices A for which the greedy algorithm solves the associated linear program. We consider a variant of (4.1).

$$\begin{aligned} &\max cx \\ &Ax \leq b \quad (4.1') \\ &0 \leq x \leq h. \end{aligned}$$

One can now show (Faigle and Kern (1993b)) that the greedy algorithm (4.3) successfully solves (4.1′) for any r.h.s. $b \geq 0$ and $h \geq 0$ if the non-negative matrix A satisfies

The positive entries in A are monotone non-decreasing in each row. (4.5.1)

The pattern matrix underlying A is totally balanced. (4.5.2)

Here the pattern matrix of A is the matrix obtained when each non-zero element of A is replaced by "1". There are various ways to define and recognize efficiently totally balanced $(0, 1)$-matrices M (cf., e.g., Anstee and Farber (1984) or Hoffman et al. (1985)). In our context, it might be interesting to note that M is totally balanced if and only if $(-M)$ is the cost matrix of a complete bipartite graph $K_{m,n}$ with a complete bisimplicial elimination scheme (cf., e.g., Faigle et al. (1986)).

One may now extend the applicability of the greedy algorithm by studying compositions of linear programs that retain greedy solvability. Motivated by the analogous graph-theoretic construction, Bein and Brucker (1986) introduced a *series* and a *parallel* composition of the linear programs

$$\begin{array}{ll} \max \sum c_i x_i & \max \sum d_j y_j \\ \sum A_i x_i \leq a & \sum B_j y_j \leq b \\ x_i \geq 0 & y_j \geq 0 \end{array} \quad (4.6)$$

to be the problems:

$$\begin{aligned} &\max \sum (c_i + d_j) z_{ij} \\ &\sum_{i,j} \begin{pmatrix} A_i \\ B_j \end{pmatrix} z_{ij} \leq \begin{pmatrix} a \\ b \end{pmatrix} \quad (4.6.s) \\ &z_{ij} \geq 0 \end{aligned}$$

$$\begin{aligned} &\max \sum c_i x_i + \sum d_j y_i \\ &\sum A_i x_i \leq a \quad (4.6.p) \\ &\sum B_j y_j \leq b \\ &x_i, y_j \geq 0 \end{aligned}$$

where the A_i's and B_j's denote column vectors (see also Bein et al. (1993)).

With this technique, it can be shown, for example, that network flow problems with an underlying series-parallel directed graph can be solved by the greedy algorithm. On the other hand, Bein et al. (1985) demonstrate that greedy solvability of flow problems generally does not extend beyond series-parallel networks.

5. Probabilistic analysis

Probabilistic analysis of algorithms traditionally studies the expected behavior of deterministic algorithms with respect to random inputs. Such an *average case analysis* very much depends on the model chosen (what is, e.g., a "random graph"?). Often random problems seem to have the property that the relative difference between the best and the worst solution is small so that any algorithm "performs well" (see, e.g., Burkard and Finke (1993)).

We must leave it to the reader to decide whether he finds a chosen model for randomness "convincing". We concentrate here on aspects of greediness and refer to Slominski (1982); Pearl (1984) and Frieze (1990) for more results an algorithmic average case analysis.

Another line of research investigates the performance of algorithms whose executions may include certain randomized decisions on concrete input data. Consider, for example, the matching problem on the fixed graph G. If we iteratively pick an edge and delete its endpoints from the graph, we will end up with a matching at least half the size of the optimum. Can this be improved by randomizing the edge selection process? There are (at least) two strategies for selecting an edge "at random":

Select an edge at random in the set of all edges. (5.1)

Select a vertex at random and then an adjacent edge at random. (5.2)

The expected performance of strategy (5.1) can be forced to come arbitrarily close to the deterministic lower bound $\frac{1}{2}$: take as G a matching with n edges and add the edges of a complete graph that comprises exactly one vertex

from each edge in the matching. On the other hand, Aronson et al. (1993) can show that (5.2) yields a performance of at least $\frac{1}{2}+\varepsilon$ with $\varepsilon \geq 0.001$, where the lower bound is conjectured to be not best possible.

Karp and Sipser (1981) have analyzed a greedy heuristic in the spirit of (5.1) that avoids the pitfalls of the worst-case example above:

Select, if possible, a vertex with exactly one incident edge at random. Otherwise, select any edge at random. (5.1′)

Karp and Sipser proved that (5.1′) produces with high probability a matching of almost optimum size on random graphs $G_{n,c}$ with n vertices according to the following model: two vertices are linked by an edge with probability $p=\frac{c}{n}$, where $c>0$ is fixed. Tinhofer (1984) computed the expected size of a matching found by (5.1) and (5.2) on random graphs $G_{n,cn}$ and concluded that (5.2) performs at least as well as (5.1) on the average, which is also supported by empirical evidence.

Returning again to bipartite graphs, recall that the maximum weight matching problem is equivalent to the assignment problem if we insist on exact algorithms. It is thus somewhat surprising that the greedy heuristic "pick always the currently best edge available" performs quite differently on both problems. As noted in Sect. 2, the greedy algorithm produces a matching at least half the weight of the maximum weight matching. In the minimum weight (i.e., assignment) case, Reingold and Tarjan (1981) bound the size of the greedy matching by a factor of order $n^{0.58496}$ times the optimum.

Nawijn and Dorhout (1989) investigate the expected performance of a preprocessing heuristic for assignment problems. In their random model, the cost coefficients are drawn independently from some continuous distribution, which ensures that the assignment steps below can be carried out unambiguously. With respect to the cost matrix C, the heuristic has two phases:

Subtract from each row of C the row minimum to obtain a reduced matrix C'. (5.3.I)

Subtract from each column of C the column minimum to obtain a reduced matrix C''. (5.3.II)

Clearly, C and C'' give rise to equivalent assignment problems. We now construct a 0-cost (partial) assignment in C'' as follows: first assign the rows where a "0" was generated in phase II greedily to corresponding columns; then assign the columns where a "0" was generated in phase I greedily to corresponding rows.

Nawijn and Dorhout derive the expected size of this partial assignment to be about 80% of a complete assignment in the limit. Numerical experiments suggest that this performance is already achieved on small test problems.

We now turn to another type of assignment problem, namely *makespan scheduling* on m indentical parallel machines. While this problem can be formulated in the language of bipartite graphs, the usual model just assumes to be presented with m identical *machines* and n *jobs* $J_1,\ldots,J_n$ with processing times $p_1,\ldots,p_n$. The jobs are to be assigned to the machines so that the maximum *makespan*, i.e., the sum of the processing times on a single machine, is minimized over the machines.

The *list scheduling* heuristic processes the jobs according to the list $J_1,J_2,\ldots,J_i,\ldots$ so that J_i is assigned to the machine with the lightest load so far.

Denoting by LS the achievement of the list schedule and by OPT the theoretical optimum, a by now classical result of Graham (1969) guarantees the performance

$$\frac{\mathrm{LS}}{\mathrm{OPT}} \leq \frac{4}{3}-\frac{1}{3m} \tag{5.4}$$

provided the jobs have been arranged so that $p_1 \geq p_2 \geq \ldots \geq p_n$.

Bruno and Downey (1986) study a random model for makespan problems where the processing times p_i are drawn for a uniform distribution. It then turns out that for every $\varepsilon>0$ there exists some $N=N(\varepsilon)$ such that

$$\Pr\left[\frac{\mathrm{LS}}{\mathrm{OPT}} < 1+\frac{4(m-1)}{n}\right] > 1-\varepsilon \tag{5.5}$$

whenever $n \geq N$. Note that (5.5) makes no assumption on the particular nature of the list. In other words, any list performs well with high probability for large enough n. Hence one may expect list scheduling to be also the algorithm of choice for on-line makespan scheduling, where decisions have to be made under incomplete information.

6. On-line algorithms

On-line algorithms apply to situations where input data are served piece by piece and each time a decision has to be made that cannot be revized later. Many on-line problems require to partition a set into "independent" subsets, which can be viewed as the problem of assigning elements on-line to certain "machines" that represent to blocks of the partition. In view of the incomplete information at the time a decision has to be made, a natural first approach to an on-line problem is the greedy principle. This principle is often indeed optimal. Refering to Faigle et al. (1989) for a general discussion of on-line algorithms for partitioning problems, we focus here on a few select newer results.

We measure the performance of an on-line algorithm by bounding the ratio between the achievement of the algorithm so far with the theoretical optimum *at any time an algorithmic decision has to be made.*

Consider, for example, the makespan scheduling problem when the jobs arrive in some a priori unknown order and in unknown quantity and have to be assigned as they arrive. Assume $m=2$ and the jobs J_1 and J_2 with processing times $p_1=p_2=1$. If J_1 and J_2 are assigned to the same machine, the ratio is poor if no further job arrives. If J_1 and

J_2 are assigned to different machines and a job J_3 with $p_3=2$ arrives, we see that no on-line algorithm can guarantee a better performance ratio than $3/2$.

Similarly, the possible sequence $(1,1,3,3,3,6)$ of processing times shows that no on-line algorithm for the 3-machine problem yields a better overall ratio than $5/3$.

On the other hand, Graham (1969) observed that list scheduling generally offers an on-line performance ratio of

$$\frac{\mathrm{LS}}{\mathrm{OPT}} \leq 2 - \frac{1}{m} \tag{6.1}$$

and hence is optimal for $m \leq 3$. Graham's upper bound is sharp for list scheduling for each m. It was therefore conjectured that no on-line algorithm for makespan scheduling has an asymptotic performance strictly better than 2.

In a remarkable paper Bartal et al. (1992) proposed an on-line algorithm that incorporates the idea of *balanced list scheduling* (BLS) and has performance ratio of

$$\frac{\mathrm{BLS}}{\mathrm{OPT}} \leq 2 - \varepsilon \tag{6.2}$$

for some $\varepsilon > 0$ (independent of m). The idea is to keep always about δm, $0 \leq \delta \leq 1$, of the machines "lightly loaded". More precisely, let $A(F_i)$ be the average load of the δm machines with the light loads at time i, $M(L_i)$ the lightest load of the $m - \delta m$ machines with the heavier loads. Job J_i is then assigned according to the rule

IF $M(L_i) + p_i \leq (2-\varepsilon)A(F_i)$, assign J_i to the machine with load $M(L_i)$;

ELSE assign J_i to the machine with overall lightest load. (6.3)

In an involved analysis, Bartal et al. prove that the bound in (6.2) can be achieved if $\delta \approx 0.445$ and $\varepsilon \leq \frac{1}{70}$. They also show that the deterministic optimal bound (6.1) for $m=2$ can be improved with the help of randomization. Note that here not the input data but the decisions of the algorithm are randomized.

The randomized strategy is as follows: assign J_i by flipping a (biased) coin so that the expected load on the heavier machine is twice the load of the lighter machine.

This way an expected performance ration of $4/3$ is achieved and one can show that no randomized algorithm performs better on 2 machines. The problem of optimal randomized algorithms for $m \geq 3$ machines is largely open.

Another on-line job scheduling problem is considered in Faigle and Nawijn (1991). Jobs $J_1, J_2, \ldots$ arrive at times $t_1, t_2, \ldots$ and require immediate service times $p_1, p_2, \ldots$ at one of m identical service stations. A job is "lost" if it is not serviced immediately without interruption. The objective is to minimize the number of jobs lost.

A greedy on-line algorithm inspects at time t_i all service stations and assigns J_i to any free station. If all stations are busy, the job J with the largest completion date t is replaced by J_i if $t_i + p_i < t$ (and hence lost). Otherwise J_i is lost. This strategy turns out to be not only an optimal on-line strategy but to be a generally optimal algorithm.

The case is mostly open if we allow the service stations to work only during given time windows. For $m=2$, however, the greedy strategy above can be modified to yield an almost optimal algorithm (Nawijn (1993)).

Let us finally return to the weighted bipartite matching problem, whose on-line complexity has been analyzed by Kalyanasundaram and Pruhs (1993). The model assumes an underlying complete bipartite graph $K_{n,n} = (V_1 \cup V_2, E)$ on $2n$ vertices. The weights on the edge set E are assumed to satisfy the triangle inequality. Their values are at first unknown. Then for each $v \in V_1$, the weights of the edges incident with v are revealed and v must be matched with a vertex in V_2.

The greedy on-line algorithm for the maximum weight problem always selects the best available edge. Kalyanasumdaram and Pruhs show that this strategy guarantees a performance ratio of 3 and that no on-line algorithm for this problem obtains a better ratio.

Again, for minimum weight matching, the situation is quite different. The greedy algorithm performs exponentially poorly while an optimal on-line algorithm exists with performance ratio $2n-1$. The latter algorithm proceeds in such a way that the partial matching constructed up to step i covers the same vertices that would be convered by an optimal partial matching up to step i. This strategy can be realized by maintaining an optimal matching and, in each step, computing an appropriate augmenting path.

References

Alon N, Cosares S, Hochbaum DS, Shamir R (1989) An algorithm for the detection and construction of Monge sequences. Lin Alg Appl 114/115:669–680

Anstee RP, Farber M (1984) Characterization of totally balanced matrices. J Alg 5:215–230

Aronson J, Dyer M, Frieze A, Suen S (1993) Randomized greedy matching II. Preprint, Department of Math., Carnegie-Mellon University, Pittsburgh

Bartal Y, Fiat A, Karloff H, Vohra R (1992) New algorithms for an ancient scheduling problem. J comp Syst Sci (to appear)

Bein W, Brucker P (1986) Greedy concepts for network flow problems. Dis Appl Math 15:135–144

Bein W, Brucker P, Hoffman AJ (1993) Series parallel composition of greedy linear programming problems. Math Programming B62:1–14

Bein W, Brucker P, Park JK, Pathak PK (1991) A Monge property for the d-dimensional transportation problem. Dis Appl Math (to appear)

Bein W, Brucker P, Tamir A (1985) Minimum cost flow algorithms for series-parallel networks. Dis Appl Math 10:1177–124

Boruvka O (1926) On jistem problemu minimalnim. Prace Moravske Prirodovedecke Spolecnosti 3:37–53

Burkard RE (1992) A note on greedily solvable time transportation problems. Report No. 243, Inst. für Mathematik, TU Graz

Burkard RE, Fincke U (1983) The asymptotic probabilistic behaviour of quadratic sum assignment problems. Z Oper Res A 27:73–82

Bruno JL, Downey PJ (1986) Probabilistic bounds on the performance of list scheduling. Math OR 15:409–417

Chandrasekaran R (1986) Recognition of Gilmore-Gomory traveling salesman problem. Dis Appl Math 14:231–238
Deineko VG, Filonenko VL (1979) On the reconstruction of specially structured matrices. Aktualnyje Problemy EVM: programmirovanije, Dnepropetrovsk, DGU, 43–45 (in Russian)
Derigs U, Goecke O, Schrader R (1986) Monge sequences and a simple assignment problem. Dis Appl Math 5:241–248
Edmonds J (1970) Submodular functions, matroids, and certain polyhedra. In: Guy R, et al. (eds) Combinatorial structures and their Applications. Gordon and Breach, New York, pp 69–87
Erdös P, Graham RL (1980) Old and new problems and results in combinatorial number theory. Monographie No. 28, L' Enseignement Mathématique, Genève
Faigle U (1987) Matroids in combinatorial optimization. In: White N (eds) Combinatorial geometries. Encyclopedia of Mathematics and its Applications 29. Cambridge University Press, Cambridge, pp 161–210
Faigle U, Goeke O, Schrader R (1986) Church-Rosser decomposition of combinatorial structures. Preprint, Institut für Operations Research, Universität Bonn
Faigle U, Kern W (1993a) Submodular linear programs on forests. (Workingpaper)
Faigle U, Kern W (1993b) (unpublished)
Faigle U, Kern W, Turàn Gy (1989) On the performance of on-line algorithms for partition problems. Acta Cyber 9:107–119
Faigle U, Nawijn WM (1991) Greedy k-coverings of interval orders. Dis Appl Math (to appear)
Frank A, Tardos É (1988) Generalized polymatroids and submodular flows. Math Prog 42:489–563
Frieze AM (1990) Probabilistic analysis of graph algorithms. Comput Supp 7:209–233
Fujishige S (1990) Submodular functions and optimization. Ann Discr Math 47
Graham RL (1969) Bounds on multiprocessing timing anomalies. SIAM J Appl Math 17:416–429
Helman P, Moret BME, Shapiro HD (1993) An exact characterization of greedy structures. SIAM J Dis Math 6:274–283
Hoffman AJ (1963) On simple linear programming problems. In: Klee V (ed) Convexity. American Math Soc Providence RI, pp 317–327
Hoffman AJ, Kolen AWJ, Sakarovitch M (1985) Totally-balanced and greedy matrices. SIAM J Alg Dis Methods 6:721–730
Kalyanasundaram B, Pruhs K (1993) Online weighted matching. J Alg 14:478–488
Karp RM, Sipser M (1981) Maximum matchings in sparse random graphs. Proc 22nd Annual IEEE Symp FOCS, pp 364–375
Klinz B, Rudolf R, Woeginger GJ (1992) On the recognition of permuted bottleneck Monge matrices. Report No. 241, Inst für Mathematik, TU Graz
Korte B, Hausmann D (1978) An analysis of the greedy heuristic for independence systems. Ann Dis Math 2:65–74
Korte B, Lovász L, Schrader R (1991) Greedoids. Algorithms and combinatorics vol 4. Springer, Berlin Heidelberg New York
Kovalev M, Vasilkov D (1992) The canonical order and greedy algorithms. Report No. 9207-DO, Dept of Appl Math, Belarusian State University Minsk
Lawler EL, Lenstra JK, Rinnoy Kan AHG, Shmoys DB (1985) The traveling salesman problem: a guided tour of combinatorial optimization. Wiley, Chichester
Nawijn WM (1993) (personal communication)
Nawijn WM, Dorhout B (1989) On the expected number of assignments in reduced matrices for the linear assignment problem. Oper Res Lett 8:329–335
Pearl J (1984) Heuristics. Addison-Wesley, Reading, Mass
Queyranne M, Spieksma F, Tardella F (1993) A general class of greedily solvable linear programs. In: Faigle U, Hoede C (eds) 3rd Twente Workshop on Graphs and Combinatorial Optimization. Memorandum No. 1132, Dept of Appl Math University of Twente, pp 176–180
Reingold E, Tarjan RJ (1981) On a greedy heuristic for complete matching. SIAM J Comp 10:676–681
Slominski L (1982) Probabilistic analysis of combinatorial algorithms: A bibliography with selected annotations. Computing 28:257–267
Tinhofer G (1984) A probabilistic analysis of some greedy cardinality matching algorithms. Ann Oper Res 1:239–254
Veen van der JAA (1992) Solvable cases of the traveling salesman problem with various objective functions. Ph.D. dissertation, University of Groningen