4

Constructive Algorithms

If the objective function to be minimized is locally regular or locally concave, the relaxation of the resource constraints does not yield a tractable problem in general. Thus, the relaxation-based approach from Chapter 3 no longer proves useful. For solving resource allocation problems with locally regular or locally concave objective function f, we have to explicitly construct the schedules from an appropriate set that contains an optimal schedule if the problem is solvable. We refer to algorithms that proceed in such a way as *constructive algorithms*. The serial schedule-generation scheme for minimizing regular objective functions is an example of a constructive algorithm. In this chapter we develop constructive algorithms that are based on the second basic representation of the set $\mathcal{S}$ of all feasible schedules as a union of disjoint equal-preorder sets (recall that the term equal-preorder set may also designate an equal-order set). As we have seen in Subsections 2.1.1 and 2.1.2, the set of all minimal points or vertices of equal-preorder sets coincides with the set of all minimal points or vertices, respectively, of schedule polytopes. In Subsection 2.2.2 we have introduced the notion of quasiactive and quasistable schedules designating those feasible schedules which represent minimal points or vertices, respectively, of their schedule polytopes. The analysis in Subsection 2.3.2 has shown that for $\mathcal{S} \neq \emptyset$, the set of all quasiactive schedules always contains some optimal schedule if the objective function f under study is locally regular. Likewise, the set of all quasistable schedules always contains an optimal schedule if f is locally concave provided that $\mathcal{S} \neq \emptyset$.

Since we again consider the general case where both renewable and cumulative resources are present, we consider relations ρ in set V of all (real and fictitious) activities. Under our assumption that the real activities use the renewable resources and that the events deplete and replenish the cumulative resources, precedence relationships need only be defined among real activities and among events of the project. More formally, instead of considering schedule-induced preorders $\theta(S) = \{(i,j) \in V \times V \mid S_j \geq S_i + p_i\}$ in set V, we may restrict ourselves to schedule-induced relations arising from the union of the respective schedule-induced strict order in set V^a and the correspond-

ing reflexive preorder in set V^e. Since the ground sets V^a and V^e of those two preorders are disjoint, the union of both preorders is again transitive and thus represents a preorder in ground set $V = V^a \cup V^e$. Accordingly, for given schedule S we define the schedule-induced preorder to be

$$\theta(S) := \{(i,j) \in (V^a \times V^a) \cup (V^e \times V^e) \mid S_j \geq S_i + p_i\}$$

We notice that preorder $\theta(S)$ is neither irreflexive nor reflexive.

Now recall that any quasiactive schedule can be represented as a spanning outtree $G = (V, E_G)$ of its schedule network $N(\theta(S))$ rooted at node 0, where each arc (i,j) of G belongs to one temporal constraint $S_j - S_i \geq \delta_{ij}$ or one precedence constraint $S_j \geq S_i + p_i$ that is active at S (cf. Proposition 2.28). Similarly, any quasistable schedule can be assigned to a spanning tree G of its schedule network. In addition, we assign the weights $\delta_{ij}^G := d_{ij}^{\theta(S)}$ to the arcs $(i,j) \in E_G$. The active temporal and precedence constraints can then be written in the form

$$S_j - S_i \geq \delta_{ij}^G \quad ((i,j) \in E_G)$$

The constructive algorithms are based on generating such spanning outtrees and spanning trees G. The corresponding schedule S is obtained from G by computing the unique solution to the system of linear equations $S_0 = 0$ and $S_j = S_i + \delta_{ij}^G$ $((i,j) \in E_G)$, which can be achieved in linear time. By constructing a spanning outtree or spanning tree G we perform two consecutive tasks simultaneously: first, finding a feasible schedule-induced preorder in set V and second, computing some appropriate vertex (the minimal point if G is an outtree) of the corresponding relation polytope.

Resource allocation problems with locally regular or locally concave objective functions, regardless of containing explicit resource constraints or not, are much harder to solve to optimality on the average than resource-constrained project scheduling problems where some regular or convexifiable objective function is to be minimized. That is why in the present chapter we are concerned with heuristic procedures. In Section 4.1 we first discuss a generic schedule-generation scheme producing one quasiactive or one quasistable schedule. This schedule-generation scheme has been proposed by Neumann et al. (2000) and goes back to priority-rule heuristics for resource levelling problems that have been devised by Neumann and Zimmermann (1999*b*, 2000). The schedule-generation scheme provides an initial quasiactive or quasistable schedule, from which we may subsequently move stepwise towards different quasiactive or quasistable schedules by using an iterative improvement procedure. In Section 4.2 we then deal with tree-based neighborhood functions presented in Neumann et al. (2003*a*). Neighborhood functions constitute the essential building block of local search algorithms such as hill climbing, tabu search, simulated annealing, or threshold accepting (for an overview of different local search techniques, see Aarts and Lenstra 2003*b*). In particular, we show that the proposed neighborhoods allow local search algorithms to reach optimal schedules independently of the initial schedule chosen. Section 4.3 is