Technische Universität Berlin

FACHBEREICH 3
MATHEMATIK

Scheduling Scarce Resources in Chemical Engineering

by

Rolf H. Möhring   Marc Uetz

No. 682/2000

Scheduling Scarce Resources in Chemical Engineering*

Rolf H. Möhring and Marc Uetz

Technische Universität Berlin, Fachbereich Mathematik, Sekr. MA 6–1, Straße des 17. Juni 136, 10623 Berlin. Email: {moehring, uetz}@math.tu-berlin.de

Abstract The efficient utilization of scarce resources, such as machines or manpower, is major challenge within production planning in the chemical industry. We describe solution methods for a resource-constrained scheduling problem which arises at a production facility at BASF AG in Ludwigshafen. We have developed and implemented two different algorithms to solve this problem, an approach which is based on Lagrangian relaxation, as well as a branch-and-bound procedure. Particularly the Lagrangian approach is applicable for a whole variety of resource-constrained scheduling problems, hence it is of interest not only for the specific problem we describe, but also for many other industrial applications. In this paper, we describe both approaches, and also report on computational results, based upon practical problem instances as well as benchmark test sets.

1 Scheduling under Limited Resources

Facing the increasing international competition, there is a growing need for planning tools in chemical engineering that allow an efficient utilization of scarce resources. Within the cooperation with BASF AG, Ludwigshafen, we focus on a scheduling problem that is typical for a production process in the chemical industry, but also occurs in many other industrial applications.

The aim is to schedule the production process for several so-called *orders*, or *campaigns*. Each order represents the demand of a certain amount of a product, which must be produced on a suited machine. Due to limited machine capacities, an order is usually split up into several identical steps, so-called *jobs*. In other words, the production process for each order consists of a sequence of identical *jobs*, each of which must be scheduled on an assigned machine. The objective is to schedule all orders, or jobs, such that the overall production time is minimized. Due dates for individual orders are given, and there may also be *temporal constraints* between jobs of different orders in the form of *time lags*, for instance if an intermediate product of an order is needed within the production process of others.

* This work is a final report on a research project with BASF AG, Ludwigshafen. The authors have been funded by the Bundesministerium für Bildung und Forschung (bmb+f) under grant 03-MO7TU1-3. Most of the results are based on joint work with different colleagues, namely Andreas Fest (TU Berlin), Andreas S. Schulz (MIT, Cambridge, MA), and Frederik Stork (TU Berlin).

Apart from temporal restriction, resource constraints are imposed by limited availability of machines and scarce personnel. The availability of personnel depends on the organization of shifts, breaks, etc. Any job itself consists of several consecutive steps, so-called *tasks*, and each of these tasks requires a specified amount of personnel. Hence, the required personnel to operate a job is varying in time, and given by a corresponding piecewise constant function like depicted in Fig. 1. In a less restrictive model, the tasks of a job are only linked by so-called *time-windows*, which means that there may be idle periods of restricted time between two consecutive tasks. This leads to a model with so-called *maximal time lags*, which will be discussed in Sect. 6.

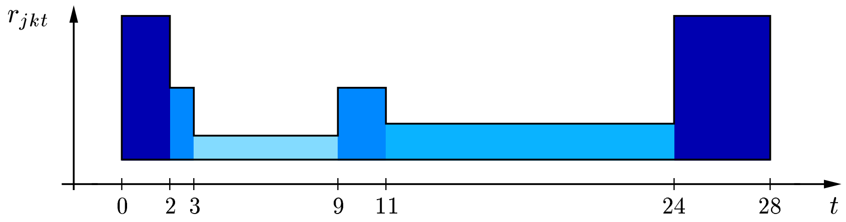

Figure1. Resource requirement of a sample job which consists of 6 tasks. Each colored rectangle corresponds to a task. The higher (darker) the rectangle, the more resources are required during the execution of the corresponding task.

The objective is to find a production schedule such that temporal as well as resource constraints are satisfied, and the overall production time, the so-called *makespan*, is minimized. For a more detailed problem description, we refer to [KW] or [MSSU3].

1.1 Notation and Formulation of the Problem

The problem described above can be formulated as a *resource-constrained project scheduling problem*. In order to put up a mathematical formulation, we introduce some notation. Let us denote by $J = \{0, \dots, n\}$ be the set of all *jobs* which have to be scheduled, where a job j has an integral processing time p_j. Except for Sect. 6, we assume that jobs must be processed without interruption, and by $S = (S_0, \dots, S_n)$ we denote a schedule, where S_j is the start time of job j. The entity of all jobs, together with their temporal constraints and resource requirements will be called a *project* in what follows. Jobs 0 and n are assumed to be dummy jobs with processing time 0, indicating the project start and the project completion, respectively. In other words, job n indicates when the production of all orders terminates. This can be easily modeled by introducing additional temporal constraints. In general, temporal constraints are given in the form of arbitrary minimal time lags between any pair of jobs, and by $d_{ij} \geq 0$ we denote the length of the required time lag

(i, j) between two jobs $i, j \in J$. $L \subseteq J \times J$ is the set of all time lags, and by $m := |L|$ we denote the number of given time lags. We assume without loss of generality that the time lags always refer to the start times of jobs, thus every schedule S has to fulfill $S_j \geq S_i + d_{ij}$ for all $(i, j) \in L$. Ordinary *precedence constraints* can be represented by letting $d_{ij} = p_i$ if job i must precede job j. Additionally, we suppose that a time horizon T as an upper bound on the *project makespan* (the time to produce all orders) is given. It can be checked in polynomial time by longest path calculations if such a system of temporal constraints has a feasible solution.

As indicated before, jobs need resources while they are operated. We assume that we are given a finite set R of different, so-called *renewable resources*, and the availability of resource $k \in R$ at time t is denoted by R_{kt}. These resources can represent machines, manpower, or any other device which is required to operate a job. We assume that a job j requires an amount of r_{jkt} units of resource k, $k \in R$, during the t-th period of its execution. Note that a job may require several resources at the same time.

The objective is to find a schedule which respects all constraints, and has minimal total processing time, or in other words, which minimizes the project makespan S_n. It is well known that this problem is NP-hard. It is even NP-hard to approximate within any constant factor (see [Sch]). That means that there is virtually no hope to find an efficient, that is, polynomial time algorithm which computes an optimum or near-optimum solution for all problem instances.

1.2 Integer Programming Formulation

In order to model the problem mathematically as an integer linear program, so-called time-indexed formulations are quite common. Pritsker, Watters, and Wolfe [PWW] were presumably the first to give an integer programming formulation in time-indexed 0-1-variables of the type

$$x_{jt} = \begin{cases} 1 & \text{if job } j \text{ starts at time } t, \\ 0 & \text{otherwise,} \end{cases}$$

where $j \in J$, and $t \in \{0, ..., T\}$. we then obtain the following integer linear programming formulation, which is crucial for our subsequent approach

$$\text{minimize} \qquad \sum_t t\, x_{nt} \tag{1}$$

$$\text{subject to} \qquad \sum_t x_{jt} = 1, \qquad j \in J, \tag{2}$$

$$\sum_{s=t}^{T} x_{is} + \sum_{s=0}^{t+d_{ij}-1} x_{js} \leq 1, \qquad (i,j) \in L,\ t = 0, \ldots, T, \tag{3}$$

$$\sum_j \sum_{s=t-p_j+1}^{t} r_{jk,t-s}\, x_{js} \le R_{kt}, \qquad k \in R,\ t = 0, ..., T, \qquad (4)$$

$$x_{jt} \ge 0, \qquad j \in J,\ t = 0, \ldots, T, \qquad (5)$$

$$x_{jt} \text{ integer}, \qquad j \in J,\ t = 0, \ldots, T. \qquad (6)$$

Note that $S_j = \sum_t t\, x_{jt}$, thus the objective (1) is to minimize the project completion time S_n. Constraints (2) indicate that each job is started exactly once, and inequalities (3) represent the temporal constraints given by the time lags L. Inequalities (4) assure that the jobs simultaneously processed at time t do not consume more resources than available. Given the temporal constraints and the time horizon T, it is easy to compute *earliest* and *latest starting times* for each job $j \in J$. For convenience of notation, however, we simply assume (without stating explicitly) that variables with time indices outside these boundaries are fixed at values zero, such that no job is started before its earliest start or after its latest start, respectively.

2 Relaxations and Lower Bounds

Recall that we are dealing with a minimization problem which is even NP-hard to approximate within any constant factor. Hence, we cannot expect to find an efficient algorithm which computes a provably good solution for any given problem instance. Is such a situation, *heuristic* solutions are usually sought. However, in order to judge the quality of a given solution, *lower bounds* on the optimal objective value are crucial. In order to compute lower bounds, a variety of different methods are available. Usually, one tries to solve *relaxations* of the problem, preferably in such a way that the relaxed problem is solvable efficiently, that is, in polynomial time.

Based upon the above integer programming relaxation, it can be shown that a *Lagrangian* relaxation of the problem can be reduced to a *minimum-cut* problem, and thus can be solved quite efficiently [MSSU3]. The idea is to dualize the resource constraints (4), and introduce nonnegative Lagrangian multipliers $\lambda = (\lambda_{kt})$, $k \in R$, $t \in \{0, ..., T\}$. Doing this, one obtains the *Lagrangian subproblem*

$$\begin{array}{ll} \text{minimize} & \sum_t t\, x_{nt} + \sum_j \sum_t \Big(\sum_{k \in R} \sum_{s=t}^{t+p_j-1} r_{jk,s-t} \lambda_{ks} \Big) x_{jt} - \sum_t \sum_{k \in R} \lambda_{kt} R_{kt} \qquad (7) \\ \text{subject to} & (2),\ (3),\ (5), \text{ and } (6). \end{array}$$

It is well known that for any set of nonnegative Lagrangian multipliers, the optimum solution of the Lagrangian subproblem is a lower bound on the optimum objective value of problem (1)– (6). Already Christofides, Alvarez-Valdes, and Tamarit [CAT] have proposed to use this Lagrangian relaxation, however, they solved the Lagrangian subproblems by branch-and-bound. If one omits the constant term $\sum_t \sum_{k\in R} \lambda_{kt} R_{kt}$ and introduces weights

$$w_{jt} = \begin{cases} \sum\limits_{k\in R} \sum\limits_{s=t}^{t+p_j-1} r_{jk,s-t}\lambda_{ks} & \text{if } j \neq n, \\ t & \text{if } j = n, \end{cases} \tag{8}$$

problem (7) can be rewritten as

$$\text{minimize} \quad c_\lambda(x) := \sum_j \sum_t w_{jt}\, x_{jt} \quad \text{subject to (2), (3), (5), and (6).} \tag{9}$$

Formulation (9) specifies the problem of finding a minimum-cost schedule for a set of jobs which must respect a set of temporal constraints, and where each job $j \in J$ incurs a cost of w_{jt} if it is started at time t. We refer to this problem as *project scheduling problem with start-time dependent costs.* It will be discussed in Sect. 3 below.

3 Scheduling with Start-Time Dependent Costs

Not only due the fact that is arises as Lagrangian subproblem for various resource-constrained scheduling problems, the scheduling problem with start-time dependent costs as formulated in (9) is of interest in its own. For instance, it generalizes the well-known net-present-value problem [Rus] to arbitrary costs w_{jt}. In the paper by Christofides et al. [CAT], this problem is solved by a branch-and-bound algorithm.

However, it is in fact solvable in polynomial time, which follows, among others, from results by Chang and Edmonds [CE]. In [MSSU2], we have given an overview of the corresponding results. We next describe a direct reduction of the scheduling problem with start-time dependent costs given in (9) to a minimum-cut problem in a directed graph. This is a crucial result with respect to the efficiency of the Lagrangian approach, and we refer to [MSSU2] and [MSSU3] for more details.

Define a digraph D by introducing a vertex u_{jt} for every job $j \in J$ and every $t = 0, \dots, T+1$. Now, define directed chains $(u_{j0}, u_{j1}), (u_{j1}, u_{j2}), \dots, (u_{jT}, u_{j,T+1})$ for any j. The corresponding arcs $(u_{jt}, u_{j,t+1})$ are called *assignment arcs.* Furthermore, define arcs between those chains which correspond to the given temporal constraints by $(u_{it}, u_{j,t+d_{ij}})$ for all temporal constraints $(i, j) \in L$. The cost coefficients w_{jt} are interpreted as the arc capacities of arc $(u_{jt}, u_{j,t+1})$, for all j and t. The capacity of the remaining arcs is set to infinity. Then, after the introduction of a dummy source a and sink b, a solution of the original scheduling problem can be computed as a minimum a-b-cut in that digraph. Fig. 2 gives an example for the construction of D.

Theorem 1 ([MSSU3]). *A minimum a-b–cut $(X, \bar{X})$ of the digraph D described above corresponds to an optimal solution of the project scheduling problem with start time dependent costs* (9) *by virtue of*

$$x_{jt} = \begin{cases} 1 & \textit{if } (u_{jt}, u_{j,t+1}) \textit{ is a forward arc of the cut } (X, \bar{X}), \\ 0 & \textit{otherwise.} \end{cases} \tag{10}$$

Moreover, the value $c_\lambda(x)$ of that solution equals the capacity $c(X, \bar{X})$ of any minimum cut $(X, \bar{X})$ of D.

Using a push-relabel maximum-flow algorithm [GT], this results in an algorithm for solving the project scheduling problem with start-time dependent costs with running time $O(n\,m\,T^2 \log(n^2T/m))$. Note that this holds for arbitrary (also maximal) time lags.

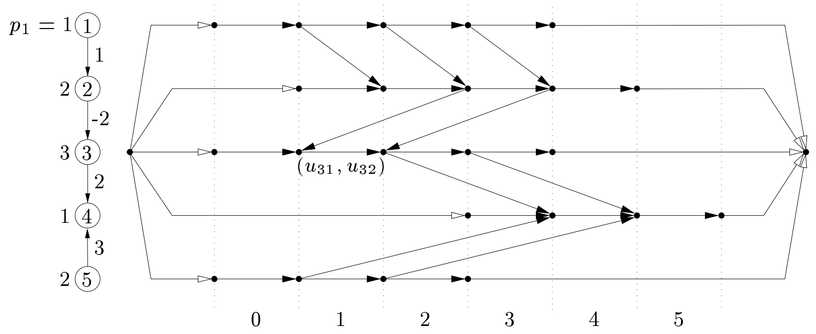

Figure2. The left digraph represents the data of the sample instance: Each vertex represents an job, each arc represents a temporal constraint. The values for the time lags are $d_{12} = 1, d_{23} = -2, d_{34} = 2$, and $d_{54} = 3$. The job processing times are $p_1 = p_4 = 1$, $p_2 = p_5 = 2$, and $p_3 = 3$. $T = 6$ is a given upper bound on the project makespan. The right digraph D is obtained by the above transformation. Each assignment arc $(u_{jt}, u_{j,t+1})$ corresponds to a binary variable x_{jt} of formulation (9).

4 Linear Programming or Lagrangian Relaxation?

Based on this insight, the computation of lower bounds via Lagrangian relaxation can be realized quite efficiently: Using a standard *subgradient optimization* procedure to compute a near-optimal set of Lagrangian multipliers, the computation of lower bounds reduces to a series of minimum-cut computations in the above defined digraph. In fact it is well known that, since the polytope defined by inequalities (2), (3), and (5) is integral, the optimum solution value of the *Lagrangian dual*

$$c_{\lambda^*} := \sup_{\lambda \geq 0} c_\lambda \,,$$

equals the value of an optimal solution of the linear programming relaxation (1) – (5). Here, c_λ denotes the value of an optimum solution of (9) for a fixed value of Lagrangian multipliers $\lambda = (\lambda_{kt})$, $k \in R$, $t = 0, \ldots, T$.

The benefit of the new insights described in the preceding section is perhaps best documented by the comparison of the corresponding computation times in Tab. 1. There, we compare the computation times when solving linear programming relaxation (1) – (5) to the computation times to solve the Lagrangian multiplier problem as described above. The figures in Tab. 1 are based upon the well-established ProGen benchmark instances for resource constrained project scheduling [KS]. We used 600 ProGen instances, each with 120 jobs. The instances are available under [Pro1]. The table shows computation times as well as the average quality of the lower bounds in terms of improvement over the so-called *critical path* lower bound, which is the length of a longest path in the project network.

Table1. Comparison of computation times to solve the linear programming (LP) relaxation (1) – (5) and the multiplier problem for the Lagrangian relaxation (LR). Computation times were obtained on a Sun Ultra 2 with 200MHz, operating under Solaris. The linear programs have been solved with CPLEX version 6.5.3.

Algorithm	Average CPU	Above Crit. Path
LP	38 min. (max. 23 h)	20.5%
LR	37 sec. (max. 8 min.)	19.6%

Obviously, the computation times are drastically reduced when using the Lagrangian approach instead of solving the linear programming relaxation. The average computation times for the Lagrangian approach are based upon an average number of 70 iterations. The quality of the corresponding lower bounds deviate only marginally, due to the fact that the subgradient optimization is not exact, but rather converges to the optimum value c_{λ^*}.

5 From Minimum Cuts to Feasible Solutions

Let us now turn to the question of how to obtain feasible solutions for the scheduling problem at BASF AG. So far, we have computed a lower bound on the best possible solution as described before. In fact, we not only have computed a lower bound, but in every iteration of the above described subgradient procedure, we have computed a schedule which respects all temporal restrictions, but might be infeasible with respect to the resource constraints. The intuition behind our approach to obtain good feasible solutions is to exploit some of the information held in these resource-infeasible schedules.

The simple idea is to use *list scheduling* algorithms which are based upon so-called α-*completion* times of the jobs.

Such *relaxation-based ordering heuristics*, combined with α-completion time variables, have been used previously to obtain worst case performance bounds for certain *machine scheduling* problems. The papers [PSW] and [HSW] are two of the early references in this direction. An application of *linear programming* based approaches to resource-constrained scheduling problems has been previously analyzed in [CDS$^+$] and [SUW].

Let us now briefly sketch the framework for a combined algorithm which computes both, lower bounds and feasible solutions. First, to obtain an initial valid upper bound to start with, we use list scheduling algorithms fed with some standard priority rules. (Remember that we use a time-indexed formulation and require a time horizon T.) Then, in each iteration of the subgradient optimization algorithm, a time-feasible (but likely resource-infeasible) schedule is computed by solving the Lagrangian subproblem (9) as described in Sect. 3. The cost of this time-feasible schedule, in terms of the w_{jt} defined in (8), is now a valid lower bound for the original problem (1)– (6). Using orderings according to α-completion times of jobs, we then compute feasible solutions by means of different list scheduling algorithms, schematically depicted in Fig. 3. In fact, we have developed new list scheduling algorithms which showed to be particularly suited for our approach. See [SU] and [MSSU3] for more details.

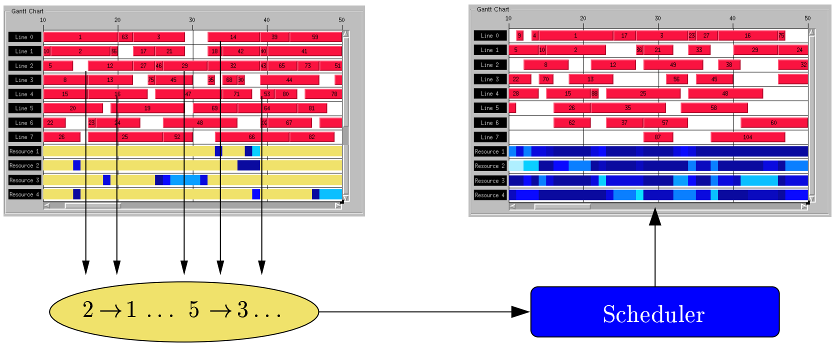

Figure3. Computing feasible solutions by relaxation-based ordering heuristics. Information is extracted from the resource-infeasible solution of the Lagrangian relaxation (left picture), and used as input for List-Scheduling heuristics.

With this Lagrangian approach, we could solve a practical instance provided by BASF AG to optimality within a couple of seconds by computing matching lower and upper bounds. The instance originally consists of 21 orders which are split up into 2121 jobs, resulting in a total number of 4545 tasks. The resulting number of time lags is 2100. Although this is a very large problem instance, it turned out to be comparatively easy to solve, since a simple preprocessing reduces its size to 129 jobs and 557 tasks.

Also on the well established ProGen benchmark instances [KS], the performance of the Lagrangian approach was very encouraging. In fact, the solutions compare to results obtained with state-of-the-art local search algorithms. The figures in Tab. 2 are again averaged over 600 ProGen instances [KS], each with 120 jobs. The values for best known lower bounds and feasible solutions for these problem instances are maintained and available at [Pro1]. As Tab. 2 suggests, the average deviations from both best known lower bounds and feasible solutions are below 3%, at an average computation time of 88 seconds per instance. Note that the best known solutions have been obtained by various researchers using different local-search, as well as branch-and-bound algorithms over the years, thus computation times cannot be given. The best known lower bounds are due to Brucker and Knust [BK]. They, however, require up to 72 h computation time per problem instance, whereas the Lagrangian approach only requires a maximum computation time of 11 minutes. In view of these figures, the Lagrangian approach seems to offer an excellent tradeoff between the quality of the bounds and the computational effort to get them.

Table2. Deviation of lower bounds and feasible solutions from best known lower and upper bounds. Computation times were obtained on a Sun Ultra 2 with 200MHz, operating under Solaris.

CPU	Average Deviations in % best known lower bound	best known solution
88 sec. (max. 11 min.)	2.7 (max. 23.0)	2.1 (max. 9.3)

Additionally, we have tested the algorithm using a benchmark set of 25 instances of the BASF-type which are available at [Cav]. These instances are known to be notoriously difficult. Tab. 3 shows the results obtained with four of these sample instances of different size. The Table shows the number of jobs and tasks as well as the critical path lower bound. T denotes the time horizon, and #it is the number of iterations in the subgradient optimization. The columns LB and UB show the lower and upper bounds obtained with the Lagrangian approach described above. Although the Lagrangian approach was not able to close the still existing gaps, are the results encouraging. Given that the computation time to solve linear programming relaxations is usually prohibitive for large instances, the computation times shown in Tab. 3 can still be ranked as moderate. (We refer to the comparison of computation times in Tab. 1, and also to [CDS+], who could solve only small problem instances due to enormous computation times.) Moreover, the feasible solutions

Table3. Results for sample instances of BASF-type with one resource type (manpower) and variable resource requirements of jobs (see Fig. 1). Computation times have been obtained on a Sun Ultra 10 with 330 MHz, operating under Solaris.

Instance	#jobs	#tasks	crit. path	T	LB	UB	CPU	#it
4o_24j_B_18	24	109	54	75	55	72	1 sec.	81
6o_44j_A_18	44	224	75	128	88	121	6 sec.	169
10o_87j_A_18	87	1001	194	633	412	595	8 min.	198
10o_106j_A_18	106	1653	383	1224	748	1126	33 min.	239

given in Tab. 3 improve upon results obtained with constraint propagation by Heipcke [Hei], and compare to results with a tabu search algorithm by Cavalcante and de Souza [CD,CDS+]. It is worth to note, however, that in comparison to tabu search, the Lagrangian approach provides both a lower bound and a feasible solution at the same time.

6 Time-Critical Tasks and Maximal Time Lags

Until now, we have assumed that the temporal restrictions are given by arbitrary minimal time lags $d_{ij} \geq 0$ between any two jobs i and j, and the sequence of tasks of a job had to be processed without interruption. An additional feature which would be 'nice to have' in some practical applications is the possibility of modeling also *maximal time distances* between certain jobs or tasks. The motivation is that the execution of certain tasks may be postponed in time, but must not be postponed too much, for instance because the temperature must not fall below a certain threshold. Such constraints can be easily anticipated by introducing *negative* time lags between jobs, or tasks, where a time lag $d_{ji} < 0$ now implies a maximal time lag of S_j relative to S_i. Hence, so-called *time windows* of the form $S_i + d_{ij} \leq S_j \leq S_i - d_{ji}$ between any two jobs (or tasks) can be modeled.

Unfortunately, this has the effect that the problem of finding a feasible solution already is NP-complete (see, e.g., [BMR]). Although the computation of lower bounds via Lagrangian relaxation as described in Sections 2 and 3 remains valid also if maximal time lags are present, the approach described in Sect. 5 to compute feasible solutions can no longer be applied. This is due to the fact that this approach is based upon list-scheduling algorithms which generally fail to find feasible solutions if maximal time lags exist. Hence, we have implemented a *branch-and-bound* algorithm for this problem (see [FMSU]). The underlying idea is that the problem is easy solvable via longest path calculations if resource-constraints are absent. Thus, the resource-constraints are relaxed, and within an enumeration tree so-called *resource conflicts* are *resolved* by introducing additional temporal restrictions. A resource conflict is a time interval where a schedule consumes more resources than available, thus violating inequalities (4). In contrast to previous approaches by Bartusch, Möhring, and Radermacher [BMR], de Reyck and Herroelen [dRH], or Schwindt [Sch2], we implemented a branch-and-bound algorithm which uses earliest start times (or *release dates*) to resolve a resource conflict. This has the effect that the necessary computations in every node of the enumeration tree can be realized very efficiently. We do not go into details here, but refer to [FMSU].

Also with this branch-and-bound algorithm, we could optimally solve the above described instance provided by BASF AG within less than a minute computation time. Applied to the notoriously difficult benchmark instances from [Cav], we could solve to optimality most of the smaller instances, however, for large instances the results with the Lagrangian approach were much better. (Note that these instances do not involve maximal time lags.)

Additionally, we tested this algorithm using a benchmark set of 1080 instances which involve also maximal time lags. The instances have been generated by Schwindt [Sch1], and consist of 100 jobs each. Tab. 4 compares

Table4. Comparison of the performance of branch-and-bound algorithm [FMSU] with previous branch-and-bound algorithms, based upon 1059 ProGen/max instances [Sch1]. Computations for [FMSU] have been obtained on a Sun Ultra 2 with 200 MHz, operating under Solaris. Computations for [dRH] and [Sch2], however, have been conducted in a different environment.

Algorithm	Time limit	Optimized	Feasible	Av. Dev.
De Reyck, Herroelen [dRH]	30 s*	57.5%	93.6%	10.0%**
Schwindt [Sch2]	30 s	63.7%	100%	7.0%
	100 s	64.7%	100%	6.9%
Fest, Möhring, Stork, Uetz [FMSU]	30 s	70.6%	100%	7.8%
	100 s	72.8%	100%	6.8%

*Corresponds to 100s on a 60MHz personal computer.
**Average deviation based upon different lower bounds.

the results of our implementation with previous branch-and-bound algorithms which also follow the paradigm to resolve resource conflicts. All algorithms have been run for a certain time limit (30 and 100 sec., respectively), for each of the 1080 instances. The figures refer to 1059 instances only, since the remaining 21 do not have a feasible solution. The table shows the number of instances that could be solved to optimality, as well as the number of instances where a feasible solution could be found within the given time limit. The last column shows the total average deviation from lower bounds, based upon a benchmark set of lower bounds obtained from [Pro2]. In terms of the average deviation, the results we obtained in [FMSU] could be further improved using constraint propagation by Dorndorf, Pesch, and Phan Huy in [DPP].

Concerning lower bounds for the ProGen/max instances, it is worth to mention that the Lagrangian approach could improve upon the best known lower bounds for 91 instances. We refer to [MSSU3] for more details.

7 Final Remarks and Outlook

We have implemented and tested two different approaches to solve resource-constrained scheduling problems which arise in a typical production process at BASG AG, Ludwigshafen. On the one hand, this is a Lagrangian approach which is suited for a whole variety of different resource-constrained scheduling problems, and hence is of practical interest not only within chemical engineering. On the other hand, this is a branch-and-bound algorithm which handles also arbitrary maximal time lags, or so-called *time windows* between jobs (or tasks). Computational experiments with practical instances from BASF AG as well as benchmark test sets were quite successful with both algorithms.

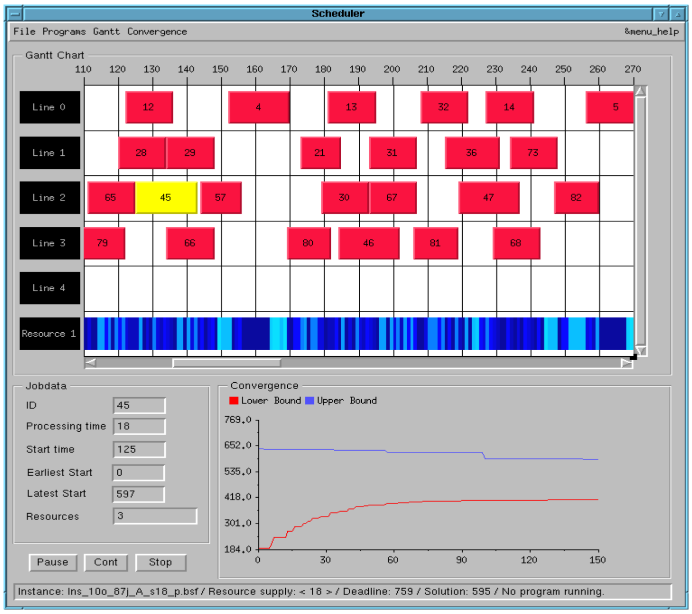

Figure4. Screen-shot of the user interface (based on instance 10o_87j_A; see Tab. 3). Jobs are red rectangles on the time axis, and the blue bottom line visualizes resource utilization, where dark blue indicates heavy utilization. The two curves show the lower and upper bound improvements during the course of the algorithm.

Apart from the algorithmic side, we have also implemented a user interface which allows to display both solutions as well as problem instances, and which enables the user to manipulate the problem instances on-line. Fig. 4 shows a screen-shot of the schedule visualization. In the meantime, several software companies indicated their interest to integrate the Lagrangian-based algorithm into their products, and on the account of the present project a cooperation on a related topic was established with ATOSS Software AG.

Acknowledgments. We would like to thank Josef Kallrath and Anna Schreieck from BASF AG for their cooperation, and Lars Stolletz and Carola Schaad for their contribution to our implementation of the algorithms.

References

[BK] P. Brucker and S. Knust, *A linear programming and constraint propagation-based lower bound for the RCPSP*, Tech. Report 204 (revised 1999), Department of Mathematics, University of Osnabrück, 1999.

[BMR] M. Bartusch, R. H. Möhring, and F. J. Radermacher, *Scheduling project networks with resource constraints and time windows*, Annals of Operations Research **16** (1988), 201–240.

[Cav] `http://www.dcc.unicamp.br/~cris/SPLC.html`, 1997.

[CAT] N. Christofides, R. Alvarez-Valdés, and J. M. Tamarit, *Project scheduling with resource constraints: A branch and bound approach*, European Journal of Operational Research **29** (1987), 262–273.

[CD] C. C. B. Cavalcante and C. C. De Souza, *A tabu search approach for scheduling problems under labour constraints*, Tech. Report IC-97-13, Instituto de Computação, UNICAMP, Campinas, Brazil, 1997.

[CDS+] C. C. B. Cavalcante, C. C. De Souza, M. W. P. Savelsbergh, Y. Wang, and L. A. Wolsey, *Scheduling projects with labor constraints*, CORE Discussion Paper 9859, Université Catholique de Louvain, Louvain-la-Neuve, Belgium, 1998.

[CE] G.J. Chang and J. Edmonds, *The poset scheduling problem*, Order **2** (1985), 113–118.

[dRH] B. de Reyck and W. Herroelen, *A branch and bound procedure for the resource-constrained project scheduling problem with generalized precedence relations*, European Journal of Operational Research **111** (1998), 152–174.

[DPP] U. Dorndorf, E. Pesch, and T. Phan-Huy, *A time-oriented branch-and-bound algorithm for resource-constrained project scheduling with generalised precedence constraints*, Management Science **46** (2000), 1365–1384.

[FMSU] A. Fest, R. H. Möhring, F. Stork, and M. Uetz, *Resource constrained project scheduling with time windows: A branching scheme based on dynamic release dates*, Tech. Report 596/1998 (revised 1999), Fachbereich Mathematik, Technische Universität Berlin, Germany, 1999. Submitted.

[GT] Andrew V. Goldberg and Robert E. Tarjan, *A new approach to the maximum-flow problem*, Journal of the Association for Computing Machinery **35** (1988), 921–940.

[Hei] S. Heipcke, *Combined modelling and problem solving in mathematical programming and constraint programming*, Ph.D. thesis, School of Business, University of Buckingham, U.K., 1999.

[HSW] L. A. Hall, D. B. Shmoys, and J. Wein, *Scheduling to minimize average completion time: Off-line and on-line algorithms*, Proc. of the 7th ACM–SIAM Symposium on Discrete Algorithms (Atlanta, Georgia), 1996, pp. 142–151.

[KS] R. Kolisch and A. Sprecher, *PSPLIB - A project scheduling problem library*, European Journal of Operational Research **96** (1996), 205–216.

[KW] J. Kallrath and J. M. Wilson, *Business optimisation using mathematical programming*, Macmillan Business, London, U.K., 1997.

[MSSU1] R. H. Möhring, A. S. Schulz, F. Stork, and M. Uetz, *Resource-constrained project scheduling: Computing lower bounds by solving minimum cut problems*, Proc. of the 7th Annual European Symposium on Algorithms (Prague, Czech Republic), J. Nešetřil (ed.), Lecture Notes in Computer Science, vol. 1643, Springer, 1999, pp. 139–150.

[MSSU2] R. H. Möhring, A. S. Schulz, F. Stork, and M. Uetz, *On project scheduling with irregular starting time costs*, Tech. Report 664, Technische Universität Berlin, Germany, 2000, Submitted.

[MSSU3] R. H. Möhring, A. S. Schulz, F. Stork, and M. Uetz, *Solving Project Scheduling Problems by Minimum Cut Computations*, Tech. Report 680, Technische Universität Berlin, Germany, 2000, Submitted. Extended abstracts appeared in [MSSU1] and [SU].

[Pro1] `ftp://ftp.bwl.uni-kiel.de/pub/operations-research/psplib/HTML/data.html`, January 2000.

[Pro2] `ftp://ftp.wior.uni-karlsruhe.de/public/ProGen-max/pspMaxLib/RCPSPMax/benchmarkC+D.txt`, May 2000.

[PSW] C. A. Phillips, C. Stein, and J. Wein, *Scheduling jobs that arrive over time*, Proc. of the 4th International Workshop on Algorithms and Data Structures (Kingston, Ontario), S. G. Akl, F. Dehne, J.-R. Sack, and N. Santoro (eds.), Lecture Notes in Computer Science, vol. 955, Springer, 1995, pp. 86–97.

[PWW] A. A. B. Pritsker, L. J. Watters, and P. M. Wolfe, *Multi project scheduling with limited resources: A zero-one programming approach*, Management Science **16** (1969), 93–108.

[Rus] A. H. Russel, *Cash flows in networks*, Management Science **16** (1970), 357–373.

[Sch] M. Schäffter, *Scheduling with respect to forbidden sets*, Discrete Applied Mathematics **72** (1997), 141–154.

[Sch1] C. Schwindt, *Generation of resource constrained project scheduling problems with minimal and maximal time lags*, Tech. Report 489, WIOR, University of Karlsruhe, Germany, 1996.

[Sch2] C. Schwindt, *A branch-and-bound algorithm for the resource-constrained project duration problem subject to temporal constraints*, Tech Report 544, WIOR, University of Karlsruhe, Germany, 1998.

[SU] F. Stork and M. Uetz, *Resource-constrained project scheduling: From a Lagrangian relaxation to competitive solutions*, Proc. of the 7th International Workshop on Project Management and Scheduling (Osnabrück, Germany), 2000, pp. 254–257.

[SUW] M. W. P. Savelsbergh, R. N. Uma, and J. Wein, *An experimental study of LP-based approximation algorithms for scheduling problems*, Proc. of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (San Francisco, California), 1998, pp. 453–462.

Reports from the group

"Combinatorial Optimization and Graph Algorithms"

of the Department of Mathematics, TU Berlin

702/2000 *Frederik Stork:* Branch-and-Bound Algorithms for Stochastic Resource-Constrained Project Scheduling

700/2000 *Rolf H. Möhring:* Scheduling under uncertainty: Bounding the makespan distribution

698/2000 *Sándor P. Fekete, Ekkehard Köhler, and Jürgen Teich:* More-dimensional packing with order constraints

697/2000 *Sándor P. Fekete, Ekkehard Köhler, and Jürgen Teich:* Extending partial suborders and implication classes

696/2000 *Sándor P. Fekete, Ekkehard Köhler, and Jürgen Teich:* Optimal FPGA module placement with temporal precedence constraints

695/2000 *Sándor P. Fekete, Henk Meijer, André Rohe, and Walter Tietze:* Solving a "hard" problem to approximate an "easy" one: heuristics for maximum matchings and maximum Traveling Salesman Problems

694/2000 *Esther M. Arkin, Sándor P. Fekete, Ferran Hurtado, Joseph S. B. Mitchell, Marc Noy, Vera Sacristánm and Saurabh Sethia:* On the reflexivity of point sets

693/2000 *Frederik Stork and Marc Uetz:* On the representation of resource constraints in project scheduling

691/2000 *Martin Skutella and Marc Uetz:* Scheduling precedence constrained jobs with stochastic processing times on parallel machines

689/2000 *Rolf H. Möhring, Martin Skutella, and Frederik Stork:* Scheduling with AND/OR precedence constraints

685/2000 *Martin Skutella:* Approximating the single source unsplittable min-cost flow problem

684/2000 *Han Hoogeveen, Martin Skutella, and Gerhard J. Woeginger:* Preemptive scheduling with rejection

683/2000 *Martin Skutella:* Convex quadratic and semidefinite programming relaxations in Scheduling

682/2000 *Rolf H. Möhring and Marc Uetz:* Scheduling scarce resources in chemical engineering

681/2000 *Rolf H. Möhring:* Scheduling under uncertainty: optimizing against a randomizing adversary

680/2000 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* Solving project scheduling problems by minimum cut computations (Journal version for the previous Reports 620 and 661)

674/2000 *Esther M. Arkin, Michael A. Bender, Erik D. Demaine, Sándor P. Fekete, Joseph S. B. Mitchell, and Saurabh Sethia:* Optimal covering tours with turn costs

669/2000 *Michael Naatz:* A note on a question of C. D. Savage

667/2000 *Sándor P. Fekete and Henk Meijer:* On geometric maximum weight cliques

666/2000 *Sándor P. Fekete, Joseph S. B. Mitchell, and Karin Weinbrecht:* On the continuous Weber and k-median problems

664/2000 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* A note on scheduling problems with irregular starting time costs

661/2000 *Frederik Stork and Marc Uetz:* Resource-constrained project scheduling: from a Lagrangian relaxation to competitive solutions

658/1999 *Olaf Jahn, Rolf H. Möhring, and Andreas S. Schulz:* Optimal routing of traffic flows with length restrictions in networks with congestion

655/1999 *Michel X. Goemans and Martin Skutella:* Cooperative facility location games

654/1999 *Michel X. Goemans, Maurice Queyranne, Andreas S. Schulz, Martin Skutella, and Yaoguang Wang:* Single machine scheduling with release dates

653/1999 *Andreas S. Schulz and Martin Skutella:* Scheduling unrelated machines by randomized rounding

646/1999 *Rolf H. Möhring, Martin Skutella, and Frederik Stork:* Forcing relations for AND/OR precedence constraints

640/1999 *Foto Afrati, Evripidis Bampis, Chandra Chekuri, David Karger, Claire Kenyon, Sanjeev Khanna, Ioannis Milis, Maurice Queyranne, Martin Skutella, Cliff Stein, and Maxim Sviridenko:* Approximation schemes for minimizing average weighted Completion time with release dates

639/1999 *Andreas S. Schulz and Martin Skutella:* The power of α-points in preemptive single machine scheduling

634/1999 *Karsten Weihe, Ulrik Brandes, Annegret Liebers, Matthias Müller–Hannemann, Dorothea Wagner and Thomas Willhalm:* Empirical design of geometric algorithms

633/1999 *Matthias Müller–Hannemann and Karsten Weihe:* On the discrete core of quadrilateral mesh refinement

632/1999 *Matthias Müller–Hannemann:* Shelling hexahedral complexes for mesh generation in CAD

631/1999 *Matthias Müller–Hannemann and Alexander Schwartz:* Implementing weighted b-matching algorithms: insights from a computational study

629/1999 *Martin Skutella:* Convex quadratic programming relaxations for network scheduling problems

628/1999 *Martin Skutella and Gerhard J. Woeginger:* A PTAS for minimizing the total weighted completion time on identical parallel machines

624/1999 *Rolf H. Möhring:* Verteilte Verbindungssuche im öffentlichen Personenverkehr: Graphentheoretische Modelle und Algorithmen

627/1998 *Jens Gustedt:* Specifying characteristics of digital filters with FilterPro

620/1998 *Rolf H. Möhring, Andreas S. Schulz, Frederik Stork, and Marc Uetz:* Resource constrained project scheduling: computing lower bounds by solving minimum cut problems

619/1998 *Rolf H. Möhring, Martin Oellrich, and Andreas S. Schulz:* Efficient algorithms for the minimum-cost embedding of reliable virtual private networks into telecommunication networks

618/1998 *Friedrich Eisenbrand and Andreas S. Schulz:* Bounds on the Chvátal rank of polytopes in the 0/1-Cube

617/1998 *Andreas S. Schulz and Robert Weismantel:* An oracle-polynomial time augmentation algorithm for integer proramming

616/1998 *Alexander Bockmayr, Friedrich Eisenbrand, Mark Hartmann, and Andreas S. Schulz:* On the Chvátal rank of polytopes in the 0/1 cube

615/1998 *Ekkehard Köhler and Matthias Kriesell:* Edge-dominating trails in AT-free graphs

613/1998 *Frederik Stork:* A branch and bound algorithm for minimizing expected makespan in stochastic project networks with resource constraints

612/1998 *Rolf H. Möhring and Frederik Stork:* Linear preselective policies for stochastic project scheduling

611/1998 *Rolf H. Möhring and Markus W. Schäffter:* Scheduling series-parallel orders subject to 0/1-communication delays

609/1998 *Arfst Ludwig, Rolf H. Möhring, and Frederik Stork:* A computational study on bounding the makespan distribution in stochastic project networks

605/1998 *Friedrich Eisenbrand:* A note on the membership problem for the elementary closure of a polyhedron

596/1998 *Andreas Fest, Rolf H. Möhring, Frederik Stork, and Marc Uetz:* Resource constrained project scheduling with time windows: A branching scheme based on dynamic release dates

595/1998 *Rolf H. Möhring Andreas S. Schulz, and Marc Uetz:* Approximation in stochastic scheduling: The power of LP-based priority policies

591/1998 *Matthias Müller–Hannemann and Alexander Schwartz:* Implementing weighted b-matching algorithms: Towards a flexible software design

590/1998 *Stefan Felsner and Jens Gustedt and Michel Morvan:* Interval reductions and extensions of orders: bijections to chains in lattices

584/1998 *Alix Munier, Maurice Queyranne, and Andreas S. Schulz:* Approximation bounds for a general class of precedence constrained parallel machine scheduling problems

577/1998 *Martin Skutella:* Semidefinite relaxations for parallel machine scheduling

Reports may be requested from:

Hannelore Vogt-Möller
Fachbereich Mathematik, MA 6–1
TU Berlin
Straße des 17. Juni 136
D-10623 Berlin – Germany

e-mail: moeller@math.TU-Berlin.DE

Reports are also available in various formats from

`http://www.math.tu-berlin.de/coga/publications/techreports/`

and via anonymous ftp as

`ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-`*number*`-`*year*`.ps`