On the benefits of using NP-hard problems in Branch & Bound

Jörg Rambau and Cornelius Schwarz

University of Bayreuth, Bayreuth, Germany
{joerg.rambau,cornelius.schwarz}@uni-bayreuth.de

Summary. We present a Brand-and-Bound (B&B) method using combinatorial bounds for solving makespan minimization problems with sequence dependent setup costs. As an application we present a laser source sharing problem arising in car manufacturing.

1 Introduction

Some car manufactures use laser welding technology for the assembly of car bodys. The equipment in a welding cell consists of a number of welding robots and one or more laser sources, each of which can supply more than one robot, but only one at a time. In usual settings only a small fraction of the process time is spent with welding. This motivates the idea of sharing laser sources between robots. Because production cycle times must not be exceeded, the question is: "How many laser sources are needed to process a given set of welding tasks with a given set of robots in a given time?" To answer this question, we propose the *Laser Source Sharing Problem (LSP)*: Given a set of robots, a set of welding tasks and a set of laser sources, find a *scheduled tour* (i.e., an order of job start and end points together with start and end times) for each welding robot and an assignment of robots to laser sources so that

- all jobs are served,
- robots assigned to identical laser sources never weld simultaneously,
- the makespan is minimized.

This problem was introduced in [6], where a mixed-integer model for the special case of fixed robot tours was developed. An extension to integrate tour optimization was proposed in [8], but could not be solved on real-world scales (3–6 robots, 1–3 sources, ≈ 30 jobs).

When we drop the resource sharing constraint, we obtain a *vehicle routing problem (VRP)* with makespan minimization. Classical exact approaches to solve large VRPs use column generation in mixed-integer models, see [4] or [7]. However: The makespan objective yields large integrality gaps in the Master Problem, and – because of few servers for many jobs – the columns are dense.

We propose a combinatorial B&B algorithm based on partial schedules. Such algorithms are common in project scheduling, and a key problem is to find good lower bounds. Most lower bound constructions in project scheduling are based on precedence constraints, for instance critical paths calculations, see [2] or [3]. Since our problem does not contain precedence constraints, we follow a different method.

Our contribution is a new B&B algorithm that solves NP-hard sub-TSPs, which provide much better bounds than LP relaxations of common mixed-integer models. This is the first algorithm that can solve industrial-scale LSP-instances to proven optimality. Since estimating real robot driving times is a non trivial practical problem, all computations had to be done with artificial data, generated from real-world welding plans, though. We are currently working on providing more realistic data using KuKa SimPro. Moreover, collision avoidance is not yet part of the algorithm but can and will be integrated later.

We believe that bounds from the solutions of NP-hard subproblems may also be helpful for other makespan minimization problems.

2 Problem definition

Let R be a set of robots, J a set of jobs and L a set of laser sources. Each robot $r \in R$ has a nullposition o_r, where the tour has to start and to end. Each job $j \in J$ has two end positions j_a, j_b. If the service of a job starts at j_a it has to finish at j_b, and vice versa. Let p_j be the processing time of Job $j \in J$. We denote the driving time of Robot r from Positions q_i to q_j by $\delta_r(q_i, q_j)$. We also introduce a latency δ_l for laser sources. When l switches robots then there is a delay of δ_l.

The task is to assign each $j \in J$ to a robot $r \in R$, each robot $r \in R$ to a laser source $l \in L$, and to create a scheduled tour for every robot through all assigned jobs so that

- each job is assigned to exactly one robot,
- each robot is assigned to exactly one laser source,
- jobs assigned to robots sharing a laser source do not overlap in time.

The cost of a scheduled tour is the time length, i.e., the time when the robot finishes its tour at o_r. The goal is to minimize the makespan, which is the maximum over the tour costs. If we restrict to one robot (the *1-server problem*) and set for all jobs $j_a = j_b, p_j = 0$ we get a TSP, which is NP-hard. Thus, the laser sharing problem is also NP-hard.

The LSP can be interpreted as a *vehicle routing problem*, where vehicles correspond to robots. The task is to find a route for every vehicle with minimum makespan subject to the resource constraints. From a scheduling point of view we can interpret the robots as machines, resulting in a parallel-machine scheduling problem with sequence dependent setup costs. The laser sources are resources with the condition that each machine can only use a unique resource.

3 The algorithm

We already showed that the TSP is a special case of our problem. Since TSPs of the usual scale of the LSP (around 30 jobs) are relatively easy to solve nowadays, we can use TSPs as relaxations. In the next section we will see that this yields better and faster bounds than LP relaxations of mixed-integer-models of the LSP.

Assume that an assignment $\mathcal{J} : R \to 2^J$ of robots to jobs and an assignment $\mathcal{L} : R \to L$ of laser sources to robots are fixed. The resulting problem is called *LSP($\mathcal{J}$,$\mathcal{L}$)*. If resource constraints are neglected, then we can solve the 1-server problems separately by auxiliary TSPs, see [5]. The duration (in the LSP) of any tour t will be denoted by $\ell(t)$. The set of jobs served in Tour t is denoted by $J(t)$.

LSP($\mathcal{J}$,$\mathcal{L}$) can now be solved as follows: Assume that for each robot r we are given a partial scheduled tour t_r ending in q_r with duration $\ell(t_r)$. Then no scheduled tour starting with t_r visiting all jobs in $\mathcal{J}(r)$ can finish earlier than the concatenation of t_r and an optimal TSP tour $t_r^{\mathrm{TSP}}(K_r, q_r)$ starting at q_r, visiting all jobs in $K_{t_r} := \mathcal{J}(r) \setminus J(t_r)$, and ending at o_r. Thus, a lower bound of LSP($\mathcal{J}$,$\mathcal{L}$) with given scheduled prefix tours $(t_r)_{r\in R}$ is given by $\max_{r\in R}\big(\ell(t_r) + \ell(t_r^{\mathrm{TSP}}(K_{t_r}, q_r))\big)$. Now we can solve LSP($\mathcal{J}$,$\mathcal{L}$) using B&B with a node for each $(t_r)_{r\in R}$ and child nodes corresponding to all single-job extensions of a single t_r.

We summarize the algorithm for LSP($\mathcal{J}$,$\mathcal{L}$): For $r \in R$, a set of jobs K, and a start position q we denote by $\mathrm{TSP}_r(K, q)$ a call to an exact TSP oracle solving the 1-server problem of Robot r starting at Position q, processing all jobs in K, and ending at o_r. The set $T^{\mathrm{TSP}} := \{t_r^{\mathrm{TSP}} \mid r \in R\}$ stores the solutions of the 1-server problems. Similarly, $T := (t_r)_{r\in R}$ keeps a partial scheduled tour t_r for every robot. We write

μ for the best upper bound and λ for the lower bound of the current node:

Algorithm 1 (Combinatorial Branch-and-Bound for LSP($\mathcal{J}$,$\mathcal{L}$))

INPUT: Data of LSP($\mathcal{R}$,$\mathcal{L}$)
OUTPUT: A set T_{OPT} with optimal scheduled tours $\{t_r\}_{r\in R}$.

1. initialize: $t_r := ()$ for all $r \in R$
2. set $\mu := \infty$, $T_{\mathrm{OPT}} := T := \big(()\big)_{r\in R}$ *// empty tours*
3. CBB(T)
4. return T_{OPT}

Procedure CBB(T) for $T = (t_r)_{r\in R}$:

1. for all $r \in R$,
   set q_r to the last position of t_r or o_r if $t_r = ()$ and set
   $$t_r^{\mathrm{TSP}} := \mathrm{TSP}_r(\mathcal{J}(r) \setminus J(t_r), q_r)$$
2. set $\lambda := \max_{r\in R}\big(\ell(t_r) + \ell(t_r^{\mathrm{TSP}})\big)$
   // length of partial scheduled tour t_r completed with t_r^{TSP}
3. if $\lambda > \mu$ return *// pruning*
4. if $J(t_r) = \mathcal{J}(r)$ *// all jobs are scheduled*
   a) complete the tours, i.e., append o_r to t_r $\forall\ r \in R$
   b) set $\mu_{\mathrm{new}} := \max_{r\in R} \ell(t_r)$ *// makespan of current solution*
   c) if $\mu_{\mathrm{new}} < \mu$, set $T_{\mathrm{OPT}} := T$, $\mu := \mu_{\mathrm{new}}$ *// new best solution*
   d) return
5. for all $r \in R$, $j \in \mathcal{J}(r) \setminus J(t_r)$, $(q_{\mathrm{start}}, q_{\mathrm{end}}) \in \{(j_a, j_b), (j_b, j_a)\}$
   // run through all not yet scheduled jobs
   a) append $(q_{\mathrm{start}}, q_{\mathrm{end}})$ to t_r
      // Job j will be welded next from q_{start} to q_{end}
   b) set the start time of j to
      $$\max\{\ell(t_r) + \delta_r(q_r, q_{\mathrm{start}}),\ \max_{\substack{s\in \mathrm{R}\setminus r \\ \mathcal{L}(s)=\mathcal{L}(r)}} \ell(t_s) + \delta_{\mathcal{L}(s)}\}$$
      // earliest time so that r can reach the job
      // and so that the laser source is available again
   c) CBB(T)
   d) remove j from t_r

Remark 1 *At any time, we can pipe the tour information given by the TSP call into a scheduling heuristic for fixed tours, e.g., [6], which gives us a feasible solution for the LSP. This primal information is not available from LP-relaxations.*

If the assignments $\mathcal{J}$ and $\mathfrak{L}$ are prescribed, then we end up in a cluster-first-schedule-second approach, which yields an optimal solution whenever we guess the "right" assignments. Since $|R|$ and $|L|$ are small, we can enumerate all $\mathfrak{L}$s. The number of potential job-robot assignments, however, is too big for a naive enumeration. However, we can reduce this big set to a small set of candidates without missing an optimal assignment. To this end, we branch over partial job-server assignments. Whenever we reach a leaf, we run an heuristic scheduling algorithm to generate a feasible solution using only one laser source. The value of this solution provides an upper bound for LSP. The lower bounds in the nodes are obtained by solving the 1-server TSPs for the partially assigned jobs. Every leaf with a lower bound not worse than the best global upper bound is a candidate. Finally, the LSP can now be solved exactly by solving the LSP($\mathcal{J}$,$\mathfrak{L}$) for all candidate assignments.

4 Computational Results

We now compare the lower bounds to the LP relaxations of two mixed-integer-models. The first one is an improved version of [8] which uses linear ordering variables and many big-M constraints. Since good LP-bounds in scheduling often come from time indexed variables we also compare to a model based on a time expanded networks.

Our test instances consist of randomly selected jobs of a real welding plan from a car manufacturer with three robots. Unfortunately, it is a non trivial practical problem to get real robot driving times. We used Euclidean distances on the 2D projection of the welding points as an approximation. The comparison was done on a Intel Core 2 Duo processor with 3 Ghz and 4 GB memory running Ubuntu Linux 8.04 in 64 bit mode. For the LP relaxations we used Ilog Cplex 11.1 (barrier for time discrete networks and dual simplex for linear orderings). The TSP relaxations were solved by `concorde` [1].

In the following table the lower bounds from various root relaxations for typical instances of $LSP(\mathfrak{L}, \mathcal{J})$ with given optimal assignments are listed. The respective optima were calculated by our method. For non-optimal assignments, LP-relaxations are no better.

problem	lin. ordering		time-exp. netw.		TSP		optimum
	cpu/s	value	cpu/s	value	cpu/s	value	
10jobs	0.08	24.7	24.5	24.7	0.02	24.7	24.7
16jobs	0.02	18.0	200.0	17.1	0.02	20.3	20.3
18jobs	0.09	20.0	282.8	19.1	0.02	22.7	22.8
20jobs	0.03	20.0	463.3	19.1	0.02	23.0	23.0
34jobs	0.11	24.7	2605.9	31.4	0.07	31.4	31.4

We see that, for a use in our B&B, linear ordering relaxations are too weak for the large instance and time expanded network relaxations are way too slow. The TSP relaxation is the strongest and the fastest throughout and was the only one with which the full B&B for the 34jobs-LSP could finish within a couple of hours.

5 Conclusions

We showed that NP-hard subproblems have the power to provide much better bounds in B&B algorithm than classical LP based approaches. The key lies inside the problem scale: A large-scale for the original problem (here: LSP) may be small for the subproblem (here: TSP). It remains to verify the method on real-world welding data and to integrate collision avoidance in the B&B.

References

1. D.L. Applegate, R.E. Bixby, V. Chvátal, and W.J. Cook. *The Traveling Salesman Problem – A Computational Study.* Princeton University Press, 2006.
2. P. Brucker and S. Knust. *Complex Scheduling.* Springer-Verlag, Berlin, Heidelberg, New York, 2006.
3. E. Demeulemeester. *Optimal Algorithms for various classes of multiple resource-CPCPs.* PhD thesis, Katholik Universiteit Leuven, 1996.
4. J. Desroisers, Y. Dumas, M.M. Solomon, and F. Soumis. Time constraint routing and scheduling. In M. Ball, T.L. Magnanti, C.L. Monma, and G. Nemhauser, editors, *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, pages 35–140. Elsevier, Amsterdam, 1995.
5. M. Dror, editor. *Arc Routing. Theory, Solutions and Applications.* Springer-Verlag, 2001.
6. M. Grötschel, H. Hinrichs, K. Schröer, and A. Tuchscherer. Ein gemischt-ganzzahliges lineares Optimierungsproblem für ein Laserschweißproblem im Karosseriebau. *Zeitschrift für wissenschaftlichen Fabrikbetrieb*, 5:260–264, 2006.
7. S.O. Krumke, J. Rambau, and L.M. Torres. Realtime dispatching of guided and unguided automobile service units with soft time windows. In R. Möhring et al., editor, *Algorithms – ESA 2002*, volume 2461 of *LNCS*, pages 637–648. Springer, 2002.
8. T. Schneider. Ressourcenbeschränktes Projektscheduling zur optimierten Auslastung von Laserquellen im Automobilkarosseriebau. Diplomarbeit, University of Bayreuth, 2006.