A heuristic neighbourhood search-based algorithm for the solution of resource-task network scheduling problems

Giancarlo Dalle Ave^{a,b}, Xing Wang^{a,c}, Iiro Harjunkoski^{a,*} and Sebastian Engell^b

^aABB AG Corporate Research Center, Wallstadter Str. 59, 68526 Ladenburg, Germany

^bProcess Dynamics and Operations Group, Department of Biochemical and Chemical Engineering, Technische Universität Dortmund, Emil-Figge-Str. 70, 44221 Dortmund, Germany

^cBusiness School University of Mannheim, Schloss Schneckenhof Ost SO 230, 68131 Mannheim, Germany

Iiro.Harjunkoski@de.abb.com

Abstract

There are many challenges facing the widespread industrial deployment of scheduling solutions. These include to find a generic scheduling model that can be applied to a broad variety of scenarios, and to design algorithms that can provide good quality solutions in industrially relevant time frames. The discrete-time Resource-Task Network (RTN) is a generic scheduling framework that has been successfully applied to many different problems. However, RTN models can quickly become intractable as problem size increases. In this work, an iterative neighbourhood-search based algorithm is developed in order to speed-up the solution of RTN scheduling models. The speed-up is achieved by limiting the binary variables of a subproblem to a specific neighbourhood in a way that allows the algorithm to iteratively move towards better solutions. Results show that this approach is able to tackle problems that are intractable for a full-space model while still finding near optimal solutions.

Keywords: Resource-task network (RTN), Heuristic, Neighbourhood search, Iterative scheduling algorithm

1. Introduction

Some of the challenges facing the use of scheduling in industry are: to find a generic scheduling framework and efficient solution algorithms, which can provide good quality solutions that can be used in practice (Harjunkoski et al. (2014)). The discrete-time Resource-Task Network (RTN) formulation has been shown to be applicable to various industrial applications with small application-specific configuration effort required. Such applications include, stainless steel-making (Castro et al. (2013)), thermo-mechanical pulping (Hadera et al. (2015)), and petrochemical production (Wassick and Ferrio (2011)). In the direction of deploying industrial scheduling solutions, Harjunkoski and Bauer (2014) suggested the use of the ISA-95 standard as a way to efficiently communicate between the layers of the so-called automation pyramid.

The advantages of the RTN formulation and of mixed-integer linear programming (MILP) scheduling solutions in general are that they provide optimal solutions to problems with complex objective functions. The downside of MILPs is that solution time often increases dramatically with problem size, to the point where even a feasible solution cannot be guaranteed. Conversely, constructive

scheduling heuristic algorithms have fast execution times and scale very well with problems size; however, they give no indication of optimality and cannot handle competing objectives.

It is therefore desirable to combine the speed and scalability of heuristics with the mathematical rigor of MILPs. Works in this area include the sequential updating procedures proposed by Roslöf et al. (2001) which updates an initial heuristic-based solution by iteratively releasing a set of jobs and optimally inserting them into the existing job sequence. Other options include the use of pre-ordering rules to eliminate sequencing variables (Méndez and Cerdá (2002)) or to use heuristics to limit the search space of a problem and to use that solution to initialize a full-space model as in Panek et al. (2005). Alternatively, Castro et al. (2011) proposed a greedy algorithm to insert orders into a schedule a few jobs at a time, while allowing for partial rescheduling at each iteration.

In this work, an algorithm is proposed that takes advantage of heuristics to assist RTN-based scheduling solutions to find good solutions with comparatively small computational effort. This is accomplished by combining the two approaches of MILP and heuristic solutions in a novel iterative neighbourhood-based approach. In this algorithm, a heuristic is used to provide an initial feasible solution which is then used to constrain the binary variables of RTN scheduling models in a way that greatly reduces the computation time of the MILP, while still allowing it to move towards better solutions.

2. RTN Formulation

The proposed approach is based on the discrete-time RTN framework to make it generic and easy to adapt to various applications. This section will briefly present the RTN-scheduling model as well as specific modeling constraints and scenarios that are necessary to make the model even more flexible, at the expense of computational efficiency.

The RTN formulation is generic as it represents the entire scheduling model as a set of resources (R) and tasks (I). In this formulation, tasks consume and produce sets of resources. Examples of resources include raw materials, final products, and processing units. In the discrete-time RTN formulation, the scheduling horizon H is divided into a set of δ sized intervals T . Tasks ($i \in I$) are characterized by two sets of variables, the binary variable $N_{i,t}$ and the continuous variable $\xi_{i,t}$. The binary variable identifies that task i starts at time point $t \in T$ while the continuous variable indicates the amount of a resource handled by the task. The parameter τ_i is the duration of task i and $\mu_{i,r,\theta}$ and $v_{i,r,\theta}$ represent the interaction of task i on resource r based on the occurrence (defined by $N_{i,t}$) and continuous (defined by $\xi_{i,t}$) extent of the task respectively. The resource consumption over the time horizon is tracked by the excess resource balance given in Equation (1).

$$R_{r,t} = R_{r,t-1} + \sum_i \sum_{\theta=0}^{\tau_i} (\mu_{i,r,\theta} N_{i,t-\theta} + v_{i,r,\theta} \xi_{i,t-\theta}) + \pi_{r,t} \quad \forall r \in R, t \in T \quad (1)$$

Equations (2) and (3) limit the excess resources and the continuous extent of a task respectively.

$$R_{r,t}^{min} \leq R_{r,t} \leq R_{r,t}^{max} \quad \forall r \in T, t \in T \quad (2)$$

$$V_{i,r}^{min} N_{i,t} \leq \xi_{i,t} \leq V_{i,r}^{max} N_{i,t} \quad \forall r \in T, r \in R^{eq}, i \in I \quad (3)$$

2.1. Problem Specific Considerations

In order to meet typical requirements the base RTN model needs to be modified. The first modification is related to product-specific batch-tracking. This means that a specific batch or order can be tracked at each stage of production with a unique batch identifier. In the base RTN model, tasks belonging to the same family are aggregated to reduce model size. To meet the batch-tracking requirement, these variables need to be disaggregated. This means that a unique task and resource needs to be associated with each stage of production for a product. This requirement aligns the

RTN formulation with the ISA-95 standard (as each order corresponds to a specific OperationsRequest) and is a scenario often found in industry. An example of this can be found in stainless steel making, where a specific batch needs to be tracked to ensure that sequencing constraints are met throughout the production process. This leads to the assumption of a fixed-batch size; since each batch needs to be specifically tracked it is assumed that the size of the batch is determined elsewhere. Thus, the continuous extent variable $\xi_{i,t}$ and Eq (3) are not considered in this RTN formulation.

Other simplifications that are often made when using RTN models are to aggregate parallel equipment, and to use simplified transfer times between the stages. In this work, a separate resource is defined for each parallel equipment at each stage of production (corresponding to a specific ISA-95 Equipment definition). This enables the consideration of machine-specific transfer tasks. This greatly expands the size of the model as a new $N_{i,t}$ variable is required for each pair of equipment between two subsequent stages as well as resources to distinguish between an intermediate product that is ready for transport or that already has been transported.

Another important feature to consider is sequence dependent changeovers. To model these, a new task is needed which ‘consumes’ a piece of equipment in a specific operation mode and ‘produces’ the same piece of equipment but in a different mode. A sample RTN containing all of the aforementioned features is shown in Figure 1.

Figure 1: Example RTN of a two-stage batch process considering all the problem features.

While these specific features add flexibility to the RTN model, and conform to the ISA-95 standard proposed for scheduling, they add considerable complexity to the formulation. This provided the motivation to investigate new solution algorithms for such problems.

3. Proposed Algorithm

The proposed iterative algorithm uses a constructive heuristic to provide an initial feasible solution. Pseudocode for the constructive heuristic can be viewed below in Figure 2. The algorithm is initialized with the set of all jobs that needs to be scheduled. For example, job A needs to be processed on Stages 1,2, and 3, and a job is created for the processing of A at each stage. The next job to be scheduled is selected according to some sequencing rule. In this work an As Soon As Possible (ASAP) rule was used, but the algorithm can be applied to any sequencing heuristic. The last step of the algorithm is to greedily schedule that job to a machine based on its required

resource availability. This is performed until all jobs have been scheduled and a complete feasible solution is obtained.

```

I  $\leftarrow$  set of jobs to be scheduled
while I  $\neq \emptyset$  do
    | Select  $i \in I$  according to some sequencing rule (e.g. As Soon As Possible)
    | Greedily schedule  $i$  as soon as possible based on its resource availability
end

```

Figure 2: Pseudocode for the constructive scheduling heuristic

The next step of the algorithm is to match the continuous-time starting times of each of the jobs to the discrete-time grid used by the RTN. This can be done using Eq.(4) where $ST_{m,i}$ is the start time of job i on machine m as given by the heuristic and t_i is the corresponding start time interval in the discrete time RTN formulation.

$$t_i = \left\lceil \frac{ST_{m,i}}{\delta} \right\rceil \quad (4)$$

3.1. Neighbourhood Definition

The heuristic solution is then saved as the best solution which then can then be used to define a subset of the original domain where a better, or even optimal, solution could be located. This neighbourhood is defined based on the start times of adjacent jobs relative to the current task. A job is constrained to start between the start times of the W jobs preceding it (the W^{th} neighbour) on the current machine and the start time of the W^{th} neighbour following it. The neighbourhood of times when a job can start is then given by Equations (5) and (6), where $ST_{m,i}^{early}$ is the earliest time a task can start and $ST_{m,i}^{late}$ is the latest a task can start.

$$ST_{m,i}^{early} = ST_{m,i-W} \quad (5)$$

$$ST_{m,i}^{late} = ST_{m,i+W} \quad (6)$$

If the W^{th} neighbour of a job does not exist, the corresponding bound is not constrained. The earliest and latest possible start times can then be mapped to the discrete time intervals. A similar approach to define the window can be followed if the job start times are already in a form matching the RTN intervals. The binary start variable $N_{i,t}$ can then be constrained to this interval which greatly reduces the number of binary variables (and thus the computation time) of the RTN model while still allowing the jobs the flexibility to switch sequencing positions with their W adjacent neighbours. Note that this applies to the binary variables related to all parallel machines, so a job may be shifted to a different parallel machine. A diagram of the neighbourhood definition can be viewed in Figure 3. In this diagram, job A1 is constrained to start in the neighbourhood defined by the start time of jobs B1 and C2. The binary variables for the processing of job A1 outside of this interval on all machines are set to zero. Once the binary variables of the RTN model have been constrained to their respective neighbourhoods, the optimization problem can be run with the current best solution providing an initial feasible point. The solution from the optimization problem can be saved as the best solution, which then can be used to initialize the next iteration. This process can be performed iteratively until one of three stopping criteria are met. The stopping criteria are: maximum computation time, maximum number of iterations, or a convergence criterion. The convergence criterion checks if the optimal value from the previous iteration and the current iteration are the same. If they are, no further improvement can be made within this given neighbourhood meaning that further iterations would yield no improvement. The downside of the heuristic nature of the algorithm is that it is unable to guarantee global optimality. The complete algorithm flowchart can be seen in Figure 4.

Figure 3: An example neighbourhood for job A1 where $W = 1$.

4. Results and Discussion

The proposed algorithm was tested on a four stage batch process with two parallel machines at each stage. The problem considers six product types with four orders per type. The algorithm was tested against a full-space RTN, however for all cases the full-space model was unable to return a solution within the allotted time of 10,000 seconds. To provide an optimal reference point, an RTN which aggregates the products by type (by considering only one binary per product type which executes multiple times) provides an exact optimal value. This is an exact aggregation, however it violates the specific batch tracking requirements of the problem. The optimization problems were solved using GAMS 24.7.4/CPLEX 12.6.3.0. The algorithm was tested for three objective functions: makespan (min), electricity cost (€, with hourly prices), and inventory minimization. For the inventory objective, the goal is to minimize the waiting times of products between stages. Note that the objective function only applies to the MILP in the algorithm, as only an ASAP heuristic was considered in this work. All problems were run with neighbourhood sizes of one, two, and three jobs (denoted $W1$, $W2$, and $W3$ respectively). The results are presented in Table 1.

The results indicate that the iterative algorithm is able to find near optimal, or optimal solutions, in orders of magnitude less time than the full-space model would require. For the makespan objective, both the $W2$ and $W3$ configurations were able to find the optimum solution in less than half the time it took to solve the aggregate problem. For the electricity cost objective, none of the scenarios were able to find the optimum, however, they all got to within 0.5% of it. For this case the $W2$ and $W3$ scenarios both took longer to solve than the aggregated model, due to the fact that they had more binary variables per iteration. For the inventory objective the algorithm was able to find the optimal value of zero for all cases. It is interesting to observe that for all objectives the $W2$ scenario took longer than the $W3$ scenario as it was difficult for this configuration to close the optimality gap. For all objectives the $W1$ case took less than four minutes but was able to get to within 2.2% of the true optimum.

5. Conclusion

In this work a novel iterative neighbourhood search-based algorithm was proposed for the solution of RTN scheduling models. The algorithm reduces the search space of the original RTN problem by limiting the task start times to a neighbourhood in a way that greatly reduces the number of

Figure 4: Algorithmic flowchart of the proposed iterative neighbourhood search algorithm

binary variables while still allowing the schedule to move towards better solutions. This procedure is repeated iteratively until the stopping criteria are met. Results show that the novel approach is able to tackle problems that are intractable for the full-space RTN in industrially relevant time frames while sacrificing optimality. Future work includes combining this approach with a rolling horizon scheme in order to deal with very large problems.

Table 1: Results from the test cases. Objective MS = Makespan, Inv = Inventory holding cost, EI = Electricity cost

Algorithm	Objective	Best Solution	Improvement from Heuristic (%)	Gap From Optimal (%)	Number of Iterations	Computation Time (s)
ASAP	MS	1620	0	18.7	N/A	<5
Agg. RTN	MS	1365	18.7	0	N/A	7,871
Iter. W1	MS	1395	13.9	2.2	4	215
Iter. W2	MS	1365	18.7	0	4	2,796
Iter. W3	MS	1365	18.7	0	2	1,166
ASAP	EI	105,114	0	4.6	N/A	<5
Agg. RTN	EI	100,293	4.6	0	N/A	1,243
Iter. W1	EI	100,658	4.2	0.4	7	121
Iter. W2	EI	100,646	4.3	0.4	4	3091
Iter. W3	EI	100,393	4.5	0.1	4	2003
ASAP	Inv	206	0	100	N/A	<5
Agg. RTN	Inv	0	100	0	N/A	439
Iter. W1	Inv	0	100	0	4	200
Iter. W2	Inv	0	100	0	3	222
Iter. W3	Inv	0	100	0	2	131

6. Acknowledgements

Financial support is gratefully acknowledged from the Marie Skłodowska Curie Horizon 2020 EID-ITN project “PRONTO” (Grant agreement No 675215). The authors also greatly appreciate many scientific discussions with Pedro Castro.

References

- P. M. Castro, I. Harjunkski, I. E. Grossmann, 2011. Greedy algorithm for scheduling batch plants with sequence-dependent changeovers. In: *AIChE Journal* 57, 373–387.
- P. M. Castro, L. Sun, I. Harjunkski, 2013. Resource-task network formulations for industrial demand side management of a steel plant. *Industrial and Engineering Chemistry Research* 52, 13046–13058.
- H. Hadera, P. Wide, I. Harjunkski, J. Mäntysaari, J. Ekström, G. Sand, S. Engell, 2015. A Mean Value Cross Decomposition Strategy for Demand-side Management of a Pulping Process. In: *European Symposium on Computer Aided Process Engineering - 12th International Symposium on Process Systems Engineering*, Copenhagen Denmark, 31 May - 4 June.
- I. Harjunkski, R. Bauer, 2014. Sharing data for production scheduling using the ISA-95 standard. *Frontiers in Energy Research* 2, 44.
- I. Harjunkski, C. T. Maravelias, P. Bongers, P. M. Caster, S. Engell, I. E. Grossmann, J. Hooker, C. Mendez, G. Sand, J. Wassick, 2014. Scope for industrial application of production scheduling models and solution methods. *Computers and Chemical Engineering* 62, 161–193.
- C. A. Méndez, J. Cerdá, 2002. An MILP Framework for Short-Term Scheduling of Single-Stage Batch Plants with Limited Discrete Resources. In: *European Symposium on Computer Aided Process Engineering* 12, 721–726.
- S. Panek, S. Engell, C. Lessner, 2005. Scheduling of pipeless multi-product batch plant using mixed-integer programming combined with heuristics. In: *European Symposium on Computer-Aided Process Engineering* 15, 29 May - 1 June 2005, Barcelona Spain, 1570–1575.
- J. Roslöf, I. Harjunkski, J. Björkqvist, S. Karlsson, T. Westerlund, 2001. An MILP-based reordering algorithm for complex industrial scheduling and rescheduling. *Computers and Chemical Engineering* 25, 821–828.
- J. M. Wassick, J. Ferrio, 2011. Extending the resource task network for industrial applications. *Computers and Chemical Engineering* 35, 2124–2140.