CHEMICAL ENGINEERING TRANSACTIONS
VOL. 70, 2018
A publication of AIDIC
The Italian Association of Chemical Engineering
Online at www.aidic.it/cet
Guest Editors: Timothy G. Walmsley, Petar S. Varbanov, Rongxin Su, Jiří J. Klemeš

ISBN 978-88-95608-67-9; **ISSN** 2283-9216
DOI: 10.3303/CET1870150

Heuristic Algorithm Utilizing Mixed-Integer Linear Programming to Schedule Mobile Workforce

András Éles[a,*], Heriberto Cabezas[b], István Heckl[a]

[a]Department of Computer Science and Systems Technology, University of Pannonia, Hungary
[b]Center for Process Systems Engineering and Sustainability, Pázmány Péter Catholic University, Hungary
eles@dcs.uni-pannon.hu

A Mixed-Integer Linear Programming (MILP) based heuristic algorithm is proposed to obtain economical schedules for mobile workforce, taking into account travelling, execution and resource utilization costs and other requirements. The problem of mobile workforce management arises when several working teams must be assigned to tasks spread over a geographic area. This means we do not only have to take scheduling and assignment into consideration, but also significant travelling times and costs. The goal is the minimization of total cost while executing the most tasks possible, which involves keeping travelled distances, and hence environmental impacts low. A novel MILP model is proposed, which is capable of handling several factors that may arise in mobile workforce management problems. These factors include time windows for tasks, resource utilization, packing and unpacking times, and precedence, mutual exclusive or parallel execution requirements between tasks. Our solution method involves a MILP model, as well as a heuristic greedy algorithm, which assigns tasks one by one to teams' schedules. This two level approach ensures practically acceptable computational time. Our method is tested over medium-scale case studies.

1. Introduction

The problem of mobile workforce management arises when activities to be performed are spread over an area. This can range from logistics inside a production facility to field workforce over a large geographic area. Nevertheless, assignment of workforce to jobs does not only require scheduling, but considerable amount of travelling or reallocation time and costs between two jobs must be taken into account. In practice, companies usually have a given set of tasks to be performed, but the exact order in which these tasks are performed greatly influences resource usage, and hence overall efficiency. This leads to a complex optimization problem for the company, which shows characteristics of both scheduling problems and Vehicle Routing Problems (VRP).
Scheduling had been extensively investigated in terms of batch processes, often with Mixed-Integer Linear Programming techniques, a review of which is from Mendez et al. (2006). The goal is usually the optimization of makespan, operating costs or throughput. If a mathematical programming model is used, it can either be a slot-based one (Pinto and Grossmann, 1995), may utilize variable-length time intervals (Maravelias and Grossmann, 2003), or exhibit the precedence relations between jobs to be executed (Mendez and Cerda, 2003). Other, hybrid or even combinatorial methods, like the S-graph framework, can also be applied to scheduling (Hegyhati et al., 2011).
Solutions for the Vehicle Routing Problem may vary greatly based on the exact conditions and goals. Camm et al. (2017) minimize weighted distances of a bus shuttle service, which leads to a linearized quadratic program, solved by a specific algorithm. Timing requirements are one of the most common constraints. Abdelaziz et al. (2017) develop efficient routing of airport buses between meeting points and the airport, minimizing costs of the company due to travelling, by stochastic programming. Time windows are crucial when products are to be delivered cannot be stored for long. Lee et al. (2014) developed a neighborhood search algorithm to solve delivery and production of radioactive medicine simultaneously, where transportation is not only a term of expenses but is a strict constraint. Paz et al. (2014) proposed mathematical programming models to address the case where multiple depots can also be taken into consideration, and give an energy-efficient solutions to

Please cite this article as: Eles A., Cabezas H., Heckl I., 2018, Heuristic algorithm utilizing mixed-integer linear programming to schedule mobile workforce , Chemical Engineering Transactions, 70, 895-900 DOI:10.3303/CET1870150

utilization of electric vehicles. Chen et al. (2009) use a nonlinear program and specific algorithm to optimize production of perishable food and its delivery simultaneously.
Specific optimization tools are usually developed for cases when the problem was to schedule or assign mobile workforce optimally. Safaei et al. (2008) solve the maintenance workforce scheduling problem by minimizing workforce cost and maximizing equipment availability utilizing simulated annealing. To reduce business costs, working area of field engineers was optimized by Starkey et al. (2016) using a multi-objective genetic algorithm based on fuzzy logic. In general, these methods are highly heuristic due to the complexity of the underlying combinatorial optimization problem. Solution methods for complex problems often address the system design and scheduling part at the same time, for example, Li and Majozi (2017) give a solution for freshwater supply optimization in a processing plant. Gregor et al. (2017) emphasize and investigates that transportation costs should be an integral part of design, as precise models often deviate from linear simplifications.
In this work, a novel Mixed-Integer Linear Programming model was developed which addresses management of mobile workforce, with a wide range of real world parameters considered. Instead of solving it to optimality, a heuristic algorithm was implemented, which selects tasks to be solved one by one, based on the objective function of the MILP model run on the actual state of the schedule.

2. Overview of problem specification

In the mobile workforce management problem of interest, a company has to assign its workforce to tasks which should be executed. The workforce is divided into individual working teams who move and execute tasks together all the time. Teams may have different capabilities in terms of task execution, move speed, resource capacities, costs, and other properties. Tasks that must be solved are spread over a geographic area. A single team must be present on the task site and spend a specific amount of time there to execute a task.
Besides the time needed, tasks may require the presence or consumption of specific resources. Moreover, deadlines and earliest starting times have to be considered. These properties of tasks can also be different based on which team is selected to solve it. Also, tasks may be dependent on other tasks as well. For example, multiple tasks might be needed to be performed in a given order. In this way, a more complex activity can be modeled, by separating it into several distinct steps. These steps can possibly be divided between multiple teams.
The company possesses several depots with working teams stationed there. The teams start from these depots at the beginning of the workday and must arrive back to the same place until the end of the day. A single team travels and then executes tasks at their corresponding sites, possibly several ones in a row at one place. Both travelling and execution have time and cost requirements.
The goal is to determine a schedule for the workday, for each working team, to solve as many tasks as possible, with minimal costs. The company possibly has more tasks that can be solved in a single day. To model an optimal selection, a fixed penalty cost is incurred for each task which is excluded from the current schedule. The objective of the MILP model is therefore the minimization of costs, which comes from several sources, including task execution, travelling, resource utilization, penalties for violating task deadlines or omitting tasks, waiting costs, and others. All the data is available a priori, and the problem is fully deterministic.
Note that, although we refer to travelling in this problem, it may represent any requirement between actual executions of tasks. The specification was motivated by field workforce optimization of an electricity service provider company, and is naturally applicable to others operating in a geographic area. However, this specification can be used to describe scenarios in a single production plant where workforce needs reallocation between tasks. Equipment changeovers between batch processes can also be modeled by the notion of travelling, as cost and time requirements can be arbitrary.

3. The proposed Mixed-Integer Linear Programming model

To address the mobile workforce management problem, a Mixed-Integer Linear Programming (MILP) model was formulated. The focus in design was on the support of a wide range of features, the ease of future extensions and the technical possibility to incorporate partial decisions into the model. The intention was to include the model in an algorithmic framework, rather than to achieve globally optimal solution with one single MILP solver running in minimal computational time for even large-scale problems.

3.1 Main idea of the mathematical model

To achieve versatility and efficiency, the MILP was designed as a slot-based scheduling model. Each team has a set of slots, which are either travelling slots or working slots, strictly alternating, starting and ending with a travelling slot (see Figure 1). Travelling slots contain all the movement of the team, while working slots are assigned to at most one task each. The one-to-one correspondence of tasks to working slots is modeled with

binary decision variables $b_{k,m,i}$ defined for each pair of a task k and a working slot (m, i) of team m, the so-called allocation variables. Note that slots of different teams are neither synchronized nor equidistant.

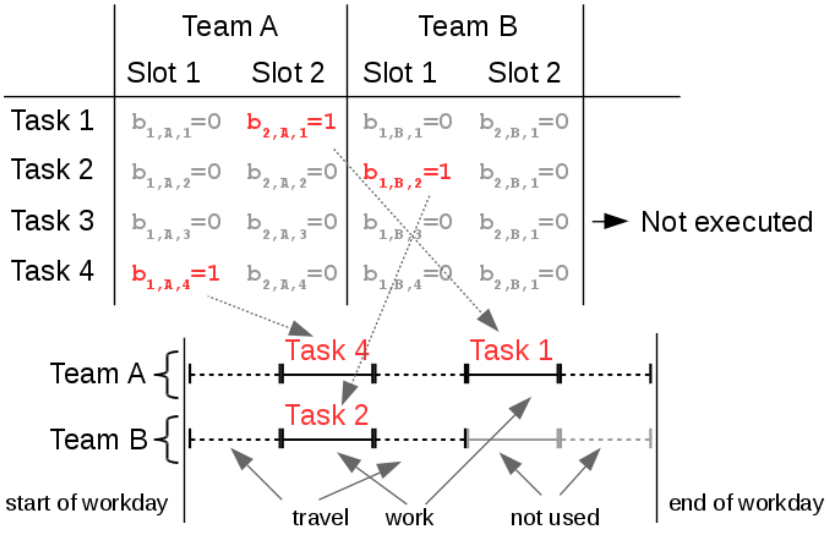

Figure 1: The binary decision variables of the MILP model determine the assignments of tasks to teams and the order for each team in which they are actually performed

3.2 Supported features in the model

The resultant mathematical model can be large due to the aforementioned selection of binary variables, but it gives the possibility to consider many details of the real world problem.

- Absolute and recommended time windows: Tasks are subject to earliest starting and latest finishing times that cannot be violated if the task is executed. Also, there is a narrower recommended time window, for which earlier start or later end may incur a penalty.
- Resource utilization: Teams may carry several resources, which must be present (for example tools), or be consumed at each task. This incurs costs, and possibly prevents teams at specific moments from doing tasks requiring too many resources.
- Packing and unpacking times: If a team arrives at a site before executing a series of tasks there, it must spend some time before working, which also incurs costs. The same applies once the team finishes and leaves the site.
- Task relations: Precedence of tasks against each other and other pair-wise relations can be included as constraints. For example, execution by the same team, security of a site after leaving a long activity unfinished, mutual exclusion and parallel execution. All can be defined for pairs of tasks.
- Teams have different maximum travel times, working times, required idle times, resource usages and carrying capacities. Task execution times and costs also depend on the team selected.

4. Algorithmic framework

Solving the Mixed-Integer Linear Programming model to global optimality for large instances can result in tedious computational effort. For this reason, an algorithmic framework is proposed which operates above the MILP model. However, the strong modeling power of the MILP can be exploited as well, if a solution algorithm uses the MILP model to make decisions in terms of exploring the search space, or the resulting solution itself.

4.1 Utilization of the MILP model

The search space of the original MILP model is selectively explored by the proposed algorithmic framework. The full search space is usually too large for any MILP solver to cope with. However, if a subset of binary decision variables is fixed beforehand, or further constraints and variables are introduced, the problem may become a much smaller subset of the original. Generally, the algorithm makes decisions about the scheduling, and also heuristically shrinks the search space. These are transformed into a greatly reduced MILP model that can be solved fast, while its results imply further decisions. Many small MILP runs are done by the algorithm until the final solution is determined completely.

4.2 Description of the heuristic algorithm

The main idea is to assign one task at a time to the appropriate team. In the beginning all tasks are ordered in some way, and initially not assigned to teams. At each point, teams already have some tasks assigned to them, in a given order. Then, a next task or set of tasks, called active tasks, is introduced, from which one task is selected and assigned into somewhere in the existing schedule: to the end of a sequence of already assigned tasks (Figure 2a), or anywhere between (Figure 2b). Allowance of the second case is a binary parameter of the algorithm. The selection of the task and team is decided by utilizing the MILP model, to allocate only one task with respect to the existing schedule, by optimizing costs. This means that the algorithm is greedy and its objective function is from the MILP model at each step.

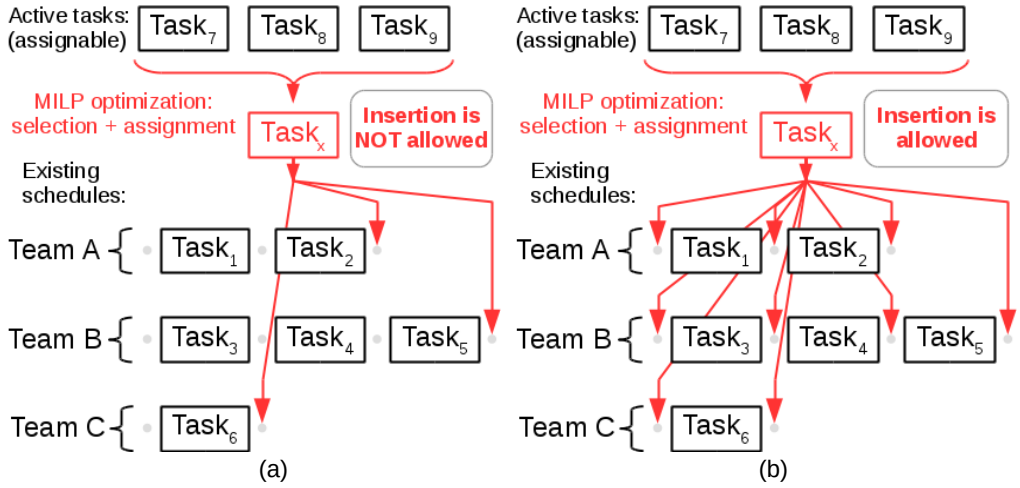

Figure 2: Options in one step of the algorithm when insertion is prohibited (a) and allowed (b).

4.3 Options in the algorithm

There are several factors in the algorithm that may have a great impact on the final solution and require a decision beforehand, three of which are investigated. Any variation of these can be used together.

- Ordering of the tasks: Initially, an ordering of tasks must be determined, in which the active tasks will be selected in the course of the greedy algorithm. The sequence must respect task data, for example precedence relations. Sorting by deadlines is one sensible order. Besides, the order can be arbitrary.
- Set of active tasks: Among the tasks not already scheduled, a subset is selected which is part of the optimization procedure of the current step of the greedy algorithm. We can select a fixed amount of the next tasks in the order as active, or even all remaining tasks.
- Insertion: We can decide if insertion of tasks before already assigned ones is possible in the schedule, or only appending them to the end of task sequences is possible.

4.4 Constraints ensuring insertion of tasks

To relax the model to allow insertion of a new task into the already constructed schedule, and yet to remain as strict as possible, the following constraints were introduced. Here K_{new} is the set of active tasks from which we can assign the next one, K is the set of all tasks, J is the set of working slots, and N_m is the index of the last working slot of team m, the only one to which a task is not assigned. For slots with smaller indices, there are triplets $(k, m, i) \in P$ which denote scheduling decisions made so far.

$$\sum_{k \in K_{new}, (m,i) \in J} b_{k,m,i} \leq 1 \tag{1}$$

Constraint (1) expresses that only one new task can be assigned at the whole step of the greedy algorithm. Note that if insertion is allowed, it can also go to any slot already occupied by tasks treated before.

$$b_{k,m,i} = 1 - \sum_{k' \in K_{new}, (m,i') \in J:\, i' \leq i} b_{k,m,i} \quad \forall \ (k, m, i) \in P \tag{2}$$

$$b_{k,m,i+1} = \sum_{k' \in K_{new}, (m,i') \in J:\, i' \le i} b_{k,m,i} \quad \forall\ (k,m,i) \in P:\, i < N_m \tag{3}$$

Constraints (2) and (3) ensures for all previously made decisions $(k, m, i) \in P$ (where task k had been previously scheduled to slot i of team m) that the task is actually executed in slot $(m, i+1)$ or (m, i) based on whether the new task k' is included earlier in the schedule of the same team or not, respectively.

5. Case studies

The MILP model was implemented in GNU MathProg programming language, and the corresponding GLPSOL v.4.57 solver was used for testing. The heuristic algorithmic framework was implemented in C++ and Perl, and run on Ubuntu Linux 16.04, on a workstation with an Intel i7-4770 CPU, and 16 GB RAM. The data files of several case studies were implemented and tested with different variations of the algorithm's options. The focus was on discovering the effect of the different options, to be able to develop more advanced heuristic algorithms in the future. Two case studies are shown below, constructed in order to test the features of the model.

5.1 The first problem

This case study consists of 6 orders that have 17 tasks in total with precedence relations between each other. All of these tasks can be scheduled by the teams in one day. Teams start from a single depot and return at the end of the workday. In Table 1, we can observe results based on different options. The selection of the active set of each step was determined by 1, 2 or 3 next tasks from the randomly predetermined order of tasks by deadlines' increasing order, or all of the remaining tasks. Insertion before already scheduled tasks was enabled and disabled for the otherwise same test cases.
It can be seen that the active set option does not have a large impact on the result. It just adds some randomness on the course of the algorithm. The insertion however usually slightly improves the objective with the more relaxed single step, but also increases computational times as a tradeoff.

Table 1: Computational results for the first case study.

Case	Active set	Insertion allowed?	Objective (HUF)	Computational time (s)
#1	1 next	no	637,867	4.963
#2	1 next	yes	635,420	4.965
#3	2 next	no	641,110	5.348
#4	2 next	yes	635,840	6.300
#5	3 next	no	639,077	5.492
#6	3 next	yes	636,560	6.546
#7	all remaining	no	639,077	5.549
#8	all remaining	yes	635,420	9.642

5.2 The second problem

The larger problem had the same sites and team data, but included 30 orders, with 90 tasks in total. It was impossible for the 6 teams to solve all tasks in time, so missing tasks were penalized with high coefficients, which turn out to be the only dominant part of the objective function. Tasks were ordered randomly. The results in Table 2 show us that insertion also allows us to include more tasks in the final schedule. However, this is not always the case. In case #7 and #8, the allowance of insertion reduced computational time and made a much worse final result. This is due to the fact that the algorithm is heuristic in nature, and more freedom in the beginning can spoil the whole further schedule. This phenomenon is subject to future investigation.

Table 2: Computational results for the second case study.

Case	Active set	Insertion allowed?	Objective (HUF)	Computational time (s)
#1	1 next	no	545	1,252.983
#2	1 next	yes	261	2,182.392
#3	2 next	no	984	851.985
#4	2 next	yes	341	2,331.488
#5	3 next	no	1,244	734.447
#6	3 next	yes	211	1,707.598
#7	all remaining	no	998	953.553
#8	all remaining	yes	2,736	779.225

6. Conclusions

A novel slot-based Mixed-Integer Linear Programming model was developed to give optimal schedules and allocations management of mobile workforce during a single workday. The model supports a wide range of parameters, including absolute and recommended time windows, team-specific execution, waiting, packing and unpacking, idle and travelling times, pair-wise relations between tasks like precedence, resource utilization and consumption, and more. The result of the model is the assignment of tasks to teams, their ordering and exact scheduling as well as determination of travelling times, with minimal cost. An algorithmic framework was implemented which uses the MILP model as a greedy step to select one next task and add it to the existing sequences of tasks. We can tune this algorithm by different methods for the initial ordering of tasks, the selection of the active tasks at each step, and the allowance of insertion of tasks inside the sequence of a team at each point.

The framework was tested on medium-scale problems. The smaller one with 17 tasks and 6 working teams involves solutions with all tasks executed and roughly similar solutions are found within a few seconds. If insertion of tasks is allowed, the computational time usually increases but the result also improves. For the larger problem with 90 tasks and 6 teams, when many tasks cannot be solved, computational times capped at roughly 40 minutes, but solutions were still obtained. Development of more advanced and efficient heuristic algorithms is subject to future research.

Acknowledgments

We acknowledge the financial support of Széchenyi 2020 under the EFOP-3.6.1-16-2016-00015.

References

Abdelaziz F.B., Masri H., Alaya H., 2017, A recourse goal programming approach for airport bus routing problem, Annals of Operations Research, 251, 383–396.

Camm J.D., Magazine M.J., Kuppusamy S., Martin K., 2017, The demand weighted vehicle routing problem, European Journal of Operational Research, 262, 151–162.

Chen H.K., Hsueh C.F., Chang M.S., 2009, Production scheduling and vehicle routing with time windows for perishable food products, Computers & Operations Research, 36, 2311–2319.

Gregor J., Somplak R., Pavlas M., 2017, Transportation Cost as an Integral Part of Supply Chain Optimisation in the Field of Waste Management, Chemical Engineering Transactions, 56, 1927–1932.

Hegyhati M., Holczinger T., Szoldatics A., Friedler F., 2011, Combinatorial Approach to Address Batch Scheduling Problems with Limited Storage Time, Chemical Engineering Transactions, 25, 495–500.

Lee J, Kim B.I., Johnson A.L., Lee K., 2014, The nuclear medicine production and delivery problem, European Journal of Operational Research, 236, 461–472.

Li Z., Majozi T., 2017, Optimal Design of Batch Water Network with a Flexible Scheduling Framework, Chemical Engineering Transactions, 61, 139–144.

Maravelias, C.T., Grossmann, I.E., 2003, New General Continuous-Time State-Task Network Formulation for Short-Term Scheduling of Multipurpose Batch Plants, Industrial & Engineering Chemistry Research, 42, 3056–3074.

Mendez C.A., Cerda J., 2003, An MILP Continuous-Time Framework for Short-Term Scheduling of Multipurpose Batch Processes Under Different Operation Strategies, Optimization and Engineering 4, 7–22.

Mendez C.A., Cerda J., Grossmann I.E., Harjunkoski I., Fahl M., 2006, State-of-the-art review of optimization methods for short-term scheduling of batch processes, Computers & Chemical Engineering, 30, 913–946.

Paz J., Granada-Echeverri M., Escobar J.W., 2018, The multi-depot electric vehicle location routing problem with time windows, International Journal of Industrial Engineering Computations, 9, 123–136.

Pinto J.M., Grossmann I.E., 1995, A Continuous Time Mixed Integer Linear Programming Model for Short Term Scheduling of Multistage Batch Plants, Industrial & Engineering Chemistry Research, 34, 3037–3051.

Safaei N., Banjevic D., Jardine A.K.S., 2008, Multi-Objective Simulated Annealing for a Maintenance Workforce Scheduling Problem: A Case Study, Chapter In: Cher Ming Tan (Ed), Simulated Annealing, InTech, 27–48.

Starkey A., Hagras H., Shakya S., Owusu G., 2016, A multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization, Information Sciences, 329, 390–411.