Hierarchical scheduling problem in the field of manufacturing operational planning

O E Semenkina¹, ², E A Popov and I S Ryzhikov³

¹Research Department, Reshetnev Siberian State University of Science and Technology, Krasnoyarsk, Russia
²System Analysis and Control Department, Reshetnev Siberian State University of Science and Technology, Krasnoyarsk, Russia
³Numerical modelling office, JSC "Krastsvetmet", Krasnoyarsk, Russia

E-mail: semenkinaolga@gmail.com

Abstract. The paper considers a job shop scheduling problem similar to those taking place in many fields such as project management, educational sphere and operational planning of manufacturing process. The considered problem, in real life, has high dimensionality and it is quite hard to find even a feasible solution, therefore, making necessary a problem-oriented heuristic for solving it in reasonable time. Manufacturing process stability requires special care about restriction and, at the same time, operational planning requires finding solutions quickly. In this paper, hierarchical problem structure is proposed where the top-level problem is the traveling salesman problem and the nested resource-constrained project scheduling problem is replaced by the simulation model. This paper considers combinatorial genetic algorithm (GA) and Lin-Kernigan heuristic (LKH). The performance comparison is fulfilled and competitive results are demonstrated.

1. Introduction

Modern standards and competition make organizations face a number of requirements on quality, speed and cost that are increasingly difficult to satisfy without innovations and efficiency improvements. Therefore, decision support systems are an important part of the modern world along with digitization. One of the important systems of this kind is a system capable of constructing an optimal schedule of works, orders or operations, allowing to increase productivity, to reduce production cost, and to control plan deviations in real-time.

If an organization already have high-quality equipment, takes into account the principles of lean manufacturing and uses information technologies for process status data collection, then the following question arises: “what is the next step?”. Operational production planning (OPP) (and, associated with it, automation) is an important part of a modern production process as they are able to solve the following tasks:

- Deadlines fulfillment in accordance with the entered contracts;
- Production rhythm abidance;
- Reduction of unfinished production processes;
- Preparation of shift daily tasks;
- Productivity increase;
• Rational uniform loading of equipment;
• Deviation detection and their elimination.

Even though such questions are usually solved by experts, they can lead to inconsistency in the decision-making process, errors caused by a human factor or inability to consider all the parameters and how decisions influence each other.

Among the currently available products the following can be highlighted:
• Enterprise Resource Planning Systems (ERP) [1] that solve common financial, personnel, logistics, and other financial management tasks.
• Manufacturing Execution Systems (MES) [2, 3] designed to solve operational problems, track and document the transformation of raw materials into finished goods.
• Customer Relationship Management Systems (CRM) that allow to save information about customers (contractors), the history of relationships with them, analyze interaction results, generate demand statistics for certain contractor groups and thereby increase sales and improve customer service [4].
• Advanced Planning & Scheduling Systems (APS) [5] with the ability to quickly create plans based on available resources and production constraints, as well as rapid re-planning using predefined scenarios.
• Laboratory Information Management Systems (LIMS), which are a necessary tool for chemical enterprises, as well as any enterprises where samples are taken, analysis of raw materials, materials and products are carried out [6]. These systems are also used in enterprise research centers where new production process technologies are being developed or existing ones are being improved.

The algorithms for scheduling in such systems are quite simple and basically use only iterative processes of obtaining a feasible solution by problem-oriented heuristics. Besides, they all operate with general concepts that do not take into account the specifics of a particular production and do not include optimization modules for scheduling considering equipment and workers’ availability. This is due to the fact that most of the methods developed for operational planning are based on simplified models, which reduces their practical significance, or these methods are applied only for certain specific conditions. In addition, all these systems do not include modules for schedule optimization, therefore, today, the main development directions for such systems are their integration with each other and transition to the use of intelligent software.

2. Problem statement
The scheduling theory's starting point is the Henry Gantt's work (Gantt H.L.) [7] released in 1903, in which diagrams now known as Gantt charts were proposed for controlling the loading of machines and workers. This type of diagrams quickly became popular in project management and remains so for more than a century, indirectly involving people in the use of specialized scheduling methods [8], since Gantt charts provide an opportunity to visually assess the quality of the found solution, thereby providing a platform for implementing interactive planning approaches. Lazarev and Gafarov in the book “Theory of Schedules: Tasks and Algorithms” [9] give the following definition: “Scheduling theory is a section of operations research in which mathematical scheduling models (i.e. sequencing in time) of various targeted actions are built and analyzed, taking into account the objective function and various constraints.”

Well-known problems like bin packing problem, knapsack problem, traveling salesman problem, as well as scheduling can be formulated and solved as resource-constrained project scheduling problem (RCPSP) [10]. This problem considers resources (machine tools, employees, etc.) with limited availability and activities (or works) that need to be done where duration of activities are supposed to be known. Another important concept is a workflow that means a sequence of activities that corresponds to a particular type of lots (parties, projects, requests). The problem is to find a schedule of minimal duration by assigning a start time, certain machine and certain employee to each activity provided that the precedence relations and resource availabilities are respected.
First of all, we need to determine RCPSP more formally. Let \( n \) be the number of activities and \( m \) the number of resources. Current project which we are scheduling consists of \( n + 2 \) activities defined by the set \( \{a_0, \ldots, a_{n+1}\} \), where activities 0 and \( n + 1 \) are fictitious activities representing by convention start and end points of the project. The set of actual activities \( A = \{a_1, \ldots, a_n\} \) need to be scheduled using renewable resource set \( R = \{r_1, \ldots, r_m\} \) where each resource \( k \) is available in quantity \( B_k \).

Let \( p = (p_0, \ldots, p_{n+1}) \in \mathbb{N}^{n+2} \) be a vector of processing times, where the \( i \)-th component, \( p_i \), is the processing time of activity \( i \). An additional condition is that there are special values \( p_0 = p_{n+1} = 0 \) in vector \( p \) because of fictitious activities. Each activity \( i \) demands \( b_k \) amount of resource \( k \) during its processing for \( k = 1, \ldots, m \). Let \( S_i \) be a start point of activity \( a_i \) for \( i = 0, \ldots, n + 1 \) where \( S_0 = 0 \) and \( S_{n+1} \) is a makespan because the project end point equals time difference between the start and finish. The precedence constraints are a set \( E \) consisting of index pairs \((i, j)\) that means activity \( i \) must be completed before activity \( j \) starts:

\[
S_i + p_i \leq S_j \quad \forall (i, j) \in E
\]  

(1)

Resource constraints require that at any time \( t \) sum of activities \( A_{in\ process} \) that uses each resource does not exceed resource availability where \( A_{in\ process} = \{i \in A \mid S_i \leq t < S_i + p_i\} \). This constraint can be determined as follows:

\[
\sum_{i \in A_{in\ process}} b_{ik} \leq B_k \quad \forall k \in \{1, \ldots, m\}, \forall t \in [0, T].
\]

(2)

where \( T \) is the scheduling horizon. \( T \) can be considered as upper bound for makespan, but usually this value is taken quite large.

Let \( S = \{S_0, S_1, \ldots, S_n, S_{n+1}\} \) be any solution of RCPSP that mean it is a schedule of activities. \( S \) is an feasible solution if it satisfies the constraints (1) and (2). So we can to determine the RCPSP as an extremum seeking problem with minimizing the criterion \( S_{n+1} \to \min \). According to the computational complexity theory, the RCPSP is NP-hard [11].

Real-world problems usually are more complicated, so you need to consider not only one type of resources like machine tools, but also employees’ availability. There can be requirement to use more than just one machine at the same activity or maybe employees have breaks. If we consider the manufacturing process, then we are talking about several projects and each of them has its own sequence of activities. Let \( L = \{l_1, l_2, \ldots, l_H\} \) is a set of all lots (projects) where each lot \( h \) needs to be processed by sequence of activities which is defined by the set \( E_h \) corresponding to \( A^h = \{a^h_1, \ldots, a^h_n\} \). In this case objective function can be presented as follows:

\[
\sum_{l=1}^H S^h_{n+1} \to \min
\]

(3)

At the same time constraints (1) and (2) will be transformed to

\[
S^h_i + p_i \leq S^h_j \quad \forall (i, j) \in E_h \quad \forall h \in \{1, \ldots, H\}
\]

(4)

\[
\sum_{i \in A_{in\ process}} b_{ik} \leq B_k \quad \forall k \in \{1, \ldots, m\}, \forall t \in [0, T],
\]

(5)

where \( A_{in\ process} = \{i \in A^1 \cap \ldots \cap A^H \mid S^h_i \leq t < S^h_i + p_i\} \). Optimization problem (3)-(5) can be considered as hierarchical optimization problem consisting of combinatorial problem of determining the order of lots and nested problem in the form of RCPSP. In this case lot order problem can be formulated as travelling salesman problem [12].

To solve the classical RCPSP problem, exact methods for finding the optimal schedule were developed, but due to computational complexity, they are applicable only to small problems. In this regard, approximate methods are actively explored that allow one to generate effective schedules for large projects in a reasonable time. While being NP-hard [13] and widespread in business analysis, the standard RCPSP is still a problem of current importance. Approaches to solving this problem are constantly being proposed but methods for solving the complicated RCPSP are not fully developed.
For example, to solve the scheduling problem in the classical formulation, genetic algorithms [14] can be used. In this case some operators need to be modified to work with chromosome type in which activities and resource indexes can be encoded as well as start points. However, this modification did not show really good results [15]. Also, some papers consider the possibility of using the ant colony optimization [16] to solve the RCPSP problem, but this requires accurate choice of heuristics for particular problem [17].

In this paper, we propose to consider a job-shop scheduling problem as hierarchical structure with lot ordering as top-level problem and nested RCPSP replaced by simulation model that makes it possible to simplify the practical application of optimization methods in operational production planning. This proposal is based on the fact that in real-life manufacturing nested RCPSP is so complicated that it is quite hard to find even feasible solution in the general formulation of the problem for just one lot. On the other hand, operational planning requires to find solutions quickly while restriction violation is unacceptable because it is extremely important to maintain the manufacturing process stability. In such cases, when the most important is the speed of obtaining feasible solutions, it is reasonable to use a simulation model that ensures the feasibility of the solution and to optimize the external parameters, for example, lot order.

3. Methods

3.1. Lin-Kernighan heuristic

One of the classical methods for solving optimization problems is a local search, in particular case of traveling salesmen problem local search may be in the form of k-opt algorithm (Lin-Kernighan heuristic [18]). Let $c$ be an objective function from a set $F$ to a set $R^1$ where $F$ is a feasible region that means all cyclic permutations of $n$ objects. The point $f \in F$ such that $c(f) \leq c(y)$ for all $y \in F$ is called global optimum. Then neighborhood function is defined as follows (k-replacements):

$$N_k(f) = \{ g : g \in F \text{ and } g \text{ can be obtained from } f \text{ by removing no more than } k \text{ edges from the original tour } f \text{ and adding } k \text{ different edges} \}$$

Point $f \in F$ is called locally optimal with respect to the neighborhood system $N$ if $c(f) \leq c(g)$ for all $g \in N(f)$. Solutions of the traveling salesmen problem that are locally optimal with respect to the system of neighborhoods $N_k$ generated by $k$-replacements are called $k$-optimal. The essence of the algorithm is to consider neighborhood of current solution. If there exists a graph $g$ in this neighborhood with better objective function value, then $g$ becomes current solution. The procedure is repeated as long as the current solution can be improved. Improve function for $f \in F$ can be defined as follows:

$$\text{improve}(t) = \begin{cases} \forall s \in N_k(f) \text{ such that } c(s) < c(t) \text{ if } s \text{ exist } \\
"\text{null}" \text{ otherwise}
\end{cases}$$

The search algorithm for the $k$-optimal tour is:

1. Set some initial tour
2. $t = \text{improve}(t)$
3. If $\text{improve}(t) = \"\text{null}\"$ stop algorithm, else go to step 2

If any local optimum $f \in F$ with respect to neighborhood system $N$ is also a global optimum, then the system $N$ is called exact. The neighborhood system $N_2$ is not exact, and $N_n$, where $n$ is the number of cities, is exact.

Lin in his work [19] showed that 3-optimal solutions exceed 2-optimal solutions, similarly to these 4-optimal solutions exceed 3-optimal solutions, but not so much how computational resources grows with the transition from 3-opt to second 4-opt. That is why this paper deals with 3-opt.

3.2. Genetic algorithm

Genetic algorithm (GA) [14] is a popular search heuristic inspired by principals of evolution, but GA for the TSP has some difference from classical version. First of all, a chromosome of combinatorial GA...
is represented as permutation of the n numbers. Secondly, there are a few changes in standard operators such as mutation and recombination while selection remains the same.

At the stage of recombination in problems on permutations, selected individuals (called parents) devolve part of their chromosomes by using certain rule. Firstly, part of the first parent chromosome is randomly selected and become a part of the offspring. Then the rest of the chromosome is filled with genes in the order in which they appear in the second parent (figure 1).

![Figure 1. Recombination in case of combinatorial GA.](image)

There are four ways to implement mutation in combinatorial GA which are shown on figures 2-5: by inversion, by insertion, by 2-exchange and by shifting.

![Figure 2. Mutation by inversion.](image)

![Figure 4. Mutation by insertion.](image)

![Figure 3. Mutation by 2-exchange.](image)

![Figure 5. Mutation by shifting.](image)

GA has many adjustable parameters because there are different operator types and each of them has its own parameters, we choose some of them to investigate GA:

- 8 different selection variants: tournament selection with the size of the tournament equals 2, 4 or 8, the rank selection with a linear ranking, the rank selection with exponential ranking with parameter λ equal to 0.95, 0.8 or 0.5, and the fitness proportional selection
- 3 different mutation levels: low, medium and high
- 4 different mutation types: inversion, insertion, 2-exchange and shifting.

Thus, we got $8 \cdot 3 \cdot 4 = 96$ different GAs that were used for comparison on 3 tasks. There is a large variation in efficiency of GA on different settings, so it is reasonable to compare not the efficiency of GA with particular settings, but the best and average one. Therefore, results below in paper will show
"best" GA that means GA with best settings (best result on the task), as well as "average" GA that means results of 96 GAs with different settings were averaged.

4. Numerical results
We created a simulation model that contains such entities as machine tools, employees, operations, lots and workflows. Machine tools and employees are renewable resources that are needed to process any lot on an operation. Every machine tool and employee has its own schedule in form of a set of intervals, when not occupied by any operation. Machine tools may belong to one of Mt types of equipment, employees may be competent in one of Et competence types. An operation is a set of activities that need one machine tool of certain type and one employee with certain competence. Each operation has a determined duration and can be assigned to several workflows. Workflow is a technological process that can be described by an operation sequence. For each lot with raw material entered to the model input, we need to assign a workflow, comprising instructions of what to do with this lot. The simulation model takes a list of lots in the order of their processing and consistently puts all necessary operations for each lot in the schedule. In this case, all restrictions on the resources are taken into account, and a start point of an operation is selected as the nearest free point with an available machine tool of certain type and available employee with certain competence types.

The simulation model is a part of the system that was developed with Spring Boot framework based on Java. The system has web interface based on framework JSF with using Spring Beans. For data persistence, it was used the open source database PostgreSQL, and for connection between database and application, it was used the framework MyBatis.

Algorithms performance was compared on solving 3 tasks which were generated in the following way. We took 3 types of machine tools and then for each task was generated random amount (from 1 to 5) of machine tools of each type. In the same way, we took 3 type of employee competence and then for each task was generated random amount (from 1 to 5) of employees with each type of competence. After that, we generated 50 operations with random machine tool type, employee competence type, and duration. Next, we took 4, 8 and 12 workflows for 3 task respectively and each workflow was generated as a random sequence of from 3 to 10 operations. At the end were generated 5 lots for each workflow for all tasks, so we got 3 tasks with dimension equal 20, 40 and 60 respectively.

All algorithms were compared on 1000000 objective function calculations. Results of numerical experiments averaged over 100 runs are presented in tables 1-3 where were used the following solution quality indicators: best of all runs value of the objective function, average value and standard deviation. Due to the abundance of possible solutions, there is no way to know whether a global optimum was found or not. However, comparing the results of different algorithms on the same problem may indicate their effectiveness.

Table 1. Algorithms comparison on the task 1.

| Task 1 (20) | Best run | Average run | Standard deviation |
|------------|----------|-------------|--------------------|
| LKH        | 15543    | 15890.5     | 182.69             |
| GA (best)  | 15436    | 15652.7     | 276.58             |
| GA (average)| 16484  | 17538.2     | 835.26             |

Table 2. Algorithms comparison on the task 2.

| Task 2 (40) | Best run | Average run | Standard deviation |
|------------|----------|-------------|--------------------|
| LKH        | 14512    | 14839.1     | 223.52             |
| GA (best)  | 14152    | 14763.5     | 342.57             |
| GA (average)| 15926  | 16842.2     | 926.92             |

Table 3. Algorithms comparison on the task 3.

| Task 3 (60) | Best run | Average run | Standard deviation |
|------------|----------|-------------|--------------------|
| LKH        |          |             |                    |
| GA (best)  |          |             |                    |
| GA (average)|        |             |                    |
As it can be seen, best GA outperform LKH significantly while average GA results are not so good. Better results of GA are most likely based on the fact that it is a global search algorithm, while HLK is a local search algorithm with a multistart. It should be kept in mind that when one solves a practical problem it is unknown beforehand which settings of the algorithm on the given task will be the best, therefore it is expected that algorithm's efficiency will be comparable to the averaged over settings one.

5. Conclusions

Scheduling problem is as complex as widespread and because of this fact, it is important to investigate new methods for solving it. Operational planning requires to find solutions quickly according to the latest data of manufacture monitoring and at the same time the problem dimension can be great - hundreds of machine tools, operations, employees and lots. In such conditions, most of the search space in classic scheduling will not contain a feasible solution while restriction violation is unacceptable because it is extremely important to maintain the manufacturing process stability. That is why the hierarchical problem structure was proposed where a top-level problem is the traveling salesman problem and the nested resource-constrained project scheduling problem was replaced by the simulation model.

In this paper, the performances of the Lin-Kernighan heuristic and the genetic algorithm solving the proposed modification of scheduling problem were compared. GA shows better results but this algorithm is difficult for permanent use in operational planning because it is highly dependent on the settings so it requires preliminary calculations to identify the best one. These results show that it is necessary to move towards some kinds of adaptive methods that allow tuning algorithm parameters during their work.

Acknowledgments

The reported study was funded by RFBR according to the research project № 18-37-00433.

References

[1] Leon A 2008 Enterprise resource planning New Dehli: McGraw-Hill 2 224
[2] Frolov E B and Zagidullin R R 2008 MES-systems as they are or evolution of production planning systems Machine park 10(55) 31-7
[3] Scholten B 2009 MES guide for executives: why and how to select, implement, and maintain a manufacturing execution system Research Triangle Park: International Society of Automation
[4] Simonova S I 2015 Data mining for CRM tasks International Journal of Open Information Technologies 2 17-22
[5] Djuha V M and Kirillov D O 2013 Potential advantages of APS systems in sales and operational planning Economic System Management : electronic scientific journal 7(55) 19
[6] Idizikovsky V I and Gorschkoeva E A 2012 Is LIMS an automated control system for laboratories or something more? Modern laboratory practice 3(19) 10-24
[7] Gantt H L 1903 A graphical daily balance in manufacture Transactions of the American. Society of Mechanical Engineers 24 322-6
[8] Wilson J M 2003 Gantt charts: a centenary appreciation European Journal of Operational Research 149(2) 430–7
[9] Lazarev A A and Gafarov A A 2011 Schedule theory: tasks and algorithms (Moscow: Moscow State University Press) p 222
[10] Anichkin A S and Semenov V A 2014 Modern models and methods of scheduling theory Proc. ISP RAS 26(3) 5-50
[11] Blazewiez J, Lenstra J and Rinnooy Kan A H G 1983 Scheduling subject to resource constraints:
classification and complexity *Discrete Applied Mathematics* **5**(1) 11–24

[12] Davendra D 2010 *Traveling salesman problem, theory and applications* (In. Tech. Publishing)

[13] Bartusch M, Mohring R H and Radermacher F J 1988 Scheduling project networks with resource constraints and time windows *Annals of Operations Research* **16** 199–240

[14] Eiben A E and Smith J E 2003 *Introduction to evolutionary computing* (Berlin: Springer-Verlag)

[15] Gromov S A and Tarasov V B 2011 Integrated intelligent systems for operational production planning *News SFU: Technical science* **7** 60–7

[16] Dorigo M and Gambardella L M 1997 Ant colony system: a cooperative learning approach to the traveling salesman problem *IEEE transactions on evolutionary computation* **53**–66

[17] Skobcov Y A and Chegnar O V 2010 Optimization of the department of machine-building production based on the ant colony optimization *Vestnik NTU KPI* **31**

[18] Lin S and Kernigan B W1973 An effective heuristic algorithm for the traveling-salesman problem *Operations research* **21**(2) 498–516

[19] Papadimitriou C H and Steiglitz K 1982 *Combinatorial optimization, algorithms and complexity* (Englewood Cliffs, NJ, Prentice-Hall)