Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 63 (2017) 762 – 767

www.elsevier.com/locate/procedia

The 50th CIRP Conference on Manufacturing Systems

Ant Colony Optimization Algorithms to Enable Dynamic Milkrun Logistics

Ulrich Teschemacher[a*], Gunther Reinhart[a]

[a]*Institute for Machine Tools and Industrial Management, Technical University of Munich, Boltzmannstraße 15, 85748 Garching, Germany*

* Corresponding author. Tel.: +49-(0)89-289-15487; fax: +49-(0)89-289-15555. *E-mail address:* ulrich.teschemacher@iwb.tum.de

Abstract

Flexibility in combination with high capacities are the main reasons for milkruns being one of the most popular intralogistics solutions. In most cases they are only used for static routes to always deliver the same material to the same stations. However, in the context of Industry 4.0, milkrun logistic also has become very popular for use cases where different materials have to be delivered to different stations in little time, so routes cannot be planned in advance anymore. As loading and unloading the milkrun requires a significant amount of time, beside the routing problem itself, both driving and loading times have to be taken into account. Especially in scenarios where high flexibility is required those times will vary significantly and thus are a crucial factor for obtaining the optimal solution. Although containing stochastic components, those times can be predicted by considering the optimal point of time for delivery. In consequence, the best tradeoff between short routes and optimal delivery times is in favor of the shortest route. To solve this optimization problem a biology-inspired method – the ant colony optimization algorithm – has been enhanced to obtain the best solution regarding the above-mentioned aspects. A manufacturing scenario was used to prove the ability of the algorithm in real world problems. It also demonstrates the ability to adapt to changes in manufacturing systems very quickly by dynamically modelling and simulating the processes in intralogistics. The paper describes the ant colony optimization algorithm with the necessary extensions to enable it for milkrun logistic problems. Additionally the implemented software environment to apply the algorithm in practice is explained.

Peer-review under responsibility of the scientific committee of The 50th CIRP Conference on Manufacturing Systems

Keywords: Industry 4.0; Milkrun Logistics; Biology-inspired Optimization

1. Introduction

As the demand for short reaction times in production is rising continuously, intralogistics as well has to cope with these new challenges. In order to enable high efficiency while remaining flexible at the same time, milkruns have become one of the most popular intralogistic solutions to deliver material to the producing areas due to their high capacity [1]. In many scenarios static routes are possible as the same material is always delivered to the same station in the same order. However, there are new use cases, especially in the context of Industry 4.0, where the mastering of flexibility and dynamic changes is a major emphasis. This challenge in combination with the higher demand for efficient material flow have led to the use of milkruns even for scenarios where material requests of the stations are unpredictable and time windows for the delivery are very short. In those environments, routes cannot be planned in advance anymore. Furthermore loading and unloading times require a significant amount of time which is not constant but depends on the time of delivery. Those times have to be predicted and need to be used as a fundamental in the optimization of the dynamic routes to obtain solutions that can be used in practice.

Problems like this can be approximated with the mathematical Vehicle Routing Problem (VRP) where a set of demands of customers need to be assigned to different routes in an optimal order [2,3]. The problem description of the VRP comes close to the practical milkrun problem in factories but needs to be enhanced by several special cases of the real environment. As a consequence also the solution methods have to be adjusted.

There are numerous possibilities to obtain the optimal solution for those problems, but as the VRP is too hard in a mathematical sense and the solution needs to be calculated in

Peer-review under responsibility of the scientific committee of The 50th CIRP Conference on Manufacturing Systems
doi:10.1016/j.procir.2017.03.125

very little time, it is not possible to use those exact methods. Instead, heuristics are mostly applied which obtain a good solution in a reasonable time but cannot guarantee the optimal solution. One algorithm that has proven its potentials for the VRP is the ant colony optimization (ACO), which is also used in this paper. Yet there are several approaches for the ACO to solve the VRP but as the VRP has to be enhanced by several fundamental constraints the ACO as well has to be adapted to the new challenges.

This paper first describes the necessary enhancements to the VRP and the consequences for the ACO. Those changes are described in detail step by step in order to empower the Ant Colony Optimization Algorithm to solve the problem of combining the material requests to milkrun routes and dispatching the individual milkruns to their routes.

2. Vehicle Routing Problems

The Vehicle Routing Problem is one of the most studied combinatorial problems. It is based on the traveling salesmen problem (TSP), where a Salesman has to visit a certain number of cities once each and tries to minimize his total distance to travel.

The vehicle routing problem (VRP) also consists of different customers that have to be visited once. Every customer requests a certain quantity q of goods. For loading and unloading a specific service time is consumed at every customer. In special use cases, customers can also request an amount of goods to be taken away. In contrast to the TSP, there are several vehicles that start from the same location and can share the load between each other. In the standard variant of the VRP, each vehicle has a maximum capacity so the problem is also called the capacitated vehicle routing problem (CVRP). In consequence, the customers are assigned to different tours as can be seen in Fig. 1.

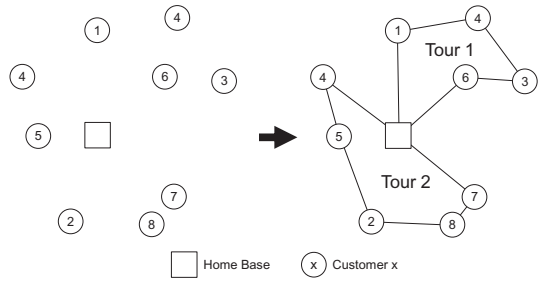

Fig. 1. Customers and Home Base to construct a Vehicle Routing Problem

Numerous variants of the VRP do exist [4]. One very common extension of the VRP is the vehicle routing problem with time windows (VRPTW). For every customer there is an earliest time for the service time to start as well as a latest time. If the vehicle arrives too early it has to wait until the beginning of the time window. If in contrast the vehicle arrives after the end of the time window, the solution will not be valid and thus has to be rejected. Applying such a solution would possibly have a great negative impact on the production processes, if e.g. there are only small buffers applied at the lines. This constraint is a so called hard constraint, i.e. satisfying the time windows is mandatory to observe a valid solution for the given problem. In contrast, a soft constraint would not render a solution invalid if the constraint could not be fulfilled but would only have a negative impact on the rating of the solution. Additionally there are extensions of the problem that consider variable times, e.g. if there are traffic jams between two customers [5]. Those can change dependent on the time of the deliveries and in consequence, a dynamic problem is created.

The complexity of these problems rises very fast (problem is NP-hard [6]) so it is not possible to obtain the optimal solution for bigger problems [7]. Even finding an initial solution, which only needs to be valid but there is no need for optimality, is an NP-hard problem for constrained VRPs [8].

3. Ant Colony Optimization

In this approach the Ant Colony Optimization (ACO) based on the ideas of Dorigo [9] is used for solving the mathematical problem. The basic idea of the ACO is that a large number of agents can build good solutions for hard combinatorial problems based on simple fundamental communication rules.

To do so, the ACO mimics the foraging behavior of ants. The search for food works as follows: As long as ants in nature do not know the location of their food, they wander around randomly until they accidentally discover a food source. They then start their way back while placing so called pheromones – a chemical substance that can be detected by other ants. This gives the ants the possibility to communicate with each other indirectly by modifying their environment. The pheromone trail serves the ants as a distributed memory that is shared between all ants and marks the trail to the food. Other ants on the search for food use this information to quickly find the way to the food source by following the pheromone trail in the direction of the highest pheromone concentration. As pheromones evaporate by time, short ways (which only take short time to walk) will have a higher pheromone concentration than long trails and thus will be preferred when choosing between two pheromone trails (see Fig. 2).

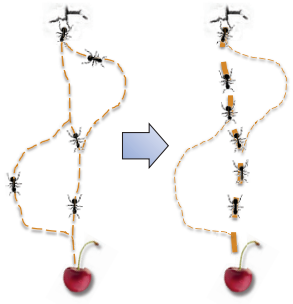

Fig. 2. Foraging and recruiting behaviour of ant colonies [10]

This behavior has been adapted to combinatorial problems in a mathematical sense by Dorigo [11]. In order to provide the necessary information to the algorithm, the problem has to be converted into a graph where a population of artificial ants try to find the shortest way from the start node to the end node. In most situations an artificial ant will have several possibilities for the next node to move on. The ant decides where to move on by a stochastic decision where the probabilities for each potential subsequent node is calculated by the so-called State Transition Rule:

$$p_{ij} = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{j \in allowed\ nodes} \tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}} \quad (1)$$

In this formula τ_{ij} is the pheromone concentration on an edge connecting two nodes i and j. The higher the value of τ, the higher the probability for the edge to be chosen by the ant. The second influencing factor is the heuristic value η_{ij}. By this factor the ant colony optimization algorithm incorporates existing knowledge of the given problem, for example in the case of a distance minimization problem the inverse of the distance to the next node. Thus, in this example, a closer node has a higher probability to be chosen than a faraway node, in general the heuristic value favors nodes that have a higher probability to turn out to be a good choice in the final solution. A proper function to calculate the heuristic value can significantly improve the quality of the solution and lower the time to obtain the solution. The pheromone concentration τ and the heuristic value η are weighted to each other by the two values α and β.

After all ants have completed the construction of the solutions, new pheromones will be placed on the graph. There are different strategies for this step discussed in literature: The behavior that matches the biological phenomenon the best is that every single ant places pheromones in a concentration correlating with the quality of its solution in comparison to the other ants. However, simulations studies revealed, that the so called elitist ant strategy (EAS) [12,13] where only the ant with the best solution of the iteration places pheromones leads to superior results [14]. The pheromone update on the edges of the best solution is then performed by the following two formulas:

$$\tau_{ij} \leftarrow (1-\rho) \cdot \tau_{ij} + \rho \cdot \Delta\tau_{ij} \quad (2)$$

$$\Delta\tau_{ij} = \begin{cases} \dfrac{Q}{f_{evaluation}(best\ so\ far)} \\ 0, otherwise \end{cases} \quad (3)$$

The mathematical representation of the pheromone update consists of two formulas, where the first formula implements two different logical functions. The left summand realizes the evaporation of pheromones already placed on the edge. To do so, the actual pheromone value on the edge is multiplied by the factor $(1-\rho)$ to reduce the pheromone value on this edge. The coefficient ρ represents the evaporation rate of the pheromones which is in the range of 0 and 1. The right part of the formula is responsible for the placement of new pheromones on the edges. The increment value $\Delta\tau_{ij}$ is calculated by formula (3) where Q is a constant and $f_{evaluation}$ is the evaluation value of the best solution so far. As the best known solution will never get worse, the amount of new placed pheromones $\Delta\tau_{ij}$ declines by time and thus leads to smaller pheromone changes in later cycles of the algorithm.

For the VRP to be solved, the problem description has to be transformed into a graph, where the ants can try to find the shortest path by moving along the edges to get the shortest chronological order to serve all customers in the problem. In this graph, the customers with their demands are represented by the nodes, the edges represent the distances between the customers (see Fig. 1).

To setup the algorithm an initial solution has to be generated. This solution does not have special requirements in the quality of the solution but needs to be valid. In a first step, the graph is created and for the given initial solution, pheromones are placed on the graph so that the artificial ants can navigate along these to start their optimization close to the given solution (see Fig. 3).

Then two versions of the ACO algorithm are started in parallel as proposed in [15] (see Fig. 3). The first algorithm tries to find a solution that uses the same number of vehicles as in the best solution so far but needs less time to deliver all demands to the customers. The second optimization algorithm

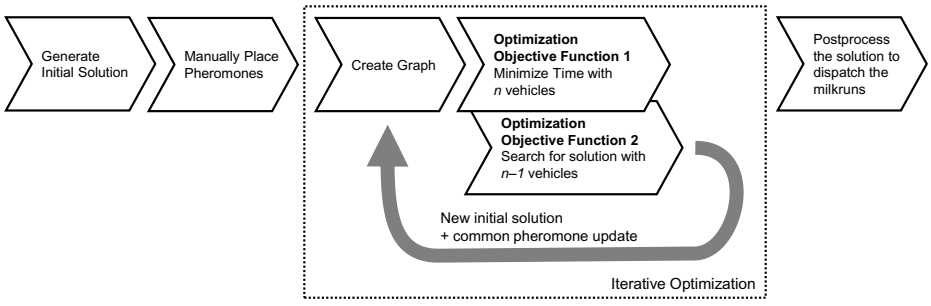

Fig. 3. Collaboration of the different parts of the algorithm

tries to find a solution with one vehicle less than in the currently best solution, no matter how much time will be needed to deliver all orders. If the second optimization algorithm discovers a new valid solution, both optimization algorithms are stopped, the mathematical graph is updated, and pheromones for the new best solution will be placed on the new graph as a start value for both algorithms. Next the two algorithms will be restarted in order to improve the new given solution.

This structure leads to the desired behavior of a dominance in the objective function of having fewer vehicles over the objective to minimize the total time.

4. Handling special requirements for intralogistics

Due to the efficiency of the ACO described above it becomes possible to solve even large Vehicle Routing Problems in short times. To use the algorithm for the challenges in intralogistics, several enhancements have to be introduced though, as the standard theoretical VRP does not cover all arising requirements in practical production environments. In this paper, several requirements are being discussed and a possible solution is given:

- In contrast to the standard VRP where vehicles only operate on one single tour, milkruns in a factory periodically return to their home base after completing their tour in order to start with a new tour. This has to be considered for the creation of the solution.
- For the given problem, the milkrun delivers a new container of material to a customer and takes the old container along. If the container is not empty yet, the driver of the milkrun has to shift the parts from the old to the new container. This process consumes a lot of time that has to be taken into account when optimizing for the best routes. The loading time is a function of the point of time when the milkrun arrives at the station as it is dependent on the filling level in the container. Apart from that shifting material should as be avoided for reasons of ergonomics as well.
- When handling theoretical problems, time windows of a given Vehicle Routing Problems are normally in a way that a valid solution is existing. This is not necessarily the case in practical problems and hence there is the need for a logic to handle those cases as well.

The requirements above are only the most important extensions to the model of the VRP but there are of course a lot more minor constraints to be considered when using the model in practice. The outpointed requirements will be discussed in the following sections.

In almost all variants of the VRP, the total demand is split to several tours that are operated by different vehicles. As long as there are no time windows, there is no difference for the optimization algorithm at all, whether there is only one vehicle that returns to the depot and then starts with the next tour or if there is the same number of vehicles as tours. As soon as time windows for the customers are being taken into consideration, a starting point for every single tour needs to be defined. Existing solutions as well as common benchmarks [16] always use the starting point in time of calculation as the beginning of each route, which is equivalent to using one vehicle for every single tour. For the use in milkrun scheduling, a combination of the two behaviors needs to be used – there are multiple milkruns that return to the depot for the next tour, but due to time windows it is not possible for a single vehicle to process all customers. Additionally, it is the objective to entirely use the capacity of the first milkrun if sufficient and only use the second milkrun if necessary. The same logic applies to the succeeding milkruns that also will only be used when needed.

To calculate the solution the earliest starting point for every single milkrun is determined by checking its current tour and its expected time of return. The ants in the algorithm try to place as many customers satisfying the time windows on the first milkrun with the given starting time. By estimating the duration of the tour, the return time and thus the next potential starting time for a tour on the same milkrun can be calculated. This procedure is repeated by the artificial ants of the ACO until there is no further possibility to assign demands to this milkrun. The same procedure is then applied to the second and all following milkruns. This leads to solutions with minimal requirements of vehicles for the current demand.

The second aspect considers the fact that loading and unloading times are not constant times in many scenarios. This effect arises when the old container has to be taken back with the milkrun when a new container is delivered. If the material in the old container has not been consumed entirely it has to be shifted to the new container which takes a relevant amount of time and in consequence should not be neglected in order to obtain an optimal solution. As the consumption of material is approximately proportional to time the optimum would be to arrive exactly in the latest point of time for delivery. If the milkrun arrives to early, the time that is needed to shift the material from the old container to the new one can be calculated by

$$t_{loading} = c_c + c_v \cdot (t_{latest} - t_{arrival}) \quad (4)$$

In this formula c_c is the constant loading time. It describes the time that is necessary for the driver to unmount his vehicle, walk to the trailer, move the new container to the station, move the old one back to the trailer, walk back to the vehicle and mount it again. This time is the same for all stations and does not depend on the fill level of the container as it does not take these steps into account. In contrast, the parameter c_v describes the variable parts of the loading time, which is the duration needed for manual shifting of old parts to the new container. The constant c_v is multiplied by the time difference between the latest possible time of point to arrive t_{latest}, which is equivalent to the end of the time window, as well as the actual time of arrival at the station $t_{arrival}$.

For the algorithm to work, it is mandatory to optimize for the shortest time for all tours as then variable service times can be considered in the objective function. By minimizing the total

time of the milkrun, the algorithm automatically uses the optimum between shortest driving times and short loading times which in most cases will be competing objectives.

As mentioned before, the ACO needs a heuristic value for each edge of the graph to efficiently calculate good solutions. The function to determine this value needs to be adapted as well as the representation in the graph. In the standard representation in the graph, every node has a certain service time which can be requested by the ants during the construction of the solution to determine whether this customer is a proper next stop for the milkrun or not. This behavior totally changes when considering variable loading times. The node has to calculate the specific loading time regarding the arrival time of the milkrun. In consequence the time that is used for the calculation will be different for every constructed solution. As this time is one of the major aspects in the function to calculate the heuristic value, the heuristic value as well changes for every ant that is constructing a solution. As the pheromone concentration will not change that fast, this can lead to bad convergence behavior if the weighting between heuristic value and pheromone concentration is not chosen properly.

The third extension to be discussed in this paper is the allowance of urgent orders. In practice those orders can occur if a material request has been submitted to late or if the material itself was not available fast enough for delivery. Problems of this kind won't occur in theoretical benchmarks but lead to severe trouble when testing the algorithm in practice if not considering deviations of this kind.

To catch those problems different strategies have to be implemented in the algorithm. While in the normal case a time window must never be neglected for those urgent deliveries it might not be possible at all to find a solution to deliver these materials on time. Thus the strategy must be to place those deliveries on the next milkrun that arrives on the station. The remaining places on the milkrun can be assigned by the ACO as described above with the difference, that not the first milkrun is favored before all others but the currently available one.

5. Embedding the algorithm in a dispatcher for application

As can be seen in Fig. 3, the solution has to be post processed for dispatching the milkruns. The algorithm calculates an optimal solution for the given problem but does not decide whether a milkrun should start immediately or wait for additional material requests to leave the depot. This is done by a separate module that handles the current state in the factory to optimize the starting of the milkruns.

The following rules are the minimum set that should be obeyed by the dispatcher, they can be extended depending on the use case:

- A milkrun should not leave the depot before it is filled to its maximum capacity. This is especially necessary during periods of little demand, when there is enough time to wait for additional material requests that can be placed on the same milkrun. This behavior should be ignored if the material requests on the milkrun can only delivered to the station when the milkrun leaves immediately.
- A milkrun that contains urgent orders as described above has to leave as soon as possible, no matter if there is still enough space for orders that potentially could arrive later.
- Even if a full milkrun could leave later and would still be on time it should leave as soon as possible. The fact that the milkrun is available earlier again predominates the effect of a shorter total travel time due to shorter variable service times.

For every solution that is computed by the ACO these rules are applied and decided whether a milkrun should start at once or not. In the case of postponing the start of a tour, the dispatcher will automatically initiate the tour as soon as it becomes urgent without the need of a recalculation by the ACO.

The dispatcher receives all its information from the ERP-System. To setup the tool, information about the plant, position of stations and parameters for the calculation are required. In order initiate the calculation with the ant colony optimization the only additional information being required is the demand of the stations including their latest delivery time. This information is available in most companies as it is also necessary for retrieving the material from the warehouse, thus it only needs to be passed to the algorithm.

The relations between the dispatcher, the incoming data, the milkruns and the algorithm are displayed in Fig. 4.

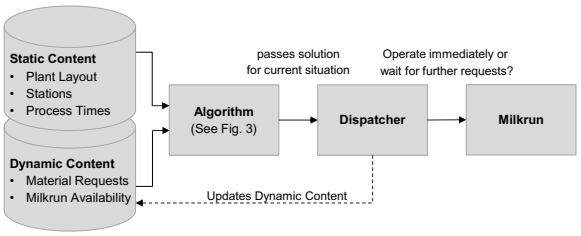

Fig. 4. Relations between the components of the logistics optimization

6. First results

To demonstrate the abilities of the algorithm, theoretical benchmarks as well as an implementation in a physical factory was performed. The theoretical examples showed that the ACO algorithm leads to results that come close to the best known solutions for the given problems. When considering variable service times no benchmark with competing approaches was possible but the comparison between taking those variable times in account or not showed very clearly that it is worth the effort of the extended complexity.

The practical implementation of the concept was done in a part of a production line of vehicle bodies. Therefore the algorithm was implemented in a program written in C#, which provided the necessary interfaces to the production systems in

order to obtain the information about the material requests. The material of the stations being covered by this test was provided in containers that had to be exchanged one by one, which means that when delivering a new full container, the empty one had to be taken in exchange. The order of the calculated tours where handed to the drivers in a printed form, containing all necessary information about stations as well as latest delivery times. The maximum time from a request at a station untill the delivery of the material was 30 minutes.

To gather the most precise values for the parameters c_c, c_v and the average milkrun speed, a regression analysis using the least squares method was carried out. The total traveling time of the milkruns was measured while sending the vehicles on predefined tours. Using the determined parameters to predict the return time of the vehicles demonstrated that the model as well as its parameters fitted very well to its real counterpart in production.

Calculation time of the algorithm was limited to two seconds for the practical use case. Although this limitation definitely deteriorates results in theory, the quality of the results was still absolutely sufficient for the given set of material requests in the practical use case which did not exceed 20 items at the same time. For bigger scenarios – which are very likely to occur – this time will need to be longer, but simulations showed that Calculation Times of 20 Seconds should be sufficient for most situations depending on the demand of solution quality.

The amount of material requests during the test was on a normal level. Departure times and returning times of the milkruns where tracked while using the algorithm. The experiment result data showed, that one of three milkruns was not necessary during the whole experiments. Simulations on historical requests however revealed that there are peaks where the third vehicle is absolutely necessary. As the algorithm currently does not provide any functionality to predict when additional capacity is needed the additional transport capacity has to be available all the time, however this lack in functionality is part of the ongoing research to exploit further potentials.

The tests proved on the one hand side that the assumptions of the concept were valid and on the other hand side that there were significant improvements over existing concepts. It also outlined the importance of the reaction to urgent orders as those are frequent use cases in real production lines. In the tests it was possible to decrease the number of vehicles to be used by up to one third without applying any changes to the adjacent processes in production. Detailed results about the performance of the algorithm including benchmark results will be presented separately in detail.

7. Conclusions and future directions

The ant colony optimization has proven its potentials in many fields including the basic VRP. As several enhancements to the algorithm were introduced in this paper it was possible to prove the ability for the use of real milkrun logistic problems. The tests confirmed the hypothesis that the ACO is a concept well prepared for problems like milkrun logistics, in theory as well as in practice.

Additionally, as pointed out in the first results it was possible to significantly reduce the number of vehicles during long times, however there is currently no chance to predict the necessary number of vehicles not even for a short period in the future. This is a major emphasis for the upcoming research which will enable to practically use the algorithm for the milkrun scheduling while being able to significantly reduce the number of necessary vehicles.

References

[1] Günthner WA, Klenk E, Galka S. Stand und Entwicklung von Routenzugsystemen für den innerbetrieblichen Materialtransport: Ergebnisse einer Studie. fml Lehrstuhl für Fördertechnik Materialfluss Logistik, München; 2012.

[2] Gyulai D, Pfeiffer A, Sobottka T, Váncza J. Milkrun Vehicle Routing Approach for Shop-floor Logistics. Procedia CIRP 7; 2013. p. 127–32.

[3] Dantzig GB, Ramser JH. The Truck Dispatching Problem. Management Science 6(1); 1959. p. 80–91.

[4] Lahyani R, Khemakhem M, Semet F. Rich vehicle routing problems: From a taxonomy to a definition. European Journal of Operational Research(241); 2015. p. 1–14.

[5] Donati AV, Montemanni R, Casagrande N, Rizzoli AE, Gambardella LM. Time dependent vehicle routing problem with a multi ant colony system. European Journal of Operational Research 185(3); 2008. p. 1174–91.

[6] Lenstra JK, Kan AHGR. Complexity of vehicle routing and scheduling problems. Networks 11(2); 1981. p. 221–7.

[7] Dittes F-M. Optimierung: Wie man aus allem das Beste macht. Springer Berlin Heidelberg, Berlin; 2015.

[8] Savelsbergh M. Local search in routing problems with time windows. Ann Oper Res 4(1); 1985. p. 285–305.

[9] Colorni A, Dorigo M, Maniezzo V. Distributed Optimization by Ant Colonies. Proceedings of ECAL91 - European Conference on Artificial Life. Elsevier Publishing. Paris; 1992, pp. 134–142.

[10] Teschemacher U, Reinhart G. Enhancing Constraint Propagation in ACO-based Schedulers for Solving the Job Shop Scheduling Problem. Procedia CIRP 41; 2016. p. 443–7.

[11] Dorigo M, Maniezzo V, Colorni A. Ant System: An Autocatalytic Optimizing Process. Technical Report 91-016. Milano; 1991.

[12] Flórez E, Gómez W, Bautista L. An ant colony optimization algorithm for job shop scheduling problem. IJAIA 4(4); 2013. p. 53–66.

[13] Dorigo M, Stützle T. Ant colony optimization. MIT Press, Cambridge, Massachusetts; 2004.

[14] Zwaan S van der, Marques C. Ant Colony Optimisation for Job Shop Scheduling. Proceedings of Third Workshop on Genetic Algorithms and Artificial Life; 1999.

[15] Gambardella LM, Taillard E, Agazzi G. MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. New Ideas in Optimization. McGraw-Hill, London; 1999, pp. 63–76.

[16] Solomon M. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. Operations Research 35(2); 1987. p. 254–65.