Component Analysis Based Approach to Support the Design of Meta-heuristics for MLCLSP Providing Guidelines

Luis Filipe de Araujo Pessoa, Carolin Wagner, Bernd Hellingrath
Department of Information Systems
Westfälische Wilhelms-Universität Münster (WWU)
Leonardo-Campus 3, 48149 Münster, Germany
{filipe.pessoa, wagner, hellingrath}@ercis.de

Fernando Buarque de Lima Neto*
Departamento de Engenharia da Computação
Universidade de Pernambuco (UPE)
Rua Benfica, 455, Madalena, Recife, Pernambuco, Brasil
fbln@ecomp.poli.br

***Abstract*—Today's supply chain are highly complex and globally set-up underlying a constant change with increasing speed. This has to be reflected by the planning processes and algorithms being utilized in the different stages of a supply chain. In the context of production planning, meta-heuristics are usually applied due to their ability to handle high complex problems. As a consequence, these algorithms require adaptation to the new scenario or even new solution approaches/strategies have to be devised. However, designing a meta-heuristic of good performance for a problem is a hard task, since it requires deep knowledge on the problem, as well as on the meta-heuristic side. Therefore, the existence of supporting guidelines for meta-heuristics might ease and speed-up the adaptation or design of these algorithms to better cope with the problem. In this paper, meta-heuristics are deconstructed into its components and an approach for component-based analysis is proposed to gain knowledge about their performance and how they perform the search. Based on the results of this analysis, guidelines can be devised. The proposed approach is applied for analyzing components of a good performing Genetic Algorithm (GA) for multi-level capacitated lot sizing problem (MLCLSP) and initial guidelines for the construction of GA in the domain of MLCLSP are generated.**

I. Introduction

Nowadays, supply chains show an ever increasing complexity due to several influences such as globalization of the supplier base and heterogeneity in customer needs [1]. Due to competitiveness and ever growing requirement of global markets they are underlying a constant change with increasing speed [2]. Changes in the supply chain are reflected in the planning process and algorithms being utilized in the different stages of a supply chain. In order to handle the ever changing requirements of global markets and the responding supply chain, the problem structure of planning algorithms needs to be adapted accordingly. In the majority of cases planning problems are hard to solve due to high problem sizes and many constraints. A focal planning decision in the supply chain is the determination of lot sizes. A well-known example is the multi-level capacitated lot sizing problem (MLCLSP). In addition to the determination of optimal production volumes and production periods under consideration of inventory holding costs and setup costs, the MLCLSP includes multiple production levels and capacity restrictions. In order to find good solutions in reasonable amount of time for suchlike problems, the usage of meta-heuristics is favored due to their flexibility [4] and ability to cope with complex optimization problems [5].

The design of an efficient and effective meta-heuristic is a laborious task. Even though, a lot of research is conducted in the field of meta-heuristics, they still represent black box methods [6]. Little is known about the characteristics of each meta-heuristic component and how they contribute to the search process. In this context, the component is referred to as an exchangeable part of a meta-heuristic procedure. Selection and composition of suitable components usually requires deep knowledge on the problem domain and on meta-heuristics.

In order to support a good design of meta-heuristics, some guidelines have already been proposed in the literature. Most publications support either the setting of parameter values [7], the choice of a particular component [8] or portray general principles for good metaheuristic construction [9]. Furthermore, the recommendation of meta-heuristic components and their combinations that results in one or more intended design aspects of the algorithm (e.g. intensification, diversification, the trade-off between solution quality and execution time, among others) is of importance. This could be achieved by a detailed analysis of the different components, their specific implementations and combinations throughout the search process. Although such guidelines would likely be dependent on the problem under analysis, they could ease and speed-up the adaptation or design of algorithms to better cope with other problems that have similar characteristics.

With this paper, we aim to propose an approach for the development of supporting guidelines on the example of the MLCLSP. Our contributions consist in finding an alternative way to evaluate meta-heuristics in supply chain planning by means of analyzing its components, their variants and combinations to the search process. The proposed approach could enable a better understanding of meta-heuristics and may provide fundaments for defining guidelines to support meta-heuristic construction in the area of supply chain management.

The structure of the paper is as follows: Section 2 introduces the procedure for component analysis and prospection of guidelines. In Section 3 the proposed procedure is evaluated and

* Visiting Professor at the University of Johannesburg, currently on Sabbatical at the Westfälische Wilhelms-Universität Münster

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.149

validated on the MLCLSP. Experiments are analyzed in Section 4, followed by the identification of resulting guidelines in Section 5.

II. Procedure for Component Analysis and Prospection of Guidelines

The design of a meta-heuristic for a problem is often based on previous knowledge and intuitions of its designer. Since there are hundreds of techniques [10], this task is becoming even more difficult due to the large amount of possible designs. Little is known about the reasons why a particular algorithm finds good or bad solutions, or which of its components are important in the search process. All the above commented problems have to be tackled by a deeper analysis of meta-heuristics.

In Watson et al. [11] a procedure was introduced to gain deeper knowledge of meta-heuristic behavior. They developed a deconstruction-based analysis of a particular version of Tabu Search (TS) which solves the Job-Shop Scheduling problem. The findings from such analysis enables to understand the benefits of some supplementary TS components, as well as the conclusion that the choice of tabu search is not responsible for the good results achieved by the meta-heuristic procedure. Main results of Watson's analysis is to determine whether different trajectory-based meta-heuristics are able to achieve results similar to the specific version of TS under analysis. This is pursued via the elaboration of a common scheme for instantiating different core meta-heuristics (i.e. meta-heuristics in their canonical form) and assessing whether they present as good results as the core TS when using the same operators. Later on, the impact of long-term memory and different probabilities for intensification and diversification are analyzed in a quantitative and qualitative manner.

The approach presented by Watson et. al for a component-based analysis portrays a good method for gaining knowledge on how meta-heuristics work. However, even more knowledge can be retrieved by the use of qualitative measures for assessing the behavior of components in regards to the search process [12]. Additionally, the analysis of several variants of core meta-heuristic components would enable a better understanding concerning their performance. The results of this thorough analysis could then be used for prospecting useful guidelines that support a faster design of an efficient meta-heuristic for a problem at hand, as shown in Fig. 1.

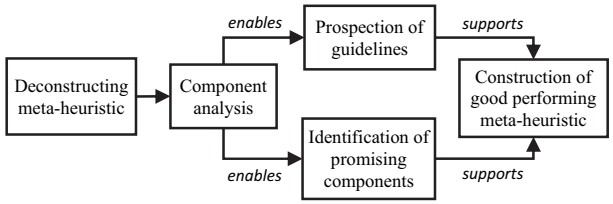


Fig. 1. Overview of the proposed processes for supporting the rapid construction of good performing meta-heuristics.

As a first step in the direction of supporting a rapid construction of good performing meta-heuristics, our goals for this work are (i) extending Watson's approach by including more qualitative measures for better assessing the behavior of meta-heuristics, (ii) analyzing several variants of core meta-heuristic components; and (iii) exemplary generating a guideline for a given problem and algorithm. Impacts of components on the search process and on the fitness has to be evaluated in an isolated manner. For this purpose, a procedure based on Watson's approach was conceived, comprising a first draft of a future procedural method for component analysis.

Fig. 2 presents the procedure proposal. The Watson's procedure is enhanced by the usage of behavior measurements, the analysis of core component variants and the elaboration of guidelines. It is a top-down approach and comprises four processes: (i) selection of a meta-heuristic, (ii) analysis of different components, and (iii) prospection of guidelines. The first process relates to the identification of good performing meta-heuristic for the problem at hand. Then, the impact of components on the search process and on the fitness has to be evaluated in an isolated way.

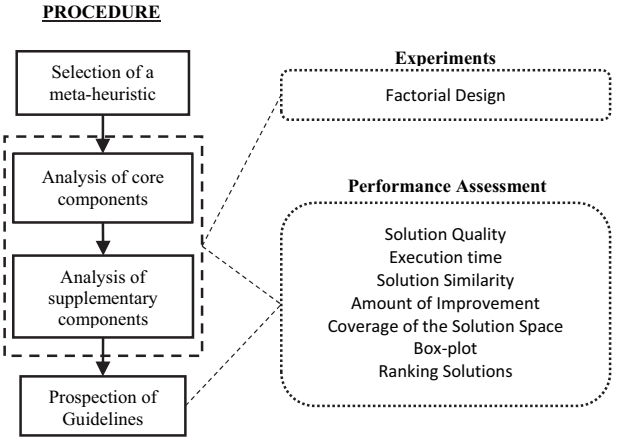


Fig. 2. Proposed procedure for component analysis and prospection of guidelines.

The component analysis encompasses two steps: (a) analysis of core components and (b) analysis of supplementary components. Both analyses aim at identifying the contribution of each component to the search success and at generating information to support the prospection of guidelines. However, each analysis focuses on a different set of components. For the first step, only variants for the core meta-heuristic components are selected. Furthermore, a set of promising and well-established configurations of those components are defined and performance is investigated in an isolated way. The configuration that results in the best algorithm's performance is taken for the second step analysis. In this step, supplementary components are iteratively included in the basic structure and further investigation regarding their contribution is conducted. Both analyses provide results for the identification of guidelines in terms of suitable configuration combinations and the application of supplementary components.

Fig. 2 also presents the experiment design and measures which are used for the analysis of components and resulting prospection of guidelines. Experiments are performed using a factorial design. Moreover, several measures can be used for assessing performance and behavior of meta-heuristics. Besides solution quality and execution time, behavioral measures enable the understanding of how an algorithm performs the search. In addition, box plot and ranking of solutions can be carried out on

the measures in order to enable and substantiate conclusions regarding the meta-heuristic components.

The next section shows how the proposed procedure is applied for the analysis of meta-heuristic components and defining guidelines. Given that MLCLSP represents a focal planning decision in the supply chain, the procedure is applied to investigate meta-heuristic components in this domain.

III. Application of the Procedure in MLCLSP

The above presented component analysis based procedure is validated and evaluated for the multi-level capacitated lot sizing problem.

A. Meta-heuristics in MLCLSP

Several meta-heuristics have been proposed in the literature and applied to the MLCLSP. Even though, they aim the same problem, formulations differ slightly in product structure and overtime acceptance. Procedures are mainly characterized by two different solution representations: production quantities and binary setup variables. The latter reduces the optimization problem to the machine setup decision, which is used in the mathematical programming solver for calculating the appropriate lot sizes. The choice of the solution representation has impact on the definition of the neighborhood structure and operators. While the quantity representation considers quantity shifting between periods, the binary representation defines flip actions of the variables. In many cases, the setup representation is favored in meta-heuristic procedures due to its simplicity and resulting reduced amount of variables to be optimized. In addition, approaches can be classified into basic and hybrid algorithms, of which the latter combines an additional optimization technique with meta-heuristics.

TABLE I. depicts an overview of identified publications, the applied meta-heuristic algorithms, its hybrid method and the solution representation. Basic approaches for meta-heuristic application in MLCLSP applying Simulated Annealing (SA), Tabu Search (TS), Genetic Algorithm (GA) and Evolution Strategy (ES) are proposed by Helber [15]. All algorithms apply a binary representation and a considerably problem-unspecific implementation with a neighborhood consisting of variable flip operations. Barabarosoglû [18] and Özdamar [19] both solve the MLCLSP with simulated annealing and by shifting production quantities each iteration. While the former evaluates different feasibility options, the latter includes a Lagrangian Relaxation (LR) approach to identify the initial solution. Toledo [21][22] proposes a genetic algorithm featuring a hierarchical structure of solutions and a mathematical optimization procedure. A Max-Min Ant System (MMAS) is applied by Almeder [25]. The meta-heuristic evaluates a binary setup pattern including pheromone-based partial fixing.

B. Choice of Algorithm

In accordance with the above described procedure, an appropriate MLCLSP meta-heuristic has to be selected for analysis. Most interesting for investigation is the best performing approach since it enables a deeper understanding on which components contribute to the good results. However, the assessment and comparison of meta-heuristics is a challenging task. On the one hand, evaluation is almost exclusively based on the final solution quality, ignoring the search process and thus discarding important information for the evaluation. On the other hand, comparison between different methods is generally not feasible without re-implementing due to different test data sets as well as different measurements and aggregated values. An assessment is thus only targetable for an agreed-upon test environment [1] and in the present case only possible to a limited extend. Consequently, the choice has to be based on other criteria.

TABLE I. Overview of Metaheuristic Approaches in MLCLSPs

Publication	MH algor.	Hybrid method	Representation
Chen [13]	LS	LR	Quantities
Zhao [14]	VNDS	LS	Quantities
Helber [15]	SA, TS, GA, ES	-	Setup
Kuik [16]	SA, TS	LP relaxation	Setup
Berretta [17]	SA/ TS	Heuristic method	-
Barabarosoglû [18]	SA	-	Quantities
Özdamar [19]	SA	LR	Quantities
Xie [20]	GA	-	Setup
Toledo [21] [22]	GA	Fix-and-Optimize	Setup
Berretta [23]	Memetic	Heuristic	Quantities
Pitakaso [24]	MMAS	Decomposition	Parameters
Almeder [25]	MMAS	LS	Setup

As a suitable meta-heuristic procedure, the two meta-heuristic approaches presented by Toledo et al. [21][22] emerged. Besides the good results achieved for established data sets, they define a recent approach which applies one of the most popular meta-heuristic algorithms (i.e. GA). In addition, the procedures comprise components that are said to perform well for other problems as well as a heuristic procedure [21]. Both approaches are essentially the same algorithm and contains the same set of components (with distinct values for some parameters) applied to solve two different MLCLSP formulations: allowing overtime [21], and considering backlogging and setup of product families [22]. In this paper we investigate the meta-heuristic components for the MLCLSP with overtime [21].

C. Algorithm Description

The realization of a component-based meta-heuristic evaluation requires a common understanding of the term component. Several publications employ the component term among them [11] and [26], yet no generally accepted definition exists. In addition, the meaning of a component differs between characteristics of an approach and deconstruction of meta-heuristics into its constituent parts. In the present work, our understanding of the term is the deconstruction of the meta-heuristic into search strategy elements and configuration elements. The component depicts an element and its specific implementation. When necessary, components can also be assigned characteristics. As an example, the components proposed by Toledo et al. [21] are presented in Fig. 3.

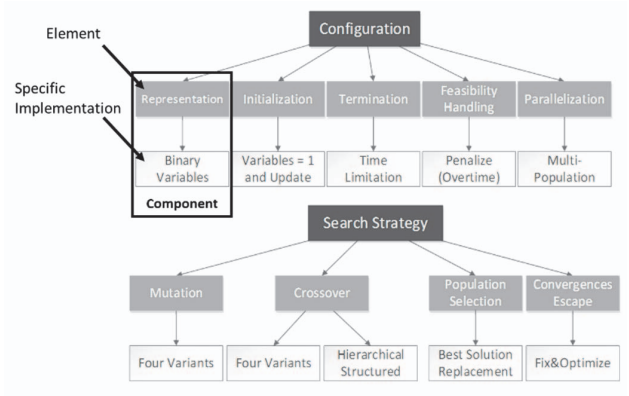


Fig. 3. Overview of components used in MGAFO procedure.

The meta-heuristic procedure that was chosen for analysis is the multi-population genetic algorithm combined with the fix-and-optimize heuristic (MGAFO) presented by Toledo et al. [21]. It consists of five configuration and five search strategy elements (cf. Fig. 3). The procedure makes use of a setup variable representation. Exceedances of capacity are penalized in the objective function. In order to obtain initial solutions, variables are fixed to 1 and the resulting quantities are calculated. If the production quantity has zero amount, the setup of this period is removed. A time limitation on 180 seconds is set on the execution of the algorithm. The procedure also includes multiple populations (i.e. 3 populations) arranged in a ring design. After all populations have converged, the best solution is spread among populations.

In addition to these configuration components, four mutation and crossover variants are defined and randomly selected during execution. Mutation varies between flip of one setup variable and two variables of either the same period, the same product or random variables. Crossover can either be uniform, single point by product or period or in form of a four sector (i.e. submatrices) crossover. In order to select parents for crossover, the population is arranged in a quaternary tree structure. One leading solution is followed by four supporting solutions (in total 21). A leader and its connected supporter are randomly chosen for mating. If the offspring exhibit better fitness, it replaces the supporting parent. After all populations have converged (i.e. no supporting parent was replaced by the offspring), the best individuals migrate to other population, and the best overall solution is improved applying a fix-and-optimize heuristic. In this heuristic, setup variables are iteratively divided up into fixed and optimized variables applying a window size on both the products and periods in a rolling horizon basis. The resulting mixed linear program is solved optimally and the best solution is updated accordingly for each rolling horizon.

D. Configuration of Experiments for the Analysis of Components

The MGAFO, as described above, comprises three supplementary components (multi-population, hierarchical structure and the fix-and-optimize heuristic) and five core GA components (initialization, selection, crossover, mutation and replacement). Although the MGAFO performs well, it is not clear which component contributes the most to the high quality solutions. For this reason, the analysis of supplementary components and the prospection of guidelines regarding their use for MLCLSP are main focus of this investigation.

Both core and supplementary components are analyzed for class B+ in accordance to the procedure proposed in the previous section. Class B+ is a group of problem instances defined by Stadtler [27] with 10 products, 3 machines and a time horizon of 24 periods. It comprises 19 instances of the problem. Unlike class A+, it takes setup times into account, which leads to an increase complexity due the inclusion of additional constraints, and it is more often used in the literature than class A+.

The first step analysis is conducted on the five core GA components. In addition to the specific implementation of core components used by MGAFO, other variants are also taken into consideration. They are either modified versions of those used in the algorithm, or variants commonly used and taken from the literature: *Initialization*: (3 variants) random; as defined in MGAFO; and, half random / half as defined in MGAFO; *Parent selection*: (3 variants) tournament with size of five; roulette wheel; and, random between tournament and roulette wheel; *Crossover*: (7 variants) uniform with three different uniform rate (0.1, 0.25 and 0.5); one point product; one point period; submatrices; and, as defined in MGAFO; *Mutation*: (7 variants) probabilistic with three different probabilistic rate (0.015, 0.05 and 0.1); random one flip; two flip product; two flip period; and, as defined in MGAFO; and, *Replacement*: (4 variants) elitism with three different elitism rate (10%, 25% and 50% of the best solutions are passed to the next population); and, best solution replacement, since it is used in MGAFO. The elitism with different rates are further referred to as soft elitism.

In order to avoid the combinatorial explosion of all 1,764 possible combinations of the basic GA components, a selection of promising components precedes the factorial analysis of possible configurations. For this selection procedure, all core GA components are set according to the variants used by MGAFO. Since the hierarchical structure is not defined as a core component, the selection component is set to random between tournament and roulette wheel. Then, at each time one component is analyzed separately and the performance is investigated in an isolated way. For example, for analyzing the crossover variants, all other core GA components are fixed but the crossover. The same procedure is carried out for all other core GA components. Subsequently, the two most promising variants of each component are selected. Experiments on all possible combinations for these components are conducted in order to identify inter-component dependencies.

The best configuration of core components obtained from the factorial analysis is taken as basis for the second analysis step. In this step all possible combinations of the supplementary components are tested. Given the data obtained from both experiments, guidelines in terms of suitable configuration combinations and the application of supplementary components can be identified.

The amount of instances used in each analysis step is gradually extended. The selection of promising component variants uses 20% of the instances (i.e. G521132, G512131, K511141 and K522131); the factorial design over the selected core components uses 50% (i.e. G521132, G512131, G511132, G522132, G522130, K512132, K521131, K521142, K511141

and K522131); and, the factorial design over the supplementary components uses all. Half of the instances selected in each analysis considers the general operation structure, while the other half considers the assembly operation structure.

Computational experiments are implemented in JAVA 8 and executed on an Intel Xeon CPU E31270 3.4GHz, 16GB of RAM, Windows Server 2008 R2 Standard 64 bits. Linear programs are solved with CPLEX. The number of calls to the fitness function is used as a stopping criteria instead of the execution time, since this is not the same environment as the one used by the authors of MGAFO [21], as well as no information concerning the implementation language was reported in the referred paper. The selected stopping criteria enables comparison of results without the necessity of having the same computational environment. For the second step analysis, which includes the multi-population structure (3 populations in total), the stopping criteria is set to 45,000 calls to the fitness function, resulting in an average of 15,000 calls for each population. For this reason, the stop criteria is set to 15,000 calls to the fitness function for the analysis of core GA components. Each configuration of components were executed 15 times for each instance.

E. Performance Assessment

In order to gain knowledge about the performance and behavior of the different component configurations for the MLCLSP, several measurements are employed. The identification of most promising configuration of components is based on the dispersion of results (i.e. box-plot), and ranking of configurations with respect to all considered instances and executions. Regarding the latter analysis, for each instance l all configurations c are ranked over all executions k according to the final solution. The rank r_{clk} of each configuration is summed up for all instances, resulting in the final rank R_c (1).

$$R_c = \sum_{l=1}^{L} \sum_{k=1}^{K} r_{clk} \tag{1}$$

In addition, other measures are used to provide information regarding behavioral aspects of meta-heuristics. Those information are used to understand why in general a set of components achieved better solution qualities than others. This could expose possible weaknesses or strengths of components, and might uncover potential directions for improvements. For this purpose, the following measures are used: *improvement over iterations* – shows how the best solution improves over iterations; *Solution similarity* [27] – measures how much a candidate solution differs from others in a population; *Amount of improvement* [27] – measures how much a solution improves after including one or more components; and, *Coverage of the Solution Space* [27] – measures the ability of an algorithm to generate distinct candidate solutions.

IV. Result of Experiments

Results of experiments are analyzed in order to gain deeper knowledge about the components and their impact on the search process. An error in the original problem formulation in [21] was discovered during the implementation of the approach. The upper bound B_{jt} in the setup constraint disregards the additional availability of overtime capacity. Since the upper bound parameter cannot be defined from the overtime capacity variable we define the upper bound parameter only by the total demand $d_{j(t..T)}$ of product j at period t. This is possible due to the fact that the updated problem formulation results in a greater upper bound, which might only cause higher computational time. On the other side the error in the previous formulation of the upper bound can prevent the meta-heuristic to reach better results in cases where overtime capacity is used.

A. Analysis of core GA Components

The first part of this step consists in the identification of the two most promising instantiations for each component. For this reason, the proposed instantiations are evaluated in an isolated way. The assessment of each component is based on ranking of the fitness values for each instance. Best performing instantiations are: *Initial Solution*: MGAFO/ Random, MGAFO; *Parent Selection*: Tournament, Random; *Crossover*: One Point Product Crossover, Submatrices Crossover; *Mutation*: Random One Flip, MGAFO; *Replacement*: Soft Elitism (0.25), Soft Elitism (0.5). For most components, results deduce a clear rank order of the corresponding instantiations. In the case of the replacement component the ranking is not definite. Soft elitism however achieved significantly better results than strict elitism. Thus, only the two best variants are considered.

The second part of this step is the factorial analysis of different configurations and the investigation of related mutual dependencies. The assessment of the best set of instantiations for the second step is realized via a box-plot representation and the above described ranking procedure. Box-plots enable the visualization of variability of experiments with regard to their fitness value. Due to their one-dimensional shape, visual comparison of different configurations is facilitated. Results indicate configuration performance differences among instances. None of the configurations clearly outperforms all the others considering single instances as well as the combination of all instances. For this reason the ranking procedure is considered for the selection based on the minimal overall rank. The best performing configuration based on the ranking approach is composed by the following instantiations: MGAFO/ random (initial solution), random (parent selection), one point period (crossover), MGAFO (mutation) and soft elitism 0.25 (replacement).

The examination of the ranks of the evaluated configurations allows for conclusion to be drawn regarding the impact of instantiations and combinations on the overall fitness. The ranking of the configurations indicate a cluster of similar good results for the best eight configurations. Those configurations are characterized by the same crossover and mutation variant. This statement is emphasized by Fig. 4. The bar chart depicts the sum of the position in the ranking of each instantiation considering all instances. The dotted line displays the minimal value. Results show that variations in crossover and mutation seems to have main impact on the final meta-heuristic performance, while the initial solution, parent selection and population selection only slightly influence the solution quality.

In addition to the evaluation of different configurations, results are investigated regarding different metrics and resulting implications of components on the search process. In this section, the improvement over iterations, the solution similarity

and the coverage of solution space is considered. For this purpose, the best and the worse configuration are chosen and compared against each other. Analysis has been performed on all instances, yet for visualization purposes values are exemplarily taken from experiments of instance K511141. Only negligible deviation between instances has been observed. In order to assess variability, minimum, average and maximum values are calculated for each metric. Fig. 5 illustrates the results for the fitness improvement metric. Two conclusions can be obtained from the time series patterns. On the one hand, greater improvement in the first iterations lead to better final fitness (for the investigated number of iterations) even though results converge. On the other hand, the improvement pattern is characteristically for the specific configuration. Deviations between different experiments of the same configuration are small. Further investigation revealed that the patterns are representative for the two different types of crossover. The one-point period crossover leads to faster decrease in the fitness value, while the submatrices crossover requires more iterations for the same improvement. This metric emphasizes the importance of the crossover component in meta-heuristic design.

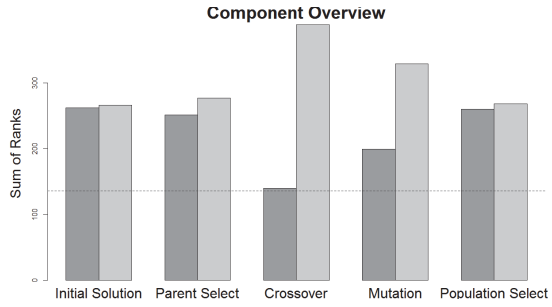


Fig. 4. Comparison of component influence.

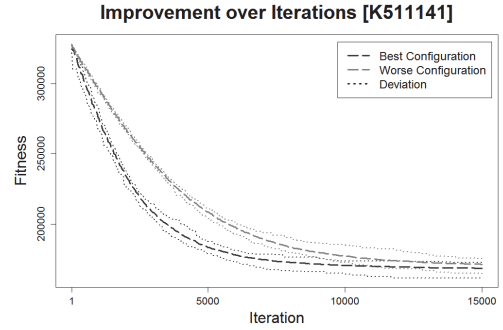


Fig. 5. Improvement over Iterations for instance K511141.

The second metric analysis considers the solution similarity. Fig. 6 presents the similarity value between the best solution and the remaining population for each iteration. The higher the value the more differs the population from the best solution. Therefore, low solution similarity values indicate exploitation of the search space. The mean development of the solution similarity over iterations displays a similar pattern for both configurations. On average, the best solution differs from the remaining population by less than 10 %. Nonetheless, the solution similarity is slightly higher for the worse configuration. In particular, the deviation from the mean value is larger. The higher solution similarity could indicate that the search is not exploited around the best solution, thus leading to worse performance. Given these results, a thorough investigation around the search space of good individuals is recommendable.

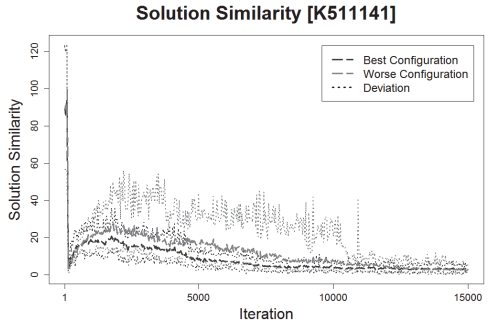


Fig. 6. Solution Similarity for instance K511141.

The third metrics comprises the coverage of the solution space. The coverage refers to the percentage of distinct solutions contrasted to the number of generated solutions during the search process. In order to explore the search space effectively, high percentage is desirable. TABLE II. depicts an overview of the mean values for this metric considering different iterations (1500, 5000, 10000 and 15000) and four configurations (best and worse configuration for each crossover variant). In general, most solutions present novel solutions. As the likelihood to generate novel solutions decreases during the search process, the coverage value decreases as well. Besides that, the crossover variant influences the generation of novel solutions. The percentage of distinct solutions is higher for the one point crossover during the first 500 iterations. Iterations 1000 to 1500 are probably more influenced by other components due to deviant behavior within the crossover variants. This metric infers that the selection of a crossover variant strongly influences the first phase of the search process. These results are in accordance with the first metric and thus strengthens the statement regarding the importance of the crossover.

TABLE II. COVERAGE OF SOLUTION SPACE

Iterations	One Point Period Crossover		Submatrices Crossover	
	Best	***Worse***	***Best***	***Worse***
1500	0.98164	0.98031	0.93778	0.93316
5000	0.97336	0.97114	0.93305	0.92624
10000	0.94768	0.91756	0.92585	0.91947
15000	0.90553	0.81692	0.91024	0.88400

B. Analysis of Supplementary Components

This analysis step focuses on the supplementary components used by MGAFO. In order to reduce computation time, the maximum window size for the fix-and-optimize component is set to seven. Due to the deterministic nature of the fix and optimize heuristic, results do not change when the procedure is called with the same inputs. Thus, it is only executed if either the best individual or the window size differs from the last execution. For representing configurations of supplementary components, the following abbreviations are used: fixed-and-optimize heuristic (FaO), hierarchical structure (Struc), multiple

populations (Multi) and the best core genetic algorithm configuration (GA).

In contrast to the analysis of core components, from the boxplot it is possible to conclude that the algorithms GA+FaO, GA+Struc+FaO and GA+Struc+Multi+FaO outperform all others. In Fig. 7 exemplarily the boxplots for instances K521131 and G521132 are shown. While other configurations present higher dispersion and median, the three aforementioned configurations have less dispersion and better solution quality. The same is observed for the other problem instances.

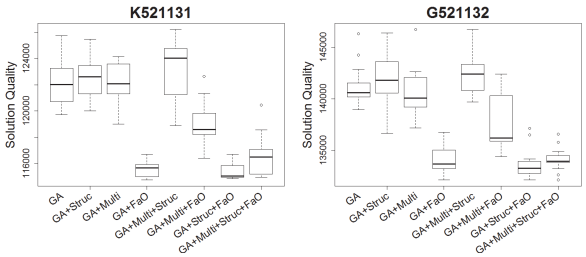


Fig. 7. Boxplot of solution quality for different configurations of supplementary components.

Among those three configurations, however, it is difficult to distinguish which outperforms the others. For this reason, the summed ranking for all configurations is computed and results are presented in Fig. 8. Algorithms GA+FaO and GA+Struc+FaO both present similar results for the problem under investigation. Taking into consideration the boxplot and the summed ranking, the following observations can be drawn:

- *Hierarchical structure*: when included into the core GA the final solution quality is diminished and, most of times, the dispersion is greater. Even worse results are achieved when included together with the multi-population. However, in combination with the fix and optimize component, results are improved.
- *Multi-Population*: only slight improvements of the fitness is observed for some instances when included in the core GA. However, combined with the fix and optimize heuristic, better overall results are achieved.
- *Fix and optimize*: This component seems to mainly contribute to the fitness improvement. With respect to the ranking and the boxplot, all configurations using FaO present better solution qualities. As can be noted, solutions are worse when multi population is used in combination with fix and optimize.

A deeper investigation on the number of calls to the fix and optimize heuristic enables a better understanding of the algorithm success rate in finding better solutions. It reveals that the success is directly proportional to the number of calls to this component. Although the maximum number of iterations and the convergence criteria (which is used for triggering the FaO component) are the same for all configurations, each different combination of components causes different convergence time. The inclusion of the hierarchical structure favors early and a higher number of calls to the fix-and-optimize heuristic because solutions are only replaced in the population if the offspring is better than the support parent, leading to a faster convergence. Additionally, the use of multiple populations results in late and reduced number of calls, since all populations need to converge in order to call the FaO heuristic.

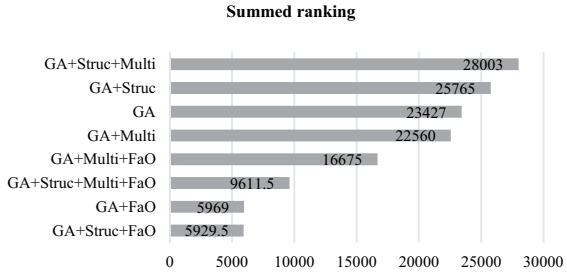


Fig. 8. Summed ranking of the supplementary components.

Fig. 9 depicts the aforementioned impact on the fitness value of the hierarchical structure and the FaO component, exemplarily shown for the problem instance K521142, for which the highest improvement in solution quality is achieved after including the FaO heuristic into the GA (i.e. 8%). The metric improvement over iterations for the configurations with and without the fix-and-optimize component overlaps each other until the FaO is called and further improves the solution. A faster improvement of solution quality is observed for configurations that use hierarchical structure.

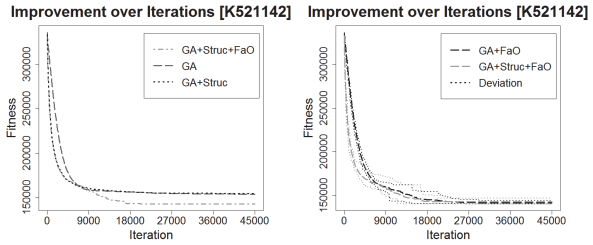
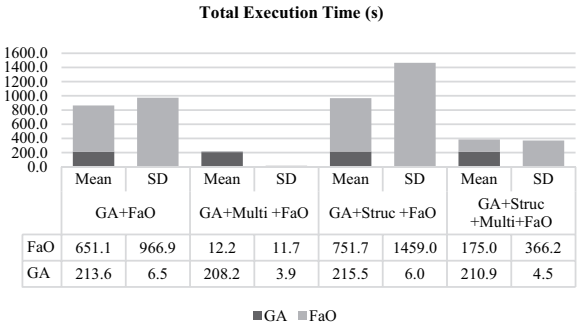


Fig. 9. Improvement over iterations for the instance K521142. Impact of hierarchical structure and fix and optimize components.

Although the fix and optimize heuristic is a very good performing component for escaping local optima, it requires high computational time. Fig. 10 depicts the mean and standard deviation of the total execution time for the different configurations including the FaO component. The total execution time includes the total time required for the FaO and for all other GA operations, including other supplementary components. The average execution time of the underlying GA remains almost the same, with slight changes on the mean and standard deviation depending on the inclusion of different supplementary components. This means that the inclusion of other components but the FaO do not imply additional time consumption. In contrast, the FaO heuristic requires much additional computation time, depending on the number of calls.

The computational cost of the FaO heuristic is further analyzed in order to compare its efficacy and efficiency across the distinct combinations of supplementary components. Fig. 11 and Fig. 12 show the share of the FaO component to the total execution cost (GA+FaO and GA+Struct+Multi+FaO, respectively) for each problem instance. The total computation cost consumed by the FaO is composed of (i) cost of executions that led to solution improvements and (ii) those that did not

improve solutions. In addition, the accumulated solution improvement after all FaO executions is reported. Shown in the figures, a substantial part of the FaO total execution time does not result in solution improvements, particular in the case of GA+FaO. Since there is no guarantee to find better values for the setup variables within the window size, it is possible that the FaO does not improve the candidate solution. Besides that, the mathematical solver takes more time to find an optimal solution when the window size increases. As Fig. 12 shows, the total cost of the FaO heuristic decreases when multiple populations are included. This is due to the reduced number of calls to the FaO component. In addition, a greater portion of the total cost led to improvement of the solution quality. Nevertheless, the accumulated solution improvement is similar for both configurations of components.



	Mean	SD	Mean	SD	Mean	SD	Mean	SD
	GA+FaO		GA+Multi +FaO		GA+Struc +FaO		GA+Struc +Multi+FaO	
FaO	651.1	966.9	12.2	11.7	751.7	1459.0	175.0	366.2
GA	213.6	6.5	208.2	3.9	215.5	6.0	210.9	4.5

Fig. 10. Mean and standard deviation of the total execution time.

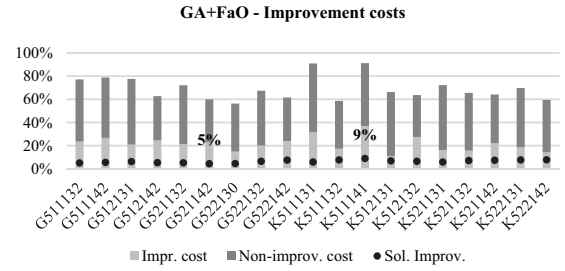


Fig. 11. Mean computational cost of the FaO heuristic in GA+FaO. Percentages on the bars indicate the minimum and maximum values of accumulated solution improvement.

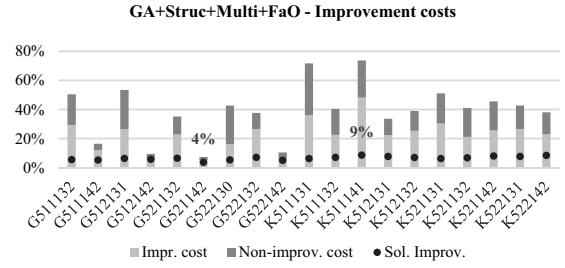


Fig. 12. Mean computational cost of the FaO heuristic in GA+Struc+Multi+FaO. Percentages on the bars indicate the minimum and maximum values of accumulated solution improvement.

The analysis of the solution similarity to the best individual indicates that the FaO heuristic collaborates for keeping the diversity of the population, due to the generation of new and best individuals. Fig. 13 exemplarily shows the solution similarity for GA+Struc and for one population of GA+Struc+Multi and their respective versions including FaO. Instances G522142 and K511141 are respectively selected, as they show the highest improvement in solution quality when the FaO heuristic is used (9% and 8%, respectively). For the GA+Struc, a higher value of solution similarity is observed in the interval from 9,000 to 27,000 iterations for the configuration that includes the FaO. Although the best solution differs greatly from the entire population within the mentioned interval, solution similarity decreases afterwards, suggesting that candidate solutions are exploiting the region around the best individual. For one of the population of GA+Struc+Multi+FaO, a similar increase of solution similarity is observed. However, at the end of the execution the best individual still differs more from other candidate solutions in relation to the version that does not use the FaO heuristic. This suggests that candidate solutions are still exploring regions which are further away from the best individual.

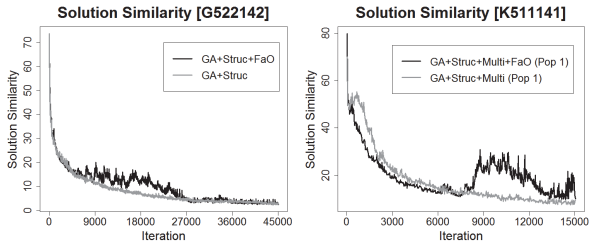


Fig. 13. Solution similarity for the instances G522142 and K511141 of the best individual for GA+Struc and GA+Struc+FaO, and for GA+Struc+Multi and GA+Struc+Multi+FaO, respectively.

V. Prospection of Guidelines for Component Selection

The selection of components in MLCLSP requires deep knowledge in meta-heuristic construction as well as in the domain of lot sizing. In order to support practitioners in the construction of meta-heuristic procedures in MLCLSP, a set of guidelines from the analysis of components and metrics are developed. The proposed guidelines depict initial results. In order to validate and possibly generalize the guidelines further experiments on different algorithms and settings are required.

A. Core Component Guidelines

The first guideline that can be deduced from the analysis of core GA components covers the impact of components on the algorithm performance. Results obtained by the rank analysis of different configurations as well as the metrics improvement over iterations and coverage of solution space emphasize the following statement:

The selection of the crossover component and the mutation component has major influence on the meta-heuristic performance.

As a consequence, a large part of the effort spent on designing a meta-heuristic for the MLCLSP should be dedicated on the instantiation of the crossover and mutation component. It is recommendable to test several variants and select the most promising combination. In the present work, results indicate that the combination of the one-point product crossover and the MGAFO mutation is a promising combination for the data under

investigation. Other components only have little influence on the algorithm performance, and thus optimization portrays a minor role.

The second guideline relates to the results from the solution similarity metric and an investigation of the promising mutation and crossover combination obtained during this analysis step leading to the following statement:

A thourough exploitation of the search space around the best solution enables the improvement of meta-heuristic performance.

The instantiation of crossover and mutation should be selected in a way that exploration and exploitation is covered simultaneously. Besides exploiting the search space in the area near to the best solution, exploration of different search spaces is required to overcome local minima. A possible explanation of why the MGAFO mutation and the one-point product crossover performs best could be that crossover enables exploration by combining different product set ups and mutation is responsible for exploiting product set ups. Thus, an exploitation of the search space around the best solutions is achieved by the mutation component. The crossover component combines promising product set ups. In general, however, a good combination of mutation and crossover is difficult to acquire. It requires a lot of knowledge and experience in the field of meta-heuristics and the application domain. In order to generate a more detailed guideline, further experiments on these components, their interaction, dependencies and the problem domain is necessary.

B. Supplementary Component Guidelines

Based on the boxplots, all configurations including the FaO heuristic present better final solution qualities. Besides that, the summed ranking and the comparison of improvement over time emphasizes this observation. While configurations without the FaO component in general converge to local optima, the use of this heuristic encourages further solution improvements. Although this component was exclusively tested with GA, other meta-heuristics which make use of the setup variables representation might profit from the usage of the FaO heuristic investigated here. Thus, the following statement can be deduced:

The fix and optimize component is a great heuristic for escaping local optima and further improving candidate solutions.

The FaO favors diversity of high quality solutions by introducing new and better candidate solutions into the population. It gives the opportunity for the meta-heuristic to escape from local optima. However, experiments revealed that it is hard for the GA to further improve solutions found by the heuristic, even though it is able to improve solutions towards the best found so far. This means that after some iterations, the FaO was mainly responsible for further solution improvements.

There are two possible reasons for that: (i) the underlying GA is not exploiting well the region, or (ii) better solutions are further away from the identified local optimum, thus requiring an exploration of the solution space. In the case of GA, one possibility to deal with this issue is by adapting the crossover and mutation to perform fewer changes in the individual as the execution approaches the end. For this reason, further investigation regarding different variants and combinations of those components, especially those that incorporates heuristics to perform fine adjustment on individuals, can lead to improvements of the solution quality.

Depending on the difficulty for solving the sub-problems, the FaO can have high computation cost, but just a small proportion results in solution improvements, as depicted in Fig. 11. Therefore, it is necessary to devise a proper criteria for triggering this heuristic and for changing window sizes, since this plays an important role for balancing the trade-off between solution quality and execution time.

Based on the comparison of the improvement over iterations between configurations with and without the hierarchical structure component, it is possible to deduce the following statement:

The use of the hierarchical structure component collaborates for a faster convergence towards a local optimum.

The reason for a faster convergence refers mainly to how individuals are replaced in the population. The hierarchy based on the solution quality is mainly used for enabling an easy and fast selection of at least one good solution. After applying the crossover and mutation, the new candidate solution is only included into the population if it is fitter than the worse parent. This strategy restricts the inclusion of diverse solutions, leading to a faster convergence. However, when used in combination with the FaO heuristic and the convergence criteria, it allows earlier and more frequent calls to the FaO. This leads to successful improvement of the best solution in a reduced number of iterations. The conclusion is supported by the analysis of improvement over iterations, boxplots, summed ranking and the number of calls to FaO when the hierarchical structure is used. Thus, the following statement can be deduced:

The joint use of the hierarchical structure component and the fix and optimize heuristic leads to more successful improvement of the best solution in a reduced number of iterations.

The multiple population component enables multiple independent executions of the meta-heuristic, increasing the chance of generating diverse solutions, due to the stochastic nature of the search procedure. Thus, it collaborates to a better exploration of the search space. With regard to the combination of the multiple population component with the hierarchical structure and the FaO, good solutions are found requiring less computation time. On the one hand, solutions are worse in comparison to those obtained by GA+FaO and GA+Struc+FaO, although difference is small as can be seen in the boxplots. On the other hand, the total time required for the GA+Struc+Multi+FaO to find good solutions is on average 45% (40%) of the total execution time of GA+FaO (GA+Struc+FaO). Thus, the following statement can be deduced:

The combination of Struc+Multi+FaO enables a better balancing between solution quality and execution time.

VI. CONCLUSION

The present paper depicts a first attempt towards a thorough component analysis based procedure that enables the generation of guidelines for an efficient and effective design of meta-heuristics. This is necessary with regard to current supply chain

requirements. Due to increasing speed of changes in the supply chain, fast adaptations of the production plans are crucial. The put forward guideline is thought of supporting the practitioner in developing a good meta-heuristic. This is done initially for the genetic algorithm. The approach was validated on the multi-level capacitated lot sizing problem (for class B+ of Stadtler's dataset) using a genetic algorithm. A set of guidelines concerning the instantiation and inclusion of components have been identified for the usage of genetic algorithm in MLCLSP. Even though the guidelines could be further improved, they portray a starting point for further research. Nonetheless, with this work a proof of concepts is attempted. Thus, the procedure and the resulting guidelines do not aim a concluded work, but rather present initial ideas for facilitating meta-heuristic design. To the best of our knowledge, no such support is provided to problems with similar characteristics, hence the deemed value of this work.

Future research will target the advancement of the presented component analysis based procedure. In order to adjust and adapt the procedure, further testing and validation of the approach is required. On the other hand, the enhancement and extension of guidelines for production planning problems need to be addressed. An elaborated catalog of guidelines for different planning problems is intended. By analyzing additional core component instantiations, further supplementary components, different meta-heuristics as well as additional planning problems more sophisticated guidelines can be acquired.

REFERENCES

[1] C.C. Bozarth, D. Warsing, B.B. Flynn and E.J. Flynn. "The impact of supply chain complexity on manufacturing plant performance", in Journal of Operations Management, vol. 27, n.1, 2009, pp. 78-93.

[2] N. Costantino, M. Dotoli, M. Falagario, M.P. Fanti and A.M. Mangini. "A model for supply management of agile manufacturing supply chains", in Advances in Optimization and Design of Supply Chains, vol. 135, n. 1, 2012, pp. 451-457.

[3] L. Buschkühl, et al. "Dynamic capacitated lot-sizing problems: a classification and review of solution approaches", OR Spectrum, vol. 32, n. 2, 2010, pp. 231-261.

[4] R. Jans and Z. Degraeve. "Meta-heuristics for dynamic lot sizing: A review and comparison of solution approaches", European Journal of Operational Research, vol. 177, n. 3, 2007, pp. 1855-1875.

[5] M. Salomon, R. Kuik, and L. N. Van Wassenhove. "Statistical search methods for lotsizing problems", Annals of Operations Research, vol. 41, n. 4, 1993, pp. 453-468.

[6] K. Sörensen. "Metaheuristics—the metaphor exposed", International Transactions in Operational Research, vol. 22, n. 1, 2015, pp. 3-18.

[7] Grefenstette, J.J. "Optimization of Control Parameters for Genetic Algorithms", IEEE Trans. Systems, Man, and Cybernetics, vol. 16, n. 1, 1986, pp. 122-128.

[8] P. W. Poon and J. N. Carter. "Genetic algorithm crossover operators for ordering applications", Computers & Operations Research, vol. 22, n. 1, 1995, pp. 135-147.

[9] A. Hertz and M. Widmer. "Guidelines for the use of meta-heuristics in combinatorial optimization", European Journal of Operational Research, v. 151, 2003, pp. 247–252.

[10] B. Xing and W.-J. Gao. "Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms", Springer: Switzerland, 2014.

[11] J.-P. Watson, A. E. Howe, and L. D. Whitley. "Deconstructing Nowicki and Smutnicki's i-TSAB tabu search algorithm for the job-shop scheduling problem", Computers & Operations Research, vol. 33, n. 9, 2006, pp. 2623-2644.

[12] A. Scheibenpflug, S. Wagner, E. Pitzer, B. Burlacu and M. Affenzeller. "On the Analysis, Classification and Prediction of Metaheuristic Algorithm Behaivor for Combinatorial Optimization Problems", Proceedings of the European Modeling and Simulation Symposium, 2012, pp. 368-372.

[13] H. Chen and C. Chu. "A Lagrangian relaxation approach for supply chain planning with order/setup costs and capacity constraints", Journal of Systems Science and Systems Engineering, vol. 12, n. 1, 2003, pp. 98-110.

[14] Q. Zhao, C. Xie, and Y. Xiao. "A variable neighborhood decomposition search algorithm for multilevel capacitated lot-sizing problems", Electronic Notes in Discrete Mathematics, vol. 39, 2012, pp. 129-135.

[15] S. Helber. "Kapazitätsorientierte Losgrößenplanung in PPS-Systemen". M&P, Verlag für Wiss. u. Forschung, 1994.

[16] R. Kuik, et al. "Linear programming, simulated annealing and tabu search heuristics for lotsizing in bottleneck assembly systems", IIE Transactions, vol. 25, n. 1, 1993, pp. 62-72.

[17] R. Berretta, P. M. França, and V. A. Armentano. "Metaheuristic approaches for the multilevel resource-constrained lot-sizing problem with setup and lead times", Asia-Pacific Journal of Operational Research, vol. 22, n. 2, 2005, pp. 261-286.

[18] G. Barbarosoğlu and L. Özdamar. "Analysis of solution space-dependent performance of simulated annealing: the case of the multi-level capacitated lot sizing problem", Computers & Operations Research, vol. 27, n. 9, 2000, pp. 895-903.

[19] L. Özdamar and G. Barbarosoglu. "An integrated Lagrangean relaxation-simulated annealing approach to the multi-level multi-item capacitated lot sizing problem", International Journal of production economics, vol. 68, n. 3, 2000, pp. 319-331.

[20] J. Xie and J. Dong. "Heuristic genetic algorithms for general capacitated lot-sizing problems", Computers & Mathematics with applications, vol. 44, n. 1, 2002, pp. 263-276.

[21] C. F. M. Toledo, R. R. R. de Oliveira, and P. M. França. "A hybrid heuristic approach to solve the multi level capacitated lot sizing problem" Evolutionary Computation (CEC), 2011 IEEE Congress on, 2011.

[22] C. F. M. Toledo, R. R. R. de Oliveira, and P. M. França. "A hybrid multi-population genetic algorithm applied to solve the multi-level capacitated lot sizing problem with backlogging", Computers & Operations Research, v.40, n. 4, 2013, pp. 910-919.

[23] R. Berretta and L. F. Rodrigues. "A memetic algorithm for a multistage capacitated lot-sizing problem", International Journal of Production Economics, vol. 87, n. 1, 2004, pp. 67-81.

[24] R. Pitakaso, et al. "Combining population-based and exact methods for multi-level capacitated lot-sizing problems", International Journal of Production Research, vol. 44, n. 22, 2006, pp. 4755-4771.

[25] C. Almeder. "A hybrid optimization approach for multi-level capacitated lot-sizing problems", European Journal of Operational Research, vol. 200, n. 2, 2010, pp. 599-606.

[26] C. Blum, and A. Roli. "Metaheuristics in combinatorial optimization: Overview and conceptual comparison", ACM Computing Surveys (CSUR), vol. 35, n. 3, 2003, pp. 268-308.

[27] H. Stadtler and C. Sürie. "Description of MLCLSP test instances", Technische Universität Darmstadt, Tech. Rep (2000).