Multi-criteria Scheduling: An Agent-based Approach for Expert Knowledge Integration

Christian Grimme, Joachim Lepping, Uwe Schwiegelshohn

Journal of Scheduling manuscript No.
(will be inserted by the editor)

Multi-criteria Scheduling: An Agent-based Approach for Expert Knowledge Integration

Christian Grimme · Joachim Lepping · Uwe Schwiegelshohn

Received: December 15, 2010 / Accepted: date

Abstract In this work, we present an agent-based approach to multi-criteria combinatorial optimization. It allows to flexibly combine elementary heuristics that may be optimal for corresponding single-criterion problems. In the multi-criteria case a smart combination of such heuristics is supposed to efficiently approximate the whole Pareto-front of non-dominated trade-off solutions. We optimize an instance of the scheduling problem $1|d_j|\sum C_j, L_{max}$ and show that the modular building block architecture of our optimization model and the distribution of acting entities enable beneficial integration of problem specific expert knowledge. We present a universal mutation operator for combinatorial problem encodings that allows to construct certain solution strategies, such as advantageous sorting or known optimal sequencing procedures. In this way, it becomes possible to derive more complex heuristics from atomic local heuristics that are known to tackle fractions of the complete problem. We show that it is possible to approximate both single-criterion problems such as $P_m|d_j|\sum U_j$ as well as more challenging multi-criteria scheduling problems, like $P_m||C_{max}, \sum C_j$ and $P_m|d_j|C_{max}, \sum C_j, \sum U_j$.

Christian Grimme · Uwe Schwiegelshohn
TU Dortmund University, Robotics Research Institute,
Otto-Hahn-Strasse 8, 44227 Dortmund, Germany
Tel.: +49-231-2634
Fax: +49-231-3251
E-mail: {*firstname.lastname*}@udo.edu

Joachim Lepping
Laboratoire d'Informatique de Grenoble
ENSIMAG - antenne de Montbonnot
ZIRST 51, avenue Jean Kuntzmann
38330 Montbonnot Saint Martin, FRANCE
Tel.: +33-476-6120-69
Fax: +33-476-6120-99
E-mail: joachim.lepping@imag.fr

Keywords Multi-criteria Scheduling · Predator-Prey Model · Parallel Machine Scheduling · Evolutionary Multi-criteria Optimization

1 Introduction

Although theoreticians and practitioners agree on the fact that real world scheduling optimization processes involve more than one criterion, the research area of algorithmic multi-criteria scheduling is a rather new topic posing many challenging, important, and unsolved problems. This might be due to the fact that optimization in the multi-criteria domain follows a fundamentally different paradigm of optimality: while in the single-criterion case only one optimum has to be found, in the multi-criteria case all optimal compromises or trade-offs have to be enumerated. For minimization problems with M criteria, a so-called optimal compromise is reached, if for a criterion q, the value f_q can only be decreased by increasing some other value f_p with $q \neq p$ and $p,q \in \{1,\ldots,M\}$. All criteria vectors $F = (f_1,\ldots,f_M)$ that fulfill this property are called members of the Pareto-front.

Since only a fistful of multi-criteria scheduling problems can be solved efficiently, many practitioners focus on the application of heuristics in order to approximate solution sets. Today, most problems can only be tackled in two ways: On the one hand, problem specific approaches may be able to solve very specific problem instances. On the other hand, black-box optimizers like randomized search or evolutionary algorithms abstract from the specific problem and offer a universal methodology.

In contrast to multi-criteria scheduling, single-criterion scheduling problems have been extensively investigated over the last 50 years. Besides a vast amount of complexity results, researchers provided numerous optimal algorithms and approximative heuristics for many problems of that domain,

see e.g. Pinedo [19] or Dutot et al. [7] for introduction and review. However, the most annoying fact is that it is so far almost impossible to make use of the comprehensive single-criterion knowledge base in a more generalized manner for the multi-criteria domain. It is easy to see that on the one hand a naive aggregation of single-criterion heuristics does not necessarily yield feasible non-dominated trade-off solutions. On the other hand, the monolithic architecture of popular multi-criteria evolutionary algorithms hardly allows to bring in any problem specific knowledge let alone to combine it in a flexible way. This drawback mainly results from the dominant selection procedure applied in most multi-criteria evolutionary algorithms which leaves almost no room to plug in special adaptation mechanisms. For this purpose, more advanced variation operators along with an alternative and more dynamic selection scheme is required which replaces the monolithic algorithmic architecture.

In this work, we propose an agent-based optimization model that offers a new way to flexibly integrate partial knowledge on specific aspects of the full problem. This model is basically inspired from the well known predation paradigm from biology but reduced to a unidirectional interaction relation between predators and preys: a population of prey is distributed on a spatial structure which is represented by an undirected graph. Predators pursue only *one criterion* each and move randomly along the edges in order to chase prey which are weak regarding that specific criterion. The presence of several predators—each representing only a single criterion—is expected to force the prey to likewise adapt to the threats from all predators and thus result in suitable trade-off solutions for multi-criteria optimization problems.

The rest of this paper is organized as follows: in Section 2, we describe the problem context of multi-criteria scheduling problems, introduce the specific problems tackled in this paper, and point out common solution strategies and complexity results. Then, in Section 3, we detail our alternative agent-based approach to multi-criteria scheduling. Next, in Section 4, we evaluate the performance of the afore defined algorithmic framework on several problems. In Section 5, we conclude this work and point to future directions.

2 Multi-criteria Scheduling Problems

In most practical scheduling scenarios and production planning tasks the consideration of only a single criterion is insufficient to provide both quality assurance and customer satisfaction. A production scheme is usually judged with respect to several criteria considering, for instance, the overall production time (Makespan), average job flow time, job delay, or total number of delayed jobs.

However, for almost 30 years, scheduling theory and algorithm design were dedicated to solve single-criterion problems. Hoogeveen [14] identifies the 1980s as a starting point for the detailed investigation of multi-criteria scheduling problems and the emergence of a research area in its own rights. Therein, a lot of energy has been put to the determination of problem complexity, demonstrating the hardness of almost all problems and taking the hope of solving those problems optimally in polynomial time. As always, this situation is the starting point for the development of heuristics which try to approximate optimal solutions in reasonable time.

In this section, we will introduce the concepts of scheduling in general, the specifics of multi-criteria problems, and a basic classification of solution strategies for this kind of problems. Then, we briefly review the theoretical and heuristic research in single and parallel machine environments. Eventually, we consider solution approaches from computational intelligence and introduce their algorithmic concepts.

2.1 Basic Notation and Concepts

A multi-criteria scheduling problem with m criteria is also formulated using the $\alpha|\beta|\gamma$ three-field notation [11], where the γ field contains the list of criteria. In this paper, we exclusively consider the enumeration problem of all Pareto-optima which is usually noted as $\#(\gamma_1,\ldots,\gamma_M)$, see T'kindt and Billaut [24]. Thus, we associate to this problem an *a posteriori* algorithm that does not use any aggregation methods but offers a set of solutions from which a decision maker can choose. For the matter of simplicity and if no confusions with other problem formulations is expected, we use the $\alpha|\beta|\gamma_1,\ldots,\gamma_M$ notation for the Pareto-enumeration problem throughout this paper. Further, we use the pure summation sign $(\sum)$ as abbreviation for a sum over all jobs $(\sum_{j=1}^{n})$.

2.2 Single Machine Problems

The single machine scenario is obviously the simplest of all machine environments. As such, it can serve as a good starting point for providing problems that have known optimal solutions. Such are required to serve as a benchmark for the quality of heuristic solutions, both in terms of convergence to and diversity of the Pareto-front. Still, among the plethora of multi-criteria scheduling problems only very few are known to be solvable within polynomial time complexity. For a general discussion on complexity of multi-criteria single machine scheduling problems, see Chen and Bulfin [2].

An example of such a polynomially solvable problem is $1|d_j|\sum C_j, L_{\max}$, which exposes Pareto-optimal solutions regarding total completion time of all jobs and maximum lateness. We further consider this problem due to two different reasons: On the one hand, the simple dispatching used for either criterion allows for the easy determination of extremal solutions in the multi-criteria search space. On the

other hand, in 1980 van Wassenhoven and Gelder [26] proposed an efficient algorithm for this problem, which is ideally suited for the matter of comparison. This algorithm bases on the aforementioned feature that both extremal points can be determined separately: One can be computed by the well known SPT/EDD rule, i.e. minimizing $\sum C_j$ by a *shortest processing time first* (SPT) ordering [21] with ties broken according to a downstream *earliest due dates first* (EDD) ordering [15]. The resulting maximum lateness of this schedule is denoted by $L_{\max}(\text{SPT/EDD})$ and can be computed easily. The other extremal solution is determined using EDD as dominant approach, which yields $L_{\max}(\text{EDD})$ and is obviously smaller or equal to $L_{\max}(\text{SPT/EDD})$ for the resulting schedules.

The algorithm now takes advantage of this by starting with an $L_{\max}(\text{EDD})$ schedule that generates the first Pareto-optimal point minimizing $\sum C_j$ as secondary criterion. From there, it determines the minimum increment in $L_{\max}$ needed to achieve a further reduction in $\sum C_j$. Then, the original and optimal $L_{max}(\text{EDD})$ condition is relaxed by the determined value and the subroutine that further minimizes the schedule for $\sum C_j$ is called. This procedure is then repeated until the algorithm finally reaches $L_{\max}(\text{SPT/EDD})$ which marks the alternative extremal point in the front. This way, it generates all Pareto-optimal trade off solutions in between. For a detailed description along with examples refer to T'kindt and Billaut [24]. It has been shown that the maximum number of Pareto-optimal solutions is $n(n-1)/2$ which results in a complexity of $\mathcal{O}(n^2)$. As the generation of one Pareto-optimal schedule can be performed within $\mathcal{O}(n \log n)$ the overall complexity is determined by $\mathcal{O}(n^3 \log n)$.

2.3 Parallel Identical Machine Problems

Among the multi-criteria problems on parallel machines, we first consider $P_m||C_{max}, \sum C_j$, often abbreviated as MAXAND-SUM, see Dutot et al. [7], where n independent jobs have to be scheduled on m identical machines while the Makespan ($C_{max} = \max_{j=1\ldots n}\{C_j\}$) and the total completion time $\sum C_j$ have to be minimized simultaneously. Although $P_m||\sum C_j$ is easy, the problem $P_2||C_{max}$ is already NP-hard in the ordinary sense as it is equivalent to the PARTITION problem. According to Garey and Johnson [9], $P_m||C_{max}$ is strongly NP-hard for $m \geq 3$. Consequently, the multi-criteria problem consisting of NP-hard sub-problems is also NP-hard: Applying the general proof concept from Chen and Bulfin [2] for the single machine case, we can certainly argue that an polynomial algorithm for a multi-criteria problem can also solve each sub-problem efficiently. As long as we cannot find an efficient approach for all NP-hard problems, there will be no algorithm for the multi-criteria problem that comprises some of those sub-problems.

However, Stein and Wein [22] propose a general algorithmic framework that allows the approximation of a single schedule that minimizes two criteria at the same time. Assuming approximated schedules for both sub-problems, truncation and composition steps are executed on the respective schedules. It is supposed that the combined new schedule is still satisfactory for both criteria. In this way, it is possible to find a solution closest to the Utopia point[1] which could be interpreted as perfect compromise [7].

At least to the authors' knowledge there exists no algorithmic approach to the three-criteria parallel machine problem $P_m|d_j|C_{max}, \sum C_j, \sum U_j$. Nevertheless, the problem is in that sense interesting as it really combines three fundamentally conflicting criteria: while C_{max} strongly focuses on utilization and favors the load-balancing among the parallel machines, $\sum C_j$ favors a higher throughput for single jobs. Additionally, $\sum U_j$ as due date related criteria brings a new focus into the problem which is fundamentally different to the processing time related criteria. Thus, it can be assumed that this problem is rather challenging and that the expected Pareto-front may have a large diversity.

Following Stein and Wein's framework idea of truncation and composition of schedules for sub-problems, we may be able to combine knowledge and solutions from sub-problems to clever compositions for the multi-criteria case. For all three considered sub-problems we have approximation algorithms or even optimal solution strategies ready to hand: $P_m||\sum C_j$ can be solved optimally using SPT while *longest processing time first* (LPT) guarantees a approximation quality of at least $\frac{4}{3} - \frac{1}{3m}$. The first rule can be proved with a simple interchange argument while the second can be estimated via the smallest possible counter example, see Pinedo [19]. Finally, Süer et al. [23] proposed SBC3 as an approach for $P_m|d_j|\sum U_j$ which is based on Moore's [18] optimal single machine algorithm $1|d_j|\sum U_j$.

The current selection of heuristic approaches points again to conflicts that arise when sub-problems must be combined and solved in the multi-criteria domain. The construction of a common heuristic from SPT and LPT is obviously not easy at all. In the remainder of this paper we will guide towards a way that seems to be a promising approach.

2.4 Heuristic Approaches from Computational Intelligence

A general scheme for optimization problems with multiple criteria is provided from the area of computational intelligence. There, various heuristics have been proposed which apply evolutionary and natural principles to multi-criteria optimization problems, see Deb or Coello et al. [4,3] for

[1] The Utopia point represents a usually infeasible solution obtained by minimizing all criteria separately, see Vincent and Grantham [25]. It may serve as reference point for measuring the quality of potential compromise solutions.

a detailed review. The basic idea of most approaches roots from a simplified view on Darwinian evolution theory, where a population of individuals is exposed to (environmental) selection pressure. Under this selection only the best adapted individuals survive and are thus able to reproduce. During reproduction variation and recombination of parental genes lead to slight deviations of the gene structure and consequently to slightly different solutions. If offspring exceed the parental qualities, new individuals will survive the next selection step as better adapted solutions [20] and become able to reproduce.

Most successful algorithmic approaches use the Pareto-dominance relation to select efficient solutions during evolution: They apply non-dominated ranking and sorting to determine the reproduction candidates. As in multi-criteria optimization the Pareto-front approximation is the primary goal, algorithms must have the simultaneous ability of diversity preservation and good convergence to the optimum. Advanced algorithms like SPEA2 [28] and PAES [16] utilize an archive as a central component for solution conservation and offspring generation. Both rely on crowding-based measures [5] as a global component for diversity preservation: the former applies this mechanism on the population itself, while the latter uses it for archive reorganization. In contrast, indicator-based methods aggregate solution quality in a single value to enable comparison. Simple approaches perform this aggregation via weighting; however, choosing adequate weights is an intricate process and requires adequate problem knowledge and at least a basic idea of the solution characteristics. This often leads to non-satisfactory results. More sophisticated approaches use elaborate aggregation methods, for example the $\mathscr{S}$-Metric Selection Algorithm (SMS-EMOA) to evaluate the whole Pareto-front with respect to a single value. Based on the hypervolume measure of Zitzler [27], the algorithm selects solutions which contribute most to the overall hypervolume. Although this approach outperforms the dominance-based approaches for more than four criteria, the indicator calculation is rather expensive [8].

Among all these algorithm the Non-dominated Sorting Genetic Algorithm (NSGA-2) of Deb [5] can be regarded as the most commonly used multi-criteria evolutionary algorithm and is thus also used as reference algorithm in this work. Following the traditional process of evaluation, selection, and variation the main modifications affect the selection process itself. There, the whole population is assessed regarding Pareto-dominance first: All non-dominated individuals of the entire population are assigned to a category classified at rank 1 and removed from the population. From the remaining individuals the non-dominated ones are classified in another category with rank 2. This procedure is performed until all individuals are categorized. Due to a rank proportional fitness assignment, it is guaranteed that individuals in a category of rank 1 are breeding more offspring than the rest of the population. Additionally, NSGA-2 also takes the density of solutions surrounding a particular solution into account, because the average distance of two neighboring solutions for each criterion is determined. This is called *crowding distance* and used as an additional comparison operator during the selection.

It it obvious from the description above that selection constitutes the primary mechanism in NSGA-2 and variations like mutation and recombination play only a minor role. This is especially true for combinatorial problems where variation reduces to a perturbation mechanism that is supposed to generate as diverse solutions as possible in order to cover most parts of the search space. In such a selection-focused algorithm it is almost impossible to support the evolutionary search by special problem knowledge. In the context of multi-criteria evolutionary algorithms, this can only be expressed by special tailored variation operators. In order to come up with those weaknesses, we propose a lightweight algorithmic framework that allows to plug in arbitrary operators in order to favor mainly variation influence over selection.

3 The Agent-based Approach for Multi-criteria Optimization

In contrast to the monolithic structure of the previously discussed approaches, Schwefel et al. [17] proposed an agent-based approach motivated from natural interaction of predators and prey to specifically solve multi-criteria optimization problems. We adapt this algorithmic idea and transfer its application with several modifications to scheduling problems. After describing the basic principles of our algorithmic scheme, we specify all components that are needed to address scheduling problems and integrate problem knowledge via variation operators.

3.1 The Basic Algorithm Structure

The natural principle of predators and prey interaction is considered by Schwefel et al. [17] as abstract model for solving multi-criteria optimization problems. In that scheme, predators represent environmental influences, which follow a single criterion each. Inside the algorithm, this component is represented by an agent that brings its influence to a spatially distributed population of solutions for the complete multi-criteria problem. The solution members are called prey individuals and reside in a graph structure. While the agents are allowed to randomly move about the population, preys are—quite different from nature—immobile.

More detailed, the interaction environment of the model is usually represented by a toroidal grid to ensure a virtu-

ally unrestricted motion of agents. According to Schwefel et al. [17], this guarantees that each prey is visited equally often by a predator on the long run. The grid is populated by both species, predator and prey. The latter represent possible solutions of a multi-criteria optimization problem and are immobile (each individual inhabits a single, fixed vertex). As such, there exist as many prey individuals as graph nodes, representing the population. Furthermore, all prey are of equal kind and therefore can be classified as single species. On the contrary, predator individuals represent a single criterion of the multi-criteria optimization problem and randomly roam throughout the graph structure, see Figure 1(a). After each movement step, a predator spans a selection neighborhood containing surrounding prey and evaluates those individuals regarding its respective criterion, see Figure 1(b). The worst prey is marked as candidate for replacement. Then, the remaining prey in the selection neighborhood are considered to reproduce a substitute prey for the worst one. Although the reproduction mechanism is not restricted in the general model, we will apply mutation only. That is, the best local individual is randomly varied by a specific mutation rule. Finally, the worst prey is replaced by the generated offspring, if its objective value is exceeded by the offspring's, see Figure 1(c). As such, the replacement—in our specific implementation—will only accept superior solutions in a strictly elitist way. While predators roam throughout the population, their interaction with prey is locally restricted to the selection and reproduction neighborhood. As such, it can be continuously repeated for every predator in parallel.

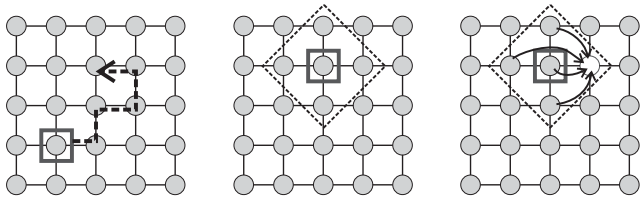


Fig. 1: Schematic depiction of the basic predator-prey algorithmic procedure. The instance exemplarily shows a cutout of the toroidal population structure that hosts prey individuals along with a single predator. The displayed movement involves two distinct walks (from positions $2,3$ to $4,2$ and from positions $4,2$ to $5,3$), each obeying the graph's vertices. Additionally, a neighborhood with a radius of 1 is shown, as well as one free vertex from which the prey has been removed. During the last step, the empty place is refilled with a new individual which is a variation of the neighboring prey.

Further, we base our approach on the generalized predator-prey model defined by Grimme et al. [13] where predator configurations comprise not only a specific criterion but also a set of variation operators for reproduction. Further, it allows to define a neighborhood function for determining the individuals that are exposed to selection or reproduction, and a walking function for the movement pattern on the spatial structure, see also Figure 1. This modular approach allows us to attach various influences in a building block fashion to predators. In this work, we focus on the variation operators as they are the critical building block for expert knowledge integration.

In the following, we instantiate the model by discussing the problem encoding, introducing the applied operators, and detailing the remaining predator building blocks.

3.1.1 Defining Structural Building Blocks and Parameters

Initially, we need to instantiate four fundamental building blocks in order to define the algorithms runtime environment.

- The *spatial population structure* is represented by a two-dimensional toroidal grid with a size of 10×10 nodes initialized with 100 random individuals.
- The *movement* of a predator follows a random walk pattern, ensuring that each position is visited equally often.
- The *number of evaluations* for each model run is restricted to 6,000.
- The *selection and reproduction neighborhood* of a predator is fixed to a radius of 1, resulting in a selection set of five prey individuals.

With this fixed setting at hand we can concentrate on the adjustments for scheduling specific problems into the general algorithmic scheme.

3.1.2 Encoding of Scheduling Problems

Since we solve scheduling problems with n jobs, we use a standard permutation encoding of length n to represent the genotype of the problem. As we consider only offline scheduling problems, an algorithm has to find the set of optimal job permutations. The schedule construction from our chosen permutation representation is quite simple for single machine and identical parallel machine environments: the jobs are dispatched in permutation order to the machines. Subsequently, each criterion value can be calculated based on the resulting machine occupations.

3.1.3 Structure of the Applied Variation Operators

Initial investigations by Grimme and Lepping [12] that special variation operators triggered by autonomously acting predators yield good approximation results especially for

multi-criteria problems. For well examined scheduling subproblems, different variation operators can be designed based on local search heuristics that assess a certain aspect of the considered multi-criteria problem. Thus, we aim to achieve an accelerated convergence towards the dedicated fractions of the overall Pareto-front.

The methodology of constructing predators from simple building blocks is founded on analysis of variation operator influences on the population. In former work [13], understanding of building block behavior and the subsequent adroit combination realized by autonomous acting predators was beneficial for improving approximation results. In the context of a practical problem, the variation operators can be regarded as local search heuristics that asses a part of the considered multi-criteria problem in order to accelerate convergence to a fraction of the Pareto-front. They collectively and simultaneously effect the population due to their tight coupling to the parallel acting predators.

We adapt this methodology by integrating problem specific expert knowledge into the operator design: The variation operators used for the reproduction of prey apply those strategies for single-criterion scheduling which are described in Sections 2.2 and 2.3. The following paragraphs describe these operators in detail.

3.2 Integration of Expert Knowledge on Scheduling Problems

For a large set of single-criterion scheduling problems, expertise on solution strategies can be expressed by optimal sorting rules. Some of them have already been briefly discussed in Sections 2.2 and 2.3. For another set of single-criterion scheduling problems there exist either polynomial time algorithms or basic sorting rules that guarantee a certain solution quality which is expressed through an approximation factor. In the latter case, these orderings can serve as educated guess for a good solution to the problem and as a good starting point to bound the set of multi-criteria solutions.

3.2.1 Ordering as Reasonable Approximation for Various Single-criterion Problems

Consider the ascending order according to the known execution times of all jobs: The SPT ordering optimally solves the problems $1||\sum C_j$ and $P_m||\sum C_j$ both on a single and on parallel identical machines, see Pinedo [19]. Similarly, the single machine maximum lateness problem $1|d_j|L_{max}$ is the best known special case of the more general cost function problem $1|prec|h_{max}$. Here, the cost function is defined as $h_j(C_j) = C_j - d_j$ and the general backward algorithm, see also Pinedo [19], yields a schedule that orders jobs in increasing order of their due dates for $1|d_j|L_{max}$, i.e., *Earliest Due Date first* (EDD).

Another due date related criterion which is also considered in this paper is $\sum U_j$, the number of late jobs. In the single machine case, the problem $1|d_j|\sum U_j$ can be solved easily by applying Moore's algorithm [18] that separates all jobs into a set of on-time and a set of late jobs. The first set has to be scheduled according to EDD to guarantee that L_{max} remains less or equal zero while the second set can be scheduled arbitrarily. The more detailed review of this approach by van den Akker and Hoogeven [1] reveals that in fact Moore's algorithm is a dynamic algorithm with a special structure and that the EDD ordering is in any case crucial in the design of the algorithm.

For the parallel machine setting $P_m|d_j|\sum U_j$, Süer et al. [23] developed the so called SBC3 heuristic that intuitively extends Moore's approach to the parallel machine environment. Also in their approach, jobs are sorted according to EDD first and then iteratively tested whether they become late when assigned to the machine with minimum load. When a late job occurs, the machine with minimum completion time is determined and the job is assigned to that machine. At the same time, the job with maximum processing time on that machine is removed from the schedule and marked late. This behavior is much more complex than other simple sorting rules, but EDD still serves as basic ordering. In Section 4.2.2 we therefore give a detailed analysis of this single criterion subproblem and explain how the solution strategy is modeled in the PPM as specific mutation operator combination.

Finally, we mention that ordering jobs in increasing order of their processing time yields a reasonable schedule for $P_m||C_{max}$. The *Longest Processing Time first* (LPT) rule leads to the well known upper bound of $\frac{4}{3} - \frac{1}{3m}$ as an indication of how well the heuristic is guaranteed to perform, see Graham [10].

3.2.2 Concept of a Problem Specific Mutation Operator

Based on the discussion above, a general variation operator is designed which allows the integration of order-based expert knowledge. With this operator, the impact of SPT, LPT, EDD, or any other sorting schemes can be brought randomly and well-dosed into the population. Figure 2 exemplary depicts the application of this operator to a given sequence with due dates d_j or processing times p_j respectively: a position is selected randomly in the permutation representation of the genotype. Then, a subsequence of $2\delta + 1$ genes is sorted according to EDD, SPT, LPT, or any other rule. The size of this δ-neighborhood is determined by a normal distribution with an externally adjustable step size of σ. Obviously, $\delta = 0$ has no effect as only the initial gene at the

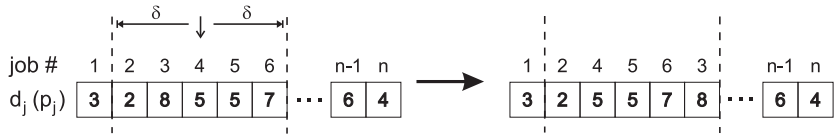


Fig. 2: Schematic depiction of the general working principle of a mutation operator with $\delta = 2$.

current position is selected. On the other hand, a larger δ leads to a higher probability of a completely ordered genome which limits the Pareto-front regarding the respective criterion.[2]

3.2.3 Coupling of Predators and Variation Operators

Due to the modular character of the considered agent-based optimization framework, predator agents and variation operators can be coupled in various ways. Two main classes of coupling can be identified: either a predator is coupled with its corresponding knowledge-based variation operator or coupled with an operator that supports another criterion's direction. For the latter one can consider a predator which selects regarding $\sum C_j$ either applying an SPT mutation or an LPT mutation respectively. For both cases of coupling, however, we expect a specific behavior which can be beneficially integrated into the final algorithm setting.

In the first case, an operator that supports the predators criterion can be expected to favor convergence towards extremal solutions. That is, solutions rapidly converge to the criterion's optimal solution and fully neglect other criteria. A similar behavior was also observed by Grimme et al. [13] and beneficially used to reach the outer perimeter of the desired Pareto-front.

In the second case, the selection/mutation coupling favors an implicit *lexicographical ordering* which conserves good solutions with respect to the predator's criterion. Additionally, this configuration favors a subsequent optimization with respect to the other criteria because the sorting is different from the actual selection. This approach can provide means to maintain good solutions while simultaneously exploring the search space regarding another criterion. For the evaluation, we consequently configure the PPM with all possible mutation/selection combinations, see e.g. Table 1, in order to take advantage of both described effects.

A fine grained analysis of these effects and a proof of concept will be given during the following evaluation.

4 Evaluation

We consider two synthetically generated job sets $\mathscr{J}_1$ and $\mathscr{J}_2$ with $n = 50$ jobs each. The processing times of $\mathscr{J}_1$ were sampled using a uniform distribution as $p_j = \lfloor \mathscr{U}(1,10) \rfloor$, $\forall j = 1 \ldots n$. In order to guarantee that all due dates can be met, we determine correspondingly $d_j = p_j + \lfloor \mathscr{U}(1,990) \rfloor$, $\forall j = 1 \ldots n$. This ensures that many Pareto-optimal solutions exist as the widely distributed due dates allow for a larger variety of L_{max} values. That property was verified by the application of the polynomial algorithm, see Section 2.2, which results in 34 Pareto-optimal solutions that form a well distributed front, see Table 7.

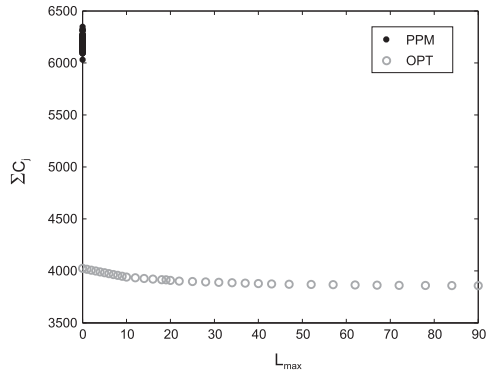
Respectively, we sampled $\mathscr{J}_2$ as $p_j = \lfloor \mathscr{U}(1,50) \rfloor$, $\forall j = 1 \ldots n$ and $d_j = p_j + \lfloor \mathscr{U}(1,100) \rfloor$, $\forall j = 1 \ldots n$. For the evaluation we apply $\mathscr{J}_1$ to single machine scheduling problems and $\mathscr{J}_2$ to the remaining parallel machine problem instances. For the parallel setup, we consider 8 identical machines, i. e. in all P_m problems we set $m = 8$. The accompanying tables in Appendix A contain the used instances and the Pareto-optimal solutions.

4.1 Evaluation of the Bi-criteria Single Machine Problem

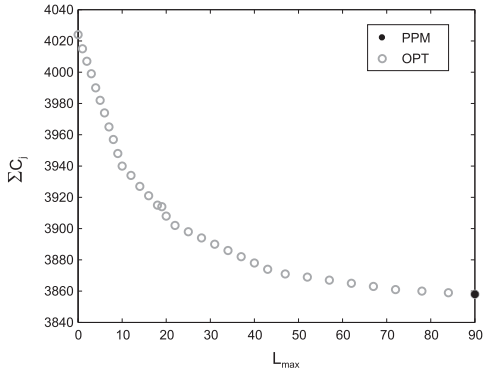
For the problem $1|d_j|\sum C_j, L_{max}$, we first investigate the influence of pure EDD and SPT mutation. Thus, we focus on the exclusive influence of EDD mutation and the corresponding single-criterion selection. As mutation width, see Section 3.2.2, we set $\sigma = 5$. The outcome is shown in Figure 3, separated by criteria.

As expected from the theoretical analysis, the EDD mutation solves the problem for the first criteria (L_{max}) optimally. Therefore, when the predator selects according to L_{max}, many solutions are found that reach the minimum possible criterion value of 0, see Figure 3(a). As no subsequent SPT optimization (in a lexicographic fashion) is performed, solutions have only comparably bad $\sum C_j$ values. The situation is different for pure SPT mutation, where we are able to find one optimal solution on the bottom right edge of the front ($\sum C_j = 3858$). The effect of SPT mutation seems to be stronger than the one of EDD mutation in terms of convergence, see Figure 3(b) and it is also more robust with respect to the selection criterion. The conclusion that can be drawn from this is ambivalent: While SPT mutation is able to favor convergence to the actual front, the population collapses into one optimal solution when the influence of the operator becomes too strong for the total completion time criterion. In order to realize an implicit lexicographic ordering of the criteria—which is especially advantageous to

[2] Since it represents the optimal solution for $L_{\max}$ and $\sum C_j$ respectively, they are obviously extremal solution on the Pareto-front.



(a) Selection regarding L_{max}



(b) Selection regarding $\sum C_j$

Fig. 3: Evaluation of the EDD mutation operator with corresponding selection. Additionally, the real Pareto-front is depicted as reference.

successfully solve this specific multi-criteria problem—we additionally apply both mutations together with the respective contrary criterion, see Section 3.2.3. The effects that could be perceived during the isolated application of each of the operators indicate that their combination could lead to a good approximation of the problem's Pareto-front. The final parametrization of each predator is shown in Table 1.

Predator	Criterion	Mutation	Parameter
P1	L_{max}	EDD	$\sigma = 4$
P2	$\sum C_j$	EDD	$\sigma = 4$
P3	L_{max}	SPT	$\sigma = 4$
P4	$\sum C_j$	SPT	$\sigma = 4$

Table 1 Parameterization of the predators for the combined problem solving scenario.

The evaluation of the combined run results in the Pareto-front approximation depicted in Figure 4. It turns out that it is possible to combine the beneficial effects to achieve a well overall approximation. It is remarkable that all identified characteristics are preserved in their combined application, resulting in a front that is almost covered as a whole.

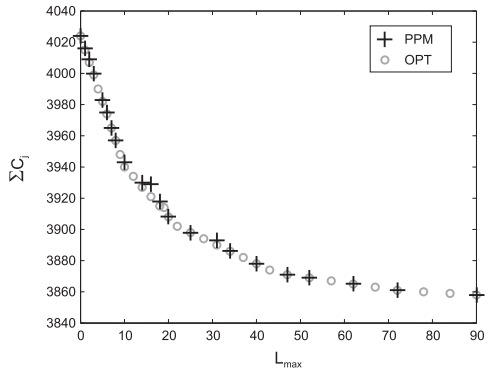


Fig. 4: Parallel application of all mutation operators in combination with the two different selections and the real Pareto-front.

4.2 Evaluation of Multi-criteria Parallel Machine Problems

In order to evaluate our algorithmic framework on more challenging problems, we apply the agent-based approach to several parallel machine problems with two and three criteria. Additionally, we clarify how $P_m|d_j|\sum U_j$ can be approximated by the interplay of several agent. The above mentioned SBC3 behavior can be modeled by the interaction of atomic sorting rules. This setup is then applied to approximate one criterion in a multi-criteria problem environment.

4.2.1 Solving the $P_m||C_{max}, \sum C_j$ Problem

As explained in Section 2.3, there exists no specific solution algorithm to the bi-criteria problem $P_m||C_{max}, \sum C_j$. We are therefore forced to apply the NSGA-2 heuristic in order to generate reference results. Here, we are faced with the general problem that NSGA-2 must be fine-tuned in order to achieve the best results for comparison. To this end, we performed several simulation studies and used the recommended NSGA-2 standard configuration, see Deb [4], as well as the same special variation operators applied in the PPM. However, as we expected, special operators applied in NSGA-2 did not lead to any better solutions than standard operators. Even all attempts with recombination schemes did not yield better results. Thus, we consider for comparison the best solutions achieved by NSGA-2 using a population size of 100 with exclusive swap mutation. The mutation randomly swaps 8 jobs in the sequence and is applied to each individual (variation probability of 1.0).

The PPM, however, uses two special tailored operators derived from the general mutation operator scheme, see Section 3.2.2: SPT mutation derived from the SPT rule, which solves $P_m||\sum C_j$ optimally as well as LPT mutation that approximates $P_m||C_{max}$ and takes its cue from the earlier discussed LPT rule. Consequently, we created a predator species for each criterion and connected them to both operators re-

spectively. We discovered that LPT mutation should be emphasized over the quite efficient SPT mutation and choose therefore $\sigma = 10$ while SPT is applied with $\sigma = 5$. Overall we run the PPM for 6,000 function evaluations per experiment and performed 50 runs of each setup; the figures show a one solution out of all runs. The whole configuration of these experiments is given in Table 2.

Predator	Criterion	Mutation	Parameter
P1	C_{max}	SPT	$\sigma = 5$
P2	$\sum C_j$	SPT	$\sigma = 5$
P3	C_{max}	LPT	$\sigma = 10$
P4	$\sum C_j$	LPT	$\sigma = 10$

Table 2 Parameterization of the predators for the combined problem solving scenario.

Both obtained Paretofronts are depicted in Figure 2 and single-criteria SPT as well as LPT solutions are additionally shown in the criteria space. Obviously, PPM outperforms NSGA-2 in both convergence and diversity although, on the first sight, one can get the impression that PPM generates a less divers front than NSGA-2. However, note that the gray shaded "overall area of non-domination" is larger for PPM. As PPM converges even to $\sum C_j(OPT)$ value, only the minimum C_{max} value is at that point considered for the Paretofront.

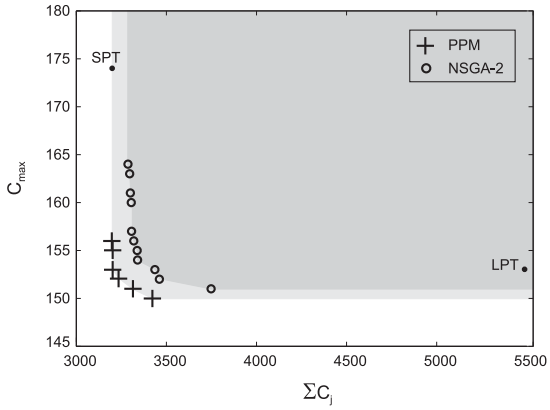


Fig. 5: Results for $m = 8$. Circles: NSGA-2; crosses: PPM

Before we present the PPM application to a three-criteria problem, we explain how the single-criterion total number of tardy jobs problem is approximated.

4.2.2 A Preliminary Study on $P_m|d_j|\sum U_j$

While the approximation of the total completion time $\sum C_j$ and Makespan C_{max} is comparatively easy, the total number of late jobs $\sum U_j$ is more challenging, as no atomic sorting heuristic is immediately applicable. As mentioned in Section 3.2.1, the SBC3 heuristic of Süer et al. transfers Moore's algorithm to the parallel machine environment. In general, we assume that EDD serves as basic principal while more detailed insights are actually required. Thus, we performed several experiments with different combinations of random swap mutation, EDD-mutation, and SPT-mutation. In Figure 6, we depict the best results for an instance of $P_8|d_j|\sum U_j$ and several operator combinations. The *Random Swap Mutation* (RD) uniformly swaps two genes in the individual encoding and serves as adaptation of the standard mutation to a permutation encoding. The performance with pure RD-mutation is relatively poor as no problem specific knowledge is provided and the algorithm has to rely on simple random search.

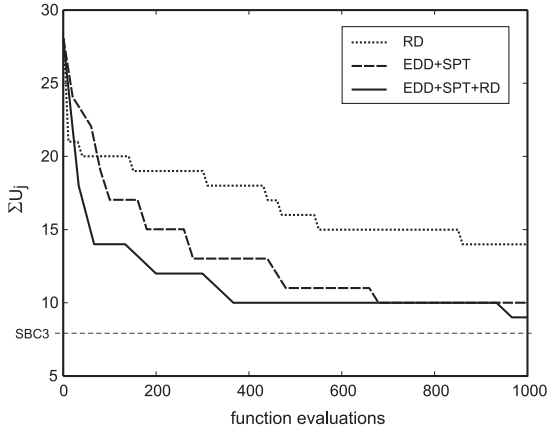


Fig. 6: Exemplary best results for $P_8|d_j|\sum U_j$ and several operator combinations and the result of the SBC3 heuristic ($\sum U_j = 8$). First, only a random mutation operator is applied to the population (RD). Second, EDD and SPT inspired operators are applied (EDD+SPT). Finally, all operators are combined (EDD+SPT+RD).

Please remember the procedure of the SBC3 heuristics: Whenever a job misses its due date, the largest currently scheduled job is shifted towards the end of the sequence and marked late. Thus, the SBC3 behavior can be modeled by a combination of multiple atomic sorting rules, e.g. EDD and SPT while EDD is the general sorting rule and SPT heuristically produced subsequences where larger jobs are scheduled more to the end. Thus, we apply a combination of EDD + SPT and obtain much faster and better convergence than with exclusive RD-mutation. Further, we discover that the performance of the PPM can be even more improved if some random influence is added. In this way, the PPM yields almost as good results as the SBC3 heuristic and we assume that the EDD + SPT combination is reasonable to approximate the $\sum U_j$ part of a corresponding multi-criteria problem.

4.2.3 Solving the $P_m|d_j|C_{max}, \sum C_j, \sum U_j$ Problem

With these findings at hand, we are now able to tackle the multi-criteria problem $P_m|d_j|C_{max}, \sum C_j, \sum U_j$ with three contradictory criteria. We apply the same configuration concept as in the previous test but use—as described above—additionally EDD and SPT mutations (with $\sigma = 5$) as they are supposed to lead solutions to the $\sum U_j$ criterion.

Predator	Criterion	Mutation	Parameter
P1	C_{max}	SPT	$\sigma = 5$
P2	$\sum C_j$	SPT	$\sigma = 5$
P3	$\sum U_j$	SPT	$\sigma = 5$
P4	C_{max}	LPT	$\sigma = 10$
P5	$\sum C_j$	LPT	$\sigma = 10$
P6	$\sum U_j$	LPT	$\sigma = 10$
P7	C_{max}	EDD	$\sigma = 5$
P8	$\sum C_j$	EDD	$\sigma = 5$
P9	$\sum U_j$	EDD	$\sigma = 5$

Table 3 Parameterization of the predators for the combined problem solving scenario.

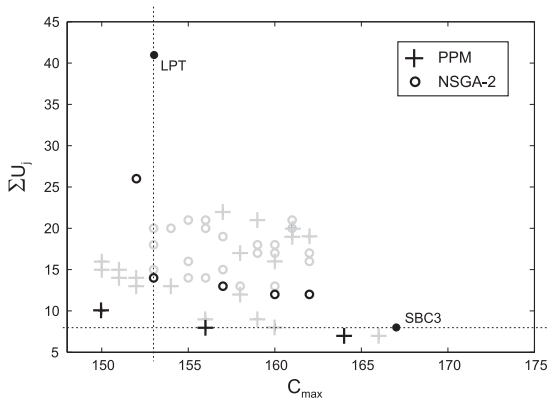
In contrast to the test with two criteria described above, problems (and corresponding solutions) with three or more criteria are hard to visualize. Thus, we follow an alternative quality assessment and compute the $\mathscr{S}$-Metric values for the obtained Paretofronts. This metric evaluates a set of non-dominated solutions in the criteria space by calculating the hypervolume which is dominated by the solution set. The dominated hypervolume corresponds to the size of the region of the criteria space—bounded by a reference point—which contains solutions that are weakly dominated by at least one of the members of the set; consequently, a larger $\mathscr{S}$-Metric value indicates a better Pareto-front approximation. In Table 4 we show the results for the three-criteria problem.

Algorithm	$\mathscr{S}$-Metric		
	Mean / Median	Std. Dev	Variance
NSGA-2	0.16496 / 0.16499	$4.3178 \cdot 10^{-3}$	$1.8643 \cdot 10^{-5}$
PPM	0.19220 / 0.19220	$3.8321 \cdot 10^{-5}$	$1.4685 \cdot 10^{-9}$

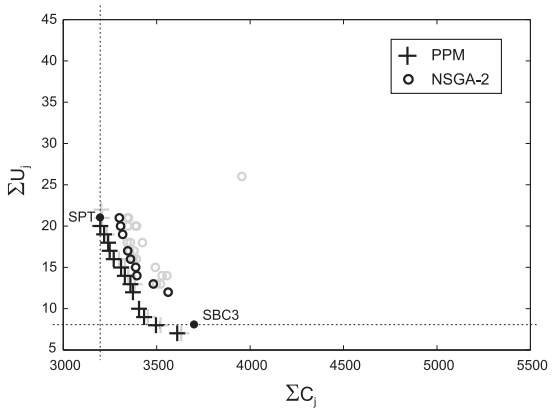
Table 4 Normalized $\mathscr{S}$-Metric value for both approaches averaged over 50 runs. Reference point: $[4000, 4000, 4000]$.

The PPM shows not only a better approximation of the front but also a grater reliability as both standard deviation and variance are significantly smaller than for the NSGA-2 results. Additionally, we show in Figure 7 the different projections of the Pareto-front approximation and corresponding heuristic results.

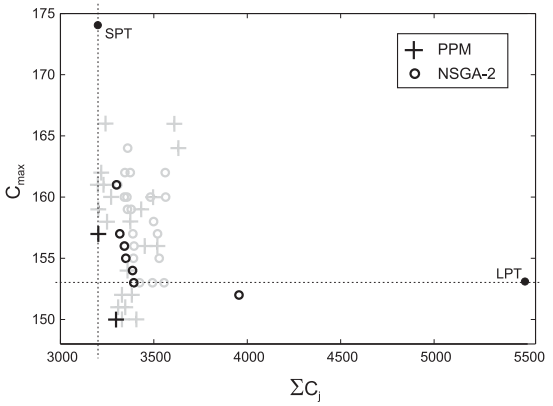
In these projection it becomes obvious that PPM outperforms NSGA-2 in terms of convergence and diversity.



(a) Projection regarding $C_{max}, \sum U_j$



(b) Projection regarding $\sum C_j, \sum U_j$



(c) Projection regarding $\sum C_j, C_{max}$

Fig. 7: Results for $m = 8$ with 3 Criteria. Circles: NSGA-2; crosses: PPM

Figures 7(b) and 7(c) prove that even for this problem the optimal SPT solution is found by PPM while NSGA-2 fails. Further, PPM can find even better solutions than those of SBC3 and LPT heuristic, see Figure 7(a). Considering both $\mathscr{S}$-Metric and the figures, we draw the conclusion that PPM is a powerful concept for solving scheduling problems with even more than two criteria.

5 Conclusion

In this work, we presented an agent-based framework for solving multi-criteria scheduling problems. The lightweight design of the approach features the flexible combination of single-criterion solution heuristics to tackle complex multi-criteria problems. It provides a decoupled and easy-to-realize randomized strategy which allows to seamlessly integrate theoretically founded results and rules from single objective scheduling. In this way it becomes possible to use problem specific expert knowledge to cope with hard-to-solve problems. More specific, we are able to express problem specific single-criterion knowledge by corresponding variation operators. The isolated analysis of those operators shows that it is possible to reliably cover certain regions of the Pareto-front. In this way, it is even possible to integrate both extremal convergence behavior and lexicographic optimization. The simultaneous and parallel realization of those discovered effects on the population yields a good set of trade-off solutions.

In this work we exemplary studied the behavior of our agent-based framework on three test problems. For the easy single machine problem $1|d_j|\sum C_j, L_{max}$ the algorithm proves to be principally able to reach the optimal Pareto-front. Here, the successful combination of well known solution strategies like SPT and EDD via specially designed operators is demonstrated. Further, the approach also proves to be able to handle hard problems with multiple criteria. For both scheduling problems $P_m||C_{max}, \sum C_j$ and $P_m|d_j|C_{max}, \sum C_j, \sum U_j$ the combination of SPT, LPT, and EDD is shown to be beneficial in the randomized scheme, outperforming modern multi-criteria evolutionary search algorithms like NSGA-2.

Acknowledgment

The authors would like to thank M. Bender, J. Blazewicz, E. Pesch, D. Trystram, and G. Zhang for the organization of the 2010 "New Challenges in Scheduling Theory" workshop in Frejus, France. Further, the authors thank J. J. Durillo, A. J. Nebro, and E. Alba for providing their jMetal framework [6] and A. Seshadri for his NSGA-2 implementation in MATLAB.

References

1. van den Akker, M., Hoogeveen, H.: Minimizing the number of late jobs in a stochastic setting using a chance constraint. Journal of Scheduling **11**(1), 59–69 (2008)
2. Chen, C.L., Bulfin, R.L.: Complexity of single machine multi-criteria scheduling problems. European Journal of Operational Research **70**, 115–125 (1993)
3. Coello, C.A.C., Lamont, G.B., Veldhuizen, D.A.V.: Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation), 2nd ed., 2nd edn. Springer-Verlag New York, Inc. (2007)
4. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms, 1st edn. Wiley-Interscience Series in Systems and Optimization. Wiley (2001)
5. Deb, K., Agrawal, S., Pratab, A., Meyarivan, T.: A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In: M. Schoenauer, et al. (eds.) Proceedings of the Conference on Parallel Problem Solving from Nature, *Lecture Notes in Computer Science*, vol. 1917, pp. 849–858. Springer (2000)
6. Durillo, J., Nebro, A., Alba, E.: The jMetal Framework for Multi-Objective Optimization: Design and Architecture. In: IEEE Congress on Evolutionary Computation, vol. 5467, pp. 4138–4325. Springer, Barcelona, Spain (2010)
7. Dutot, P.F., Rzadca, K., Saule, E., Trystram, D.: Introduction to Scheduling, 1st edn., chap. Multi-Objective Scheduling, pp. 219–251. CRC Press (2010)
8. Emmerich, M., Beume, N., Naujoks, B.: An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. In: Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization, pp. 62–76 (2005)
9. Garey, M.R., Johnson, D.S.: "Strong" NP-Completeness Results: Motivation, Examples, and Implications. Journal of the ACM **25**(3), 499–508 (1978)
10. Graham, R.L.: Bounds on multiprocessing timing anomalies. SIAM Journal on Applied Mathematics **17**, 416–429 (1969)
11. Graham, R.L., Lawer, E.L., Lenstra, J.K., Kan, A.H.G.R.: Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey. Annals of Discrete Mathematics **5**, 287–326 (1979)
12. Grimme, C., Lepping, J.: Designing Multi-Objective Variation Operators Using a Predator-Prey Approach. In: Proceedings of the International Conference on Evolutionary Multi-Criterion Optimization, *Lecture Notes in Computer Science*, vol. 4403, pp. 21–35. Springer (2007)
13. Grimme, C., Lepping, J., Papaspyrou, A.: Exploring the Behavior of Building Blocks for Multi-Objective Variation Operator Design using Predator-Prey Dynamics. In: D. Thierens, et al. (eds.) Proceedings of the Genetic and Evolutionary Computation Conference, pp. 805–812. ACM Press (2007)
14. Hoogeveen, H.: Multicriteria scheduling. European Journal of Operational Research **167**(3), 592–623 (2005)
15. Jackson, J.R.: Scheduling a Production Line to Minimze Maximum Tardiness. Management Science Research Project, Research Report 43, University of California, Los Angeles (1955)
16. Knowles, J., Corne, D.: Approximating the Nondominated Front Using the Pareto Archived Evolution Strategies. Evolutionary Computation **8**(2), 149–172 (2000)
17. Laumanns, M., Rudolph, G., Schwefel, H.P.: A Spatial Predator-Prey Approach to Multi-Objective Optimization: A Preliminary Study. In: T. Bäck, et al. (eds.) Proceedings of the Conference on Parallel Problem Solving from Nature, pp. 241–249. Springer (1998)
18. Moore, J.M.: An n Job, One Machine Sequencing Algorithm for Minimizing the Number of Late Jobs. Management Science **15**, 102–109 (1968)
19. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems, 3rd edn. Springer (2009)
20. Schwefel, H.P.: Evolution and Optimum Seeking, 1st edn. Wiley (1995)
21. Smith, W.E.: Various Optimizers for Single-stage Production. Naval Research Logistics Quarterly **3**, 59–66 (1956)
22. Stein, C., Wein, J.: On the existence of schedules that are near-optimal for both makespan and total weighted completion time. Operations Research Letters **21**, 115–122 (1997)

23. Süer, G.A., Báez, E., Czajkiewicz, Z.: Minimizing the number of tardy jobs in identical machine scheduling. Computers and Industrial Engineering **25**(1–4), 243–246 (1993)
24. T'kindt, V., Billaut, J.C.: Multicriteria Scheduling. Theory, Models and Algorithms, 2nd edn. Springer (2006)
25. Vincent, T.L., Grantham, W.J.: Optimality in Parametric Systems, 1st edn. Wiley, New York (1981)
26. van Wassenhove, L.N., Gelders, F.: Solving a Bicriterion Scheduling Problem. European Journal of Operational Research **2**(4), 281–290 (1980)
27. Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications. Ph.D. thesis, ETH Zürich (1999)
28. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Communication Networks Lab (TIK), ETH Zürich (2001)

j	L_{max}	$\sum C_j$	j	L_{max}	$\sum C_j$	j	L_{max}
1	0	4024	10	9	3948	19	25
2	1	4015	11	10	3940	20	28
3	2	4007	12	12	3934	21	31
4	3	3999	13	14	3927	22	34
5	4	3990	14	16	3921	23	37
6	5	3982	15	18	3915	24	40
7	6	3974	16	19	3914	25	43
8	7	3965	17	20	3908	26	47
9	8	3957	18	22	3902	27	52

Table 7 Maximum lateness (L_{max}) and total completion time ($\sum C_j$) of all 34 Pareto-optimal solutions for job set $\mathscr{J}_1$.

A Job Sets $\mathscr{J}_1$, $\mathscr{J}_2$, and Solutions for $1|d_j|\sum Cj, L_{max}$

j	p_j	d_j	j	p_j	d_j	j	p_j	d_j	j	p_j	d_j
1	10	793	14	5	170	27	3	526	40	7	100
2	1	827	15	1	158	28	7	859	41	10	645
3	1	50	16	4	522	29	2	571	42	10	776
4	1	484	17	9	533	30	8	393	43	5	190
5	3	513	18	3	829	31	2	610	44	7	10
6	6	69	19	1	167	32	2	156	45	9	530
7	3	824	20	3	928	33	3	632	46	10	362
8	1	951	21	5	292	34	1	357	47	7	487
9	7	537	22	8	391	35	2	849	48	1	800
10	1	90	23	2	151	36	3	785	49	8	466
11	1	968	24	9	302	37	6	192	50	5	749
12	6	702	25	5	279	38	2	678			
13	3	4	26	10	375	39	8	260			

Table 5 Properties of $\mathscr{J}_1$ as problem instance for $1|d_j|L_{max}, \sum Cj$.

j	p_j	d_j	j	p_j	d_j	j	p_j	d_j	j	p_j	d_j
1	6	75	14	22	57	27	35	50	40	22	96
2	12	50	15	18	100	28	6	26	41	8	22
3	50	118	16	12	70	29	26	31	42	18	106
4	11	39	17	28	49	30	38	127	43	26	108
5	33	34	18	18	57	31	34	60	44	33	39
6	5	54	19	36	69	32	36	57	45	29	70
7	44	129	20	17	67	33	24	73	46	8	84
8	47	147	21	21	112	34	5	94	47	28	32
9	45	138	22	50	127	35	43	60	48	40	100
10	19	41	23	5	8	36	8	34	49	12	48
11	23	77	24	29	53	37	12	82	50	11	97
12	42	126	25	18	88	38	10	74			
13	27	28	26	15	73	39	33	55			

Table 6 Properties of $\mathscr{J}_2$ as problem instance for $P_m|d_j|\gamma_1 \ldots \gamma_k$.