An Approach to Instantly Use Single-objective Results for Multi-objective Evolutionary Combinatorial Optimization

Christian Grimme[1] and Joachim Lepping[2]

[1] Robotics Research Institute, TU Dortmund University, 44221 Dortmund, Germany,
christian.grimme@udo.edu
[2] INRIA Rhône-Alpes, Grenoble University, 38330 Montbonnot-Saint-Martin, France
joachim.lepping@inria.fr

Abstract. Standard dominance-based multi-objective evolutionary algorithms hardly allow to integrate problem knowledge without redesigning the approach as a whole. We present a flexible alternative approach based on an abstraction from predator-prey interplay. For parallel machine scheduling problems, we find that the combination of problem knowledge principally leads to better trade-off approximations compared to standard class of algorithms, especially NSGA-2. Further, we show that the incremental integration of existing problem knowledge gradually improves the algorithm's performance.

Keywords. Predator-Prey Model, Evolutionary Multi-Objective Optimization, Multi-objective Scheduling, Knowledge Integration

1 Introduction

In multi-objective evolutionary optimization, *dominance-based methods* are currently used as quasi-standard. They extend the concept of original single-objective evolutionary algorithms to the multi-objective domain introducing mechanisms for selecting solutions regarding multiple objectives. For the NSGA-2 [2] algorithm, the particular fitness assignment is based on sorting the population into different fronts using the non-domination order relation. To form the next generation of candidate solutions, NSGA-2 combines the current population and its offspring generated by standard variation operators. Such a strong focus on selection may devalue variation operators to a subordinate influence. That means, for expertise integration advanced variation operators can unfold their full benefit only along with an alternative and more dynamic selection scheme which replaces the monolithic algorithmic architecture of dominance-based approaches. Such an alternative appears in this paper.

Our approach uses the predator-prey model (PPM) proposed by Laumanns et al. [4] which adapts the well-known predation paradigm from biology: a population of prey is arbitrarily distributed on a spatial structure which is represented by a toroidal grid. Predators pursue only *one objective* and favor only *one special variation operator* each. They randomly roam the population to chase prey which

are weak regarding their specific objective. Multiple predators are expected to force the prey likewise to adapt to the threats and thus result in suitable trade-off solutions for the complete optimization problem. In our approach the coupling of special heuristics (which realize the actual variation) to predators allows to integrate expert knowledge from single-objective problems.

2 Multi-objective Optimization and Scheduling Problems

In multi-objective optimization, a problem instance comprises multiple and (at least partly) contradicting goals that should be fulfilled simultaneously. Usually it is impossible to find a single optimal solution but only a set of good trade-offs among those goals. This solution set is called Pareto-optimal set and forms the Pareto-front in solution space.

A scheduling problem—denoted by $\alpha|\beta|\gamma$—is commonly concerned with allocating n jobs to a machine environment α with m machines such that all constraints β are met [3]. The resulting schedule should be optimal for one or more given objective(s) γ. For example, a check-in counter queue problem is denoted as $P_m|r_j, d_j|\sum U_j$. Here, P_m denotes an environment of m identical counters (machines). Passenger j arrives at time r_j and needs to catch a flight at time d_j. The objective is to minimize the total number of passengers $\sum U_j,\ U_j \in \{0, 1\}$ missing their flight ($U_j = 1$). Under multiple objectives, the γ-field contains all objectives that have to be optimized simultaneously. Regarding our example the problem $P_m|r_j, d_j|\sum C_j, \sum U_j$ states that not only the flight misses should be minimized but also all customers should be served as fast as possible. This is expressed as minimizing the sum of all completion times, while the condition $C_j \geq r_j + p_j$ holds. There, p_j is the processing time of job j.While for the $\sum U_j$ objective ascending ordering of due dates d_j is reasonable the $\sum C_j$ can be solved optimally sequencing jobs in ascending order of p_j. However, as due dates and processing times might be unrelated (except that all due dates can be always met, thus $d_j > p_j$ holds.) both objectives are fundamentally conflicting.

As almost all multi-objective scheduling problems are NP-hard, practitioners have to use general techniques or randomized heuristic approaches like multi-objective evolutionary algorithms (MOEAs), see Coello et al. [1] for a detailed overview. Today, the practitioner finds a huge amount of standard algorithms that either apply non-dominated ranking, sorting and archiving techniques (e.g., NSGA-2, SPEA2, PAES) or indicator-based selection mechanisms (e.g., SMS-EMOA, HypE) to generate a precise and diverse Pareto-front approximation. As these methods are rather general, the practitioner still faces the problem of integrating his already available expert knowledge into the algorithm. That is more complicated than expected: Due to the monolithic and integrated structure of most approaches it is usually not sufficient—in contrast to single-objective problems—to only change variation operators. He rather has to redesign many parts of the original algorithmic scheme in order to bring in expertise. We address this problem by revisiting the predator-prey idea and show that it offers the property to integrate expertise seamlessly.

3 Predator-Prey Model for Multi-objective Optimization

The nature-inspired principle of predator and prey interaction proposed by Laumanns et al. [4] considers *prey* as solutions for multi-objective problems which are placed at vertices of a two-dimensional toroidal grid representing the spatially distributed population. *Predators* move across the spatial structure according to a random walk scheme (usually a uniformly distributed movement) and chase the prey only within their current neighborhood on the torus. This "hunting" process consists of evaluating all prey in the direct neighborhood of a predator's position according to a *single objective* assigned to it. The *worst* prey within this neighborhood is "eaten" and replaced by an offspring prey, which is created out of neighboring prey using variation operators. In our realization, the replacement approach follows an elitist philosophy: the worst prey is only replaced, if the offspring is better regarding the predators objective. The process is repeated until a termination criterion is reached. As the described action is restricted to each predator and completely self-contained, multiple predators can act in parallel and bring their influence to the distributed population.

For transferring the PPM to scheduling problems we encode schedules in prey using a permutation encoding of length n which represents the sequence of jobs. To map the permutation into schedules, we use a non-delay First-Come-First-Served (FCFS) approach. The main goal of expertise integration is to foster convergence to the Pareto-front. This expertise is here provided by simple sequencing heuristics: The *shortest processing time first* (SPT) rule is known to be optimal for the total completion time objective ($1||\sum C_j$, $P_m||\sum C_j$). Further, for the number of late jobs problem on parallel machines ($P_m||\sum U_j$) the *earliest due date first* (EDD) rule is reasonable[3]. We designed a variation operator which allows to bring the effects of SPT, EDD, or any other sorting scheme randomly and well-dosed into the population. Figure 1 exemplary depicts the application of this operator to a given sequence with processing times p_j. A position is selected randomly in the permutation representation of the genotype. Then, a subsequence of $2\delta + 1$ genes is sorted according to the heuristic. Here, we show the application of SPT sorting. The size of this δ-neighborhood is determined by a always positive normal distributed value with adjustable step-size σ. A larger δ leads to a higher probability of a completely ordered genome, $\delta = 0$ leads to no reordering at all.

4 Experiments and Results

To evaluate our approach, we generated 50 synthetic job sets, $(\mathcal{J}_1^{50} \dots \mathcal{J}_{50}^{50})$ containing 50 jobs each. We sampled all sets with characteristics of processing time $p_j = \lfloor \mathcal{U}(1, 50) \rfloor,\ \forall j = 1 \dots n$ and due dates $d_j = p_j + \lfloor \mathcal{U}(1, 100) \rfloor,\ \forall j = 1 \dots n$. Release dates are generated depending on p_j and d_j according to $r_j = \mathcal{U}(0, d_j - p_j)$ in 90 % of the cases and $r_j = 0$ otherwise. As we consider a parallel machine

[3] Certainly, a better way is to apply SBC3 by Süer et al. which however incorporates aspects of EDD and SPT [5].

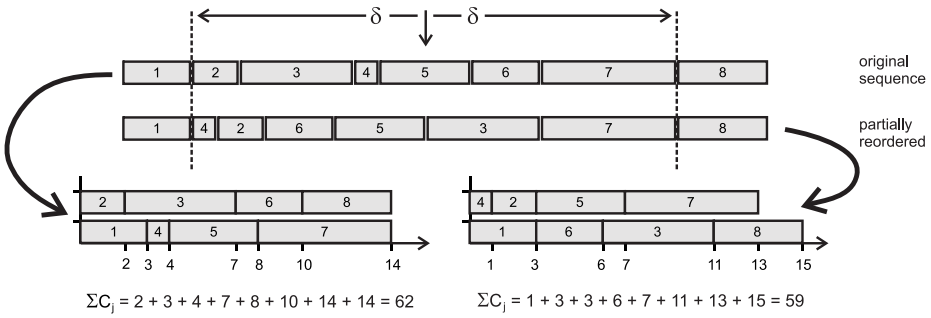


Fig. 1. Schematic depiction of the mutation operator concept with $\delta = 2$ and SPT-mutation and FCFS schedule generation.

setup, we fix the machine size for this benchmark to $m = 8$ identical machines. For statistical soundness, we simulated every instance 30 times.

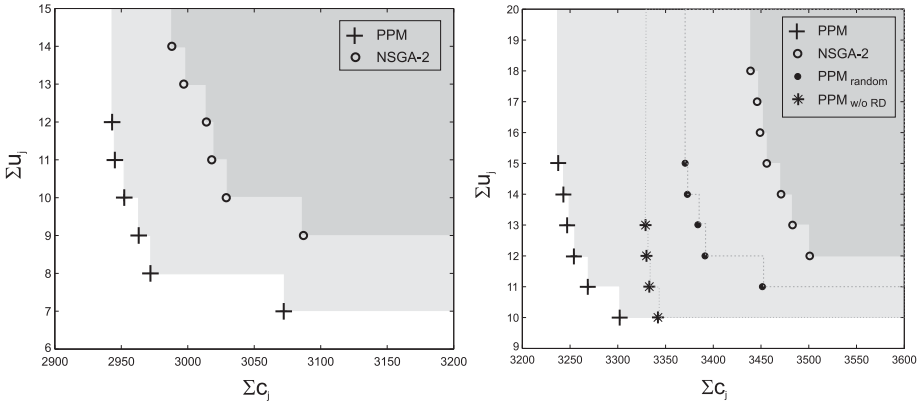
The PPM is applied a standard configuration consisting of a 10×10 toroidal grid with 100 immobile prey individuals and a uniformly distributed predator step size of 1. Overall, we allow a maximum of $12,000$ function evaluations (and fix this maximum number as termination criterion).

For comparison reasons, we also apply NSGA-2 to the considered problems to acquire some landmark results. There, we also chose a population size of 100 individuals and allow also a maximum of $12,000$ function evaluations. Based on extensive pre-experimental testing, we used NSGA-2 with a random swap mutation operator and step size setting $\delta = 8$. This mutation randomly swaps δ jobs in the sequence and is applied to each individual (variation probability of 1.0). With this setting we achieved the best NSGA-2 results. Mutation with expertise integration as well as (interestingly) recombination have shown negative effects on the solution quality and are thus excluded.

For the qualitative evaluation, we apply two well-known metrics: the hypervolume metric as well as the ε-Indicator, see Zitzler at al. [6]. For all evaluations, we use the reference point $\boldsymbol{r} = (\sum C_j, \sum U_j) = (5500, 50)$ that is beyond all maximum solution values in function space.We further apply the ε-dominance metric $I_\varepsilon(A, B)$ [6] which determines, whether a solution set A dominates another solution set B entirely. Only if $I_\varepsilon(A, B) > 0$ and the inverse comparison $I_\varepsilon(B, A) \leq 0$ holds, the set A dominates set B completely. Otherwise, the intersections of the determined solution fronts do not allow a domination statement.

First, we compare the PPM with NSGA-2 on a parallel machine problem with two criteria: $P_8||\sum C_j, \sum U_j$. We apply an SPT-based mutation which sorts a randomly selected part of the genome according to SPT ($\delta = 2$), an EDD-based mutation ($\delta = 2$) which orders according to EDD, and a Gaussian swap mutation ($\delta = 4$). These operators are each attached to two predators of which one predator selects regarding $\sum C_j$ and the other on $\sum U_j$ (resulting in overall six predators).

Evaluation results are shown in Figure 2(a). Compared to the application of NSGA-2, the expertise-guided PPM generates a better Pareto-front approx-



(a) $P_8||\sum C_j, \sum U_j$ (b) $P_8|r_j|\sum C_j, \sum U_j$

Fig. 2. Result of two examined scheduling problems.

imation. In this figure, gray shaded areas depict the hypervolumes enclosed by the generated solution fronts. Over all 50 examined instances, the comparison to NSGA-2 revealed a significant dominance of the PPM approximations (Wilcoxon rank-sum test with $p \leq 0.05$ regarding the enclosed hypervolume). Further, Table 1 summarizes the ε-Indicator point of view on the acquired solutions. There, columns show how often (in %) the results of PPM dominate the results of NSGA-2 and vice versa compared pair-wise over all instances. The operator $A \rhd_\varepsilon B$ denotes the percentage domination count of A over B with respect to ε-dominance. The first line shows that about 65 % of all PPM solutions dominate NSGA-2 solutions completely. However, NSGA-2 often finds some solutions that are missed by PPM and also not dominated by any PPM solution. Still, none of the NSGA-2 solutions dominates the PPM results.

Table 1. Results of the ε-Indicator evaluation, $\bullet$ for pure random, $*$ for only SPT and EDD and $+$ for SPT, EDD, and RD configurations.

Problem	PPM $\rhd_\varepsilon$ NSGA-2		NSGA-2 $\rhd_\varepsilon$PPM
	% (mean)	std.	%
$P_8\|\|\sum C_j, \sum U_j$	64.54	25.25	0.00
$P_8\|r_j\|\sum C_j, \sum U_j$	$96.42^\bullet$	$3,70^\bullet$	$0.01^\bullet$
	92.55^*	9.47^*	0.03^*
	98.94^+	2.16^+	0.00^+

Adding release dates to the previous problem results in $P_8|r_j|\sum C_j, \sum U_j$ which is far more difficult to optimize, even for each objective separately. SPT is not optimal anymore for $P_8|r_j|\sum C_j$ and EDD is far from being optimal for $P_8|r_j|\sum U_j$. For our experiments, we use the same settings as before but extend the set of variation operators by a release date related operator. The

RD-mutation operator orders a subsequence according to release dates ($\delta = 3$). Some exemplary results are shown in Figure 2(b). The statistically more detailed evaluation (Wilcoxon rank sum test, $p \leq 0.05$) proves that the PPM with solely mutation significantly dominates NSGA-2 on the hypervolume. The remaining two extensions also proof their benefit significantly. Integrating the expertise from our previous case study strongly improves convergence while adding RD-mutation increases the solution quality once more. Further, the ε-Indicator results from Table 1 show for all three setups a strict domination of PPM over NSGA-2 in more that 90 % of the cases. Only in the completely random setup, NSGA-2 can by chance dominate a solution in very few cases. We were able to show that expertise integration with the PPM is highly beneficial for improving solution quality.

5 Conclusion

We presented the predator-prey model that allows to effectively support the optimizer by integrating available problem specific knowledge. The expertise is expressed by variation operators which can be seamlessly used in the algorithm. Our results show that this is a great advantage over traditional dominance-based methods. In our case studies, we investigated multi-objective combinatorial scheduling problems and found that we can reliably achieve better trade-off approximations. Furthermore, the incremental integration of existing problem knowledge gradually improves the algorithms performance.

References

1. Coello Coello, C., Lamont, G.B., Veldhuizen, D.v.: Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, New York, 2 edn. (2007)
2. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2), 182–197 (2002)
3. 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)
4. Laumanns, M., Rudolph, G., Schwefel, H.P.: A spatial predator-prey approach to multi-objective optimization: A preliminary study. In: Parallel Problem Solving From Nature V, pp. 241–249. Springer, Berlin (1998)
5. 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)
6. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Grunert da Fonseca, V.: Performance Assessment of Multiobjective Optimizers: An Analysis and Review. IEEE Transactions on Evolutionary Computation 7(2), 117–132 (2003)