HYBRID EA/MIP METHOD FOR SOLVING TWO-STAGE STOCHASTIC INTEGER PROGRAMMING PROBLEMS IN CHEMICAL BATCH SCHEDULING

Michael T.M. Emmerich‡, Maren Urselmann†, Jochen Till†, Guido Sand†, and Sebastian Engell†
†Process Control Lab (BCI–AST), Universität Dortmund, 44221 Dortmund, Germany
{m.urselmann, j.till, g.sand, s.engell}@bci.uni-dortmund.de
‡LIACS, University of Leiden, 2333 CA-Leiden, The Netherlands, emmerich@liacs.nl

ABSTRACT

This paper presents the design of a problem specific evolutionary algorithm (P-EA) for a chemical batch scheduling problem under uncertainty. Chemical batch scheduling problems are concerned with the optimal allocation of processing steps to resources over time with respect to an (economic) objective. As a real-world application problem, the production of polystyrene in a multi-product batch plant with incompletely known demand is considered. Due to the uncertain demand the problem is modelled as a two-stage stochastic integer program. The first stage problem is to find an optimal schedule under uncertainties, whereas the second stage reflects later recourse actions in face of realised uncertainties.

The optimisation of the first stage variables turns out to be a high-dimensional integer optimisation problem with multiple constraints and a complex objective which requires the optimisation of the second stage variables. The first stage problem is addressed by a specific evolutionary algorithm. To calculate the fitness the second stage problem is solved by a standard MIP solver. The design of this evolutionary algorithm is based upon a systematic design approach for metric-based evolutionary algorithms. The aim is to incorporate as much expertise and problem specific knowledge as possible in a transparent manner, to improve the performance of the EA.

As a concise representation of the feasible search space a decision structure is proposed which enables the definition of a similarity measure for search points. Thus, it supports the design of scalable metric-based mutation operators. Compared to a standard integer encoding, the new representation leads to a significant reduction of the cardinality of the set of solution candidates. The results for the application problem underpin the necessity of problem specific adaptations in order to exploit the full potential of the EA approach.

1. INTRODUCTION

Evolutionary algorithms (EAs) are known as powerful direct optimisation methods in many application areas. It has been often emphasised in literature that the full potential of EAs in complex problem domains can only be exploited, if the algorithm is adapted to the problem. Typical modifications can affect the representation, the variation operators and the selection operators [1]. In order to improve the optimisation algorithm we incorporated expertise and problem specific knowledge in the design of the EA. At the same time we tried to avoid adding any bias to the search. To achieve this goal we adopted the design approach of Wiesmann [15] which enables a systematic and transparent knowledge integration in EAs.

As a real-world application problem, the production of polystyrene in a multi-product batch plant with incompletely known demand is considered. This problem belongs to the class of chemical batch scheduling problems which are concerned with the optimal allocation of processing steps (e.g. reaction, separation) to resources (e.g. processing equipment, raw materials) over time with respect to an (economic) objective (e.g. maximisation of profit). Because of the uncertain demand the problem is modelled as a two-stage stochastic integer program (2-SIP). 2-SIPs provide systematic methods to deal with uncertainties that are usually represented by a finite number of scenarios. However, the huge probability spaces in combination with integrality requirements in both stages lead to large-scale optimisation problems [10], which are hardly accessible to standard mixed integer (linear) programming (MIP) solvers such as CPLEX [4].

Two-stage stochastic integer programs are characterised by a two-stage information and decision structure. The set of decision variables is partitioned into a set of first stage variables and a set of second stage variables. The first stage decisions have to be made ‘here and now’, without knowing which scenario will realise in the future. The second stage decisions are recourse actions which can be taken to react to uncertain events, i.e. the realised scenario. Within the scope of chemical batch scheduling, the first stage decisions can be, for example, the allocation decisions for one week. After this week uncertainty in demand may have realised such that recourse decisions can be made. These recourse decisions may be allocation decisions for the second week which depend on the realised scenario and the already made first stage decisions.
Therefore, the possible decision space for the second stage is reduced by the choice of the first stage variables.

The algorithmic approach for 2-SIPs presented here follows a primal decomposition scheme according to which evolutionary algorithms solve the first stage problem, while MIP techniques (CPLEX, cf. [4]) solve the second stage problem. This hybrid algorithm has already been described by the authors in [12]. There, a standard EA for parameter optimisation was used to optimise the first stage variables, namely the integer evolution strategy as proposed by Rudolph [9] and later applied in the context of chemical process design by Emmerich et al. [5].

In this paper we compare a problem specific EA, which is based on a) a problem specific representation of the feasible solutions and b) a scalable mutation operator using a standard integer encoding. The central question of the research reported in this paper is to find out, whether the integration of problem specific knowledge can improve the algorithm performance and thus help to fully exploit the potential of EA as optimiser for the first stage variables. Moreover, this paper can be seen as an instructive example for the systematic design of a problem specific evolutionary algorithm.

In Section 2 we introduce evolutionary algorithms at the example of an evolution strategy for integer programming. Then, in Section 3 we define the general problem of two-stage stochastic programming with application to batch chemical scheduling. Later in Section 4 we propose a problem specific representation and we elaborate scalable mutation operators on it in Section 5. Finally, we compare the problem specific EA to the state-of-the-art approaches in Section 6 and sum up our conclusions in Section 7.

2. EVOLUTIONARY ALGORITHMS

The approach to use evolutionary algorithms as direct optimisation techniques dates back to the early 60ties, when evolution strategies, evolutionary programming and genetic algorithms were originated, all of which now being subsumed in the class of evolutionary algorithms (EA). Since then, the approach has been continuously refined such that evolutionary algorithms are now established in many fields of applied optimisation

EA were also utilised for the purpose of integer programming. Besides a binary coding, also a direct encoding of the integer variables was tried and successfully applied to application problems. For instance, Rudolph [9] developed a variant of the (μ + λ) – evolution strategy [2] for integer programming. Besides, Paramei et al. [3] suggested dual mutation operators for solving mixed-integer programs in power station design with genetic algorithms.

Rudolph’s (μ + λ) – integer evolution strategy (I-ES) [9] starts with an initialisation of μ individuals in the integer search space forming the population \( \mathbb{P}_1 \). Then the following loop is repeated until a termination criterion is fulfilled: Firstly, λ pairs of individuals are taken from the population \( \mathbb{P}_t \). From each pair, an offspring individual is generated by means of uniform crossover and geometrical mutation on the results of each recombination. The μ best individuals, with regard to the objective function, among all individuals in \( \mathbb{G}_t \) and \( \mathbb{P}_t \) that do not exceed a maximal ‘life-span’ of \( \kappa \) generations (iterations) are then selected in order to form a new population \( \mathbb{P}_{t+1} \).

The I-ES was applied in [12] for the optimisation of two-stage stochastic integer programs in chemical batch scheduling. The results of this study are used as a reference for this paper. In [12] the problem was represented as a box-constrained integer optimisation problem such that all feasible solutions are included in the search space. Due to the limitations of parametric representations, the dependencies between integer values were not modelled in this approach such that the variation operators were likely to produce infeasible solutions. For infeasible solutions a penalty value was computed following a metric penalty approach [7] that penalises the constraint violations proportionally to their number and impact.

3. PROBLEM DEFINITION

Two-stage stochastic integer programs represent optimisation problems in which some of the decisions have to be made under uncertainty in the model parameters (first stage) and the remaining decisions can be made after the uncertainty has been realised (second stage). The term ‘integer’ refers to integrality requirements for at least some of the second stage variables.

Formally, two-stage stochastic programs read:

\[
\begin{align*}
\min f(x) & = c^T x + E_{o}[Q_o(x)] \\
\text{s.t.} & \quad A x \leq b, \quad x \in X \\
E_{o}[Q_o(x)] & = \sum_{\omega \in \Omega} \pi_\omega \min_{y_\omega} (q_\omega^T y_\omega) \\
\text{s.t.} & \quad W_o y_\omega \leq h_o - T_o x, \quad y_\omega \in Y \forall \omega \in \Omega
\end{align*}
\]

The parametric uncertainty (e.g. the uncertain demand) in the optimisation problem is reflected by a finite number of scenarios \( \omega = 1, \ldots, Q \) with corresponding probabilities \( \pi_\omega \). The optimisation variables are represented by the vectors \( x \in X \) for the first stage (e.g. the allocation decisions for the first week) and \( y_\omega \in Y \) for the second stage (e.g. the scenario-dependent recourse allocation decisions for the second week), whereby \( X \) and \( Y \) may be subject to integrality requirements. The overall objective (1) is a sum of the first stage costs and the expected value \( E_o \) of the
second stage costs $Q_2(x)$. The latter are a function of $x$ as
the second stage constraints depend on $x$.

The primal decomposition approach is based on the
property that for fixed first stage variables $x$, the second
stage decomposes into $\Omega$ independent subproblems which
can be solved separately for all $\omega$ scenarios using a
standard MIP solver (e.g. CPLEX cf. [4]). Equation (1)
subject to the constraints (2) is called the master problem
and equation (3) subject to (4) is called the subproblem
of the 2-SIP.

In this paper, the real-time scheduling of the multi-product
batch plant (see Fig. 1) for the production of polymers with
uncertain demand is considered [11]. Two types (A/B) of
expandable polystyrene (EPS) in five grain size fractions
each are produced from a number of raw materials (E). The
preparation stage and the polymerisation stage are operated
in batch mode, and the finishing lines are operated
continuously. For each EPS-type, five recipes exist which
determine the grain size distribution (Fig. 2) such that each
polymerisation batch yields a main product and four
coupled products. The capacity of the polymerisation stage
limits the number of batches produced in each time interval
(polymerisation capacity constraint). The feed into the
finishing lines is given by the number of polymerisation batches produced per time interval and is required to be
either zero or between the minimum and the maximum
capacity of the finishing lines (finishing line capacity
constraint). If a minimum feed to a finishing line cannot be
provided, this line has to be shut down temporarily. Due to
the costs, the number of state changes (on/off) of the
finishing lines is given by the number of polymerisation batches produced per time interval and is required to be
either zero or between the minimum and the maximum
capacity (finishing line dynamics constraints). The shortage
of supply and the inventory of products need to be
determined for the penalty costs for late demand supply and
for the storage costs, respectively (cost constraints).

Main scheduling decisions are the discrete choice of the
recipes for the polymerisation batches and their starting
times. The objective is to maximise the profit (minimise
the costs) calculated from sales revenues, production costs,
storage costs and penalties for lateness and for finishing

![Figure 1: The flow sheet of the EPS-plant](image1)

![Figure 2: Distribution of grain sizes](image2)
products. The decisions of both stages have to be made subject to the polymerisation capacity constraints, the finishing line capacity constraints and the finishing line dynamics constraints, whereas the cost constraints appear in the second stage only.

4. Problem specific representation

The first stage problem for the example process at hand is highly constrained. Only a small ratio of the integer combinations represents feasible solutions. By application of combinatorial rules it was found that the cardinality of the feasible set is $|K| = 8.50 \cdot 10^5$. Compared to the cardinality of $|S| = 2.62 \cdot 10^3$, the contingent of feasible solutions in $S$ is evanescent small [13].

A problem specific representation of the feasible set was developed using the constraint programming concepts constraint propagation and domain reduction [8]. It is based on the observation that feasible subsets correspond to feasible operating modes of the finishing lines ($z_{i,p}$) and feasible sums of subsets of recipes ($F_{i,p}$, see below) respectively [13]. However, fixing the decision variables in the order $z_{i,p} \rightarrow F_{i,p} \rightarrow s_{i,p}$, enables propagating constraints in a strictly hierarchical fashion.

Based on these techniques, a decision structure representing the feasible set of first stage decisions can be defined. Special attributes of the constraints enable a concise representation of this structure. It is based on the observation, that only the finishing line dynamics constraints define dependencies between first stage decision variables of different intervals namely dependencies between the variables $z_{i,p}$. Once these variables are fixed the remaining decisions decompose into three independent decisions corresponding to each interval. For each interval, the polymerisation capacity constraint is the only constraint which defines dependencies between recipe assignments of different products, $s_{i,A}$ and $s_{i,B}$, by constraining the feasible values for the sum of their entries $F_i = \sum_{p=1}^{3} s_{i,p}^j = F_{i,A} + F_{i,B}$, with $F_{i,p} = \sum_{j=1}^{3} s_{i,p}^{(j)}$. Once the values $F_{i,A}$ and $F_{i,B}$ are fixed, the recipe assignment decisions decompose into two independent decisions, one for each product.

A decision tree is used to determine all feasible values for the variables $z_{i,p}$. The operating modes are subject to the constraint that a finishing line must keep its operation mode for at least two intervals. Each level of the so called operating mode tree (OMT) corresponds to one of the three first stage intervals, see Fig. 3. Each node on the i-th level of the OMT is labelled with feasible values for the variables $z_{i,A}$ and $z_{i,B}$. The node '00' on level 1, for example, stands for the combination $z_{i,A} = 0$ and $z_{i,B} = 0$, which indicates that both finishing lines A and B are not operated. Each path from the root to a leaf in the OMT represents a feasible sequence of operation modes.

All dependencies between decisions of different intervals are coded in the OMT such that the assignment decisions of each interval can be considered independently. There are only four possible combinations of $z_{i,p}$ for one interval: '00', '10', '01' and '11'.

Whenever a finishing line is not operated ($z_{i,p} = 0$) all entries of the corresponding recipe assignment vector $s_{i,p}$ as well as the sum $F_{i,p}$ have to be zero in order to meet the capacity constraint of the finishing line. Whenever $z_{i,p} = 1$, the capacity constraints bound the sums $F_{i,p}$ (capacity of a finishing line $p$) and $F_{i}$ (capacity of the polymerisation stage). $F_{i,p}$ corresponds to the number of batches of EPS-type $p$, and $F_{i}$ corresponds to the total number of batches that are produced in interval $i$. The sum $F_{i,p}$ is bounded to the set of $5$ to $12$, and $F_{i}$ is bounded to the set of $0$ to $12$.

All feasible values for $F_{i,A}$ and $F_{i,B}$ can be represented in a concise decision tree for each combination of $z_{i,A}$ and $z_{i,B}$, see Fig. 4. The root of each capacity tree (CT) is labelled with the corresponding value pair $z_{i,A}$ and $z_{i,B}$.

The dependencies between assignment decisions of different products are coded in the CTs such that the recipe assignments for each product can be considered independently with respect to the possible values of the sum $F_{i,p} = k$, with $k = 0$ or $k = 5, \ldots, 12$.

The set of vectors $s_{i,p}$ with a given sum $F_{i,p} = k$ can be easily generated and indexed. The number of vectors

![Figure 3: Operating mode tree (OMT) – decision tree of the operating mode variables $z_{i,p}$](image)

![Figure 4: Capacity trees (CT) – decision trees of the sums $F_{i,p}$](image)
whose sum is \( k \) is \( \begin{bmatrix} 5-k \end{bmatrix} \). For each \( k \), the set of vectors \( s_{i,p} \) is represented by a decision tree called index tree (IT), shown in Fig. 5. The leaves of the ITs represent one index each.

By fixed values of \( z_{i,p} \) and \( F_{i,p} \) a leaf (an index) of the corresponding IT and thus a feasible assignment for the vector \( s_{i,p} \) can be chosen for each interval and finishing line. By means of these simple trees (OMTs, CTs and ITs) each feasible solution for the EPS-model \( s \in S \) can be identified uniquely. In addition to the trees, all five-dimensional vectors with sums of \( k = 5,\ldots,12 \) are stored.

The genotype of an individual is defined as a path in the OMT, three leaves of the CT (one for each interval), and six leaves in the IT (one for each interval and product). Each set of path and leaves as mentioned above represents a unique feasible solution \( s \in S \), which can be determined easily.

5. Mutation Operator

The design of a mutation operator is based upon the design approach for metric based EAs (MBEA) as presented by Wiesmann ([14, 15]). He provides methods which support the design of variation operators such that the distance in the search space and the difference in the objective function value is positively correlated. Thereby, by adjusting the sampling radius of the variation operators the impact of the mutation operator on the change in objective functions can be controlled. Consequently, the search strategy can be tuned within the spectrum from coarse exploration to local search. The minimal moves concept, as described by Emmerich et al. [6] is employed to establish a metric on the search space of decision paths.

The mutation operator applies a sequence of randomly selected elementary modifications – so called minimal moves – in order to generate a variation of a given individual. The set of minimal moves is complete, if any solution can be reached from any other solution by a finite number of minimal moves. To each minimal move a weight is assigned, the value of which should be chosen proportionally to its impact on the alteration of the objective function value. From such a complete set of weighted minimal moves, a scalable mutation operator is constructed. By a single application of this mutation operator a finite number of minimal moves is executed in a sequential manner on the same individual. Minimal moves are executed as long as the sum of their weights does not exceed a threshold \( T \). The value of \( T \) can be considered as the counterpart of a step-size in discrete spaces, because it scales the impact of the mutation. It is adapted during the course of evolution by means of mutative self adaptation [2].

To guarantee the completeness of the set of minimal moves, minimal moves are defined on each level of the decision trees (OMTs, CTs and ITs, see Figs. 3, 4 and 5). Their weights correspond to the expected alterations of the fitness/objective value which can be approximated by experiments. If minimal moves can be defined which cause the same fitness alteration for many different individuals, the fitness alteration caused by the overall mutation operator is expected to be strongly correlated to a given step size. In the ideal case, the correlation coefficient of the fitness value of an individual and an individual varied by a minimal move is close to 1. In this case the search behaviour of the algorithm is predictable to some extend.

The smallest steps in the decision space are changes in an IT. The grain size distributions as depicted in Fig. 2 imply a similarity in the final product if neighbouring recipes are assigned. A swap of two neighbouring positions in one of the six five dimensional vectors \( s_{i,p} \) is considered as a minimal move on the IT level. Note, that this move does not alter the sum \( F_{i,p} \), such that no changes of the CT-leaves or in the OMT-path have to be made to ensure the feasibility of the individual. Fig. 6 reveals clearly the strong correlation between neighbouring solutions obtained by this minimal move.
The correlation of objective function values between a randomly selected individual and a randomly generated neighbour by means of this minimal move was 0.996 in mean over 100 randomly chosen initial individuals.

A change in a CT is considered as a second minimal move. The smallest change in the final product profile is obtained if the number of only one recipe is de- or increased by one. This move corresponds to altering one sum $F_{i,p}$ by one, which can be done by choosing a neighbour leaf of one of the three CT-leaves. However, changes of the corresponding IT-leaves have to be made to ensure feasibility of the new individual, because each IT belongs to a certain sum $k$. If the sum changes from $k_1$ to $k_2$, an IT-leaf of the IT with the root $k_2$ has to be chosen. Fig. 7 shows the strong correlation between neighbouring solutions caused by this second minimal move.

The correlation of objective function values between a randomly selected individual and a randomly generated neighbour by means of this second minimal move was 0.990 in mean over 100 randomly chosen parent individuals.

In addition to the minimal moves on the levels of the CTs and the ITs, minimal moves have to be defined on the OMT levels in order to guarantee the reachability of all solutions in the search space. Since each CT corresponds to a certain value pair of $z_{i,A}$ and $z_{i,B}$, changes in the OMT-path require changes on the CT level and the IT level (see above). However, various experiments (cf. Fig. 8a-c) did not show any correlation between the fitness values of individuals on neighboured OMT-paths. Nevertheless, it is obvious that the average fitness alteration of these individuals is significantly higher than the fitness alteration of individuals within the same OMT-path. Thus a higher weight is assigned to minimal moves on the OMT level, which makes changes on this level less likely to occur.

---

**Figure 7:** Objective function values of parent individuals vs. offspring individuals generated by random application of the minimal move on the CT level and adaptation of the corresponding IT-leaves

**Figure 8a:** Objective function values of parent individuals vs. offspring individuals generated by random application of the minimal move on the third level of the OMT and adaptation of the corresponding CT- and IT-leaves

**Figure 8b:** Objective function values of parent individuals vs. offspring individuals generated by random application of the minimal move on the second and third level of the OMT and adaptations of the corresponding CT- and IT-leaves

**Figure 8c:** Objective function values of parent individuals vs. offspring individuals generated by random application of the minimal move on all three levels of the OMT and the adaptations of the corresponding CT- and IT-leaves
As no correlation was found for the neighborhood based moves, minimal moves are applied, which randomly choose one of the nodes with the same father node on one level of the OMT. The nodes on the lower levels in the OMT – if there are lower levels left – and the leaves of the other trees (CTs and ITs) are chosen randomly. Search on these levels corresponds to a fully random search.

All minimal moves are weighted and applied within the mutation operator of the P-EA. Fig. 9 shows the results of the employment of the mutation operator with given step sizes up to the size of 70 with three different weighting vectors $g_1 = (1, 2, 15, 16, 17)$, $g_2 = (1, 2, 25, 30, 35)$, and $g_3 = (1, 2, 35, 45, 50)$. 100 randomly chosen parent individuals were mutated with each step size and the absolute value of the fitness difference of parent and offspring was calculated.

Fig. 9 reveals clearly that there is a strong correlation between the given step size of the mutation operator and the fitness difference of parent and offspring for step sizes up to a size of 45. As mentioned before each minimal move is only taken into consideration by the overall mutation operator, if its weight does not exceed the given threshold $T$. The significant increase of the fitness alteration at a step size of 15 ($g_1$), 25 ($g_2$) and 35 ($g_3$) is caused by the application of the minimal moves on the OMT levels. Due to the random search strategy on the levels of the OMT, the tests with step sizes greater than a size of 20 ($g_1$), 30 ($g_2$) and 45 ($g_3$) did not show such a correlative relation. Weighting vector $g_3$ was chosen, because the correlative relation between the fitness alteration and the step sizes is given for step sizes up to 45.

For a detailed description of the analysis of the first stage search space, the design of the mutation operator, the strategy parameters and the empirical results of the P-EA we refer to the technical report by Urselmann [13].

6. RESULTS

The given problem is a real-time application and the CPU-time is limited to four hours. An integer evolution strategy (I-ES) [9] on the full parameter space, a problem specific evolutionary algorithm (P-EA) and a standard MIP solver (CPLEX, cf. [4]) were compared. The characterisation of the reported algorithms was as follows: For the I-ES a population size $\mu = 10$ and an offspring population size of $\lambda = 70$ were chosen. For the P-EA the corresponding values were $\mu = 5$ and $\lambda = 35$. The initial mutation rate for the I-ES was 1.2 and for the P-EA the initial step size was 15.0. All values were obtained after some tuning of the algorithms (cf. [12, 13]). All algorithms were initialised with a feasible solution that was generated randomly by means of the decision structure described in Section 4.

Fig. 10 shows the median progress curves of the three different optimisers. The best solutions found were -51.45 for the I-ES, -51.7 for the P-EA and -49.07 for CPLEX. An analytically determined lower bound for the given problem is -52.63, such that the gap between the global optimum and the found solution is at most 2.29% for the I-ES, 1.80% for the P-EA and 7.25% for CPLEX. Table 1 summarises the exact values of the best fitness values found after 0.5, 1, 2, 3 and 4 hours of computation and the corresponding optimality gaps with respect to the mentioned lower bound.

Fig. 10 and Table 1 reveal that the P-EA can obtain good results much faster than the conventional EA. The final result of the I-ES was obtained in about 2½ hours, and the final result of the monolithic algorithm was obtained by the P-EA in only 35 minutes.
Table 1: Median of the best objective value found after fixed CPU-time.

<table>
<thead>
<tr>
<th>CPU-time in [h]</th>
<th>P-EA</th>
<th>I-ES</th>
<th>CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>-48.69</td>
<td>-46.67</td>
<td>-48.33</td>
</tr>
<tr>
<td>1</td>
<td>-50.10</td>
<td>-48.24</td>
<td>-49.07</td>
</tr>
<tr>
<td>2</td>
<td>-51.27</td>
<td>-50.68</td>
<td>-49.07</td>
</tr>
<tr>
<td>3</td>
<td>-51.55</td>
<td>-51.11</td>
<td>-49.07</td>
</tr>
<tr>
<td>4</td>
<td>-51.7</td>
<td>-51.45</td>
<td>-49.07</td>
</tr>
<tr>
<td>lower bound</td>
<td>-52.63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>optimality gap</td>
<td>1.80%</td>
<td>2.29%</td>
<td>7.25%</td>
</tr>
</tbody>
</table>

Table 2: Best, worst and median objective value found after four hours of computation in five optimisation runs.

<table>
<thead>
<tr>
<th>algorithm</th>
<th>best</th>
<th>median</th>
<th>worst</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-EA</td>
<td>-51.83</td>
<td>-51.7</td>
<td>-51.27</td>
</tr>
<tr>
<td>I-ES</td>
<td>-51.66</td>
<td>-51.45</td>
<td>-50.69</td>
</tr>
<tr>
<td>CPLEX</td>
<td>-49.07</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Best first stage solutions found by the applied algorithms.

<table>
<thead>
<tr>
<th>algorithm</th>
<th>best first stage solution x</th>
<th>f(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P-EA</td>
<td>(0, 1, 2, 3, 1, 0, 0, 0, 0, 0, 2)</td>
<td>-51.83</td>
</tr>
<tr>
<td>I-ES</td>
<td>(0, 0, 1, 2, 3, 0, 0, 0, 0, 2, 2)</td>
<td>-51.66</td>
</tr>
<tr>
<td>CPLEX</td>
<td>(0, 0, 4, 0, 1, 2, 0, 0, 0, 0, 7)</td>
<td>-49.07</td>
</tr>
</tbody>
</table>

The significant reduction of computation time and the improved quality of solutions can be explained by the more powerful search operators and the reduced search space complexity. Thanks to the decision structure the algorithm is able to stay within the feasible domain such that evaluations of infeasible individuals are avoided.

To confirm the good results Table 2 shows the best, the worst and the median objective value of the five optimisation runs after four hours of computation. Due to the deterministic behaviour of the algorithm CPLEX, the table comprises only one solution for CPLEX.

The best first stage solutions found by the three different algorithms are given in Table 3.

7 Conclusions and Outlook

A hybrid EA/MIP method for solving two-stage stochastic programming problems in chemical batch scheduling has been proposed. A problem specific approach for the optimisation of the first stage variables was developed by implementing the metric based EA approach proposed by Wiesmann for this specific problem domain. As a problem representation a decision structure was applied, on which problem specific weighted minimal moves were defined. By means of correlation analysis, it was proven that the resulting mutation operator is scalable with regard to the impact on the alteration of the objective function value and thus complied with the requirements for a mutation operator within the MBEA framework. The resulting evolutionary algorithm outperformed deterministic methods (CPLEX) and an integer evolution strategy for optimising the first stage variables. The new approach comes up with a significantly improved solution quality. The best solution found by the P-EA is better than the best solution of the other algorithms at each point of time during the optimisation run. These results further encourage the use of knowledge integration techniques in EA and show that the full potential of the EA approach cannot be exploited by using EA on standard representations. In particular this statement seems to hold for problems with highly constraint spaces, like the one tackled in this paper.

In order to further improve the optimisation process of the P-EA, it would be conceivable to search for smooth transitions on the OMT, so that the search on those levels does not accord to a random search. Another direction of future research could be to try different non-panmictic population models, in order to maintain a higher diversity during the final stage of the search.

Acknowledgements

The financial support by the German Research Foundation (DFG) for the Collaborative Research Center (Sonderforschungsbereich) Design and Management of Complex Technical Processes and Systems by Means of Computational Intelligence Methods (SFB 531) at the Universität Dortmund is gratefully acknowledged. M. Emmerich acknowledges financial support by the Netherlands Organisation for Scientific Research (NWO).
REFERENCES