SPECIALIZED GENETIC OPERATORS FOR MULTI–SKILL RESOURCE–CONSTRAINED PROJECT SCHEDULING PROBLEM

Paweł B. Myszkowski, Marek E. Skowroǹski

Wrocław University of Technology
Department of Artificial Intelligence
Wybrzeże Wyspiańskiego 27, Wrocław, Poland
pawel.myszkowski@pwr.wroc.pl, m.e.skowronski@pwr.wroc.pl

Abstract: *In this paper specialized genetic operators for Evolutionary Algorithms in the Multi–Skill Resource–Constrained Scheduling Problem have been proposed. The problem objective is to assign resources to tasks to make the final schedule as short or/and cheap as possible (multi–objective optimization). In our approach domain knowledge has been applied to create more domain–based, specialized operators. They have been compared with the classical ones – simple mutation, one–point crossover and two–point crossover. Experiments have been conducted on a dataset, that was created artificially, based on real–world project instances got from Volvo IT enterprise.*

Keywords: *scheduling, multi–objective optimization, evolutionary algorithms, specialized operators*

1 Introduction

The Resource–Constrained Project Scheduling Problem (RCPSP) [7] is constrained, multi–objective, combinatorial problem, which objective is to assign given resources to given tasks, to make the project schedule as short and / or cheap as possible. In RCPSP many constraints related to resources and tasks have to be satisfied, e.g. given task cannot start before its predecessors would be finished or given resource cannot have more than one resource assigned in overlapping periods of time (such situation is called *conflict* and would be explained in details later in this paper).

The classical RCPSP can be extended by introducing skills domain [1, 3, 10], being transformed to Multi–Skill Resource–Constrained Project Scheduling Problem (MS–RCPSP). Every task requires some skill on appropriate level to be performed. While every resource owns some given skills pool, not every resource is capable to be assigned to every task. MS–RCPSP is very practical – many companies look for ways of optimizing their schedules. Hence it is worth to be investigated and researched, as every saved day or money during the project planning stands concrete value for enterprise.

Because of the combinatorial nature of MS–RCPSP, it could be regarded as NP–hard problem [2]. It means that no optimal solution can be found in polynomial time. It suggests using some soft computing methods, like metaheuristics. One of proposed method from this group are Evolutionary Algorithms (EA) which are capable of flexible searching large solution space. The ability to extend the EA model by own domain–specialized genetic operators enhance the potential of EA robustness. The goal of this paper is to investigate whether proposed domain knowledge–based operators can enhance the quality of EA in solving MS–RCPSP.

The rest of the paper is organised as follows. Section 2 shows the state-of-the-art in using EA–based methods to solve RCPSP, Section 3 explains the MS–RCPSP statement. In Section 4 an investigated approach is presented with the detailed description of proposed specialized operators, while Section 5 shows and discusses obtained results. Finally, Section 6 provides conclusions that have been made on the base of conducted experiments and gives the directions of future work.

2 Related work

EA have been often used in the literature as a successful approach for solving any variations of RCPSP [4, 5, 6, 9, 12]. An example is presented in [4], where an individual structure is related to the sequence of tasks in the order they would be performed. Each task is sequenced as early as possible but with respect to precedence– and resource–constraints. In this approach several classical crossover operators have been proposed: classical one–point, two–point and multi–point crossover operators. The uniform crossover operator has been also presented in that paper. Mutation is provided by changing the position of given task in the scope between the last of its predecessors and the first of its successors.

The same task sequence–based, chromosome representation has been proposed in [5]. Proposed crossover operator is some hybrid of two–point classical crossover and position–based crossover. The genotype of parents is split into three parts to generate one offspring. Genes from first and third part are transferred from the first parent to the offspring. The middle part is filled with the remaining tasks in the order preserved by second parent. The mutation operator chooses two tasks and swaps them if it is possible – the mutation would not disturb the precedence constraints. In that paper tournament selection with the tournament size equals to 2 is provided.

Similar approach has been presented in [6], where the individual's structure is based on task list and the schedule is generated by serial generation scheme. Crossover operators that are proposed there: classical position–based crossover and partially–mapped crossover (PMX) with the precedence–relation structure's fixing. The fixing procedure is required, as PMX tends to disturb an individual. Beside the classical swap mutation, which performs the same like in [5], some other mutation, based on the local search procedure, has been proposed. It is performed by creating *neighbourhood* of given individual, in which individuals are varied by the change of the only one task in the order.

Modified representation has been proposed in [9], where the chromosome consists $2n$ genes, where n is a number of tasks in a schedule. The first n genes describe tasks' priority, while the second n provide the delay times between tasks execution. The uniform crossover operator is provided with the parametrized threshold parameter. The parents for crossover are selected in a way that one of them belongs to group of the best individuals of population and the second one is taken randomly. The mutation operator is performed as being replaced by randomly created chromosome that preserves precedence– and resource–constraints.

The approach proposed in [12] presents a hybrid of some methods with classical tasks–list chromosome representation and serial generation scheme [12]. Innovative peak crossover operator has been proposed, which concerns the load of resources in a schedule. Given two parents, the offspring would get the genotype that reflects the resource load peak from the first parent and completes the genotype with the remaining genes from the second parent. Parents are selected to crossover similarly to [9]. In that approach swap mutation has been proposed but does not regard only one pair in chromosome, rather than all available pairs.

3 Problem statement

In MS–RCPSP some general elements can be pointed out. Each schedule consists of tasks with their precedence relations, resources and skills linking tasks with resources. Every **task** is described by task duration, its start and finish dates and potential successor or predecessor tasks (**precedence relations**). It is because some tasks have to be performed before some others. **Resources** are described by their salary and specified pool of owned **skills**, while each task requires some skill in a given skill level to be performed. It means that resource has to be capable of performing given task – not every task can be performed by each resource.

The objective of MS–RCPSP is to assign resource to every task to get the shortest / cheapest schedule as possible. Thus it could be presented as a **multi–objective optimization problem**: minimization of the project schedule's performance cost and minimization of project schedule's duration.

Those two objectives are in opposition to each other (time & cost trade–off problem). To show it, let's take into account some sample project, where some resources are given and their salaries are different. Leaving the skill constraints for a moment, we can assume that if the objective would be to find the cheapest project, each task would need to have the same – cheapest resource assigned. However, if only one resource would be able to perform each task in a project, they would need to have their start dates shifted to avoid conflicts. In fact it would cause the tasks' serial execution, enlarging the project duration enormously. On the other hand, if we would like to shorten the project duration, we would need to make as many tasks as possible to be performed in parallel. That forces using different resources. Thus there is no possibility to perform each task by the same, the cheapest resource in the same time.

4 Proposed approach

In this paper, an EA–based approach has been proposed. EA is stochastic search method that uses the metaphor of natural Darwinian Evolution. EA operates on a population of potential solutions applying the principle of survival of the fittest ones to produce better solutions. At each generation, a new set of solutions is created by the process of selecting individuals according to their value of fitness in the problem domain and breeding them together using crossover and mutation operators inspired by natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.

An individual is represented as a resource–to–task assignments vector. Each assignment involves only one specified task. Proposed approach assumes creating the project schedule in two main steps: (1) assign resources to tasks (setting an individual) and (2) build a feasible schedule from such assignments. In the first step the

resource pool that is capable of performing indicated task is generated. Only a resource from such pool can be assigned to given task, to preserve skills constraints between tasks and resources. Next, when each task has its own assigned resource, the schedule is ready to be built (2). First **the earliest start time date** within the task pool is found. Each task is then **set to be started in this specified date**. However it leads to conflicts, because every involved resource could have more than one assigned task during the schedule. Hence, **the conflict fixing procedure is launched**. Conflicts are removed by **shifting the start date** of conflicting tasks assigned to given resource until no conflicts would be found. Mutually to schedule fixing procedure (conflict elimination), the **critical path** need to be preserved. It is because after shifting some tasks to avoid conflict within the range of given resource, the critical path can be disturbed. The critical path is preserved by **setting start time date** of given task **no earlier than the latest finish time date of its predecessors**. Hence the critical path fixing procedure is executed after each execution of conflict fixing procedure.

4.1 Evaluation function

Individuals are evaluated by two opposite functions – the project duration and its performance cost. However, to make it possible to merge those two aspects in one evaluation function, their values had to be normalized to [0;1] value range. The normalization has been performed by dividing values for both functions by their maximal possible value: the length of serial task execution for project duration and cost of performing each task by the most expensive resource for performance cost.

The project schedule's duration $f_\tau(PS)$ and cost $f_c(PS)$ components are weighted by the values that sum to one (see Eq. 1) e.g. if the weight for duration aspect (w_τ) would be set to 1, the weight for cost aspect would be 0 and vice versa. If the balanced optimization is desired, both weights should be set to 0.5. It reflects the trade–off character of optimization. The evaluation function of given individual PS is minimized and defined as follows:

$$f(PS) = w_\tau f_\tau(PS) + (1 - w_\tau) f_c(PS) \tag{1}$$

4.2 Classical genetic operators

The classical operators from the literature have been compared with specialized operators proposed in this paper. The classical operators that have been taken into consideration are one–point crossover (1X), two–point crossover (2X) and simple mutation (SM). The *1X* splits the parents' structure in the middle, transfers the first half of parents to its offspring and exchange the second half of genotype within the offsprings. The *2X* cuts the genotype in two points, creating three equal sized parts. First and the last part of genotype is transferred straightforward to offsprings, while the middle part is exchanged. The *SM* mutation operator changes the only given gene in genotype. However it cannot be made randomly. In MS–RCPSP the mutation can involve only such gene (assignment) that regards the task to which more than one resource can be assigned, respecting the skills constraints. Hence the task for which an assignment would be changed is sought until above mentioned condition is not satisfied.

4.3 Specialized operators

Due to the semi–blind character of classical genetic operators, we proposed some new specialized ones, including the domain knowledge, to improve the robustness of EA in solving MS–RCPSP: two crossover and one mutation operator. In proposed crossover operators only one offspring is created from two parents. However the decision the gene from which parent is taken to the offspring is completely different.

To get bigger flexibility of proposed operators, two stages of parents' selection have been investigated [9]: selecting individuals to the next population (1) and selecting parents from that population's parental pool to crossover (2). Parents can be chosen randomly or after sorting in parental pool. Therefore pair of parents for crossover is always chosen in a way to preserve that each pair has one *good* and one *bad* solution, while *good* solution is any from the first half and *bad* solution is from the second half. Parental pool approach (2) is made independently to the general selection method (1) (e.g. tournament selection), applied in this paper.

4.3.1 Cheaper Resource Crossover (CRX)

The main idea of CRX is to get cheaper assignments from parents. For each task assignments from parents are investigated and the resource with smaller salary–rate is taken to be assigned to the same task in offspring. If the salary rates are equal, number of assignments is considered and the resource with the smaller one is taken to be assigned. If it also cannot indicate the winner resource, it is chosen randomly within the resources from that task in parents. It provides that produced offspring would not be more expensive than any of its parents. In most cases it would be significantly cheaper. However, such approach could cause more conflicts than less resource–salary–rate based approaches, what could enlarge the project duration.

4.3.2 Less Assignments Crossover (LAX)

LAX approach is duration–oriented. It analyses assigned resources for each task in parental individuals and assigns the one with the smallest number of assignments to given task in the offspring. If the number of assignments for investigated resources in both parents is equal, the one with the smallest salary–rate is taken to be assigned in offspring. The created offspring is expected to have less conflicts, because resources are assigned to tasks in more balanced way. This approach has one drawback – if there are two parents in which only one resource performs each task, the crossover operator would not be able to balance the workload.

4.3.3 Conflict Avoidance Mutation (CAM)

The CAM operator bases on the knowledge of fixing conflicts procedure – it shifts the task start date just after finish date of other conflicted task. Based on that domain knowledge, we can assume that each pair of assignments within the range of one resource assignments pool should be considered. Hence, assignments are sorted by their start dates and then are compared to their followers. If a duration between finish date of previous and start date of the next assignment is less or equals one day, we can assume that those two tasks have been processed by conflict fixing procedure. Such pair of assignments is taken as an input to be mutated.

The mutation is performed as follows. From pair of assignments the task with higher number of resources capable of performing it is chosen, because this task would be easier to be re–assigned. Then the resource that has smallest number of assignments within the pool obtained in previous step is chosen to be assigned to specified task. If number of resources capable of performing both tasks is the same, the task to be re-assigned is chosen randomly. If there are more than one resource than can be assigned to selected task, the first one with the smallest number of assignments is taken into account.

As a result, an individual is expected where there would be less conflicting assignments. Consequently less conflict fixing method executions would be performed. Thus, less shifting should be made and shorter project duration should be obtained.

5 Experiments and results

As the MS–RCPSP is regarded as a multi–objective optimization problem, results obtained during experiments have been evaluated both by their project duration (expressed in days) and their performance cost (expressed in some abstract currency units – [c.u.]). During the experiments, approaches with proposed specialized crossover and mutation operators have been compared with the classical ones.

5.1 Experiments' set-up

Because the goal of this paper is to compare different approaches in performing crossover and mutation operators, specialized to the MS–RCPSP, several other parameters of Evolutionary Algorithm have to be set up as constant, to provide the same environment for experiments. Both population size and number of generations were arbitrary set to 100. Crossover probability was set to 0.3, while mutation probability was set to 0.1. Tournament selection method with the tournament size equal 2 has been proposed. Furthermore the clone replacing mechanism has been introduced – if any clone is found in a population, it is replaced by new randomly created individual. Those EA parameters have been set up experimentally.

In our research the following notation has been proposed for weights configuration: Duration Optimization (DO), where $w_\tau = 1$, Balanced Optimization (BO), where $w_\tau = 0.5$ and Cost Optimization (CO), where $w_\tau = 0$. Parental pool type (R – random, S – sorted) has been investigated both for proposed specialized operators (CAM, CRX, LAX) but also for classical (SM, 1X, 2X) ones. The parameter configuration notation has been proposed as follows: *parental pool selection mode(crossover type)+mutation type*, e.g. *R(CRX)+SM* means random parental pool selection, cheaper resource crossover (CRX) and simple mutation (SM), while *S(LAX)+CAM* reflects sorted parental pool selection, less assignments crossover (LAX) and conflict avoidance mutation (CAM).

The goal of experiments was to research, whether proposed specialized operators could enhance the EA robustness in optimizing project duration or performance cost in comparison to classical, semi–blind operators.

5.2 Dataset

Due to lack of cost-oriented details in classical benchmark project scheduling dataset (e.g. PSPLIB [8]), the dataset provided for our experiments had to be artificially created. However, we base on real–world project schedules data, received from Volvo IT Department in Wrocław. It let us to get know, what are the typical sizes of project schedule – how many tasks and how many resources are involved and how tasks are related between them

by precedence relations. Finally, we constructed and published the dataset (http://www.ii.pwr.wroc.pl/~myszkowski/scheduli
based on our research, which has been briefly described in the Tab.1.

Table 1: MS–RCPSP dataset description

Property	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**
Tasks	100	100	100	200	200	200
Resources	20	10	5	40	20	10
Skills	9	9	9	9	9	9
Relations	20	26	22	133	148	129

The dataset instances have been mainly divided into two groups – containing 100 and 200 tasks. For each group of project instances, respective number of resources was provided, to make the average resource load the same. Project instances from the group of 200 tasks have been modified in comparison to those from the group consisting 100 tasks by the number of relations – tasks in D4–D6 project instances are significantly more related and therefore more difficult to schedule.

5.3 Results

Due to stochastic character of EA, each experiment has been repeated ten times. Values presented in the Tab.2, where the summary of performed experiments has been provided, are the best obtained ones: the smallest duration for DO, the smallest cost for CO and the smallest value of evaluation function for BO. If there are more than one value with the smallest first property, the smallest second one is taken as the best (non–dominated solutions). Due to the lack of free space in the article, no average and standard deviation from obtained results is presented. Moreover, from the end–user point of view, the average values are not as valuable for project manager as the best obtained ones, because the end–user would like to make the project completed as cheapest or / and shortest as possible. Even though such solution is obtained only once in many EA runs. One EA run with 100 individuals in population and 100 generations lasts about 10-15 minutes for Intel Core 2 Duo P8700 (2.53 GHz at each core) and 4 GB memory RAM.

Table 2: Project duration and performance cost – classical and specialized operators – best results summary

ID	Specialized operators						Classical operators — R(1X)+SM					
	DO		BO		CO		DO		BO		CO	
	days	cost	days	cost	days	cost	days	cost	days	cost	days	cost
D1	R(CRX)+CAM		R(1X)+CAM		S(CRX)+SM		35	41217	46	37190	56	35760
	32	**41509**	**32**	**42975**	**116**	**30158**						
D2	R(1X)+CAM		R(LAX)+CAM		R(CRX)+SM		52	37248	77	31888	94	31328
	34	**42804**	**40**	**40387**	**133**	**26691**						
D3	R(LAX)+CAM		S(1X)+CAM		S(CRX)+SM		64	40242	77	35527	**84**	**34160**
	52	**40768**	**57**	**38486**	163	34361						
D4	S(CRX)+SM		S(2X)+CAM		R(CRX)+SM		112	87487	**114**	**79854**	120	78928
	112	**66196**	112	87107	**196**	**52027**						
D5	R(LAX)+SM		R(LAX)+CAM		R(CRX)+SM		183	81555	**211**	**72918**	230	72338
	179	**90753**	188	84067	**417**	**52400**						
D6	R(CRX)+SM		R(CRX)+CAM		S(CRX)+SM		216	99462	216	92602	216	91972
	216	**81344**	**216**	**88317**	**294**	**74897**						

All combinations of crossover and mutation operators (classical and specialized) have been investigated. The complete summary of proposed experiments is presented in the Tab.2. Based on obtained results, some conclusions can be made. First of all, CRX operator outclassed any other operators in cost–optimization perspective, as it provided the best results for cost optimization in all experiments. What is more, the differences between cost property for CRX operator and other are really significant – performance cost for classical crossover operators with SM operator is, in average, over 24% bigger than for CRX operator with the same mutation operator.

The parameter configuration has been investigated in detail by checking how many times proposed value of given parameter became the best. The random parental pool selection mode for crossover became the best 12 times (67%), while the sorted one was *the winner* 6 times (33%). An interesting observation concerns the distribution of parental pool selection mode. For DO sorted mode was the best once, twice in BO and three times in CO. Base on that observation, a conclusion could be made that the more cost–oriented an optimization is, the more sufficient would be to use sorted parental pool selection mode.

Taking into investigation the distribution of mutation type, there is no difference between number of SM or CAM *wins*. Both operators provided 9 best results. However, for CO only SM operator became the best, while for BO only configurations with CAM appeared. Furthermore the influence of crossover operator for obtaining best results has been analysed. 1X operator became the best three times (16%), while 2X – only once (6%). CRX operator became the best most times – 10 (56%), *winning* for each dataset instance in CO. LAX operator was the best four times (22%), but only for DO and BO.

Results obtained for combinations of specialized operators have been compared with the results obtained for classical operators' configuration – *R(1X)+SM*. Best results from combinations became *winners* in 15 (83%) times, while results from classical operators configuration turned out to be better in only 3 times (17%). It shows, how specialization of genetic operators could enhance the robustness of evolutionary algorithms.

6 Conclusion

Provided results of conducted experiments suggest that proposed specialized operators can be successfully applied in EA, while specific direction of optimization of project scheduling is expected. CRX operator could be applied with success for cost–oriented optimization. On the other hand, when a project manager is more interested in getting more duration–oriented optimization for resulted schedule, the CAM + LAX combination is suitable.

Due to presenting the first results with fixed EA parameters' configuration, they should be extended in the further work by experiments that would help to provide the best parameters' configuration for each of proposed operator. We claim the cooperation of specialized operators as a promising further research direction. To avoid setting weights, what requires experience from a project manager, we consider a Pareto–optimization as a potential future work. We also plan to compare other specialized crossover operators for scheduling, like Precedence Preservative Crossover or Subsequence Exchange Crossover. What is more, proposed dataset is going to be extended in the further work.

Before a decision, which genetic operator choose for specified project to be scheduled, it cannot be forgotten that first–objective–optimization would cause (mostly) negatively in the second aspect. Thus the question is how big the negative spin–off effect would be. This decision should be made individually by a project manager, because deciding person in each enterprise has completely different toleration criteria.

References

[1] Al–Anzi, F. S., Al–Zamel, K., Allahverdi, A., Weighted Multi–Skill Resources Project Scheduling, *Journal of Software Engineering & Applications* (3), pp. 1125–1130, 2010.

[2] Błazewicz, J., Finke, G., Haupt, R., Schmidt G., New trends in machine scheduling, *European Journal of Operational Research* (37), pp. 303–317, 1998.

[3] Hegazy, T., Shabeeb, A. K., Elbeltagi, E., Algorithm for Scheduling with Multiskilled Constrained Resources, *Journal of Construction Engineering and Management* (126/6), pp. 414–421, 2000.

[4] Hindi, K. S., Yang, H., Fleszar, K., An Evolutionary Algorithm for Resource-Constrained Project Scheduling, *IEEE Transactions on Evolutionary Computation* (6/5), pp. 512–518, 2002.

[5] Khazandi, M., Soufipour, R., Rostami, M., A new improved genetic algorithm approach and a competitive heuristic method for large–scale multiple resource-constrained project–scheduling problems, *International Journal of Industrial Engineering Computations* (2), pp. 737-748, 2011.

[6] Kim, K. W., Gen, M., Yamazaki, G., Hybrid genetic algorithm with fuzzy logic for resource–constrained project scheduling, *Applied Soft Computing* (2/3F), pp. 174-188, 2003.

[7] Kolisch, R., Serial and parallel resource–constrained project scheduling methods revisited: Theory and computation, *European Journal of Operational Research* (90), pp. 320–333, 1996.

[8] Kolisch, R., Sprecher, A., PSPLIB – A project scheduling problem library, *European Journal of Operational Research* (96), pp. 205–216, 1996.

[9] Mendes, J. J. M., Goncalves, J. F., Resende, M. G. C., A random key based genetic algorithm for the resource constrained project scheduling problem, *Computers & Operations Research* (36), pp. 92-109, 2009.

[10] Santos, M. A., Tereso, A. P., On the Multi–Mode, Multi–Skill Resource–Constrained Project Scheduling Problem - computational results, *1st ICOPEV*, pp. 93-99, 2011.

[11] Tseng, L. Y., Chen, S. C., A hybrid metaheuristic for the resource–constrained project scheduling problem, *European Journal of Operational Research* (175), pp. 707-721, 2006.

[12] Valls, V., Ballestin, F., Quintanilla, S., A hybrid genetic algorithm for the resource-constrained project scheduling problem, *European Journal of Operational Research* (185), pp. 495-508, 2008.