An estimation of distribution algorithm for resource-constrained project scheduling problem

Chen Fang, Ling Wang, Ye Xu

TNList, Department of Automation, Tsinghua University, Beijing, 100084
E-mail: {fangchen06@mails.tsinghua.edu.cn; wangling@tsinghua.edu.cn; xuye05@mails.tsinghua.edu.cn}

Abstract: An estimation of distribution algorithm (EDA) is proposed to solve resource-constrained project scheduling problem (RCPSP). In the EDA, individual is encoded based on the extended active list, and a probability model of the distribution for each activity in a project and its updating mechanism are proposed. The algorithm determines the initial probability matrix according to an initial set of solutions generated by the regret-based sampling method and priority rule, and decodes the individuals by using serial schedule generation scheme. Meanwhile, a permutation based local search method is incorporated into the algorithm to enhance the exploitation ability so as to further improve the searching quality. Simulation results based on benchmarks and comparisons with some existing algorithms demonstrate the feasibility and effectiveness of our proposed EDA.

Key Words: Estimation of distribution algorithm, Resource-constrained project scheduling, Probability model

1 INTRODUCTION

The resource-constrained project scheduling problem (RCPSP) is concerned with the scheduling of the project activities over time and resources [1]. So far, research on RCPSP has extended to a variety of engineering fields, such as manufacturing modeling, software development, railway scheduling, logistics, and so on [2].

The RCPSP is NP-hard, so classic mathematic methods can hardly solve the problems with large scale effectively. During the past decade, study on RCPSP with the intelligent optimization algorithms has gain increasing attention and rapid development. In [3], a permutation based genetic algorithm (GA) was proposed, which adopted regret-based sampling method and priority rule to produce initial population. In [4], an active list based GA was presented where a gene is used to decide either forward or backward schedule generation scheme (SGS) was used. In [5], an adaptive GA was proposed, which employed a gene to decide either parallel SGS or serial SGS was used to evaluate the individuals. In [6], an active list based simulated annealing (SA) was proposed, where serial SGS was used to generate schedule and insert operator was employed as local search strategy. In [7], a random key based SA was introduced, where some activities were delayed on purpose to expand search space. In [8], a global shift operator-based SA was proposed, which adopted multiple cooling chains with different initial solution. In [9], a forward-backward Tabu search (TS) was proposed to solve time-varying RCPSP, where active list and serial SGS were adopted. In [10], the abandoned solution was inserted based on a flow network model. In [11], TS was presented to solve RCPSP with schedule-dependent setup times, where a pre-processing procedure was used to delete all inefficient modes, non-executable modes, and incapable locations. In [12], both permutation-based particle swarm optimization (PSO) and priority-based PSO were proposed. In [13], a combinatorial PSO algorithm was developed to solve multi-mode RCPSP, where a swap operator was used as local search and a penalty function was employed to estimate the infeasibility level of a particle solution.

Estimation of distribution algorithm (EDA) is a novel kind of evolutionary algorithms [14]. Different from GA, EDA generates new individuals according to a probability model but not with genetic operators, and good individuals are selected to update the probability model at each generation. As a newly proposed stochastic optimization algorithm, nowadays EDA has become a hot topic in field of evolutionary computation based optimization [15-18]. We refer [14] for more details about EDA. However, to the best of our knowledge, there has no EDA based research for RCPSP so far. Thus, we aim at developing an effective EDA based algorithm for solving RSPSP in this paper.

The remainder of the paper is organized as follows: In Section 2, RCPSP is described; in section 3, the EDA is proposed for RCPSP in detail; computational results and comparisons are listed in Section 4; and finally we end the paper with some conclusions in Section 5.

2 RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEM

The RCPSP is to study the reasonable utilization of resource and the scheduling of project activities to optimize certain objective function. The RCPSP is of wide engineering background. The well-known job-shop, flow-shop and open-shop scheduling problems can be regarded as the special cases of the RCPSP.

The classic RCPSP can be stated as follows. A project consists J activities that are labeled as $j = 1,...,J$, where the duration of activity j is denoted by d_j . There are

This research is partially supported by NSFC (60774082, 70871065 and 60834004) and Program for New Century Excellent Talents in University (NCET).

precedence relations between some activities of project. The precedence relations are given by sets of immediate predecessors P_j indicating that an activity j may not be started before each of its predecessors $i \in P_j$ is completed. The set of renewable resources is referred as K^{ρ}. For each resource $k \in K^{\rho}$ the per-period-availability is assumed to be constant that is denoted by R_k^{ρ}. Activity j requires r_{jk} units of resource k in each period of its non-preemptable duration d_j. The activities $j = 0$ and $j = J + 1$ are dummy activities, which represent the start and end of the project respectively. It assumes that the dummy activities do not request any resource and their durations are equal to zero. The set of all activities including the dummy activities is denoted as $J^+ = \{0, \ldots, J+1\}$. The common used objective for the RCPSP is to minimize the makespan of the project, which is also considered in this paper.

The mathematic model for the RCPSP can be formulated as follows:

$$\min \sum_{t=EFT_{J+1}}^{LFT_{J+1}} t \cdot x_{J+1,t} \tag{1}$$

subject to

$$\sum_{t=EFT_j}^{LFT_j} x_{jt} = 1, \ j \in J^+ \tag{2}$$

$$\sum_{t=EFT_h}^{LFT_h} t \cdot x_{ht} \le \sum_{t=EFT_j}^{LFT_j} (t - d_j) x_{jt}, \ j \in J^+; \ h \in P_j \tag{3}$$

$$\sum_{j=1}^{J} \sum_{b=\max\{t, EFT_j\}}^{\min\{t+d_j-1, LFT_j\}} r_{jk} x_{jb} \le NR_k, \ \ k \in R_k^{\rho} \tag{4}$$

$$x_{jt} = \begin{cases} 1, \text{if activity } j \text{ finishes at time } t. \\ 0, \text{otherwise.} \end{cases}, \ j \in J^+ \tag{5}$$

where LFT_j is the latest finish time of activity j, EFT_j is t he earliest finish time of activity j.

3 EDA FOR RCPSP

In this section, the EDA for RCPSP is proposed after introducing solution representation, construction of the probability model, local search strategy, and learning based updating mechanism.

3.1 Individual representation and schedule generation scheme

Ideally, an individual representation should meet the following five conditions [19]: 1) The transformation between solutions should be computationally fast; 2) For each solution in the original space, there is a solution in the encoded space; 3) Each encoded solution corresponds to one feasible solution in the original space; 4) All solutions in the original space should be represented by the same number of encoded solutions; 5) Small changes in the encoded solution should result in small changes in the solution itself.

An extended activity list (EAL) is proposed in this paper, which contains three parts: an activity list (AL), a list with the start time of every activity $(st_0, st_1, st_2, \cdots, st_{n+1})$ and a list with the finish time of every activity $(ft_0, ft_1, ft_2, \cdots, ft_{n+1})$. It concluded from experimental tests that procedures based on AL representations outperform the other procedures [20]. A list with the start (finish) time of every activity will be helpful when optimizing backward (forward) scheduling. The EDA does not operate on a schedule but on the EAL representation of a schedule. This character makes the algorithm convenient and effective for solving RCPSP.

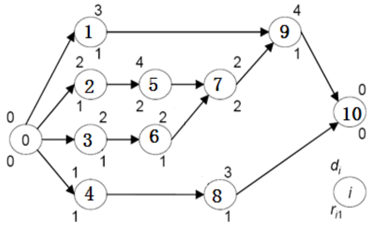
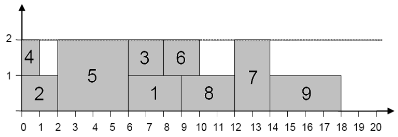


Fig. 1. An example of RCPSP

With the above encoding mechanism, a schedule generation scheme (SGS) should be used to transform the representation to a schedule. When the EAL is changed (e.g. performing crossover), the three parts of EAL need to be synchronized with of SGS. When a new EAL is generated, SGS is used to evaluate the EAL. With the help of SGS, the EAL can satisfy the five conditions mentioned above easily.



Part one	2	4	5	1	3	6	8	7	9
Part two	0	0	2	6	6	8	9	12	14
Part three	2	1	6	9	8	10	12	14	18

Fig. 2. A schedule and its EAL

Consider a simple RCPSP illustrated in Fig. 1, a feasible schedule and its EAL represent are depicted in Fig. 2.

As for the SGS, there are two kinds in the literature, including serial SGS and parallel SGS. Contrary to the serial SGS, parallel SGS is sometimes unable to obtain an optimal schedule [21]. Thus, in our procedure the serial SGS is adopted.

3.2 Probability model

Different from GA that produces offspring through crossover and mutation operators, EDA does it by sampling according to a probability model. How to construct the probability model is the main issue when designing EDA, and the model has a great effect on the performances of EDA [22].

In this paper, the probability model is a following $n \times n$ probability matrix.

$$Prob(t)=\begin{pmatrix} prob_{11} & \cdots & prob_{1J} \\ \vdots & \ddots & \vdots \\ prob_{J1} & \cdots & prob_{JJ} \end{pmatrix} \quad (6)$$

where the element $prob_{ji}$ represents the probability that the activity *j* is put at place *i* of the EAL at generation *t*.

In other words, $prob_{ji}$ is an index to indicate how good it seems to put activity *j* at the place *i*. At every position *i*, the selection probability of activity *j*, i.e. p_{ji}, is calculated according to probability matrix *Prob*(*t*) over the set of eligible activities, that is

$$p_{ji}=\frac{prob_{ji}}{\sum_{h=1}^{J} prob_{hi}} \quad (7)$$

If activity *j* has already been placed in some position, the whole line $prob_{j1}, prob_{j2}, ..., prob_{jJ}$ of probabilistic matrix *Prob* is set zero.

3.3 Initial population

In our EDA, a regret-based biased random sample method [23] with priority rule is used to generate initial population. In particular, starting with an empty activity sequence, we obtain a job sequence with feasible precedence by repeatedly applying the following step: The next activity in the activity sequence is taken from the set of those currently unselected activities, all non-dummy predecessors of which have already been selected for the job sequence. We employ latest finish time (LFT) priority rule [24] to derive the probabilities that are used to select the next activity for the activity sequence at each decision stage.

Let η_j be the probability of choosing activity *j* that has a priority value v_j. The value of η_j is calculated as follow:

$$r_j=\max_{i\in D} v_i - v_j \quad (8)$$

$$\eta_j=\frac{(r_j+\varepsilon)^{\alpha}}{\sum_{i\in D}(r_j+\varepsilon)^{\alpha}} \quad (9)$$

LFT priority rule is adopted in our procedure, which has been validated to generate good schedules for RCPSP. The priority value v_j is calculated for the eligible activities according to the following equation:

$$v_j=\max_{i\in D} LFT_i - LFT_j + 1 \quad (10)$$

where, LFT_j is the latest finish time of activity *j*, *D* in equation (8)~(10) is the set of eligible activities ready to be scheduled.

Kolisch found that the regret-based biased random sample method was of good performance when $\varepsilon=\alpha=1$ [25]. Accordingly, we also choose $\varepsilon=\alpha=1$ in our EDA.

3.4 Local search strategy

To enhance the exploitation ability, a permutation-based local search strategy PBLS controlled by a threshold *Pper* is proposed to explore the neighborhood of an individual. This operator is a variation of SWAP operator, but the PBLS does not break the feasibility of the EAL if the EAL is feasible before performing local search. The procedure of PBLS is described as follows:

```
Procedure PBLS
For (i=1,2,…,n)
{
   Randomly generate q where 0<q<1;
   If (q<Pper)
   {
      If(EAL(i) is not the predecessor for EAL(i+1))
      {
         Exchange EAL(i) and EAL(i+1);
         Evaluation the new EAL;
      }
   }
}
```

Fig. 3. Procedure of PBLS

3.5 Forward-backward iteration

Forward-Backward Iteration (FBI) [26] is an effective technique for RCPSP. Basically, the procedure iteratively employs SGS forward and backward scheduling until no further improvement in the makespan of project can be found. The activity finish times (part 3 of EAL) of a forward schedule determine the activity priorities for the next backward schedule. Similarly, the activity start times (part 2 of EAL) of a backward schedule determine the activity priorities for the forward schedule. FBI can be easily incorporated into many algorithms for RCPSP to improve the solution quality. In Fig. 4, a single iteration step is used to illustrate the procedure of FBI.

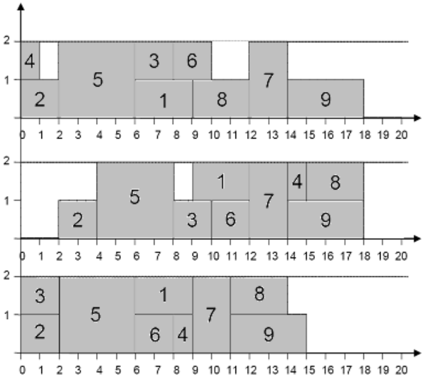


Fig. 4. A single iteration step of FBI

To reduce the makespan of Fig. 2 FBI is used by shifting each activity to the right as much as possible, in decreasing order of activity end times. For example, activity 9 and activity 7 cannot be scheduled later. Activity 8 can be right shifted to start at time 15. Activity 6 can be shifted two time units and start at time 10. Since the right shift of activity 8 has made some additional resources available, activity 1 can be shifted three time units to start at time 9. Activities 2, activity 3 and activity 5 can be shifted two time units. Finally, activity 4 is shifted to time 14. In this way, we obtain a schedule with a makespan of 16 units. Further improvements of the schedule are possible by shifting activities as much as possible to the left. This process reduces the makespan by one further time unit, as illustrated at the bottom of Fig. 4.

3.6 Updating mechanism

In this paper, a population based updating mechanism is adopted to refresh the probabilistic matrix *Prob*. First, a number of *Pop* individuals are created according the matrix *Prob*, and FBI is employed to renew every individual. Then the best *N* individuals are chosen to update the probabilistic matrix *Prob* according to the following equation:

$$prob_{ji}(t+1)=(1-\beta)\cdot prob_{ji}(t)+\frac{\beta}{N}\sum_{k=1}^{N}I_{ji}^{k},(i\geq 1;j\leq J) \tag{11}$$

where β is the learning speed and I_{ji}^{k} is the indicator function. For the *k*th individual, if activity *j* is placed at position *i*, $I_{ji}^{k}=1$, else $I_{ji}^{k}=0$.

3.7 Procedure of EDA

With the above design, the procedure of EDA is summarized as follows.

```
Procedure EDA
Step 1. Generate an Initial population {$\pi_1,\ldots\pi_{Init\_pop}$}
        with regret-based sampling method and LFT.
Step 2. Apply FBI to {$\pi_1,\ldots\pi_{Init\_pop}$}.
Step 3. Initialize probability matrix $Prob$ with
        {$\pi_1,\ldots\pi_{Init\_pop}$}, and then calculate all $p_{ji}$.
Step 4. Generate a new population {$\pi_1,\ldots\pi_{pop}$}
        according to $p_{ji}$.
Step 5. Apply FBI to {$\pi_1,\ldots\pi_{pop}$}.
Step 6. Apply PBLS to the best $N$ individuals of
        {$\pi_1,\ldots\pi_{pop}$}.
Step 7. Update $Prob$ with the best $N$ individuals, and
        then calculate all new $p_{ji}$.
Step 8. If the stopping condition is not met, go to Step 4.
```

Fig. 5. Procedure of EDA

4 COMPUTATIONAL EXPERIMENTS

We code the procedure in Visual C++ 2005 on IBM Thinkpad T61 with a Core 2 T7500 2.2GHz processor and use the well-known data sets PSPLIB for testing. This data set contains subsets J30, J60, J90, and J120 with problem instances of 30, 60, 90, and 120 activities, respectively. In this paper, we use J30, J60, and J120 to test the performances of EDA and to compare with some existing algorithms in literature.

4.1 Parameter Settings

As for the parameter setting, we use the orthogonal method of design of experiment (DOE) to determine a set of suitable parameters for the proposed EDA. Since the J120 contains the hardest problems, we choose 60 instances from J120 to carry out the DOE test. Combinations of the parameter values in Table 1 are evaluated.

Due to the limitation of page requirement, we omit the details of the DOE. The best combination of parameters is as follow: the number of initial population *Init_pop* is fixed at 200; the population size of each generation *Pop* is set equal to 50; the number of individuals used to update the probability matrix *N* is fixed at 5 (10%*Pop*); the learning speed β is set equal to 0.1; the local search accept rate *Pper* is fixed at 0.8.

Table 1. Combinations of parameter values

	1	2	3	4
Init_pop	50	100	150	200
Pop	10	20	50	100
N	10%Pop	20%Pop	30%Pop	60%Pop
β	0.1	0.3	0.5	0.8
Pper	0.1	0.3	0.5	0.8

4.2 Computational results of EDA

For our experiments, we set the maximum number of generated schedules (500, 1000 and 5000) as the stopping condition. Table 2 lists the results obtained by running our proposed EDA 20 independent times for each instance. The performances include the average, minimum and maximum percentage deviation from the lower bounds as well as the variance of deviation. As for the lower bound, we use the theoretically optimal values for set J30 and employ the critical-path based lower bound for set J60 and set J120.

From Table 2, it can be seen that the derivation values increase as the size of the problem increases. It shows that RCPSP with large scale is more difficult to solve since the problem is NP-hard. For each set of problems, the derivation values decrease as the maximum number of generated schedules increases. It shows that more schedules are explored better results can be obtained. For J30, the results obtained by the proposed EDA are much close to the optimal bounds, so the EDA is very effective for RCPSP with small scale. In the next sub-section, we will compare our EDA with some existing algorithms to further show the effectiveness of our EDA. In addition, the variance values of deviation for all the sets are very small, which means that our EDA is very robust.

Table 2. Results overview of proposed EDA

Problem set	Number of schedules	Ave. LB Dev.(%)	Min. LB Dev.(%)	Max. LB Dev.(%)	Var.
J30	500	0.48	0.22	0.86	2.17E-05
	1000	0.44	0.22	0.79	1.86E-05
	5000	0.32	0.22	0.49	7.94E-06
J60	500	11.92	11.27	12.45	5.15E-05
	1000	11.88	11.27	12.35	4.81E-05
	5000	11.70	11.21	12.12	3.19E-05
J120	500	34.71	33.08	35.87	9.40E-05
	1000	34.61	33.00	35.77	9.32E-05
	5000	34.28	32.90	35.30	8.16E-05

4.3 Comparisons of EDA with existing algorithms

Next, we compare our EDA with some existing algorithms based on the PSPLIB data sets. Tables 3, Table 4, and Table 5 list the results of EDA and other algorithms in literature for data sets J30, J60, J120, respectively. Same as before, the average deviation from the optimal solution is used as a measure of quality for J30 and the average deviation from the critical path based lower bound for J60 and J120.

Table 3. Average deviations (%) for J30

Algorithm	Maximum No. of schedules	
	1000	5000
Kochetov and Stolyar (2003) [27]	0.10	0.04
Debels and Vanhoucke (2007) [28]	0.12	0.04
Alcaraz et al. (2004) [29]	0.25	0.06
Valls et al. (2002) [30]	0.27	0.06
Debels et al. (2006) [31]	0.27	0.11
Nonobe and Ibaraki (2002) [32]	0.46	0.16
Hartmann (2002) [5]	0.38	0.22
Bouleimen and Lecocq (2003) [8]	0.38	0.23
Our EDA	**0.44**	**0.32**
Coelho and Tavares (2003) [33]	0.74	0.33
Schirmer (2000) [34]	0.65	0.44
Kolisch and Drexl (1996) [35]	0.74	0.52
Hartmann (1998) [3]	1.03	0.56
Kolisch (1995) [36]	1.44	1.00
Leon et al. (1995) [23]	2.08	1.59

Table 4. Average deviations (%) for J60

Algorithm	Maximum No. of schedules	
	1000	5000
Debels and Vanhoucke (2007) [28]	11.31	10.95
Valls et al. (2002) [30]	11.56	11.10
Debels et al. (2006) [31]	11.73	11.10
Kochetov and Stolyar (2003) [27]	11.71	11.17
Alcaraz et al.(2004) [29]	11.89	11.19
Our EDA	**11.88**	**11.70**
Hartmann (2002) [5]	12.21	11.70
Bouleimen and Lecocq (2003) [8]	12.75	11.90
Nonobe and Ibaraki (2002) [32]	12.97	12.18
Schirmer (2000) [34]	12.94	12.58
Hartmann (1998) [3]	13.30	12.74
Kolisch and Drexl (1996) [35]	13.51	13.06
Coelho and Tavares (2003) [33]	13.80	13.31
Leon et al.(1995) [23]	14.33	13.49
Kolisch (1995) [36]	14.89	14.30

From the these tables, it can be seen that among the 15 algorithms with 5000 generated schedules our EDA is the ninth best for J30 and is the sixth best for both J60 and J120. If only 1000 schedules are generated, our EDA procedure is the eighth, fifth, and third best heuristic among all the fifteen algorithms for J30, J60, and J120 respectively. The gaps between our EDA and the best algorithm for each set are very small. For the problem set J30, the gaps are 0.34% (1000 schedules) and 0.28% (5000 schedules); for J60, the gaps are 0.57% (1000 schedules) and 0.75% (5000 schedules); for J120, the gaps are 1.06% (1000 schedules) and 2.10% (5000 schedules). So, it can be concluded that our EDA competitive to those well-performed existing algorithms, especially for large-scale problems. All in all, the EDA proposed in this paper is feasible, effective and robust for the RCPSP.

Table 5. Average deviations (%) for J120

Algorithm	Maximum No. of schedules	
	1000	5000
Debels and Vanhoucke (2007) [28]	33.55	32.18
Valls et al. (2002) [30]	34.07	32.54
Debels et al. (2006) [31]	35.22	33.10
Kochetov and Stolyar (2003) [27]	34.74	33.36
Alcaraz et al. (2004) [29]	36.53	33.91
Our EDA	**34.61**	**34.28**
Hartmann (2002) [5]	37.19	35.39
Bouleimen and Lecocq (2003) [8]	42.81	37.68
Nonobe and Ibaraki (2002) [32]	40.86	37.88
Hartmann (1998) [3]	39.93	38.49
Schirmer (2000) [34]	39.85	38.70
Kolisch and Drexl (1996) [35]	41.37	40.45
Coelho and Tavares (2003) [33]	41.36	40.46
Leon, et al. (1995) [23]	42.91	40.69
Kolisch (1995) [36]	44.46	43.05

5 CONCLUSION

This was the first reported work about estimation distribution algorithm for solving the resource-constrained project scheduling problems. We encoded the individual of the problem based on extended active list, constructed the initial population by using a regret based sampling method and LFT priority rule, decoded the individual by using serial SGS, and applied a permutation based local search for exploitation. According to the DOE method, we determined a set of suitable values for parameter setting. Simulation results based on the PSPLIB benchmarks J30, J60, and J120 and comparisons with some existing algorithms demonstrated the effectiveness of the proposed EDA. The further work is to study the more suitable probability model of EDA and to develop adaptive EDA with learning mechanism and hybrid EDA with problem-dependent local search for RCPSP.

REFERENCES

[1] S. Hartmann, Project Scheduling under Limited Resources: Models, Methods, and Applications, Springer, Berlin, 1999.

[2] J. Weglarz, Project Scheduling, Recent Models, Algorithms and Applications, Kluwer, Dordrecht, 1998

[3] S. Hartmann, A competitive genetic algorithm for resource-constrained project scheduling, Naval Research Logistics, Vol.45, No.7, 733-750, 1998.

[4] J. Alcaraz, C. Maroto, A robust genetic algorithm for resource allocation in project scheduling, Annals of Operations Research, Vol.102, No.1, 83-109, 2001.

[5] S. Hartmann, A self-adapting genetic algorithm for project scheduling under resource constraints, Naval Research Logistics, Vol.49, No.5, 433-448, 2002.

[6] F. F. Boctor, A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes, European J of Operational Research, Vol.90, No.2, 349-361, 1996.

[7] J. H. Cho, Y. D. Kim, A simulated annealing algorithm for resource constrained project scheduling problems, J of the Operational Research Society, Vol.48, No.7, 736-744, 1997.

[8] K. Bouleimen, H. Lecocq, A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version, European J of Operational Research, Vol.149, No.2, 268-281, 2003.

[9] R. Klein, Project scheduling with time-varying resource constraints, International J of Production Research, Vol.38, No.16, 3937-3952, 2000.

[10] C. Artigues, P. Michelon, S. Reusser, Insertion techniques for static and dynamic resource-constrained project scheduling, European J of Operational Research, Vol.149, No.2, 249-267, 2003.

[11] M. Mika, G Waligóra, J Weglarz, Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup time, European J of Operational Research, Vol.187, No.3, 1238-1250, 2008.

[12] H. Zhang, X. Li, H. Li, Particle swarm optimization-based schemes for resource-constrained project scheduling, Automation in Construction, Vol.14, No.3, 393-404, 2005.

[13] B. Jarboui, N. Damak, P. Siarry, A combinatorial particle swarm optimization for solving multi-mode resource -constrained project scheduling problems, Applied Mathematics Computation, Vol.195, No.1, 299-308, 2008.

[14] P. Larraanaga, J. A. Lozano, Estimation of Distribution Algorithms, a New Tool for Evolutionary Computation, Kluwer, Boston, 2002.

[15] M. Hohfeld, G. Rudolph, Towards a theory of population based incremental learning, In: Proceedings of the 4th IEEE International Conference on Evolutionary Computation, 1-5, 1997.

[16] Q. Zhang, On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm, IEEE Transactions on Evolutionary Computation, Vol.8, No.1, 80-93, 2004.

[17] B. Jarboui, M. Eddaly, et al, An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems, Computer & Operations Research, Vol.36, No.9, 2638-2646, 2009.

[18] U. Aickelin, E. K. Burke, J. Li, An estimation of distribution algorithm with intelligent local search for rule-based nurse rostering, J of the Operational Research Society, Vol.58, No.12, 1574-1585, 2007.

[19] C. C. Palmer, A. Kershenbaum, Representing trees in genetic algorithms, In: Proceedings of the first IEEE International conference on Evolutionary Computation, 376–384, 1994.

[20] R. Kolisch, S. Hartmann, Heuristic algorithms for solving the resource-constrained project scheduling problem: classification. computational analysis, In: Weglarz J (ed) Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Berlin, 147–178, 1999.

[21] R. Kolisch, Serial and parallel resource-constrained project scheduling methods revisited: theory and computation, European J of Operational Research, Vol. 90, No.2, 320-333, 1996.

[22] J. A. Lozano, P. Larranaga, I. Inza, E. Bengoetxea, Towards a New Evolutionary Computation Advances on Estimation of Distribution Algorithms, Springer, Berlin, 2006.

[23] V. J. Leon, B. Ramamoorthy, Strength and adaptability of problem space based neighbourhoods for resource-constrained scheduling. OR Spektrum, Vol.17, No.2-3, 173-182, 1995.

[24] E. W. Davis, J. H. Patterson, A comparison of heuristic and optimum solutions in resource-constrained project scheduling, Management Science, Vol.21, No.8, 944-955, 1975.

[25] R. Kolisch, Project scheduling under resource constraints: efficient heuristics for several problem classes, Springer, Heidelberg, 1995.

[26] K. Y. Li, R. J. Willis, An iterative scheduling technique for resource constrained project scheduling. European J of Operational Research, Vol.56, No.3, 370–379, 1992.

[27] Y. Kochetov, A. Stolyar, Evolutionary local search with variable neighborhood for the resource constrained project scheduling problem, in: Proceedings of the 3rd International Workshop of Computer Science and Information Technologies, Russia, 2003.

[28] D. Debels, M. Vanhoucke, A decomposition-based genetic algorithm for the resource-constrained project scheduling problem, Operations Research, Vol.55, No.3, 457-469, 2007.

[29] J. Alcaraz, C. Maroto, R. Ruiz, Improving the performance of genetic algorithms for the RCPS problem, in: Proceedings of the 9th International Workshop on Project Management and Scheduling, 40-43, 2004.

[30] V. Valls, F. Ballestín, S. Quintanilla, A population-based approach to the resource-constrained project scheduling problem, Annals of Operations Research, Vol.131, No.1-4, 305-324.

[31] D. Debels, B. De. Reyck, R. Leus, M. Vanhoucke, A hybrid scatter-search/electromagnetism meta-heuristic for the resource-constrained project scheduling problem, European J of Operational Research, Vol.169, No.3, 638-653, 2006.

[32] K. Nonobe, T. Ibaraki, Formulation and tabu search algorithm for the resource constrained project scheduling problem (RCPSP), C. C. Ribeiro, P. Hansen, eds, Essays and surveys in Meta-heuristics, Kluwer, Boston, 557-588, 2002.

[33] J. Coelho, L. Tavares, Comparative analysis of meta-heuristics for the resource constrained project scheduling problem, Technical report, Department of Civil Engineering, Instituto Superior Tecnico, Lisbon, 2003.

[34] A. Schirmer, Case-based reasoning and improved adaptive search for project scheduling, Naval Research Logistics, Vol.47, No.3, 201-222, 2000.

[35] R. Kolisch, A. Drexl, A adaptive search for solving hard project scheduling problems, Naval Research Logistics, Vol.43, No.1, 23-40, 1996.

[36] R. Kolisch, Project scheduling under resource constraints-efficient heuristics for several problem classes, Physica-Verlag, Heidelberg, Germany, 1995.