UDC 519.8

EVOLUTIONARY FRAGMENTARY ALGORITHM FOR PERMUTATION FLOW SHOP PROBLEM

© Bondarenko O.S., Kozin I.V.

Zaporizhzhya National University
The Department of Economics
The Chair of Economic Cybernetics
Zhukovsky str., 66, Zaporizhzhya, 69600, Ukraine
e-mail: *buenasdiaz@gmail.com*

Abstract. The article tackles the strongly $\mathscr{NP}$-hard permutation flow shop problem. The fragmentary structure of the problem is pointed out. The evolutionary fragmentary approach for optimal solution is proposed. The testing of evolutionary fragmentary algorithm on instances' set from the ORLib [1] library is conducted.

Introduction

In 1954 Selmer Martin Johnson proposed an algorithm for finding optimal solution to a problem [2], which has been known since as flow shop problem.

Suppose that n jobs $J_i (i = 1, \ldots, n)$ are to be executed on m machines $M_j (j = 1, \ldots, m)$. Each job consists of m operations $O_{i1}, \ldots, O_{im}$. Each operation O_{ij} is associated with the processing time p_{ij}. Every job is processed on all machines in the order of the machine indexing. It means that the execution of the operation O_{ij+1} is not allowed until the execution of the operation O_{ij} is finished. By the schedule we mean the mapping from the set of the job numbers (i, j) to the completion times set C_{ij}. In the given problem, we are supposed to find such a schedule in which comletion time of the last job on the last machine is minimized. Preemption, processing more than one job on one machine at a time, and execution one job on more than one machine at a time are not allowed.

According to the notation [3] the problem under consideration is denoted $F||C_{max}$. The problem has $(n!)^m$ feasible solutions, hence, in the literature the easier case is studied, which admits $n!$ schedules with the same order of processing on all machines (permutation schedules) and is denoted $F|prmu|C_{max}$. The latter case is considered in the following.

The exact algorithm proposed by Johnson solves the problem to optimality for $m = 2$ case and for special case with $m = 3$. All the efforts to find exact polynomial algorithm for $m > 2$ were unsuccessfull. With the advent of the first results in the early 70-s [4, 5, 6] in computational complexity theory that fact had obtained an explanation. $\mathscr{NP}$-hardness in the strong sense was proved in [7]. Since then researchers' efforts have been focused on approximation and heuristic algorithms design.

Some time after that it has become clear that some problems are hard not only for solving to optimality but are hard for approximating. It was proved [8] for considered problem there is no $(1 + \epsilon)$-approximation algorithm for $1 + \epsilon < 5/4$. Furthermore, so far the problem whether there exist polynomial approximation scheme for $Fm||C_{max}$ where m

is fixed is open. Hence, the design of algorithms with no theoretical bounds and worst-case ratios, which solve complex practical problems, is now urgent. As such we consider stochastic local search [9] algorithms: evolutionary algorithms, taboo search, simulated annealing, ant colony systems, variable neighbourhood search.

The survey of local search algorithms for flow shop problem can be found in [10]. The comparative analysis of the most efficient local search methods from the literature, called also metaheuristics, was conducted in [11].

The goal of the paper is the inquiry of the new solution approach to combinatorial problems based on the combination of the evolutionary and the fragmentary approaches.

1. The Fragmentary Structure

Definition 1. Fragmentary structure [12] is tuple $(\mathscr{X}, \mathscr{Y}, \mathscr{R})$, where $\mathscr{X}$ – a finite set of the fragments, $\mathscr{Y}$ – a family of subsets of $\mathscr{X}$, $\mathscr{R}$ – a combining rule, i.e. the decision rule for recognizing whether a union of subsets of $\mathscr{X}$ is a feasible set of $\mathscr{Y}$.

$F||C_{max}$ can be considered as the fragmentary structure. By fragmments we will have any tuples of jobs, respectively by the elementary fragments we will mean jobs. The rule of combination – no job repetition in the combined tuples. Maximal by inclusion fragment will be a feasible solution the problem under consideration. So the feasible solution can be considered as certain fixed permutation of jobs.

In the papers [12, 13] it has been showed that the search of feasible solutions can considered in the fragmentary structure framework. For the costruction of feasible solutions the fragmentary algorithm $\mathscr{F}$ is applied. In the general case, the input of $\mathscr{F}$ algorithm are a tuple $I = (i_1, \ldots, i_n)$ and the empty solution set $\mathscr{S}$. The input is looked through by $\mathscr{F}$ in the machine indexing order. On every step in the tuple $\mathscr{I}$ the first element is looked for which the combining rule is fulfilled. The found elementary fragment is added into $\mathscr{S}$. The procedure is repeated until the maximal by inclusion fragment is built.

2. The Evolutionary Fragmentary Algorithm

The fragmentary algorithm allows to obtain only some feasible solutions for the optimization problem. For finding optimal solution of $F|prmu|C_{max}$ the combination evolutionary and fragmentary approaches is proposed. A few analogous approaches grounded on the evolutionary mechanism and performance evaluation of such approaches are considered in [14, 15].

The description of evolutionary fragmentary algorithm (EVF algorithm), explored in the work, is the following. The feasible solutions of the problem are presented by job number permutations, which form the set of all permutations $I_1, \ldots, I_{n!}$. Each permutation will be treated as chromosome. The search can be described as the following.

`Stage 1.` *Initialization.*

The initial population, consisting of N chromosomes, is generated at random.

`Stage 2.` *Selection.*

Select K pairs of parents from the current population for reproduction. The selection is random without replacement.

`Stage 3.` *Recombination.*

The recombination in the EVF algorithm is implemented by the following n-step procedure. All the pairs of chromosome-parents' I-tuples are being looked through. On every step the minimal element from the first two elements of I-tuples' pair is chosen and is inserted into tuple-offspring. The inserted element is deleted from both tuples-parents. For instance, in the case with $m = 5$ chromosome-parents [12543] and [53142] yield the chromosome-offspring [12534].

`Stage 4.` *Mutaion.*

The mutation is implemented by transposition of jobs' indexes from two randomly selected positions in chromosome.

`Stage 5.` *Replacement.*

The chromosomes-offsprings are added in the current population. The shrinking of the current population to predefiened size is provided by sequential deletion of chromosomes with maximal criterion's values, where the criterion's value is computed by the following recursive formula: $C_{i,j} = max\{C_{i-1,j}, C_{i,j-1}\} + p_{i,j}$

`Stage 6.` *Termination.*

If predefiened termination criterion is achieved, then stop the algorithm, otherwise go to the second stage. The termination criterion, used by authors, is the number of generations in evolution, i.e. the number of the search stages repetitions.

3. The Test Results

The necessity of test conduction is explained by the difficulties and, sometimes, by imposibility of theoretical estimation of stochastic local search algorithms performance derivaton in general, and of evolutionary algorithms, in particular.

For the EVF algorithm performance verification the test was being conducted on well-known in the Taillard's [16] problem set. The problem set comprises twelve groups consisting of ten instances each with the numbers of jobs from 20 to 500 and the number of machines from five to 20. It is the most used in the literature for $F|prmu|C_{max}$ testing. Thus, there appears an opportunity for the comparison of the proposed algorithm with already existing and tested ones on the given set.

EVF algorithm was tested against random search (RS) algorithm as well as NEH algorithm [17], which is curently considered the best performing deterministic heuristic.

All the three algorithms were implemented in Visual Basic for Applications and run on computer with Celeron procesor 1800 Mhz and 256 MB of the main memory. The termination criterion was chosen to be the number of generations. achieved by the algorithm during its run.

The algorithms' performance was measured by relative deviance (RD) from optimum (OPT) or the least known upper bound [18] of the value (A) found by given algorithm: $RD = \frac{A-OPT}{OPT} \cdot 100$.

In Table 1 each group of problems is denoted as $n \times m$, where n is the number of jobs, m is the number of machines. In it RD for each instance is averaged on the group size. For EVF algorithm in the table in the parentheses the number of generations during which it

Group of problems	RS	NEH	EVF
20×5	6.95	2.46	1.25 (300)
20×10	10.29	4.88	2.86 (300)
20×20	8.52	3.72	3.28 (300)
50×5	5.18	0.81	0.92 (300)
50×10	14.39	5.23	4.51 (300)
50×20	16.71	6.21	5.85 (300)
100×5	4.14	0.67	0.73 (500)
100×10	10.74	2.45	2.4 (500)
100×20	16.85	5.72	6.08 (500)
200×10	8.74	1.8	1.52 (1000)
200×20	15.66	4.66	4.95 (1000)
500×20	11.79	2.41	3.74 (1000)
Average	10.83	3.42	3.17

Таблица 1. The Test Results for $F|prmu|C_{max}$

was run is pointed out. From Table 1 it can be concluded that in the average *evolutionary fragmentary algorithm has showed the higher results than other two tested algorithms.*

Conclusion

The basic result of the paper is the new approach to scheduling problems solution. The fragmentary structure has been showed which allows the application of EVF algorithm. The testing of the algorithm on the standard problem set is fulfilled.

From the test results, represented in Table 1, it appears promising to apply the proposed aproach to scheduling problems solution.

Список литературы

1. *Beasley J. E.* ORLib - Operations Research Library. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/flowshop2.txt, 2008.
2. *Johnson S. M.* Optimal Two and Three-Stage Production Schedules with Setup Times Included // Naval Research Logistics Quarterly. – 1954. – Vol. 1, No. 1. – P. 61–68.
3. *Graham R. L.* Optimization and approximation in deterministic sequencing and scheduling: A survey / R. L. Graham, E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan // Discrete Optimization II / P.L. Hammer, E.L. Johnson and B.H. Korte (eds.). – Amsterdam : North-Holland, 1979. – (Annals of Discrete Mathematics ; Vol. 5.). – P. 287–326.
4. *Cook S. A.* The complexity of theorem-proving procedures / Stephen A. Cook // STOC '71: Proceedings of the third annual ACM symposium on Theory of computing. – New York : ACM, 1971. – P. 151–158.
5. *Karp R.* Reducibility among Combinatorial Problems // Complexity of Computer Computations / R. E. Miller and J. W. Thatcher (eds.). – New York : Plenum Press, 1972. – P. 85–104.
6. *Левин Л. А.* Универсальные задачи перебора / Л. А. Левин // Проблемы передачи информации. – 1973. – Т. 9, № 3. – С. 115-116.

7. *Garey M. R.* The Complexity of Flow Shop and Job Shop Scheduling / M. R. Garey, D. S. Johnson, R. Sethi // Mathematics of Operations Research. – 1976. – Vol. 1, No. 2. – P. 117–129.
8. *Williamson D. P.* Short shop schedules / D. P. Williamson, L. A. Hall, J. A. Hoogeveen, C. A. J. Hurkens, J. K. Lenstra, S. V. Sevastianov, D. B. Shmoys // Operations Research. – 1997. – Vol. 45, No. 2. – P. 288–294.
9. *Hoos H. H.* Stochastic Local Search : Foundations and Applications / Hoos H. H., Stuetzle T. – San Francisco : Morgan Kaufmann Publishers, 2004. – 660 p. – ISBN 1-55860-872-9.
10. *Ruiz R. A* comprehensive review and evaluation of permutation flowshop heuristics / R. Ruiz, C. Maroto // European Journal of Operational Research. – 2005. – Vol. 165. – P. 479–494.
11. *Ruiz R.* A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem / R. Ruiz, T. Stuetzle// European Journal of Operational Research. – 2007. – Vol. 177. – P. 2033–2049.
12. *Козин И. В.* Фрагментарные алгоритмы в системах поддержки принятия решений // Питання прикладної математики і математичного моделювання, збірник наукових праць. - Дніпропетровськ. – 2006. – С. 131–137.
13. *Козин И. В.* Фрагментарный алгоритм для задачи симметричного размещения // Радиоэлектроника, информатика, управление. – 2005. – № 1. – С. 76–83.
14. *Ruiz R.* Two new robust genetic algorithms for the flowshop scheduling problem / R. Ruiz, C. Maroto, J. Alcaraz // OMEGA, the International Journal of Management Science. – 2006. – Vol. 34. – P. 461–476.
15. *Nagano M. S.* A Constructive Genetic Algorithm for permutation flowshop scheduling / M. S. Nagano, R. Ruiz, L. A. N. Lorena // Computers & Industrial Engineering. – 2008. – Vol. 55. – P. 195–207.
16. *Taillard E.* Benchmarks for basic scheduling problems // European Journal of Operational Research. – 1993. – Vol. 64, No. 2. – P. 278-285.
17. *Nawaz M.* A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem / M. Nawaz, E. E. Enscore Jr., I. Ham // OMEGA, The International Journal of Management Science. – 1983. – Vol. 11, No. 1. – P. 91–95.
18. *Taillard E.* Summary of best known lower and upper bounds of Taillard's instances. http://ina2.eivd.ch/Collaborateurs/etd/problemes.dir/ordonnancement.dir/flowshop.dir/best_lb_-up.txt, 2008.

Статья поступила в редакцию 15.08.2009