Flow shop, multi-objective, genetic algorithm, makespan, average flow time

Jarosław PEMPERA, Czesław SMUTNICKI, Dominik ŻELAZNY*

MULTI-OBJECTIVE OPTIMIZATION OF PRODUCTION SCHEDULES

This paper deals with the production/manufacturing system having sequential structure of service stages (line), where various products flow along the line by the same technology routes. The task of seeking optimal production plan is known in the literature as the permutation flow shop scheduling problem. We refer to the multi-objective variant of this problem, the most appropriate from the practice point of view. In this paper we consider fundamentally bi-criteria case with two highly useful functions: the makespan and the mean flow time. In order to find an approximation of Pareto frontier, we proposed new effective algorithm based on Elitist Non-dominated Sorting Genetic Algorithm (NSGA). We developed also some new properties of the problem, which were used next to construct and modify the Elitist NSGA. Results of computational experiments were also provided.

1. INTRODUCTION

Production scheduling plays a key role in manufacturing systems of an enterprise for maintaining competitive position in the fast changing market, so developing effective and efficient advanced scheduling technologies is extremely important. So called flow shop scheduling problem represents a class of widely studied cases based on ideas derived from production engineering, which currently modeled nearly a quarter of manufacturing systems, assembly lines, information service facilities [2], and has earned a reputation for being hard (NP-hard) to solve [3]. Most of the currently used single objectives are easily adaptable to real world applications.

Since pioneer works in the middle of fifties of the previous age, the permutation Flow Shop Scheduling Problem (FSSP) has received considerable theoretical, computational, and empirical research work. Because of its complexity, branch and bound techniques and classical mathematical programming [4], which provide exact solution, are only applicable to small-scale instances. That is why a lot of various approximate solution methods have been proposed, including constructive heuristics,

* Nazwa instytucji, adres - pisać 10 punktów (pod linią długości 2,5 cm)

improvement metaheuristics, and hybrid algorithms. Multi-objective FSSP constitutes natural evolution of models and solution methods, oriented on practice. Actually, scheduling decisions usually has to take into account several economic indexes simultaneously. During the last decade a number of multi-objective algorithms have been suggested. Primarily because of their ability to find multiple Pareto-optimal solutions in single run. Since it is not possible to have a single solution simultaneously optimizing all objectives, algorithms that give solutions lying on or near the Pareto-optimal front are of great practical value.

2. PROBLEM DESCRIPTION

The flow shop problem consists of two main elements; a group of m machines from the set $M = \{1,2,\ldots,m\}$ and a set of n jobs from the set $J = \{1,2,\ldots,n\}$ to be processed on those machines. Each job in the sequence has to pass through each machine and the machine sequence is fixed, i.e. $1,2,\ldots,m$. Each job is processed only once on each machine. Each operation, once started, have to be performed till completion and must be completed before any other operation, which it has to precede. Thus job j, $j \in J$ consists of a sequence of m operations; each of them corresponding to the processing of job j on machine k during an uninterrupted processing time $p_{jk} > 0$.

Each machine k, $k \in$M can execute at most one job at a time, each job can be processed on at most one machine, and it is assumed that each machine processes the jobs in the same order. A feasible schedule is defined by completion times C_{jk}, $j \in J$, $k \in M$ of job j on machine k, such that the above constraints are satisfied. For the given processing order represented by permutation $\pi = (\pi(1),\ldots,\pi(n))$ on set J, the feasible schedule can be found by using the following recursive formula:

$$C_{\pi(j),k} = \max(C_{\pi(j-1),k}, C_{\pi(j),k-1}) + p_{\pi(j),k} \tag{1}$$

where $\pi(0)=0$, $C_{j,0}=0$, $j \in J$, $C_{0,k}=0$, $k \in M$.

The objective function consists of the average flow time and total completion time (makespan). These functions are calculated as follow:

– $C_{avg}(\pi) = \frac{1}{n}\sum_{j=1}^{n} C_{\pi(j),n}$ – average flow time,

– $C_{\max}(\pi) = \max_{1 \le s \le n} C_{\pi(s),m} = C_{\pi(n),m}$ – total completion time.

3. ELITIST MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS

Recent results [4] show clearly that elitism can speed up the performance of the genetic algorithms significantly. It also helps to prevent the loss of good solutions,

once they have been found. Among the existing elitist multi-objective evolutionary algorithms (MOEAs) one can find few promising approaches, namely: strength Pareto EA (SPEA) of Zitzler and Thiele's [5], Pareto-archived evolution strategy (PAES) of Knowles and Corne' s [2], and elitist GA of Rudolph [3].

In paper [1] Deb suggested an Elitist Non-dominated Sorting Genetic Algorithm. Based on the non-dominated sorting GA (NSGA), criticized for high computational complexity of non-dominated sorting, lack of elitism and need for specifying the sharing parameter, he modified the approach to alleviate those difficulties. By applying fast non-dominated sorting, density estimation and crowded comparison operator he allowed to lessen the computational complexity and guide the selection process of the algorithm towards a uniformly spread out Pareto-optimal front.

4. LOCAL SEARCH METHOD AND ELITISTS NSGA (LS NSGA-II)

Our research and general studies show that minimum values of the average flow time tend to grow larger with the increase of makespan. Furthermore, it is observed that good solutions can be found near other good solution for the flow shop scheduling problem with both optimization criteria. We use those features to construct some modification of NSGA II algorithm to improve its efficiency.

In every iteration of hybrid LS NSGA II algorithm, for each offspring, given number of iteration of stochastic Local Search (LS) algorithm is performed to enhance the offspring. There is searched neighborhood generated by adjacent interchange moves. New solution is generated by random swapping two adjacent jobs in current permutation. If it dominates the old solution, then it replaces the parent solution in next iteration. In other case it gets discarded. Although the computational complexity increases in this way, we were able to produce more non-dominated solutions.

In our implementation the individuals in population are represented by jobs permutation, values of their criteria functions, Pareto-rank and crowding distance. The LS NSGA-II algorithm uses PMX crossover and tournament selection, while fitness value is based on non-domination level and crowding distance.

5. COMPUTATONAL RESULTS

The algorithms NSGA-II and LS NSGA-II were coded in Personal 6.0 Builder C++, run on a PC with Intel Core 2 Duo 2.66 GHz processor and the WINDOWS XP operating system. The algorithms were tested on 5 benchmark instances of different size provided by Taillard [9], i.e. TA05, TA25, TA41, TA60.

In our tests, both algorithms are terminated after performing 10 000 of iterations on each instance. Number of iterations of LS in LS NSGA II was equal to the number of jobs. For each test instance, we collected the set of Pareto optimal solutions P^A,

A∈{NSGA-II, LS NSGA-II}. Next, we determined the set P^* consists of non-dominated solutions of both sets. Finally, for each algorithm A, we determined the number of solutions d(A) from P^A included in P^*. The number of non-dominated solutions of both algorithms as well as number of elements of every sets are shown in Table 1.

Tab. 1. Pareto-optimal solutions generated by algorithms

Instance	NSGA-II		LS NSGA-II		$\|P^*\|$
	d	$\|P\|$	d	$\|P\|$	
TA05	0	9	13	13	13
TA25	0	12	15	15	15
TA41	0	7	16	16	16
TA60	0	7	12	12	12

As we can see in Table 1, all solutions found by NSGA-II are dominated by solutions generated by proposed algorithm LS NSGA-II. Furthermore, the number of Paterto-optimal solutions found by LS NSGA-II is significantly greater than the corresponding number in classic NSGA-II. We observed that the number of Pareto-optimal ones is relatively small.

REFERENCES

[1] DEB K., PRATAP. A, AGARWAL, S., AND MEYARIVAN, T., *A fast and elitist multi-objective genetic algorithm: NSGA-II*. IEEE Transaction on Evolutionary Computation, 6(2), 181-197, 2002.

[2] M. PINEDO, *Scheduling: Theory, Algorithms and Systems, 2nd ed.* Englewood Cliffs, NJ: Prentice-Hall, 2002.

[3] GAREY M. R. AND JOHNSON D. S., Computers *and Intractability: A Guide to the Theory of NP-Completeness.* San Francisco, CA: Freeman, 1979.

[4] LAGEWEG B. J., LENSTRA J. K., AND RINNOOY KAN A. H. G., *A general bounding scheme for the permutation flow-shop problem,* Oper. Res., vol. 26, no. 1, pp. 53–67, 1978.

[5] KNOWLES, J. AND CORNE, D., *The Paretoarchi ved evolution strategy: A new baseline algorithm for multiobjective optimisation.* Proceedings of the 1999 Congress on Evolutionary Computation, Piscatway: New Jerse y: IEEE Service Center , 98–105, 1999.

[6] RUDOLPH, G., *Evolutionary search under partially ordered sets.* Technical Report No. CI-67/99, Dortmund:Department of Computer Science/LS11, University of Dortmund, Germany, 1999.

[7] ZITZLER, E., DEB, K., AND THIELE, L. (in press) *Comparison of multiobjective evolutionary algorithms: Empirical results*. Evolutionary Computation, 8.

[8] ZITZLER, E. AND THIELE, L., *Multiobjective optimization using evolutionary Algorithms - A comparati ve case study.* In Eiben, A. E., B¨ ack, T., Schoenauer , M., and Schwefel, H.-P., editors, Parallel Problem Solving from Nature, V, pages 292–301, Springer , Berlin, Germany, 1998.

[9] TAILLARD E., *Benchmarks for basic scheduling problems*. European Journal of Operational Research vol. 64, 1993, pp. 278-285.