A Modified Genetic Algorithm and Its Application to the Flowshop Sequencing Problem with Objective of Minimizing Mean Total Flowtime *

Tang Lixin

(Institutes of Systems Engineering, Northeastern University·Shenyang, 110006, P.R.China)

Liu Jiyin

(Department of Industrial Engineering and Engineering Management, The Hong Kong University of Science and Technology, Hong Kong)

Abstract: A modified genetic algorithm (MGA) framework was developed and applied to the flowshop sequencing problems with objective of minimizing mean total flowtime. To improve the general genetic algorithm routine, two operations were introduced into the framework. Firstly, the worst points were filtered off in each generation and replaced with the best individuals found in previous generations; Secondly, the most promising individual was selectively cultivating if a certain number of recent generations have not been improved yet. Under conditions of flowshop machine, the initial population generation and crossover function can also be improved when the MGA framework is implemented. Computational experiments with random samples show that the MGA is superior to general genetic algorithm in performance and comparable to special-purpose heuristic algorithms. The MGA framework can also be easily extended to other optimizations even though it will be implemented differently in detail.

Key words: modified genetic algorithm; flowshop scheduling; mean total flowtime

改进遗传算法及其在带有目标是最小平均总流程时间的流水调度排序中的应用

唐立新　　刘继印

(东北大学系统工程研究所·沈阳, 110006)　(香港科技大学工业工程与工程管理系·香港)

摘要: 提出了一个改进遗传算法的结构, 并且应用于带有目标是最小平均总流程时间的流水调度排序中. 为了改进一般遗传算法的程序, 两个新的操作被引进到这个操作中. 这两个操作为: 1) 过滤操作: 过滤掉在每一代中的最坏的个体, 用前一代中的最好的个体替代它; 2) 培育操作: 当在一定代数内算法不改进时, 选择一个培育操作用于培育最有希望的个体. 通过大量的随机产生的问题的例子的计算机实验显示出, 提出的算法的性能明显好于一般遗传算法, 并且和此问题的最好的专门意义的启发式算法相匹配. 新的 MGA 框架很容易扩展到其它最优化当中, 只是实施的详细的步骤有所不同.

关键词: 改进遗传算法; 流水调度问题; 平均总流程时间

1 Introduction

Genetic algorithms (GAs), based on the mechanism of natural genetics/selection, are used as an adoptive searching technique for optimization. Goldberg[1] has put forward a survey on practical applications of genetic algorithms. Although GA is considered as among the best performing general heuristic methods, its performance is yet not as good as special-purpose heuristic algorithm for many problems, especially when computation time is limited. Further improvements are needed to make it more cost (time) effective. This paper attempts to improve the GA procedure by introducing two additional steps: a filtering step and a cultivating step. To demonstrate effect of the modified genetic algorithm procedure, it is implemented for a flowshop sequencing problem with objective of minimizing mean total flowtime (completion times). For the scheduling problem to minimize total flowtime, heuristic algorithms have been proposed by Gupta[2], Miyazaki et al.[3], Ho and Chang[4] and Rajendran[5]. Among them, Rajendran's algorithm is most efficient so far.

2 The modified genetic algorithm (MGA)

* This Project is Supported by National Natural Science Foundation of China (79700006) and by National 863/CIMS of China (863-511-708-009) and The Hong Kong University of Science and Technology Direct Allocation (DAJ95/96.E08).
Manuscript received Jun.8, 1998, revised Nov.23, 1998.

2.1 The framework

When GA is used to solve an optimization problem, it is expected to converge quickly. On the other hand, we do not want to see pre-mature convergence (trapped in local optimum). It is often difficult for the general GA procedure to keep a good balance between the two (computational time and solution quality). A modified GA is proposed in this section for a better trade off between these two conflicting criteria.

The framework of the modified genetic algorithm is described in Fig. 1.

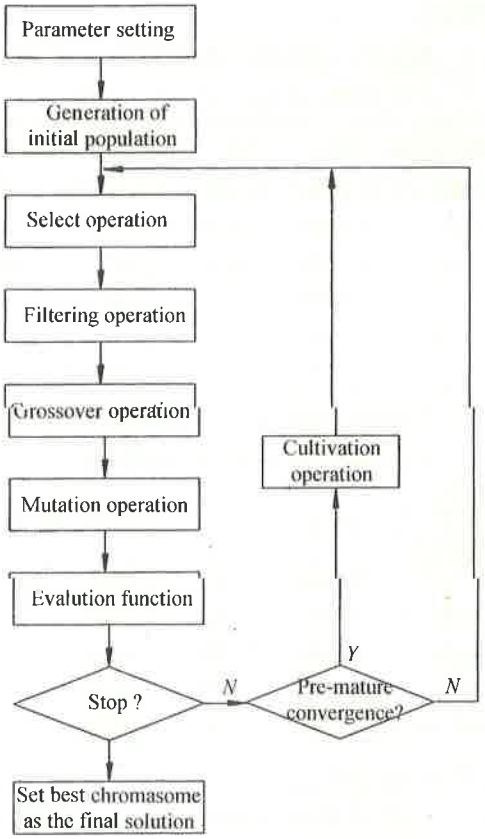

Fig. 1 Framework of the modified genetic algorithm

Two new types of operations called filtering and cultivating are introduced in this modified procedure. Details of these new operations are described below.

2.2 Filtering

The filtering operation is shown in Fig. 2. In the selection step of GA, solutions are selected with probability. Although the probabilities are so assigned that good solutions have better chance to be selected, there is no guarantee that the best solution will be selected. To increase the chance for the optimal solution being approached quickly, an additional step is introduced here to the GA procedure. In this step of each generation, the two' worst selected solutions are filtered out before genetic operations. Their positions are filled with the best solution of the current population and the best solution recorded so far.

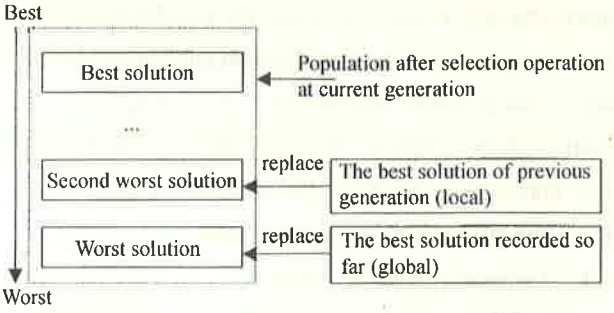

Fig. 2 Schematic diagram of the filtering operation

Among the reasons for adding this new step are 1) according to GA convergence theorem in the general sense, if the best solution will be held in each generation, then the GA converges to optimal solution when $N \to \infty$; 2) accrding to knowledge of genetic evolution, if an excellent individual is put into population, evolution of the population will be improved.

2.3 Cultivating

The cultivating operation is shown in Fig. 3. Computation shows that when a genetic algorithm keeps unimproved for a certain number (around 30) of generations, then it will be difficult for the solution quality to be improved if iteration continues. Therefore, we modify the GA procedure further by introducing another additional step (called cultivating operation).

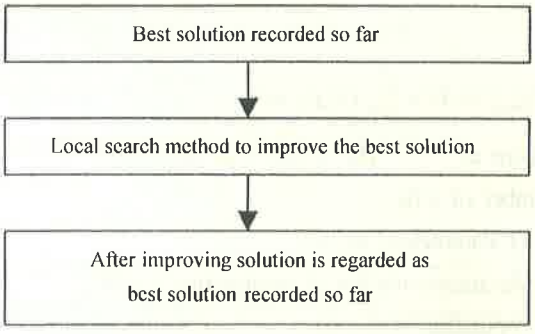

Fig. 3 Schematic diagram of the cultivating operation

After passing half of the maximum generation, when the number of iterations without improving the best solution is greater than a priori fixed constant, the best solution recorded so far is improved by using one step of local search. The improved solution will be put into next iteration.

From the filtering operation, we know that the solution

cultivated in this step will pass to the next generation. So it brings new feature to the population (just like the effect of mutation). Therefore, if the fact of no improvement for certain generations is due to pre-mature convergence, the new feature may make the population start improving again. We note that although cultivation and mutation both diversify chromosomes in the population, cultivation always improves the best solution while mutation may improve or deteriorate the solution.

3 Computation experiments

3.1 Implementing of MGA

An MGA is implemented for our sequencing problem based on the frame work described above. In this MGA, the representation of solutions, generation of initial solutions, fitness function, and mutation operation are all accomplished in the same way as in the GA of [1]. Steps of two new operations are added as described in the last section.

1) The objective formulation.

If we have processing time $T(i,j)$ and for job i on the machine j, for a job permutation $\{J_1, J_2, \cdots, J_n\}$ we calculate the makespan $C_{\max}$ and the mean total completion time C_{mean} as follows:

$$C(J_1,1) = T(J_1,1),$$

$$C(J_i,1) = C(J_{i-1},1) + T(J_i,1), i = 2,\cdots,n;$$

$$C(J_1,j) = C(J_1,j-1) + T(J_1,j), j = 2,\cdots,n;$$

$$C(J_i,j) = \max\{C(J_{i-1},j), C(J_i,j-1)\} + T(J_i,j), i = 2,\cdots,n; j = 2,\cdots,m.$$

$$C_{\text{mean}} = 1/n \sum_{i=1}^{n} C(J_i,m).$$

Where m —— The number of machines; n —— The number of jobs.

2) Parameters are set.

Maximum number of generations = 1000,

Population size = 80,

Probability of crossover = 0.99,

Probability of mutation = 0.12.

3.2 Computational results

To evaluate the performance of the MGA, computational experiments have been done on a large number of carefully selected representative problem instances. The two heuristic algorithms including Ho & Chang's algorithm, and Rajendran's algorithm[5] were used for comparison.

Two criteria are used for the comparison and evaluation —— optimality and computation time.

1) Optimality.

We used a relative optimality measurement C/C^*, where C^* is the best among the results for a problem instance given by the four algorithms and C is the result for the instance given by the algorithm being evaluated. The average optimality performances of these algorithms against different sizes of jobs and against different numbers of machines are shown in Table 1 and 2 respectively.

Table 1 Average optimality performances of algorithms with different sizes of jobs loading

Number of jobs	MGA	GA	Rajendran	Ho & Chang
50	1.0000	1.1245	1.0314	1.1547
75	1.0000	1.1445	1.0318	1.1643
100	1.0000	1.1499	1.0277	1.1761
125	1.0000	1.1601	1.0287	1.1924
150	1.0000	1.1616	1.0243	1.1870
Overall average	1.0000	1.14812	1.02878	1.1749

Table 2 Average optimality performances of algorithms with different sizes of machines

Number of machines	MGA	GA	Rajendran	Ho & Chang
5	1.0000	1.1717	1.0212	1.2590
10	1.0000	1.1529	1.0302	1.1727
15	1.0000	1.1402	1.0314	1.1438
20	1.0000	1.1278	1.0324	1.1241
Overall average	1.0000	1.14812	1.02878	1.1749

2) Running time.

Table 3 and 4 show respecitvely the average running times of the algorithms against different problem structures, different sizes of jobs and different sizes of machines. All the time shown is in seconds on a Pentium PC.

Table 3 Average computational time of algorithms with different sizes of jobs

Number of jobs	MGA	GA	Rajendran	Ho & Chang
50	62.4250	64.7750	0.3750	0.2500
75	107.4750	106.8750	0.9250	0.4250
100	166.1250	155.9000	1.9750	0.6000
125	242.3250	212.1500	3.6500	0.9000
150	338.5750	274.2500	6.1250	1.2250

(Continued on page 447)

3 Conclusion

A new realization algorithm is proposed on the basis of the approach for computing the transfer function matrix for 2-D SGM in [3]. It is simpler and more explicit than that in [7]. The minimum realization problem is not discussed in this paper.

References

1 Lu W S. Some new results on stability and robustness of two-dimensional discrete systems. Multidimensional Systems and Signal Processing, 1994,5(2):345 – 361

2 Zou Yun and Yang Chengwu. An algorithm for computation of 2D eigenvalues. IEEE Trans. Automat. Contr.,1994,39(7):1436 – 1439

3 Zou Yun, Du Chunling and Zhou Xiangzhong. Disturbance decoupling control for 2-D linear shift-invariant systems. IFAC Youth Automation Conference, Beijing, 1995

4 Zou Yun and Yang Chengwu. Algorithms for computation of the transfer function matrix for 2D regular and singular general state space models. Automatica, 1995,31(9):1311 – 1315

5 Kaczorek T. The singular general model of 2-D systems and its solution. IEEE Trans. Automat. Contr.,1988,33(11):1060 – 1061

6 Kaczorek T. General response formula and minimum energy control for the general singular model of 2-D systems. IEEE Trans. Automat. Contr., 1990,35(4):433 – 436

7 Kaczorek T. Two-Dimensional Linear System. Berlin: Springer-Verlag, 1985

8 Kaczorek T. Realization problem for singular 2-D systems. Bulletin of the Polish A academy of Science, Tech. Sci.,1989,37(1):37 – 48

本文作者简介

邹 云 见本刊 1999 年第 2 期第 308 页.

杨成梧 见本刊 1999 年第 2 期第 308 页.

(Continued from page 444)

Table 4 Average computational time of algorithms with different sizes of machines

Number of machines	MGA	GA	Rajendran	Ho & Chang
5	97.6200	114.6200	1.2200	0.3200
10	155.9800	147.6600	2.1400	0.5800
15	212.9800	179.5800	3.0800	0.8000
20	266.9600	209.3000	4.0000	1.0200

4 Conclusions

From the figures in the Table 1 to Table 4, we can see that:

1) The MGA show consistent improvement over the general GA. The average improvement of MGA is significant of 2.878% better than that of the Rajendran's heuristic.

2) The computational time of all these algorithms increases with increasing problem size. The running time of the MGA and GA is directly proportional to the number of jobs. The computational effort of the MGA is less affected by the number of machines. MGA takes similar amount of computational time with GA, while both of them take much longer time than the heuristic algorithms.

Referencess

1 Goldberg D E. Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, Mass: Addison-Wesley, 1989

2 Gupta J N D. Heuristic algorithms for multistage flowshop scheduling problem. AIIE Transaction, 1972,4(1):11 – 18

3 Miyazaki S, Nishiyama N and Hashimoto F. An adjacent pairwise approach to the mean flowtime scheduling problem. J Operations Research Society of Japan, 1978,21:287 – 299

4 Ho J C and Chang Y L. A new heuristic for the n-job, M-machine flow-shop problem. European J Operational Research, 1991,52(2):194 – 202

5 Rajendran C. Heuristic algorithm for scheduling in a flowshop to minimize total flowtime. Int. J Production Economics, 1993,29(1):65 – 73

本文作者简介

唐立新 见本刊 1999 年第 2 期第 216 页.

刘继印 1962 年生. 1993 年英国 Nottingham 大学博士毕业, 现为香港科技大学工业工程系助理教授. 研究方向为生产计划与调度.