ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 1, No. 1, 2006, *pp*. 61-64

A Hybrid Genetic Algorithm for Bin Packing Problem Based on Item Sequencing

Xianghe Jing[1, 2 +] Xianzhong Zhou[3] Yangyong Xu[2]

[1] Dept. of Automation, Nanjing University of Science & Technology, Nanjing 210094, China
[2] Dept. of Information & Control, Air Defence Command College, Zhengzhou 450052, China
[3] School of Management & Engineering, Nanjing University, Nanjing 210093, China

(*Received Sep. 14, 2005, accepted Dec. 04, 2005*)

Abstract. The bin packing problem based on item sequencing is defined as follows: given a set of sequenced items of different sizes, how should one pack them all into bins of different capacity, in order to make the utility ratio of bin capacity as higher as possible? In this paper, next fit algorithm, a heuristic method for bin packing problem, is introduced into simple genetic algorithm, and a hybrid genetic algorithm is proposed for solving bin-packing problem based on item sequencing. In the hybrid genetic algorithm, the idea of simple genetic algorithm is used to search the solution of the bins sequence, and the idea of next fit algorithm is used to pack the sequenced items into the bins sequence obtained. Finally, the effectiveness of the hybrid genetic algorithm is convinced through computational results of an example.

Keywords: genetic algorithm, next fit algorithm, bin packing, item sequencing.

1. Introduction

The bin-packing problem is an NP-completeness problem [1]. The one-dimensional bin-packing problem based on item sequencing is defined as follows: given a set of M sequenced items $\{I_1, I_2, \ldots, I_M\}$ of different sizes $\{S_1, S_2, \ldots, S_M\}$, and different kinds of bins $\{B_1, B_2, \ldots, B_N\}$ of different capacities $\{C_1, C_2, \ldots, C_N\}$. The problem is how to pack all the items into the bins so that the total size of items in any bin must not exceed its capacity and the position of the sequenced items cannot be changed and the utility ratio of bin capacity is as higher as possible.

Since genetic algorithm (GA) was first introduced some 30 years ago [2], it has been widely recognized as an effective search paradigm in artificial intelligence, image processing, job scheduling, pattern recognition and many other areas, particularly as a method for gaining relatively good solutions to NP-hard optimization problems [3]. In references [4 ~ 7], some methods for solving bin-packing problem have been studied. In this paper, a hybrid genetic algorithm (HGA) is proposed for solving one-dimensional bin-packing problem based on item sequencing.

2. Design of hybrid genetic algorithm

In bin packing problem based on item sequencing, there are two key factors that influence the solution of the problem. The first factor is how to select and sequence the different capacity bins. The other factor is how to pack the sequenced item into the bin sequence. Therefore, in our HGA, the idea of GA is used to search the solution of the bin sequence, and the idea of next fit algorithm (NF) [8] is used to pack the sequenced items into the bin sequence obtained.

2.1. NF procedure

NF is a heuristic method for bin packing problem. On the assumption that bins have been selected and sequenced and stored in a list L, the *k-th* bin number in the list L is denoted as L_k ($\in \{1, 2, \ldots, N^*\}$, $N^* \leqslant M$). Then we can pack the given sequenced items into the bin sequence. In NF procedure, the items are packed in the following way.

+ Corresponding author. Tel.: +86-25-84303047-8;
E-mail address: luckycrane@hotmail.com

Step 1: Set $m \leftarrow 1$, $n \leftarrow 1$. The variable i and j represent the item number and the bin number, respectively.

Step 2: If $m \le M$, turn to step 3. Otherwise, turn to step 5.

Step 3: Let SI_k be the set of items in bin L_k. If $S_m + \sum_{n^* \in SIn} S_{n^*} \le C_n$, pack the item I_m into the bin L_n and set $m \leftarrow m+1$ and return to step 2. Otherwise, close the bin L_n and turn to step 4.

Step 4: If $n < N^*$, set $n \leftarrow n+1$ and return to step 2. Otherwise, select a new bin and add it into the end of the bin sequence, set $N^* \leftarrow N^*+1$, $n \leftarrow n+1$, and return to step 2.

Step 5: Terminate the procedure.

2.2. Individual description

In our HGA, the kind number of the bins corresponds to a gene. The chromosome is expressed by a set of sequence numbers. For example, a chromosome

$$c(k) = \underbrace{2 \quad 5 \quad 1 \quad \ldots\ldots \quad 2}_{m}$$

It means that the kind of the first and the *m-th* bin in the list L is B_2, and that the kind of the second bin in the list L is B_5, and that the kind of the third bin in the list L is B_1. From the chromosome, we can obtain the kinds of the bins used in the packing problem. The initial solutions are a set of N^* ($N^* \le M$) numbers, generated from 1 to N randomly.

2.3. Fitness of algorithm

In the packing problem, our aim is to make the utility ratio of bin capacity as higher as possible. Let Tn be the total number of bins that are used in a solution after packing by NF procedure, and $Ts(I)$ be the total size of the items, and $Tc(B)$ be the total capacity of the bins used in a solution. We define the objective function as follows.

$$\min f(X) = Ts(I) / Tc(B) = \sum_{m=1}^{M} S_m / \sum_{n=1}^{Tn} C_n$$

Let $E(X)$ be the fitness function of HGA. Because the value of $f(X)$ is positive, we look $f(X)$ as the fitness function, that is to say,

$$E(X) = f(X)$$

In the algorithm, the larger the value of fitness function is, the better the fitness is, and the better the solution is.

2.4. Selection operator

The selection operator should be designed in such a way that better individuals can be copied to offspring. In simple genetic algorithm, the selection operation depends on the ratio of individual fitness to average fitness, which cannot show the ascendant of better individuals when the fitness of every individual is almost the same size. And this phenomenon may reflect premature convergence problem of genetic algorithm. So we present our following design of selection operator.

Step 1: Sort the individuals in order of non-increasing value of fitness.

Step 2: Make two copies from the first 1/3 part of the sequenced individuals.

Step 3: Make one copy from the second 1/3 part of the sequenced individuals.

Step 4: Make above copies be the new individuals (offspring).

The design of selection operator above can keep the ascendant of better individuals.

2.5. Crossover operator

In crossover operator, offspring should inherit important factors of the parents. In our design of crossover operator, one-point crossover is adopted. And the crossover procedure is as follows:

Step 1: Select two individual c_1 and c_2 from the whole individuals randomly.

Step 2: Generate two crossover points at random.

Step 3: Exchange the genes between the two crossover points for the two individual c_1 and c_2, and we can get two new individuals d_1 and d_2.

For example, c_1 and c_2 are the two selected individuals.

c_1 : 2 3 6 4 1 5 2 6
c_2 : 3 5 1 2 5 6 3 4

The crossover points generated randomly are 2 and 6. In order to get the new individuals, we copy the genes in c_1 before the 2nd gene and after the 6th gene into the same position of d_1, and copy the other genes of c_1 into the same position of d_2. In the same way, we copy the genes in c_2 before the 2nd gene and after the 6th gene into the same position of d_2, and copy the other genes of c_2 into the same position of d_1. Thus we obtain the whole new individuals d_1 and d_2. The process of crossover operation is shown in figure 1.

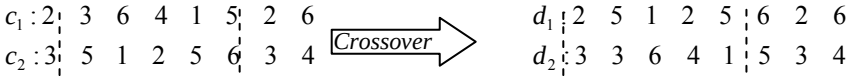

Figure 1: Example of crossover operation

2.6. Mutation operator

In our HGA, we adopt a double-point dynamic mutation operator, which is applied to change the values of the genes in the individuals. The mutation point and mutation value are generated randomly for every individual in each mutation operation. In each mutation operation, the mutation value is generated from 1 to N randomly.

2.7. Outline of algorithm

From the sections above, we can summarize the outline of our hybrid genetic algorithm, which is described as follows.

Step 1: Generate the initial population $G(t)$, and set $t \leftarrow 1$, the variable t represents the generation.

Step 2: Pack the sequenced items into the bin sequence gained by NF procedure.

Step 3: Evaluate the fitness of the individuals, and sort the individuals in the order of non-increasing value of fitness.

Step 4: If $E \geq T$, terminate the algorithm and list the solutions obtained. The parameter E and T represent the fitness value and the given satisfying value of fitness (utility ratio of bin capacity), respectively. Otherwise continue.

Step 5: Apply the selection operation to the sequenced individuals by the method of section 2.4.

Step 6: Apply crossover operation to the individuals by the method of section 2.5.

Step 7: Generate the mutation points and mutation values randomly for every individual. Apply the mutation operation to the individuals. Set $t \leftarrow t+1$, $G(t) \leftarrow G(t+1)$, and return to step 3.

3. Experimental results

In order to examine the performance of the hybrid genetic algorithm, an experiment is performed, and the experimental results are shown.

For example, there are 20 sequenced items $\{I_1, I_2, \ldots, I_{20}\}$ and 4 kinds of bins $\{B_1, B_2, B_3, B_4\}$. The problem is how to pack the sequenced items into the bins so that the utility ratio of bin capacity is not less than 0.95. The sizes of the items S_i and the capacities of the bins C_i are shown as follows.

$$S_i = \begin{cases} 4 & 1 \le i \le 7 \\ 7 & 8 \le i \le 13 \\ 5 & 14 \le i \le 17 \\ 6 & 18 \le i \le 20 \end{cases}, \quad C_i = \begin{cases} 12 & i=1 \\ 13 & i=2 \\ 15 & i=3 \\ 16 & i=4 \end{cases}$$

In this experiment, the population size is 100, the crossover probability is 0.8, and the mutation probability is 0.02. The procedure based on HGA has run for 100 times, and every time satisfying solutions are obtained. The average genetic generation is 9.75 when the satisfying solutions are obtained. Three of the satisfying solutions obtained by hybrid genetic algorithm are shown in table 1.

Table 1: The experimental results for bin packing problem

Bin number	Solution 1		Solution 2		Solution 3	
	Bin kind	Item	Bin kind	Item	Bin kind	Item
1	B_1	$I_1I_2I_3$	B_4	$I_1I_2I_3I_4$	B_2	$I_1I_2I_3$
2	B_4	$I_4I_5I_6I_7$	B_1	$I_5I_6I_7$	B_4	$I_4I_5I_6I_7$
3	B_3	I_8I_9	B_3	I_8I_9	B_3	I_8I_9
4	B_3	$I_{10}I_{11}$	B_3	$I_{10}I_{11}$	B_3	$I_{10}I_{11}$
5	B_3	$I_{12}I_{13}$	B_3	$I_{12}I_{13}$	B_3	$I_{12}I_{13}$
6	B_3	$I_{14}I_{15}I_{16}$	B_3	$I_{14}I_{15}I_{16}$	B_3	$I_{14}I_{15}I_{16}$
7	B_1	$I_{17}I_{18}$	B_1	$I_{17}I_{18}$	B_1	$I_{17}I_{18}$
8	B_2	$I_{19}I_{20}$	B_1	$I_{19}I_{20}$	B_1	$I_{19}I_{20}$
Utility ratio	0.9558		0.9643		0.9558	

As is shown in table 1, the utility ratios of the three solutions are all over 0.95. Obviously, the solution 2 is better than the other two. Although solution 1 and solution 3 have the same utility ratio of bin capacity, the bins used in each solution are not the same. This method can give us a chance to select a solution that we like from several satisfying solutions.

4. Conclusion

In this paper, a hybrid genetic algorithm is proposed for solving one-dimensional bin packing problem based on item sequencing. For obtaining a better solution, the idea of simple genetic algorithm is used to search the solution of the bin sequence, and the idea of next fit algorithm is used to pack the sequenced items into the bin sequence obtained by simple genetic algorithm. Finally, the effectiveness of the hybrid genetic algorithm is convinced through computational results of an example. The method, based on the hybrid genetic algorithm for item sequencing based bin packing problem, can give us a chance to select the solutions that we like from several satisfying solutions, which is of great value in its application.

5. References

[1] M.R.Garey and D.S.Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, W.H.Freeman and Company, 1979.

[2] J. H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence, University of Michigan, 1975.

[3] Erik D. Goodman, Alexander Y. Tetelbaum, and Victor M. Kureichik, A Genetic Algorithm Approach to Compaction, Bin Packing, and Nesting Problems, Michigan State University, July 12, 1994.

[4] Hitoshi Iima, Tetsuya Yakawa, A New Design of Genetic Algorithm for Bin Packing, The 2003 Congress on Evolutionary Computation, 2(2003), 1044-1049.

[5] Krzysztof Fleszar, Khalil S.Hindi, New Heuristics for One-dimensional Bin Packing, Computers & Operations Research 29(2002), 821-839.

[6] Bruno Codenotti, Gianluca De Marco, Mauro Leoncini, Manuela Montangero and Massimo Santini, Approximation Algorithms for a Hierarchically Structured Bin Packing Problem, Information Processing Letters, 89(2004)5, 215-221.

[7] Gregory Gutin, Tommy Jensen and Anders Yeo. Batched Bin Packing, Discrete Optimization, 2(2005)1, 71-82.

[8] D.Hochbaum(ed.), Approximation Algorithms for NP-Hard Problems, PWS Publishing, Boston (1996), 46-93.