IEMS Vol. 4, No. 2, pp. 136-144, December 2005.

A Genetic Algorithm for Order Picking in Automated Storage and Retrieval Systems with Multiple Stock Locations

Yaghoub Khojasteh Ghamari†
Graduate School of Systems and Information Engineering
University of Tsukuba, Tsukuba, Ibaraki 305-8573, JAPAN
Tel: +8180-5521-1062, E-mail: khojast@sk.tsukuba.ac.jp

Shouyang Wang
Institute of Systems Science, Academy of Mathematics and Systems Science
Chinese Academy of Sciences, Beijing 100080, CHINA
E-mail: swang@mail.iss.ac.cn

Abstract. This research deals with an order picking problem in automated storage and retrieval systems (AS/RS). When retrieval requests consist of multiple items and the items are in multiple stock locations, the storage/retrieval (S/R) machine must travel to numerous storage locations to complete each order. The aim of this research is to propose algorithms for the resolution of order picking problems with multiple stock locations to minimize the total time traveled by the S/R machine. We present and compare three alternatives for solving the problem based on enumeration, ordinary heuristic and genetic algorithms. We used a set of 180 different problems that are solved by these three algorithms. The results show that our proposed genetic algorithm is more efficient than the other two.

Keywords: Order Picking, Automated Storage and Retrieval Systems, Genetic Algorithm

1. INTRODUCTION

For an engineer and manager seeking to reduce the manufacturing costs, material handling within the plant or warehouse, involving storage, retrieval and inspections, is a fertile field. In the retrieving process (order picking), one is interested in determining optimal picking schedules which minimize the total distance (time or cost) traveled within the warehouse.

Order picking is a fundamental component of the retrieval function performed in warehouses. The main purpose of an order picking system is to fill customer orders by selecting the appropriate amount of material from a pre-designated storage medium known as the picking or forward area. Order picking represents only a subset of the material handling operations performed in warehousing. However, it is "one of the most costly and time-consuming functions of warehousing. In many warehouses, the difference between profit and loss depends on how well the order picking operation is run" (Bozer and White, 1990).

Since automated storage and retrieval systems (AS/RS) were introduced in the 1950s, the technology has advanced far beyond its original function, which was to eliminate the walking that accounted for 70% of manual retrieval time. AS/RS have been adopted not only as alternatives to traditional warehouses but also as a part of advanced manufacturing systems. The number of installed AS/RS is expected to grow rapidly. This is because AS/RS have many benefits including savings in labor costs, improved material flow and inventory control, improved throughput level, high flow–space utilization, and increased safety and stock rotation (Lee and Schaefer, 1997).

There are many studies on order picking problems in AS/RS and automated warehousing systems. Ratliff and Rosenthal (1983) gave an efficient algorithm to find the shortest path to visit a set of pick locations in a ladder layout. Elsayed (1981) made a chain of studies on the problem of optimally batching several orders in a two-dimensional warehouse with ladder structure. Recognizing that the exact solutions of the problem are very

† : Corresponding Author

difficult and time consuming to obtain, Elsayed and Stern (1983) presented some heuristic algorithms, but reported that none of them produces consistently superior results through experimentations. Hwang *et al.* (1988) studied a similar order picking problem in a single-aisle AS/RS and presented heuristic algorithms, which determine an efficient batching of orders for each tour of the S/R machine. The algorithms were based on cluster analysis with some similarity measures. Through simulation, they compared performances of the proposed algorithms with Elsayed and Sterns' results in 1983.

However, we know of no papers in the literature that address the order picking problem in automated storage and retrieval systems where each item can be stocked at several storage locations. For example, some manufacturers, whose products have a large variety of types, shapes, sizes and figures, are faced with this problem in their finished goods warehouses. Since classifying and zoning each individual type of product in the warehouse needs a warehouse with large space, the storage of an item in several places is unavoidable.

2. PROBLEM DESCRIPTION

In this paper, we assumed an end-of-aisle order picking system in a unit-load automated storage and retrieval system, where there are one or more aisles. Each aisle contains a storage rack on both sides of the aisle. There is an input/output (I/O) station at the end of each aisle. Only one storage/retrieval (S/R) machine carries out all of the storage and retrieval operations. The S/R machine capacity to deposit, pick up and carry is only one unit load. Each item is retrieved from an aisle by the S/R machine, then it is delivered to the I/O station of the same aisle. Therefore, there is no difference in delivery items by any I/O station. The S/R machine is positioned at one of the I/O stations before the receipt of each order. The starting place of the machine depends on the storage location (aisle) of the last item of the last order. An order can be a request for more than one item. Also each item can be in several storage locations in the warehouse.

When retrieval requests consist of multiple items and the items are in multiple stock locations, the S/R machine must travel to numerous storage locations to complete each order. The aim of this research is to propose algorithms for the order picking problem with multiple stock locations that minimizes the total time traveled by the S/R machine.

3. THE ALGORITHMS

Three algorithms were presented to solve the problem: an enumeration algorithm, an ordinary heuristic and a suitable genetic algorithm. To show the superiority of our proposed genetic algorithm, it was necessary to compare its results with the results of other methods and also with the optimal solution. Since our problem is new, no research has been conducted in this field; therefore, we first presented an algorithm to obtain the best solution to the problem so we called this algorithm the enumeration algorithm. We used the results from this algorithm as benchmark solutions for comparison with our proposed genetic algorithm.

In the enumeration algorithm, we identified all feasible solutions and compared them with each other to find the best solution. To do this, the method first finds all feasible ways to pick an order. Then, after assigning the picking sequence for each feasible way, it calculates the total time traveled by the S/R machine for each one, and finally selects the solution requiring the least amount of time to accomplish the order. This solution is considered as the optimum solution for the problem. Khojasteh and Sepehri (2000) discussed in detail the calculation of the number of feasible solutions, conceptualization of all feasible solutions, assigning a picking sequence and calculating the travel time of the S/R machine method.

Having solved various problems by the enumeration algorithm and identifying the best solution that had the minimum amount of travel time, we found that the existing items in the current aisle (i.e., the aisle in which the S/R machine is in at the beginning of the retrieving process) are the key to the final solution. We developed an algorithm on the basis of the mentioned results, and called it the current-aisle heuristic algorithm.

In the current-aisle heuristic algorithm, the existing items in the current aisle are selected first for retrieval. Afterwards, the remainder of the order (if any) is selected and all the various retrieval methods are studied. We can simplify the previous statements by stating that if m denotes the number of the ordered items existing in the current aisle, and if m=0, the method becomes similar to the enumeration algorithm. If m=1, the method first calculates the time traveled by the S/R machine (t_1) for only one existing item in the current aisle, then removes that item from the list of ordered items, and then for the remaining items (if any), it proceeds like the enumeration algorithm to obtain the minimum travel time (t_2). The total travel time of the S/R machine will be sum of t_1 and t_2 as the final solution.

If m>1, the method first assigns the picking sequence to pick up all m items which exist in the current aisle. After calculating the travel time (t_1), it removes the items from the list of ordered items. Then for the remaining items (if any), it proceeds like the enumeration algorithm, i.e., assigning the picking sequence for each feasible way and calculating the travel time for each one and selecting the minimum value among them (t_2). The total traveled time of the S/R machine will be sum of t_1 and t_2 as the final solution.

Khojasteh and Sepehri (2000) also discussed in detail the method of assigning a picking sequence of the ordered items existing in the current aisle. If any ordered items exist in the current aisle, the number of ways studied will be divided by the number of the items existing in the warehouse. Therefore, this task causes the total number of potential solutions to decrease dramatically; as a result, the CPU time (process time of the program) would be decreased as well.

3.1 Genetic Algorithm

A genetic algorithm is an optimization process that employs genotypes (individuals or chromosomes) in a population, and the genotypes are made of units called genes arranged in linear succession. Each genotype would represent a potential solution to a problem; an evaluation process run on a population of chromosomes corresponds to a search through a space of potential solutions.

In each generation, we evaluate each chromosome, select a new population with respect to the probability distribution based on fitness values, and recombine the chromosomes in the new population by mutation and crossover operators. After a number of generations, when no further improvement is observed, the best chromosome represents an optimal (possibly the global) solution. The algorithm is often terminated after a fixed number of iterations depending on speed and resource criteria (Michalewicz, 1992).

3.1.1 Representation

A chromosome represents a potential solution, where each one is viewed as a sequence of items each with its own associated allele. By analogy, each gene in a chromosome represents the item type and its associated allele represents the storage location. Therefore, each potential solution consists of a chromosome where the number of genes in the chromosome is equal to the number of items in the received order. An example is given in Figure 1.

03	02	01	04
6	4	3	2

Figure 1. Representation of a potential solution

Figure 1 shows a potential solution in which the items with codes 01, 02, 03 and 04 have been ordered; and item code 03 with location number 6, item code 02 with location number 4, item code 01 with location number 3, and finally item code 04 with location number 2 have been selected for retrieval, respectively.

It should be noted that the sequence of picking items has also been considered in the representation. In the example, item code 03 with location number 6 will be retrieved first, followed by item 02, item 01 and, finally, item 04.

3.1.2 Initialization

Initial population is randomly generated, so that the chromosomes are generated to the required size of the population, such that in each chromosome, at first, the ordered items are randomly distributed, and then a number is assigned to each of item. It should be noted that the condition of feasibility for each solution is necessary. Therefore, in the population-generating process, a suitable procedure is required to make each solution feasible. Thus, in each chromosome, the ordered items are randomly distributed without repetition, and the location numbers for each item are randomly selected. The assigned numbers range from 1 to the total warehouse inventory of that item.

For instance, in Figure 1, the actual total number of items is shown in Table 1.

Table 1. Warehouse inventory list for the items

Item type	01	02	03	04
Total inventory	5	8	6	3

As mentioned, in order to form the solution shown in Figure 1, first the ordered items are randomly selected (03, 02, 01 and 04), then for item 03, the selected integer number has been generated randomly between [1, 6], also for item 02, a number between [1, 8], for item 01 between [1, 5] and finally for item 04, a number between [1, 3] has been randomly selected.

3.1.3 Crossover

Among the described operators for permutation problems, the Partially Matched Crossover (PMX) has been used for the order picking problem. Partially matched crossover is viewed as a crossover of permutations, which guarantees that all items are found exactly once in each offspring, i.e., both offspring receive a full complement of genes, followed by the corresponding filling in of alleles from their parents. In Figure 2 there are two parents denoted by p_1 and p_2, and the crossover points are 1 and 3. According to the corresponding between [36, 72] and [98, 01], the repeated items are replaced; that is, 01 and 98 in first parent will be replaced by 72 and 36, respectively; while for second parent 72 and 36 will be replaced by 01 and 98, respectively. The generated offspring are o_1 and o_2 (Figure 2).

Meanwhile, according to PMX for the order picking problem, the role of crossover operator in this problem is to change the sequence of the items in a chromosome, without changing the associated alleles.

p_1:	24	36	72	01	98
	9	8	2	3	7
p_2:	72	98	01	36	24
	4	3	5	6	10
o_1:	24	98	01	72	36
	9	7	3	2	8
o_2:	01	36	72	98	24
	5	6	4	3	10

Figure 2. PMX operator

3.1.4 Mutation

Contrary to binary implementation that each gene is replaced with a complementary amount (0 with 1 and vice versa), in the order picking problem, the associated allele of each gene that has been selected by a mutation operator can be replaced with another allele in the range of total inventory of the item. This operator, on the other hand, does not have any role in changing the sequence of items, but can only select another number (storage location) for an item.

Suppose that in the o_1, the third gene has been selected by mutation, and also according to the Table 1, actual number of item 01 in various storage locations of the warehouse is five. Therefore, the mutation operator generates an integer random number between [1, 5] to replace the third gene (Figure 3). Of course, when the generated number is equal to the current number (that is 3), the operator repeats random number generating until it obtains a number that is not 3. In this example, the number 4 has been generated.

o_1 :	24	98	**01**	72	36
	9	7	**3**	2	8
o :	24	98	**01**	72	36
	9	7	**4**	2	8

Figure 3. The mutation operator

3.1.5 Evaluation and Selection

During each generation, chromosomes are evaluated using some measure of fitness. In most optimization applications, fitness is calculated based on the original objective function. In the order picking problem, the objective function is to minimize the travel time of the S/R machine. The total time traveled by the S/R machine is the criteria to select the chromosome for the next generation. Khojasteh and Sepehri (2000) explained the procedure for the calculation of travel time of the S/R machine.

Because this is treated as a minimization problem, we must convert the objective function value for each chromosome into a fitness value, so that a fitter chromosome has a larger fitness value. This can simply be done by the inverse of its value as follows (Cheng *et al.*, 1995):

$$eval(v_k) = \frac{1}{f(v_k)} \quad , \quad k = 1, 2, \ldots , pop\_size \qquad (1)$$

where $eval(v_k)$ is the fitness function for the k th chromosome and $f(v_k)$ is the total time traveled by the S/R machine for the k th chromosome. Population size (*pop_size*) determines how many chromosomes should be in the population at any given time.

Then, we use a roulette wheel as the basic selection method to reproduce the next generation based on the current enlarged population, in which a fitter chromosome has a large chance to be reproduced into the next generation. In this selection method, solutions with short travel times have higher probabilities of being chosen for the next generation. The roulette wheel is performed as follow:

1. Calculate the total time traveled by the S/R machine $f(v_k)$ for each chromosome v_k (k =1, 2, …, *pop_size*)
2. Calculate the fitness value $eval(v_k)$ for each chromosome v_k (k =1, 2, … , *pop_size*)
3. Find the total fitness of the population

$$F = \sum_{k=1}^{pop\_size} eval(v_k) \qquad (2)$$

4. Calculate the probability of a selection p_k for each chromosome v_k (k =1, 2, … , *pop_size*):

$$p_k = eval(v_k)/F \qquad (3)$$

5. Calculate the cumulative probability q_k for each chromosome v_k (k =1, 2, … , *pop_size*):

$$q_k = \sum_{i=1}^{k} p_i \qquad (4)$$

The selection process is based on spinning the roulette wheel *pop_size* times; each time a single chromosome is selected for the new population in the following way:

- Generate a real random number r between [0, 1];
- If $r \le q_1$ then select the first chromosome (v_1); otherwise select the k th chromosome v_k ($2 \le k \le$ *pop_size*) such that $q_{k-1} < r \le q_k$.

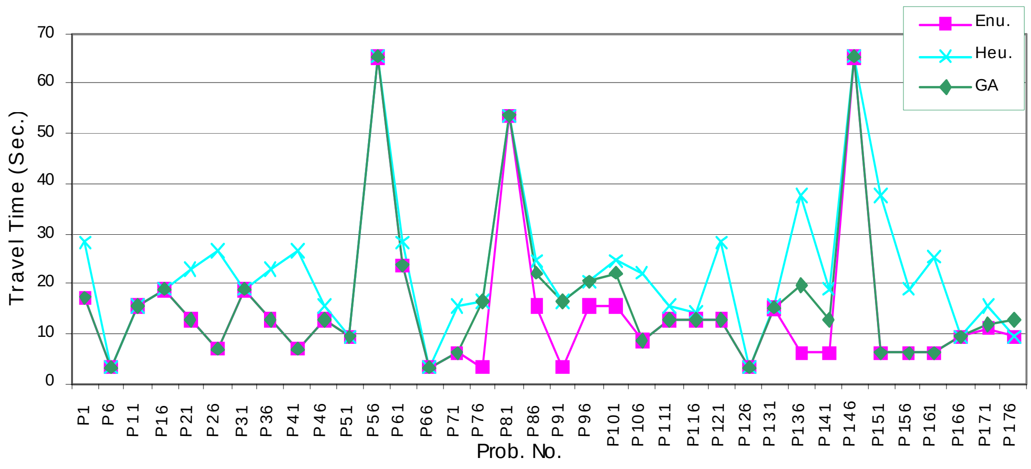
Using the deletion technique, all chromosomes of the last population are replaced by the newly generated chromosomes. Hence, the population size in each generation is constant and equals the initial given population size (Michalewicz, 1992).

4. NUMERICAL RESULTS

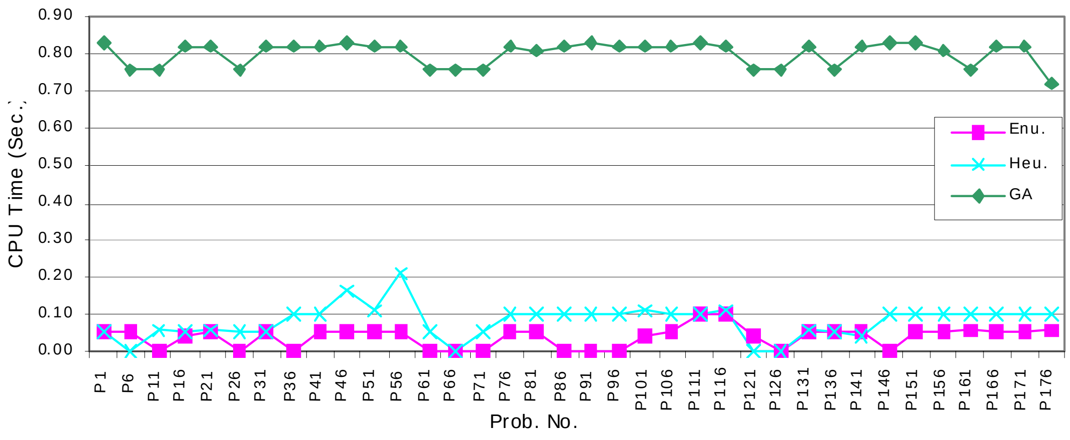
We programmed the three algorithms separately in Visual Basic 6, and then analyzed the order picking problem under the three proposed algorithms and compared their performances. We developed a set of 180 different problems that were solved by the three algorithms. Each problem was first solved by the enumeration algorithm to obtain "travel time" and "CPU time (process time)"; then the problem was solved by the other two algorithms. A summary of results obtained from the calculations are presented in several figures.

4.1 The 180 Problems

Designing 180 different problems is based on following four main warehouse parameters: warehouse density, warehouse capacity, shape factor (b), and kind of orders. In the case of warehouse density, we assumed three states, as 60%, 75% and 90% of total warehouse capacity used. Warehouse capacity deals with the number of aisles in the warehouse. We considered four states for warehouse capacity; a warehouse with one, two, three and four aisles. Each storage rack contains 780 storage locations. Since each aisle contains two racks, the capacity of each aisle is 1560 storage locations. A main reason not to consider a warehouse with five or more aisles is the assumption that there is only one S/R machine serving all aisles. Assigning a machine to a



(a) The travel time of the S/R machine



(b) The CPU time

Figure 4. Comparison of the travel time of the S/R machine and the CPU time under the three algorithms, where the order includes only one item.

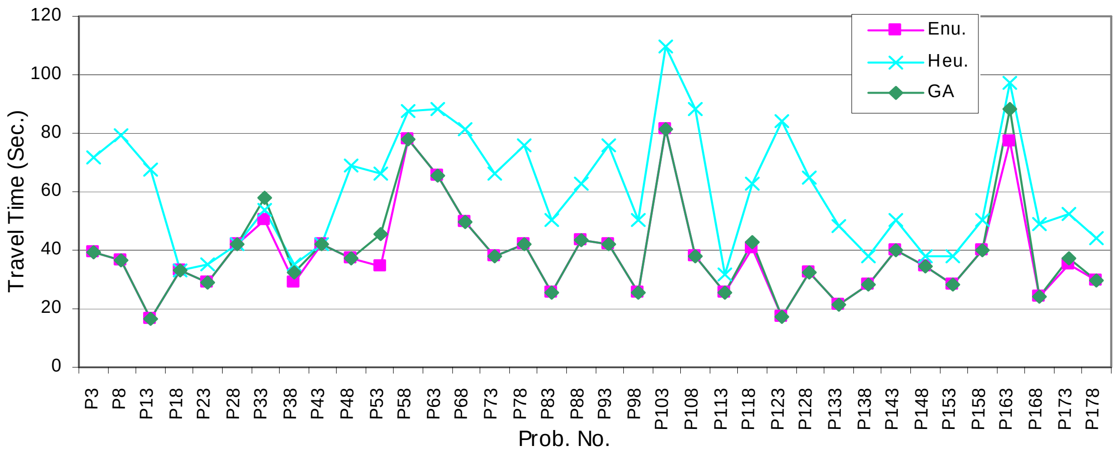
warehouse with five or more aisles will decrease the practical efficiency of the system.

Configuration of rack or shape factor, just as Bozer and White (1984) have described, is the time ratio of length and height of the rack, supposing that rack capacity and both horizontal and vertical velocity of S/R machine are fixed. Similar to Han *et al.* (1987) we used three values (0.6, 0.73 and 1) for the shape factor. Finally, we considered five kinds of orders, so that for orders 1, 2, 3, 4 and 5, the number of requested items to retrieve is one, two, three, four and five items, respectively.

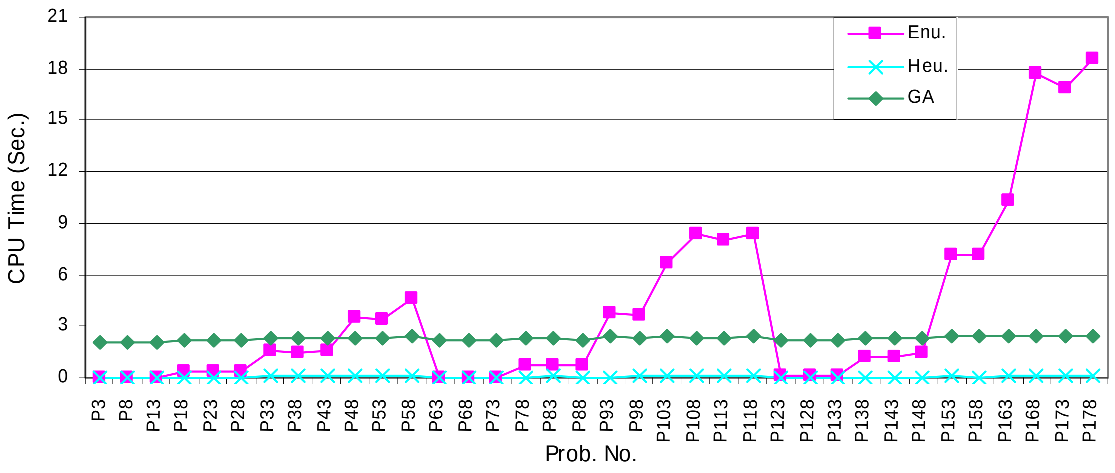
4.2 Results

The three algorithms were used to solve the 180 problems in a personal computer with the following specifications: Pentium III, CPU equal to 500MHZ, 128 MB RAM and 4 GB of assigned virtual memory. The results are shown in the following five figures. Figures 4, 5 and 6 show the comparison between "travel time of the S/R machine" and "CPU time" for three kinds of orders. Figures 4(a), 5(a) and 6(a) show the travel time and figures 4(b), 5(b) and 6(b) show the CPU time in the orders for one, three and four items, respectively. In addition, figures 7 and 8 show the comparison of the travel time and CPU time with respect to the number of aisles, when "b" is equivalent to 1 and 0.73, respectively.

In all graph series, the enumeration, current-aisle heuristic and genetic algorithms are denoted by "Enu.", "Heu." and "GA", respectively.

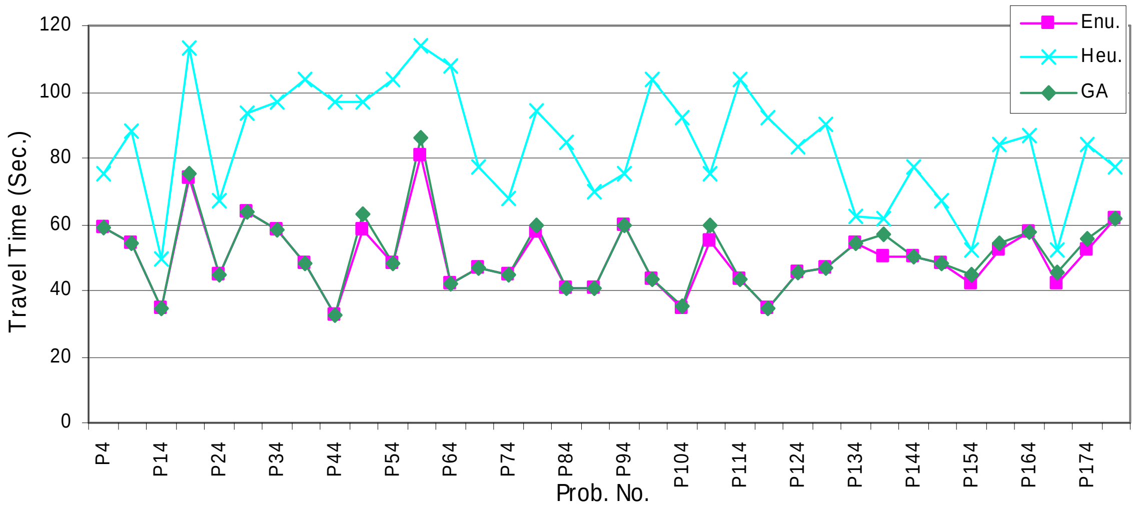


(a) The travel time of the S/R machine

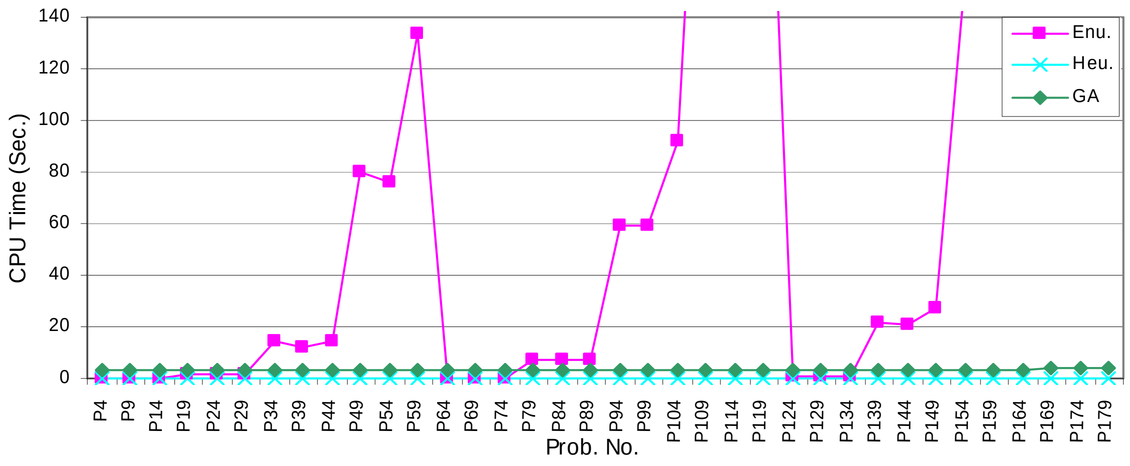


(b) The CPU time

Figure 5. Comparison of the travel time of the S/R machine and the CPU time under the three algorithms, where the order includes three items.

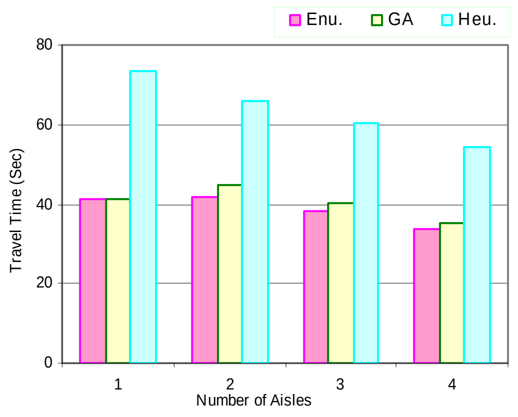


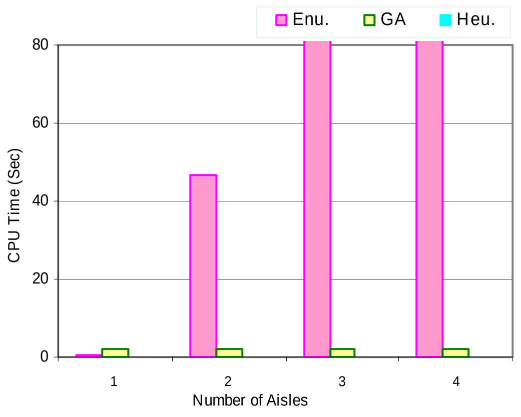
(a) The travel time of the S/R machine



(b) The CPU time

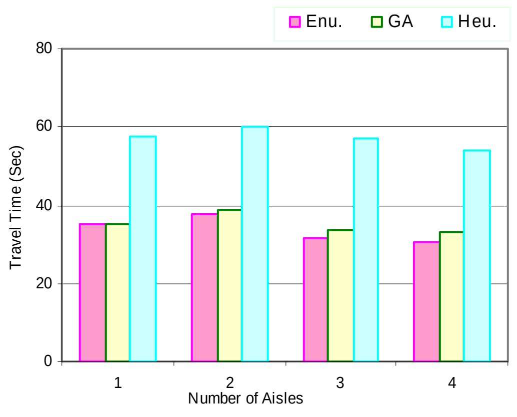
Figure 6. Comparison of the travel time of the S/R machine and the CPU time under the three algorithms, where the order includes four items



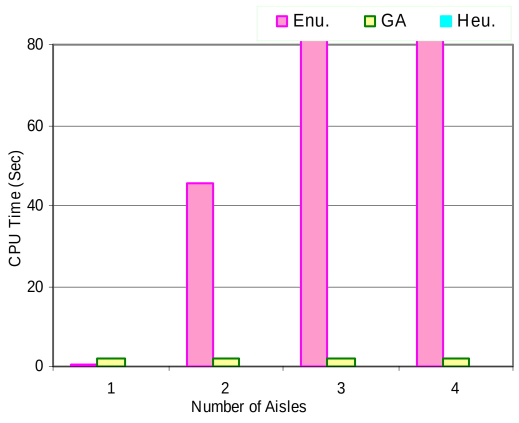


(a) The average travel time of the S/R machine (b) The average CPU time

Figure 7. Comparison of the three algorithms with increasing warehouse capacity (number of aisles), when b=1



(a) The average travel time of the S/R machine



(b) The average CPU time

Figure 8. Comparison of the three algorithms with increasing warehouse capacity (number of aisles), when b=0.73

5. ANALYSIS OF THE RESULTS

In the first case, where the order includes only one item, 77% of the solutions obtained by the genetic algorithm are optimal. In the other cases where the order included two, three, four and five items, the corresponding values are 77%, 86%, 72% and 70%, respectively. Most of the solutions generated by the genetic algorithm corresponded to the optimal solution and in the remaining 25%, the differences between the sub-optimal solutions from the optimal one were very slight. However, in the current-aisle heuristic algorithm, optimal solutions were found in only very rare cases and the differences between the sub-optimal solutions from the optimal one were very large (Figures 4 (a), 5 (a) and 6 (a)).

Increasing the number of items in an order affects the performance of these algorithms. CPU time of the enumeration algorithm is increased dramatically, in order to obtain the optimal solution. In the current-aisle heuristic algorithm, the probability of existence of some or all the ordered items in the current aisle is greater so the CPU time of this algorithm is decreased when the number of items in the order increases. However, compared to the other two algorithms, the current-aisle heuristic algorithm generated solutions with the longest travel times calculated for the S/R machine. In the genetic algorithm, 75% of the solutions are identical to the optimal solution, and the differences between the sub-optimal solutions from the optimal ones were very slight.

Rack configuration, b, also affected the performance of the three algorithms. Figures 7 and 8 show a comparison of the travel time of the S/R machine and the CPU time with respect to the number of aisles, when "b" is equivalent to 1 and 0.73, respectively. These figures compare the performance of these algorithms as affected by increasing warehouse capacity in various rack configurations.

In the case of a warehouse with just one aisle and various b levels, the enumeration algorithm gives the best solution, with a CPU time less than the CPU time of the genetic algorithm. If the warehouse has more aisles, the genetic algorithm requires less CPU time than the enumeration algorithm. Since the results of the figures for various b levels were the same, hence, we showed only the figures corresponding to b=1 and b=0.73. As the performance of the three algorithms for various b is approximately the same, thus, the rack configuration in the warehouse has no essential effect on the performance of the algorithms.

6. CONCLUSIONS AND EXTENSIONS

In this study, we address an order picking problem in AS/RS with a new property, multiple stock locations. To show the efficiency of our proposed genetic algorithm, we presented the enumeration algorithm, which obtains the optimal solution to the problem but with a long CPU time, hence making the method unsatisfactory. Then we developed the current-aisle heuristic algorithm that obtains solutions in a minimum CPU time but these solutions were mostly sub-optimal and required dramatically longer travel times for the S/R machine.

With the genetic algorithm, 75% of the solutions are identical to the optimal solution, and the differences between the sub-optimal solutions from the optimal ones were very slight. Thus, our proposed genetic algorithm is more efficient than the other two algorithms.

In future, meta-heuristic methods as well as branch and bound algorithms will be evaluated against various storage methods for their utility and a dual command (DC) S/R machine cycle in generating optimal solutions for order picking problems in AS/RS.

ACKNOWLEDGMENT

The authors would like to thank Professor M.M. Sepehri, Tarbiat Modarres University for his helpful comments and suggestions, which improved the paper.

REFERENCES

Bozer, Y. A. and White, J. A. (1984), Travel-time models for automated storage/retrieval systems, *IIE Transactions*, **16**, 329-338.

Bozer, Y. A. and White, J. A. (1990), Design and performance model for end-of-aisle order picking systems, *Management Science*, **36**, 852-866.

Cheng, R., Gen, M., and Sasaki, M. (1995), Film-copy deliverer problem using genetic algorithms, *Computers and Industrial Engineering*, **29**, 549-553.

Elsayed, E. A. (1981), Algorithms for optimal material handling in automatic warehousing systems, *International Journal of Production Research*, **19**, 525-535.

Elsayed, E. A. and Stern, R. G. (1983), Computerized algorithms for order processing in automated warehousing systems, *International Journal of Production Research*, **21**, 579-586.

Han, M. -H., McGinnis, L. F., Shieh, J. S., and White, J. A. (1987), On sequencing retrievals in an automated storage/retrieval system, *IIE Transactions*, **19**, 56-66.

Hwang, H., Baek, W., and Lee, M. -K. (1988), Clustering algorithms for order picking in an automated storage and retrieval system, *International Journal of Production Research*, **26**, 189-201.

Khojasteh, G. Y. and Sepehri, M. M. (2000), Order picking problem in an AS/RS with multiple stock locations, *M.Sc. Thesis*, *Tarbiat Modarres University*.

Lee, H. F. and Schaefer, S. K. (1997), Sequencing methods for automated storage and retrieval systems with dedicated storage, *Computers and Industrial Engineering*, **32**, 351-362.

Michalewicz, Z. (1992), *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, Berlin.

Ratliff, H. D. and Rosenthal, A. S. (1983), Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem, *Operations Research*, **31**, 507-521.