A Genetic Algorithm Approach
to
Door Assignments in Breakbulk Terminals

Roberto Bermúdez
Michael H. Cole, Ph.D.
Department of Industrial Engineering
University of Arkansas
Fayetteville, AR 72701, USA

Abstract
This paper presents a genetic algorithm for assigning doors in LTL (less-than-truckload) breakbulk terminals. Typically, strip (incoming freight) doors and stack (outgoing freight) doors are assigned to city areas. Incoming trailer loads are broken up and moved to appropriate outgoing trailers. The objective is to minimize the total weighted travel distance, a surrogate for labor cost and cycle time. The underlying problem is a Quadratic Assignment Problem. The experiments are based on real-world data.

Keywords
LTL breakbulk terminal, genetic algorithm, quadratic assignment problem

1. Introduction
LTL (less-than-truckload) freight is managed in *breakbulk* terminals between pickup and delivery (see Figure #1). Incoming trucks carry diverse loads from their origins to the breakbulk terminal. Typically, each incoming truck picks up freight from a specific origin zone. Incoming trucks are assigned to a "strip" door in the terminal where the freight is unloaded. After unloading, the freight is separated into individual loads according to destinations. Each individual load is moved to its appropriate "stack" door and loaded on a truck outbound to a specific destination. Typically, each destination is assigned to a single stack door, whereas origin freight is assigned to any available strip door. In addition to strip and stack doors, "open" doors exist which are either unassigned or frequently reassigned. Terminals range in size from fewer than 20 doors to more than 200 doors.
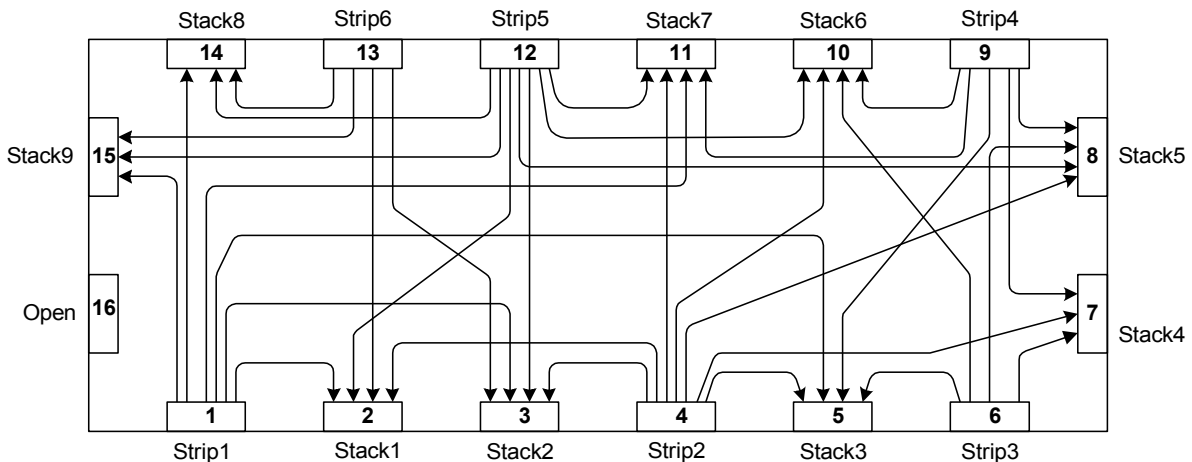

Figure 1. Sample LTL breakbulk terminal with 16 doors; 6 strip (incoming freight) doors, 9 stack (outgoing freight) doors, and 1 open door (no assignment)

The *door assignment problem* entails designating the arrangement of strip doors and stack doors, and the assignment of destinations to stack doors in such a way that cost (material handling, labor) or time is minimized (Gue, 1999). Doors are usually reassigned on an occasional basis, such as once or twice a year. The problem can be quite

complex due to the number of doors and the dynamic nature of freight flow patterns. Good solutions to the door assignment problem would help LTL carriers increase productivity (by reducing labor and equipment usage) and provide more reliable customer service (by reducing cycle times). Gue (1995) estimates that handling represents 15 to 20 percent of total costs for an LTL carrier.

The research discussed in this paper develops a genetic algorithm (GA) and prototype decision support tool for solving the door assignment problem. Traditional optimization methods tend to perform poorly for large-sized door assignment problems, as the underlying problem is the NP-hard quadratic assignment problem (QAP).

The objective function for the genetic algorithm is the minimization of the total weighted travel distance within the terminal, the total sum of the product of each shipment's weight times the distance it is transported between strip and stack door as defined by each solution to the door assignment problem. Minimizing the door-to-door distance that an operator travels to transport a load reduces labor cost by reducing the freight handling time.

2. Underlying Mathematical Model

This section defines a basic mathematical model for the door assignment problem (based on Tsui and Chang, 1990). The objective function minimizes the total weighted travel distance. The constraints ensure that each door is assigned to a single origin/destination, and that each origin/destination is assigned to a single door. The model is easily modified to account for cases when an origin or destination zone requires more than one door.

2.1 Parameters

M – number of origins
N – number of destinations
I – number of doors, $I \geq (M+N)$
D_{ij} – travel distance from door i to door j
W_{mn} – weight of loads from origin m to destination n

2.2 Decision Variables

x_{mi} - 1 if origin m is assigned to door i, 0 otherwise ($i = 1$ to I)
y_{nj} - 1 if destination n is assigned to door j, 0 otherwise ($j = 1$ to I)

2.3 Formulation

Minimize

$$\sum_{i=1}^{I} \sum_{j=1}^{I} \sum_{m=1}^{M} \sum_{n=1}^{N} D_{ij} W_{mn} x_{mi} y_{nj} \tag{1}$$

Subject to

$$\sum_{m=1}^{M} x_{mi} + \sum_{n=1}^{N} y_{ni} \leq 1 \qquad i = 1,2,...I \tag{2}$$

$$\sum_{i=1}^{I} x_{mi} = 1 \qquad m = 1,2,...M \tag{3}$$

$$\sum_{i=1}^{I} y_{ni} = 1 \qquad n = 1,2,...N \tag{4}$$

$$x_{mi} = 0 \ or \ 1 \qquad m = 1,2,...M \ , \ i = 1,2,...I \tag{5}$$

$$y_{nj} = 0 \ or \ 1 \qquad n = 1,2,...N \ , \ j = 1,2,...I \tag{6}$$

3. Literature Review

The literature is reviewed in three parts: (1) LTL terminal design and operations, (2) genetic algorithms (general), and (3) quadratic assignment problems and genetic algorithms.

3.1 LTL Terminals

Peck (1983) develops a simulation that mines data from a shipment database. His main contribution is defining and simulating the "full floating dock" in which terminal space is allocated on a continuous basis. His model assigns doors in a heuristic fashion to (approximately) minimize freight transfer time.

Tsui and Chang (1990, 1992) develop search procedures for the basic door assignment problem, with the objective of minimizing the weighted distance between incoming and outgoing trailers. Their initial approach (1990) is very sensitive to starting solutions. Tsui and Chang (1992) observe that computation time increases dramatically as the size of the problem increases. They recognize the need for further research so that large size problems can be solved within a reasonable amount of time.

Gue (1995) develops a model that seeks to minimize travel costs, congestion costs, and interference costs. His algorithm combines traditional swap heuristics and queuing analysis. Gue does not report statistics for calculation time of his algorithm. He states that LTL carriers can increase profits by 10 to 20 percent according to his model. Gue also stresses the fact that minimizing the weighted distance alone might create congestion and interference of material handling equipment in LTL terminals that use dragline conveyors. Our industrial supporter uses dragline conveyors in one major distribution center only. For this reason, congestion is not critical at the majority of these terminals and is left for future research. The minimization of the weighted distance is our initial approach.

3.2 Genetic Algorithms

Genetic algorithms (GAs) are used to solve design problems in a similar fashion as natural selection solves biological design problems. Heitkotter and Beassley (1999) give a basic introduction to genetic algorithms. Genetic algorithms generate populations of individuals. Typically, each individual represents a problem solution. Individuals exchange information by mating and produce offspring (new solutions). The probability of an individual surviving to reproduce is based on its fitness value (objective function value). Mutations prevent getting stuck in a local optimum. In the end, after numerous generations, the surviving individuals represent (hopefully) good solutions. According to Reeves (1993), from an operational research perspective, the idea of a genetic algorithm can be understood as an intelligent exploitation of random search.

In pseudocode, adapted from Hussain and Sastry (1995), a typical GA works as follows

> Initialize the parameters of the GA (selection, Pc, Pm, etc.)
> Randomly generate the initial population
> While convergence = false
>   Calculate the fitness value of each member of the population
>   While (number_of_individual <= population_size)
>   Select two parents (parent1, parent2) using a selection strategy
>   Perform the crossover operation between parent1 and parent2 based on Pc
>   Mutate each offspring based on Pm
>   Increase counter according to the number of offspring
>   End While
>   Construct the intermediate population with set of offspring
>   Construct a new population set using a replacement scheme
> End While
> Output best individual(s)

The *selection* method determines which individuals from the current population will mate to create new individuals (solutions). The number of individuals selected is based on the probability of crossover, Pc, which is the percentage of the current population that will mate. The *crossover* technique establishes how the genetic information from the parents is exchanged to create the offspring. As each new individual is created, there is a probability Pm that it will mutate and its genetic structure will be altered slightly. After the offspring population is generated, the *replacement*

method determines which parent solutions will be combined with the offspring to form the new population in the following generation. This process continues until the convergence criterion is met, such as a maximum number of generations.

3.3 QAPs and Genetic Algorithms
Kochhar, et al., (1998) apply GAs to the facility layout problem (a version of the quadratic assignment problem). They find solutions within 1% to 5% of the best-known solutions. In some cases, their GA provides better solutions than previously known.

Tate, *et al.*, (1994), state "genetic algorithms provide a particularly robust method for the QAP and its more complex extensions." They report that their GA performed well on a standard test bank of QAP problems, even without extensive fine-tuning of its parameters.

Nissen (1994) presents an evolutionary strategy (not strictly a GA) to solving quadratic assignment problems. He finds that approaches modeled after biological natural selection can outperform traditional heuristics such as 2-OPT.

Miagkikh, *et al*, (1996), propose the use of a hybrid algorithm that combines the advantage of local search and GAs as the global search technique for the solution of the QAP. Their GA method has non-standard features related to the mutation operator and the representation of chromosomes.

Tavakkoli-Moghaddain, *et al*, (1998), also approach the facility layout problem by implementing genetic algorithms. They state "GAs have successfully been applied to NP hard problems such as those resulted in mathematical modeling of facilities design problems." Their GA provided results that bettered previous solution values calculated by the branch-and-bound technique by up to 11.8%.

4. Genetic Model
This section explains encoding (problem representation) and crossover (mating) for the LTL door assignment problem.

4.1 Encoding
Each individual in the GA is encoded to define a solution for the door assignment problem in the context of LTL terminals. For this purpose, freight comes into strip doors indexed from 1 to M, freight for a specific destination is loaded at stack doors indexed from $(M+1)$ to $(M+N)$, and the physical doors at the terminal are indexed from 1 to I $(I \geq M+N)$. A set of individuals (valid solutions) can be represented as a matrix, with each row representing an individual and each column *index* representing a physical door at the terminal. The *value* of a (row, column) element represents the zone (origin/destination) assigned to that door. In Table I, for example, element (1,3) has a value of 2, which means that individual 1 (row number) assigns zone 2 to door 3 (column number). In GA vocabulary, each individual (row vector) represents a chromosome made of a string of genes. Therefore, each gene defines the assignment of that zone number to the physical door number at the terminal. Nissen (1994) also uses this encoding.

Table I. Two sample individuals for a 16-door LTL terminal

Column:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Individual 1	13	6	2	4	11	8	9	7	0	1	15	10	5	14	3	12
Individual 2	3	4	10	14	0	15	2	6	11	13	7	12	8	5	1	9

This representation is problem specific because it has to adhere to constraint (2) of the underlying QAP. According to this constraint, a door can only have one assignment. This implies that the value of the "genes" in the chromosome has to be unique; a zone (origin/destination) cannot be assigned at two different door locations. Consequently, the number of zones to be assigned cannot exceed the number of doors available at the terminal. The only gene value that can be repeated along the chromosome is zero, which indicates that those doors have no assignment and that there are more doors than zones.

4

4.2 Crossover

The crossover procedure ensures that offspring represent a valid solution by complying with constraint (2) of the QAP. Since the encoding of the individuals is not the standard binary representation used in GAs, our crossover technique, Swap Window, is also designed specifically for the door assignment problem.

Suppose that the two individuals in Table I were selected to mate. Crossover of genes is performed as follows. First, randomly select a door "swap window" as presented in Table II. Second, identify genes within the swap window that are common to the two parents. In this case, genes with value 1, 5, 7, 8, and 15 are commonly distributed in the swap window.

Table II. Common genes within swap window [6,15]

Swap window [6,15]

| Individual 1 | 13 | 6 | 2 | 4 | 11 | **8** | 9 | **7** | 0 | **1** | **15** | 10 | **5** | 14 | 3 | 12 |
| Individual 2 | 3 | 4 | 10 | 14 | 0 | **15** | 2 | 6 | 11 | 13 | **7** | 12 | **8** | **5** | **1** | 9 |

Third, exchange the common genes to create two offspring (Table III). In the parents, Individuals 1 and 2, the common genes are ordered (8, 7, 1, 15, 5) and (15, 7, 8, 5, 1), respectively. The offspring Individual 3 has its common genes ordered (15, 7, 8, 5, 1) as with parent 2. The offspring Individual 4 has its common genes ordered (8, 7, 1, 15, 5) as with parent 1. The unique genes within the swap window are kept by the children at the same gene locations.

Table III. Two new offspring

| Individual 3 | 13 | 6 | 2 | 4 | 11 | **15** | 9 | **7** | 0 | **8** | **5** | 10 | **1** | 14 | 3 | 12 |
| Individual 4 | 3 | 4 | 10 | 14 | 0 | **8** | 2 | 6 | 11 | 13 | **7** | 12 | **1** | **15** | **5** | 9 |

These two new individuals represent two new solutions to the door assignment problem. Their fitness is measured by calculating the objective function (equation (1) in section 2).

5. Experimental Design

The experimental design is intended to provide an overview of GA performance on the door assignment problem. The experimental factors can be divided into two areas: real-world scenarios, and GA parameters. By running tests that combine these experimental factors, we are able to verify whether GA is a good approach for the door assignment problem and to make suggestions on how to "fine tune" the GA parameters for optimum performance.

5.1 Real World Factors

In order to test the benefits of GAs in solving door assignment problems of different complexity, the following characteristics of LTL terminals were experimental factors,

1. **Terminal shape**: rectangular with symmetric door arrangements, rectangular with asymmetric door arrangement. Other shapes, like "T" or star-shaped, were considered. However, these are uncommon in LTL practice and no data for the freight distribution patterns were available.

2. **Number of doors in terminal**: small (16), medium (43), large (195)

5.2 GA Parameters

The GA parameters investigated as experimental factors are

1. **Starting population size**: small (100~200), large (500~1500)

2. **Selection**: these techniques are commonly implemented for GA applications (Chambers, 1995 a). They range from genetically conservative to genetically disruptive
   a. Fit-fit. Highly conservative of genetic information. When the population is sorted by fitness value, the fittest individual is mated with the next fittest, and on.

b. Tournament. Moderately conservative of genetic information. It selects two individuals to compete according to their fitness values. The one with the better fitness becomes a parent and is placed in a mating pool. The tournament continues with randomly selected sets of individuals.

c. Fit-weak. Maximally disruptive of genetic information. In a population that is sorted by fitness value, the fittest individual is mated with the least fit, the next fittest with the next least fit, and on.

3. **Replacement**: each replacement technique defines a different genetic development model as explained above. These are

a. Elitism: preserves the better parents from a population by appending them into the next generation's population that stores the newly create offspring

b. Strongest Individual: every couple of parents generate two children, out of the four, only the best two (highest fitness value) solutions are kept as members of the next generation's population

4. **Pc**, probability of crossover (incremental population model): low (0.40), high (0.70)

5. **Swap Window size** (number of genes)
In each case, the swap window is located randomly along the chromosome. The size of the window can be

a. Random, but constrained to minimum window size

b. Fixed window size, a function of the number of doors in each terminal

6. **Pm**, probability of mutation: low (0.01), medium (0.05), high (0.10)
A standard mutation technique is used: two genes are chosen at random and their locations switched

7. **Removal of Duplicate solutions**: removing clones, not removing clones
Removing duplicate individuals helps maintain population diversity and avoids unnecessary calculations

8. **Convergence criteria**: maximum number of generations

9. **Generation Size**:
a. Increasing population size: The population size is incremented throughout the genetic development when Elitism replacement is combined with any of the three selection methods (fit-fit, tournament, and fit-weak). The population in the future generations is composed of the newly created offspring and the "elite" parents whose fitness value is above the population's mean fitness value. Therefore, the best individuals are never lost, but are carried over to the next generation. If *clones* (duplicate individuals) are removed each generation, then the population does not grow as rapidly as otherwise.

b. Constant population size: The *generational* model is another common form of GA (Chambers, 1995 b). This model keeps the population size constant by replacing the entire parent population by the offspring. However, this may not be efficient because not every child solution is guaranteed to be better than the parents. We adapted the concept of the generational GA model by using Strongest Individual replacement. For every generation, all the individuals in the current population are mated in couples (Pc = 1.0). Two parents generate two children, but out of the four, only the two strongest individuals are carried over to the next population. For the LTL GA, Strongest Individual replacement is combined with Fit-Fit or Fit-Weak selection only. In order to keep the population size constant, duplicates cannot be removed.

5.3 GA Testing

A small 16-door terminal was created to test the GA decision support tool during development (Microsoft Excel 97 and VBA). The freight distribution patterns and the physical layout of the terminal were not derived from a real-world case. The terminal has 9 stack doors, 6 strip doors, and 1 open door.

Initially, a test was performed to study the differences in the performance of each genetic algorithm model that had been programmed for the LTL GA. This decision support tool is able to perform under an incremental population model and a constant population model. Figure 2 shows how the two GA models compare in their efficiency to find better solutions for the door assignment problem.
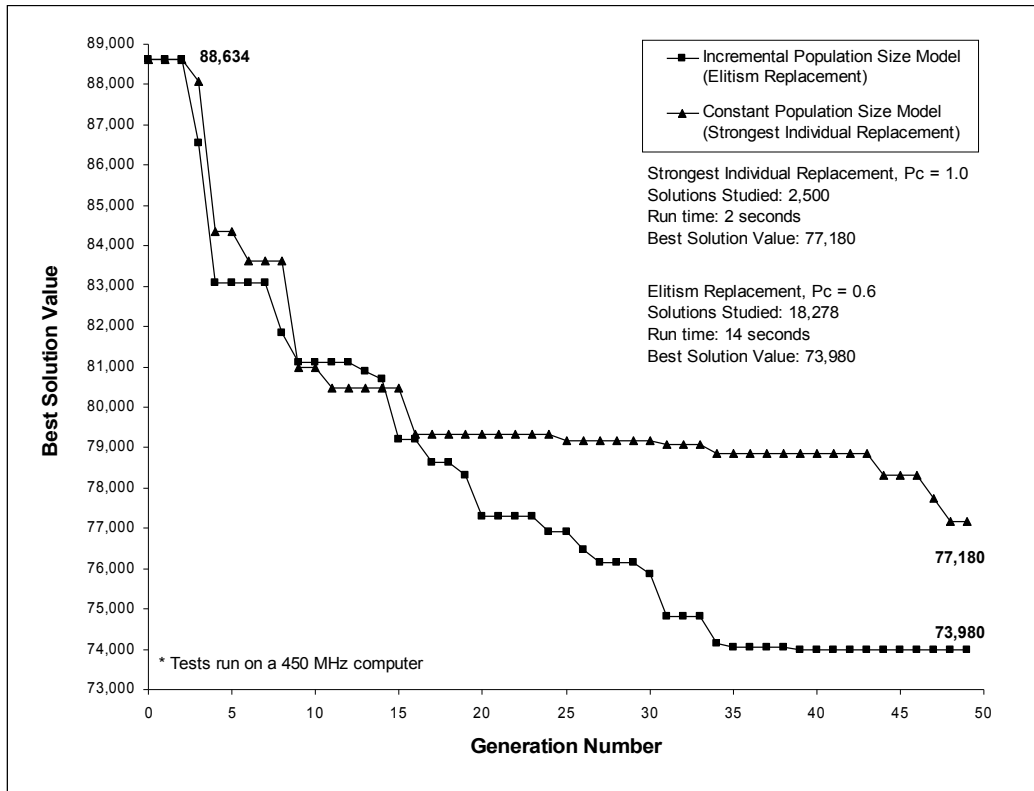
Figure 2. Genetic algorithm performance for the incremental and the constant population models

Both tests had the same starting population of 50 individuals; the incremental population model obtained a better solution value (73,980). The incremental population model evaluated a total of 18,278 solutions in 14 seconds and the constant population model evaluated 2,500 solutions in only 2 seconds; both ran through 50 generations. The incremental population size model evaluates an increasing number of solutions in each generation, which also demands a longer computation time. The constant population model evaluates 50 new solutions per generation. Over 50 generations, a fewer number of solutions would have been considered, yielding a less favorable solution value. There is also less diversity in the population and it is likely that the genetic development might get stuck in a local optimum. This might be the case as the constant population model curve shows smaller improvements in the solution value between the 15th and 40th generation. However, if the starting population sizewere larger (more diverse) and the genetic development were run for longer generations, the constant population model might outperform the incremental population mode in a shorter computation time.

A second experiment was performed to check how different starting populations might affect the performance of the genetic algorithm. In Figure 3, each test was seeded with 50 different solutions as the starting population. At the end of 50 generations, under the incremental population model, both tests had obtained very similar solution values. This suggests that the LTL GA is not sensitive to the starting population as long as the starting population is sufficiently large and diverse.
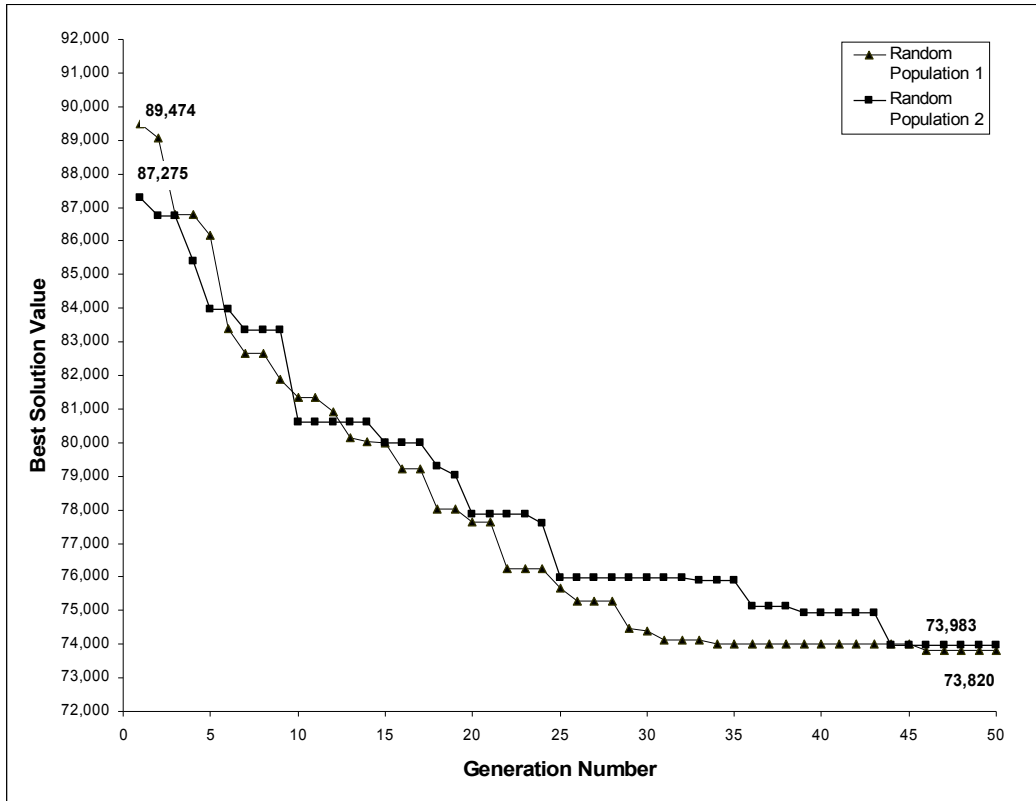
Figure 3. Performance of the genetic algorithm under two different starting populations

Following these preliminary tests, a set of experiments was run to test for the most efficient combination of GA parameters. All the experiments were run on a 450 MHz computer with 128 MB of RAM. According to the freight and door-to-door distance data, the best objective function value ever found, from all the experiments that were performed, was 72,839 pound-feet.

Table IV shows the results obtained from running the incremental population size model of the LTL GA by using Elitism Replacement. The initial population size was 50 and the genetic development was run over 40 generations. All experiments were run for Pc = 0.6 and Pm = 0.03, both are moderate values for each criterion. The largest improvement in the best solution value was 16.28% from 86,999; the best (random) solution in the initial population.

Table IV. Preliminary experiments for the incremental population model (Pc = 0.6, Pm = 0.03)

Selection	Crossover	Min Win Size	Fix Win Size	Delete Clones	Solutions Studied	Run Time (min:sec)	Best Sol Value	% > (Best Solution Found)	Found in Generation
Tournament	Rnd Win	7		Y	11,768	0:13	72,995	0.21%	38
Tournament	Rnd Win	10		Y	48,441	0:54	73,315	0.65%	37
Tournament	Rnd Win	12		Y	21,606	0:24	73,818	1.33%	39
Tournament	Rnd Win	12		N	206,724	2:51	73,315	0.65%	39
Fit-Fit	Rnd Win	7		Y	16,287	0:13	73,378	0.73%	35
Fit-Fit	Rnd Win	10		Y	26,252	0:22	73,717	1.19%	40
Fit-Fit	Rnd Win	12		Y	14,965	0:12	73,930	1.48%	40
Fit-Fit	Rnd Win	12		N	78,232	1:12	72,839	0.00%	29
Fit-Weak	Rnd Win	7		Y	353,037	4:54	73,946	1.50%	40
Fit-Weak	Rnd Win	10		Y	398,764	5:43	74,178	1.81%	40
Fit-Weak	Rnd Win	12		Y	420,945	6:24	74,865	2.71%	40
Fit-Weak	Rnd Win	12		N	488,197	7:09	73,525	0.93%	39
Tournament	Fix Win		8	Y	5,122	0:05	73,887	1.42%	36
Tournament	Fix Win		10	Y	11,796	0:12	74,454	2.17%	30
Tournament	Fix Win		12	Y	11,902	0:13	72,883	0.06%	36
Tournament	Fix Win		12	N	170,192	2:10	72,839	0.00%	36
Fit-Fit	Fix Win		8	Y	982	0:33	76,755	5.10%	22
Fit-Fit	Fix Win		10	Y	49,446	0:37	72,839	0.00%	37
Fit-Fit	Fix Win		12	Y	10,534	0:08	72,912	0.10%	35
Fit-Fit	Fix Win		12	N	127,841	1:46	73,315	0.65%	29
Fit-Weak	Fix Win		8	Y	315,992	3:53	73,317	0.65%	39
Fit-Weak	Fix Win		10	Y	593,171	7:41	74,454	2.17%	39
Fit-Weak	Fix Win		12	Y	560,364	7:52	75,457	3.47%	40
Fit-Weak	Fix Win		12	N	468,656	6:01	74,148	1.77%	38

The results above suggest the following preliminary observations:

* Under Elitism replacement, Fit-Weak selection performs poorly. The genetic information is disrupted and it makes it difficult to reach a good solution early in the genetic development. Note that the population diversity is very high, even if clones are not removed; nearly the same number of solutions are evaluated. This combination is not efficient because it studies a large number of solutions without good results, thus spending too much computing time

* Fit-Fit and Tournament are the most efficient selection techniques under the incremental population model. They provide a good balance between number of solutions studied and the quality of the best solutions. Fit-fit selection is highly conservative of the genetic information, thus converging to a solution in the fastest amount of time. For both selection methods, in general, removing duplicate solutions (clones) makes the search more efficient, but does guarantee a better final solution.

* For the most part, random swap window crossover yielded better results than fixed swap window crossover. Constraining the random window to a minimum window size allows the crossover method to exchange both small and large number of genes between parents, as opposed to exchanging only a fixed number of genes as with fixed window crossover. This is clearly seen in the results obtained with minimum window size of 7 and fixed window size of 8.

* For random window crossover, under both tournament and fit-fit selection and removal of duplicates, the values for the minimum size of the swap window that generated better results in shorter time were 7 and 12. These values represent 44% and 75% of chromosome length (16 genes) respectively. In the case of fix window crossover,

window sizes of 8 (50%) and 12 (75%) generated better results for tournament selection, and sizes of 10 (63%) and 12 (75%) generated better results for fit-fit selection.

Table V presents the results obtained with the constant population size model of the LTL GA. Under this model, there are fewer parameters to combine for the experiments since, by default, Pc is set to 1.0, only Fit-fit and Fit-weak selection can be used, and elimination of clones is disabled. The genetic development was run through 145 generations with a population size of 250. Pm was set to 0.03. As with elitism replacement, the size of the swap window was selected to explore the optimum value according to the percentage of genes in a chromosome that would be exchange at the time of mating solutions. The range of values represent near 50%, 63%, and 75% of the chromosome length. The largest improvement in the best solution value was 12.56% from 83,302; the best (random) solution in the initial population.

Table V. Preliminary Experiments and results for the constant population model (Pc = 1.0, Pm = 0.03)

Selection	Crossover	Min Win Size	Fix Win Size	Solutions Studied	Run Time (min:sec)	Best Sol Value	% > (Best Solution Found)	Found in Generation
Fit-Fit	Rnd Win	7		36,250	0:25	72,839	0.00%	68
Fit-Fit	Rnd Win	10		36,250	0:26	72,839	0.00%	139
Fit-Fit	Rnd Win	12		36,250	0:26	72,839	0.00%	94
Fit-Weak	Rnd Win	7		36,250	0:23	72,839	0.00%	48
Fit-Weak	Rnd Win	10		36,250	0:26	72,839	0.00%	82
Fit-Weak	Rnd Win	12		36,250	0:27	73,539	0.95%	51
Fit-Fit	Fix Win		8	36,250	0:22	72,839	0.00%	84
Fit-Fit	Fix Win		10	36,250	0:22	72,839	0.00%	123
Fit-Fit	Fix Win		12	36,250	0:23	72,839	0.00%	134
Fit-Weak	Fix Win		8	36,250	0:23	73,493	0.89%	41
Fit-Weak	Fix Win		10	36,250	0:22	72,839	0.00%	66
Fit-Weak	Fix Win		12	36,250	0:24	72,839	0.00%	68

The results above suggest the following preliminary observations:

- Out of the two selection methods, Fit-weak was the most efficient since it found a solution at early generations in the genetic development. Strongest individual replacement, which keeps the population size constant, is efficient as long as there is great diversity in the population. Since duplicate solutions can be easily created, and are not removed, trapping the optimization process in a local minimum is likely to occur. To avoid this phenomenon mutations can be increased by a larger value of Pm or the population size has to be very large. Because of this, it is expected that Fit-weak might outperform Fit-fit selection because it is maximally disruptive of the genetic information and enhances population diversity. Fit-fit selection quickly converges to a local minimum and takes more time (later generations) to improve its solution search when small values of Pm (e.g., 0.03) are used
- For random window crossover, under both fit-weak and fit-fit selection, the values for the minimum size of the swap window that generated better results at earlier generations were 7 and 12. These values represent 44% and 75% of chromosome length respectively. In the case of fix window crossover, window sizes of 10 (63%) and 12 (75%) generated better results for fit-weak selection, and sizes of 8 (50%) and 10 (63%) generated better results for fit-fit selection.
- Comparing the results of Table IV and V, strongest individual replacement can arrive at better solutions even after evaluating fewer solutions compared to elitism replacement. Since the population remains constant, there is no population overflow to slow the genetic development at later generations, as is the case with the incremental population model.

From the experiments, it was also observed that due to the symmetrical layout of the terminal any solution to the door assignment might have an alternate solution with the same objective function value. This is because of

symmetries in the terminal itself. The solutions presented in Table VI are different though they have the same objective function value of 72,839.

Table VI. Two different solutions with symmetrical parallelism and equal objective function value

Door:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Value
Individual 1	*6*	13	3	9	10	11	1	8	7	*14*	*15*	2	4	*12*	0	5	72,839
Individual 2	*12*	*4*	*2*	*15*	*14*	7	8	1	11	10	9	3	13	6	5	0	72,839

The physical doors numbered 1-6 and 9-14 are located along the longer sides of the rectangular terminal (refer to Figure 1). Doors numbered 7-8 and 15-16 are located at opposite ends of the terminal.

6. Results

This section presents the results obtained for the real world scenarios of LTL terminals. The results from these experiments were obtained by running the LTL GA on a 450 MHz computer with 128 MB of RAM. The goal of these experiments is to provide some guidance for fine-tuning of the LTL GA. More complete statistical analysis is a subject of future research.

6.1 Industrial Data

Experiments were conducted using data derived from an industrial sponsor (the data was suitably modified to meet the sponsor's proprietary concerns). The data provides information about the contents of each trailer, the weight and the destination of the shipments. Since each destination zone is assigned one or more stack doors, the data provides a reference for the stack door. However, a reference for the strip door where the trailer is unloaded is not available. Typically, in LTL terminals, incoming trailers are unloaded at any available strip door. The strip doors are not reserved for trailers coming from a specific origin zone. Therefore, it was necessary to manipulate the data and link each trailer to a specific strip door. We approached this problem by distributing the total incoming trailer freight into equal amounts to create the same door pressure at each strip door. In this way, each strip door is treated without any bias by the genetic algorithm optimization. The same approach was taken for destination zones that require more than one stack door.

The industrial supporter provided data for the historical freight patterns of the 43-door and 195-door terminals over a 7-day period. Both of these terminals are rectangular with an asymmetrical layout. The characteristics of the terminals are summarized in Table VII.

Table VII. Terminal and freight characteristics for 43-door and 195-door LTL terminals

Characteristic	43-Door Terminal	195-Door Terminal
Total Freight (lbs.)	1,845,823	17,154,236
Average Destinations per Trailer	2.41	8.63
Total Destination Zones	32	86
Strip Doors	8	63
Stack Doors	34	120
Open Doors	1	12

6.2 43-Door LTL Terminal

The results obtained from the preliminary experiments (16-door terminal) were considered when testing the GA for the 43-door and 195-door terminals. From the preliminary experiments it was learned that fit-weak selection does not perform well with elitism replacement, removing duplicates is preferred for the incremental population model, and that strongest individual replacement is more efficient than elitism in finding good solutions.

Table VIII presents the results obtained from exploring the feasibility of using the incremental population size model (elitism replacement) for the 43-door case. The initial population size was 100 and the genetic development had to be stopped due to overflow in the number of individuals in the final generations. The experiments were run

for Pc = 0.6, Pm = 0.05, and random window crossover only. The largest improvement in the best solution value was 39.52% from 140,858,123; the best (random) solution in the initial population.

Table VIII. Experimental tests for the 43-door terminal under incremental population model

Selection	Min Window Size	Last Generation	Delete Clones	Solutions studied	Run Time (min:sec)	Best Solution	% > (Best Solution Found)	Population in Last Generation	
								Size	Calculation Time
Tournament	14	51	Y	765,010	57:48	86,940,369	2.02%	282,099	24:36
Tournament	36	57	Y	741,838	47:47	89,170,428	4.47%	247,223	15:13
Fit-Fit	14	58	Y	859,388	47:53	85,701,455	0.60%	391,808	14:24
Fit-Fit	36	76	Y	757,464	40:40	85,185,287	0.00%	246,021	11:41

Table VIII show that as the number of doors increases, the incremental population model of the GA becomes less efficient in solving the problem. Each one of the tests was stopped due to population overflow at the time of the last generation. As the population size increases according to the probability of crossover (0.6), the calculation time per generation increases exponentially. Therefore, it would take a very large amount of time to run the genetic development over more generations to satisfactorily cover the search space and obtain a good solution.

To run the constant population model of the LTL GA for the 43-door case, it was necessary to run preliminary tests to determine a good combination of population size and last generation criteria. Since the best possible solution is not known, it is desirable to find a good solution at early generations and continue to run the genetic development to validate that the solution might be the best possible solution that the GA can find. For example, if a good solution is found near the last generation, it is likely that a better solution can be found by running the GA for more generations. The value for the probability of mutation, Pm, is also important because it avoids getting trapped in a local minimum. Table IX presents some of the results obtained from the preliminary testing. In all cases, the constant population model outperformed the incremental population model in much less time.

Table IX. Preliminary tests for the constant population model of the LTL GA

Trial Number	Initial Population	Convergence (generation)	Remarks (generation)	Best Sol Value	Pm	Run Time (min:sec)
1	2,000	800	Stopped at 141	~ 83,000,000	0.01	10:47
2	3,000	800	Stopped at 137	~ 82,000,000	0.01	15:43
3	3,000	800	Found at 245	82,834,534	0.05	93:46
4	600	300	Found at 232	82,586,233	0.10	7:04
5	600	250	Found at 239	82,201,300	0.10	6:34
6	200	800	Found at 219	84,080,071	0.20	6:18
7	50	800	Found at 330	85,019,727	0.30	1:35

Trial numbers 1 and 2 were stopped early because of lack of diversity in the last population. Since Pm was very low for these two trials, population diversity was poor. For the third trial, Pm was increased to allow the genetic development to reach the final generation with enough population diversity. However, run time was very large because it evaluated 2,400,000 solutions (population size times number of generations). Trials 4 and 5 were attempts to reduce run time while still obtaining a good solution. Their results suggest that smaller populations with larger values of Pm (to enhance population diversity) might provide better results in shorter time. However, the population sizes of trials 6 and 7 were too small and yielded poor results.

After considering the results obtained from the preliminary tests, the entire set of experiments was run for a population size of 1,200 through 800 generations (960,000 solutions studied). This represented a balance between the population size and last generation convergence according to the results of Table IX. Also, different values for Pm are used in the experiments shown in Table X, starting with 0.10, which provided a good solution in trial 4 of

Table IX. The largest improvement in the best solution value was 39.29% from 134,811,823; the best (random) solution in the initial population.

Table X. Experiments for the 43-door terminal for a constant population size of 1,200

Selection	Crossover	Min Window Size	Fix Window Size	Pm	Run Time (min:sec)	Best Solution	% > (Best Solution Found)	Found in Generation
Fit-Weak	Rnd Win	20		0.10	36:55	82,082,688	0.29%	163
Fit-Weak	Rnd Win	20		0.15	37:13	81,968,985	0.15%	129
Fit-Weak	Rnd Win	20		0.20	37:50	81,853,396	0.01%	143
Fit-Weak	Rnd Win	36		0.10	37:08	83,904,220	2.46%	332
Fit-Weak	Rnd Win	36		0.15	37:04	81,994,575	0.19%	448
Fit-Weak	Rnd Win	36		0.20	37:18	81,908,017	0.08%	233
Fit-Weak	Fix Win		24	0.10	40:18	81,924,254	0.10%	237
Fit-Weak	Fix Win		24	0.15	32:16	81,911,548	0.08%	146
Fit-Weak	Fix Win		24	0.20	31:00	81,869,577	0.03%	154
Fit-Weak	Fix Win		36	0.10	33:31	81,998,106	0.19%	257
Fit-Weak	Fix Win		36	0.15	33:53	81,909,305	0.08%	205
Fit-Weak	Fix Win		36	0.20	33:48	81,869,224	0.03%	149
Fit-Fit	Rnd Win	20		0.10	30:55	81,853,396	0.01%	496
Fit-Fit	Rnd Win	20		0.15	31:05	*81,842,182*	0.00%	576
Fit-Fit	Rnd Win	20		0.20	31:10	*81,842,182*	0.00%	510
Fit-Fit	Rnd Win	36		0.10	37:20	82,611,215	0.93%	774
Fit-Fit	Rnd Win	36		0.15	37:01	82,805,018	1.16%	747
Fit-Fit	Rnd Win	36		0.20	37:05	82,136,946	0.36%	697
Fit-Fit	Fix Win		24	0.10	25:22	*81,842,182*	0.00%	524
Fit-Fit	Fix Win		24	0.15	25:31	81,853,396	0.01%	563
Fit-Fit	Fix Win		24	0.20	25:29	81,853,396	0.01%	620
Fit-Fit	Fix Win		36	0.10	33:31	82,831,095	1.19%	784
Fit-Fit	Fix Win		36	0.15	33:38	82,165,545	0.39%	798
Fit-Fit	Fix Win		36	0.20	33:44	82,309,624	0.57%	757

According to the results in Table X,

- Seventy five percent of the tests (16 of 24) converged to better solution values than those found from the preliminary experiments in Table IX. All combinations of GA factors performed about equally well.
- The results seem to suggest that fit-fit selection was more efficient in finding better results than fit-weak selection. Fit-fit selection converged to the best solution; 81,842,182, and the second best solution; 81,853,396, on three of its tests
- The better solutions found through fit-weak selection were, consistently, those that used the largest value of Pm (0.20). Similarly, for fit-fit selection, larger values of Pm yielded better results on most of the tests.
- The best solution (81,842,182) was found under fit-fit selection by both crossover methods when the smaller swap window sizes were implemented. This suggests that the crossover technique might perform better when exchanging smaller portions of the genetic information. For example, fix window of size 24 exchanges 55.8% of the chromosome length and performs better than the fix window of size 36, which exchanges 83.7% of the genetic information between solutions
- The best solution (81,842,182) was found at generations 576, 510, and 524 by three different tests. This means that there were, at least, 224 more generations through which the genetic development was able to validate that there was not a better solution. This suggests that 81,842,182 pounds-feet is the best solution that the LTL GA can find for the 43-door terminal

A second set of experiments was performed as an attempt to reduce computation time while still obtaining good results, as suggested by trial 6 in Table IX. This time, the constant population model was run on a population size of 200 through 800 generations (160,000 solutions studied). Since the size of this population is very small, much larger values of Pm were used to maintain population diversity. The results are shown in Table XI. The largest improvement in the best solution value was 39.48% from 135,250,020; the best (random) solution in the initial population.

Table XI. Experiments for the 43-door terminal for a constant population size of 200

Selection	Crossover	Min Window Size	Fix Window Size	Pm	Run Time (min:sec)	Best Solution	% > (Best Solution Found)	Found in Generation
Fit-Weak	Rnd Win	20		0.20	5:33	*81,847,991*	0.00%	516
Fit-Weak	Rnd Win	20		0.25	5:50	82,277,508	0.52%	220
Fit-Weak	Rnd Win	20		0.30	5:37	82,056,835	0.25%	295
Fit-Weak	Rnd Win	36		0.20	7:30	83,774,673	2.30%	269
Fit-Weak	Rnd Win	36		0.25	6:43	82,998,845	1.39%	356
Fit-Weak	Rnd Win	36		0.30	6:37	82,084,818	0.29%	275
Fit-Weak	Fix Win		24	0.20	4:43	83,755,806	2.28%	148
Fit-Weak	Fix Win		24	0.25	5:00	82,241,339	0.48%	202
Fit-Weak	Fix Win		24	0.30	4:56	81,951,866	0.13%	224
Fit-Weak	Fix Win		36	0.20	6:13	82,506,160	0.80%	348
Fit-Weak	Fix Win		36	0.25	6:31	83,347,253	1.80%	318
Fit-Weak	Fix Win		36	0.30	6:09	81,939,814	0.11%	406
Fit-Fit	Rnd Win	20		0.20	5:07	81,859,845	0.01%	388
Fit-Fit	Rnd Win	20		0.25	5:09	81,917,102	0.08%	536
Fit-Fit	Rnd Win	20		0.30	5:09	82,612,567	0.93%	530
Fit-Fit	Rnd Win	36		0.20	6:26	83,000,395	1.39%	533
Fit-Fit	Rnd Win	36		0.25	6:18	82,219,406	0.45%	531
Fit-Fit	Rnd Win	36		0.30	6:13	83,794,521	2.32%	727
Fit-Fit	Fix Win		24	0.20	4:11	82,225,349	0.46%	489
Fit-Fit	Fix Win		24	0.25	4:11	82,575,865	0.88%	610
Fit-Fit	Fix Win		24	0.30	4:16	81,924,254	0.09%	509
Fit-Fit	Fix Win		36	0.20	5:35	82,030,729	0.22%	761
Fit-Fit	Fix Win		36	0.25	6:11	81,992,923	0.18%	662
Fit-Fit	Fix Win		36	0.30	5:33	81,968,985	0.15%	743

Analyzing the results in Table XI,

- These tests present an advantage in that the computation time is relatively short. As the constant population size is reduced in size by 1/6; from 1,200 (Table X) to 200, the computation time decreases at nearly the same ratio. Furthermore, the minimum solution value obtained from these tests (81,847,991) is very similar to the best solution value from Table X (81,842,182), which was the best solution found for the 43-door terminal case.
- The best solution values from Table XI; 81,847,991 and 81,859,845, were obtained under fit-weak and fit-fit selection respectively. As far as crossover, these solutions were obtained with a random window size of at least 20 genes. This suggests that exchanging smaller number of genes to create new solutions might be a more efficient approach for the crossover method.

6.3 195-Door LTL Terminal

The conclusions drawn from the experiments for the 43-door terminal were considered in the search for a solution to the door assignment problem for the 195-door terminal. Based on these observations, the incremental population GA model is not implemented for the 195-door terminal because it is not an efficient approach for finding solutions

for large terminal layouts. Furthermore, previous experiments also suggested that running the constant population model on a small population size reduces computation time and might still provide solutions that are not much different from solutions obtained with larger population sizes. For this reason, the preliminary experiments for the 195-door terminal started with similar experiments as those in Table XI of the 43-door terminal case. The constant population model was run on a population size of 200 throughout 800 generations. The results are shown in Table XII. The largest improvement in the best solution value was 27.24% from 6,818,917,113; the best (random) solution in the initial population.

Table XII. Preliminary experiments for the 195-door terminal based on a constant population size of 200

Selection	Crossover	Min Window Size	Fix Window Size	Pm	Run Time (hh:mm:ss)	Best Solution	% > (Best Solution Found)	Found in Generation
Fit-Weak	Rnd Win	78		0.20	01:34:48	4,997,847,865	0.73%	800
Fit-Weak	Rnd Win	78		0.25	01:34:17	4,988,992,265	0.56%	800
Fit-Weak	Rnd Win	78		0.30	01:34:49	*4,961,299,580*	0.00%	798
Fit-Weak	Rnd Win	146		0.20	01:52:25	5,015,600,299	1.08%	800
Fit-Weak	Rnd Win	146		0.25	01:52:58	5,013,982,770	1.05%	798
Fit-Weak	Rnd Win	146		0.30	01:53:20	5,051,495,608	1.79%	800
Fit-Weak	Fix Win		98	0.20	01:17:08	5,028,900,244	1.34%	796
Fit-Weak	Fix Win		98	0.25	01:17:07	5,093,181,712	2.59%	799
Fit-Weak	Fix Win		98	0.30	01:17:02	5,055,680,705	1.87%	800
Fit-Weak	Fix Win		146	0.20	01:40:10	4,990,468,086	0.58%	800
Fit-Weak	Fix Win		146	0.25	01:40:04	5,022,484,947	1.22%	797
Fit-Weak	Fix Win		146	0.30	01:39:31	4,983,792,356	0.45%	792

The computation time for the 195 door terminal is considerably larger than that for the 43 door terminal. Table XII shows that, for the 195-door terminal, the computation has increased to an average of 90 minutes. This is to be expected as the optimization of the combinatorial problem becomes more difficult for larger size problems. Another factor is that the freight patterns are more complex for larger terminals and the calculation time for the objective function for each solution also increases.

It is also important to recognize that the best solution from each test was found near the 800th generation, the convergence criterion. This indicates that the genetic development might be able to obtain even better solutions by running for longer generations. In order to determine a more appropriate convergence criterion, a single test was run through 1,300 generations. The result is shown in Table XIII. The percent improvement from the best solution in the initial population is now 27.91%.

Table XIII. Test for the 195-door terminal on the same population size of 200 through 1,300 generations

Selection	Crossover	Min Window Size	Fix Window Size	Pm	Run Time (hh:mm:ss)	Best Solution	Found in Generation
Fit-Weak	Rnd Win	78		0.20	02:11:41	4,916,072,668	1277

The test was able to obtain an improved solution value by continuing to perform the genetic development during 500 more generations than the tests from Table XII. The final population in the 1,300[th] generation lacked diversity, even when the probability of mutation was large (Pm=0.2). The final population was composed of duplicates of two individuals. The solution values for these two individuals were 4,916,072,668 (in Table XIII) and 4,916,096,045. This suggests that the last generation criterion of 1,300 was appropriate to run the genetic development on a population size of 200. It is very unlikely that a better solution value would have been found by continuing to mate and mutate the two different solutions left in the final population.

Based on the result from Table XIII, it was decided to run the complete set of experiments on a slightly larger population size and perform the genetic development through a larger number of generations. Table XIV shows the results obtained from a population size of 300 through 1,550 generations. The largest improvement in the best solution value was 27.80% from 6,832,788,181; the best (random) solution in the initial population. However, the best solution found was worse than the best solution found for the previous case (population size = 200)

Table XIV. Experiments for the 195-door terminal based on a constant population of size 300

Selection	Crossover	Min Window Size	Fix Window Size	Pm	Run Time (hh:mm:ss)	Best Solution Value	% > (Best Solution Found)	Found in Generation
Fit-Weak	Rnd Win	78		0.20	03:57:28	4,941,398,887	0.16%	1537
Fit-Weak	Rnd Win	78		0.25	03:56:55	4,936,099,171	0.05%	1520
Fit-Weak	Rnd Win	78		0.30	03:57:05	4,986,630,909	1.07%	1498
Fit-Weak	Rnd Win	146		0.20	04:40:22	4,983,860,384	1.01%	1547
Fit-Weak	Rnd Win	146		0.25	04:40:23	5,013,050,146	1.59%	1545
Fit-Weak	Rnd Win	146		0.30	04:39:49	4,969,479,647	0.73%	1550
Fit-Weak	Fix Win		98	0.20	03:11:33	*4,933,392,314*	0.00%	1539
Fit-Weak	Fix Win		98	0.25	03:10:53	4,990,482,859	1.14%	1538
Fit-Weak	Fix Win		98	0.30	03:10:33	5,011,733,600	1.56%	1544
Fit-Weak	Fix Win		146	0.20	04:06:49	4,934,607,471	0.02%	1547
Fit-Weak	Fix Win		146	0.25	04:05:02	4,951,118,602	0.36%	1550
Fit-Weak	Fix Win		146	0.30	04:05:00	5,025,155,702	1.83%	1549
Fit-Fit	Rnd Win	78		0.20	03:55:56	4,955,810,489	0.45%	1550
Fit-Fit	Rnd Win	78		0.25	03:55:49	4,959,941,129	0.54%	1550
Fit-Fit	Rnd Win	78		0.30	03:55:07	4,961,692,834	0.57%	1549
Fit-Fit	Rnd Win	146		0.20	04:38:31	4,934,979,674	0.03%	1549
Fit-Fit	Rnd Win	146		0.25	04:39:07	5,039,233,878	2.10%	1550
Fit-Fit	Rnd Win	146		0.30	04:38:37	4,952,670,172	0.39%	1549
Fit-Fit	Fix Win		98	0.20	03:08:42	5,021,185,161	1.75%	1550
Fit-Fit	Fix Win		98	0.25	03:08:52	4,969,374,987	0.72%	1549
Fit-Fit	Fix Win		98	0.30	03:08:52	5,029,020,073	1.90%	1545
Fit-Fit	Fix Win		146	0.20	04:02:48	5,009,287,767	1.52%	1550
Fit-Fit	Fix Win		146	0.25	04:04:52	4,960,438,380	0.55%	1550
Fit-Fit	Fix Win		146	0.30	04:04:27	4,947,465,271	0.28%	1549

The heuristic nature of genetic algorithms becomes evident in the results from Table XIV. Solution values below 4.950 billion were obtained through both selection and crossover methods; both small and large swap window sizes, and the different values of Pm. This suggests that for larger terminal layouts with more number of doors, it becomes difficult to define the combination of genetic algorithm parameters that will generate the best results.

6.4 2-OPT

2-Opt, a popular optimization technique, was used to validate and compare results. Table XVII presents the best solution values obtained by both optimization techniques for each of the terminal layouts.

Table XVII. Genetic algorithm and 2-Opt results for three representative terminal sizes

Terminal Size	Genetic Algorithm Best Solution Value	2-Opt Best Solution Value
16-Door (small size)	72,839	72,839
43-Door (medium size)	81,842,182	83,612,191

| 195-Door (large size) | 4,916,072,668 | 5,071,565,397 |

As presented in Table XVII, the genetic algorithm performs a more efficient search of the optimum solutions than 2-Opt. The complexity and difficulty of the optimization process increases as the number of doors in the terminal increases. 2-Opt found the same best solution value as the genetic algorithm for the small terminal, 16-doors. As the door assignment problem becomes more complex with a larger terminal, the genetic algorithm outperforms 2-Opt. The genetic algorithm was able to obtain considerably better solutions than those obtained with 2-Opt for the 43 and the 195 door terminals.

7. Conclusions

A genetic algorithm tool was developed to solve the door assignment problem in LTL breakbulk terminals. The door assignment problem requires assigning doors to freight origins and destinations in such a way that the total weighted travel distance is minimized. This minimizes the door-to-door distance that operators travel in proportion to the total weight that is transported between strip and stack doors. As a consequence, the freight handling time (a surrogate for labor cost and cycle time) is minimized.

A few guidelines are proposed for the fine-tuning of the parameters that would provide the best performance of the LTL GA.

- The constant population model is more efficient than the incremental population model.
- Fit-weak selection might provide better results than fit-fit selection when the LTL GA is run for smaller terminal layouts. The choice of selection rule is not clear-cut for larger size terminals (over 40 doors) .
- The population size must be set large enough to maintain diversity since the probability of crossover is set at a fixed value of 1.0 and duplicates cannot be removed.
- The larger the population size, the larger the computation time the LTL GA requires. If computation time is critical, the population size can be reduced. To account for the lack of diversity in smaller population sizes, the probability of mutation, Pm, has to be large enough. Values between 0.15 and 0.30 are recommended.
- In the case of the crossover techniques (random or fixed swap window), it is difficult to determine which technique performs better. It is recommended that the size of the swap windows be equal to or less than 50% of the chromosome length.

8. Future Research

This paper has pointed out some preliminary observations based on limited experimentation. Thus, the key need is for more extensive experimentation on a larger set of test cases.

The following are topics for further research:

- Experimental parameters for the LTL terminals
    - Work with industry to develop a test bank of problems
    - Experiment with different numbers of strip (incoming freight) doors and stack (outgoing freight) doors
    - Experiment with different trailer load diversity and more complex freight flow patterns
    - Trips rather than weight
    - Measure the cost savings that the genetic algorithm solutions to the door assignment problem can generate by minimizing the total weighted distance

- Experimental Parameters for the LTL GA program
    - Review the code of the genetic algorithm software to improve computational performance
    - Modify the total weighted distance objective function to take into account the cost of using material handling equipment in transporting freight between doors
    - Formulate an objective function that would evaluate the effect of interference between material handling equipment and congestion caused by drag line conveyors for a given door assignment
    - Use variable values of Pm when the GA operates under the constant population model and runs for very long number of generations. Large values of Pm in early generations might direct the genetic development in the wrong search for solutions by mutating good solutions at the beginning of the

process (Shaffer, 1999). Larger values of Pm might only be necessary at later generations, as the population diversity is being lost

References

Bermúdez Ruiz, R.A., A Genetic Algorithm Approach to LTL Breakbulk Terminal Door Assignment, M.S. Thesis, University of Arkansas, May 2000.

Chambers LD. (editor), *Practical Handbook of Genetic Algorithms, vol. I.*, CRC Press, Boca Raton, FL, 1995. (a)

Chambers LD. (editor), *Practical Handbook of Genetic Algorithms, vol. II.*, CRC Press, Boca Raton, FL, 1995. (b)

Gue, K., 1995, *Freight Terminal Layout and Operations*, Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, Georgia.

Gue, K., 1999, "The Effects of Trailer Scheduling on the Layout of Freight Terminals," Transportation Science, vol. 33, No. 4, 419-428

Heitkotter, J., and Beassley, D. (editors), 1999, "The Hitch-Hiker's Guide to Evolutionary Computation (FAQ for comp.ai.genetic)," <http://alife.santafe.edu/~joke/encore/www/> (30 September 1999)

Hussain, S., and Sastry, V., 1995, "Application of Genetic Algorithm for Bin Packing," *Computer Math*, 63 (September), 203-214.

Kochhar, J., Foster, B., and Heragu, S, 1998, "HOPE: A Genetic Algorithm for the Unequal Area Facility Layout Problem," *Computers & Operations Research*, 25(7-8), 583-594.

Miagkikh, V.V., Topchy, A.P., Kureichik, V.M., and Tetelbaum, A.Y., 1996, "Combined Genetic and Local Search Algorithm for the Quadratic Assignment Problem," working paper to appear in proceedings of IC on Evolutionary Computation and its Applications, Moscow, June 1996. <http://web/cps/msu.edu/~miagkikh/web/4.ASC> (28 April 1999)

Michalewicz, Zbigniew. *Genetic Algorithms + Data Structures = Evolution Programs.* Second, Extended Edition. Springer-Verlag, 1994

Nissen, V., 1994, "Solving the Quadratic Assignment Problem with Clues from Nature," *IEEE Transactions on Neural Networks*, 5(1), 66-72.

Peck, K, 1983, *Operational Analysis of Freight Terminals Handling Less Than Container Load Shipments*, Ph.D. Dissertation, University of Illinois at Urbana-Champaign, Urbana-Champaign, Illinois..

Reeves, Colin R., *Modern Heuristic Techniques for Combinatorial Problem*. Halsted Press, 1993

Shaffer, Ron., 1999, "*Practical Guide to Genetic algorithms*," Naval Research Laboratory, <http://chem1.nrl.navy.mil/~shaffer/practga.html> (March 20 1999)

Tate, D., and Smith, A. 1994, "A Genetic Approach to the Quadratic Assignment Problem," *Computers & Operations Research*, 22(1), 73-83.

Tavakkoli-Moghaddain, R., and Shayan, E., 1998, "Facilities Layout Design by Genetic Algorithms," *Computers and Industrial Engineering*, 35(3-4), 527-530.

Tsui, L., and Chang, C., 1990, "A Microcomputer Based Decision Support Tool for Assigning Dock Doors in Freight Yards*," Computers and Industrial Engineering*, 19(1-4), 309-312.

Tsui, L, and Chang, C., 1992, "An Optimal Solution to a Dock Door Assignment Problem," *Computers and Industrial Engineering*, 23(1-4), 283-286.