CONSTANTINE PORPHYROGENETUS INTERNATIONAL ASSOCIATION

Journal of Management Sciences and Regional Development
Issue 3, July 2001
Correspondence: ikarkazis@aegean.gr

http://www.stt.aegean.gr/geopolab/GEOPOL%20PROFILE.htm
ISSN 1107-9819
Editor-in-Chief: John Karkazis

GENERALIZED SHORT-STAGE MULTICHANNEL QUEUING MODELS USING GENETIC ALGORITHMS: A REAL-WORLD APPLICATION TO SEAPORTS

Athanasios Tsakonas, Helen Kitrinou, Georgios Dounias

University of the Aegean
Greece

Abstract. This paper introduces genetic algorithms for inducing high-level knowledge from available domain data, succeeding to obtain generalized solutions for a short-stage multi-channel queuing model. The domain of application, refers to the transportation problem of transit storage and re-load in seaports. Specifically, when a ship approaches the port, can be served by more than one service channel, in other words the seaport represents a queuing system. The seaport system forwards ships and lorries into the port, moves vehicles and cranes between two positions i.e. warehouses and berths, and finally loads and unloads cargoes from ships and lorries. Between the two load/unload processes taking place in both, ships and lorries, the transit storage process is embedded, thus forming in fact a three stage multi-channel queuing system. The standard process of working with such a queuing problem supposes Poisson distribution in all the service stages, definition of the service and waiting costs and the construction of an objective function for finding the best-cost solution. The solution produced above is generalized by applying a genetic algorithm approach for finding the best seaport configuration (i.e. optimal number of cranes, warehouses and lorries needed) among a possible set of them, which will offer the minimum seaport operating cost. The paper demonstrates that, when a set of possible configurations is effectively coded into a genetic population, the best solution might be achieved in a reasonably short time and well approximated.

Keywords: queueing systems, genetic algorithms, computational intelligence, seaport operating cost optimization, transportation problems.

1. INTRODUCTION

International trends focus our attention on international trade more than ever before. Marine transportation systems play a central role in the trade scene, and various complex problems of port planning and port operations management are marked out as major issues. The transfer of goods by both, sea and land transportation into a port, is usually represented by a chain of links. The pattern of arrivals and service times of ships, often seems to follow well-known probability distributions. This justifies the application of queuing theory to problems of port operations and capacity investments (Evans, 1986) with serious positive impact to optimization of port economics and managerial decision making problems (Janson and Shneerson, 1982), (Chaudhry and Templeton, 1983). The conventional queuing theory based approach (Saaty, 1961), (Cox and Smith, 1961), (Takacs, 1962), (Prablu, 1965), (Cohen, 1969), (Conolly, 1975), (Kleintock, 1975), (Gross and Harris, 1976), requires definition of some more or less standard parameters, such as arrival pattern of customers, service times, capacity of the system, etc. When a ship approaches the port, the cargo must be served by more than one "service facilities", i.e. the seaport is equivalent to a short-stage queuing system. The seaport system includes the following basic activities:

1. Forwarding ships and land transport vehicles into the port.
2. Moving vehicles and cranes between two positions, i.e. warehouses and berths.
3. Loading /unloading cargo from ships and land transport vehicles.

Between the two loading /unloading stages taking place in both, ships and land transport vehicles, the transit storage process is embedded, thus forming in fact a three-stage multi-channel queuing system.

Furthermore, a combined transport problem is possible to occur, extending our system under consideration, into a short-stage multi-channel queuing model. Within this paper, the standard process of working with the queuing problem described above, supposes a Poisson distribution in all the service stages, followed by the definition of the service and waiting costs. An objective function is then formed having as decision variables the number of cranes, warehouses and lorries needed for finding the best-cost solution. An obvious weakness of the overall approach – addressed but not resolved within this paper - lies into the accurate definition of the waiting costs for different kinds of ships and cargoes.

The overall queuing system approach described above is precise and reliable but is limited from the obvious complexity involved in calculations as stages and service points increase. In recent years, a large variety of methods has appeared, competitive to standard optimization techniques and operations research. Most of these methods are heavily computer-based, data and not model driven (Cherkassky and Mulier, 1998), and contain search methods based in heuristics or innovative algorithmic schemes (Mars, et. al,

1996). They are usually characterized under the term of "computational intelligence" applications (Zimmermann, et. al, 2001), (Chen, 2000), (Nilsson, 1998) and they are particularly efficient in all domains of application containing high degree of uncertainty, complexity, vagueness, or lack of precise and complete data. Amongst the whole family of methods, machine learning (Witten and Frank, 2000), (Pyle 1999), genetic algorithms (Man et. al, 1999), (Chambers, 1999), as well as, neural networks, soft computing and fuzzy logic (Konar, 1999) are found in literature to be the most famous and widely. It should be claimed that the set of computational intelligence tools and techniques described in detailed within the recent literature, complements the variety of standard quantitative methods for management contained in the operations research field (Winston, 1997), (Anderson, et. al, 1999). Note here that computational intelligence is not exactly competitive to OR techniques, in the sense that when a problem can be approached through OR always offers a better quality solution. Yet, computational intelligence should remarkably contribute complementary to OR methods, by reducing processing times. It also offers the ability for generalization from available data, and the "intelligence" contained in algorithms and computer programs for efficient, fast and reliable search of a very large solution space, usually intractable by standard OR approaches. The increase of the number of research papers focusing on new domains of application, such as time series forecasting, chaotic data modeling and analysis and qualitative data analysis proves that fact. A strong potential of computational intelligence is observed in complex real-world domains such as stock exchange forecasting (Tsakonas, et. al, 2000), fault diagnosis in complex dynamic systems, medical imaging and signal processing, (Zimmermann, et. al, 2001), etc.

Within this paper the authors propose the utilization of computational intelligence and operations research, in order to fast and efficiently produce a generalized solution for the seaport queuing system described in the previous paragraph. This is done by generalizing the solution produced from the queuing theory based approach, by applying a genetic algorithm approach for finding the best seaport configuration (i.e. optimal number of cranes, warehouses and lorries needed) among a possible set of them, which will offer the minimum seaport operating cost. Although -according to queuing theory- the cost function is well defined within the paper, a mathematical solution of the problem would likely drive the result into real values, a case that requires further painful computational effort. Therefore, a one step and easy to implement approach -such as a genetic algorithm technique- seems more desirable in real-world cases which are constituted of many parameters and consequently a large solution space. The paper demonstrates that, when a set of possible configurations is effectively coded into a genetic population, the best solution might be achieved in a reasonably short time and well approximated.

The rest of the paper is organized as follows:
Section 2 analyzes the operating characteristics of the seaport queuing system. Section 3 presents the mathematical formulation of the model related to the queuing system. In Section 4 the genetic algorithms based approach is described. Results are extensively presented and discussed in Section 5. The paper concludes in Section 6, where further research directions are addressed.

2. THE OPERATING CHARACTERISTICS OF THE SYSTEM

The queuing system described above is based on two assumptions, which are common to most applications of queuing theory to port operations:

- The pattern of arrivals at all three stages of the system follows a Poisson probability distribution, with the same mean arrival rate λ for all stages ("equivalence property" of multi-stage queuing systems).
- The service time at all three stages of the system follows an exponential probability distribution. Let $\mu_i, i = 1,2,3$ be the mean service rate for each channel, at stage i=1,2,3.

We use the following notation:
N_i = the number of service stations at stage i=1,2,3.

- A service station in stage 1 is a crane (or a berth)
- A service station in stage 2 consists of a part of the total storage area sufficient to hold a representative shipload.
- A service station in stage 3 is a land transport vehicle.

ϕ_i = the occupancy rate of stage i=1,2,3.

We have: $\phi_i = \dfrac{\lambda}{N_i.\mu_i}$, i=1,2,3.

We also suppose that all the three stages of the queuing system are in statistical equilibrium. That means: $\phi_i = \dfrac{\lambda}{N_i.\mu_i} < 1$, i= 1,2,3.

The probability of no units in the stage i= 1,2,3, is:

$$p_{0i} = \frac{N_i!(1-\phi_i)}{(\phi_i N_i)^{N_i} + N_i!(1-\phi_i)\sum_{n=0}^{N_i-1}\frac{1}{n!}(\phi_i N_i)^n}$$

The probability of n units in the stage i= 1,2,3, is:

$$p_{ni} = \frac{(\phi_i . N_i)^n}{n!} p_{0i}, n = 1,2,\ldots, N$$

$$= \frac{N_i^{N_i}}{N_i!}\phi_i^n . p_{0i}, n > N$$

The average number of units in stage i=1,2,3, is:

$$L_i = \frac{\phi_i(\phi_i . N_i)^{N_i}}{N_i!(1-\phi_i)^2} . p_{0i} + \phi_i . N_i$$

3. THE MODEL

The problem is to determine the number of the required service stations of the three stages defined above. This case in which service can be supplied to an arriving customer (cargo), only if all three succeeding service stations are unoccupied (free), is not covered by the existing models in standard queuing theory. The next step is therefore to develop a model for the three-stage case, in order to derive a formula to estimate the optimum number of service stations in each stage.

The objective is to minimize the sum of expected total service cost, E(TSC) (that depends on the number of service stations) and the expected total waiting cost, E(TWC) (that depends on the number of customers (cargoes) in the system).

$$\min E(TC) = \min\{E(TSC)+E(TWC) \qquad (1)$$

If C_i is the service cost per time period for each service station at stage i=1,2,3 then the expected total service cost is:

$$E(TSC) = N_1.C_1 + N_2.C_2 + N_3.C_3$$

Let $g_i(n), i = 1,2,3$ be the function of waiting cost for one cargo per day and let n be the number of cargoes in each stage of the system. If we assume that $g_i, i = 1,2,3$, represent linear functions, we have:

$$g_1(n) = 0, if (0 \le n \le N_1)$$
$$\equiv C_{w1}.(n - N_1), if (n > N_1)$$
$$g_2(n) = 0, if (0 \le n \le N_2)$$
$$\equiv C_{w2}.(n - N_2), if (n > N_2)$$
$$g_3(n) = 0, if (0 \le n \le N_3)$$
$$\equiv C_{w3}.(n - N_3), if (n > N_3)$$

when C_{w1}, C_{w2}, C_{w3} are the expected waiting costs per time period for each unit, at stages 1,2,3. An obvious weakness of the approach lies into the accurate definition of the waiting costs for different kinds of ships and cargoes.

The expected total waiting cost is:

$$E(TWC) = E(g_1(n)) + E(g_2(n)) + E(g_3(n))$$
$$= \sum_{n=0}^{\infty} g_1(n).p_{n1} + \sum_{n=0}^{\infty} g_2(n).p_{n2} + \sum_{n=0}^{\infty} g_3(n).p_{n3} = \sum_{i=1}^{3}\sum_{n=0}^{\infty} g_i(n).p_{ni}$$

$$= \sum_{i=1}^{3} C_{wi} \cdot \sum_{n=N_i+1}^{\infty} (n - N_i) p_{ni} =$$

$$= \sum_{i=1}^{3} C_{wi} \cdot \sum_{n=N_i+1}^{\infty} n p_{ni} - \sum_{i=1}^{3} C_{wi} \cdot N_i \cdot \sum_{n=N_i+1}^{\infty} p_{ni} =$$

$$= \sum_{i=1}^{3} \left[C_{wi} \cdot \left(\sum_{n=0}^{\infty} n.p_{ni} - \sum_{n=0}^{N_i} n.p_{ni} \right) \right] - \sum_{i=1}^{3} \left[C_{wi} \cdot N_i \cdot \sum_{n=N_i+1}^{\infty} \frac{N_i^{N_i}}{N_i!} \cdot \phi_i^{\,n} \cdot p_{0i} \right] =$$

$$= \sum_{i=1}^{3} \left[C_{wi} \cdot L_i - \sum_{n=0}^{N_i} n.p_{ni} \right] - \sum_{i=1}^{3} \left[C_{wi} \cdot \frac{N_i^{N_i}}{(N_i - 1)!} \cdot p_{0i} \cdot \sum_{n=N_i+1}^{\infty} \phi_i^{\,n} \right] =$$

$$= \sum_{i=1}^{3} \left[C_{wi} \cdot L_i - \sum_{n=0}^{N_i} n.p_{ni} \right] - \sum_{i=1}^{3} \left[C_{wi} \cdot \frac{N_i^{N_i}}{(N_i - 1)!} \cdot p_{0i} \cdot \frac{\phi_i^{\,N_i+1}}{1 - \phi_i} \right]$$

The objective function (1) is thus:

$$\min E(TC) = \min \left\{ \sum_{i=1}^{3} C_i \cdot N_i + \sum_{i=1}^{3} \left[C_{wi} \cdot L_i - \sum_{n=0}^{N_i} n.p_{ni} \right] - \sum_{i=1}^{3} \left[C_{wi} \cdot \frac{N_i^{N_i}}{(N_i - 1)!} \cdot p_{0i} \cdot \frac{\phi_i^{\,N_i+1}}{1 - \phi_i} \right] \right\}$$

Given: $\lambda, \mu_1, \mu_2, \mu_3, C_1, C_2, C_3, C_{w1}, C_{w2}, C_{w3}$
Decision variables: N_1, N_2, N_3

4. GENETIC ALGORITHM

In order to find the "low cost" configuration, a standard genetic algorithm has been implemented. Genetic algorithms have been generally proved capable of solving problems where a direct search method is insufficient or non-existent (Chambers 1999). In the problem discussed through this paper, a mathematical solution of finding the roots of the cost function (using a high-level tool such as *Mathematica*[1]) for the simpler problem presented in the paper (a system of three stages), would offer the real values, requiring by the user to estimate the integer ones, a procedure which may be hard to accomplish, due to the fact that the immediate rounding of the real values does not necessarily offer the best configuration. Moreover, when encountering such complex functions, having much more than two variables and complex restrictions in the allowed solution values (as this is the second case in our problem), computer-assisted mathematical tools will be probably unable to solve it, even for computing the real values of the best configuration. On the other hand, for large problems, a direct and exhaustive search among possible solution space would be an awkward and trivial technique while it can be proved a very time-consuming and eventually ineffective procedure, as

[1] Trademark of Wolfram Research Co.222

demonstrated for the second example in Sections 4 and 5. In the rest of this paragraph, the genetic algorithm specifications built for the needs of this paper are described.

4.1 Encoding

For the chromosome encoding, the algorithm first calculates the maximum value range of all stages. This range (which is an integer), for each stage, is the difference between maximum allowed units and minimum allowed units plus one. It then computes the bits needed to represent this maximum value range, by using the following well-known formula:

$$b = x : 2^{x} \quad \tau \sim 2^{x-1}$$

where τ is the maximum range and b is the number of bits. It creates then a chromosome consisting of b* S where S is the number of the stages. The latter means that in the chromosome representation all stages have an equal number of bits to be represented. This selection has been decided, in order to ensure the GA-efficiency during basic operations, i.e. crossover and mutation. As long as different stages have different range of values, a checking sub-procedure was added after each genetic operation, enabling only allowed values for each range. Encoding only the permissible range of the stages, and not the maximum value of these stages, enables the algorithm to handle more general problems effectively. For example a case with 10 stages that have a stage with allowable values between 240 and 248 would require a chromosome of 80 bits ($=10 * log_2 256$) if coded using the possible values, but it requires only 30 bits in the configuration presented. In Figure 1, the structure of the chromosome is presented.

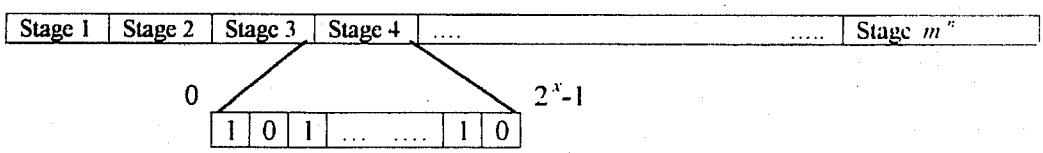
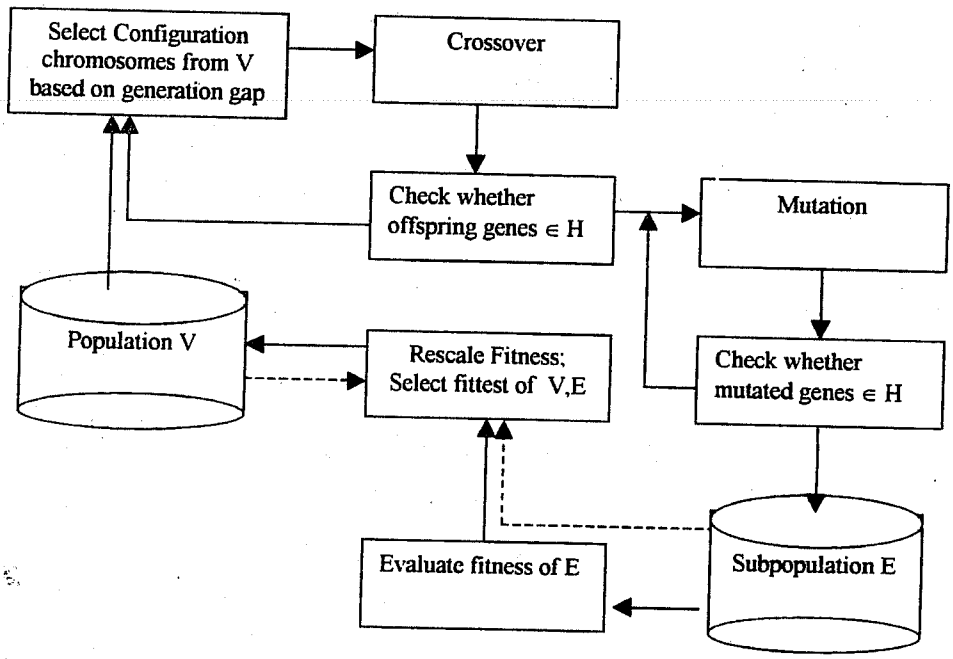


Figure 1. Chromosome encoding.

4.2 Operators

The operations applied in the genetic algorithm were (1) recombination, (2) crossover and (3) mutation. The selection procedure was the stochastic sampling with replacement (SSR). We re-scaled the fitness values after each epoch in order to avoid premature convergence, hence speed up the search (Baker, 1987). After experiments with various operation rates, it was shown that a 0.8 rate of crossover and a 0.2 rate of mutation, for the populations considered in our examples was an effective operation scheme (Man 1999).

Figure 2. The Genetic process cycle.



Trials using different types of crossover showed that using "uniform crossover" outperformed other crossover types such as "single" or "double point crossover" or our heuristic "variable crossover" (Tsakonas et. al., 2000), (Tsakonas and Dounias, 2000). The reason is that the encoding of the chromosome requires few allele size (bits) for internal stage representation, in which case a uniform crossover may achieve higher search performance. Changing the generation gap from 0.5 was not encountered, as long as tuning the rest of the parameters offered an efficient performance. The genetic process cycle is presented in Figure 2, where a set H is assumed to be consisting of the permissible values of each stage.

4.3 Software

A genetic algorithm standalone MDI[2] application named "*QueueGA*" was built in order to support experimentation for this paper. The software is a windowed application enabling the user to fully describe the problem and freely set the genetic parameters.

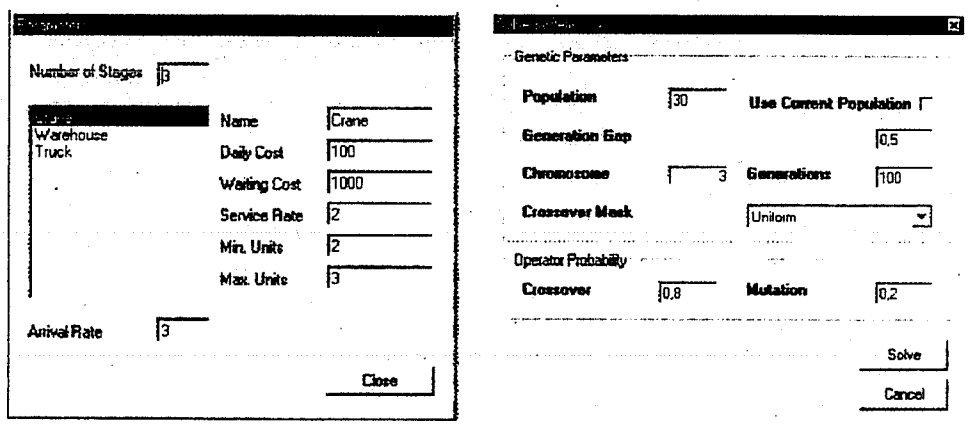


Figure 3. System parameters input screen (left), and genetic parameters setup screen (right).

Furthermore the user has the possibility to direct the output to a specified file for further processing (i.e. graphs, statistics etc.). Figure 3 shows screenshots of two menus. The reader may download a free version of "*QueueGA*" in the web in the web address www2.aegean.gr/QueueGA

5. RESULTS AND DISCUSSION

In the example shown in Table 1, a first run of the algorithm is demonstrated, using as input the data presented on the previous paragraph, i.e. a seaport queue consisting of cranes, warehouse and trucks. The arrival rate of ships is considered λ=3 ships/day. The program computation makes indeed the problem trivial. It takes less than one second in order to find the best solution, therefore it is shown here rather for the proof of the methodology accuracy.

[2] Multi-Document-Interface

Table 1. Parameters of seaport problem

Stage #	Name	Daily Cost	Working Cost	Service Rate	Min.Units Allowed	Max.Units Allowed
1	Crane	100	1000	2	2	3
2	Warehouse	50	200	4	3	4
3	Truck	40	200	3	2	3

Table 2. GA solution for the seaport problem

Name	Low Cost Configuration
Crane	3
Warehouse	3
Truck	2

After Generation	100
Cost	2255.03979618163

Table 3. Genetic Algorithm parameters for the seaport problem.

Genetic Parameters	
Chromosome	3
Population	5
Generation Gap	0.5
Crossover Mask	Variable
Crossover Rate	0.8
Mutation Rate	0.2
Generations	100
Time taken for 100 generations in P-200MHz	Less than 1 sec

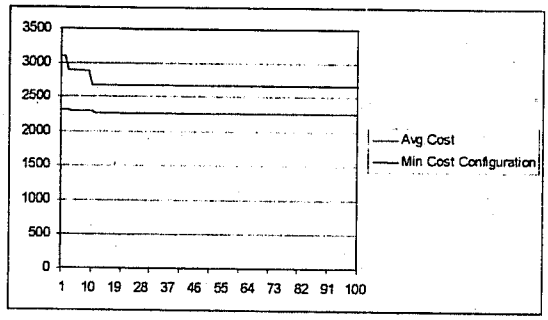


Figure 4. Average population cost and minimum cost of best chromosome for seaport problem, evolving in time.

Tables 2 and 3 refer to the genetic algorithm (GA)-solution and the corresponding GA-parameters used for the simple seaport problem. The best configuration contains three cranes three warehouses and two trucks , and after one hundred of generations is found the optimum cost value of this solution. Finally, Figure 4 represents the variation of the average population cost and the minimum cost of the best chromosome for the simple seaport example, as it evolves over time.

In the next example (Table 4), a more complex problem is encountered, consisting of 20 stages, each of them having different allowable minimum and maximum values in the configuration. Minimum allowable units for the configuration may correspond to existing infrastructure, and maximum allowable units may depend on the organization's budget. Arrival rate of jobs is also considered λ=3 jobs/day. Less than 100 generations were needed in order to find the best solution, roughly corresponding to one minute computing working-time (for a P200MHz PC).

Many real-world problems (e.g. communication routing) may consist of even larger sizes, rendering the exhaustive search among solutions an ineffective method. In fact, the possible solution space increases based on the following equation:

$$\prod_{i=1}^{N} m_i$$

where N is the number of stages, and m_i is the values' range of stage i. For example, the second complex example presented in Table 4, creates a possible solution space consisting of 696,570,347,520,000 values.

Table 4. Parameters of a complex example.

Stage #	Name	Daily Cost	Working Cost	Service Rate	Min.Units Allowed	Max.Units Allowed
1	Stage 1	399,28	537,17	6,71	5	8
2	Stage 2	438	396,31	6,14	9	17
3	Stage 3	364,7	400,94	2,28	9	16
4	Stage 4	546,28	439,08	6,52	4	7
5	Stage 5	240,68	489,63	2,96	6	14
6	Stage 6	425,1	506,57	5,23	5	9
7	Stage 7	289,03	533,53	9,99	5	9
8	Stage 8	489,35	457,22	8,59	7	12
9	Stage 9	455,71	235,29	4,78	3	7
10	Stage 10	217,34	284,25	7,31	3	5
11	Stage 11	504,71	268,6	4,96	5	9
12	Stage 12	240,02	447,91	8,59	2	10
13	Stage 13	528,36	547,11	4,28	8	15
14	Stage 14	525,58	321,74	5,33	2	5
15	Stage 15	296,74	376,2	7,48	4	7
16	Stage 16	271,83	315,47	4,76	3	11
17	Stage 17	408,67	400,04	5,48	5	10
18	Stage 18	418,88	536,85	8,72	9	16
19	Stage 19	306,74	356,46	2,05	3	4
20	Stage 20	594,67	555,23	7,87	7	12

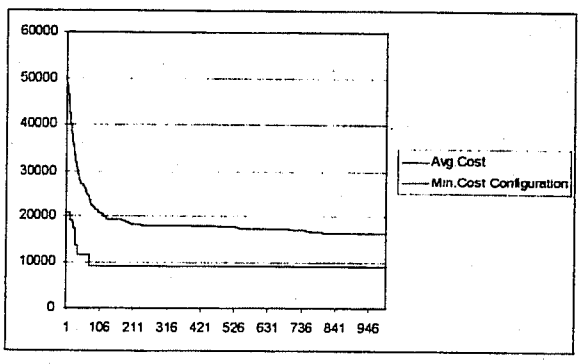


Figure 5. Average population cost and minimum cost of best chromosome for the complex example, evolving in time.

Assuming that all these values must be calculated in an exhaustive search for computing the best solution -requiring a P200MHz PC to work for more than a year- the reader may acknowledge the efficiency of the GA-approach. Based on the results of this second example, only 5,040 value computations[3] where needed, in less than one minute of time, in order for the algorithm to find the best solution[4].

Table 5. GA solution for the complex example.

Name	Min. Cost Configuration (Units)
Stage 1	8
Stage 2	16
Stage 3	16
Stage 4	7
Stage 5	12
Stage 6	5
Stage 7	5
Stage 8	7
Stage 9	4
Stage 10	4
Stage 11	8
Stage 12	9
Stage 13	15
Stage 14	4
Stage 15	4
Stage 16	4
Stage 17	8
Stage 18	16
Stage 19	3
Stage 20	11

After Generation	1000
Cost	9013,302036

[3] The best solution was found after 72th generation using a population of 70 possible solutions (chromosomes)

In all cases, the accuracy of the solution is easily ensured, either by incrementing the population size, or by subsequent runs of the algorithm. Thus, in response to either human work or computer exhaustive search, genetic algorithms prove capable of handling domain problem sizes, efficiently, i.e. fast, as well as accurately.

Table 6. Genetic Algorithm parameters for the complex problem.

Genetic Parameters	
Chromosome	60
Population	70
Generation Gap	0.5
Crossover Mask	Variable
Crossover Rate	0.8
Mutation Rate	0.2
Generations	1000
Time taken for 1000 generations in P200MHz	Less than 11 min

Tables 5 and 6 contain additional information about the solution and the corresponding GA-parameters used for the complex example analyzed before. Finally, Figure 6 again represents the average population cost and the minimum cost of the best chromosome for the complex example, as it evolves over time.

6. CONCLUSIONS AND FURTHER RESEARCH

The paper demonstrated how genetic approximation techniques can be embedded into short-stage multi-channel queuing models applying to seaports. The proposed methodology proved efficient, accurate and fast for the requested task. Further research is directed into many different aspects of theory and applications:

In relation with the genetic approximation methodology used, the authors suggest that an attempt for tuning of the variable crossover should be made. Due to the small deviations of all possible configurations from the best one, the variable crossover achieved in doubling the crossover points only once in the second example. Therefore tuning is necessary, in order to achieve an even better performance, probably reducing the necessary error decrease in order to double the points. In the examples presented within this paper, it had a value of (0.5), meant that the error -which is equal to the best

[4] It should be noted that the time-consuming computations within a genetic algorithm are the objective values computation.

configuration cost found when the algorithm runs- should be decreased by half to double the points. This way an almost "uniform-mask" will be resembled, just before finding the best stages configuration.

In respect to the queuing system based approach, admissions made within this paper about major factors involved in the calculations such as, distributions of service, processing times, etc., should be explored also with possible alternative initial configurations.

Generalizing, the suggested genetic algorithm based approach, should be taken into account in any other similar types of queuing / operations research problems related to seaport transportation, especially those having high complexity and demanding painful computational effort. As modern computer capabilities (processing speed, data storage, memory capacity) increase dramatically over time, most OR techniques combined with proper computational intelligence algorithms, will soon have the opportunity to produce accurate computer-aided solutions for very high complexity domains of application. Intelligent OR techniques and data-driven methods for management seem to be on the way to success, and it is very likely that they will soon become the method of choice for modern operations management era.

7. REFERENCES

Baker, J. E. (1987), Reducing bias and inefficiency in the selection algorithms, Proceedings of the 2nd International Conference on Genetic Algorithms, Lawrence Erlbaum Assoc., Hillsdale, NJ, 14-21.

Chambers, L. D. (1999), *Genetic Algorithms: Complex Coding Systems, Vol.III*, CRC Press.

Man, K. F. , Tang, K. S., and Kwong. S. (1999), *Genetic algorithms: concepts and designs*, Springer- Verlag, London, 1999.

Mars P., Chen J. R., Nambiar R. (1996), *Learning algorithms: theory and applications in signal processing, control and communications*, CRC Press.

Tsakonas A., Dounias G., and Merikas A. (2000), The Role of Genetic Algorithms and Wavelets in Computational Intelligence-based Decision Support for Stock Exchange Daily Trading, in Proceedings of VII Congress of SIGEF, 195-208.

Janson, J. O. and Shneerson, D. (1982), *Port Economics*, MIT Press Series in Transportation Studies

Evans J. J. (1986), *Quantitative Methods in Maritime Economics*, Second Edition,

Fairplay Publications

Winston W. I. (1997), *Operations Research, Applications and Algorithms*, Third Edition, Duxbury Press

Anderson D. R. , Sweeney D. J., Williams T. A. (1999), *An Introduction to Management Science - Quantitative Methods to Decision Making*, Ninth Edition, West Publishing Company

Cox R. D., and Smith , W. L. (1961), *Queues*, Methuen.

Saaty, T.L. (1961), *Elements of Queuing Theory*, McGraw Hill, New York

Prablu, N.U. (1965), *Queues & Inventories: a study of their basic Stochastic Processes*, Wiley, New York

Cohen, J.W (1969), *The Single Server Queue*, North Holland Amsterdam

Conolly B. R. (1975), *Queuing Systems*, Ellis Horwood, Chichesber

Takacs, L. (1962), *Introduction to the theory of Queues*, Oxford University Press, New York

Kleintock, L. (1975), *Queuing Systems*, Wiley, New York

Gross D. and Harris C. M. (1976), *Fundamentals of Queuing Theory*, Wiley, New York

Chen Z. (2000), *Computational Intelligence for Decision Support*, CRC Press

Cherkassky V., and Mulier, F. (1998), *Learning from Data: Concepts, Theory, and Methods*, Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control, USA.

Witten I., Frank E. (2000), *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufmann Series in Data Management Systems

Konar A. (1999), *Artificial Intelligence and Soft Computing: Behavioral and Cognitive Modeling of the Human Brain*, CRC Press.

Pyle D., (1999), *Data Preparation for Data Mining*, Morgan Kaufmann.

Nilsson N. (1998), *Artificial Intelligence: A New Synthesis*, Morgan Kaufmann.

Zimmermann H-J., Tselentis G., Van Someren M., Dounias G. (Eds.) (2001), *Advances in Computational Intelligence and Learning: Methods and Applications*, Kluwer Academic Publishers (in print, to appear summer 2001)

Man K. F., Tang K. S. and Kwong S. (1999), *Genetic algorithms: concepts and designs*, Springer-Verlag, London

Koza J. R. (1992), *Genetic Programming - On the Programming of Computers by Means of Natural Selection*, The MIT Press

Chambers L. D. (1999), *Practical Handbook of Genetic Algorithms, Complex Coding Systems Vol. III*, CRC Press

Chaudhry M. L. and Templeton- James G.C. (1983), *A First Course in Bulk Queues.* Wiley. New York