Self-Adaptive Fitness Formulation: Theory and Application

R. FARMANI *, J.A. WRIGHT, G.A. WALTERS and D.A. SAVIC
* Department of Engineering
University of Exeter
Exeter, EX4 5QF
UNITED KINGDOM

Abstract: - A self-adaptive fitness formulation is presented for constrained evolutionary optimization. The method has been formulated to ensure that slightly infeasible solutions with a low objective function value remain fit. This is seen as a benefit in solving highly constrained problems that have solutions on one or more of the constraint bounds. In contrast, solutions furthest from the constraint bounds are seen as containing little genetic information that is of use and are therefore penalized. The dimensionality of the problem is reduced by representing the constraint violations by a single infeasibility measure. The infeasibility measure is used to form a two-stage penalty that is applied to the infeasible solutions. The performance of the method has been examined by its application to a set of eleven test cases from the specialized literature and also by its application to a water distribution network from literature. The results have been compared with previously published results from the literature. It is shown that the method is able to find the optimum solutions with less computational effort. The proposed method requires no parameter tuning and can be used as a fitness evaluator with any evolutionary algorithm. The approach is also robust in its handling of both linear and nonlinear equality and inequality constraint functions. Furthermore, the method does not require an initial feasible solution.

Key-Words: - Constraint handling, evolutionary algorithm, fitness, penalty, self-adaptive, real world problems

1 Introduction

In the last two decades, genetic algorithms have received much attention regarding their potential as global optimization techniques. More recently, the solution of constrained optimization problems has been addressed by many researchers ([1], [2], [3], [4], [5], [6], [7]).

Penalty function methods are among the most common methods used to solve constrained optimization problems. In these methods, a penalty term is added to the objective function, the penalty increasing with the degree of constraint violation (static penalty) or the degree of constraint violation and generation number (dynamic penalty) ([8], [9]). In general the weakness of penalty methods is that they often require several parameters, to adjust the relative weights of each constraint in the penalty, and the weight of the penalty against the objective function. However, due to their simplicity and ease of implementation they are the most common methods used in solving real world problems.

Among other types of constraint handling methods are the specialized operators method (GENOCOP method which is based on designing specialized operators that incorporate knowledge of the constraints) [10] and the methods based on the idea of repairing infeasible solutions [11]. Superiority of feasible solutions method [1] is another method for handling constrained problems.

Hybrid methods combine evolutionary techniques with deterministic optimization procedures for numerical optimization problems [12].

Adaptive techniques are appealing due to their ability to adjust their own parameters and make use of information in the population [2][4].

Koziel and Michalewicz (1999) presented the homomorphous mapping approach for solving constrained optimization problems. The method incorporates a homomorphous mapping between an n-dimensional cube and the feasible search space. The disadvantages of the homomorphous mapping method are that it requires an initial feasible solution and that all infeasible solutions are rejected. Another limitation is the need for problem-dependent parameters in the method.

Runarsson and Yao (2000) introduced a stochastic ranking method in which the objective function values are used for ranking the solutions in the infeasible region of the search space. A

probability parameter is used to determine the likelihood of two individuals in the infeasible space being compared to each other. Although the method proved to be effective in solving a wide range of constrained optimization problems, it was also sensitive to the choice of probability parameter.

Most of these constraint handling methods are problem dependent. They often require user supplied parameters to be adjusted in order to obtain good performance from the method. Some of the methods are also able to handle only specific constraint types and therefore lack generality. Some of the approaches limit the search to the feasible search space. However, a good search should approach the optimum solution from both sides of the feasible/infeasible border [13]. Smith and Coit (1997) pointed out that there is need for development of completely adaptive penalty functions that require no user specified constants and development of improved adaptive operators to exploit characteristics of the search as they are found.

Wright and Farmani (2001) and Farmani and Wright (2003) presented a fitness formulation which addresses the limitations of some of the existing constraint handling methods. In particular, it does not require parameter tuning and can be used without an initial feasible solution being given. In what follows a brief description of the self-adaptive fitness formulation [7] is presented for constraint evolutionary optimization. The general procedure for the method is illustrated by a flowchart, which summarizes possible scenarios. The paper describes the evaluation of the algorithm's performance in solving eleven standard test functions taken from the literature. The performance of the algorithm is compared to a number of previous studies. The method is also applied to a benchmark water distribution network and the results are compared with previously published results from literature.

2 Self-Adaptive Fitness Formulation

The self-adaptive fitness formulation method for constraint optimization was proposed by Wright and Farmani (2001) and Farmani and Wright (2003). The proposed formulation has the advantage of being a self-adaptive method, where all the information from the search space is used. The method has been formulated to ensure that slightly infeasible solutions with a low objective function value remain fit. This is seen as a benefit in solving highly constrained problems that have solutions on one or more of the constraint bounds. In contrast, solutions furthest from the constraint bounds are seen as containing little genetic information that is of use and are therefore penalized. The approach does not require parameter tuning, can be used without any initial feasible solution being given and requires fewer function evaluations (this being an advantage in real world applications having many optimization variables). The approach is also robust in its handling of both linear and nonlinear equality and inequality constraint functions. The self-adaptive fitness formulation method can be used as a fitness evaluator with any evolutionary algorithm.

The algorithm has three stages, first each individual is assigned an infeasibility, second the bounding solutions of the search space are identified, and finally, the infeasible solutions are penalized.

2.1 Chromosome infeasibility

The infeasibility values are represented by the sum of the normalized constraint violation values.

$$\iota(X) = \frac{\sum_{j=1}^{m} \frac{c_j(X)}{c_{\max,j}}}{m} \qquad (1)$$

where $\iota(X)$ is the solution's infeasibility, $c_j(X)$ is the value of the constraint violation of individual X for constraint j, $c_{\max,j}$ is the maximum value of the constraint violation for constraint j in the current population, and m is the number of constraints.

The infeasibility measure has the properties that it increases in value with both the number of active constraints and the extent to which each constraint is violated. The constraint violation values are normalized since large differences in the magnitude of the constraint values can lead to dominance of the infeasibility by constraints having the highest values. The scaling factor for each constraint $c_{\max,j}$, is taken as the maximum value of the constraint violation in the current population. Resetting the scaling factor for each population provides a further dynamic element to the infeasibility calculation.

2.2 Identification of the bounding solutions

The penalty functions are applied in relation to three bounding solutions:

$\overset{\vee}{X}$, the "best" individual;

$\hat{X}$, the "worst" of the infeasible solutions; and

$\overset{\cup}{X}$, the solution with the highest objective function value in the current population.

For a population containing at least one or more feasible solutions, the "best" individual $\overset{\vee}{X}$, is the feasible solution having the lowest objective function value. However, if all individuals are infeasible then the best solution is taken as the solution having the lowest infeasibility value (regardless of the objective function value of the individuals).

The "worst" of the infeasible solutions $\hat{X}$, is selected by comparing all infeasible individuals against the best individual ($\overset{\vee}{X}$). Two potential population distributions exist in relation to this comparison.

- The first population distribution occurs when **one or more** of the infeasible solutions have an objective function value that is **lower** than the "best" solution. In this case, the "worst" of the infeasible solutions is taken as the infeasible solution having the highest infeasibility value and an objective function value that is lower than the "best" solution's. If more than one individual exists with the same highest infeasibility values, then the $\hat{X}$ is taken as the solution with maximum infeasibility value and the lower of the objective function values.
- The second population distribution occurs when **all** of the infeasible solutions have an objective function value that is **greater** than the "best" solution. Here the "worst" of the infeasible solutions is identified as being the solution with the highest infeasibility value. If more than one individual exists with the same highest infeasibility value, then $\hat{X}$ is taken as the solution with the maximum infeasibility value and the higher of the objective function values.

2.3 Chromosome Fitness

The infeasibility measure is used to form a two stage dynamic "penalty" applied to the infeasible solutions. The penalties applied to the infeasible solutions are a function of the infeasibility and objective function values $f(X)$ for the "best" ($\overset{\vee}{X}$) individual, the individual with maximum infeasibility value, "worst" ($\hat{X}$), and the individual with maximum objective function value ($\overset{\cup}{X}$) in the current population. Note that if a feasible solution exists, then the "best" solution is feasible and will have zero infeasibility, otherwise the "best" solution is the infeasible individual with minimum infeasibility value. The first stage only applies if one or more infeasible solutions have a lower, and therefore, potentially better objective function value than the "best" solution. The first penalty ensures that the worst of the infeasible solutions (the "worst" individual) has a penalized objective function value $\dot{f}(X)$ that is higher or equal to that of the best solution in the population. The remaining infeasible individuals are then penalized in proportion to their infeasibility. This has been implemented using a simple linear relationship between the objective function values and the infeasibility of the "best" and "worst" solutions (Equations 2 and 3).

$$\tilde{\iota}(X) = \frac{\iota(X) - \iota(\overset{\vee}{X})}{\iota(\hat{X}) - \iota(\overset{\vee}{X})} \tag{2}$$

$$\dot{f}(X) = f(X) + \tilde{\iota}(X)(f(\overset{\vee}{X}) - f(\hat{X})) \tag{3}$$

Note that if the penalty is not applied then $\dot{f}(X) = f(X)$.

The second penalty increases exponentially the first penalized objective function values such that the second penalized objective function value $\ddot{f}(X)$ of the "worst" individual is equal to that of the individual with maximum objective function value in the current population $(\overset{\cup}{X})$ (Equations 4 and 5).

$$\ddot{f}(X) = \dot{f}(X) + \gamma \left| \dot{f}(X) \right| \left(\frac{\exp(2.0\ \tilde{\iota}(X)) - 1.0}{\exp(2.0) - 1.0} \right) \tag{4}$$

$$\gamma = \begin{cases} \dfrac{f(\overset{\cup}{X}) - f(\overset{\vee}{X})}{f(\overset{\vee}{X})}, & if\ (f(\hat{X}) \le f(\overset{\vee}{X})) \\ 0.0, & if\ (f(\hat{X}) = f(\overset{\cup}{X})) \\ \dfrac{f(\overset{\cup}{X}) - f(\hat{X})}{f(\hat{X})}, & if\ (f(\hat{X}) > f(\overset{\vee}{X})) \end{cases} \tag{5}$$

It is important to note that the exponential weighting parameter of 2.0 in Equation 4, is a constant and does not require tuning. The exponential function gives a slight reduction in the rate of penalty applied to solutions of low infeasibility, thus helping to maintain the fitness of the slightly violated solutions. A study of the effect of the weighting parameter indicated that the algorithm was insensitive to the parameter provided that the parameter values had the effect of only slightly reducing the penalty weight. However, it was concluded that the slight reduction in penalty weight resulting from a penalty value of 2.0, gave good algorithm performance over a range of test problems.

The conversion to fitness $F(X)$ is by the simple subtraction of the penalized objective function values $\ddot{f}(X)$, from the maximum penalized value in the current population.

The scaling factor γ, simply ensures that the penalized value of "worst" infeasible solution is equivalent to the highest objective function value in the current population. The second case in Equation (5) ($\gamma = 0.0$), applies when the "worst" infeasible individual has an objective function value equal to the highest in the population. Here, no penalty is applied since the infeasible solutions would naturally have a low fitness and should not be penalized further. The use of absolute values of the objective function in Equation (4) is necessary to allow the minimization of objective functions having negative values.

It is evident that the approach is dynamic in the allocation of the penalty in that the absolute value of the penalty depends on the objective values of the "best", the "worst" and the "$\breve{X}$" individuals. The penalty also accounts for the range of infeasibility in the current population and the distribution of the infeasible solutions in relation to the "best" individual in the population.

Fig. 1 illustrates the general procedure for self-adaptive fitness formulation method.

3 Test Cases

3.1 Eleven test cases

The performance of the proposed constraint handling method has been evaluated using a set of eleven test cases [3][5]. These test cases include various forms of objective function (linear, quadratic, cubic, polynomial, nonlinear), and each test case has a different number of variables. The test problems also pose a range of constraint types and number of constraints (linear inequalities, LI; nonlinear equalities, NE; and nonlinear inequalities, NI).

The self-adaptive fitness formulation described here has been implemented and evaluated using simple genetic algorithm with Gray encoding of the variables (25 bits used to represent each variable). The implementation uses proportional (roulette wheel) selection strategy, single point crossover, random bit mutation, and finally an elitist replacement strategy.

The performance of any evolutionary algorithm for constrained optimization is determined by the constraint handling technique used as well as the evolutionary search algorithm (including parameters) [5]. The performance of the algorithm described here has been compared to the results reported for the homomorphous mapping method [3], and therefore, where possible, the same genetic algorithm (GA) parameter values have been adopted (a population size of 70; 90% probability of crossover; and a probability of mutation between 0.3%-0.5%). The small tolerance of $\delta = 0.0001$ is applied to the equality constraints. For each test case, 20 runs, each starting from a different randomly generated population were performed; and the maximum number of generations was set to 20,000. A comparison of the algorithm performance is also made to the results obtained by Runarsson and Yao (2000) and Ben Hamida and Schoenauer (2000) for the same test problems although it is understood that both methods used an evolution strategy algorithm and performed different experiments to those given in this paper.

3.1.1 Performance of the algorithm

Farmani and Wright (2003) presented results for different experiments, concluding that the algorithm described here found good, if not optimum solutions for all eleven test cases. This algorithm is a development of that described in [6]. The results for the best solutions for the current and the earlier version of the fitness formulation algorithm are given in Table 1. A comparison of the results for the two versions of the algorithm indicate that adding a further adaptive element to the fitness formulation has resulted in an improvement in results for all test functions. It should also be noted that the standard deviations for the test results are very small [7]. This is an important characteristic for the application of the algorithm to the solution of real world problems

where cost and time constraints prohibit repeated runs of the algorithm. The small standard deviations for the algorithm described here indicate that it is robust in finding a near optimum solution without the need for repeated runs of the optimization.

3.2 Optimum design of a water distribution network

The performance of the method also has been examined by its application to the analysis of the expansion of New York Tunnel (NYT) water supply system Fig. 2. The New York pipe network has been studied using evolutionary techniques by a large number of researchers in the past ([15][16][17][18]).

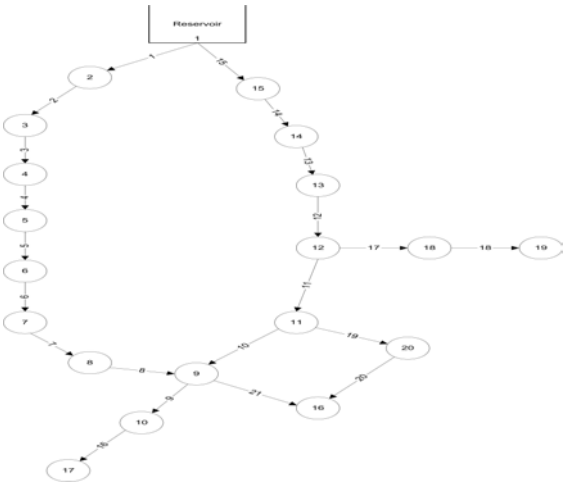

Fig. 2 Existing New York City Water Supply Tunnels

The objective of the NYT problem was to determine the most economically effective design for addition to the existing system of tunnels that constituted the primary water distribution system of the city of New York. Because of age and increased demands, the existing gravity flow tunnels were found to be inadequate to meet the pressure requirements (at nodes 16,17,18,19 and 20) for the projected consumption level (Savic and Walters 1997). The construction of additional gravity flow tunnels parallel to the existing was considered. The node and link data are from (Murphy et al. 1993).

Tunnel (pipe) diameters are considered as design variables. There are 15 available discrete diameters [36, 48, 60, 72, 84, 96, 108, 120, 132, 144, 156, 168, 180, 192, 204 inches] and one extra possible decision which is the "do nothing" option. All twenty one tunnels are considered for duplication. Supplying demand at an adequate pressure to consumers is the main constraint in the design of water distribution systems. The performance of each candidate design solution is evaluated through simulation of the network flows. EPANET2 computer program (Rossman 2000) is the network solver used in this work.

The cost function is non-linear, $C = 1.1 D_{ij}^{1.24} L_{ij}$, in which cost C is in dollars, diameter D_{ij} is in inches, and length L_{ij} is in feet. The optimization problem was set based on capital expenditure as an objective function and the minimum pressures as constraints.

The self-adaptive fitness formulation has been implemented using exactly the same genetic algorithm setting as 11 test cases. The parameter values adopted were a population size of 20; 80% probability of crossover; 3% probability of mutation; and the maximum number of generations was set to 2000. Again 20 runs, each starting from a different randomly generated population, were performed. A comparison of the algorithm performance is made to the results obtained by other researchers ([15][16][17][18]).

Table 2 summarizes the results for the NYT reported by different researchers and those obtained by the self-adaptive fitness formulation. Table 2 indicates that the algorithm described here found good, if not optimum, solutions for the New York Tunnel network. The solution listed in table for Lippai et al. (1999) is based on linkage of the simulation model (WinPipes) to Evolver (1996) which is a commercial genetic algorithm based optimizer. Another solution reported by Lippai et al. (1999), which is linkage of WinPipes to GENOCOP [10] was not as successful as the above linkage: the best solution of 45.73 million dollars was found after 80,000 evaluation trial.

Savic and Walters (1997) reported two sets of solutions based on different hydraulic coefficients. The genetic algorithm parameters used were population size of 100, the probability of crossover of 1, the probability of mutation equal to 1/84 and the number of generations allowed was 10000. The result of 37.13 million dollars for their work is perhaps the best to date, which has been found based on several different runs.

Self-adaptive fitness formulation found the optimum solution with less computational effort. Both population size and number of generations allowed one fifth of those of Savic and Walters (1997) (population size = 20, the number of generations allowed = 2000). This is an important characteristic for the application of the algorithm to the solution of real world problems where cost and

time constraints prohibit repeated runs of the algorithm and evaluations of the network.

The results obtained by the self-adaptive fitness formulation presented in this paper are compared in detail to those for previously published algorithms. In general, the self-adaptive fitness formulation performs as well as, and in some case better than the alternative algorithms. Again it is seen that the algorithm presented in this paper was able to find the optimum or near optimum solution with considerably less computational effort.

The improved performance of the algorithm described here may be due to the deterministic handling of constraint violations and suggests that it has better performance in solving highly constrained problems. The main advantage of the method is that it does not require any parameter tuning. It simply uses the characteristic of the search space in each generation. It does not require an initial feasible solution and can start with a completely infeasible population. The ability to find a feasible solution as well as the optimum solution represents a significant improvement in algorithm performance.

4 Conclusion

This paper introduces a self-adaptive fitness formulation for evolutionary constraint optimization. The infeasibility values are represented by the sum of the normalized constraint violation values. The infeasibility measure has the properties that it increases in value with both the number of active constraints and the magnitude of each constraint violation. The infeasibility measure is used to form a two-stage dynamic penalty which is applied to the infeasible solutions. The penalty is applied such that slightly infeasible solutions having a low objective function value are allowed to remain fit. It is shown that this approach gives improved or comparable results to those of existing methods. The main advantages of the approach are that first, it does not require any parameter tuning; second, it is able to find the optimum or near optimum solution starting with a completely infeasible population of solutions; third having low population size and generation number is important in real world problems where cost and time constraints prohibit repeated runs of the algorithm and simulation and finally, self adaptive fitness formulation method can be used as a fitness evaluator with any evolutionary algorithm. The method handles constraint violations in a deterministic way, and suggests that it has better performance in solving highly constrained problems. The performance of the algorithm has been illustrated by application to 11 test cases and a benchmark network. It has been shown that the self-adaptive fitness formulation is able to find optimum or near optimum solutions much more efficiently and with much less computational effort.

5 Acknowledgement

This work was supported by the UK Engineering and Physical Science Research Council. (Platform grant GR/R14712/01 and grant GR/M23601/01).

References:

[1] D. Powell and M.M. Skolnick, Using Genetic Algorithms in Engineering Design Optimization with Non-linear Constraints. *Proceedings of the Fifth International Conference on Genetic Algorithms*, Ed. by Forrest S., Morgan Kaufmann Publishers, Vol. 5, 1993, pp. 424-431.

[2] D.W. Coit, A.E. Smith and D.M. Tate, Adaptive Penalty Methods for Genetic Optimization of Constrained Combinatorial Problems. *INFORMS Journal on Computing*, Vol.8, 1996, pp. 173-182.

[3] S. Koziel and Z. Michalewicz, Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization. *Evolutionary computation*, Vol.7, No.1, 1999, pp. 19-44.

[4] S. Ben Hamida and M. Schoenauer, An Adaptive Algorithm for Constrained Optimization Problems. *Proceedings of Parallel Problem Solving from Nature*, Vol.VI, 2000, pp. 529-538.

[5] T.P. Runarsson and X. Yao, Stochastic Ranking for Constrained Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, Vol.4, No.3, 2000, pp. 284-294.

[6] J.A. Wright and R. Farmani, Genetic Algorithm: A Fitness Formulation for Constrained Minimization. *Proceedings of the Genetic and Evolutionary Computation Conference*, San Francisco, California, 2001, pp. 725-732.

[7] R. Farmani and J.A. Wright, Self-Adaptive Fitness Formulation for Constrained Optimization. *IEEE Transactions on Evolutionary Computation*, Vol.7, No.5, 2003, pp. 445-455.

[8] A. Homaifar, C.X. Qi and S.H. Lai, Constrained Optimization via Genetic Algorithms. *Simulation*, Vol.62, No.4, 1994, pp. 242-254.

[9] J.A. Joines and C.R. Houck, On the use of Non-Stationary Penalty Functions to Solve Non-Llinear Constrained Optimization Problems with GA's. *IEEE Conference on Evolutionary Computation*, Vol. 2, 1994, pp. 579-584.

[10] Z. Michalewicz and C.Z. Janikow, Handling Constraints in Genetic Algorithms. *Proceedings of the International Conference on Genetic Algorithms*, Vol. 4, 1991, pp. 151-157.

[11] Z. Michalewicz, G. Nazhiyath and M. Michalewicz, A Note on Usefulness of Geometrical Crossover for Numerical Optimization Problems. *Proceedings of the Fifth Annual Conference on Evolutionary Programming*, San Diego, CA, MIT Press, Cambridge, MA, 1996, pp. 305-312.

[12] H. Myung and J.H. Kim, Constrained Optimization using Two-Phase Evolutionary Programming. *Proceedings of IEEE International Conference on Evolutionary Computation*, 1996, pp. 262-267.

[13] J.T. Richardson, M.R. Palmer, G. Liepins and M. Hilliard, Some Guidelines for Genetic Algorithms with Penalty Functions. *Proceedings of the International Conference on Genetic Algorithms*, Vol. 3, 1989, pp. 191-197.

[14] A.E. Smith and D.W. Coit, Constraint Handling Techniques-Penalty Functions. *Handbook of Evolutionary Computation*, T. Baeck, D. Fogel and Z. Michalewicz (Editors), Oxford University Press, 1997.

[15] L.J. Murphy, A.R. Simpson and G.C. Dandy, Pipe Network Optimization using an Improved Genetic Algorithm. *Research Report* No. R109, Dept of Civil and Environment Engineering, University of Adelaide, Australia, 1993.

[16] D.A. Savic and G.A. Walters, Genetic Algorithms for Least Cost Design of Water Distribution Networks. *Journal of Water Resources Planning and Management*, ASCE, Vol. 123, No. 2, 1997, 67-77.

[17] I. Lippai, J.P. Heaney and M. Laguna, Robust Water System Design with Commercial Intelligent Search Optimizers. *Journal of Computing in Civil Engineering*, 13(3), (1999). 135-143.

[18] Z.Y. Wu, P.F. Boulos, C.h. Orr and J.J. Ro, Using Genetic Algorithm to Rehabilitate Distribution Systems. *Journal AWWA*, Vol. 93, No. 11, 2001, pp. 74-85.

[19] L.A. Rossman EPANET, Users manual. U.S. Envir. Protection Agency, Cincinnati, Ohio, 2000.

[20] Evolver, The Genetic Algorithm Solver: Version 3.5, Axcelis Crop., Seattle, Wash., 1996.

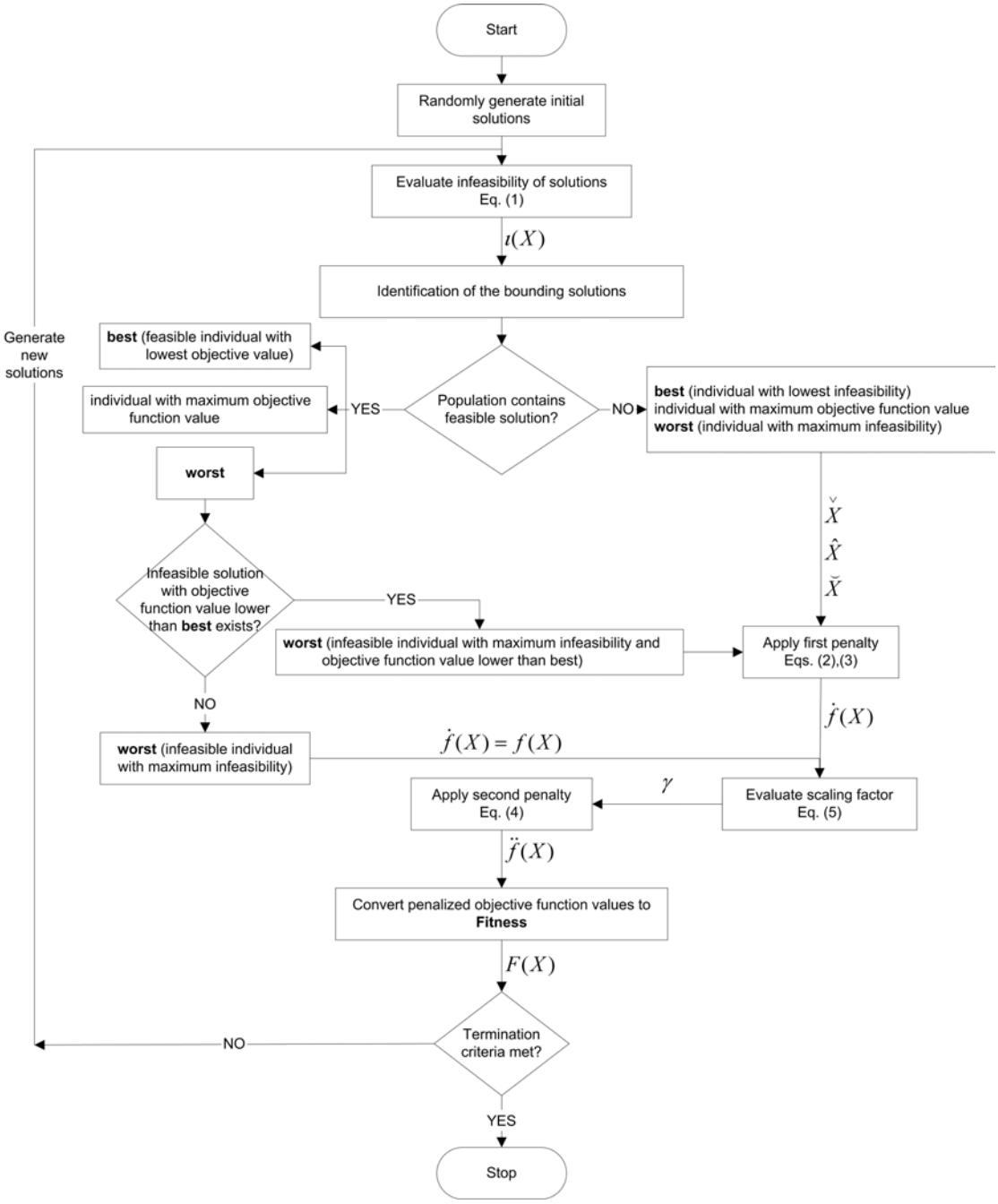

Fig. 1: The Optimization procedure

Table 1: Results for Best Values

Function	Optimum value	Koziel and Michalewicz 1999	Ben Hamida and Schoenauer 2000	Runarsson and Yao 2000	Wright and Farmani 2001	Self-Adaptive Fitness formulation
G1	-15	-14.7864	-15.0000	-15.0000	-14.9996	-15.0000
G2	0.803553	0.799530	0.800781	0.803515	0.796640	0.802970
G3	1.0	0.9997	1.0000	1.0000	0.9994	1.0000
G4	-30665.5	-30664.900	-30665.500	-30665.539	-30661.100	-30665.500
G5	5126.4981	-	4707.5200	5126.4970	5126.6398	5126.9890
G6	-6961.8	-6952.100	-6961.810	-6961.814	-6961.370	-6961.800
G7	24.306	24.620	24.360	24.307	24.671	24.480
G8	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825
G9	680.63	680.91	680.63	680.63	681.20	680.64
G10	7049.33	7147.90	7095.15	7054.32	7152.83	7061.34
G11	0.75	0.75	0.75	0.75	0.75	0.75

Table 2: Results for New York Tunnel

Link	Murphy et al. 1993	Savic and Walters 1997		Lippai et al. 1999	Wu et al. 2001	Self-Adaptive Fitness formulation
		$\omega = 10.5088$	$\omega = 10.9031$			
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
6	0	0	0	0	0	0
7	0	108	0	132	108	108
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
12	0	0	0	0	0	0
13	0	0	0	0	0	0
14	0	0	0	0	0	0
15	120	0	144	0	0	0
16	84	96	84	96	96	96
17	96	96	96	96	96	96
18	84	84	84	84	84	84
19	72	72	72	72	72	72
20	0	0	0	0	0	0
21	72	72	72	72	72	72
Cost ($ million)	38.8	37.13	40.42	38.13	37.13	37.13
Number of Evaluation	96,750	1,000,000(Total)		46,016	37,186	26,340