

Old Dominion University

ODU Digital Commons

---

Civil & Environmental Engineering Theses &  
Dissertations

Civil & Environmental Engineering

---

Summer 2004

## Genetic Algorithm-Based Model for Determination of Efficient Management Strategies for Irrigation Canal Networks

Talaat Taher El Gamel  
*Old Dominion University*

Follow this and additional works at: [https://digitalcommons.odu.edu/cee\\_etds](https://digitalcommons.odu.edu/cee_etds)



Part of the [Bioresource and Agricultural Engineering Commons](#), and the [Civil Engineering Commons](#)

---

### Recommended Citation

El Gamel, Talaat T.. "Genetic Algorithm-Based Model for Determination of Efficient Management Strategies for Irrigation Canal Networks" (2004). Doctor of Philosophy (PhD), Dissertation, Civil & Environmental Engineering, Old Dominion University, DOI: 10.25777/q10x-3858  
[https://digitalcommons.odu.edu/cee\\_etds/35](https://digitalcommons.odu.edu/cee_etds/35)

This Dissertation is brought to you for free and open access by the Civil & Environmental Engineering at ODU Digital Commons. It has been accepted for inclusion in Civil & Environmental Engineering Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact [digitalcommons@odu.edu](mailto:digitalcommons@odu.edu).

**GENETIC ALGORITHM-BASED MODEL FOR DETERMINATION  
OF EFFICIENT MANAGEMENT STRATEGIES FOR  
IRRIGATION CANAL NETWORKS**

by

Talaat Taher El Gamel  
M.S. 2001, Old Dominion University, USA  
B.S. 1987, Alexandria University, Egypt

A dissertation submitted to the Faculty of  
Old Dominion University in Partial Fulfillment of the  
Requirements for the Degree of


DOCTOR OF PHILOSOPHY

CIVIL ENGINEERING

OLD DOMINION UNIVERSITY  
August 2004

Approved by:

 Laura Harrell (Director)

 A. Osman Akan (Member)

---

Jaewan Yoon (Member)

---

Resit Unal (Member)

## **ABSTRACT**

### **GENETIC ALGORITHM-BASED MODEL FOR DETERMINATION OF EFFICIENT MANAGEMENT STRATEGIES FOR IRRIGATION CANAL NETWORKS**

Talaat Taher El Gamel  
Old Dominion University, 2004  
Director: Dr. Laura Harrell

An optimization model for the determination of efficient management strategies for an irrigation canal network is developed. The objective is to minimize the total water consumed while satisfying various system constraints. An unsteady flow model is used to simulate the flow in the network. A genetic algorithm- (GA-) based framework is used to solve the model. The suitable GA parameters that should be used within the model, as well as the performance of various constraint-handling techniques, are studied. Uncertainties in crop pattern and water consumption rates are incorporated into the search procedure to identify more reliable solutions. A graphical interface is also developed to make the model more user-friendly.

## ACKNOWLEDGMENTS

My sincere thanks and appreciation go to the following people:

- My family; my father, my mother and my wife for their patient during my study abroad in USA.
- My graduate committee, Dr. Laura Harrell, Dr. A. Osman Akan, Dr. Jaewan Yoon, and Dr. Resit Unal for their participation and advice.
- Eng. Soliman Abu Zeid, and the people in the telemetry project in the Ministry of Water Resources and Irrigation in Egypt for giving me the chance to begin my graduate study in USA.



4.1 Introduction.....	71
4.2 Population size.....	75
4.3 Crossover probability.....	76
4.4 Mutation rate.....	76
4.5 Blend crossover extension.....	77
4.6 Analysis.....	77
4.7 Results.....	80
4.7.1 Test 1 (Mean Values).....	80
4.7.2 Test 2 (Parameter interaction charts).....	89
4.7.3 Test 3 (Comparing runs that have the same parameters).....	93
4.7.4 Test 4 (Parameters of the best solution).....	95
4.8 Summary.....	96
4.9 Validate the results.....	98
4.10 Conclusion and future work.....	102
4.11 References.....	103
5. Constraint-handling techniques.....	105
5.1 Introduction.....	105
5.2 Techniques investigated for the current model.....	106
5.2.1 Penalty functions.....	106
5.2.1.1 Additive static penalty.....	109
5.2.1.2 Multiplicative static penalty.....	110
5.2.1.3 Additive linear dynamic penalty.....	110
5.2.2 Multiobjective optimization.....	111
5.2.2.1 Multiobjective method used in the current study.....	112
5.2.3 Self-adaptive penalty function.....	113
5.2.4 Stochastic techniques.....	114
5.2.4.1 Stochastic tournament selection.....	115
5.2.5 Adaptive penalty function.....	117
5.3 Comparisons.....	118
5.4 Results.....	120
5.4.1 Test 1.....	120
5.4.2 Test 2.....	122
5.4.3 Test 3.....	124
5.4.4 Test 4.....	125
5.4.5 Summary.....	129
5.5 The performance of STSF technique.....	129
5.6 The performance of adaptive technique.....	134
5.7 Conclusions and future work.....	135
5.8 References.....	136
6. Generating more reliable management strategies under conditions of uncertainty.....	139
6.1 Introduction.....	139
6.2 Crop data and uncertainty.....	140
6.3 Chance-constrained technique.....	142

6.4	Generating sampling (MCS vs. LHS).....	143
6.5	Chance-Constrained Genetic Algorithm (CCGA) model.....	145
6.6	Analysis.....	148
6.7	Results.....	148
6.7.1	Effect of the number of realizations used to evaluate each solution.....	154
6.7.2	Effect of changing the reliability target.....	154
6.8	Conclusion and future works.....	159
6.9	References.....	160
7.	Graphical user interface for the model.....	162
7.1	Introduction.....	162
7.2	File Commands.....	164
7.2.1	Open project.....	164
7.2.2	New project.....	165
7.2.3	Save project As.....	166
7.3	Data commands.....	166
7.3.1	Hydraulics data.....	168
7.3.1.1	Canals Data.....	168
7.3.1.2	Regulators Dialog.....	170
7.3.1.3	Reaches Dialog.....	170
7.3.1.4	Regulators initial data Dialog.....	171
7.3.1.5	Operation Time and Boundary Time Dialogs.....	172
7.3.1.6	Operations Data Dialog.....	173
7.3.2	Settings dialog.....	173
7.4	Reports Commands.....	174
7.5	Run Menu.....	176
7.5.1	Check command.....	177
7.5.2	Run command.....	177
7.6	Results.....	179
7.6.1	Water level data.....	179
7.6.2	Regulators Data.....	181
7.6.3	Fitness and cost data.....	181
7.6.5	Constraints Violations Data.....	182
7.6.6	Feasible Solutions.....	183
7.6.7	STS average probabilities.....	184
7.6.8	Final Report.....	184
8.	Conclusions and recommendations.....	186
VITA	.....	189

## LIST OF TABLES

Table	Page
4.1-a	Summary of MMC output for crossover probability..... 82
4.1-b	Summary of MMC output for mutation rate..... 82
4.1-c	Summary of MMC output for blend crossover extension..... 82
4.1-d	Summary of MMC output for population size..... 83
4.2-a	Number of wins for each crossover probability value..... 93
4.2-b	Number of wins for each mutation rate value..... 93
4.2-c	Number of wins for each blend crossover extension value..... 93
4.2-d	Number of wins for each population size value..... 94
4.3-a	Parameters of the best value for the first scenario of the case study... 95
4.3-b	Parameters of the best value for the second scenario of the case study 95
4.3-c	Parameters of the best value for the third scenario of the case study.. 95
4.3-d	Population sizes of the best value for different scenarios of the case study..... 95
4.4	GA parameters for different alternatives..... 99
5.1	GA parameters associated with each scenario..... 119
5.2	The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 1..... 120
5.3	The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 2..... 120
5.4	Techniques that produce the five best values and the worst value regarding the first measure..... 123
5.5	Techniques that produce the five best values and the worst value regarding the second measure..... 123
5.6	Number of times each technique wins from runs that have same seed value regarding the first measure..... 124
5.7	Number of times each technique wins from runs that have same seed value regarding the second measure..... 125
6.1	Reliability level obtained from 1000 Monte Carlo samples..... 147
6.2	Target reliability level..... 147



## LIST OF FIGURES

Figure		Page
2.1	Number of the strings that satisfy each constraint, and that satisfy all constraints without and with constraints tolerance respectively.....	11
2.2	Simple example of an irrigation canal network.....	13
2.3	Flow chart of genetic algorithm (Deb, 2001).....	16
2.4	Examples of binary-coded crossover.....	21
2.5	Examples of real-coded crossover.....	23
2.6	El Monofiya irrigation canal network, Egypt. (Main branches).....	26
2.7	First scenario of the case study.....	29
2.8	Upstream and downstream water level of El Quarinien regulator during the whole run on the first scenario of case study.....	30
2.9	Second scenario of the case study.....	33
2.10	Upstream and downstream water level of El Quarinien regulator during the whole run on the second scenario of case study.....	34
2.11	Water level at the reach just before second regulator of canal 3 for the second scenario of case study.....	35
2.12	Water level at the reach just before the intake of canal 45 for the second scenario of case study.....	35
2.13	Third scenario of the case study.....	39
2.14	Upstream and downstream water level of El Quarinien regulator during the whole run on the third scenario of case study.....	40
2.15	Water level at the reach just before second regulator of canal 3 for the third scenario of case study.....	41
2.16	Water level at reach 2 of canal 46 for the third scenario of case study	41
3.1.	Flow through constrictions.....	49
3.2	Sluice gate equation.....	50
3.3	Submerged jump.....	51
3.4	Preissmann implicit scheme.....	53
3.5	Amein implicit scheme.....	54
3.6	Implicit rectangular net.....	57
3.7	A junction between two canals.....	58
3.8	Total energy before the gate and after the submerged jump.....	59
3.9	System of equations.....	61
3.10	Jacobian Matrix.....	61
3.11	Current end of the canal is not an end of a reach.....	65
3.12	Current end of the canal is an end of a reach followed by constriction	65

3.13	Current end of the canal is the end of a reach followed by branch.....	66
3.14	When the water level in the main canal increases to enter the branch..	66
4.1	Example of SAS output: crossover probability for measure 2 of the third scenario of the case study.....	81
4.2	Difference between means of different crossover probabilities for both measures.....	85
4.3	Difference between means of different mutation rates for both measures.....	86
4.4	Difference between means of different blend crossover extensions for both measures.....	87
4.5	Difference between means of different populations sizes for both measures.....	88
4.6	Parameter interaction charts for the first scenario of the case study...	89
4.7	Parameter interaction charts s for the second scenario of the case study.....	90
4.8	Parameter interaction charts of the third scenario of the case study...	91
4.9	Parameter interaction charts for different scenarios for population size.....	92
4.10	Different alternatives of recommended parameters for the first scenario.....	100
4.11	Different alternatives of recommended parameters for the second scenario.....	101
4.12	Different alternatives of recommended parameters for the third scenario.....	102
5.1	The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 1.....	121
5.2	The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 2.....	121
5.3	Mean and standard deviation of the first measure of different scenarios.....	127
5.4	Mean and standard deviation of the second measure of different scenarios.....	128
5.5	Average STS probabilities for a specific initial seed value in the first and third scenarios.....	130
5.6	Average constraint violations and the ratio of feasible solutions for scenarios 1 and 3.....	131
5.7	Sorted values of the objective function value and the constraints violation as a ratio of the maximum value for first and third scenarios in the ninth generation.....	131
5.8	Minimum and average feasible solutions for first and third scenarios.	132
5.9	Average STS probabilities for the worst and best runs of the second scenario.....	133
5.10	Average STS probabilities for the worst and best runs of the third scenario.....	133

5.11	The value of BFc/BFo for the three scenarios for adaptive technique..	134
6.1	Upper and lower bounds for water consumption rates for different crops used in the case study.....	141
6.2	Upper and lower bounds for crop percentages for crops used in the case study.....	142
6.3	Monte Carlo Sampling.....	143
6.4	LHS Sampling for uniform and normal distribution.....	144
6.5	An example of LHS Sampling.....	145
6.6	Best feasible solution obtained in each run for each number of realizations.....	149
6.7	Likelihood of satisfying the water shortage constraint in CCGA model for different runs and the average for each number of realizations.....	150
6.8	Likelihood of satisfying the water shortage constraint in CCGA model for different runs and the average for each number of realizations using 200 realizations.....	150
6.9	Likelihood of satisfying the required water levels constraint in CCGA model for different runs and the average for each number of realizations using 200 realizations.....	151
6.10	Average target satisfactions values for different constraints per generation for the random seed run that resulted in the best solution using 10 realizations.....	152
6.11	Average target satisfactions values for different constraints per generation for the random seed run that resulted in worst solution using 10 realizations.....	153
6.12	Number of feasible solutions per generation for the random seed runs that resulted in the best and worst solutions using 10 realizations.....	153
6.13	The effect of increasing target reliability of water shortage to 95%....	155
6.14	The effect of decreasing target reliability of required water levels to 90%.....	155
6.15	Effect of increasing target reliability of water shortage to 95% for the random seed run that resulted in the best solution.....	156
6.16	Effect of increasing target reliability of water shortage to 95% for the random seed run that resulted in the worst solution.....	157
6.17	Effect of decreasing target reliability of required water levels to 95% for the random seed run that resulted in the best solution.....	158
6.18	Effect of decreasing target reliability of required water levels to 95% for the random seed run that resulted in the worst solution.....	158
7.1	Main page of the interface.....	163
7.2	Open project dialog.....	164
7.3	Save project As dialog.....	166
7.4	Canals data dialog.....	169
7.5	Example of how the canals must be numbered.....	169
7.6	Regulators data dialog.....	170

7.7	Reaches data dialog.....	171
7.8	Regulators initial data dialog.....	172
7.9	Operation time dialog.....	172
7.10	Operations data dialog.....	173
7.11	Settings dialog.....	174
7.12	Canals data report.....	175
7.13	Genetic algorithm data report.....	176
7.14	The interface during the run as an optimization model.....	178
7.15	Water level data for a given point over time.....	180
7.16	Water levels for a whole channel and for a specific point.....	180
7.17	Data of a specific regulator at different time steps.....	181
7.18	Average fitness values per generation.....	182
7.19	Average flood violation ratio per generation.....	182
7.20	Number of feasible solutions per generation.....	183
7.21	Average probabilities for STS method.....	184
7.22	Final reports of an optimization model and an unsteady flow simulation model.....	185

## CHAPTER 1

### INTRODUCTION

Enhancing irrigation systems in order to maximize the net benefit or minimize the irrigation cost is an important issue, especially in arid and semi-arid countries where the water is scarce, and the irrigation is the main water consumer. The fact that the water demand increases rapidly as the result of the population increase makes this issue more important. According to Schultz, and DeWrachien (2002), “Based on the forecasts for population growth and the improvement in the standard of living, it is expected that food production will have to be doubled in the next 25 years. In addition it is expected that 90% of the increase in food production will have to come from existing cultivated land and only 10% from new land reclamation, either in the highlands or in the lowlands. There is no way that the cultivated area without a water management system can contribute significantly to the required increase in food production.” According to the authors, good management and efficient operation are basic requisites for improving agricultural water management. This means that efficient management and operation of irrigation networks is a critical issues. In Egypt, enhancing irrigation efficiency is especially important, as the population is increasing rapidly while the water supply remains constant. There is much room for improvement in Egypt, considering that “the structures, management and technical properties of the Egyptian irrigation system have been designed and operated within the situation of water abundance, which means that up to the late 1980s very little emphasis was placed on improving the efficiency of the water use.” (Hvidt, 1998). This makes Egyptian agriculture is one of the most consumptive irrigation in the world and the reason for this, according to (Samaha, 1979), is related to the wasteful use of irrigation water. Given that the likelihood of increasing the water supply through establishing new projects in the south countries is small, “The most promising way of tackling the water problem [in Egypt] is, therefore, to expend

---

The journal model for this thesis is *ASCE, Journal of Hydraulics Division*.

resources through water conservation in the old lands by introducing more effective on-farm irrigation technologies and practices” (Water Bank, 1993).

Based on Gates and Alshaikh (1993), parameters representing physical properties and boundary conditions of irrigation systems can be classified into three categories:

- ❑ Hydrologic properties: streamflow, crops evapotranspiration, precipitation, and infiltration, etc.
- ❑ Hydraulic parameters: cross-section geometry, resistance coefficients, etc.
- ❑ Management parameters: irrigation application efficiency and water delivery schedule.

Most of the studies that have been done to improve irrigation canal networks were done by means related to a combination of first and third categories, which are crops pattern and operations schedule (reservoir routing). These studies were done either to design a new irrigation network, or rehabilitation of an existing irrigation network. These studies were done using linear programming, dynamic programming, non-linear programming, simulation models, or real time operations (Yeh, 1985). Some of these studies are summarized below.

Anderson (1968) developed a simulation model to define the optimal crop pattern to be grown on irrigated farms. Crop pattern is calculated based on different input data such as, the anticipated water seasonal supply of an organization based on its water rights and reservoir supply, number and sizes of farms, minimum and maximum acreage of each crop, costs and gross return for each crop, water requirement for each crop, and yield loss from not watering in specific periods.

Matanga and Miguel (1979a, 1979b) used a linear optimization model to decide the best allocation of three crops based on total water supply and maximum amount of water that can be delivered for irrigation. The model considered some constraints. Such as the total crop area cannot exceed total area, and total the irrigation depth cannot exceed the capacity of the water distribution system. Then they used stochastic dynamic programming to define the optimal amount of water to be used for leaching prior to the irrigation season and seasonal irrigation depths to maximize the gross margin until the end of a finite planning horizon, or to maximize gain in gross margin per stage in an infinite planning horizon.

Afshar et al. (1991) used a mixed integer linear optimization model for a river basin development for irrigated agriculture in the planning and design phases. The model has 4 components: surface reservoir, conveyance and distributed canals, limited hectares of land to be developed, and limited number of crops to be considered. The model is a monthly chance-constrained optimization model with a one-year horizon.

Malek-Mohammadi (1998) made an improvement to Afshar et al. (1991) by adding the effect of groundwater and spring withdrawal, the delivery system capacity, and the effect of the cost due to the drainage, land leveling and irrigation network construction. He used a chance-constraint optimization technique, and he implemented his model for an irrigation canal network with three plains and nine cells.

Rovikumar and Venugopal (1998) developed a three-phase optimization model for the optimal operation for a large-scale south Indian irrigation system. The first phase is a simulation model that uses the historical rainfall data to estimate the irrigation demand sequence. The second phase is a stochastic dynamic programming model that treats both irrigation demand on the reservoir and inflow into the reservoir stochastically. The third part is the simulation model that models the reservoir using the optimal release policy from the second phase.

In Egypt, Fawzy (1999) developed a linear optimization model to define the optimal crop pattern in Egypt. He used three different alternatives for the objective function. The first alternative is to maximize the net benefit of land and water per feddan. The second alternative is to maximize the net return of irrigation water volume. The third alternative is to rationalize the use of the available water resources by minimizing the irrigation needs. Ali (2000) studied the optimal crop patterns through a multiobjective linear optimization model that aims to minimize the irrigation water consumption, maximum the return from the water unit, and maximize the farmers' profits. He divided the Egyptian cultivated land into three main regions: the upper, middle, and lower region, each of which has its climate and though its water consumption rate.

Aside from mathematical models, simulation models provide an effective tool to improve the irrigation networks. According to Yeh (1985), "From practitioner's point of view, mathematical programming techniques have, thus far, not proven to be widely useful because of the complexity of water resources and non-commensurable objective in water resources management. In this regard, simulation is an effective tool for studying the operations of the complex water resource system incorporating the experience and judgment of the planner or design engineer into the model." However, direct incorporation of complex simulation models into an optimization model is computationally prohibitive (Neelakantan and Pundarikanthan, 2000). The conventional way to incorporate a simulation model into an optimization model is that the optimization model passes decision variables to the simulation model, receives the output of the simulation model, and then decides the next step based on evaluation of the objective value. In that case, the direct search methods, such as Hooke and Jeevs method (Gates and Alshaikh, 1993, Neelakantan and Pundarikanthan, 2000) is used to solve the problem. Evolutionary computation provides another effective way to incorporate simulation models in an optimization model.

Another means of enhancing irrigation is by controlling the canals operations. The automatic gate operation technique is used to increase the crop productivity and prevent damage due to flooding. Among these studies, Reddy et al. (1992) presented a technique for operation of irrigation canals in the presence of arbitrary external disturbances. They solved a linearized form of the continuity and gate-discharge equations. They assumed the lateral canals to be located immediately upstream of the last node in each pool. They verify their model using a nonlinear open-channel flow simulation model. The simulation model estimates the flow rates and water depths at each point in the reach, then these data will be used by the observer and the controller to calculate the change in the gate opening. After this, the flow through this regulator will be calculated and used as a boundary condition in the next time step.

The current study treats the problem of enhancing the irrigation networks differently. The goal of the current study is to define the optimal irrigation schedule for a short-term irrigation period (eg. For a typical irrigation period of five days in



Egypt), which can minimize the total water consumed while satisfying the system constraints, which are:

- ❑ No water shortage at any point in the network at any time during the irrigation period.
- ❑ No flood at any point in the network at any time during the irrigation period.
- ❑ The difference between the upstream water level and the downstream water level of any regulator is less than the maximum allowable difference at any time during the irrigation period.
- ❑ Water volume in the network at the end of the irrigation period is enough to start the next irrigation period.

The importance of tackling the problem this way stems from the following two facts:

- ❑ For some irrigation networks, such as that in the case study described in Chapter 2, defining the optimal crop pattern is not a practical issue, as it is hard to implement it in reality. This is because the cultivated area in such networks is divided among thousands of owners, who have the freewill to decide the cultivated crops. In the current model, the crop pattern will be treated as input data, and it will be treated stochastically as there is uncertainty associated with it.
- ❑ Using mean seasonal inflow or monthly inflow can be used while drawing a general strategy, but it cannot guarantee prevention of flood or water shortage during daily operations, unless suitable operations are defined based on the actual consumption rate and the hydraulic characteristic of the network.

Thus, the current study aims to develop an optimization model to define the best set of gate operations, and the best boundary conditions to minimize the total water consumed and prevent damages caused by water shortage, flooding or instability of regulators. This optimization model will be solved using a genetic algorithm (GA) based-search based procedure, and incorporates an unsteady flow model to evaluate each potential solution. A user-friendly interface was developed to make it easier for the user to enter the data and present the results. The model is applied to a case study involving a large-scale irrigation canal network in Egypt.

The current study is organized as follows:

Chapter 2 describes the optimization model and gives a brief introduction to GAs, and how the GA is implemented on the current study. Also the details of the case study in Egypt are presented at the end of this chapter. Chapter 3 describes the unsteady flow model that was used within this model. The GA parameter values used within this model are tested and discussed in Chapter 4. Different ways to handle the constraints are discussed and compared in Chapter 5. Chapter 6 addresses the uncertainty that is associated with crops pattern and water consumption rates. Chapter 7 gives a brief description of the user-friendly interface that was built for this model. The conclusions and recommended future works are presented in Chapter 8.

## References

- Afshar A., Marino, M. A., and Abrishamchi, A., "Reservoir Planning for Irrigation District," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 117 No. 1, JAN./FEB. 1991, pp. 74-85
- Ali, H. M., "Determining optimal crop pattern in Egypt by the use of multicriteria analysis," *Water Science (The National Water Research Center magazine, Ministry of Water Resources and Irrigation, Egypt)*, 28th – 29th Issue, October 2000-April 2001, pp. 31-39
- Anderson, R. L. "A Simulation Program to Establish Optimal Crop Pattern on Irrigation Farms Based on Preseason Estimates of Water Supply," *American Journal of Agricultural economics*, Vol. 50 No. 4, 1968, pp. 1586-1590
- Fawzy, G. M., "The efficient use of irrigation water in the Egyptian agriculture," *International conference on integrated management of water resources in the 21<sup>st</sup> century*, Cairo, Egypt, November 21-25, 1999, pp. 467-477
- Gates, T. K., and Alshaikh, A. A., "Stochastic Design of Hydraulic Structures in Irrigation Canal Networks," *Journal of Irrigation and Drainage Engineering*, American Society of Civil Engineers, Vol. 119 No. 2, MAR./APR. 1993, pp. 346-363
- Hvidt, M., *Water, Technology and Development: Upgrading Egypt's Irrigation System*, Tauris Academic Studies, 1998

- Malek-Mohammadi E. "Irrigation Planning, Integrated Approach," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 124 No. 5, SEP./OCT. 1998, pp. 272-279
- Mataga, G. B., and Marino, M. A., "Irrigation Planning: 1. Crop Pattern," *Water Resources Research*, Vol. 15, No 3, MAR 1979, pp. 672-678
- Mataga, G. B., and Marino, M. A., "Irrigation Planning: 1. Water Allocation for Leaching and Irrigation Purpose," *Water Resources Research*, Vol. 15, No 3, MAR 1979, pp. 679-683
- Neelakantan, T. R., and Pundarikanthan, N. V. "Neural Network-Based Simulation-Optimization Model For Reservoir Operation," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 126 No. 2, MAR./APR. 2000, pp. 57-64
- Reddy J.M., Dia A., Oussou A., "Design of control algorithm for operation of irrigation canals", *Journal of Irrigation and Drainage Engineering*, Vol. 118, No 6, 1992, pp. 852-867
- Ravikumar V., and Venugopal, K., "Optimal Operation of South Indian Irrigation Systems," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 124 No. 5, SEP./OCT. 1998, pp. 264-271
- Samah, M. A., "The Egyptian Master Water Plan," *Water Supply and Management*, Vol. 3, 1979, pp. 251-266.
- Schultz, B. and DeWrachien, D., "Irrigation and drainage systems research and development in the 21st century." *Irrigation and Drainage* Vol. 51, No 4, 2002, pp. 311-327.
- Yeh, W. W. G. "Reservoir Management and Operations Model: A State-of-the-Art Review," *Water Resources Research*, Vol. 21, No 12, DEC 1985, pp. 1797-1818
- World Bank "Arab Republic of Egypt, An Agricultural Strategy for the 1990s," *A World Bank Country Study* (World Bank, Washington, D.C.), 1993

## CHAPTER 2

### OPTIMIZATION MODEL

#### 2.1 Introduction

The goal of the current model is to define an efficient irrigation canal operation schedule (initial gate opening, gate operation and boundary condition) that minimizes the irrigation water volume consumed in an irrigation canal network, while satisfying four constraints, which are:

- The water level must stay at or above minimum-required water levels. In most irrigation canals, these minimum-required water levels are zero meaning that the canals should not run dry.
- The water levels must not exceed maximum-allowable water levels, which are channels' banks levels.
- The difference between the water levels upstream and downstream any regulator must not exceed the maximum-allowable difference.
- For some canals in the network, the water levels must not go below some pre-defined levels at the end of the routing. This constraint ensures that the water volume at the end of the flow routing will be sufficient for the beginning of the next irrigation period.

An optimization model is developed using the above defined objective and constraints, and is solved using a Genetic Algorithm (GA), which has been shown to be a powerful tool for solving very complex models without any simplification. An unsteady flow model is used to evaluate each potential solution (string) in the GA. This chapter describes the optimization model, gives a brief introduction to GAs, and how a GA is implemented in the current study. Also, a case study in Egypt will be presented in the end of this chapter.

## 2.2 Optimization model

The optimization model used herein is as follows:

$$\text{Minimize } z = \sum_{t \in NT} Q_t - \sum_{t \in NT} \sum_{o \in NO} Q_{t,o} \dots \dots \dots (2.1)$$

Subject to:

$$y_p(t) > y_r \quad \forall p, \forall t \dots \dots \dots (2.2)$$

$$WL_p(t) \leq FL_p \quad \forall p, \forall t \dots \dots \dots (2.3)$$

$$USWL_g(t) - DSWL_g(t) \leq MD_g \quad \forall g, \forall t \dots \dots \dots (2.4)$$

$$WL_p(t_{end}) \geq RWL_p \quad \forall p \in RP \dots \dots \dots (2.5)$$

Where:

$NT$  : Number of time steps of the flow routing.

$Q_t$  : Discharge at the inflow point during time step  $t$ .

$NO$  : Total number of outflow points.

$Q_{t,o}$  : Discharge at the outflow point  $o$  during time step  $t$ .

$y_p$  : Water depth at point  $p$ .

$y_r$  : Minimum required water depth, and for irrigation, it was considered as zero to just prevent the water shortage.

$WL_p$  : Water level at point  $p$ .

$FL_p$  : Maximum allowable water level at point  $p$ .

$USWL_g$  : Upstream water level of regulator  $g$ .

$DSWL_g$  : Downstream water level of regulator  $g$ .

$MD_g$  : Maximum allowable difference between upstream water level and downstream water level for regulator  $g$ .

$RP$	Number of points that have required water levels at the end of simulation.
$RWL_p$	Required water level at point $p$ .
$t$ :	Routing time step.
$t_{end}$ :	Time at the end of the flow routing.

### 2.2.1 Decision variables

This model contains two types of decision variables; gate opening values and boundary conditions. Boundary conditions include the upstream boundary condition, which is the water level at the upstream end of the network, and the downstream boundary conditions, which represent the discharge at the downstream end of each regulator. Also gate-opening values include both initial gate opening (at the beginning of the routing) and operations during the routing.

### 2.2.2 Constraint violations tolerance

In a real irrigation network such as the one presented in the case study, there may be some weak points, such as a bank with a low elevation, or a branch with an entrance that has a higher bed level than that of the adjacent point in the main canal. These points could be actual weak points or could be a result of inaccuracy in data input. These weak points, even if very few, can make finding a feasible solution very difficult. Assuming a small tolerance for constraint violations can prevent these few points from controlling the whole network, and can lead to better solutions.

Figure 2.1 presents two examples of the same scenario of the case study, with and without allowing for a small constraint violation tolerance. Without considering tolerance (case the left graph of Figure 2.1), the number of feasible solutions during the whole run is zero, and there are no strings that satisfy the first constraint (water shortage). Only 14 strings in the first four generations satisfied the second constraint (Flood). The second graph in Figure 2.1 presents the same scenario while using the following constraint violations tolerance levels:

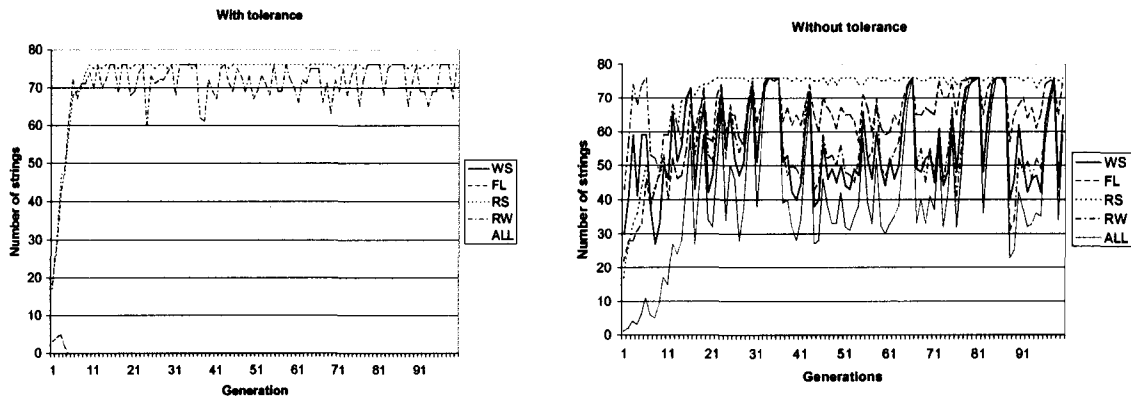


Figure 2.1

Number of the strings that satisfy each constraint, and that satisfy all constraints without and with constraint violation tolerance respectively

- ✍ Water shortage: 0.01
- ✍ Flood: 0.005
- ✍ Regulator stability: 0.0
- ✍ Required water level: 0.05

The difference between numbers of feasible solutions is very clear. In the final optimal solution, the total flooded length is 90 m (0.0005 of the total length), and total cultivated land affected by water shortage is 630 feddan (0.0009 of the total cultivated area). The method for calculating the violation for each constraint is discussed in section 2.4.2.

### 2.3 Solving the optimization model

Many optimization techniques have been used in hydraulics or water resources systems optimizations, including linear, dynamic and non-linear programming, direct search methods, evolutionary computation, and complete enumeration techniques.

Linear, and dynamic programming techniques cannot be used with the current study because of the complex nature of the problem. Also, complete enumeration would be impractical, as the decision variables are continuous, and the computational time required would be prohibitive.

Regarding nonlinear optimization, Yeh (1985) compared nonlinear programming with other techniques (linear and dynamic) for a reservoir routing problem, and stated that nonlinear programming has not been as popular in water resources systems analysis as other methods due to the complication in implementing the technique and the difficulty to account for the stochastic nature of the system. Simpson et al. (1994) compared genetic algorithm (GA) techniques with complete enumeration and nonlinear optimization for a water distribution problem, and they concluded that the complete enumeration approach is only applicable with small problems with few pipes due to the heavy computational requirements. Nonlinear programming is an efficient technique when applied to small network. GA is an efficient technique with computational effort relatively high compared to nonlinear optimization, but very small compared to total enumeration. Yoon and Shoemaker (1999) compared different methods for a groundwater problem, including some evolutionary computational methods, some direct search methods, and some derivative-based optimization methods. In their study, the binary-coded genetic algorithm performed poorly, but an evolution strategy technique achieved a good balance between speed and accuracy. Other researchers refer to similar drawbacks of using gradient-based programming compared to genetic algorithm techniques in water resources problems (Wu and Simpson, 2001).

Regarding the current study, the complication of implementing gradient-based (nonlinear) programming can be explained by assuming a very simple network with 4 points (Figure 2.2) and considering the optimization model (Equations 2.1 to 2.5). The following points could be mentioned:

- The decision variables in the problem ( $B_1$ ,  $B_2$ , and  $g$ ) are not explicitly expressed in the optimization model. However, there is a system of differential equations related stated variables ( $A$  and  $u$ ) with decision variables included in equations F1 to F8. (Details of these equations are in section 3.3)
- Obtaining a relationship between any of the stated variables and decision variables, and their derivatives, is difficult. For example, defining a direct relationship between  $A_1$  and  $B_2$ , should be obtained through relationships of  $A_1$  with  $A_2$ ,  $A_2$  with  $A_3$ ,  $A_3$  with  $A_4$  and  $A_4$  with  $B_2$ . Considering the equations that



are used within the simulation model, and considering a typical example, like one that is used in this study, with hundreds of points and tens of decision variables, obtaining the derivatives would be very difficult.

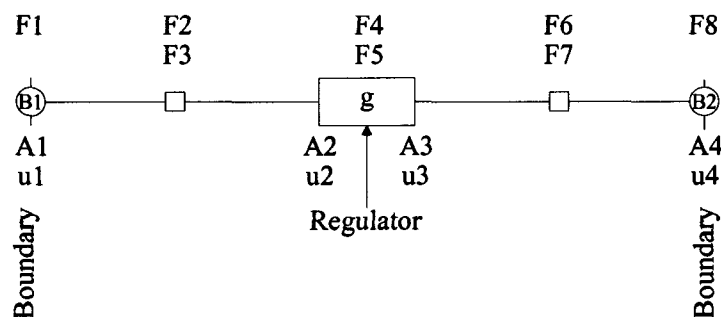


Figure 2.2  
Simple example of an irrigation canal network

- For some situations, relationships between stated variables (A and u), and some decisions variables do not exist. As an example, assume a gate operation during the routing with a given range (decision variable), and assume that the water level at this regulator during the time of the operation is less than the gate opening with the given range. In this case, this regulator will be treated as a constriction, and this decision variable will not be included in the system of the equations. Thus, one cannot obtain a relationship between any stated variable and this decision variable. This situation may happen frequently, especially in small channels.
- The fact that the problem is dynamic, where values of A and u are calculated for different time steps, and that the number of stated variables and decision variables keep changing from one time to the other, based on the operations or water shortage, and some variables should be treated stochastically, all increase the difficulty for using nonlinear programming in this problem.
- Another drawback of gradient-base optimization is that it can get trapped in local optima, and thus many policies (starting points) should be used to guarantee achieving optimal or near-optimal solutions.

Unlike traditional optimization techniques, direct search methods and evolutionary algorithms do not require the derivative information. They can be easily combined with simulation models by using the output of the simulation model to define the next step. An example of how these methods works was presented in Neelakantan and Pundarikanthan (2000) “In Hooke and Jeeves algorithm, the step length along the decision parameter axes is kept constant for each cycle of moves, and a probe is made first in the positive direction and then in the negative direction of each axis. Iterative improvement can get stuck in a local minimum, as the algorithm is essentially ‘greedy’ and accepts only those moves that optimize the objective function. As a result, the solution depends upon the starting configuration. Hence, several starting points (policies) are used to make sure that a better solution is found.” Many direct search methods were used with hydraulics problems, such as Hooke and Jeeves (Neelakantan and Pundarikanthan, 2000, Gates and Alshaikh, 1993) or Nelder and Mead (Yoon and Shoemaker, 1999) or response surface method (Gates et al., 1992). Comparing direct search methods with evolutionary computation, the following observations can be noted:

- Both direct search methods and evolutionary computational can easily incorporate a simulation model inside the procedure.
- Direct search methods are “greedy” optimization techniques that can get trapped at local optima, while evolutionary algorithms are more robust, and can move to optimal or near optimal solutions.
- Although direct search methods are considered faster in general, this may depend on different factors. One of these factors is the number of starting points that will be used with direct search methods to make sure a good solution is found. Also, the type of GA that is used associated with the parameters and constraint-handling technique, affects the rate of convergence as well as the accuracy. An example of this is what was concluded by Yoon and Shoemaker (1999) while comparing different optimization methods including direct search methods and evolutionary computational methods. They found that an evolution strategy method was the best in combination of speed and accuracy, while a binary-coded genetic algorithm performs poorly regarding the accuracy and the speed.

The current study will use a genetic algorithm to solve the optimization problem and the output of the simulation model will be used to evaluate each potential solution.

## 2.4 Genetic Algorithms (GAs)

Genetic Algorithms (GAs) are a class of techniques that mimic the processes of natural selection and genetic propagation in nature to “evolve” good solutions to a problem. The interest in genetic algorithms is mainly due to their ability to handle very complex problems, which do not easily fit into the traditional optimization frameworks. The GA search procedure maintains a population of potential solutions to the problem, each of which is represented as a string of design features. Unlike traditional optimization techniques, a GA requires no gradient information, but instead uses an evaluation function to determine the “fitness” or goodness of a solution. The GA-based search framework can incorporate complex simulation models without any simplification.

According to Davis (1987), genetic algorithms have five basic components:

- A genetic representation of a solution to the problem.
- A way to create an initial population of solutions.
- An evaluation function rating solutions in terms of their fitness.
- Genetic operators that alter the genetic composition of children during reproduction.
- Values of the parameters that the genetic algorithm uses (population size, crossover probability, etc.)

A global structure for genetic algorithms is shown in the Figure 2.3.

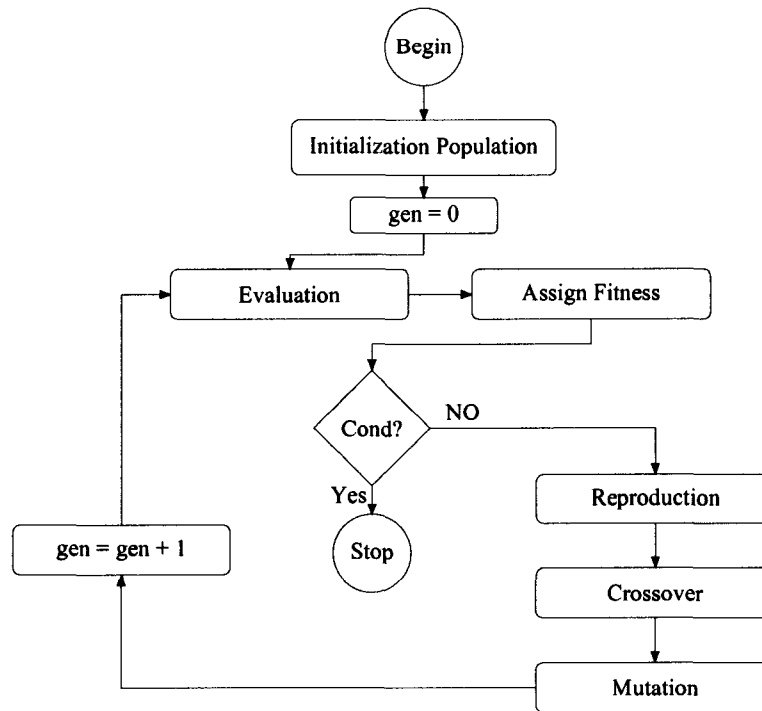


Figure 2.3  
Flow chart of genetic algorithm (Deb, 2001)

### 2.4.1 Representation

According to Herrera et al. (1998), “Representation is the key issue in GA work because GAs directly manipulate a coded representation of the problem and because the representation schema can severely limit the window by which a system observes its world.” Regarding representation types, there are two main categories, binary and real representation. Binary representation has dominated the field of GAs since its beginning until the early 1990’s. The reason for this is that there are theoretical results that show them to be the most appropriate ones, and they are amenable to simple implementation. However, binary representations have two main drawbacks: Hamming cliff, which means that two adjacent values are different in all of their bits, and redundancy, which means the decoding of a given code doesn’t belong to the domain. For most real-world problems, binary encoding is not the most suitable. According to Davis (1989), “We cannot handle most real-world problems with binary representations and an operator set consisting only of binary crossover and binary

mutation. One should incorporate real-world knowledge in one's algorithm by adding it to one's decoder or by expanding one's operator."

The other way to encode a real-world problem is real representation. The interest in real representation began in the 1990's. There are many advantages to real representations such as the following (Wright 1991, Gen and Cheng 2000; Michalewicz 1996, Herrera et al., 1989):

- It moves the genetic algorithm closer to problem space, as the distance between the points in the representation space is analogous to the distance between the points in the problem space.
- The use of real parameters makes it possible to use large domains for the variables.
- The capacity of real representation to exploit the *graduality* of the functions with continuous variables, where *graduality* refers to the fact that slight changes in the variables correspond to slight changes in the function.
- It increases the efficiency and the precision.
- It doesn't require a lot of memory.

The current study uses real representation to encode the decision variables.

#### 2.4.2 Evaluation

This step plays the role of the environment, and it rates solutions based on their fitness. Each potential solution (string) in the population will be evaluated using the objective function equation, or a simulation model, to check its fitness. This is a straightforward step in unconstrained optimization problems. However, in an optimization problem with constraints, a heuristic must be used to handle the constraints. Handling constraints in a GA can be challenging and will be discussed in detail in Chapter 5.

To evaluate each string, the unsteady flow model is used to route the flow, and the output from the model will be used to calculate fitness parameters. These outputs are calculated as follows:

- During the routing and for each time step, the following items will be calculated:

- ✍ The difference between the inflow discharge at the first point in the network, and outflow discharges from the downstream ends of the canals that convey water outside the network is calculated for each time step. The cumulative value of all these differences during all time steps presents the total water consumed (objective value).
- ✍ The water shortage will be checked at each time step. Whenever there is a zero or negative (numerically) water depth anywhere in the network, the program will assume that the part of the channel downstream is dry, and the cultivated area downstream of this point will be used to calculate a penalty for water shortage. If the end regulator of the channel is not closed, the program will add the cultivated area downstream this regulator to the shortage area. Even if the water comes back to this part of the channel during the routing, the program will still consider it as a violation of the first constraint. The only exception is with the operations. When a new channel is opened, the program will assume a traveling time for each opened reach, and if the reach is dry only during this time, the program will not consider this as a violation of the water shortage constraint.
- ✍ For the flood penalty, the program will determine all points that have a water level higher than the flood level at any time during the flow simulation. The total flooded length is used to calculate the flood penalty term. Regardless of the number of the time steps the water level exceeds the maximum allowable water level, the program will consider this as a violation of the second constraint.
- ✍ For the regulator stability, the program will check each regulator for the difference between upstream water level and downstream water level and compare this value against the maximum allowable difference of this regulator. If the difference between water levels is higher, this will be considered as a violation of the third constraint.
- At the end of the routing, the water volume in canals that have a required ending water level will be calculated and compared with the volume of the water based on the given required water levels. If the actual water volume is less than the

required ending water volume, this will be considered as a violation of the fourth constraint.

### 2.4.3 Selection

The selection operator imitates the natural selection and survival of the fittest in nature. It gives strings that have better fitness values a higher probability to get more copies, while strings with poor fitness values have a higher probability to die off. According to Gen and Cheng (2000), “Selection provides the driving force in genetic algorithms. With too much force, genetic search will terminate prematurely; with too little force, evolutionary progress will be slower than necessary”. The most commonly used selection procedures (Goldberg and Deb (1991), Gen and Cheng (2000), Runarsson and Yao (2000)) are:

- Proportionate Selection: in this class of selection, a chromosome has a probability to be selected proportional to its fitness. In these types of selection, the number of copies of an individual in any generation is related to the ratio between the fitness of this individual and the average fitness

$$P_{i,t+1} = P_{i,t} \frac{f_i}{f_i}$$

Proportionate selection can be preformed using roulette wheel, stochastic remainder selection, or stochastic universal selection. According to Goldberg and Deb (1991), proportionate selection is found to be significantly slower than other methods.

- Ranking selection: this technique was proposed by Baker (1985), then by Grefenstette and Baker (1989). In ranking selection, the population is sorted from the best to the worst, and assigns the number of copies that each individual should receive according to a non-increasing assignment function, and then performs proportionate selection according to that assignment.
- Tournament Selection: (Goldberg and Deb, 1991), tournament selection is based on randomly selecting a few strings and picking the best from them, and repeating until the mating pool is filled. The number of strings that is compared defines the

sub-category of this method. Binary tournament selection, where two strings are compared at a time is the most commonly used selection technique.

- Stochastic Random Selection: Runarsson and Yao (2000) proposed this method as a constraint-handling technique method to avoid the fine-tuning through using penalty functions. The idea is to use only the objective function or the constraints for the selection, rather than using the fitness function that is a combination of both. The one (objective or constraints) that will be used to determine the winning individual in the selection is chosen randomly. They suggested a probability between 0.4 & 0.5 for using the objective to rank the individuals (besides the case when both individuals are feasible, in which the objective function is used as well); otherwise the ranking will be based on the level of constraint satisfaction.

Three selection techniques were tested in the current study, which are:

- Binary tournament selection.
- Binary tournament selection with superiority of feasible solution.
- Stochastic tournament selection, which is a new proposed technique. The details about this technique are given in Chapter 5.

#### **2.4.4 Crossover**

The selection process increases the average fitness by increasing copies of good solutions and eliminating some bad solutions, but it doesn't add any new information to the problem. The way of exploring more of the search space is done through crossover and mutation. In crossover, two parents, from strings that survive after the selection process, will exchange a part of their data. Just a portion of the population will undergo crossover, while the rest of the population will move to the next generation as they are. The portion is defined by the crossover probability. The importance of this probability and suggested values will be discussed in Chapter 4.

According to the representation, there are two main categories of crossover, binary-coded crossover, and real-coded crossover.



### 2.4.4.1 Binary-coded crossover

Binary crossover is used for GAs with binary representations, and it has three different types, which are:

- One point crossover
- Two point crossover
- Uniform crossover

Figure 2.4 illustrates examples about these three types of binary crossover. In one point crossover, a location along the string length is selected at random, and all bits to the right of this location will exchange their data. In two point crossover, two locations are defined randomly, and the bits between these two locations will exchange their data. In uniform crossover, each bit in the first offspring decides (with some probability  $p$ ) which parent will contribute its value to it. The second offspring receives the bit from the other parent. The probability that is normally used within uniform crossover is 0.5, and so it could be done using a mask with digits of 0 and 1. If the value of the mask's chromosome is zero, each parent will give its value to the corresponding child (parent 1 for child 1 and parent 2 for child 2). If the value of the mask's chromosome is 1, the values of parents' chromosomes will be exchanged.

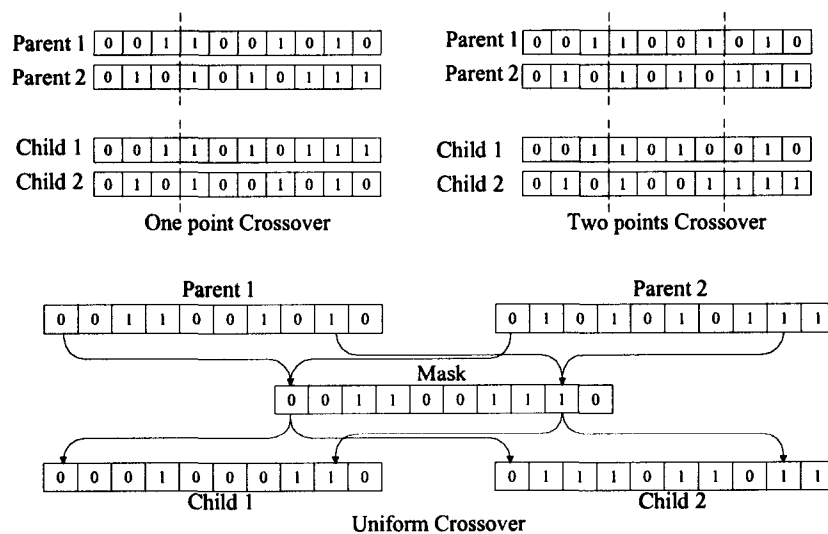


Figure 2.4  
Examples of binary-coded Crossover

#### 2.4.4.2 Real-coded crossover:

There are many real-coded crossover techniques have been proposed since the 1990's. The difference between these methods is how to generate the children from their parents. In linear crossover, reported by Wright (1991), three children are generated from two parents in the locations  $(-0.5P_1 + 1.5P_2)$ ,  $(0.5P_1 + 0.5P_2)$ , and  $(1.5P_1 - 0.5P_2)$  then the best two children will be chosen to the next generation. In simulated binary crossover (SBX) (Deb and Agrawal, 1995), new solutions will be randomly chosen from a specific probability distribution around the parents based on a random number  $u_i$  and a distribution index  $\eta_c$  as in Figure 2.5. A large distribution index indicates that the offspring will be close to their parents. In Unimodal Normally Distributed Crossover (UNDX) (Ono and Kobayashi, 1997), two children are generated from a region of normal distribution defined by three parents. These two children are generated around the center of mass of their parents. Simplex crossover (SPX) (Tsutsui et al., 1999) assigns a uniform probability distribution for creating offspring in a restricted search space around the region marked by the parents. In this method, the center of parents is calculated, then from a space defined by this point with the parents, a number of solutions (200 is suggested) is created, then two parents will be replaced by the best from these solutions and parent solutions. In blend crossover, proposed by Eshelman and Schaffer (1993), two children are generated from the range  $[p_2 + \alpha I, p_1 - \alpha I]$ , where  $p_1$  and  $p_2$  are the values of the parents,  $p_2 > p_1$ ,  $I = p_2 - p_1$ , and  $\alpha$  is a coefficients between 0 and 1. Many other types of real-coded crossover are listed in Herrera et al. (1998), Gen and Cheng (2000), and Deb (2001).

Figure 2.5 illustrates different types of real-coded crossover.

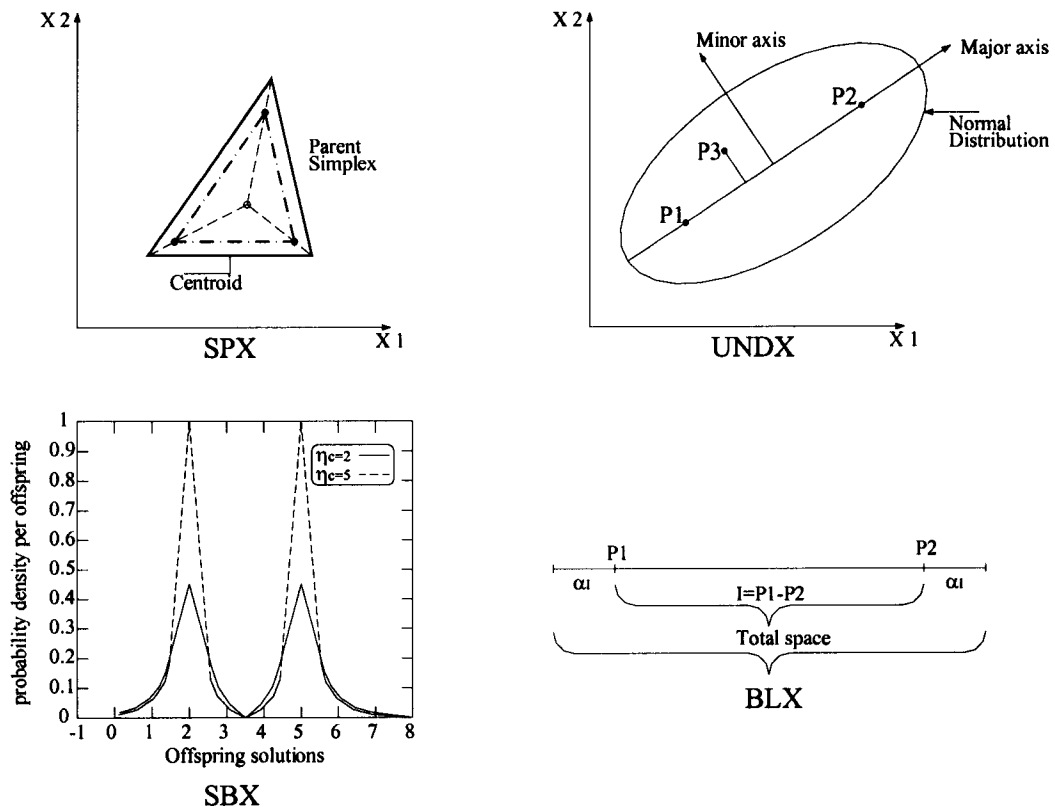


Figure 2.5  
Examples of real-coded crossover

Herrera et al. (1998) conducted an experiment to compare different binary and real coded crossover techniques, and they stated, “Generally, BLX- $\alpha$  crossover allows the best final results to be obtained. The higher the  $\alpha$  is, the better the results are. As  $\alpha$  grows, the exploration level is higher, since the relaxed exploitation zones spread over exploration zones, increasing the diversity levels in the population”

The current study uses blend crossover. The optimal value for blend crossover extension  $\alpha$  is discussed in detail in Chapter 4.

### 2.4.5 Mutation

Originally, the mutation operator was considered to be a background operator. According to Holland (1975), “mutation is a ‘background’ operator, assuming that the crossover operator has a full range of alleles so that the adaptive plan is not trapped on local optima.” However, later researchers argued about this fact, and they stated that mutation has a stronger role than previously recognized (Schaffer et al., 1989). The objective of mutation, like crossover, is to increase the variance of the population and prevent the GA from converging to local optima. In this step, the values of some strings that are selected randomly will be changed. In binary encoding, the value of the bit will be changed from 0 to 1 or vice versa. In real encoding, there are many proposed mutation implementations. The one that is used in the current study is random mutation, where a new random value will be selected between the maximum and minimum allowable values of the gene that will be mutated. The details of other different mutation techniques can be found in Herrera et al. (1989).

The number of strings that will undergo mutation is decided based on the mutation rate. The effect of the mutation rate, and suggested values will be discussed in Chapter 4.

## 2.5 Case study

An irrigation canal network in El Monofiya, Egypt is used as a case study (see Figure 2.6). In Egypt, the Nile River is the sole source of irrigation water. It provides Egypt with about 55.5 billion cubic meters of water per year, which barely meets the water demand (Abu-Zeid, 1992). It is expected that the water demand in Egypt will soon exceed the supply as the population increases. It is estimated that Egyptian agriculture consumes between 84% (Abu-Zeid and Rady, 1992) and 95% (Naff and Matson, 1984) of the water used in Egypt. Also, more water is consumed in Egyptian agriculture than in many other areas, primarily because of the wasteful use of irrigation water (Samah, 1979). This means that any plan to address the water supply for the future should include more efficient use of irrigation water. A part of the wasteful use of irrigation water is the result of the inability to determine efficient

strategies to make the best use of the irrigation water in the network. The network used for the case study is shown in Figure 2.6, and consists of man-made canals used mainly for irrigation purposes. The total cultivated area that is served by the network, on a rotating basis, is about 187,320 hectares (483,708 acres). All of the channels have mild slopes, as the longitudinal bed slope changes from 0.0 (horizontal bed) to 0.0001, and thus the flow is subcritical and water levels are gradually varied in the entire network. The network contains a main canal (El Monofy Rayah), for which the intake at the Nile River is the upstream end of the network.

All branches in the network divert from this main canal or from its branches. The case study considers the network from El Monofy Rayah intake to Meleg regulator (km 53.51 on El Monofy Rayah). In this reach of the main canal, there is one middle regulator, which is El Quarinien Regulator at km 29.30. There are two main branches: El Bagoriya Canal and Tanta Navigation Canal, which carry discharges to other directorates. The water is distributed through the branches on the basis of a periodic system, whereby a part of the network is opened for five days and then closed for ten days.

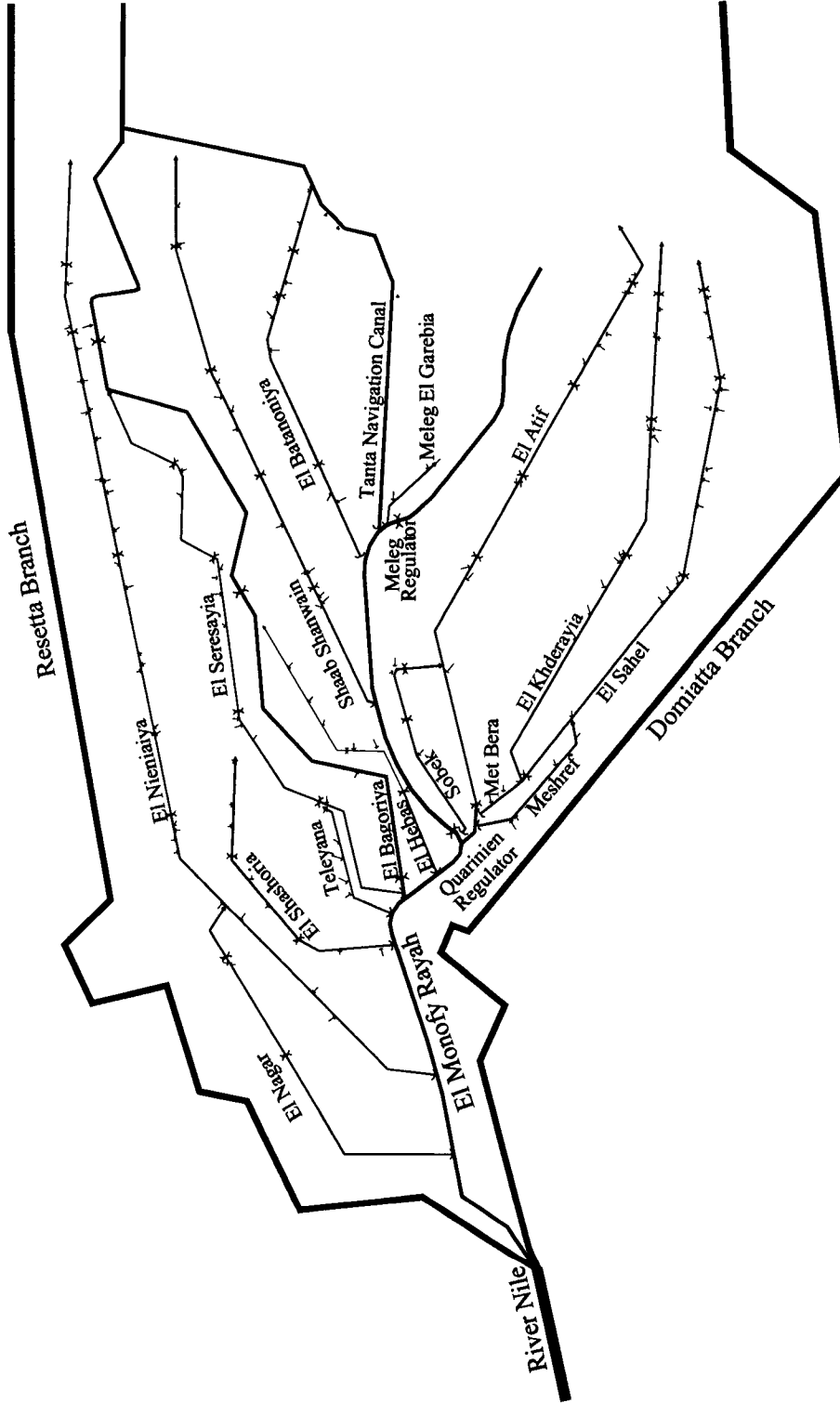


Figure 2.6  
El Monofiya irrigation canal network, Egypt. (Main branches)

### 2.5.1 The Data and its accuracy

The data used by the simulation model includes the following:

- Canals data: this category includes the canal length, the total cultivated land, number of regulators, and number of branches. These data tend to be very accurate.
- Reaches geometry data: this category includes the length, cultivated land area, and cross sectional area of each reaches. The accuracy may be affected if the actual cross sectional in some places has been changed from the design values. Also, the bank levels at each point are interpolated between the values at regulators and branches. The actual levels may deviate from this.
- Regulators design data: this category includes gate width, regulator bed level, cultivated land area downstream of the regulator, regulator thickness, and the maximum allowable difference between upstream water level and downstream water level. Also, this category of data includes the discharge coefficient of this regulator. The accuracy of the discharge coefficients is questionable especially with small regulators, where there are no field measurements to obtain empirical equations for them. In the absences of better information, the value 0.61 is used for such regulators.
- Initial data: these mainly are the initial water levels upstream of each canal and upstream and downstream of each regulator. Initial water levels were assumed with an average of levels at the time that was used for routing the flow.
- Boundaries and gate openings: the boundaries and gate openings are decisions variables unless they are fixed values. Downstream boundaries for canals that carry the water to downstream directorates will always be decisions variables. For some branches that the program will route only a part of them, the boundary might be fixed value, and it will be calculated based on the cultivated land area of the downstream part of this branch, and the average water consumption rate.
- Water consumption rates and crop allocation data: regarding the water consumption rate, the average values defined by the agricultural departments and by other previous researchers are used. For the crop allocation ratio, the ratios

were assumed based on the average ratios for crop allocation in Lower Egypt as was presented in previous studies (Ali, A. S. (1999), Ali, H. M. (2000), Fawzy, G. M. (1999), and El Qusoy, D. (1995)).

### 2.5.2 Suggested scenarios

Three scenarios of the case study are considered in this study. They are different in the number of decisions variables, number of the constraints and in the difficulty to find a feasible solution as a result of some sudden changes in the flow routing.

The first scenario (Figure 2.7) is the simplest one. It assumes that gate openings are constant during the whole run. The boundaries change gradually in four points and they are fixed in all other points.

This scenario consists of the following:

- Number of decision variables:
  - ✎ There are 19 decision variables as follows:
    - ✓ 11 initial gate openings (No operations).
    - ✓ 8 Boundaries conditions at 4 points (one upstream point and three downstream points at canals 1, 6, and 12).
- Number of constraints:
  - ✎ For both water shortage and flood: the model checks 646 points for 120 time steps
  - ✎ For regulator stability: the model checks 12 regulators for 120 time steps
  - ✎ For required water level: the model checks 83 points at the last time step
- Constraint violation tolerance:
  - ✎ Constraint violation tolerance for this scenario is zero meaning that the solution must satisfy each constraint perfectly to be considered feasible.

The boundaries at the end of all branches are fixed values, and one gate opening is assumed a free opened regulator.

Figure 2.8 displays the water level upstream and downstream of El Quarinien regulator. Water levels change smoothly during the routing. There is an effect from the initial condition in the first part of the routing,



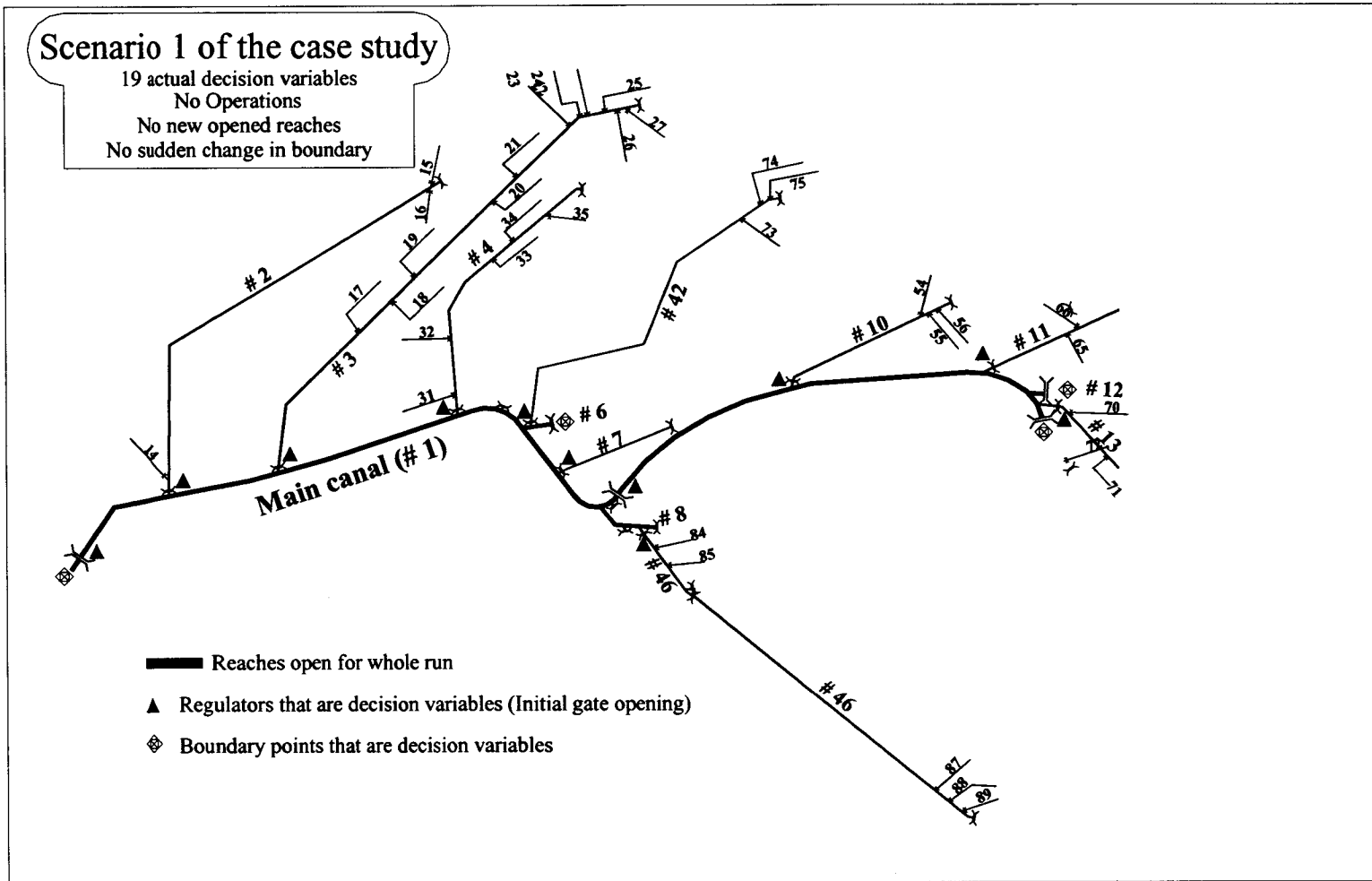


Figure 2.7  
First scenario of the case study

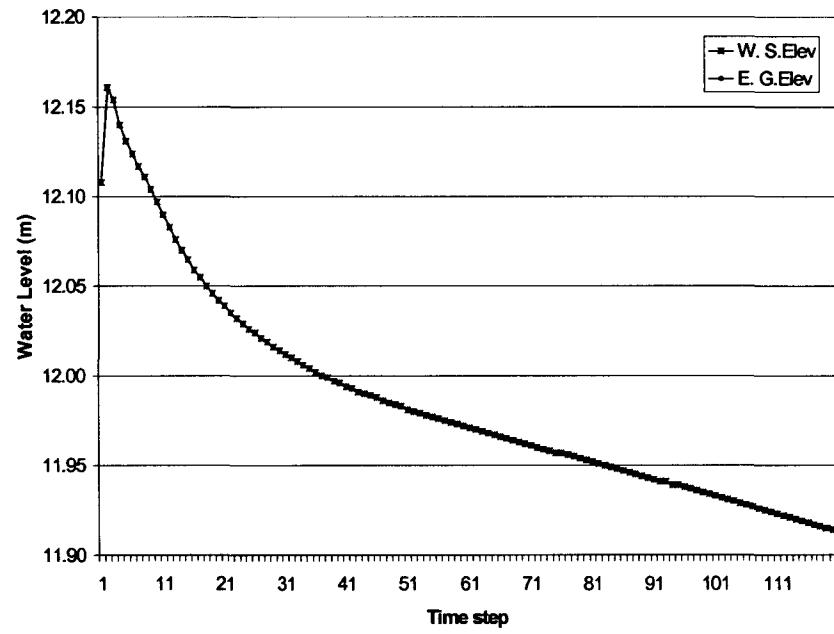
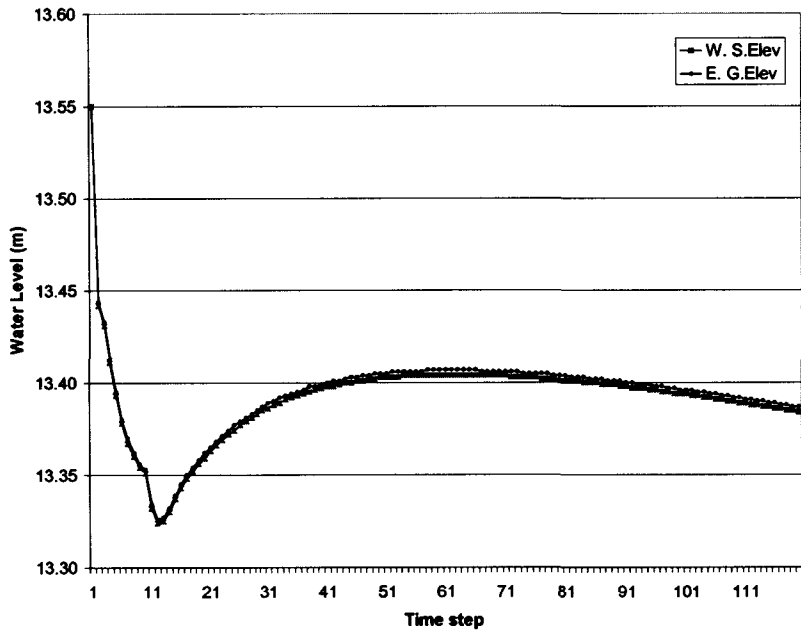


Figure 2.8  
Upstream and downstream water level of  
El Quarinien regulator during the whole run for the first scenario of case study

The second scenario (Figure 2.9) presents the case during a typical irrigation period, when there are few changes in the schedule of the operations. Also, this scenario presents the case when the amount of water delivered to the downstream directorates is changed between branches (Increase the discharge of one branch at the expense of other branches).

This scenario consists of the following:

- Number of decision variables:
  - ✍ There are 31 decision variables as follows:
    - ✓ 19 initial gate openings and 7 gate operations.
    - ✓ 12 Boundaries conditions at 5 points (one upstream point and four downstream points at canals 1, 6, 10, and 12).
- Number of constraints:
  - ✍ For both water shortage and flood: the model checks from 795 to 837 points for 120 time steps.
  - ✍ For regulator stability: the model checks from 15 to 18 regulators for 120 time steps.
  - ✍ For required water level: the model checks 83 points at the last time step
- Constraint violation tolerance:
  - ✍ Constraint violation tolerance for this scenario is as follows:
    - ✓ Water shortage: 0.005
    - ✓ Flood: 0.0
    - ✓ Regulators stability: 0.0
    - ✓ Required water levels: 0.01
- Operations in the main regulator
  - ✓ El Quarinien regulator: gate opening increased twice, at time step 24 and at time step 96.
- Boundary at the main outflow
  - ✓ Canal 1: gradually changes until time step 24, and then becomes constant.
  - ✓ Canal 6: suddenly decreases after 24 time steps.
  - ✓ Canal 12: suddenly increases after 36 time steps.

- Changes that have fixed values (Not decision variables)
  - ✓ Canal number 5 will be closed at time 96.

Figures 2.10 to 2.12 represent the water levels from one of the runs of this scenario. Figure 2.10 presents the water level upstream and downstream El Quarinien regulator. The effect of opening the gate at time step 24 is clear in the downstream, as is the effect of increasing boundary conditions at canal 12 at time step 36. Also, at the downstream, the increased difference between water surface elevation and energy grade line elevation indicate the increased velocity, and thus the discharge. At the upstream, the effect of decreasing the boundary of canal number 6 with increasing El Quarinien gate opening at time step 24 can be seen. Also, the effects of opening canals 9 and 46 at time step 48, and increasing El Quarinien gate opening after time step 96 are clear.

Figure 2.11 presents the water level upstream of the second regulator of canal 3. Water levels increase for the beginning, but the rate of increase changed after time step 24, when the gate opening of the intake increased. The water levels begin to decrease after this due to the opening of the second regulator.

Figure 2.12 presents the water level upstream of the intake regulator of canal 45. It is close to the water level upstream of El Quarinien regulator, as it shares it the same pool with no structures between them. The effect of opening the gate at time step 48 has no significant effect than the upstream of El Quarinien.

This change in the water levels during the routing increases the chance of violating any constraint, and thus finding a feasible solution is harder than for the first scenario.

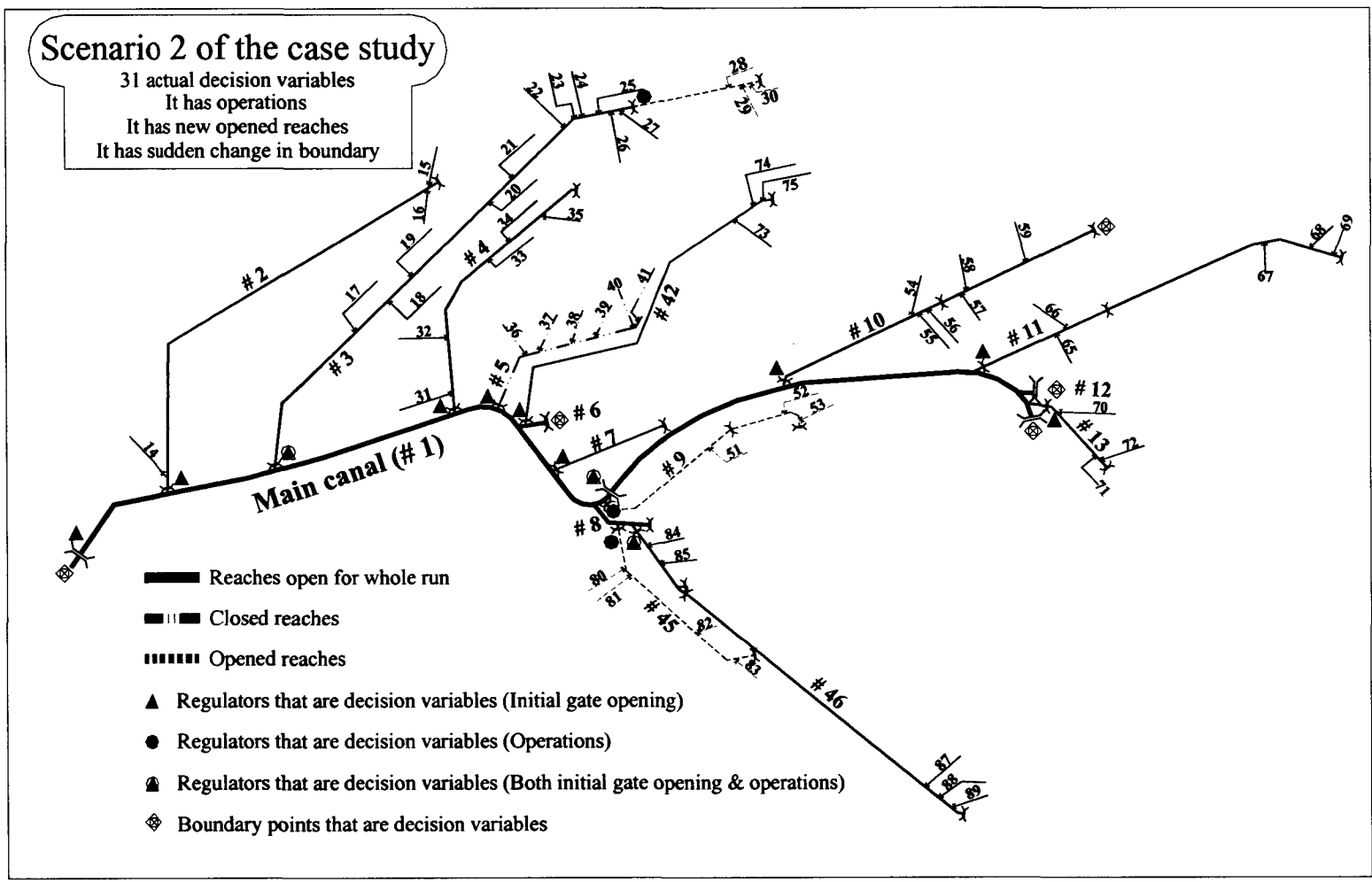


Figure 2.9  
 Second scenario of the case study

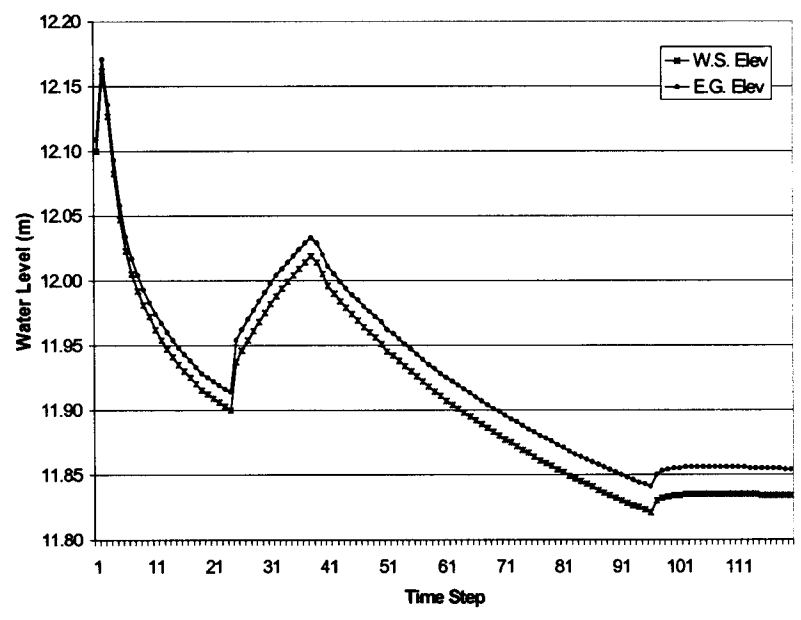
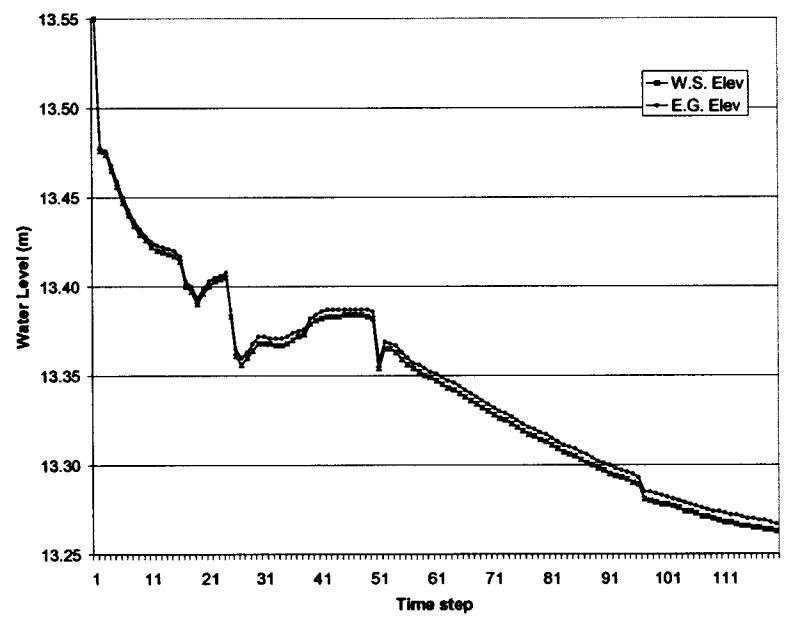


Figure 2.10  
Upstream and downstream water level of  
El Quarinien regulator during the whole run for the second scenario of case study

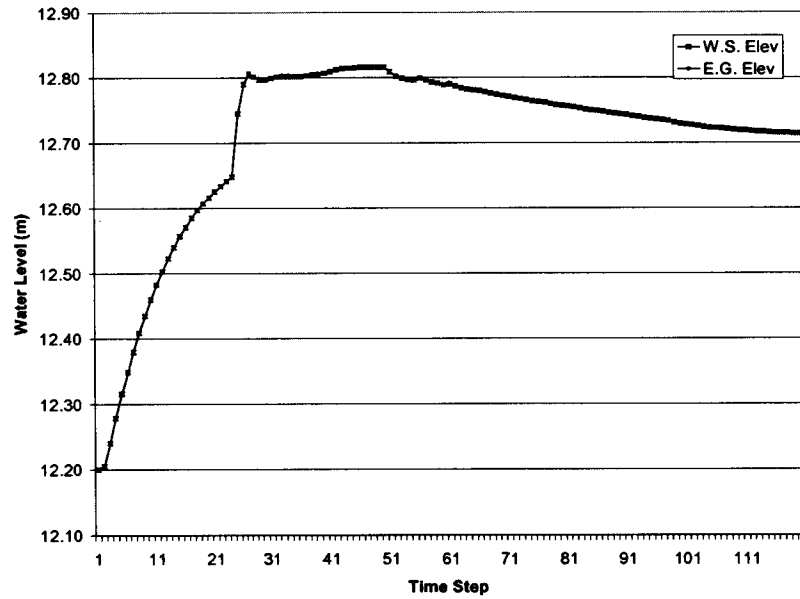


Figure 2.11  
Water level at the reach just before second regulator of canal 3 for the second scenario of case study

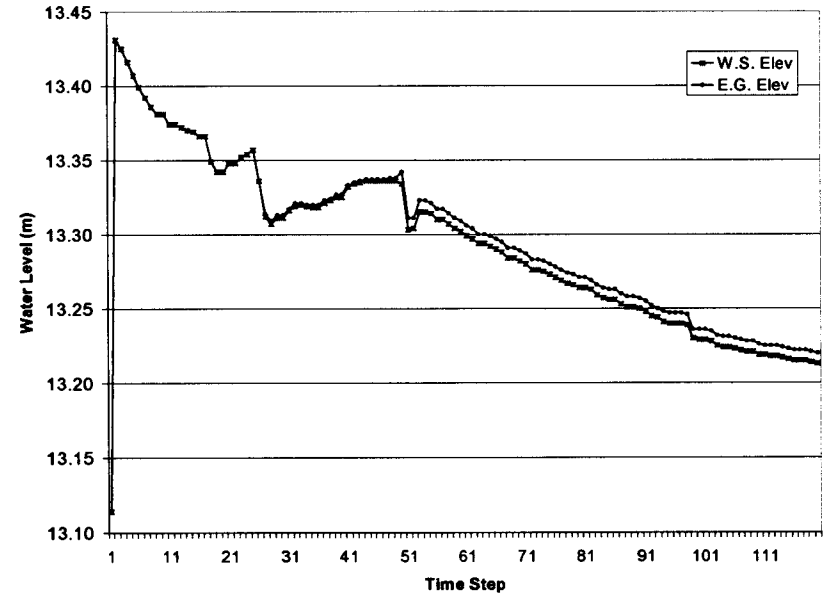


Figure 2.12  
Water level at the reach just before the intake of canal 45 for the second scenario of case study

The third scenario of the case study (Figure 2.13) represents a typical change of an irrigation period. The flow will be routed for 6 days, the last day in the current irrigation period with the five days of the next irrigation period. The irrigation period changes mainly from the branches upstream of El Quarinien regulator to the branches downstream of it, in addition to some other branches upstream it. The cultivated land for the new irrigation period is less than the cultivated land for the previous one, so the gate opening for El Monofy intake will be reduced, and the outflow to the directorates downstream of the network will increase. At the end of the routing, the gate opening of El Monofy intake will increase again to prepare the network for the next irrigation period.

This scenario consists of the following:

- Decision variables:
  - ✎ There are 52 decision variables as follows:
    - ✓ 14 initial gate openings and 18 gate operations.
    - ✓ 20 Boundaries conditions at 12 points (one upstream point and 11 downstream points at 11 different canals as in Figure 2.13).
- Constraints:
  - ✎ For both water shortage and flood: the model checks from 735 to 716 points for 144 time steps.
  - ✎ For regulator stability: the model checks from 15 to 14 regulators for 144 time steps.
  - ✎ For required water level: the model checks 83 points at the last time step.
- Constraint violation tolerance:
  - ✎ Constraint violation tolerance for this scenario is as follows:
    - ✓ Water shortage: 0.01
    - ✓ Flood: 0.005
    - ✓ Regulators stability: 0.0
    - ✓ Required water levels: 0.05
- Operations in the main regulator
  - ✓ First regulator: gate opening is decreased 2 times (time steps 12 and 36) and then it is increased 2 times (time steps 108 and 120).



- ✓ Second regulator: gate opening is increased 5 times (time steps 12, 24, 36, 48, and 60).
- Boundary at the main outflow
  - ✓ Canals 1, 6, and 12: Sudden increase after 24 time steps
- Changes that have fixed values (Not decision variables)
  - ✓ Six branches that divert from the main canal are closed (3 after 24 hours, and other 3 after another 12 hours).
  - ✓ The boundary of 11 small branches that divert from canal 3 will change after 24 hours to 0.0.

Figures 2.14 to 2.16 present the water levels in some points of the network during the routing in one of the runs of this scenario.

- The water level upstream El Quarinien is decreasing until time step 109 when it begins to increase again as an effect of increasing the gate opening of El Monofy intake.
- The water level downstream of El Quarinien is increasing until time step 24, then it begins decreasing when two main branches downstream of it are opened, and the discharge to other directorates increases. From time step 100, it begins to increase again. With the decreasing water level, the difference between water surface and energy grade line elevation increases meaning that the velocity increases. In a typical run of this scenario, the discharge increases from 43.7 m<sup>3</sup>/sec at the beginning of the routing to 83.8 m<sup>3</sup>/sec at the end of the routing.
- Figure 2.15 presents the last point in canal 3 before the second regulator that was opened at time step 24. Also Figure 2.16 presents the point on canal 46 upstream of canal 86 that was opened at time step 12. The effect of opening new reaches or new canals is clear.

This increase of the decision variables with the sudden changes of the boundaries increases the difficulty in finding a feasible solution unless the decision variables are chosen suitably.

These three scenarios of the case study present different levels of difficulty to find feasible solutions and will be used to check the best parameters that should be used within the GA and suitable constraint-handling techniques in later chapters.

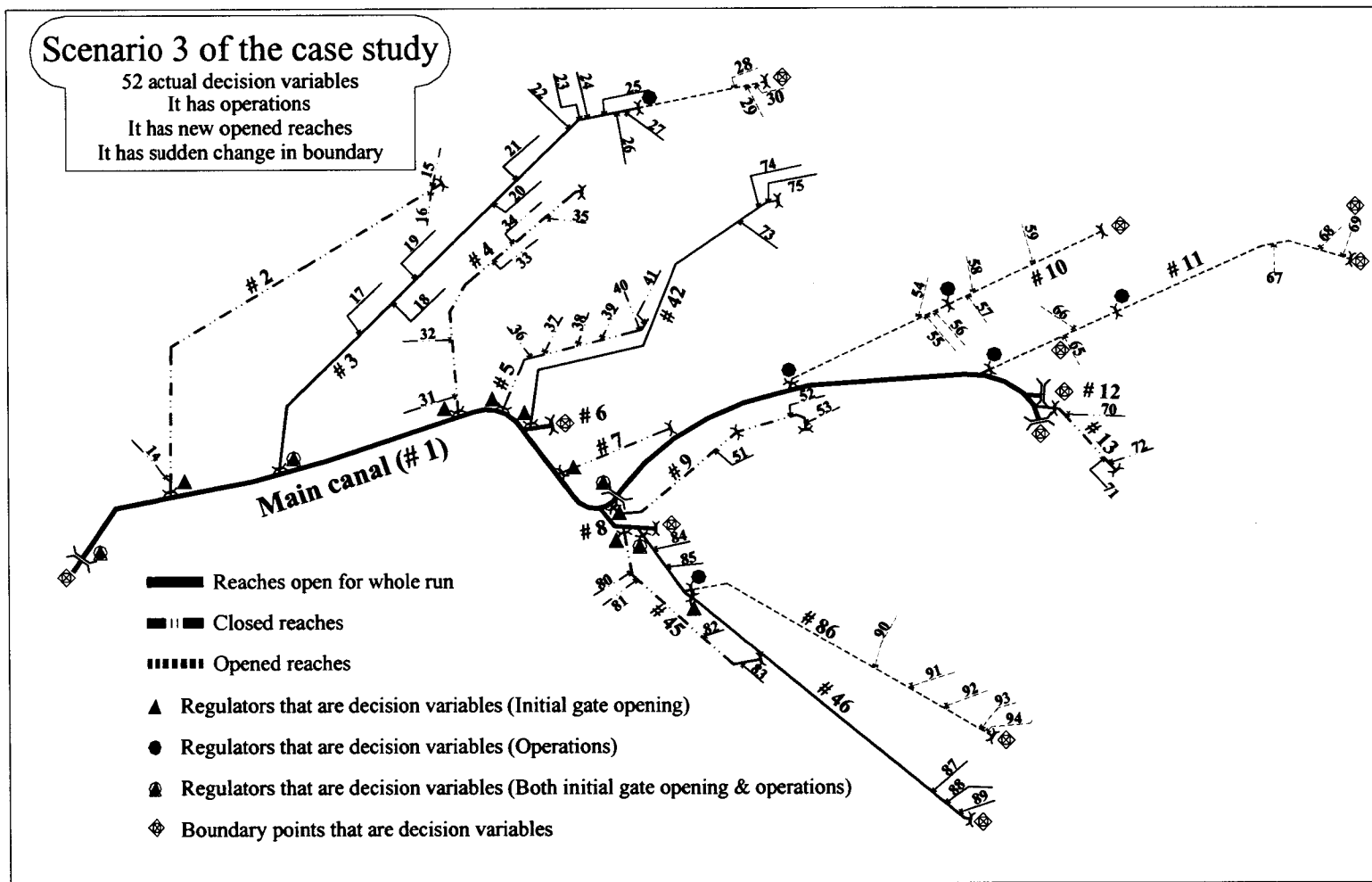


Figure 2.13  
 Third scenario of the case study

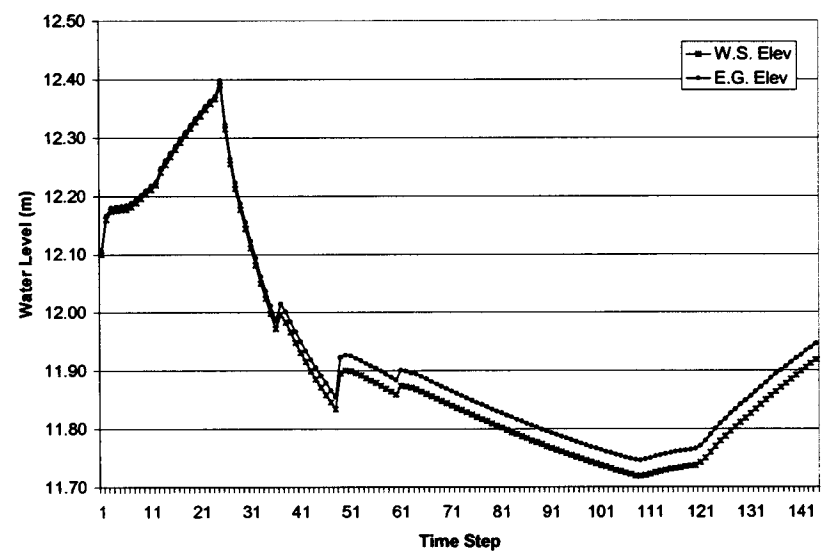
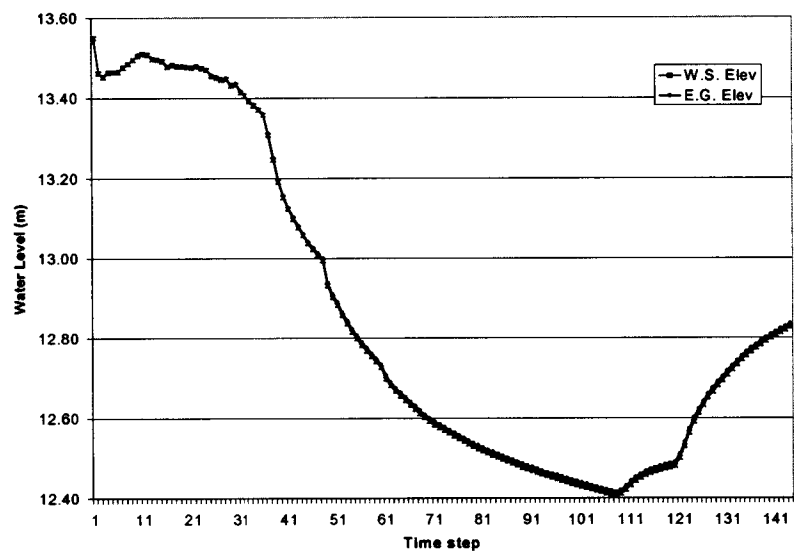


Figure 2.14  
Upstream and downstream water level of  
El Quarinien regulator during the whole run for the third scenario of case study

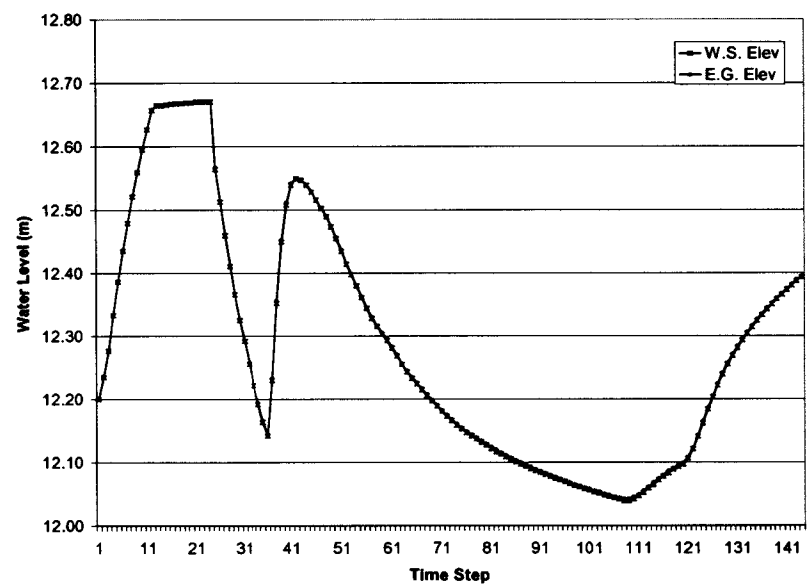


Figure 2.15  
Water level at the reach just before second regulator of canal 3 for the third scenario of case study

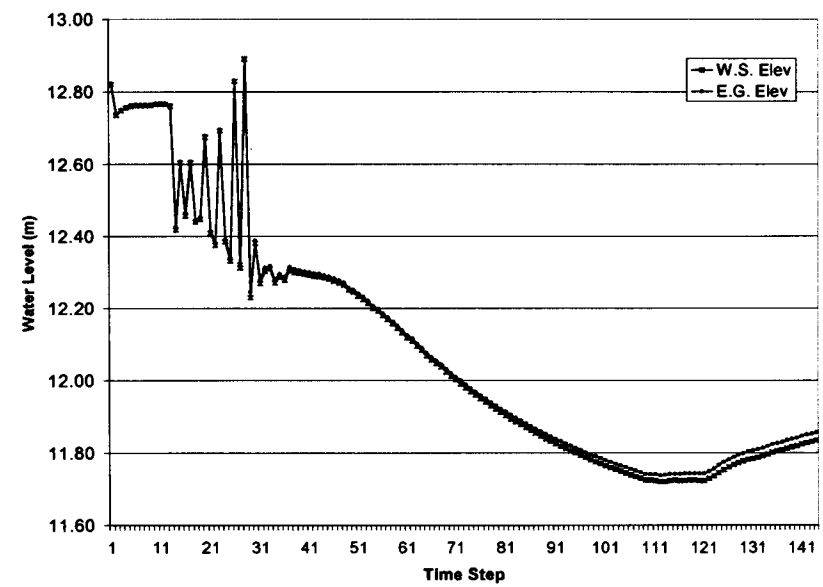


Figure 2.16  
Water level at reach 2 of canal 46 for the third scenario of case study

## 2.6 References

- Abu-Zeid, M. A., "Water Resources Assessment for Egypt," *paper presented at roundtable on Egyptian Water Policy*, Alexandria, Egypt, April 11-13, 1992.
- Abu-Zeid, M. A. and Rady, M. A. "Water Resources Management and Policies in Egypt," *Country Experiences with Water Resources Management, World Bank Technical Paper Number 175* (World Bank, Washington, D.C.), 1992, pp. 93-101
- Ali, A. S., "Water productivity in Egyptian Agriculture," *International conference on integrated management of water resources in the 21<sup>st</sup> century*, Cairo, Egypt, November 21-25, 1999, pp. 478-487
- Ali, H. M., "Determining optimal crop pattern in Egypt by the use of multicriteria analysis," *Water Science (The National Water Research Center Magazine, Ministry of Water Resources and Irrigation, Egypt)*, 28<sup>th</sup> – 29<sup>th</sup> Issue, October 2000-April 2001, pp. 31-36
- Baker, J. E., "Adaptive Selection Methods for Genetic Algorithms," *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, 1985, pp. 100-111.
- Davis, L., "Adapting Operator Probabilities in Genetic Algorithms," *proceedings of the third International Conference on Genetic Algorithms*, Morgan Kaufman, 1989 pp. 61-69
- Davis, L., *Genetic Algorithms and Simulated Annealing*, Morgan Kaufman Publishers, San Mateo, CA, 1987
- Deb, K. and Agrawal, S., "Simulated binary crossover for continuous search space," *Complex Systems*, Vol. 9, No. 2, 1995, pp. 115–148
- Deb, K. and Kumar, A., "Real-coded genetic algorithms with simulated binary crossover: Studies on multi-modal and multi-objective problems," *Complex Systems* Vol. 9, No. 6, 1995, pp. 431–454
- El Qusoy, D., "Possibilities of using the treated sewage water in Irrigation" (In Arabic), *Annual conference of the National Water Research Center*, Cairo, Egypt, December 1995, 26-27

- Eshelman, L. J. and Schaffer, J. D., "Real-coded Genetic Algorithms and Interval-Schemata," *Foundations of Genetic Algorithms*, Vol. 2, Edited by Whitley, L. D., 1993, pp. 187-202
- Fawzy, G. M. "The efficient use of irrigation water in the Egyptian agriculture," *International conference on integrated management of water resources in the 21<sup>st</sup> century*, Cairo, Egypt, November 21-25, 1999
- Gates, T. K., Alshaikh, A. A., Ahmed, S. I., and Molden, D. J., "Optimal Irrigation Delivery System Design Under Uncertainty," *Journal of Irrigation and Drainage Engineering*, American Society of Civil Engineers, Vol. 118 No. 3, MAY./JUN. 1992, pp. 433-449
- Gates, T. K., and Alshaikh, A. A., "Stochastic Design of Hydraulic Structures in Irrigation Canal Networks," *Journal of Irrigation and Drainage Engineering*, American Society of Civil Engineers, Vol. 119 No. 2, MAR./APR. 1993, pp. 346-363
- Gen, M. and Cheng R., *Genetic Algorithms & Engineering Optimization*, John Wiley & Sons, New York, 2000
- Goldberg, D.E. and Deb, K. A "Comparative Analysis of Selection Schemes Used in Genetic Algorithms," *Foundation of Genetic Algorithms*, Edited by Rawlins, G. J. E., 1991, pp. 69-93
- Grefenstette, J. J. and Baker J. E., "How Genetic Algorithms work: A critical look at implicit parallelism," *proceedings of the third international Conference on Genetic Algorithms*, 1989, pp. 20-27
- Herrera, F., Lozano, M. and Verdegay, J. L. "Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis," *Artificial Intelligence Review* Vol. 12, No. 4, 1998, pp. 265– 319
- Holland, J. H. *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI., 1975
- Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, New York, 1996
- Naff, S. and R. C. Matson, *Water in the Middle East*, Westview Press, Colorado, 1984

- Neelakantan, T. R., and Pundarikanthan, N. V. "Neural Network-Based Simulation-Optimization Model For Reservoir Operation," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 126 No. 2, MAR./APR. 2000, pp. 57-64
- Ono, I., and Kobayashi, S., "A Real-Coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover," *proceedings of the seventh international Conference on Genetic Algorithms*, 1997, pp. 246-253
- Runarsson, T. P., and X. Yao, "Stochastic Ranking for Constrained Evolutionary Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 4, pp. 284-294, Sept 2000
- Schaffer, J., Caruana, R., Eshelman, L., and Das R., "A Study of Control Parameters Affecting Online Performance of Genetic Algorithms for Function Optimization," *proceedings of the third International Conference on Genetic Algorithms*, Morgan Kaufman, 1989, pp. 51-60
- Samah, M. A. "The Egyptian Master Water Plan," *Water Supply and Management*, Vol. 3, 1979, pp. 251-266
- Simpson, R. A., Dandy, G. C., and Murphy, L. J., "Genetic-Algorithms compared to other techniques for pipe optimization," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 120 No. 4, JUL./AUG. 1994, pp. 423-443
- Tsutsui, S., Yamamura, M., and Higuchi, T., "Multi-Parent Recombination With Simplex Crossover in Real-Coded Genetic Algorithms," *proceedings of the Genetic and Evolutionary Computation Conference (GECCO-1999)*, pp. 657-664
- Yeh, W. W. G. "Reservoir Management and Operations Model: A State-of-the-Art Review," *Water Resources Research*, Vol. 21, No 12, DEC 1985, pp. 1797-1818
- Yoon, J-H., and Shoemaker, C. A., "Comparison of Optimization Methods for Ground-Water Bioremediation," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 125 No. 1, JAN./FEB. 1999, pp. 54-63
- Wright, A. *Genetic Algorithms for real parameter optimization*, Foundation of Genetic Algorithms Edited by Rawlins, G.J.E., 1991, pp. 205-218



Wu, Z. Y., and Simpson, R. A., "Competent Genetic-Evolutionary Optimization of Water Distribution Systems," *Journal of Computing in Civil Engineering*, American Society of Civil Engineers, Vol. 15, No. 2, April 2001, pp. 89-101

## CHAPTER 3

### THE UNSTEADY FLOW MODEL

#### 3.1 Introduction

To evaluate the solutions in the GA-based optimization model that was defined in Chapter 2, an unsteady flow model is used. The model is based on the unsteady flow equations with other equations for the junctions, the regulators, and the constrictions. The implicit method was used to express the equations mathematically, with a weighting factor of 1.0. The Newton-Raphson method was used to solve the system of the equations, with some modifications to save memory and computational effort. The model is designed to handle operations during the routing, and a new technique for zero or negative (numerically) water depth that can achieve the stability without affecting the accuracy is proposed. A summary of the unsteady model that is used in the current study is given in this chapter. More complete description can be found in El Gamel (2001).

#### 3.2 Governing equations and their solution

##### 3.2.1 Governing equations

The governing equations for routing the flow through the reaches, the junctions, the constrictions, and the regulators (sluice gates) in a canal network are as follows:

##### 3.2.1.1 Governing equations for the reaches

The complete Saint Venant equations are used to route the flow in the reaches, and have the well-known form:

$$\frac{\partial A}{\partial t} + \frac{\partial(Au)}{\partial x} - q = 0 \dots\dots\dots(3.1)$$

$$s_0 - s_f = \frac{1}{g} \frac{\partial u}{\partial t} + \frac{u}{g} \frac{\partial u}{\partial x} + \frac{\partial y}{\partial x} \dots\dots\dots(3.2)$$

Where:

A= cross sectional area (L<sup>2</sup>).

u= mean velocity (LT<sup>-1</sup>).

s<sub>0</sub>= longitudinal bed slope.

s<sub>f</sub>= friction slope.

y= water depth (L)

g= acceleration due to the gravity (LT<sup>-2</sup>).

x= distance (L).

t= time (T).

These two equations represent the continuity and the momentum equations. The same equations can be represented in the following form:

$$\frac{\partial Q}{\partial x} + \frac{\partial A}{\partial t} - q = 0 \dots\dots\dots(3.3)$$

$$\frac{\partial Q}{\partial t} + u \frac{\partial Q}{\partial x} + gA \frac{\partial y}{\partial x} - gA(s_0 - s_f) - qu = 0 \dots\dots\dots(3.4)$$

Where:

q= lateral inflow or outflow (LT<sup>-2</sup>), defined as positive in inflow and negative in outflow, and s<sub>f</sub> can be calculated using the Manning equation

The equations use the cross sectional area and the velocity as variables. Equations 3 and 4 are used to route the flow through each reach in the network. For the junctions, regulators, and bridges, and the energy equation will substitute for the momentum equation.

### 3.2.1.2 Governing equations for the junctions and constrictions

In the junctions and constrictions (see Figure 3.1), the energy equation is used with the continuity equation to route the flow. The energy equation between two sections can be expressed as follows:

$$z_1 + y_1 + \frac{u_1^2}{2g} = z_2 + y_2 + \frac{u_2^2}{2g} + h_f \dots\dots\dots(3.5)$$

In the junctions, the head loss is negligible. For the constrictions (see Figure 3.1), the following equation, presented by Chow (1959), can be used to calculate the friction loss  $h_f$ :

$$h_f = L_a \left( \frac{Q}{\sqrt{K_1 K_3}} \right)^2 + L \left( \frac{Q}{K_3} \right)^2 \dots\dots\dots(3.6)$$

Where

$L$ : Regulator thickness.

$L_a$ : Acceleration length.

$K$  = the total conveyance that can be calculated as:

$$K = \frac{\phi}{n} AR^{2/3} \dots\dots\dots(3.7)$$

In the current study, the acceleration length is assumed to be zero and equation (3.6) becomes:

$$h_f = L \left( \frac{Q}{K_2} \right)^2 \dots\dots\dots(3.8)$$

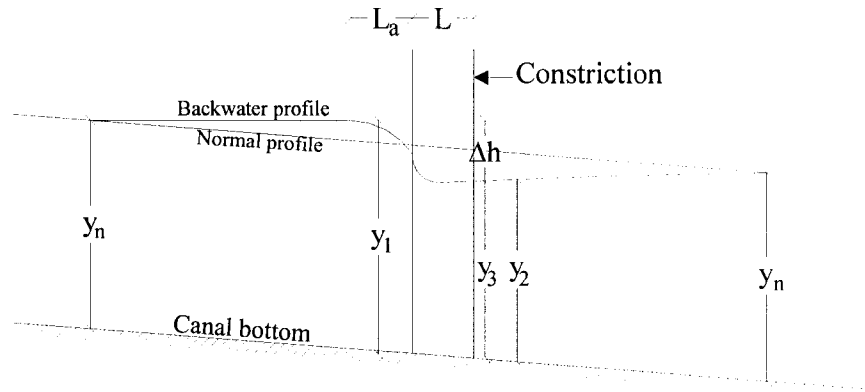


Figure 3.1  
Flow through constrictions

3.2.1.3 Governing equations for the sluice gates

An equation for the flow through a sluice gate (see Figure 3.2) can be obtained by applying the energy equation between the water sections upstream and the downstream of the gate, assuming that the energy loss through the gate is negligible and the pressure distribution is hydrostatic. According to Rajaratnam and Subramanya (1967), the sluice gate equation can be represented as follows:

$$q_G = \frac{c_c}{\sqrt{\alpha_s - c_c^2 (og / y_1)^2}} og \sqrt{2g(y_1 - y_3)} \dots \dots \dots (3.9)$$

Where:

$q_G$  = discharge through the regulator per unit width ( $L^2T^{-1}$ ).

$og$  = height of the gate opening (L).

$c_c$  = contraction coefficient.

$\alpha_s$  = kinetic energy correction factor.

Since  $y_2$ , rather than  $y_3$ , is typically recorded in an irrigation canal network, the previous equation was modified for use in the model as follows:

$$Q_G = c_d * o g * b * \sqrt{2g(y_1 - y_2)} \dots\dots\dots(3.10)$$

Where:

$Q$ = the discharge through the regulator.

$b$ = the width of the gate opening.

$c_d$  is calibrated for each regulator using field measurements.

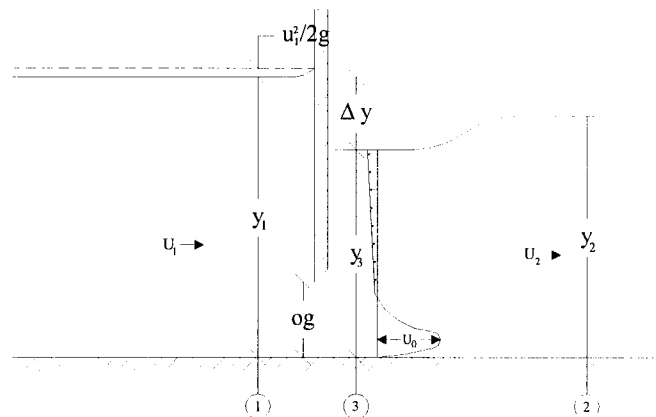


Figure 3.2  
Sluice gate equation

#### 3.2.1.4 Governing equations for submerged hydraulic jumps

The submerged jump exists when the actual tail water depth is greater than the corresponding tail water depth due to the free jump (see Figure 3.3). This phenomenon occurs downstream of the sluice gates when the flow is subcritical.

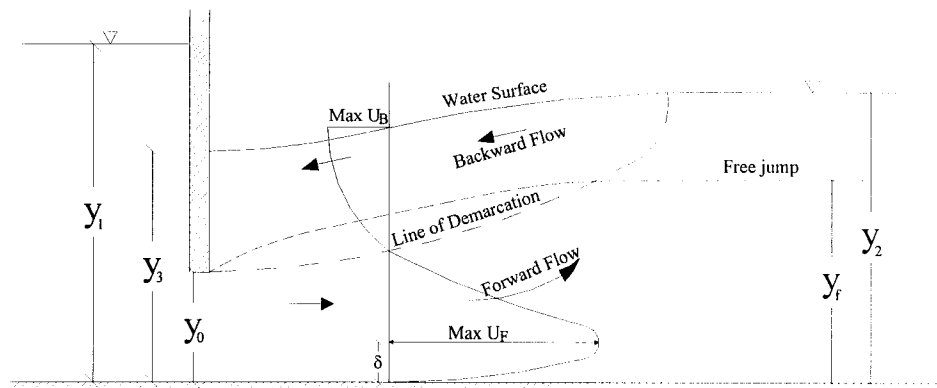


Figure 3.3  
Submerged jump

The hydraulic jumps create energy losses that can be calculated by applying the energy equation upstream and downstream of the jump. Different equations have been presented in the literature to calculate the energy losses due to submerged hydraulic jumps (Chow, 1959; GovindaRao and Rajaratnam, 1963; Ohtsu et al., 1999). The equations presented by Ohtsu et al. (1999) are used in the model presented herein. The ratio between the head loss and the energy at sections 3 is calculated as follows:

$$\frac{\varepsilon}{E_3} = \frac{2(Y_3 - Y_2) + F_0^2 \left[ 1 - \left( \frac{k}{Y_2} \right) \right]}{2Y_3 + F_0^2} \dots\dots\dots(3.11)$$

Where:

$$E_3 = y_3 + \frac{u_0^2}{2g} \dots\dots\dots(3.12)$$

$$Y_3 = \text{relative water depth at section 3} = \frac{y_3}{og}$$

$$Y_2 = \text{relative water depth at section 2} = \frac{y_2}{og}$$

$y_3$  = water depth at section 3 just behind the regulator (L)

$u_0$  = mean velocity through the gate ( $LT^{-1}$ )

$k$  = the ratio between the regulator width and the downstream width.

Also Ohtsu et al. (1999) presented the following relationship between  $Y_3$  and  $Y_2$ :

$$Y_2^3 - (Y_3^2 + 2F_0^2 k)Y_2 + 2F_0^2 k^2 = 0 \dots\dots\dots(3.13)$$

Equation (3.13) is based on the assumption that the pressure is hydrostatic and the momentum correction coefficient is unity for both sections upstream and downstream of the jump.

### 3.2.2 Solution of the governing equations

#### 3.2.2.1 Solving the equations for the reaches

The governing equations presented above cannot be solved analytically; thus, a numerical model is used to solve them. There are two numerical methods that can be used to solve the unsteady flow equations: the method of characteristics and the fixed points method. The method of characteristics is a technique that converts two simultaneous partial differential equations to four ordinary differential equations (Abbott, 1975). The interest in this method has decreased in the last few decades, but it is still often used as the boundary equation in the fixed points explicit methods. The main drawback of the method of characteristics is that it calculates the flow in non-fixed locations and times.

The fixed points methods, either explicit or implicit, use the finite difference scheme to approximate the derivatives of the partial differential equations. These methods depend on filling the plane of  $(x,t)$  with a grid representing the required locations and times to calculate the flow variables. The finite difference approximations are based on the Taylor series and express the derivative of the function based on the discrete points as follows:



$$\frac{\partial f}{\partial x} = \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta(x)} \dots\dots\dots(3.14)$$

The explicit scheme solves the flow for one point at a time. The calculation is easier and the requirement of the memory is less than the implicit method, but the stability of these schemes is restricted by the Courant number, which requires that the computed wave celerity is greater than or equal to the actual wave celerity.

Various explicit schemes have been developed, including the Leap Frog method, the Lax-Wendroff second order scheme, and Dronkers' Explicit scheme (Abbott and Basco, 1989; Dronkers, 1964). The implicit scheme is more robust and it has no restriction for the time interval. It solves the equations for all points of the canal at once for each time step. Although the system of equations is more complicated, the accuracy is better and the time interval is larger than the explicit methods. Preissmann and Cunge presented the first implicit scheme in the early 1960s (Liggett and Cunge, 1975).

The implicit scheme (see Figure 3.4), expresses the variables at one point as a function of the conditions at four surrounding points. These four points represent the current and the advanced location and the current and the advanced time. Preissmann and Cunge expressed the partial differential equations using a finite difference technique and then linearized the equations.

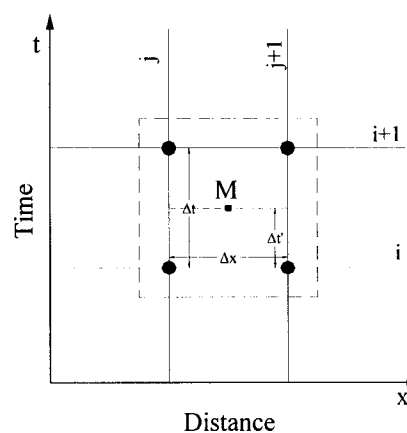


Figure 3.4  
Preissmann implicit scheme

The system consists of two equations with four unknowns at each node. For a reach of N points, the system will contains 2N-2 equations with 2N unknowns. Adding 2 boundary equations, a unique system of 2N equations and 2N unknowns is obtained.

Other implicit schemes were presented in the 1960s and 1970s. The most important of them is the Amein four points scheme (Amein and Fang, 1970). This scheme expresses the variables at each point using four surrounding points, as in the Preissmann scheme.

However, Amein and Fong suggested solving a system of nonlinear equations instead of linearizing the equations. Amein and Fong solved the following unsteady flow equations:

$$u \frac{\partial A}{\partial x} + A \frac{\partial u}{\partial x} + \frac{\partial A}{\partial t} - q = 0 \dots\dots\dots(3.15)$$

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + g \frac{\partial y}{\partial x} = g(S_0 - S_f) - \frac{qu}{A} \dots\dots\dots(3.16)$$

Defining the variables at point M using the four points surrounding it as in Figure 3.5 as follows:

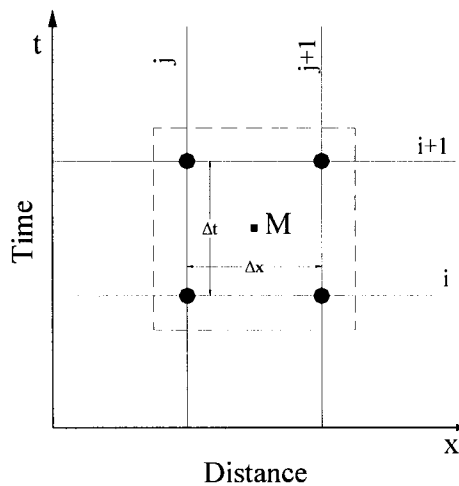


Figure 3.5  
Amein implicit scheme

$$\frac{\partial \alpha(M)}{\partial t} = \frac{1}{2\Delta t} [(\alpha_j^{i+1} + \alpha_{j+1}^{i+1}) - (\alpha_j^i + \alpha_{j+1}^i)] \dots \dots \dots (3.17)$$

$$\frac{\partial \alpha(M)}{\partial x} = \frac{1}{2\Delta x} [(\alpha_{j+1}^{i+1} + \alpha_{j+1}^i) - (\alpha_j^i + \alpha_j^{i+1})] \dots \dots \dots (3.18)$$

$$\frac{\partial \alpha(M)}{\partial t} = \frac{1}{2\Delta t} [(\alpha_j^{i+1} + \alpha_{j+1}^{i+1}) - (\alpha_j^i + \alpha_{j+1}^i)] \dots \dots \dots (3.19)$$

Substituting equations 3.17 through 3.19 into equations 3.15 and 3.16, the following 2 equations are obtained:

$$F_i = \frac{1}{2 * \Delta t} (y_{j+1}^{i+1} + y_j^{i+1} - y_{j+1}^i - y_j^i) + \frac{1}{2\Delta x} (u_{j+1/2}^{i+1/2}) * (y_{j+1}^i + y_{j+1}^{i+1} - y_j^i - y_j^{i+1}) \\ + \frac{1}{2\Delta x} * \left(\frac{A}{T}\right)_{j+1/2}^{i+1/2} * (u_{j+1}^{i+1} + u_{j+1}^i - u_j^i - u_j^{i+1}) - \left(\frac{q}{T}\right)_{j+1/2}^{i+1/2} \dots \dots \dots (3.20)$$

$$F_{i+1} = \frac{g}{2\Delta x} (y_{j+1}^{i+1} + y_{j+1}^i - y_j^{i+1} - y_j^i) + \frac{1}{2\Delta t} (u_{j+1}^{i+1} + u_j^{i+1} - u_{j+1}^i - u_j^i) + \frac{1}{2\Delta x} u_{j+1/2}^{i+1/2} \\ (u_{j+1}^{i+1} + u_{j+1}^i - u_j^{i+1} - u_j^i) + \frac{g}{4} (s_f)_{j+1/2}^{i+1/2} - \left(\frac{g}{\Delta x}\right) (z_j - z_{j+1}) + q \left(\frac{u}{A}\right)_{j+1/2}^{i+1/2} \dots \dots \dots (3.21)$$

As in the Preissmann scheme, the unsteady flow equations with the boundary equations will generate a unique system of equations. The Amein four points scheme is used in the model presented herein. Other implicit schemes and implementations of the previous scheme for different studies can be found in the literature (Fread, 1971 and 1973, Quinn and Wylie, 1972, Amein and Chu, 1975, and Fread and Smith, 1978).

### 3.2.2.2 Solving the equations for the junctions

Several suggestions for routing the flow through channel junctions can be found in the literature (Stoker, 1957, Li et al., 1983, Quinn and Wylie, 1972, Fread, 1973, and Jotiffe, 1984). The procedure suggested by Fread (1973) can be summarized as follows:

- 1) Specify the initial conditions and the upstream boundary condition for the principle river and the tributary; specify the downstream boundary condition for the principle river.
- 2) Estimate the tributary flow  $Q_{tc}$  occurring at the confluence for the time  $t + \Delta t$ .
- 3) Solve the implicit difference equations for the principle river by using the lateral inflow  $Q_{tc}/\Delta x_c$  along the finite reach  $\Delta x_c$  (the width of the tributary). The solution obtained for the water surface elevation at the midpoint of  $\Delta x_c$  is denoted as  $h_c$ .
- 4) Solve the implicit difference equation for the tributary by using  $h_c$  as the downstream boundary condition. The solution obtained for the tributary flow is denoted as  $Q_{ts}$ .
- 5) If  $|Q_{tc} - Q_{ts}| < \varepsilon$  (predetermined error tolerance), increment the time and return to step 2; otherwise, use  $Q_{ts}$  as an improvement estimate of the tributary flow  $Q_{tc}$  and return to step 3.

The current study uses a technique that was developed based on this one to route the flow through the junctions.

### 3.3 Governing equations and solution methods used in the model

The model solves the following unsteady flow equations

$$A \frac{\partial u}{\partial x} + u \frac{\partial A}{\partial x} + \frac{\partial A}{\partial t} + q = 0 \dots\dots\dots(3.22)$$

$$A \frac{\partial u}{\partial t} + u \frac{\partial A}{\partial t} + u^2 \frac{\partial A}{\partial x} + A \frac{\partial u^2}{\partial x} + g \frac{\partial(Ay)}{\partial x} - gA(S_0 - S_f) + qu = 0 \dots\dots\dots(3.23)$$

It uses the implicit scheme to express the previous equations mathematically, using the Amein four-point scheme. For any arbitrary variable  $\alpha$  at point M, the value of  $\alpha$  and the derivatives of it with respect to the unknowns can be expressed using the variables at points a, b, c and d (see Figure 3. 6) as follows:

$$\alpha_m = \frac{(1-\theta)}{2}(\alpha_a + \alpha_b) + \frac{\theta}{2}(\alpha_c + \alpha_d) \dots\dots\dots(3.24)$$

$$\frac{\partial \alpha}{\partial x} = \alpha(\Delta x) = \frac{[(1-\theta)(\alpha_b - \alpha_a) + \theta(\alpha_d - \alpha_c)]}{\Delta x} \dots\dots\dots(3.25)$$

$$\frac{\partial \alpha}{\partial y} = \alpha(\Delta t) = \frac{[(\alpha_c + \alpha_d) - (\alpha_a + \alpha_b)]}{2\Delta t} \dots\dots\dots(3.26)$$

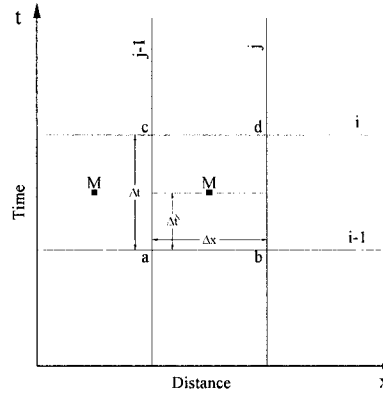


Figure 3.6  
Implicit rectangular net

Using Equations 3.24 to 3.26 with  $\theta$  equals 1.0, equations 3.22 and 3.23 can be written as follows:

$$F_1 = \frac{1}{2}[A_{j-1}^i + A_j^i] * \frac{u_j^i - u_{j-1}^i}{\Delta x} + \frac{1}{2}[u_{j-1}^i + u_j^i] * \frac{A_j^i - A_{j-1}^i}{\Delta x} + \frac{A_j^i + A_{j-1}^i - A_{j-1}^{i-1} - A_j^{i-1}}{2\Delta t} + q = 0.0 \dots\dots\dots(3.27)$$

$$F_2 = \frac{1}{2}[A_{j-1}^i + A_j^i] * \frac{u_{j-1}^i + u_j^i - u_{j-1}^{i-1} - u_j^{i-1}}{2\Delta t} + \frac{1}{2}[u_{j-1}^i + u_j^i] * \frac{A_{j-1}^i + A_j^i - A_{j-1}^{i-1} - A_j^{i-1}}{2\Delta t} + \frac{1}{2}[(u^2)_{j-1}^i + (u^2)_j^i] * \frac{A_j^i - A_{j-1}^i}{\Delta x} + \frac{1}{2}[A_{j-1}^i + A_j^i] * \frac{(u^2)_j^i - (u^2)_{j-1}^i}{\Delta x} + g * \frac{(Ay)_j^i - (Ay)_{j-1}^i}{\Delta x} - \frac{gs_0}{2} * [A_{j-1}^i + A_j^i] + \frac{g}{4}[A_{j-1}^i + A_j^i] * [s_{f_{j-1}}^i + s_{f_j}^i] + \frac{q}{2}[u_{j-1}^i + u_j^i] = 0.0 \dots\dots\dots(3.28)$$

For the flow through the junctions, the continuity and the energy equations will be used for the continuous canal, while the upstream boundary equation will be used for the branched canal as in Figure 3.7. The equations for the continuous canal are as follows:

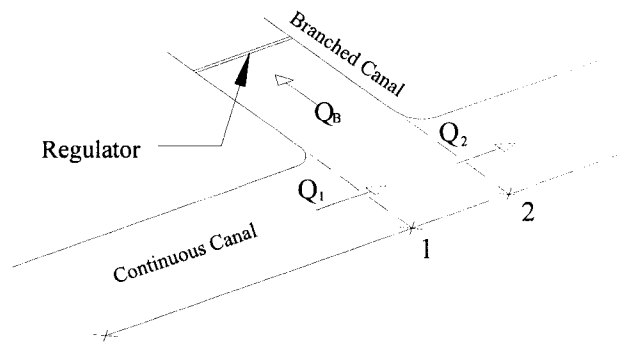


Figure 3.7  
A junction between two canals

$$F_1 = A_1 * u_1 - A_2 * u_2 - Q_B \dots\dots\dots(3.29)$$

$$F_2 = z_1 + y_1 + \frac{u_1^2}{2g} - z_2 - y_2 - \frac{u_2^2}{2g} \dots\dots\dots(3.30)$$

For the regulators, the continuity and the energy equations will be used between sections 1 and 2, as shown in Figure 3.8.

For the continuity equation, the sluice gate equation will be used as follows:

$$F_1 = A_2 * u_2 - cd * b * og * \sqrt{2g(y_1 + z_1 - y_2 - z_2)} \dots\dots\dots(3.31)$$

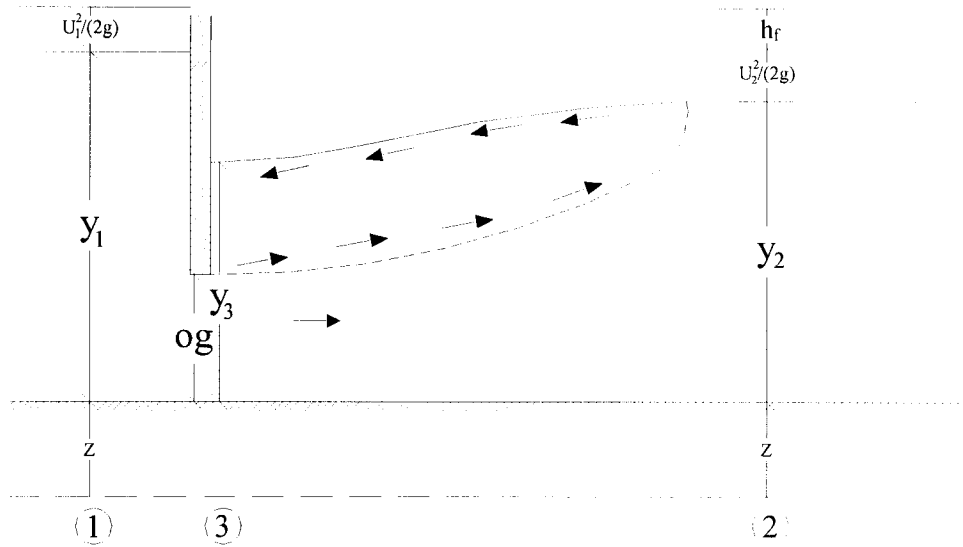


Figure 3.8  
Total energy before the gate and after the submerged jump

For the energy equation between sections 1 and 2, the head loss due to the submerged jump will be added to the equations as follows:

$$F_2 = z_1 + y_1 + \frac{u_1^2}{2g} - z_2 - y_2 - \frac{u_2^2}{2g} - \varepsilon \left[ y_1 + \frac{u_1^2}{2g} \right] \dots\dots\dots(3.32)$$

Where  $\varepsilon$  is the energy loss ratio due to the submerged jump as a function of the upstream specific energy, and is expressed as follows:

$$\varepsilon = \frac{2(Y_3 - Y_2) + F_0^2 \left\{ 1 - \left( \frac{k}{Y_2} \right)^2 \right\}}{2Y_3 + F_0^2} \dots\dots\dots(3.33)$$

For the constrictions, the continuity and energy equations will be used between sections 1 and 2 (before and after the constrictions). The head loss due to friction will be used in the energy equation. The equations are as follows:

$$F_1 = A_1 * u_1 - A_2 * u_2 \dots\dots\dots(3.34)$$

$$F_2 = z_1 + y_1 + \frac{u_1^2}{2g} - z_2 - y_2 - \frac{u_2^2}{2g} - h_f \dots\dots\dots(3.35)$$

$$h_f = L \left( \frac{Q}{K_2} \right)^2 \dots\dots\dots(3.36)$$

For the boundary conditions, a stage hydrograph will be used at the upstream of the main canal, and a discharge hydrograph will be used at the downstream end of each canal. The equations are as follows:

$$F_1 = w(t) - z_1 - y_1 \dots\dots\dots(3.37)$$

$$F_{2*N} = Q(t) - A_{2*N} * u_{2*N} \dots\dots\dots(3.38)$$

For the upstream end of the other canals (i.e., all canals except the main canal), the program will define the stage hydrograph as the average water level between sections 1 and 2 (as shown in Figure 3.7) as follows:

$$w^i = \frac{1}{2} (y_1^i + z_1 + y_2^i + z_2) \dots\dots\dots(3.39)$$

Using these equations, the model uses the following method to route the flow:

- The flow conditions in the entire network will be solved at once. Each reach of the canal between two branches will be divided into some user-specified distance intervals ( $\Delta x$ ).
- The initial conditions are specified for each point in the network.
- The unsteady flow equations will be used for the internal points. For the junctions, the regulators, the constrictions and the boundaries, the equations previously presented for each are used.



- The user will specify the boundary conditions at the upstream end of the main canal and the downstream end of each canal. The program will calculate the boundary condition at the upstream end of all canals other than the main canal.
- After defining the equations and their derivatives for all canals, the Newton-Raphson procedure will be used to calculate the residuals, and the convergence will be checked.
- After the convergence criterion is met, the results for the current time step will be used as the initial data for the next time step, and the procedure repeats.

Using the previous equations for the entire network, a unique system of 2N equations with 2N unknowns will exist, where N is the total number of points in the network. The system of equations should look those shown in Figure 3.9. The Jacobian matrix, which is required by Newton-Raphson method, will be 2N\*2N as in Figure 3.10.

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ F_{2 \times N} \end{bmatrix} = \begin{bmatrix} \left\{ \begin{array}{l} y(t) - y = 0 \\ A_m u(\Delta x) + \dots = 0 \\ A_m u(\Delta t) + \dots = 0 \end{array} \right\} \dots Canal I \\ \vdots \\ \left\{ \begin{array}{l} Q(t) - Q = 0 \end{array} \right\} \\ \vdots \\ \left\{ \begin{array}{l} y(t) - y = 0 \\ A_m u(\Delta x) + \dots = 0 \end{array} \right\} \dots Canal I + 1 \\ \vdots \\ \vdots \end{bmatrix}$$

Figure 3.9  
System of equations

Figure 3.10  
Jacobian Matrix

The Jacobian matrix is a banded matrix that has a maximum of 4 columns for each row. The factorization, forward and backward procedure will be used to solve the system of linear equations. The Jacobian matrix will be saved as 2N\*4 instead of 2N\*2N, and data will be overwritten after being factorized. To take advantage of the sparsity of the matrix, the factorization, forward and backward procedures will be implemented as follows:

For the factorization procedure:

- The original matrix J will be factorized to lower matrix L and upper matrix U.
- The first row for each canal will not be computed.
- Only the first two items of each second row will be computed using the following equations:

$$L_{2,1} = \frac{J_{2,1}}{U_{1,1}} \dots \& U_{2,2} = J_{2,2} - L_{2,1} * U_{1,2} \dots \dots \dots (3.40)$$

- For the third row, only the first three items will be computed using the following equations:

$$L_{3,1} = \frac{J_{3,1}}{U_{1,1}} \dots \& L_{3,2} = \frac{J_{3,2} - L_{3,1} * U_{1,2}}{U_{2,2}} \dots \& U_{3,3} = J_{3,3} - L_{3,2} * U_{2,3} \dots (3.41)$$

- For the remaining rows, until the last row of each canal, the procedure is as follows:

⌘ For the lower matrix, only one item will be modified in the even rows as follows:

$$L_{J,1} = \frac{J_{J,1}}{U_{(j-1),3}} \dots \dots \dots (3.42)$$

and only two items will be modified in odd rows as follows:

$$L_{J,1} = \frac{J_{J,1}}{U_{(j-2),3}} \dots \& L_{J,2} = \frac{J_{J,2} - L_{J,1} * U_{(j-2),4}}{U_{(j-1),2}} \dots \dots \dots (3.43)$$

⌘ For the upper matrix, only one item will be modified in the odd rows as follows:

$$U_{J,3} = J_{J,3} - L_{J,2} * U_{(j-1),3} \dots \dots \dots (3.44)$$

and only two items will be modified in even rows, as follows:

$$U_{(j-1),4} = J_{(j-1),4} - L_{(j-1),2} * U_{(j-2),4} \dots \dots \dots (3.45)$$

$$U_{J,2} = J_{J,2} - L_{J,1} * U_{(j-1),4} \dots \dots \dots (3.46)$$

- For the last row, both non-zero values will be computed as follows:

$$L_{J,3} = \frac{J_{J,3}}{U_{(j-1),3}} \dots \& U_{J,4} = J_{J,4} - L_{J,3} * U_{(j-1),4} \dots \dots \dots (3.47)$$

For the forward procedure:

- The first value for each canal can be calculated directly using the following equation:

$$y(1) = -F(1) \dots\dots\dots(3.48)$$

- The values from the second one to the one before the last will be computed, each two sequential rows together (unrolled=2), as follows:

$$y(i) = -F(i) - y(i-1) * L_{i,1} \dots\dots\dots(3.49)$$

$$y(i+1) = -F(i+1) - y(i-1) * L_{(i+1),1} - y(i) * L_{(i+1),2} \dots\dots\dots(3.50)$$

- The last value will be computed as follows:

$$y_{2*N} = -F_{2*N} - y_{(2*N-1)} * L_{2*N,3} \dots\dots\dots(3.51)$$

For the backward procedure:

- The last value of each canal can be calculated directly using the following equation:

$$\Delta x_{2*N} = \frac{y_{2*N}}{U_{2*N,4}} \dots\dots\dots(3.52)$$

- The values from the one before the last to the second one will be computed, each two sequential rows together (unrolled=2), as follows:

$$\Delta x(i) = \frac{y(i) - \Delta x(i+1) * U_{i,4}}{U_{i,3}} \dots\dots\dots(3.53)$$

$$\Delta x(i-1) = \frac{y(i-1) - \Delta x(i+1) * U_{(i-1),4} - \Delta x(i) * U_{(i-1),4}}{U_{(i-1),2}} \dots\dots\dots(3.54)$$

- The first value will be computed as follows:

$$\Delta x(1) = \frac{y(1) - \Delta x(2) * U_{1,2}}{U_{1,1}} \dots\dots\dots(3.55)$$

### 3.4. Methods used to improve the robustness of the model

The robustness of the unsteady flow model is an important issue. There are two issues that affect the robustness: zero or negative water depth and failure to meet the convergence criteria. To address the occurrence of zero or negative water depths, a new technique is suggested as follows:

- A) For each time step, the program must guess the initial solution for the advanced water levels and discharges. The program uses the data of the previous time step (or initial data in the first time step) for this guess.
- B) The program will run through iterations until convergence. During this running, the data of the previous time step is saved.
- C) Whenever the program finds a zero or negative water depth for the next time step, the following is done:
  - ✍ The program will assume the point of this zero or negative water depth as an artificial end for this canal, and it will ignore all the areas behind it.
  - ✍ It will automatically redefine the number of reaches, the number of structures (regulators and bridges), and the number of distance intervals in the last reach for this canal.
  - ✍ All branches behind this point will have complete water shortage and will be ignored from the routing.
  - ✍ If the point with zero or negative water depth is the first or second point in the channel, the whole channel will be ignored from the routing.
  - ✍ The downstream boundary condition (discharge boundary condition in this case) will be redefined, so it will contain the lateral outflow that was used in the ignored parts.
  - ✍ The initial guess, which remains unchanged, will be assigned for the associated points in the network.
  - ✍ The iteration counter will be reset to 1.
  - ✍ The program will return to step B.
- D) For each time step, and if there is any water shortage in the network, the program checks if the flow should return back to the water shortage areas. If the flow

should return back to the water shortage areas, the program will add a new distance interval with one or two joints based on the current end of the channel, and assume the initial conditions at these joints. These initial conditions are an approximated guess that should become actual value with the convergence of the next time step. The program add the joints and assume the initial conditions on the following basis:

✍ If the flow at the current end of the canal (Joint J in figure 3.11) satisfies conditions in equations 3.56 and 3.57, the program will add a new distance interval with one or two joints based on the location of the current end.

$$w_J > BL_{J+1} + \Delta x_J * s_J \dots \dots \dots (3.56)$$

$$Q_J > \Delta x_J * q_J \dots \dots \dots (3.57)$$

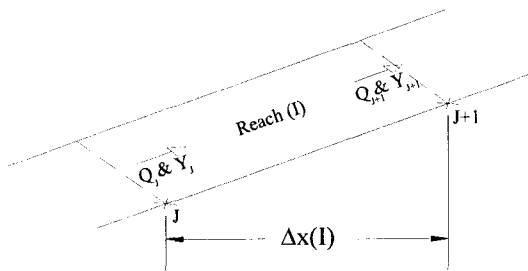


Figure 3.11  
Current end of the canal is not an end of a reach

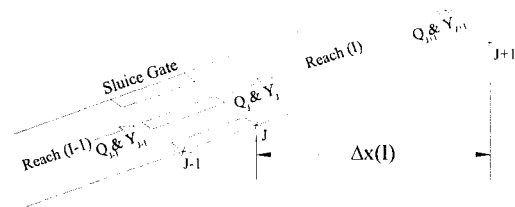


Figure 3.12  
Current end of the canal is an end of a reach followed by constriction

E) If the current end of the canal is not an end of a reach (Figure 3.11), the program will add one joint and define the initial condition for it using equations 3.58 and 3.59.

$$w_{J+1} = w_J - \Delta x_J * s_J \dots \dots \dots (3.58)$$

$$v_{J+1} = (Q_J - \Delta x_J * q_J) / A_{J+1} \dots \dots \dots (3.59)$$

- ✍ If the current end of the channel is the end of a reach, and there is a regulator or constriction following this reach (Figure 3.12), the program will add two joints (J) and (J+1) and define the initial condition for them. The data in (J) is equivalent to the data at (J-1), and the data at J+1 is calculated using equations 3.58 and 3.59.
- ✍ If the current end of the channel is the end of a reach, and there is a branch following the reach (Figure 3.13), equations 3.57 to 3.59 are used to check and define the initial conditions for joints J and J+1 as in the previous point. Then equations 3.60 and 3.61 are used to check if the water should enter the branch (Figure 3.14). If both conditions are satisfied, equations 3.62 to 3.65 are used to define the initial conditions for the first two joints in the branch

$$w_J > BL_{BJ} + \Delta x_B * s_B \dots\dots\dots(3.60)$$

$$Q_J > \Delta x_I * q_I + \Delta x_B * q_B \dots\dots\dots(3.61)$$

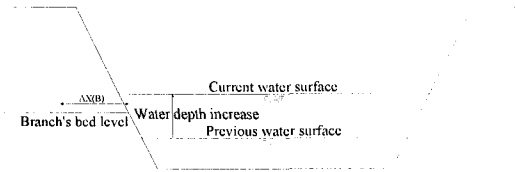
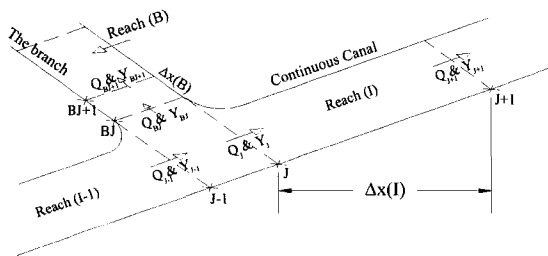


Figure 3.13  
Current end of the canal is the end of a reach followed by branch

Figure 3.14  
When the water level in the main canal increases to enter the branch

$$w_{BJ} = w_J \dots\dots\dots(3.62)$$

$$w_{BJ+1} = w_{BJ} - \Delta x_B * s_B \dots\dots\dots(3.63)$$

$$v_{BJ} = (A_{BJ} * v_i - \Delta x_B * q_B) / A_{BJ} \dots\dots\dots(3.64)$$

$$v_{BJ+1} = v_i \dots\dots\dots(3.65)$$

- ✍ The same procedure that was used to check the flow and add joints for branches will be used in the case where the water is already found in the main canal but it was not enough to enter the branch (Figure 3.14), and then it becomes enough to enter it.
- ✍ In all previous cases, and when a new distance interval is added to a canal, the downstream boundary condition of this channel will be redefined by decreasing it by an amount equal to the lateral outflow of the added distance interval. When there is no water shortage areas in the channel, the boundary value should be returned to its original value.

To address the problem of non-convergence, whenever the program reaches the maximum number of iterations without convergence, it does the following:

- The program multiplies the convergence criteria by a specified factor greater than 1.0, and it will give an error message.
- It will reassign the initial guess for all points in the network.
- The iteration counter will be reset to 1, and the procedure continues.

### 3.5 References

- Abbott, M. B., "Method of Characteristics," *Unsteady Flow in Open Channel*, Mahmoud K., and Yevjevich V., ed., Vol. 1, Water Resources Publications, Fort Collins, Co., 1975, pp. 63-88.
- Abbott, M. B., and Basco, D. R., *Computational Fluid Dynamics*, John Wiley, New York, 1989.
- Amein M., and Chu H. L., "Implicit Natural Modeling in Unsteady Flows," *Journal of Hydraulic Division*, American Society of Civil Engineers, Vol. 101, HY6, June 1975, pp. 717-731.
- Amein M., and Fang, C. S., "Implicit Flood Routing in Natural Channels," *Journal of Hydraulic Division*, American Society of Civil Engineers, Vol. 96, HY12, December 1970, pp. 2481-1500.

- Barkau, Robert L., *UNET, One-Dimensional Unsteady Flow Through a Full Network of Open Channels, Computer Program, St. Louis, MO, 1992.*
- Chow V. T., *Open Channel Hydraulics*, McGraw- Hill Book Company, Inc, New York, 1959
- Clemmens, A. J., Holly, F. M., and Schuurmmans, W. (1991). "Description and evaluation of Program: DUFLOW," *Proceedings of Irrigation and Drainage National Conference*, pp. 418-424.
- Dronkers J. J., *Tidal Computations in Rivers and Costal Waters*, North-Holland, Amsterdam, Netherlands, 1964.
- El Gamel, T. T. (2001). Unsteady Flow Model For an Irrigation Canal Network, Master's Thesis, under the supervision of Laura Harrell, Old Dominion University, Norfolk, VA, May 2001.
- Fread D. L., "Discussion of Implicit Flood Routing in Natural Channels by M. Amein and C. S. Fang," *Journal of Hydraulic Division*, American Society of Civil Engineers, Vol. 97, HY7, July 1971, pp. 1156-1159.
- Fread, D. L., "Technique for Implicit Dynamic Routing in Rivers With Tributaries," *Water Resources Research*, Vol. 9, No. 4, August 1973, pp. 918-926.
- Fread, D. L., "Theoretical Development of the Implicit Dynamic Routing Model." National Weather Service, Office of Hydrology, Silver Spring, Maryland, 1978.
- Fread, D. L., and Smith, G. F., "Calibration Technique for 1-D Unsteady Flow Models," *Journal of Hydraulic Division*, American Society of Civil Engineers, Vol. 104, HY7 July 1978, pp. 1027-1044.
- Fread, D. L., "DAMBRK: The NWS Dam Break Flood Forecasting Model," National Weather Service, Office of Hydrology, Silver Spring, Maryland, 1980.
- Gooch, R. S., and Keith, J. (1991). "Description and evaluation of Program: SNUSM," *Proceedings of Irrigation and Drainage National Conference*, pp. 397-406.



- GovindaRao, N. S., and Rajaratnam, N., "The Submerged Hydraulic Jump," *Journal of Hydraulic Division*, American Society of Civil Engineers Vol. 89, HY1, January 1963, pp. 139-162.
- Holly, F. M., and Parrish, J. B. (1991). "Description and evaluation of Program: CAIMA," *Proceedings of Irrigation and Drainage National Conference*, pp. 432-437.
- Johnson, B. H., "Development of Numerical Modeling Capabilities the Computation of Unsteady Flow on the Ohio River and Its Major Tributaries." Technical Report HL-82, U.S. Army, Corps of Engineers, Waterways Experiment Station, Vicksburg, Mississippi, 1982.
- Jotiffe, I. B., "Computation of Dynamic Waves in Channel Networks," *Journal of Hydraulic Division*, American Society of Civil Engineers, Vol. 110, 1984, pp. 1358-1369.
- Li, Z. C., Zhan, L. J., and Wang, H. L., "Difference Methods of Flow in Branch Channel," *Journal of Hydraulic Division*, American Society of Civil Engineers, Vol. 109, YH3, March 1983, pp. 424-446.
- Liggett, J.A., and Cunge, J.A., "Numerical Methods of Solution of the Unsteady Flow Equations," *Unsteady Flow in Open Channels*, Mahmoud K., and Yerjevich V., ed., Vol. 1, Water Resources Publications, Fort Collins, Co., 1975, pp. 89-182.
- Merkley, G. P., and Rogers, D. C. (1991). "Description and evaluation of Program: CANAL," *Proceedings of Irrigation and Drainage National Conference*, pp. 391-396.
- Ohtsu, I., Yasuda, Y., and Ishikawa, M., "Submerged Hydraulic Jumps below Abrupt Expansions," *Journal of Hydraulic Engineering*, May 1999, pp. 492-499.
- Quinn, F. H., and Wylie, E. B., "Transient Analysis of the Detroit River by the Implicit Method," *Water Resources Research*, Vol. 8, No. 6, Dec. 1972, pp. 1461-1469.

- Rogers, D. C., and Merkley, G. P. (1991). "Description and evaluation of Program: USM," *Proceedings of Irrigation and Drainage National Conference*, pp. 425-431.
- Rajaratnam, N., and Subramanya, K., "Flow Equation For the Sluice Gate," *Journal of Irrigation and Drainage Division*, American Society of Civil Engineers, Vol. 93, IR3, 1967, pp. 167-186.
- Schaffranek, R. A., et al. "A Model for the Simulation of Flow in Singular and an Interconnected Network of Cannels." *Techniques of Water Resources Investigation*, Chapter 3, U.S. Geological Survey, Reston, Virginia, 1981.
- Schuurmans, W. (1991). "Description and evaluation of Program: MODIS," *Proceedings of Irrigation and Drainage National Conference*, pp. 408-417.
- Stoker, J. J., *Water Waves, The Mathematical Theory with Applications*, Interscience Publisher Inc, New York, 1957.

## CHAPTER 4

### GENETIC ALGORITHM PARAMETERS

#### 4.1 Introduction

GAs require the user to define the parameters used during the GA process such as population size, crossover probability, and mutation rate. These parameters adversely affect the performance of GA if not chosen suitably. However, finding a good value for each parameter is a difficult task because of the following:

- GA parameters interact with each other in a complex way, and a complete analysis of their interactions is difficult to achieve.
- Suitable parameters depend on the class of problems to be optimized. For instance the noise in the function might require a larger population size (Goldberg et al., 1992; Deb, 2001). Also the mutation-based approach and crossover-based approach are suitable for different classes of problems based on the difficulty (Deb and Agrawal, 1999).
- The parameters must be chosen such that there is a balance between the exploitation caused by the selection operator, and the exploration caused by recombination and mutation operators. Otherwise, the GA may converge to local optima or behave as a random search process.

According to Hart and Belew (1991), “GA parameters interact in complex ways, making the task of finding a suitable parameter scenario not always straightforward. In addition, a GA, which excels with a given class of problems, might yield poor results when applied to another class.”

The study of GA parameters began in 1975 with the work of De Jong. He constructed a test environment of five functions that present difficulty to gradient techniques, and he used two different measures: online performance (measures of the convergence), which is the average performance of all tested structures over the course of the search, and offline performance (measures the ongoing performance), which is the best performance achieved in the time interval. De Jong studied the

effect of population size ( $N$ ), crossover probability ( $p_c$ ), and mutation probability ( $p_m$ ), in addition to other parameters. De Jong made the following recommendations:

- Increasing the population size was shown to reduce the stochastic effect, and improve the long-term performance at the expense of slower initial response.
- Increasing Mutation rate was seen to improve offline performance at the expense of online performance.
- Reducing the crossover rate resulted in an overall improvement in the performance.

The De Jong equations have been revisited several times by other researchers. Grefenstette (1986) restudied the De Jong equations with a meta-GA. This meta-GA was used to locate the parameter scenarios which themselves were used for the GA search. He used the De Jong equations with the following ranges:

- 16 different population sizes from 10 to 160 with increment of 10.
- 16 different crossover rates from 0.25 to 1.0 with increment of 0.05.
- 8 different mutation rates from 0.0 to 1.0.

He conducted two experiments for online and offline performance. Then he validated his results by testing them against a standard GA (with parameter values suggested by De Jong). During both the experiments and the validation, his suggestions outperformed De Jong's parameters. However, the difference was statistically significant in only online performance. Grefenstette was aware that this work has limitation as some recombination operators were ignored, and the tested problems are unconstrained problems.

Goldberg (1985,1989) performed theoretical studies about the optimal population size in binary encoding. He derived an expression for optimal population size based on the number of new schemata per population number.

Schaffer et al. (1989) restudied the De Jong functions with 5 other test functions to include a wider range of search characteristics. They used gray encoding instead of binary encoding, and they used the following ranges for GA parameters:

- 6 different population sizes (10,20,30,50,100,200).
- 10 different crossover rates from 0.05 to 0.95 with increment of 0.10.
- 7 different mutation rates (0.001,0.002,0.005,0.01,0.02,0.05,0.10).

They were concerned only about online performance, arguing that offline performance is surely quite different. They found that mutation rate has more effect than was indicated by previous works. They stated that “naive evolution (NE) (a GA using only selection and mutation) does perform hillclimb-like search and given the range of strategies that can be achieved by varying population size and mutation rates, it is likely to be a powerful search algorithm, even without the assistance of crossover.” They also stated that criterion used by Goldberg (1985), (for the optimal population size) was too conservative, leading him to recommend unnecessarily large populations, based on the argument that “a large population size can achieve a large sampling of the space (exploration) at least in the initial generation. However, a large population imposes a large cost per generation, and the exploration for schemata not presented in the initial population can be achieved by the operators.”

Deb and Agrawal (1999) studied the interactions between different GA parameters (crossover probability, mutation rate, and population size) for five different functions representing different levels of difficulty. They solved the functions using a mutation-based approach, crossover-based approach, and both operators (crossover and mutation) approach. They concluded with the following points:

- ❑ For unimodal and simple functions, the mutation-based approach has performed better than the crossover-based approach.
- ❑ With a fixed number of function evaluations, a mutation-based GA performs best with moderate population size.
- ❑ When GAs are applied to more complex problems, mutation-based approach fails miserably to solve these functions, while the crossover-based approach is able to solve these problems.
- ❑ GAs with both crossover and mutation have performed better than only crossover or mutation-based GAs in simpler problems.

Besides defining fixed values or theoretical equations for GA parameters, many researchers attempt to adapt the parameters during the run, either through an adaptive or self-adaptive process. Hinterding et al. (1996) attempted to adapt the population size by using different sub-populations, adjusting their size at regular intervals based

on the results. Arabas et al. (1994) attempted to use the concept of age, which is the number of generations the chromosomes stay alive, to influence the size of the population at each stage of the process.

For mutation rate, Fogarty (1989) used varying mutation rate, either increasing or decreasing, and he concluded that varying the probability of mutation significantly improved the performance of GA if the problem started by a conservative initial population, but not in a randomly generated initial population. Janikow and Michalewicz (1991) presented a non-uniform mutation, where the value at time  $(t+1)$  is shifted from the previous value by  $\Delta(t, y)$ . This  $\Delta(t, y)$  returns a value between 0 and  $y$ , and it is closer to zero as the generation number  $(t)$  increases. Thus, the model searches globally space in the first stages, but very locally in the last stages.

Most of the attempts to adapt crossover probability were by the means of using different sub-populations, each of which has a different crossover probability, and different mutation rates as well in some procedures. Through the process of the GA, the subpopulations exchange their values, and shift towards the most successful population. Some details about these attempts are given in Eiben et al. (1999).

Considering the previous studies described above, studying the optimal GA parameters that should be used within the current model is an important issue as most of the previous studied were done using binary GA operators, using binary or gray encoding, with one point or two point crossover, while the current model uses real GA encoding and parameters. Also, most of the previous works used explicit equations and unconstrained problems to test these parameters, while the current model uses a simulation model to evaluate solutions, so this section is intended to test the recommended GA parameters within the current model.

The following issues control the range to be tested for each parameter:

- The cumulative works in GA parameters gave evidence about an expected range for each parameter, although there is no fixed number. An example of this is what was stated by Eiben et al. (1999) about crossover probability: “Currently, it is commonly accepted that the crossover rate should not be too low and values below 0.6 are rarely used.” Thus the current study will just go slightly outside this

range, and the crossover probability is tested from 0.5 to 0.9. The same type of procedure was used to define the range for other parameters.

- The computational time requirement is very large for the current model, which makes it very hard to test big ranges of all parameters.

Considering the interaction between the parameters, and the factors that can affect them, the goal of this chapter is to provide a guidelines for the user of the model about the recommended ranges for the parameters and the effect of decreasing or increasing them, rather than giving a fixed values that must be used.

## 4.2 Population size (N)

Selecting a suitable population size is an important decision that affects the GA performance. Based on Grefenstette (1986), “GAs normally do poorly with very small populations because the population provides an insufficient sample size for most hyper-planes. A large population discourages premature convergence to sub-optimal solutions. On the other hand, a large population requires more evaluations per generation, possibly resulting in an unacceptably slow rate of convergence.” However, the results don’t always support that idea that the larger population size will always converge to better optimal point. Based on Syswerda (1991) “General wisdom dictates that a larger population will work more slowly but will eventually achieve better solutions than a smaller population. Experience indicates, however, that this rule of thumb is not always true, and that the most effective population size depends on the problem being solved, the representation used, and the operators manipulating the representation.” Also, according to Deb and Agrawal (1999), “when GAs are applied to simpler problems, an interesting feature of mutation-based GAs is observed. There seems to be two distinct ranges of population sizes (with a dip in performance in intermediate population sizes), where these GAs work the best.”

Some of the suggestions made for population size in the literature are:

- $N = 50$  to 100 (De Jong, 1975)
- $N = 30$  (Grefenstette, 1986)
- $N = 20$  to 30 (Schaffer et al., 1989)

In the current study, four different population sizes are tested (26, 50, 76, and 100).

### 4.3 Crossover probability ( $p_c$ )

Crossover probability defines the ratio of the population that will exchange its data during the recombination process to produce new strings (children). The rest of the population will pass as they are to the next generation. Based on Grefenstette (1986), “If the crossover rate is too high, high-performance structures are discarded faster than selection can produce improvements. If the crossover probability is too low, the search may stagnate due to the low exploration rate.”

Some of the suggestions made for crossover probability in the literature are:

- $p_c = 0.60$  (De Jong, 1975)
- $p_c = 0.95$  (Grefenstette, 1986)
- $p_c = 0.75$  to  $0.95$  (Schaffer et al., 1989)

In the current study, five different crossover probabilities are tested (0.5 to 0.9 with increment of 0.1).

### 4.4 Mutation rate ( $p_m$ )

Based on Grefenstette (1986), “A low level of mutation serves to prevent any given bit position from remaining converged to a single value in the entire population. A high level of mutation yields an essentially random search.”

Some of the suggestions made for mutation rate in the literature are:

- $p_m = 0.001$  (De Jong, 1975)
- $p_m = 0.01$  (Grefenstette, 1986)
- $p_m = 0.005$  to  $0.01$  (Schaffer et al., 1989)
- $p_m = \frac{1}{L}$  as ( $L$ ) is the bit-string length. (Introduced by Muhlenbien, (Eiben et al., 1999)).



In the current study, three different mutation rates are tested (0.001, 0.01 and 0.05).

#### 4.5 Blend crossover extension ( $\alpha$ )

Blend crossover extension ( $\alpha$ ) defines the range of real number variables that will be used to randomly select the children in the recombination process. Higher values of blend crossover extension have a better chance to explore the search space with the risk of discarding good values that were already found; however, smaller values have a better chance for convergence. According to Deb (2001), “BLX- $\alpha$  has an interesting property: if the difference between the parent solutions is small, the difference between the offspring and parent solutions is also small.” This also brings the point that if the difference between the parent solutions is small, the effect of  $\alpha$  decreases, and for small  $\alpha$  it may be negligible. Vice versa, if the difference is high, the higher values of  $\alpha$  may affect the convergence. This may require an adaptive process for choosing  $\alpha$ , which may be changeable based on the difference between parent solutions or through generations. However, in the current study, we will limit ourselves to the fixed values of  $\alpha$ . As was stated by Deb (2001),  $\alpha = 0.5$  is the best-suggested value for blend crossover extension. In the current study, this value (0.5) will be compared with smaller values (0, 0.1, and 0.25)

#### 4.6 Analysis

The GA parameters are tested as follows:

- First, all combinations of crossover probabilities, mutation rates, and blend crossover extension are tested for each scenario of the case study.
- The population size will be tested with different crossover probabilities (as it is the one that has the higher range), with fixed values for mutation rate and blend crossover extension.
- The multiobjective technique, proposed by Coello (2000) (described in detail in the next chapter) is used as a constraint handling technique.

For each run, two different values will be measured:

- Best feasible solution found during the whole run (measure 1).
- The improvement of the minimum feasible solution during the run (measure 2).

The objective of the second measure is to check if the technique will stop at a local optima or if it will keep improving to the end. This measure is defined using the distance from the optimal feasible value, and it is calculated using the following procedure:

- The best feasible obtained from all runs in the scenario is noted, and it is considered as the global optimum.
- For each run, generations are divided into sub-generations.
- For each sub-generation, the minimum feasible solution is defined.
- The distance from the optimal value is calculated as follows:

$$DFO(I) = \sum_{J=1}^{SG} J * \frac{(SG\_M - G\_M)}{G\_M} \dots\dots\dots(4.1)$$

Where:

DFO        distance from optimal value.

SG         number of sub-generations.

SG\_M      minimum value achieved in the sub-generation.

G\_M        global optimal (best value achieved in the scenario).

According to the previous equation, the inability to get closer to the global minimum in the later sub-generation is worse than the inability to get closer to it in the early sub-generations.

There are also 4 different tests that will be performed to define the best value for each parameter:

- Test 1, (Mean values): The difference between means of the runs that are related to each parameter is tested. First multiple means comparison will be used. If the difference between the means is not confirmed statistically, means will be drawn to explore which parameter performs better. Multiple means comparison refers to

making several tests for statistical significance between means within a group of means. The null hypothesis that is tested is:

$$H_0 : \mu_1 = \mu_2 = \dots = \mu_K$$

The alternative hypothesis is

$$H_0 : \text{not all means are equal}$$

Rejecting the null hypothesis means that there is significant difference between the means. The statistical technique used in this case is called single-factor ANOVA or F-test. MCM can be categorized into single-step or stepwise procedure. Stepwise procedure makes comparisons on a series of steps, where the result of the current step influences which, if any, comparisons are made in the next step. They can be divided into step-down, and step-up. Duncan multiply range tests are an example of stepwise/step-down procedure, and it is used in the current study with confidence level 90% (ALPHA=0.1). This test will determine if the difference between the means of the different values of each parameter reflects a true difference between the means or if it is a random effect. Besides using multiple means comparison, the means of all parameter values are presented in different charts.

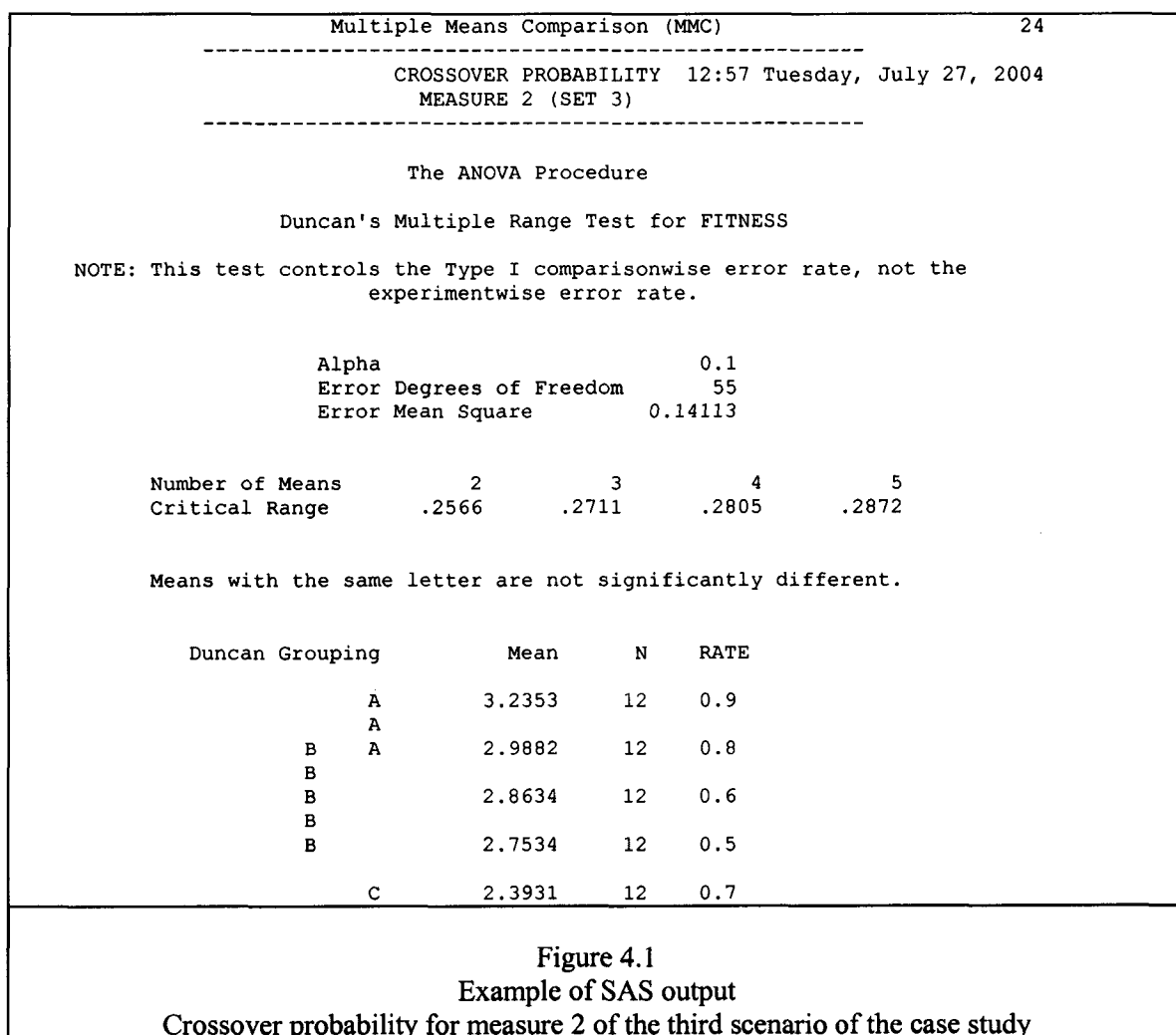
- Test 2, parameter interaction charts: these charts are drawn between crossover probability and mutation rate for different measures and different scenarios of the case study. Regarding the population size, a parameter interaction chart between the population size and the crossover probability will be drawn. Some examples from these parameter interaction charts are presented to explore which parameter dominates the parameter interaction chart.
- Test 3, comparing similar runs: comparisons between runs that share the same GA parameters (more than the one in that test) are made, and the number of times each parameter wins is recorded. For example, with crossover probability, from the runs that have the same mutation rate, and the same blend crossover extension, the crossover probability that gives the best solution is the winner.
- Test 4, parameters of the best solution: the parameter values that resulted in the best solution for each measure in each scenario of the case study are recorded and presented.

## 4.7 Results

The results of the different tests for different scenarios of the case study are presented here. A discussion about each test is given after presenting the results. Section 4.8 presents a summary of all results with a suggestion about the best parameter values that for this model. The parameters that perform the best for each scenario will be validated by testing them against other parameters with different initial seed values.

### 4.7.1 Test 1 (Means Values)

Figure 4.1 presents an example of SAS program (for the crossover probability second measure of the third case study), and Table 4.1(a-d) presents a summary of the output of SAS program.



In Figure (4.1), crossover probability parameters are divided to three groups based on the mean value of different runs. The maximum (worst) mean is related to group A, which includes  $p_c = 0.9$  and  $p_c = 0.8$ . Group B includes  $p_c = 0.8$ ,  $p_c = 0.6$  and  $p_c = 0.5$ . The best mean value is related to group C, which has  $p_c = 0.7$  and it is the best value for crossover probability for this measure. There is an overlap between the first two groups, meaning that the difference is not significant. Also, from Figure (4.1), Alpha=0.1, which means that these results are obtained with 90% confidence level.

Tables 4.1(a) to 4.1(d) present summary of MMC output for different parameters.

Table 4.1-a Summary of MMC output for crossover probability

Scenarios	Results			
	Measure 1		Measure 2	
	Pr > F	Best Value	Pr > F	Best Value
Scenario 1	0.2900	--	0.01508	--*
Scenario 2	0.4766	--	0.3787	--*
Scenario 3	0.0008	0.7	0.0001	0.7

Table 4.1-b Summary of MMC output for mutation rate

Scenarios	Results			
	Measure 1		Measure 2	
	Pr > F	Best Value	Pr > F	Best Value
Scenario 1	0.0367	0.01 & 0.05	0.1462	--*
Scenario 2	0.0002	0.01 & 0.05	0.0003	0.01 & 0.05
Scenario 3	0.2286	--	0.6264	--

Table 4.1-c Summary of MMC output for blend crossover extension

Scenarios	Results			
	Measure 1		Measure 2	
	Pr > F	Best Value	Pr > F	Best Value
Scenario 1	0.8676	--	0.8266	--
Scenario 2	0.1154	--	0.0378	0.5
Scenario 3	0.2328	--	0.3053	--

Table 4.1-d Summary of MMC output for population size

Scenarios	Results			
	Measure 1		Measure 2	
	Pr > F	Best Value	Pr > F	Best Value
Scenario 1	0.1898	--*	0.0656	--*
Scenario 2	0.3882	--	0.4442	--
Scenario 3	0.0720	--*	0.0817	--*

Remarks about the previous table:

- Pr > F Check the validity of the data. The data is valid if this value < 0.05.  
 -- All means are in one group (there is no significant difference between the means)  
 --\* Means are divided to many groups, but there is an overlap between the groups. (Still there is no significant difference between the means)

From Table (4.1), the following points could be noticed:

- Regarding crossover probability, it is highly likely that  $p_c = 0.7$  is the best value for the third scenario. It is statistically confirmed in both measures. There are no statistically confirmed values for the first two scenarios.
- For mutation rate, higher mutation rates ( $p_m = 0.01$  and  $p_m = 0.05$ ) perform better for the first two scenarios. There are no statistically confirmed values for the third scenario of the case study.
- Recalling that Deb and Agrawal (1999) stated that in simpler problems, a mutation-based approach performs better, while in complex problems, a crossover-based approach performs better, a similar observation might be made here that in complex problems, there is only an evidence about the best value of the crossover probability, and in simpler problems, there is only an evidence about the best value of the mutation rate.
- For blend crossover extension,  $\alpha = 0.5$  is the best value for the second measure of the second scenario. There are no other statistically confirmed values.
- Regarding population size, no value is confirmed statistically.

Given that the MMC didn't confirm a winner for many cases, the difference between the means of measure 1 and measure 2 for runs using different GA parameter values is presented in Figures 4.2 to 4.5. The best mean value is considered as a reference and it has zero value, while the difference between other means values and this best mean value is considered.

- Regarding crossover probabilities (Figure 4.2),  $p_c = 0.7$  is the best value for both measures of the third scenario, with a clear difference than other values. For the second scenario,  $p_c = 0.6$  is the best value in both measures. For the first scenario,  $p_c = 0.8$  is the best value in both measures with very small difference than  $p_c = 0.6$  in the second measure. In general, the difference between values in the third scenario is higher than the differences in the first two scenarios.
- Regarding mutations rates (Figure 4.3), higher values ( $p_m = 0.01$  and  $p_m = 0.05$ ) are the best values for all scenarios.  $p_m = 0.05$  is the best value for both measures of the third scenario.  $p_m = 0.01$  is the best value for both measures of the second scenario. For the first scenario, the best value is different between both measures.
- Regarding blend crossover extension (Figure 4.4),  $\alpha = 0.5$  is the best value for both measures of second and third scenarios, with clear difference from the other values. For the first scenario,  $\alpha = 0.0$  is the best value, with very small difference from  $\alpha = 0.5$  in the first measure and slightly big difference from  $\alpha = 0.5$  in the second measure.
- Regarding the population sizes (Figure 4.5),  $N=50$  is the best value for both measures of the third scenario.  $N=26$  is the best value for the first measure of the second scenario, with very small difference than  $N=76$ .  $N=76$  is the best value of the second measure of the second scenario. For the first scenario,  $N=100$  is the best value for the first measure, and  $N=76$  is the best value of the second measure with small difference than  $N=100$ .



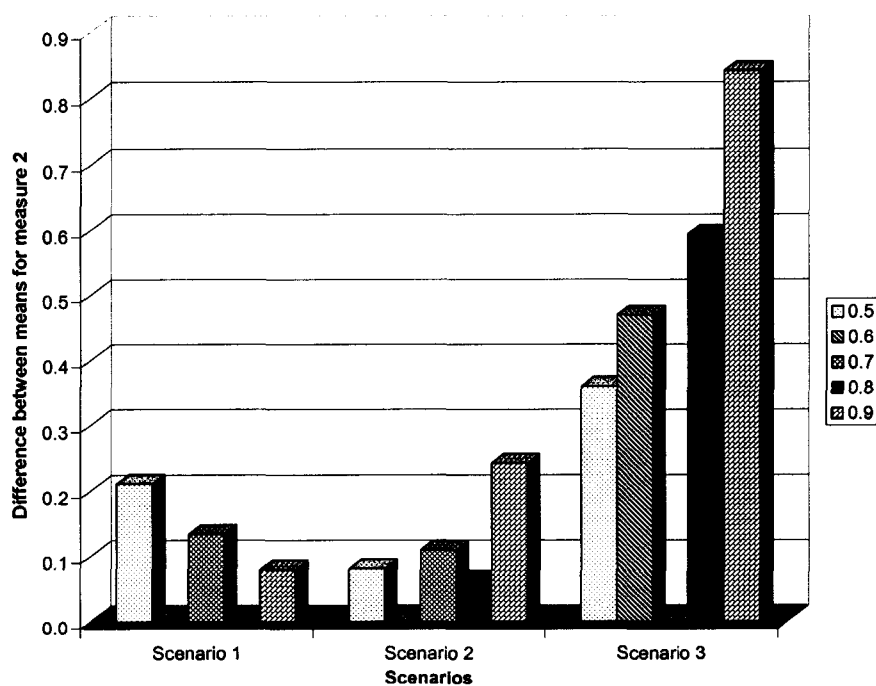
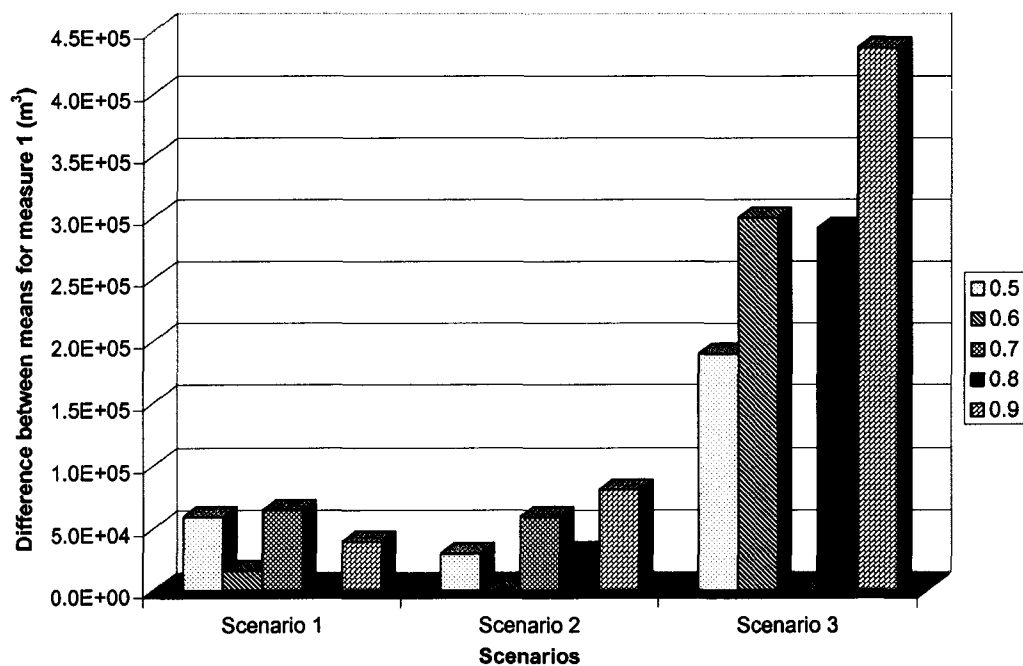


Figure 4.2  
Difference between means of different crossover probabilities for both measures

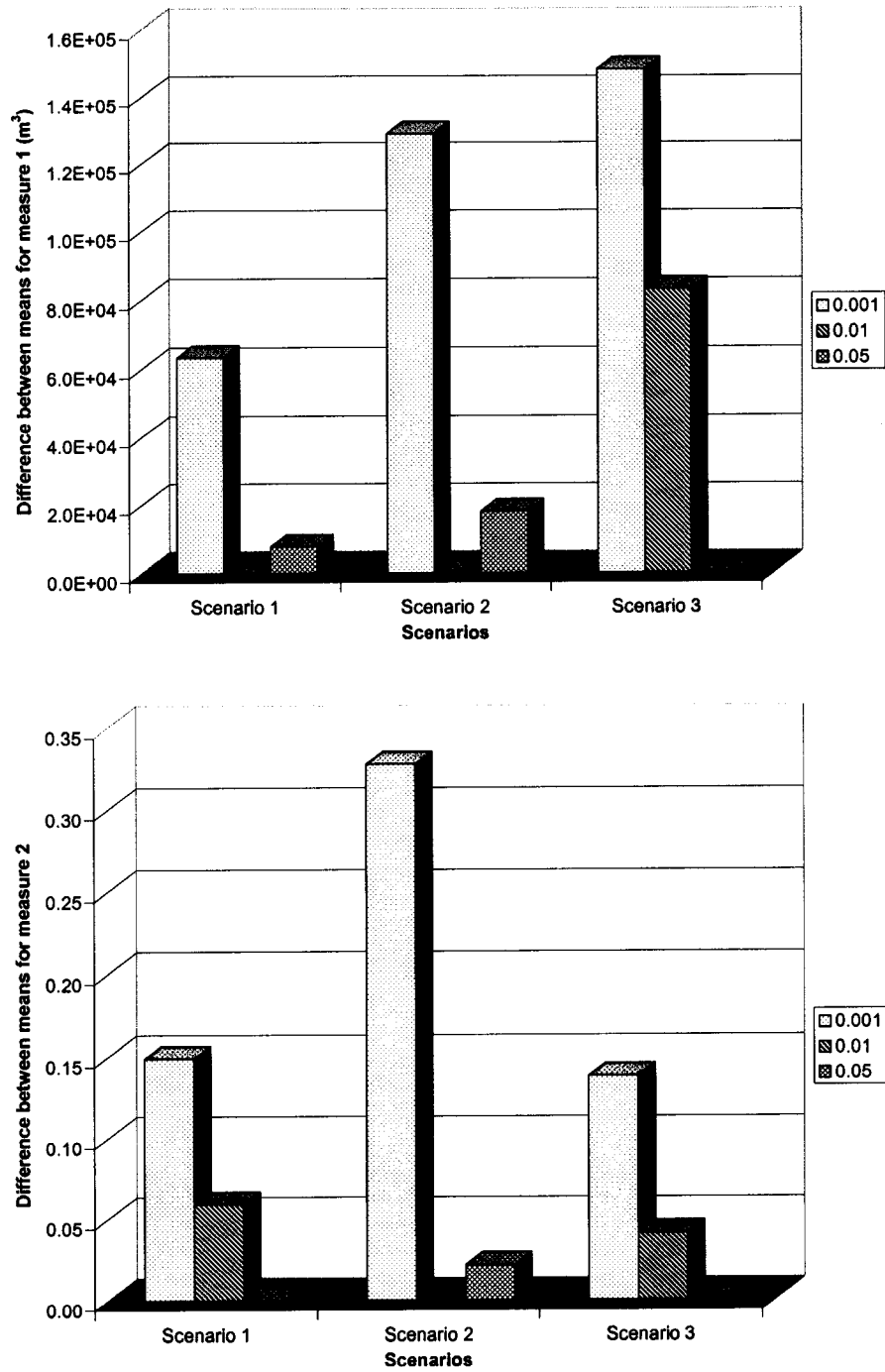


Figure 4.3

Difference between means of different mutation rates for both measures

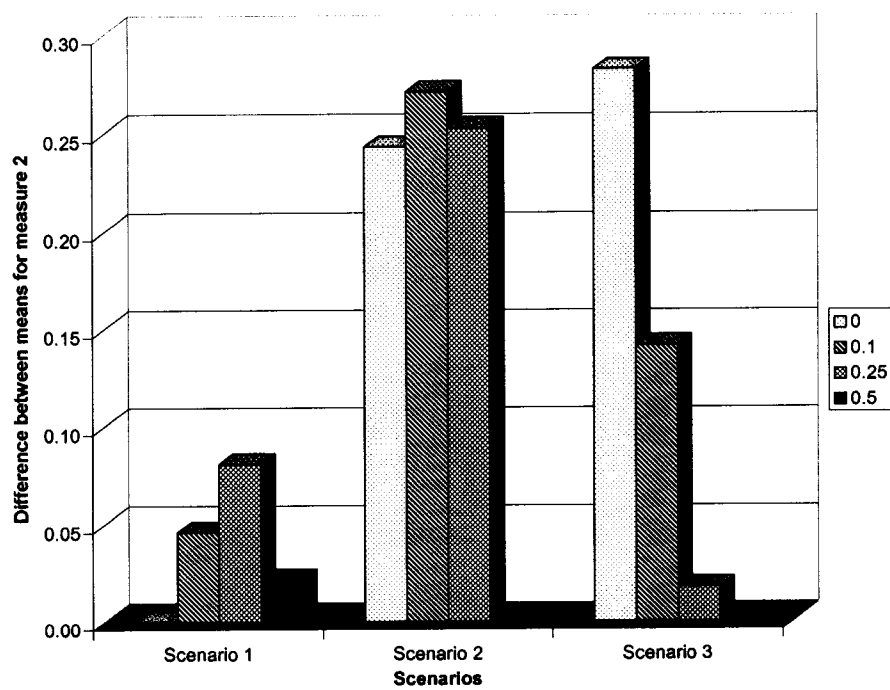
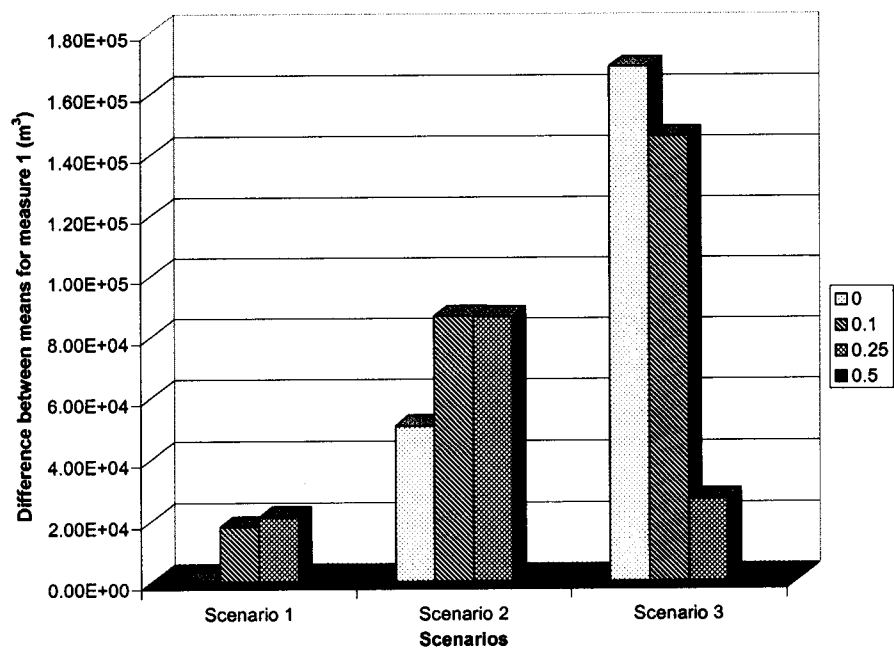


Figure 4.4

Difference between means of different blend crossover extensions for both measures

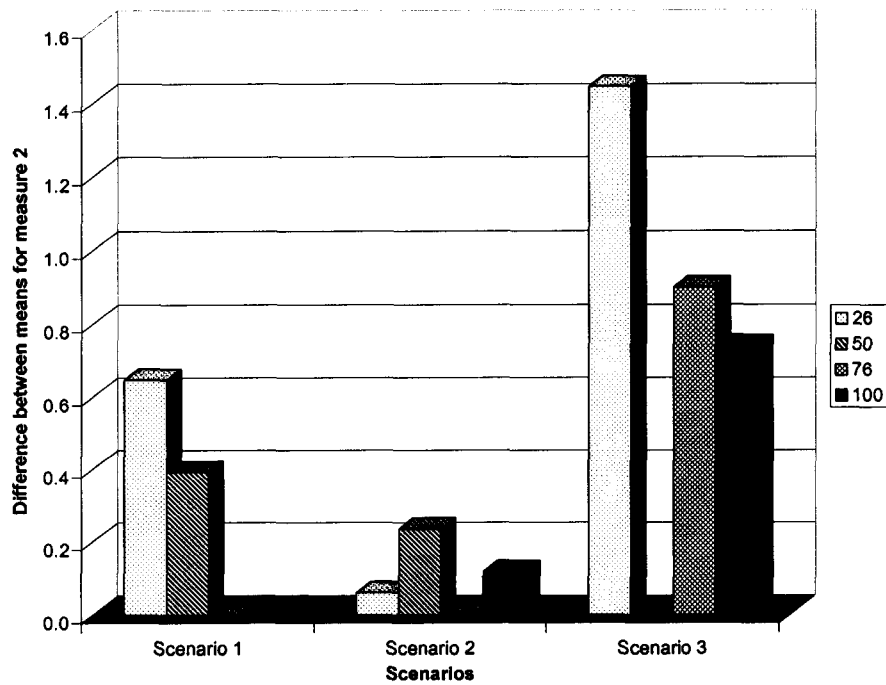
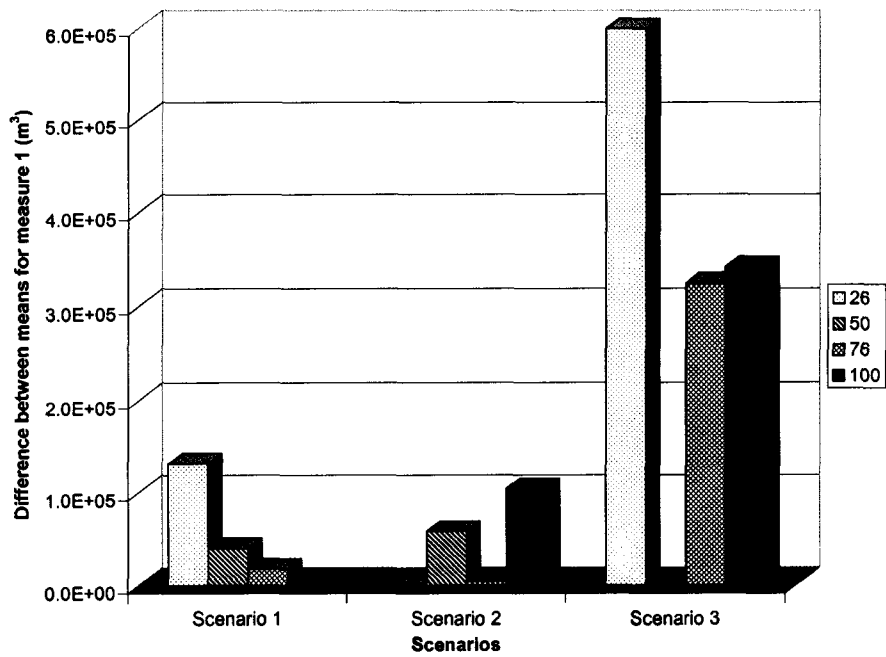


Figure 4.5  
Difference between means of different populations sizes for both measures

#### 4.7.2 Test 2 (Parameter interaction charts)

The following figures present examples of the parameter interaction charts for different scenarios of the case study. The parameter interaction charts are drawn between crossover probabilities and mutation rates. It could be noticed that the differences in the second measure are higher than the differences of the first measure for all scenarios. Figure 4.6 presents four examples for the first scenario of the case study. For the second measure with  $\alpha = 0.1$  and  $\alpha = 0.25$ , higher mutation rates perform better for most of crossover probabilities, although  $p_m = 0.001$  has best result with  $p_c = 0.9$  in one of the figures.

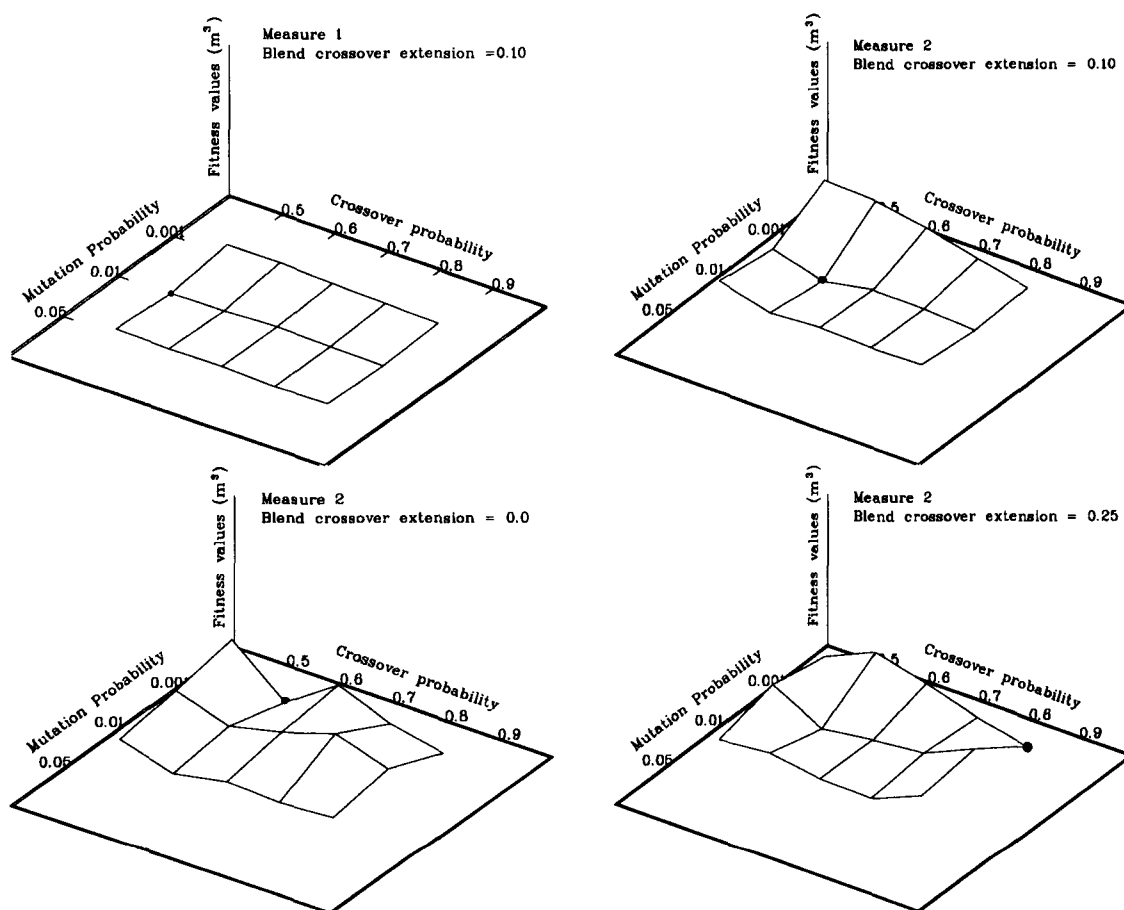


Figure 4.6  
Parameter interaction charts for the first scenario of the case study

For crossover, the best values are  $p_c = 0.5$ ,  $p_c = 0.6$  and  $p_c = 0.8$ . The significance of these values could be noticed from different interaction charts, but no specific value is consistent in all figures. Also,  $p_c = 0.9$  has the best value for one of the figures. In general, evidence about mutation rate can be noticed from some charts, but there are no significant effects between crossover probabilities in this scenario.

Figure 4.7 presents examples for the second scenario of the case study. From this figure, increasing mutation rate is associated with an improvement in the results. This is more clear in the second measure. There are no clear evidence about the best crossover probability, but  $p_c = 0.5$ ,  $p_c = 0.6$  and  $p_c = 0.8$  have good results.

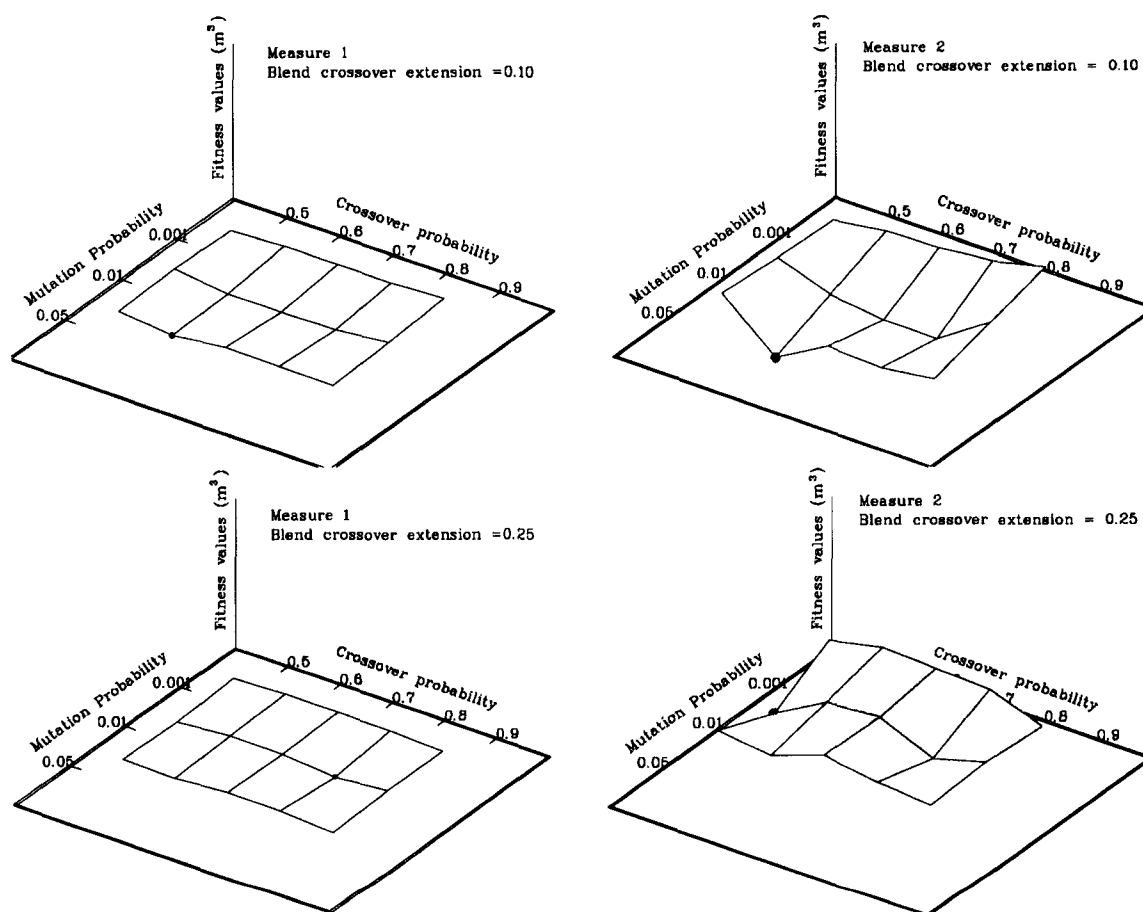


Figure 4.7  
Parameter interaction charts for the second scenario of the case study

For the third scenario of the case study (Figure 4.8),  $p_c = 0.7$  outperform all other crossover probabilities for both measures. There is no any clear evidence about the best mutation rate.

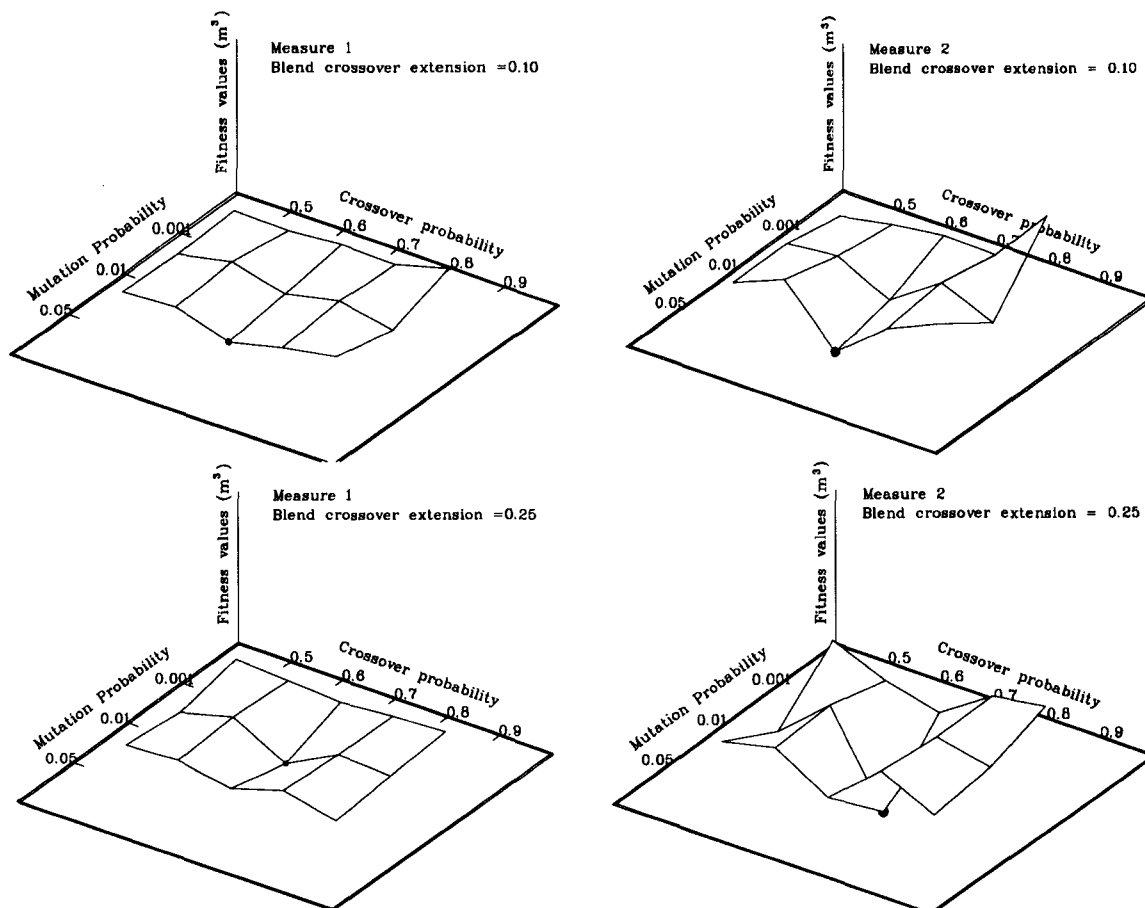


Figure 4.8  
Parameter interaction charts of the third scenario of the case study

These parameter interaction charts for all scenarios of the case study support the results of test 1 regarding the following points:

- ❑ Crossover probability  $p_c = 0.7$  is the best value for the third scenario.
- ❑ There is no clear evidence about the best crossover probability value in the first two scenarios.  $p_c = 0.5$ ,  $p_c = 0.6$  and  $p_c = 0.8$  have good results. Also,  $p_c = 0.9$  has some good results in the first scenario.

□ Higher mutation rates ( $p_m = 0.01$  and  $p_m = 0.05$ ) perform better in the first two scenarios.

□ There is no clear evidence about the mutation rate in the third scenario.

Figure 4.9 presents parameter interaction charts for the population sizes for different scenarios of the case study. The following points can be noticed:

□ In the first two scenarios of the case study, higher population sizes (N=76 and 100) perform consistently well for all crossover probabilities. Although smaller populations sizes have the smallest point in some case, it is not consistent between different crossover probabilities.

□ N=50 is clearly the best population size for the third scenario of the case study.

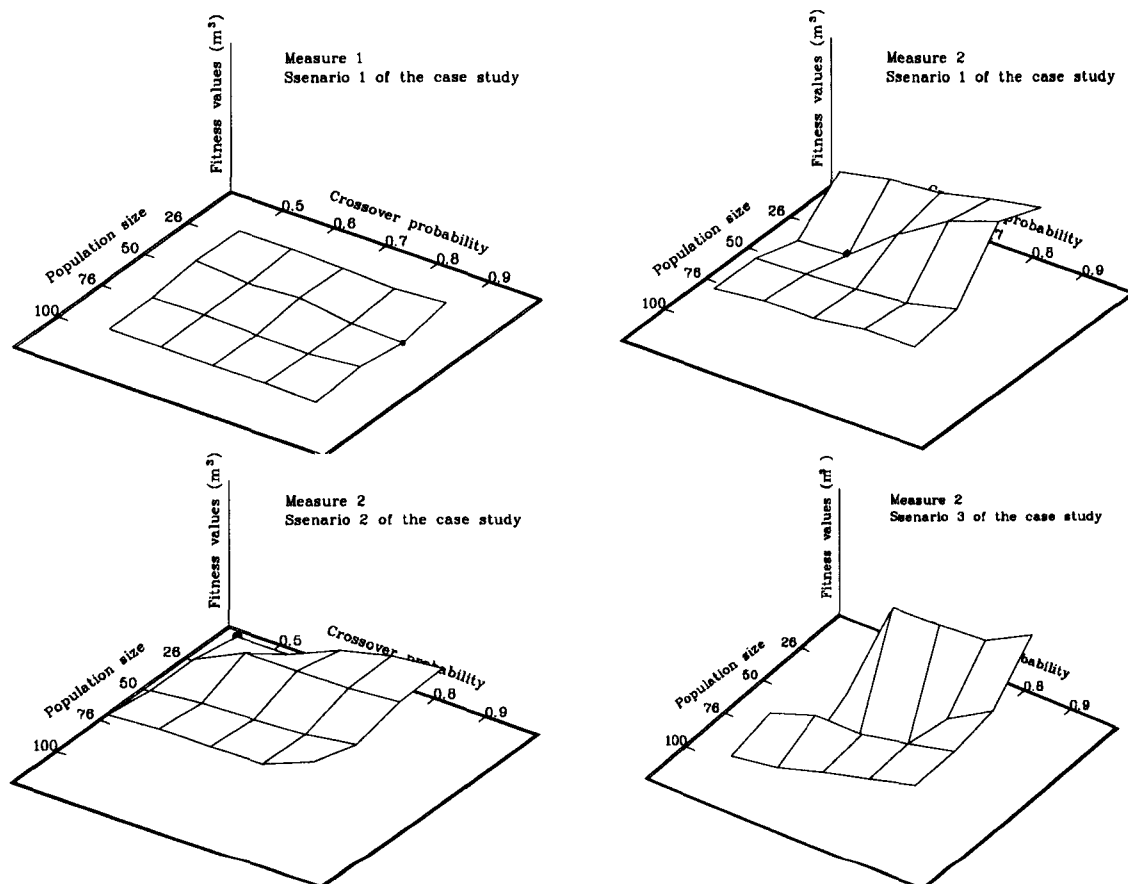


Figure 4.9  
Parameter interaction charts for different scenarios for population size



### 4.7.3 Test 3 (comparing runs that have the same parameters)

60 runs were conducted for each scenario of the case study for all combinations of five crossover probabilities, three mutation rates, and four blend crossover extension values. The runs that share the same parameters are as follows:

- For each crossover probability, there are 12 categories have the same mutation rate and blend crossover extension.
- Similarly, for each mutation rate, there are 20 different categories, and for each blend crossover extension, there are 15 different categories.
- For each population size, there are five different categories have the same crossover probability.

Table 4.2 presents the results of the comparisons for all of these categories, and how many each parameter wins from these comparisons.

Table 4.2(a-d): Test 3: number of wins for each value of the GA parameters

Table 4.2 (a): Number of wins for each crossover probability value

Crossover Probability	Different measures of scenario 1		Different measures of scenario 2		Different measures of scenario 3	
	1	2	1	2	1	2
0.5	4	1	1	4	1	2
0.6	4	7	5	5	1	0
0.7	1	0	1	1	9	8
0.8	2	2	3	1	1	2
0.9	1	2	2	1	0	0

Table 4.2 (b): Number of wins for each mutation rate value

Mutation Rate	Different measures of scenario 1		Different measures of scenario 2		Different measures of scenario 3	
	1	2	1	2	1	2
0.001	6	5	0	3	6	6
0.01	9	8	11	9	6	6
0.05	5	7	9	8	8	8

Table 4.2 (c): Number of wins for each blend crossover extension value

Blend Crossover Extension	Different measures of scenario 1		Different measures of scenario 2		Different measures of scenario 3	
	1	2	1	2	1	2
0.0	3	4	4	3	0	1
0.1	5	3	1	2	2	3
0.25	4	2	0	2	4	5
0.5	3	6	10	8	9	6

Table 4.2 (d): Number of wins for each population size value

Population Size	Different measures of scenario 1		Different measures of scenario 2		Different measures of scenario 3	
	1	2	1	2	1	2
26	0	0	3	2	0	0
50	2	2	1	1	4	4
76	1	2	1	1	0	0
100	2	1	0	1	1	1

- Regarding crossover probability (Table 4.2(a)), and for the first two scenarios of the case study,  $p_c = 0.5$  and  $p_c = 0.6$  have the best results for both measurements.  $p_c = 0.6$  performs better. For the third scenario of the case study,  $p_c = 0.7$  is the best value for both measurements.
- Regarding mutation rate (Table 4.2(b)),  $p_m = 0.05$  has the best results for the third scenario, and  $p_m = 0.01$  has the best results for the first and second scenarios.
- Regarding blend crossover extension (Table 4.2(c)),  $\alpha = 0.5$  is the best value in all cases except for the first measure of the first scenario.
- For population size (Table 4.2(d)),  $N=50$  is the best value in the third scenario of the case study. The difference between this value and other values is clear.  $N=50$  has also better results in the first scenario, and  $N=26$  has better results in the second scenario, but the differences are not clear as in the third scenario.

There are some points, which are consistent with the previous results, such as:

- $p_c = 0.7$  and  $N=50$  are the best values for the third scenario of the case study.
- Higher mutation rates ( $p_m = 0.01$  and  $p_m = 0.05$ ) outperform  $p_m = 0.001$  for all scenarios.

For crossover probabilities of the first two scenario, this test explores that smaller crossover probabilities ( $p_c = 0.5$  and  $p_c = 0.6$ ) outperform higher values ( $p_c = 0.8$ ). Also, there is no clear evidence about the best population size in the first two scenarios.

#### 4.7.4 Test 4 (parameters of the best solution)

Table 4.3 presents the parameters for the value found in any run for each measure for each scenario of the case study.

Table 4.3(a-d): Parameters of the best value found for each measure.

Table 4.3 (a): Parameters of the best value for the first scenario of the case study

	$\alpha$	$p_m$	$p_c$
<b>Measure 1</b>	0.5	0.05	0.5
<b>Measure 2</b>	0.5	0.05	0.5

Table 4.3 (b): Parameters of the best value for the second scenario of the case study

	$\alpha$	$p_m$	$p_c$
<b>Measure 1</b>	0.5	0.05	0.6
<b>Measure 2</b>	0.5	0.05	0.6

Table 4.3 (c): Parameters of the best value for the third scenario of the case study

	$\alpha$	$p_m$	$p_c$
<b>Measure 1</b>	0.25	0.01	0.7
<b>Measure 2</b>	0.25	0.01	0.7

Table 4.3 (d): Population sizes of the best value for different scenarios of the case study

	Population size		
	Scenario 1	Scenario 2	Scenario 3
<b>Measure 1</b>	50	26	50
<b>Measure 2</b>	50	26	50

- Regarding blend crossover extension, higher values ( $\alpha = 0.25$  and  $\alpha = 0.5$ ) perform better.  $\alpha = 0.5$  is the best value for both measurements of the first two scenarios of the case study.  $\alpha = 0.25$  is the best value for both measurements of the third scenario of the case study.
- Regarding mutation rate, higher values ( $p_m = 0.01$  and  $p_m = 0.05$ ) perform better. For the first two scenarios of the case study,  $p_m = 0.05$  is the best value for both

measurements. For the third scenario of the case study,  $p_m = 0.01$  is the best value for both measurements.

- Regarding crossover probability, the best value increases from  $p_c = 0.5$  to  $p_c = 0.7$ , while increasing the difficulty of the problem and increasing the number of decision variables.
- Regarding population sizes,  $N=50$  is the best value for the first and third scenarios, and  $N=26$  is the best value for the second scenario of the case study.

## 4.8 Summary

A summary of the results of the different tests previously described is presented here.

- Crossover probability:
  - ✎ Test 1 (Mean values):
    - ✓ There is statistical confidence that  $p_c = 0.7$  is the best crossover probability for the third scenario.
    - ✓  $p_c = 0.6$  and  $p_c = 0.8$  have best means for the first and second scenarios, but they are not confirmed statistically.
  - ✎ Test 2 (parameter interaction charts):
    - ✓ There is no clear evidence for the first and second scenarios.  $p_c = 0.5$ ,  $p_c = 0.6$ , and  $p_c = 0.8$  are the best values for these two scenarios.  $p_c = 0.7$  is the best for the third scenario.
  - ✎ From test 3 and test 4,  $p_c = 0.7$  is the best value for the third scenario and smaller values  $p_c = 0.5$  and  $p_c = 0.6$  are the best values for the first two scenarios.
  - ✎ Conclusion:
    - ✓  $p_c = 0.7$  is the best crossover probability for the complex scenarios of this model. This is confirmed by all tests. Smaller values ( $p_c = 0.5$  and

$p_c = 0.6$ ) are the best values for simple scenarios of the model, but this is not confirmed by all tests.

□ Mutation rate:

✎ Test 1 (Mean values):

✓ For the first two scenarios of the case study, higher mutation rates ( $p_m = 0.01$  and  $p_m = 0.05$ ) are the best group with no statistical difference between them. For scenario 3 of the case study, there is no statistical evidence.

✓  $p_m = 0.01$  has the best mean for the first two scenarios.  $p_m = 0.05$  has the best mean for the third scenario.

✎ Test 2 (parameter interaction charts):

✓ The results are consistent with the first test. The range of  $p_m = 0.01$  to  $p_m = 0.05$  perform better in the first two scenarios of the case study, while there is no clear evidence about the third scenario.

✎ From test 3 and test 4, higher mutation rates  $p_m = 0.01$  and  $p_m = 0.05$  are the best values for all scenarios, but the best of them is different from test to the other. For the third scenario,  $p_m = 0.05$  is the best value in test 3, and  $p_m = 0.01$  is the best value in test 4. The opposite is true for the first two scenarios.

✎ Conclusion:

✓ Statistically,  $p_m = 0.01$  to  $p_m = 0.05$  is the best range for the mutation rate in the first two scenarios.

✓ From other tests, it looks like that  $p_m = 0.01$  is the best value for simpler scenarios, and  $p_m = 0.05$  is better for complex scenarios of the model.

□ Blend crossover extension:

✎ There is only statistical evidence that  $\alpha = .5$  is the best value for the second measure of the second scenario. From other tests, it looks like that this value is the best value for all scenarios. Although  $\alpha = 0$  has a better mean in the first scenario, this is not confirmed by other tests.

- Population size:
  - ✎ Test 1 (Mean values):
    - ✓ There is no statistical evidence about the population size.
    - ✓ N=50 has the best mean for the third scenario. For the first two scenarios, the best mean is different from one measure to the other.
  - ✎ Test 2 (parameter interaction charts):
    - ✓ N=50 is the best value for the third scenario. For the first two scenarios, higher populations sizes are more stable for most of the runs, although small population sizes have some good results.
  - ✎ From test 3 and test 4, N=50 is the best value for the third scenario, and smaller population sizes (N=26 and 50) are the best values for the first two scenarios.
  - ✎ Conclusion:
    - ✓ N=50 is the best population size for the complex scenarios of the case study.
    - ✓ There is some doubt about the best population size for simpler scenarios. Some tests support that higher population sizes are the best, while others support that smaller population sizes are the best. It is the same phenomena mentioned by Deb and Agrawal (1999), where “two distinct ranges of population sizes (with a dip in performance in intermediate population sizes) works the best.”

#### 4.9 Validate the results

To validate the previous results, and to check the results that have some doubt, different alternatives for each scenario are tested in this section with different initial seed values, and with different constraint-handling technique for each scenario.

For the third scenario, where most of the parameters are confirmed, only two alternatives are tested. The first alternative represents the recommended parameters, and the second alternative represents different parameters for the comparison.

For the first and second scenarios, as there is doubt about some parameters, five different alternatives are tested. The first four alternatives represent different stages of the recommended data, and the fifth represents the different parameters for the comparison.

Table 4.4 represents the parameters for all of these alternatives. The constraint-handling techniques that are used with different scenario are (See Chapter 5 for the explanation of each technique):

- Multiobjective technique is used for the first scenario.
- Adaptive penalty technique, with original tournament selection is used with the second scenario.
- Stochastic tournament selection is used with the third scenario.

Table 4.4: GA parameters for different alternatives

Scenarios	Alternatives	GA Parameters			
		N	$p_c$	$p_m$	$\alpha$
Scenario 1	Alternative 1	76	0.5	0.05	0.5
	Alternative 2	76	0.8	0.05	0.5
	Alternative 3	76	0.6	0.05	0.5
	Alternative 4	76	0.6	0.05	0.0
	Alternative 5	50	0.7	0.001	0.5
Scenario 2	Alternative 1	76	0.5	0.05	0.5
	Alternative 2	76	0.6	0.05	0.5
	Alternative 3	76	0.6	0.01	0.5
	Alternative 4	26	0.6	0.05	0.5
	Alternative 5	50	0.7	0.001	0.5
Scenario 3	Alternative 1	50	0.7	0.05	0.5
	Alternative 2	76	0.5	0.001	0.5

The results are presented in Figures 4.10 to 4.12.

- Regarding the first scenario, alternative 3 has the best results. The worst average is related to alternative 5 (non-recommended parameters). Average values of alternatives 1 and 2 are close to the average value of alternative 5.
- Regarding the second scenario, the best results are obtained by alternatives 2 and 3. The averages are very close to each other. However, alternative 2 slightly outperforms alternative 3. The worst results are related to alternative 4, which

uses a small population size ( $N=26$ ). Except alternative 4, the recommended values (alternatives 1 to 3) outperform other values (alternative 5).

- Regarding the third scenario, the recommended parameter values (alternative 1) outperforms the other values (alternative 2) for most of the runs and for the average.

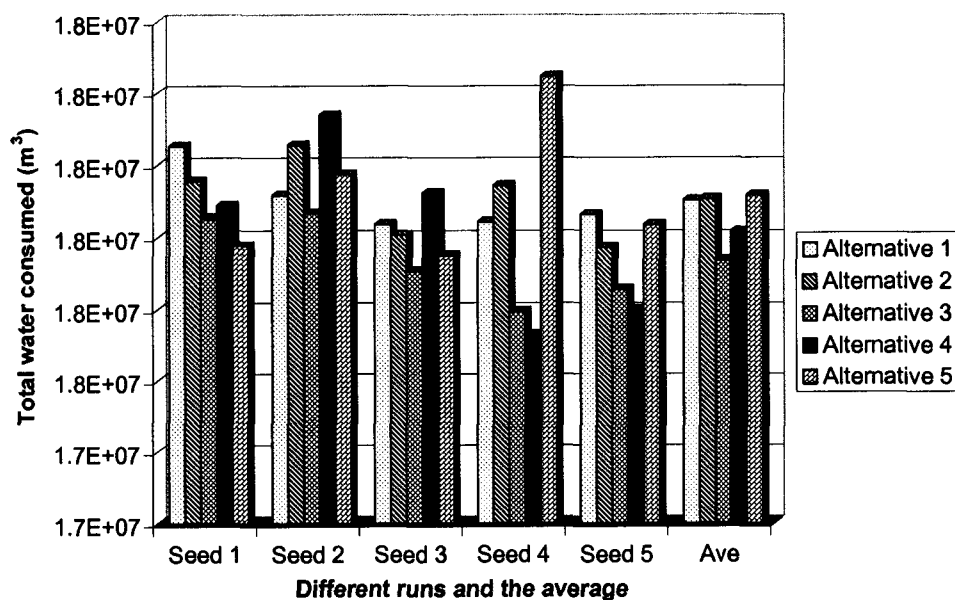


Figure 4.10  
Different alternatives of recommended parameters for the first scenario



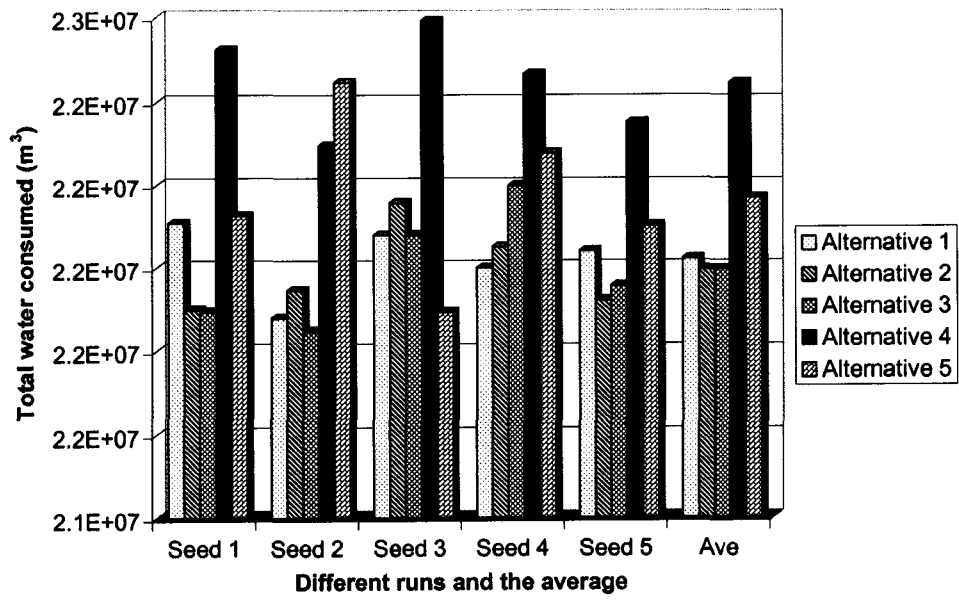


Figure 4.11  
Different alternatives of recommended parameters for the second scenario

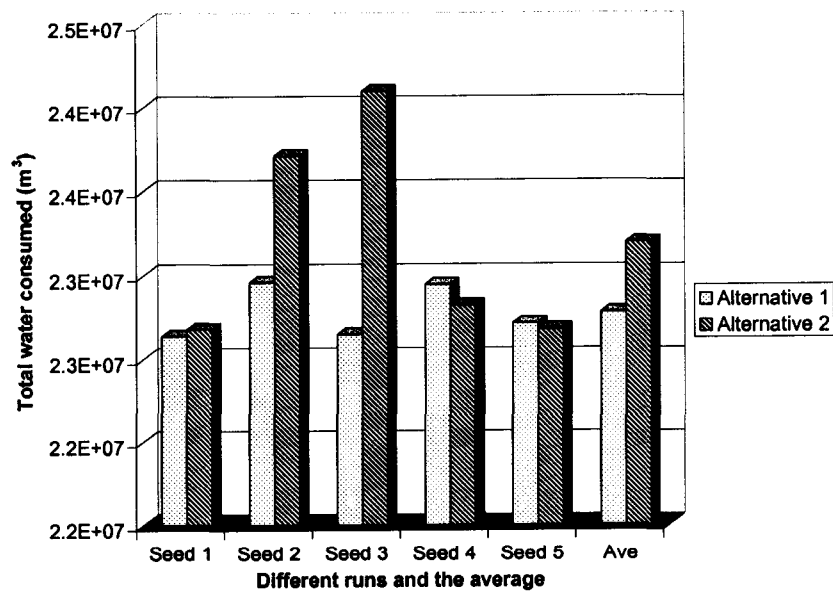


Figure 4.12  
Different alternatives of recommended parameters for the third scenario

## 4.10 Conclusions and future work

An experiment was conducted to define the best values for GA parameters for the current model. The importance of defining these parameters stems from the fact that most of the previous works regarding this point were done using binary encoding and operators, and with explicit equations and unconstrained optimization models. The current model evaluates the strings using a simulation model, and uses real GA encoding and operators.

All combinations of the values of crossover probabilities, mutation rates, and blend crossover extensions have been tested. Population sizes were tested for different crossover probabilities. The best parameters values obtained were validated by testing them against other parameters values with different initial seed values.

As a conclusion of this work, it is likely that the following values are most suitable for the GA parameters:

- The best crossover probabilities values are between 0.6 and 0.7. Higher values work better for scenarios and higher number of decision variables.
- The best mutation rates are between 0.01 and 0.05. Higher values (0.05) are recommended for most of the scenarios.
- The best blend crossover extension value is 0.5. This value is recommended for all scenarios of the model
- The best population sizes are between 50 and 76. Smaller population sizes work better for harder scenarios and higher number of decision variables.

For future work, the adaptive and self-adaptive techniques may be useful for the current model, as the best parameter values depend on the scenario.

## 4.11 References

- Coello, C. A. C., "Constraint-Handling Using an Evolutionary Multiobjective optimization Technique," *Civil Engineering and Environmental Systems*, Vol. 17, pp. 319-346, 2000.
- Deb, K., (2001) *Multi-objective optimization using evolutionary algorithms*, John Wiley & Sons, Ltd, 2001
- Deb, K. and Agrawal, S., "Understanding interactions among genetic algorithm parameters," *Foundation of Genetic Algorithms 5 (FOGA 5)*, 1999, pp. 265-286
- De Jong, K. A. *The Analysis of the Behavior of a Class of Genetic Adaptive System*, PhD thesis, Department of Computer Science, University of Michigan, Ann Arbor, Michigan, 1975
- Eiben, A.E., Hinterding, R., and Michalewicz, Z., "Parameters Control in Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, Vol.3, No.2, 1999, pp.124-141
- Fogarty, T. C., "Varying the probability of mutation in the Genetic Algorithms," *proceedings of the third International Conference on Genetic Algorithms*, Morgan Kaufman, 1989, pp. 104-109
- Grefenstette, J. J. "Optimization of Control Parameters for Genetic Algorithm", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 16, No 1, 1986- pp. 122-128
- Goldberg, D. E., "Optimal Initial Population Size for Binary-Coded Genetic Algorithms," (*TCGA Report No. 85001*), Tuscaloosa: University of Alabama, The Clearinghouse for Genetic Algorithms, 1985
- Goldberg, D. E., "Sizing populations for Serial and Parallel Genetic Algorithms," *proceedings of the third International Conference on Genetic Algorithms*, Morgan Kaufman, 1989, pp. 70-79
- Goldberg, D. E., Deb, K., and Clark, J. H., "Genetic Algorithms, noise, and the sizing of population," *Complex Systems* Vol. 6, No 4, 1992, pp. 333-362
- Hart, W. E. and Belew, R. K. "Optimizing an Arbitrary Function is Hard for the Genetic Algorithm," *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufman, 1991, pp. 190-195

- Hinterding, R., Michalewicz, Z., and Peachey, T.C., "Self-Adaptive Genetic Algorithm for Numeric Functions," *Proceedings of the Fourth International Conference on Parallel Problem Solving from Nature*, Berlin, Sept 1996, pp. 420-429
- Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, New York, 1996
- Schaffer, J., Caruana, R., Eshelman, L., and Das R., "A study of control parameters affecting online performance of Genetic Algorithms for Function Optimization," *proceedings of the third International Conference on Genetic Algorithms*, Morgan Kaufman, 1989, pp. 51-60
- Syswerda, G., "Schedule optimization using genetic algorithms," *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991, pp. 332-349.

## CHAPTER 5

### CONSTRAINT-HANDLING TECHNIQUES

#### 5.1 Introduction

The presence of the constraints always increases the difficulty of an optimization problem, whether using a gradient-based or evolutionary optimization technique. Evolutionary techniques are affected more since they cannot handle constraints explicitly. This inability to handle constraints requires using a heuristic to guide the search toward feasible and good-performing solutions. However, this heuristic is affected by many things, including the complexity of the problem to be solved, type of constraints, and number of constraints. In this chapter, the performance of various constraint handling techniques will be compared to determine which techniques perform best for the current model, which of them should be used with simpler problems, and which are more suitable for more complex problems. Various techniques from the literature, as well as two new proposed techniques, are investigated with the goal being to check which of these techniques is more suitable for this model based on the level of difficulty of the problem to be solve and based on the number of constraints. According to Deb (2001), Michalewicz and Schoenauer (1996), and Michalewicz et al. (2000), most constraint handling techniques which exist in the literature, can be classified into the following five categories:

- Methods based on preserving feasibility of solutions.
- Methods based on penalty functions.
- Methods biasing feasible over infeasible solutions.
- Methods based on decoders.
- Hybrid methods.

In the current study, most of the techniques investigated are related to the second category, which is penalty functions, including static, dynamic, adaptive and self-adaptive forms. These methods also incorporate the third category, as each penalty function will be tested twice, with one of these implementations biasing feasible over

infeasible solutions. Other methods are also related to the third category, including multiobjective and stochastic methods.

## 5.2 Techniques investigated for the current model

Techniques that are tested in the current study can be categorized as follows:

- Penalty functions techniques.
- Multi-objective optimization techniques.
- Self-adaptive techniques.
- Stochastic techniques.
- Adaptive techniques.

### 5.2.1 Penalty Functions

Penalty functions are by far the most commonly used constraint-handling technique. Penalty functions essentially degrade the fitness of solutions that violate constraints by including a penalty term in the fitness function.

According to Michalewicz et al. (1994), the rule to design a penalty function is “the penalty should be kept as low as possible, just above the limit below which infeasible solutions are optimal.” However, as the authors stated, it is difficult to implement this rule effectively.

Also, according to Michalewicz (1995), the appropriate choice of penalty function depends on

- The ratio between sizes of the feasible and the whole search space.
- The topological properties of the feasible search space.
- The type of the objective function
- Number of variables
- Number of constraints
- Types of constraints
- Number of active constraints at the optimum

There are no specific regulations for creating penalty functions, but Richardson et al. (1989) gave guidelines that should be considered while selecting the penalty function, which are:

- Penalties that are functions of the distance from feasibility perform better than those that are only functions of the number of violated constraints.
- For a problem having few constraints and few feasible solutions, penalties which are solely functions of the number of violated constraints are not likely to produce any feasible solutions.
- Good penalty functions can be constructed from two quantities: the maximum completion cost and the expected completion cost. The completion cost refers to the distance to feasibility.
- Penalties should be close to the expected completion cost, but should not frequently fall below it. The more accurate the penalty, the better will be the solution found. When a penalty often underestimates the completion cost, the search may fail to find a solution.

There are many approaches to implement penalty functions. The first approach is static penalty functions, for which the parameters are kept constant during the whole run. This is the easiest form of penalty function to implement, but may be the least efficient one. According to Eiben et al. (1999), “any static set of parameters, having the values fixed during an EA run (parameter tuning), seems to be inappropriate.” The reason for this, as they stated, is that “EA is an intrinsically dynamic, adaptive process. The use of rigid parameters that don’t change their values is thus in contrast to this spirit.”

The second approach is dynamic penalty functions, where the parameters are changing during the run. The easiest way is to change the parameters based on the number of generations. According to Siedlecki and Sklanski (1989), “the genetic algorithms with a variable penalty coefficient outperform the fixed penalty factor algorithms.” Harrell and Ranjithan (1999) tested 22 different penalty functions with a watershed management design problem including constant, increasing, and decreasing penalty functions, and they stated, “In general, increasing the penalty value with generation seems to perform the best in most cases”.

Another way is to change the penalty function based on some criteria instead of changing it based on the generation number. An example of this is what was proposed by Michalewicz and Attia (1994) based on the idea of simulated annealing. In their technique, the penalty coefficient is changed once in many generations after the convergence to local optima.

Another approach of penalty functions is to adapt the penalty coefficients based on the feedback of the previous generations. Many techniques have been proposed regarding this approach. Bean and Hadj-Alouane (1992) introduced a procedure in which the penalty increases or decreases based on whether the best individual in the last  $k$  generations was always or was never feasible. Also Homaifar et al. (1994) suggested creating several levels of violations ( $l$ ) for each constraint, and defining a penalty function for each constraint and each level of violation. A new adaptive technique is proposed and tested in the current study.

Another way of implementing penalty functions is to use different penalty functions simultaneously with different sub-populations, as was introduced by Le Riche et al. (1995).

Also some researchers, including Coello (1999) used self-adaptive techniques, whereby the GA itself is used to find the best penalty parameters.

Other penalty techniques give superiority to feasible solutions over infeasible solution regardless of the fitness values, including those proposed by Powell and Skolnick (1993) and Michalewicz and Xiao (1995). In the current study, each penalty function will be tested once using original tournament selection, and again using tournament selection with superiority of feasible solutions.

Regardless of its widespread use in GAs, penalty functions have the following weaknesses:

- The requirement to fine-tune penalty parameters, which makes the penalty functions problem dependent.
- Penalty functions, especially in the static form, don't get any feedback from the search.
- The coefficients in the penalty function may lead to under-penalization or over-penalization, which means that the penalty terms could be too small to influence



the selection of individuals, or so large that the objective function has little or no influence (Runarsson and Yao, 2000).

Among penalty function techniques, three different techniques will be tested for the current model, which are described in the following subsections.

#### 5.2.1.1 Additive Static Penalty (ASP)

In this technique, the amount of the violation is multiplied by factors (penalty coefficients), and then added to the objective function. The fitness equation is as follows:

$$F = \sum_{i \in NT} Q_i - \sum_{i \in NT} \sum_{o \in NO} Q_{i,o} + n_1 \sum_{r \in R} UIA_r + n_2 \sum_{r \in R} FL_r + n_3 \sum_{g \in NSG} DSA_g + n_4 * \max((RWV - WV), 0.0) \dots\dots\dots(5.1)$$

Where

$F$	Fitness equation.
$R$	Number of reaches in the network.
$UIA_r$	The un-irrigated cultivated area at reach r.
$FL_r$	The flood length in reach r.
$NSG$	Number of unstable regulators in the network.
$DSA_g$	The cultivated area downstream the regulator g.
$RWV$	The required water volume at the end of the routing.
$WV$	The actual water volume at the end of the routing.
$n_1, n_2, n_3, n_4$	Coefficients.

The violation is measured as follows:

- For the water shortage and regulator stability constraints, the violation will be measured in a cultivated area (Feddan).
- For the flood constraint, the violation will be measured as a length (m).

- For the required water level constraint, the difference in water volume between the required water volume and actual water volume will be used (m<sup>3</sup>).

The violations for the first three constraints will be considered if they happen at any time step, except for the water shortage of the open branch that will be calculated after the traveling time. In the fourth constraint, it will be calculated at the last time step.

### 5.2.1.2 Multiplicative Static Penalty (MSP)

In this technique, the ratio between the amount of the violation and the total possible violation will be used in the fitness equation, which is defined as follows:

$$F = \sum_{t \in NT} Q_t - \sum_{t \in NT} \sum_{o \in NO} Q_{t,o} + n_1 \frac{\sum_{r \in R} UIA_r}{\sum_{r \in R} TCA_r} + n_2 \frac{\sum_{r \in R} FL_r}{\sum_{r \in R} TL_r} + n_3 \frac{\sum_{g \in NSG} DSA_g}{\sum_{g \in TG} DSA_g} + n_4 * \frac{\max((RWV - WV), 0.0)}{RWV} \dots\dots\dots(5.2)$$

Where

- TCA*            The total cultivated area of reach r.
- TL*             The total length of reach r.
- TG*             The total number of regulators.

### 5.2.1.3 Additive Linear Dynamic Penalty (ALDP)

In this technique, the fitness equation is similar to equation 5.1, but the coefficients will be calculated based on the current generation, as follows.

$$F = \sum_{t \in NT} Q_t - \sum_{t \in NT} \sum_{o \in NO} Q_{t,o} + n_1^g \sum_{r \in R} UIA_r + n_2^g \sum_{r \in R} FL_r + n_3^g \sum_{g \in NSG} DSA_g + n_4^g * \max((RWV - WV), 0.0) \dots\dots\dots(5.3)$$

$$n_i^g = n_i * \left( n_{base} + n_{inc} * \frac{g}{G} \right) \dots\dots\dots(5.4)$$

Where

$n_{base}$ & $n_{inc}$	Coefficients.
$g$	The current generation.
$G$	Total generation.

### 5.2.2 Multiobjective Optimization

The idea of converting a single objective optimization problem, such as the one in the current study, to multiobjective optimization, is to treat constraints as objectives; thus, there will be  $(1+m)$  objective functions, where  $(m)$  is the number of constraints. Thus, the ideal solution  $\bar{X}$  would have  $f_i(\bar{X})=0$  for  $1 \leq i \leq m$  and  $f(\bar{X})=f(\bar{Y})$  for all  $\bar{Y} \in F$ .

One main approach in multiobjective optimization is to use Pareto-optimal (non-dominated) solutions. The idea of non-dominated solutions presented by Srinivas and Deb (1995) is “In a typical multiobjective optimization problem, there exists a set of solutions which are superior to the rest of solutions in the search space when all objectives are considered but are inferior to other solutions in the space in one or more objectives. These solutions are known as Pareto-optimal solutions or non-dominated solutions. Since none of the solutions in the non-dominated set is absolutely better than any other, any one of them is an acceptable solution.” Based on that idea, Srinivas and Deb (1995) and Deb and Goal (2000), presented the Non-Dominated Sorting Genetic Algorithm (NSGA & NSGA-II). In this technique, the fitness will be calculated as follows:

- All strings will be ranked using Pareto Fronts based on non-dominance.
- The fitness values of all the strings in any front will be the number (rank) of this front. So, the minimum fitness is 1.0, and it will be increased to 2.0, 3.0, and so on.
- In the last rank, and to choose few strings to complete the population, a distance will be used as a way to select these few. In this method, the strings that have fewer individuals around it will be selected.

- From the second generation, two generations will be used for ranking, and the best number (equal to the population size will be selected).

Another technique based on multiobjective optimization that is tested within the current model is based on that presented by Coello (2000). This technique is presented in the following subsection.

#### 5.2.2.1 Multiobjective method used in the current study

This technique, proposed by Coello (2000), sorts the solutions based on their objective values and their violations of the constraints, and assigns fitness values for different solutions based on that sorting. In this technique, a feasible solution will always be superior to infeasible solutions. The procedure of this technique is as follows:

- The count of all individuals in the current generation is initialized to zero.
- Each individual will be tested against every other individual in the population using pair wise comparison.
- If both individuals are feasible, the count of both will remain unchanged.
- If one of the individuals is feasible and the other is infeasible, the count of the infeasible will be increased by one.
- If both are infeasible and one violates more constraints than the other, the count of the individual that violates more constraints will increase by one.
- If both are infeasible, and both violate the same number of constraints, and one has total amount of violations larger than the other, the count of that one will increase by one.
- Rank the individuals and make selection based on rank.

In this study, the procedure will be modified as follows:

- The fitness of feasible solutions will be normalized between 0.0 and 1.0, so the highest fitness value of a feasible solution will be 1.0. This normalized fitness is used as the fitness value for each feasible solution.
- The fitness of any infeasible solution will be  $(1 + \text{Count})$ .
- Binary tournament selection is used.

### 5.2.3 Self-adaptive penalty function

The idea of self-adaptive techniques stems from considering setting suitable penalty parameters as an optimization problem itself, and uses the GA to solve this problem in addition to solving the original problem. Thus, the GA is used to progressively improve the penalty function parameters based on feedback of the progress through the generations. As in a conventional GA, the model will randomly define several penalties, in parallel within the optimization problem, and check how much each of them will improve the solution. This measurement of the improvement is treated as the fitness function in the GA. Then GA operators will be applied to these parameters, and at the end of the GA run, the best coefficients are identified, as well as the prescribed problem solution.

The following technique is based upon the technique presented by Coello (1999). In this technique, two populations, P1 and P2, are used. The first population is to evolve solutions (as in a conventional GA) and the second is to evolve penalty factors. For each member of P2, an instance of P1 is used. The fitness of each member of P1 will be calculated as usual, and after a certain number of generations, an average fitness that considers the number of feasible solutions (count-feasible) and the average fitness value of the feasible solutions will be calculated. This average fitness will be used to evolve the penalty factors.

Coello (1999) proposed an equation to calculate the average fitness as follows:

$$average\ fitness_j = \sum_{i=1}^{M1} \left( \frac{Fitness(X)_i}{count\_feasible} \right) + count\_feasible \quad \forall X \in F \dots \dots \dots (5.5)$$

Coello pointed out that the average fitness should be scaled before adding to count-feasible. As the problem investigated in this study is a minimization problem (rather than maximization as in the work of Coello), and to avoid the scaling of count-feasible, Equation 5.5 is modified in this study as follows:

$$average\ fitness_j = \sum_{i=1}^{M1} \left( \frac{Fitness(X)_i}{count\_feasible - 1} \right) \quad \forall X \in F \dots \dots \dots (5.6)$$

If count-feasible equals 1, the average fitness is set to  $1.5 * Fitness(X)_i$ , and if count-feasible equals 0, the average fitness is set equal to the maximum (worst)

fitness obtained during the sub-generations. This will give an advantage to the penalty factors that result in more feasible solutions, as the average value will be less than a set of penalty factors that gets the same average from fewer feasible solutions. In this study, the penalty function given in Equation 5.2 is used to calculate the fitness values of each member of P1.

#### 5.2.4 Stochastic techniques

The idea of using stochastic techniques is to avoid the fine-tuning required by penalty functions. Among the stochastic methods, Surry and Radcliffe (1997) presented the COMOGA method that combine multiobjective optimization with stochastic selection, and it works as follows:

- Calculate constraint violations for all solutions.
- Pareto rank based on constraints violations.
- Evaluate the cost of solutions.
- Select a portion of parents  $p_{cost}$  based on the cost, and the rest based on constraints ranking.
- Apply genetic operators (crossover, mutation)
- Adjust  $p_{cost}$  if the proportion of the feasible solution in the population is not close to the target proportion.

Another approach was proposed by Runarsson and Yao (2000). This technique was presented in section 2.5.3 among selection techniques. In this technique, the authors compare all solutions in order to rank them. This comparison is made for N times, and during any time, if there is no change in the ranking, the procedure will stop. The rank is made based on the objective value with probability  $p_f$  or when both solutions are feasible, otherwise they rank based on constraints violations. They suggested the number of comparisons N to be equal to the population size, and  $p_f$  to be between 0.4 and 0.5. They used this technique with an evolution strategy.

Another technique using stochastic tournament selection is proposed in this study, and is described in the following subsection.

#### 5.2.4.1 Stochastic Tournament Selection (STS)

This technique uses binary tournament selection, but instead of using the fitness value (a combination between the objective and the constraints), it will select based on only one of them as follows:

The technique considers the following points:

- The difference between the objective values and constraint violations of the two solutions. If the difference between the costs of the solutions is big, and the difference between constraints violations of the solutions is small, it is better to use the objective values for the comparison to take advantage of this big difference at the expense of small constraint violation.
- The number of feasible solutions in the current generation. If the number of feasible solutions is small, it is better to encourage the model to make more copies of these feasible solutions during selection. If there are not any feasible solutions at all, more pressure will be added to select based on the constraint violations as a way of finding a feasible solution.
- Average improvement of both objective values and constraints violations in the last few generations. More pressure will be applied to the one that has less improvement in recent generations to prevent the model from diverging or converging to local optima.
- The selection will be done stochastically.

The technique works as follows:

- Primary probabilities for both objective and constraints are calculated as follows:

$$PP_o = \frac{C(I) - C(I+1)}{Obj\_Max\_Diff} * BF_o \dots \dots \dots (5.7)$$

$$PP_c = \frac{V(I) - V(I+1)}{Cons\_Max\_Diff} * BF_c * FF \dots \dots \dots (5.8)$$

Where

- $PP_o, PP_c$  Primary probabilities for the objective and constraints respectively.
- $C(I), C(I + 1)$  The cost and constraint violation values for both individuals in the binary tournament selection
- $V(I), V(I + 1)$
- $BF_o$  and  $BF_c$  Balance factors for both objective and constraints respectively.
- $FF$  Feasibility factor.

Balance factors are used to put more pressure for selecting based on the criteria that improved less in the previous generations. These balance factors are calculated as follows:

$$BF_J = 1.0 + \frac{\sum_{K=G1}^{G2} \frac{(Ave(K + 1) - Ave(K)) * K^2}{ABS(Ave(K))}}{\sum_{K=G1}^{G2} K^2} \dots\dots\dots(5.9)$$

Where J refers to the objective functions or the constraints.  $Ave(K)$  is the average of the objective values or constraints violation ratios during generation K.  $G1$  and  $G2$  refer to the first and last generation to be used in calculating the balance factors.  $G2$  is the generation just before the current generation, and  $G1 = G2 - UG$ , where  $UG$  is the user-specified number of generations used in calculating the factor.

- The feasibility factor can have one of the three following values:
  - ✗ If one of the individuals is feasible and the other is not, the primary probability for constraint violation ratios will be multiplied by the following feasibility factor:

$$FF = \sqrt{\frac{\text{Population}}{\text{Number of feasible solutions}}} \dots\dots\dots(5.10)$$



- ⌘ If there are no feasible solutions, all primary probabilities for constraint violation will be multiplied by  $\sqrt{\text{Gen}}$ , where Gen is the number of the current generation.
- ⌘ Otherwise  $\text{FF}=1.0$ .
- Both primary probabilities are normalized to defined the final probabilities as follows:

$$FP_o = \frac{PP_o}{PP_o + PP_c} \dots\dots\dots(5.11)$$

$$FP_c = \frac{PP_c}{PP_o + PP_c} \dots\dots\dots(5.12)$$

- Based on the final probabilities, one of the categories (cost or constraints violations) is selected stochastically, and the solution that performs better in this category is the winner.

### 5.2.5 Adaptive penalty function

This technique works as follows:

- A primary penalty coefficient for each constraint  $PPC(I)$  and an expected average violation  $EAV(I)$  associated with this coefficient are defined.
- The current penalty coefficient for each constraint during the current generation  $CPC(I)$  is calculated as follows:

$$CPC(I) = \frac{\sum_{J=1}^P \Phi(I, J)}{P} * \frac{PPC(I)}{EAV(I)} * \frac{BF_c}{BF_o} \dots\dots\dots(5.13)$$

Where

- $\Phi(I, J)$             The violation of the constraint I in the solution J.
- $BF_o$  and  $BF_c$        Same balance factors used in the STS technique (see Equation 5.9)
- $P$                         Population size.

This means that the current penalty coefficients are shifted linearly from the primary penalty coefficient based on the ratio between the actual average violation and expected average violation. The ratio of the constraint violations, instead of the amount of the violation, is used in this procedure.

The procedure will be as follows:

- During the generation, for each string in the population, the cost and the constraints violation ratio will be calculated.
- At the end of the generation, the current penalty coefficient for each constraint will be calculated, and the fitness equation is calculated for each solution.
- GA operators continue as usual.

### 5.3 Comparisons

There are seven techniques that were tested within the model, which are:

- Additive Static Penalty (ASP)
- Multiplicative Static Penalty (MSP)
- Additive Linear Dynamic Penalty (ALDP)
- Multiobjective technique (MO)
- Self-Adaptive (SA)
- Stochastic Tournament Selection (STS)
- Adaptive technique (AD)

The penalty function techniques and adaptive technique are tested twice, first with original tournament selection, and second with tournament selection with superiority of feasible solutions (note: in tables and charts, TS term is used to refer to original tournament selection, and SF term is used to refer to tournament selection with superiority of feasible solutions). The self-adaptive technique is used only with tournament selection with superiority of feasible solutions for the first scenario, due to the heavy computational time required by the technique. Thus, there are a total of 11 techniques for the first scenario, and 10 techniques for the second and third scenarios. Each of these techniques will be used with five different initial random seed values.

The GA parameters used with each scenario are presented in Table 5.1, based on the results presented in Chapter 4.

Table 5.1: GA parameters associated with each scenario

	<b>Population Size</b>	<b>Crossover Probability</b>	<b>Mutation Rate</b>	<b>Blend Crossover Extension</b>
<b>Scenario 1</b>	76	0.6	0.05	0.5
<b>Scenario 2</b>	76	0.6	0.05	0.5
<b>Scenario 3</b>	50	0.7	0.05	0.5

For STS and adaptive techniques, the effect of the previous 10 generations is considered.

Two measures are used to compare different techniques:

- Best feasible solution obtained during the whole run.
- The improvement of the minimum feasible solution during the run. The details about this measure were given in Chapter 4.

Four different tests are considered to compare different technique, which are:

- Best solution achieved by each technique: the best value obtained by each technique, considering both measurements, during each scenario is defined. The best value in the whole scenario is used as a reference, and the difference between this value and other values is calculated.
- Best and worst values: the best five values and the worst value, regarding both measures, are recorded, and the technique that produced each of them is noted.
- Comparing techniques with the same seed value: for each seed value, the technique that obtains the best value, regarding each measure, is recorded as a winner. The number of times each technique wins from the five seeds is defined.
- Means and standard deviations: from the 5 runs of each technique, and regarding both measurements, the mean and the standard deviation are calculated. Statistically, there was no significant difference between means, so a schematic drawing is drawn to represent the differences between different techniques. In both measurements, the difference between the best value in the whole scenario, and the best value obtained by each technique is used.

## 5.4 Results

The next paragraphs present the results for both measurements. A summary of these results is presented at the end.

### 5.4.1 Test 1

Table 5.2: The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 1

<b>Method</b>	<b>Scenario 1</b>	<b>Scenario 2</b>	<b>Scenario 3</b>
<b>STS</b>	248330	62332	0
<b>TS AD</b>	210970	0	168550
<b>SF AD</b>	9632	192396	262638
<b>MO</b>	83566	207050	184286
<b>TS ASP</b>	110042	22694	179364
<b>SF ASP</b>	75374	60304	646236
<b>TS MSP</b>	134308	131152	264920
<b>SF MSP</b>	131140	213116	264920
<b>TS ALDP</b>	83302	116726	181400
<b>SF ALDP</b>	43654	208320	357210
<b>SF SA</b>	0	--	--

Table 5.3: The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 2

<b>Method</b>	<b>Scenario 1</b>	<b>Scenario 2</b>	<b>Scenario 3</b>
<b>STS</b>	0.823	0.301	0
<b>TS AD</b>	1.028	0	0.534
<b>SF AD</b>	0.155	0.401	0.796
<b>MO</b>	0.179	0.523	0.575
<b>TS ASP</b>	0.230	0.149	1.020
<b>SF ASP</b>	0.009	0.414	1.814
<b>TS MSP</b>	0.376	0.341	0.813
<b>SF MSP</b>	0.292	0.403	0.813
<b>TS ALDP</b>	0.184	0.368	1.066
<b>SF ALDP</b>	0	0.326	1.246

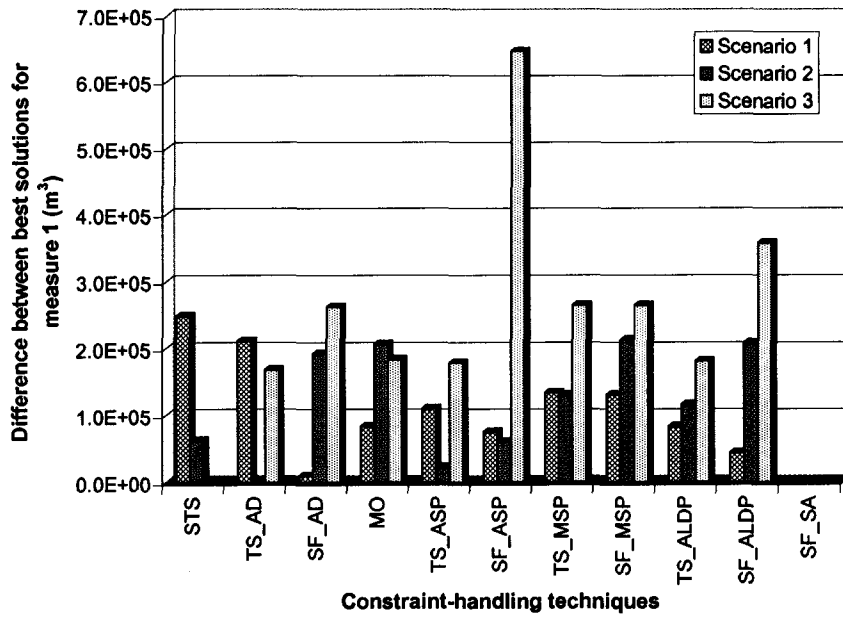


Figure 5.1

The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 1

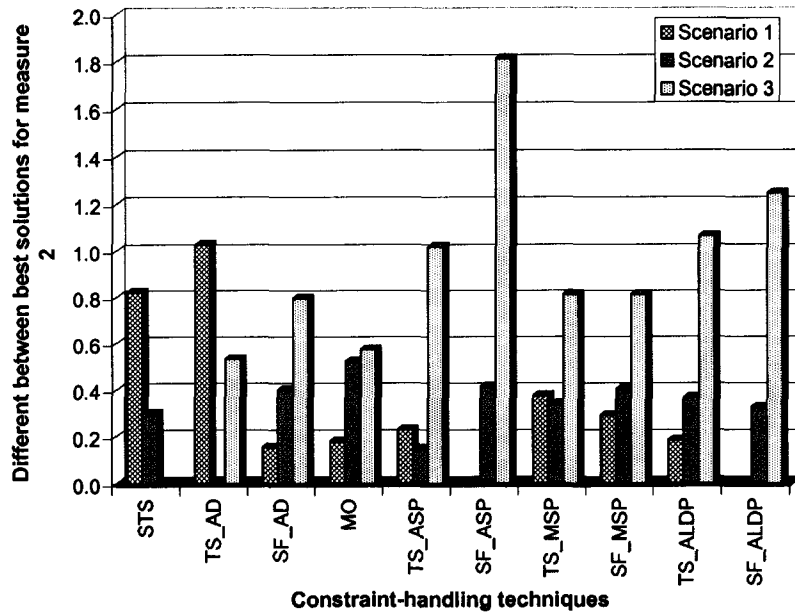


Figure 5.2

The total water consumed for the best run using each technique, relative to the best run overall, for each scenario for measure 2

From Figures 5.1 and 5.2, and Tables 5.2 and 5.3, the following points can be noticed:

- STS (Stochastic tournament selection) is the best technique for the third scenario for both measurements.
- TS\_AD (Adaptive technique that used original tournament selection) is the best technique for the second scenario for both measurements.
- For the first scenario, SF\_SA (Self-adaptive technique that used tournament selection with superiority of feasible solutions) is the best scenario in the first measurement, (note: it considered only in this measurement). For the second measurement, SF\_ALDP (Additive linear dynamic technique that used tournament selection with superiority of feasible solutions) is the best technique.
- Regarding techniques that are using two different selection methods, they work better with original tournament selection in second and third scenario, while working better with tournament selection with superiority of feasible solutions in the first scenario.
- The differences between techniques in the third scenario are more significant than the differences in the first two scenarios.
- SF\_ASP (Additive static technique that used tournament selection with superiority of feasible solutions) is least stable technique between different scenarios.
- MO (Multiobjective technique) is most stable technique between different scenarios.

#### **5.4.2 Test 2**

Tables 5.4 and 5.5 present the best five techniques and the worst technique for each scenario.

Table 5.4: Techniques that produce the five best values and the worst value regarding the first measure

Rank	Scenario 1	Scenario 2	Scenario 3
<b>First</b>	SF_SA	TS_AD	STS
<b>Second</b>	SF_AD	TS_ASP	STS
<b>Third</b>	SF_ALDP	TS_AD	STS
<b>Fourth</b>	SF_ALDP	TS_AD	TS_AD
<b>Fifth</b>	SF_ASP	SF_ASP	TS_ASP
<b>Last</b>	TS_AD	SF_ASP	MO

Table 5.5: Techniques that produce the five best values and the worst value regarding the second measure

Rank	Scenario 1	Scenario 2	Scenario 3
<b>First</b>	SF_ALDP	TS_AD	STS
<b>Second</b>	SF_ASP	TS_AD	STS
<b>Third</b>	SF_ASP	TS_AD	STS
<b>Fourth</b>	SF_AD	TS_ASP	TS_AD
<b>Fifth</b>	MO	STS	MO
<b>Last</b>	TS_AD	SF_ASP	SF_ASP

The following points can be noticed from these tables:

- Consistent with the first test, techniques perform better in the first scenario when they consider superiority of feasible solutions during the selection, while they perform better in second and third scenarios while they don't consider it.
- In the first scenario, SF\_SA (Self-adaptive technique that used tournament selection with superiority of feasible solutions) got the best optimal, and SF\_ALDP (Additive linear dynamic technique that used tournament selection with superiority of feasible solutions) is the technique that produced two from the best five.
- It is confirmed that TS\_AD (Adaptive technique that used original tournament selection) is the best in the second scenario, and STS (Stochastic tournament selection) is the best technique in the third scenario, as they got the most of the five best values in both measures.
- MO (Multiobjective technique) performs differently for the two measures. Although it has the worst value in the third scenario of the first measure, it got the fifth best in the first and third scenario of the second measure.

- SF ASP (Additive static technique that used tournament selection with superiority of feasible solutions) is the least stable technique as it showed up among the best values and as the worst value many times, as in the first test.

### 5.4.3 Test 3

Tables 5.6 and 5.7 present the numbers of times each technique wins compared with other techniques that have the same seed value for both measures.

Table 5.6: Number of times each technique wins from runs that have same seed value regarding the first measure

Method	Scenario	Scenario	Scenario
	1	2	3
STS	0	1	3
TS AD	0	3	1
SF AD	0	0	0
MO	0	0	1
TS ASP	0	1	0
SF ASP	1	0	0
TS MSP	0	0	0
SF MSP	0	0	0
TS ALDP	0	0	0
SF ALDP	2	0	0
SF SA	1	--	--



Table 5.7: Number of times each technique wins from runs that have same seed value regarding the second measure

Method	Scenario	Scenario	Scenario
	1	2	3
<b>STS</b>	0	1	3
<b>TS AD</b>	0	3	1
<b>SF AD</b>	2	0	0
<b>MO</b>	1	0	0
<b>TS ASP</b>	0	1	1
<b>SF ASP</b>	1	0	0
<b>TS MSP</b>	0	0	0
<b>SF MSP</b>	0	0	0
<b>TS ALDP</b>	0	0	0
<b>SF ALDP</b>	1	0	0

From the tables, it can be noticed that:

- The results for second and third scenarios are consistent with other tests.
- The results for first scenario are different than the first test. For example, although SF\_SA (Self-adaptive technique that used tournament selection with superiority of feasible solutions) is the best in the first measure, it wins only once in this measure. Similarly, although SF\_ASP (Additive static technique that used tournament selection with superiority of feasible solutions) is the best in the second measure, it wins only once in this measure.

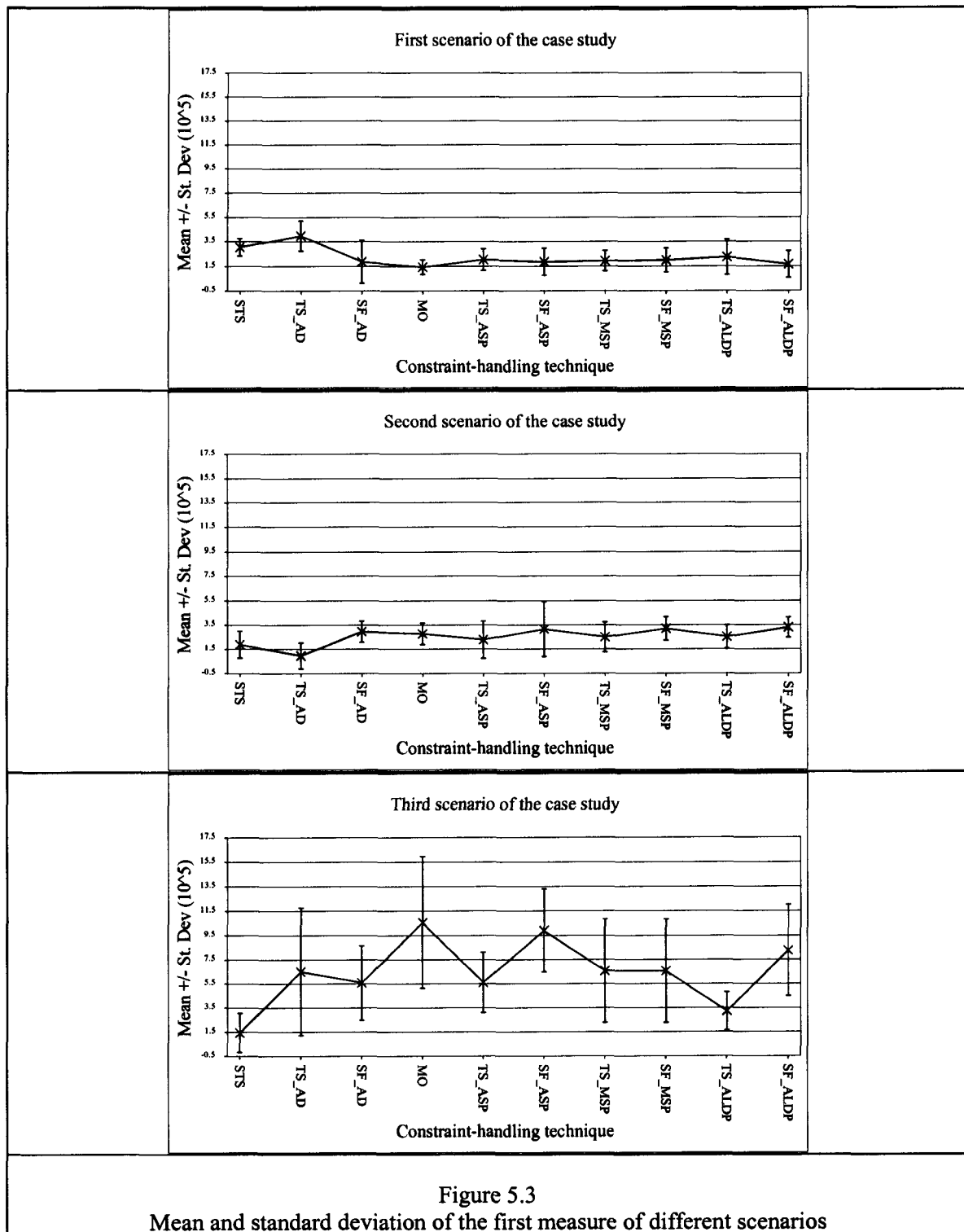
#### 5.4.4 Test 4

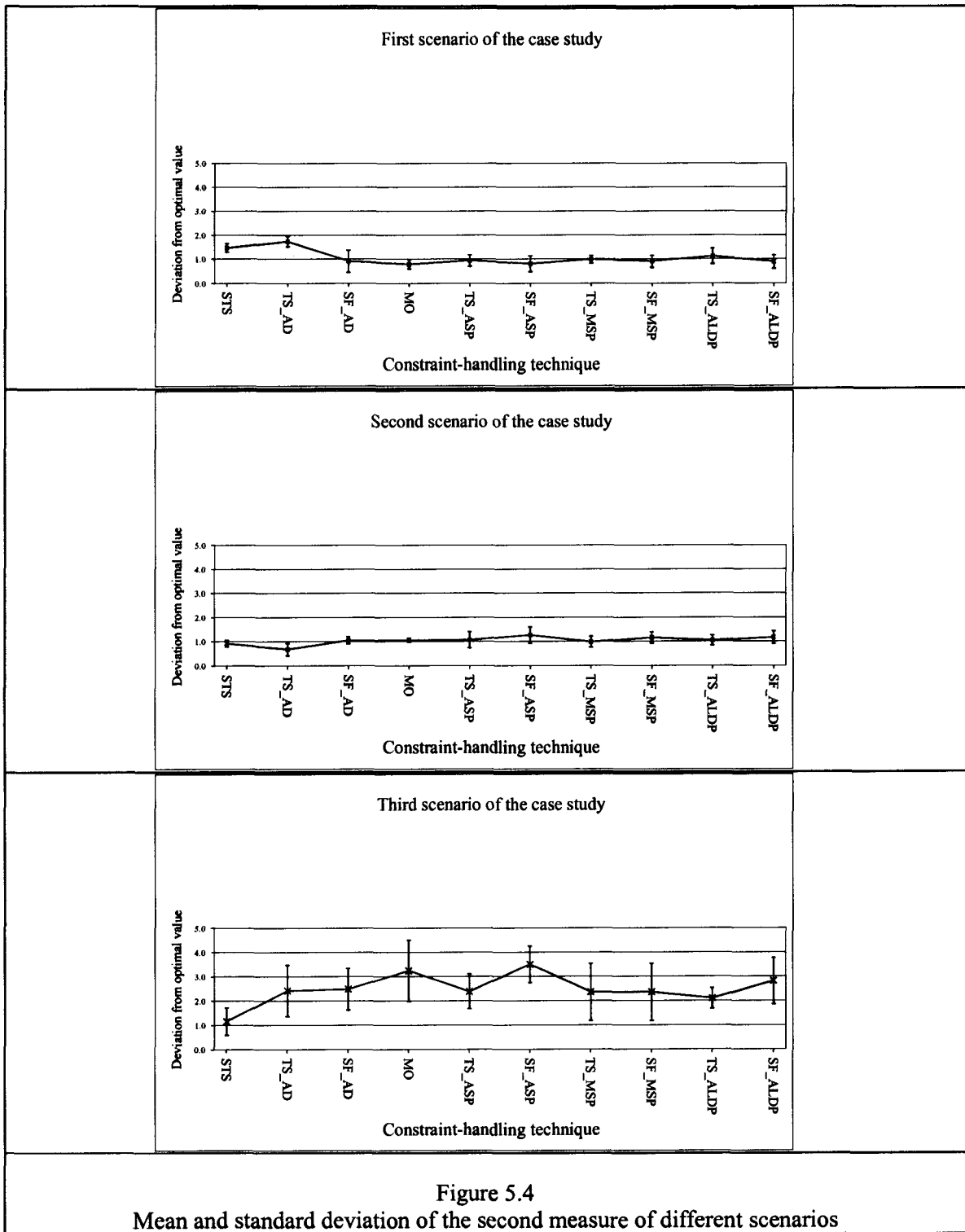
Figures 5.3 and 5.4 present the mean and standard deviation of the difference between the best value in the scenario and all other values for the five runs of each technique. From these charts, it can be noticed that:

- In the first scenario, and considering both measures, MO (Multiobjective technique) is the best candidate followed by SF\_ALDP (Additive linear dynamic technique that used tournament selection with superiority of feasible solutions). The difference between them in the mean value is small, but MO is more consistent in both measures. The small values of standard deviation of MO in both

measures might explain why the technique has the best mean in this test, while it didn't give good results in the previous tests. The worst techniques are TS\_AD (Adaptive technique that used original tournament selection) and STS (Stochastic tournament selection). Except for the worst two techniques, the difference between different techniques is not significant. Also in general, the techniques perform better while considering superiority of feasible solutions during the selection. This is more clear for the adaptive technique than for other techniques.

- In the second scenario, TS\_AD (Adaptive technique that used original tournament selection) is the best technique, followed by STS (Stochastic tournament selection), in both measures, which is consistent with previous tests. In general, techniques perform better when they don't consider superiority of feasible solutions during the selection. This is more clear in adaptive technique than other techniques. The difference between different techniques is more significant than the first scenario.
- In the third scenario, the difference between techniques is more significant than for the first two scenarios. In this scenario, STS (Stochastic tournament selection) outperforms all other techniques, followed by TS\_ALDP (Additive linear dynamic technique that used original tournament selection), in both measures. Also, in general, techniques perform better when they don't consider superiority of feasible solutions during the selection, as in the second scenario. The multiobjective technique is the worst technique for this scenario.





### 5.4.5 Summary

The previous results can be summarized as follows:

- For first scenario, techniques that consider superiority of feasible solutions during the selection, including multiobjective technique, outperform other techniques. The multiobjective technique is the technique that is the most consistent when using different seed values regarding this scenario.
- TS\_AD (Adaptive technique that used original tournament selection) outperforms all other techniques in the second scenario. Also in general, techniques perform better for the second scenario when they don't consider superiority of feasible solutions during the selection.
- STS (Stochastic tournament selection) outperforms all other techniques in the third scenario. The difference between STS and other techniques in this scenario is more significant than the differences between different techniques in the first two scenarios.

## 5.5 The performance of STS technique

Considering the results that were displayed in the last part, STS performed well in third scenario, somewhat well in the second scenario, and poorly in the first scenario (the simplest example). In this section an attempt is made to explain the reason behind that, highlighting the characteristics of the technique. Also, suggestion are made to examine if the results obtained by the technique is good or bad, given that the optimal value is usually not known in such problems, and there are no other runs to compare with.

It seems that the technique works well when its average probabilities of objective function value and constraints are interacting with each other around the value of 0.5. This is explained in the following paragraph. It should be mentioned that this value is related to the average of probabilities of all solutions, not the probability for each comparison.

Figure 5.5 presents the population average probability of using objective function value and constraints as the basis for selection of STS for a specific initial seed value with which the technique performs poorly in the first scenario, and performs well in the third scenario. In the first scenario, and for the first four generations, the probabilities were around 0.5, and constraints probability is higher. After this generation, the objective function value probability increases, and the constraints probability decreases with a diverging pattern. This means the selection is made mainly based on the objective function values without paying enough attention to the constraints violations. For the third scenario, both probabilities are fairly close to 0.5, and they alternate which is smaller and which is higher.

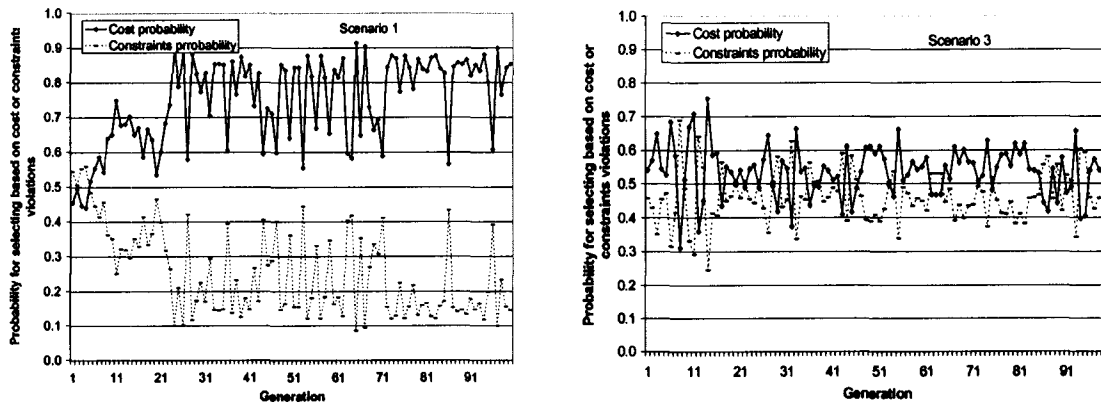


Figure 5.5  
Average STS probabilities for a specific initial seed value in the first and third scenarios.

The reason for this could be obtained from Figures 5.6 and 5.7. Figure 5.6 presents the average constraint violations and the ratio of feasible solutions per generation for each of the same two runs. From figure 5.6, the average constraint violations of the first scenario decreases suddenly and the number of the feasible solutions increases suddenly after a few generations in the first scenario given that the problem is simple, and it is easy to find feasible solutions. In the third scenario, the constraint violations decrease gradually, and number of feasible solutions increases gradually during generations. At the end of the run, third scenario performs better than the first scenario regarding ratio of the feasible solutions.

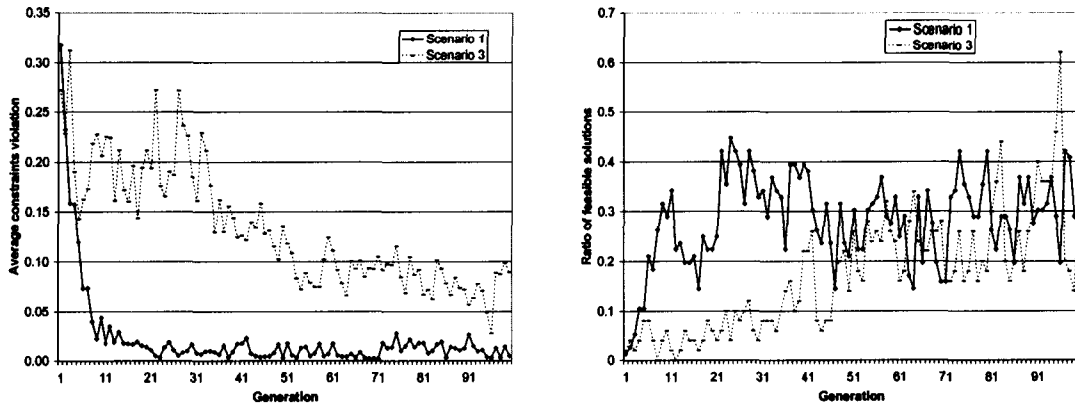


Figure 5.6

Average constraint violations and the ratio of feasible solutions for scenarios 1 and 3.

Figure 5.7 presents the sorted values of the objective function value and the constraint violation as a ratio of the maximum value for both scenarios in the ninth generation.

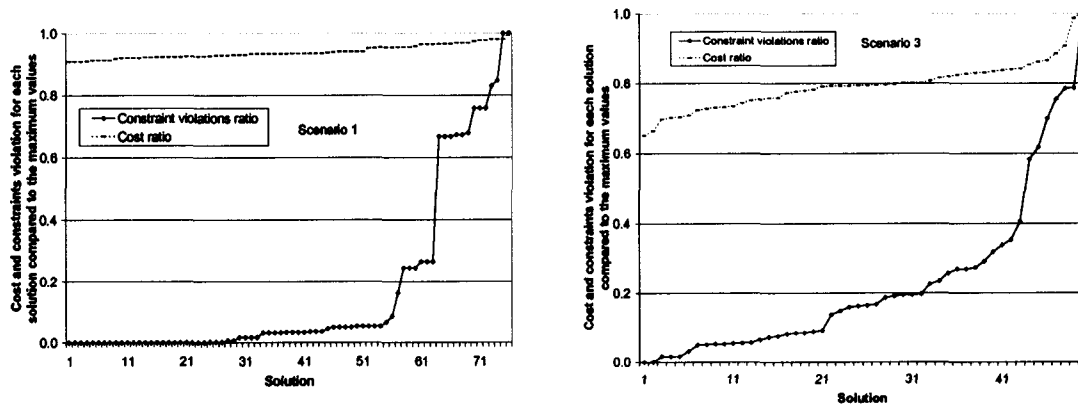


Figure 5.7

Sorted values of the objective function value and the constraints violation as a ratio of the maximum value for first and third scenarios in the ninth generation

From Figure 5.7, recalling the equations of the technique, and considering the first scenario, the maximum difference for constraints is big, while many value are close to each other (eg., values between solutions 35 and 55). Considering that the number of feasible solutions is big and the improvement in the previous generations is high, the final average probability for constraints is expected to be small value, and thus the average probability for cost is high value. In the third scenario, the constraint

violation values are changing gradually, the maximum difference for the objective function value is relatively bigger than for the first scenario, the number of the feasible solutions is small, and the improvement in the previous generation is moderate, and so the constraints average probability is not so small.

As a result of selecting based on the objective function value without paying attention to the constraint violations in the first scenario, the minimum feasible solution doesn't increase gradually after the first few generations, and it reached the minimum value at generation 4, which is not a good value compared to other runs in the same scenario (see Figure 5.8). In the third scenario, the minimum feasible solution keeps decreasing gradually during generations until the end, and it reaches the minimum value at generation 94, which is a good value compared to other runs in the scenario.

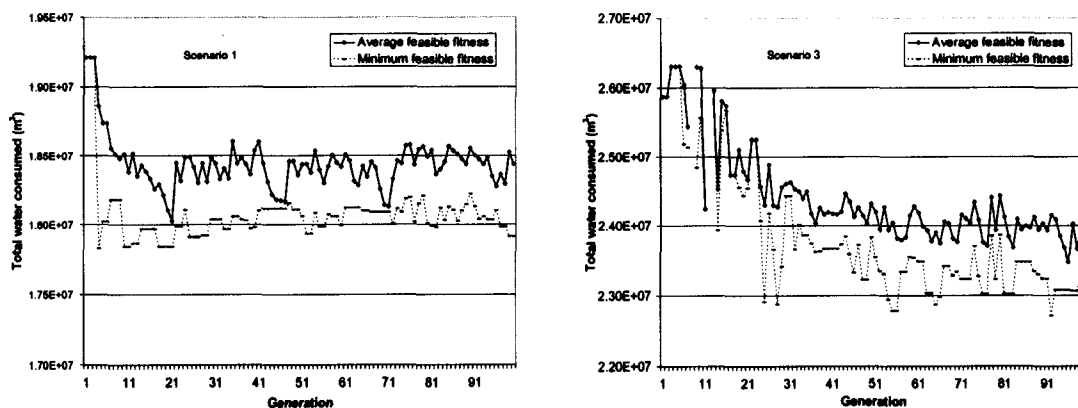


Figure 5.8  
Minimum and average feasible solutions for first and third scenarios

To provide further evidence that the STS technique performs the best when both average probabilities are close to 0.5, and exchanging their positions about which is higher and which is smaller, another two examples from the second and the third scenarios are presented in Figures 5.9 and 5.10. From both figures, considering that the technique performs the best in the third scenario, while performs somewhat well in the second scenario, it could be noticed that the best run is associated with the average probabilities closer to 0.5 in both cases.



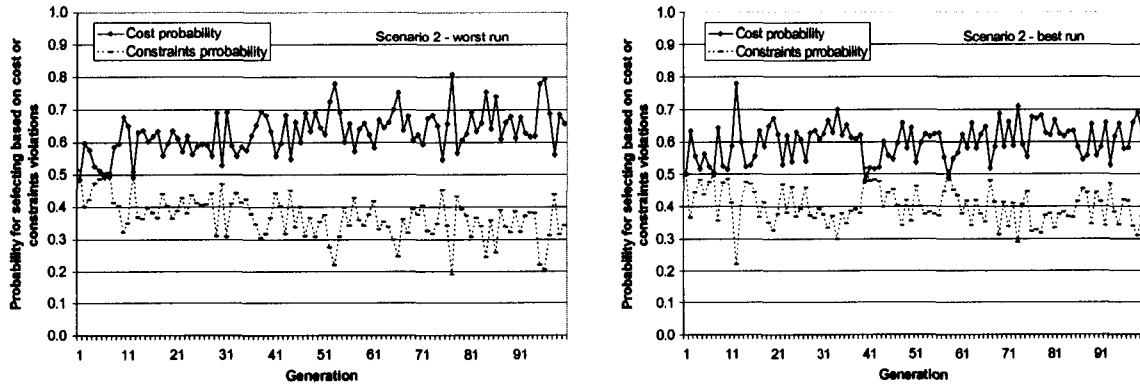


Figure 5.9  
Average STS probabilities for the worst and best runs of the second scenario.

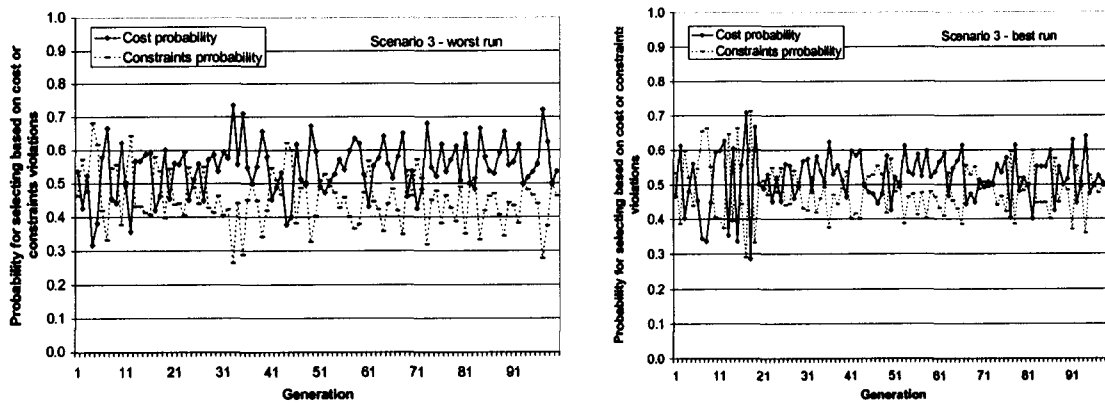


Figure 5.10  
Average STS probabilities for the worst and best runs of the third scenario.

Although the previous explanation proves that the technique may converge to a local optimal, it also shows that watching the average probabilities of the technique, which is available through the model, could be used as an indicator of the quality of the results, given the optimal value is normally not known, and the user will not try different techniques to select from. So, whenever both probabilities are close to 0.5, interacting and exchanging their positions, it could be expected that the results have good quality. Whenever one of probabilities denominates the selections, it could be expected that the results are poor, and it might be better for the user to switch to another technique.

## 5.6 The performance of adaptive technique

In this section, it is investigated whether maintaining the balance between the improvement of both cost and constraint violations could be used as the sign for the quality of this technique, as the average probabilities are in STS technique. The assumption is that the technique works better if  $BF_C / BF_0$  is moving around 1.0 or closer to it. For each scenario and for the same initial seed value, the results of both selection methods (to assume selection with and without superiority of feasible solutions) are considered.

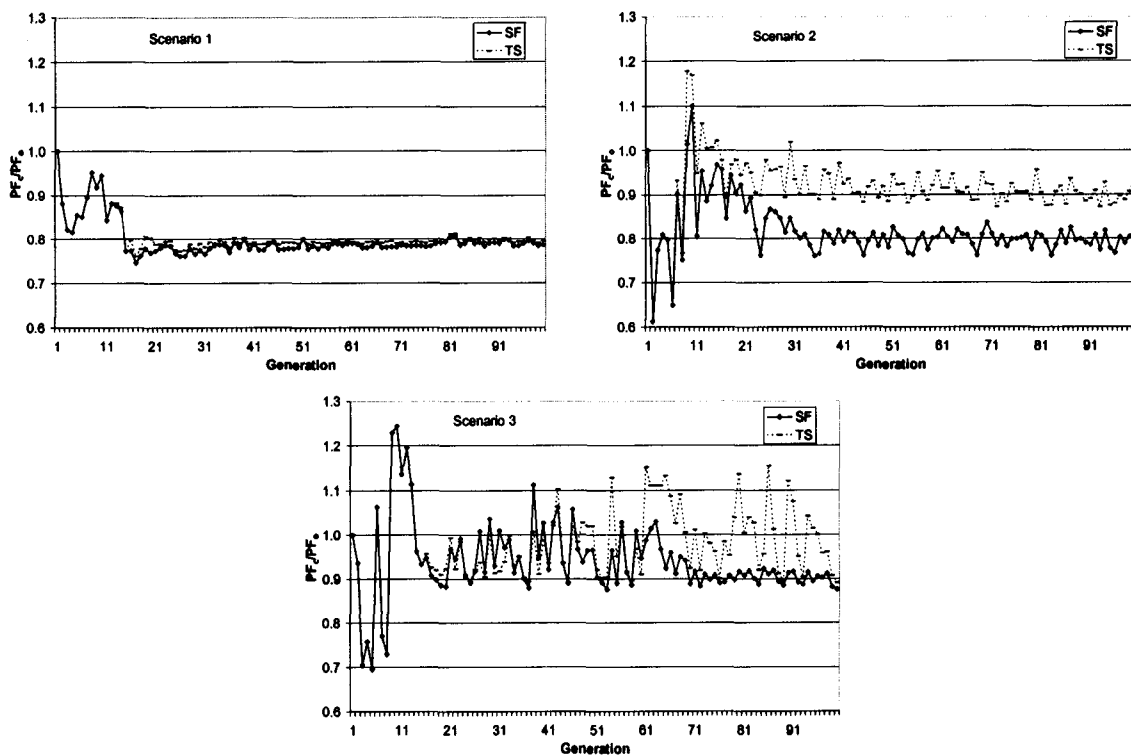


Figure 5.11

The value of  $BF_C / BF_0$  for the three scenarios for adaptive technique

From Figure 5.11, and given that the technique works better in the first scenario while consider superiority of feasible solutions (SF), and works better with other selection method in the second and third scenario, it could be noticed from Figure 5.12 that the results for the second and third scenarios support the assumption, while

it doesn't support it in the first scenario. Also in the second scenario, the better results are closer to 1.0, but it doesn't reach it.

## 5.7 Conclusions and future work

From the results, the following points can be noticed:

- For self-adaptive technique, the large computational time requirement is not acceptable, especially since its solution quality is not superior to other methods.
- Techniques perform differently from one scenario to the other, and with different selection methods.
- For simpler problems with relatively few decision variables, such as the first scenario, the multiobjective technique and penalty techniques that support superiority of feasible solutions perform better. Among these solutions, multiobjective seems to be the most consistent. It has the smallest standard deviation using different seed values, and since it does not require any fine-tuning, it is the most recommended technique for such cases of the model.
- For harder scenarios with many decision variables, such as the second and third scenarios, STS (Stochastic tournament selection) and penalty techniques that don't support superiority of feasible solutions during selection perform better. STS performed the best for the third scenario, and TS\_AD (Adaptive technique that used original tournament selection) performed the best for the second scenario.
- A suggestion was made to check the quality of the output of STS method in the absence of the optimal value, and when there are no other runs to compare with.
- Another suggestion was made for the adaptive technique, but it is not supported by all scenarios.
- Since in reality, it is hard to determine the difficulty of the problem that is being analyzed, it is preferable for the user to use STS method, and if its output doesn't show evidence that the result is likely to be of good quality, the user could switch to any other method that didn't support superiority of feasible solution. TS\_AD is the good alternative in this case.

Suggested future work:

- STS method should be tested with different type of problems to check its ability to handle the constraints in different situations.
- The STS technique should be re-studied regarding maintaining the balance between the probabilities for the constraints and the cost, and how this could prevent the technique from converging to local optima or diverging.

## 5.8 References

- Arabas, J., Michalewicz, Z., and Mulawka, J., "GAVaPS – a Genetic Algorithm with Varying Population Size," *Proceeding of the first IEEE Conference on Evolutionary Computation*, IEEE Press, 1994, pp. 73-78
- Bean, J. C., and Hadj-Alouane, A. B., "A Dual Genetic Algorithm for Bounded Integer Programs" *Department of Industrial and Operations Engineering*, The university of Michigan, TR 92-53, 1992.
- Coello, C. A. C., "Self-Adaptive Penalties for GA-based Optimization," *Proceedings of the Congress on Evolutionary Computation 1999 (CEC'99)*, Piscataway, New Jersey, Vol. 1, pp. 573-580, July 1999.
- Coello, C. A. C., "Constraint-Handling Using an Evolutionary Multiobjective optimization Technique," *Civil Engineering and Environmental Systems*, Vol. 17, pp. 319-346, 2000.
- Deb, K. and Goel, T., "Controlled Elitist Non-dominated Sorting Genetic Algorithm for Better Convergence" *KanGAL Report No. 200004*, 2000
- Deb, K., *Multi-objective optimization using evolutionary algorithms*, John Wiley & Sons, Ltd, 2001
- Eiben, A.E., Hinterding, R., and Michalewicz, Z., "Parameter Control in Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 2, pp.124-141, 1999
- Harrell, L. J. and Ranjithan, S. R., "Evaluation of Alternative Penalty Function Implementations in a Watershed Management Design Problem," *GECCO-99*:

- Proceedings of the Genetic and Evolutionary Computation Conference, 1999*, pp. 1551-1558.
- Homaifer, A., Lai, S. H., and Qi, X., "Constrained Optimization via Genetic Algorithms, Simulations, Vol. 62, No 4, 1994, pp. 242-254
- Janikow, C., and Michalewicz, Z., "An Experimental Comparison of Binary and Floating Point Representation in Genetic Algorithms," Proceeding of the fourth international conference on genetic algorithms, Morgan Kaufmann, 1991, pp. 151-157
- Le Riche, R., Knoop-Lenoir, C., Haftka, R. T., "A Segregated Genetic Algorithm for Constrained Structural Optimization" *Proceeding of the Sixth ICGA*, Morgan Kaufmann, 1995, pp. 558-565
- Michalewicz, Z., and Attia, N., "Evolutionary Optimization of Constrained problems" *Proceedings of the third Annual Conference on EP*, World Scientific, 1994, pp. 98-108
- Michalewicz, A., Dasgupta, D., Le Riche, R., and Schoenauer, M., "Evolutionary Algorithms for Constrained Engineering Problems," *Computers and Industrial Engineering Journal* Vol. 30, No 4, SEP 1994, pp. 851-870
- Michalewicz, Z., "A Survey of Constraint Handling Techniques in Evolutionary Computation Methods" *Proceedings of the fourth Annual Conference on Evolutionary Programming*, MIT Press, Cambridge, MA, 1995, pp. 135-155.
- Michalewicz, Z., and Xiao, J., "Evaluation of Paths in Evolutionary Planner/Navigator" *Proceeding of the 1995 International Workshop on Biologically Inspired Evolutionary Systems*, Tokyo, Japan, 1995, pp. 45-52
- Michalewicz, Z., and Schoenauer, M. "Evolutionary Algorithms for Constrained Parameter Optimization Problems" *Evolutionary Computation Journal* vol. 4, No 1, 1996, pp. 1-32
- Michalewicz, Z., Deb, K., Schmidt, M., and Stidsen, T., "Test-Case Generator for Nonlinear Continuous Parameter Optimization Techniques" *IEEE Transactions on Evolutionary Computation* Vol. 4, No 3, 2000, pp. 197-215

- Powell, D. and Skolnick, M.M., "Using Genetic Algorithms in Engineering Design Optimization with Non-linear Constraints" *Proceedings of the Fifth ICGA*, Morgan Kaufman, 1993, pp. 424-430
- Richardson, J. T., Palmer, M. R., Liepins, G., and Hilliard, M. "Some Guidelines for Genetic Algorithms with Penalty Function", *proceedings of the third international Conference on Genetic Algorithms*, 1989, pp. 191-197
- Runarsson, T. P., and X. Yao, "Stochastic Ranking for Constrained Evolutionary Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 4, SEP 2000, pp. 284-294
- Siedlecki, W., and Sklanaki, J., "Constrained Genetic Optimization via Dynamic Reward-Penalty Balancing and its Use in Pattern Recognition", *proceedings of the third international Conference on Genetic Algorithms*, 1989, pp. 191-197
- Srinivas, N. and Deb, K., "Multiobjective function optimization using nondominated sorting genetic algorithms", *Evolutionary Computation Journal*, Vol. 2, No 3, 1995, pp. 221—248
- Surrym P.D., and Radcliffe, N.J., "The COMOGA method: Constrained Optimization by Multiobjective Genetic Algorithms" *Control and Cybernetics*, Vol. 26, No 3, 1997

## CHAPTER 6

# GENERATING MORE RELIABLE MANAGEMENT STRATEGIES UNDER CONDITIONS OF UNCERTAINTY

### 6.1 Introduction

Defining the crop patterns in an irrigation canal network to simulate future conditions is associated with a significant amount of uncertainty. Additionally, the water demand per unit area of each crop varies with time and space due to changes in conditions such as temperature, soil characteristics, and the farmers' actions. The solution prescribed by a deterministic model may not perform well when evaluated under conditions of uncertainty. To address this, the deterministic model should be extended to incorporate estimates of reliability in the search procedure to identify more robust solutions under conditions of uncertainty. Incorporating uncertainty in hydraulic engineering began in the 1970's. The pioneer works considered the parameters ambiguity in the search space while designing hydraulic structures (Yen and Ang, 1971; Mays, 1979; Tung and Mays 1982). Regarding water distribution systems, most of the works were related to pipeline distribution systems. Among these works, Lansey et al. (1989) used a chance-constrained formulation to determine the least cost water distribution network, considering uncertainty in water demands, required pressure heads, and pipe roughness coefficients. Regarding irrigation canal networks, Molden et al. (1989) incorporated the hydrologic and management uncertainty in the hydraulic design of an open-channel irrigation system. Gates et al. (1992) extended this work by incorporating the hydraulic as well as hydrologic and management uncertainties in the optimal design of hydraulic structures. Uncertainty has been incorporated within genetic algorithm frameworks in application to many hydraulic fields such as groundwater (Ritzel et al., 1994; Cieniawski et al., 1995; Chan-Hilton and Culver, 2000) and watershed management (Harrell, 2001)

Because a GA evaluates the fitness of each potential solution, it is a straightforward extension to evaluate each solution repeatedly using a set of realizations of uncertain parameters generated based upon their probability distributions. The ratio of the number of realizations for which a criterion is satisfied to the total number of realizations for which the potential solution is evaluated provides an estimate of its reliability, which is included in the model as an additional constraint. Such a framework is referred to as a chance-constrained genetic algorithm, or CCGA. CCGAs have been implemented using 200 Monte-Carlo realizations to evaluate each potential solution's reliability (Ritzel et al., 1994; Cieniawski et al., 1995). In the current study, considering the heavy computational time required, it would be prohibitive to work with such a large number of realizations. Some research has been performed to investigate ways to reduce the computational time required for successful CCGA implementation. Loughlin and Ranjithan (1999) investigated various MC sampling strategies for a chance-constrained air quality management problem, with promising results for reducing the computational burden by using much smaller MC sample sizes. Latin Hypercube Sampling provides a good alternative for Monte Carlo, and can yield good results with fewer realizations.

## **6.2 Crop data and uncertainty**

In an irrigation network such as the one presented in the case study, where the cultivated land consists of many parcels owned by a large number of people, defining deterministic values for crop pattern is a difficult issue, as these values are always associated with a considerable level of uncertainty. Water consumption rates for different crops also vary over time and space. To account for this, probability distributions for both water consumption rate and crop pattern should be defined and used instead of deterministic values. In the current study, probability distributions for the crop pattern and water demands were calculated based on some studies that were conducted to optimize the crop allocation and water productivity in Egypt, as well as some studies that estimated the water demand rate for various crops (Ali, A. S., 1999; Ali, H. M. 2000; Fawzy, G. M. 1999; El Qu soy, D. 1995), and from information



provided by the irrigation and agriculture directorates in Egypt. In the absence of better information, uniform probability distributions were assumed for both crop pattern and water demand rate, where the upper and lower bounds for each probability distribution were defined based on the reported data. Seven seasonal crops and one permanent crop (gardens) were considered. Regarding crop pattern, after randomly selecting a ratio of each crop at each reach, the values of all crop patterns in each reach will be normalized, so the summation of them is unity.

Figures 6.1 and 6.2 present the upper and lower limits for water consumption rates and crop pattern for different crops used in the case study.

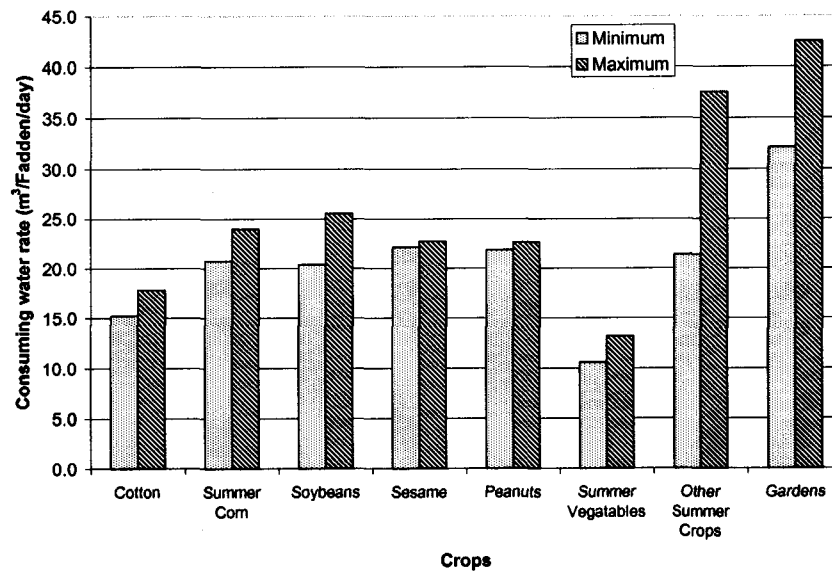


Figure 6.1

Upper and lower bounds for water consumption rates for different crops used in the case study

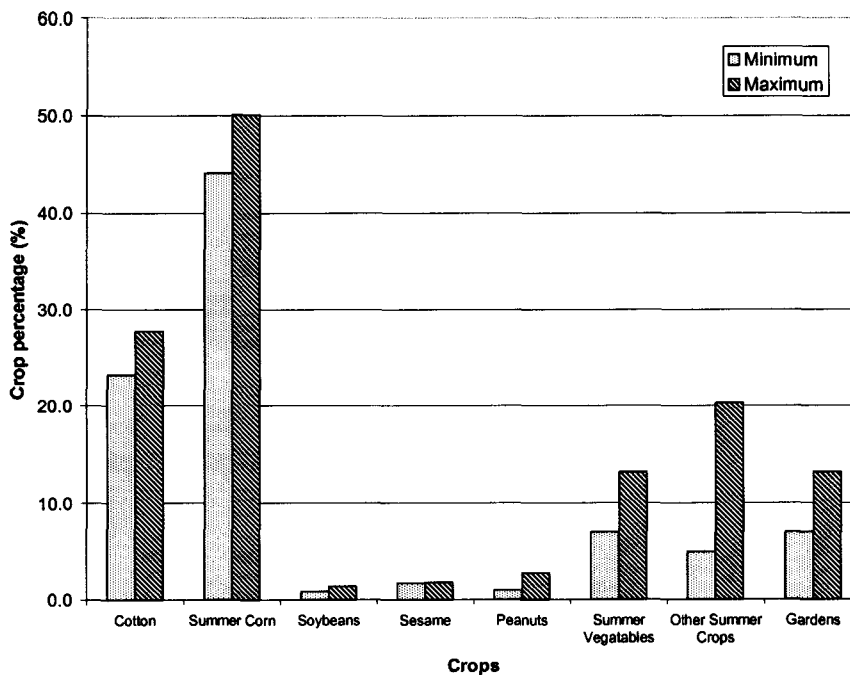


Figure 6.2

Upper and lower bounds for crop percentages for crops used in the case study

### 6.3 Chance-constrained technique

Stochastic programming is an optimization technique in which the constraints or objective function of an optimization problem contain stochastic parameters. Chance-Constrained Programming (CCP) is one method of stochastic programming that attempts to treat optimization problems with uncertain constraints. The name “chance-constrained” follows from the fact that each constraint is realized with a minimum probability of  $1 - \alpha_i$ , where  $0 < \alpha_i < 1$  (Taha, H. A., 2003). So, in CCP, instead of satisfying the constraints under the deterministic, or average conditions, the goal is to provide some confidence level of satisfying the constraints under conditions of uncertainty. Each constraint can only be violated for a fraction  $\alpha$  of the realizations, and the value  $(1-\alpha)$  is called a reliability target or safety margin, which is defined by the user.

For example, the following deterministic constraint:

$$\sum_{j=1}^n a_{ij}x_j \leq b_j \quad \forall i \text{ and } j \dots\dots\dots(6.1)$$

where  $a_{ij}$ ,  $b_j$  are deterministic values will be modified to be:

$$P\left\{\sum_{j=1}^n a_{ij}x_j \leq b_j\right\} \geq 1 - \alpha_i \quad \forall i \text{ and } j \dots\dots\dots(6.2)$$

where  $a_{ij}$ ,  $b_j$  or both are random values. This chance-constrained technique requires generation of random samples, and there are two techniques that can be used for this: Monte Carlo Sampling and Latin Hypercube Sampling. The details of each are described in the following section.

### 6.4 Generating sampling (MCS vs. LHS)

One of the basic steps for chance-constrained programming is to generate random samples for realizations of uncertain parameters. Monte Carlo Sampling (MCS) is the conventional method for generating random samples. Generating a sample using Monte Carlo (Figure 6.3) is done as follows:

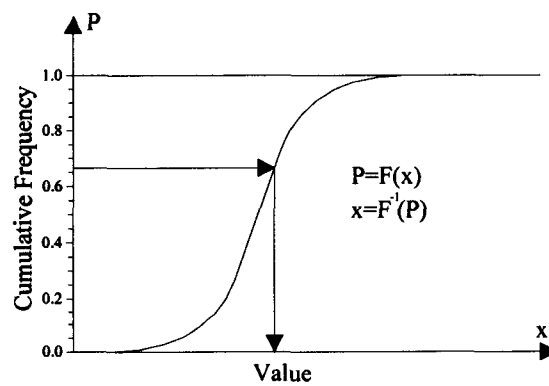


Figure 6.3  
Monte Carlo Sampling

- Generate the Cumulative Distribution Function (CDF) for the random variable.

- Generate a random number between 0 and 1 using any random number generator.
- Read the quantile associated to that random number.
- Repeat many times and check the percentage that satisfies the conditions.
- Check this percentage against target reliability.

Monte Carlo sampling requires using a large number of realizations to achieve good results.

Latin Hypercube Sampling (LHS) is a good alternative to the Monte Carlo sampling technique that can achieve good results with fewer realizations. Based on Wyss and Jorgensen (1998), the procedure works as follows:

- Divide the range of each variable into  $n$  non-overlapping intervals on the basis of equal density. (Examples of dividing variables with normal distribution probability and uniform probability are presented in Figure 6.4).
- One value from each interval is selected with respect to the probability density in the interval.
- The  $n$  values obtained from the variable X1 are paired in a random manner with  $n$  values of variable X2.
- These  $n$  pairs are combined with  $n$  values of variables X3 to form  $n$  triplets, and so on.

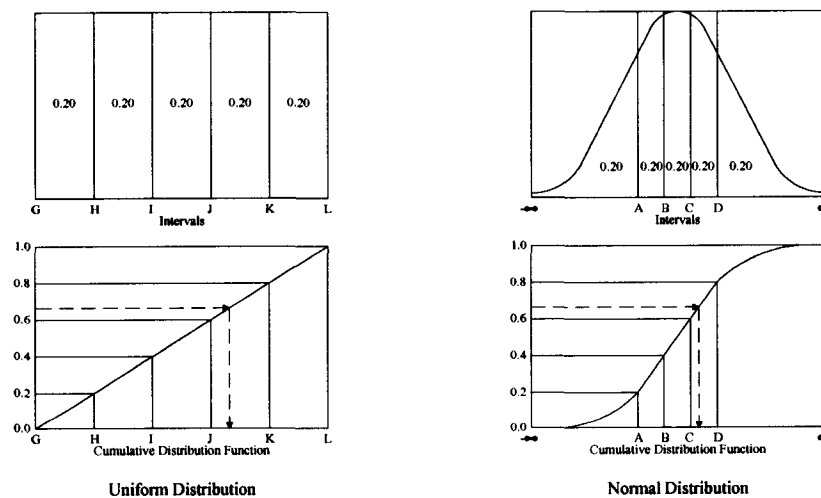


Figure 6.4  
LHS Sampling for uniform and normal distribution

Figure 6.5 illustrates an example of two variables with five intervals (n=5) for each. Each class of each variable is selected only once.

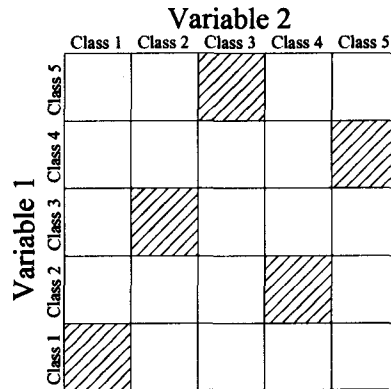


Figure 6.5  
An example of LHS Sampling

### 6.5 Chance-Constrained Genetic Algorithm (CCGA) model

In the CCGA model, the deterministic GA model is modified to incorporate estimates of likelihood of satisfying the constraints under conditions of uncertainty. Thus, the fitness equation includes additional penalty terms for each reliability constraint. In this application, a new set of realizations is generated for the evaluation of each string. Latin Hypercube Sampling is used to sample the data for each realization, where the probability distribution function for each uncertain variable is divided into certain number of classes. The fitness equation for each string is affected by the percentage of the realizations that satisfy each constraint. If the percentage of realizations satisfying the constraint is less than the target confidence level for this constraint, the fitness is penalized. The new penalty terms for the reliability constraints are calculated as follows:

$$CLS_i = A * \left[ 1 - \left( \frac{R - r_{ni}}{R} \right) \right]^2 \quad \text{if } (r_{ni} < (1 - \alpha)_i * R) \dots \dots \dots (6.3)$$

or

$$CLS_i = 1.0 \quad \text{if } (r_{ni} \geq (1 - \alpha)_i * R) \dots \dots \dots (6.4)$$

Where:

$CLS_i$	The penalty function for constraint $i$ when the number of realization that satisfy the required confidence is less than the target value.
$A$	A constant.
$R$	Total number of realizations.
$(1 - \alpha)_i$	Target reliability for constraint $i$ .
$r_{ni}$	Number of the realizations that satisfy the constraint.

To apply the penalty, the fitness value of each string will be divided by  $CLS_i$ .

The procedure to implement the CCGA model in the current study is as follows:

- Third scenario of the case study (see section 2.5.2) was selected as a case study for the uncertainty runs.
- The best deterministic solution for the third scenario is noted.
- The same GA parameters that were used with the third scenario in Chapter 5 will be used with it here. Also, STS (Stochastic tournament selection) will be used as the constraint-handling technique, as it is the technique that resulted in the best solution for the third scenario.
- This best solution was run using the unsteady flow model, using the average data for the water consumption rate and crop pattern. The objective (total water consumed) obtained from this run is used as reference to compare the results when uncertainty is incorporated into the search procedure.
- Then, the best solution was run for 1000 Monte Carlo samples, using the uniform probability distribution functions for water consumption rate and crop pattern.
- The results of satisfying the constraints in the Monte-Carlo simulation are shown in Table 6.1.
- The goal is to increase the reliability for satisfying the water shortage and required water constraints to the target level shown in Table 6.2

Table 6.1: Reliability level obtained from 1000 Monte-Carlo samples

<b>Constraints</b>	<b>Reliability level (%) Obtained from Monte Carlo runs</b>
Water shortage	61.9
Flood	100.0
Regulators stability	100.0
Required water levels	69.8

Table 6.2: Target reliability level

<b>Constraints</b>	<b>Target Reliability level (%)</b>
Water shortage	90.0
Flood	100.0
Regulators stability	100.0
Required water levels	100.0

The procedure for this part of program is as follows:

- A. The total number of realizations (R), and the target reliability are defined.
- B. The cumulative values for cost, constraint violations, and Reliability Satisfaction (RS) are initialized to 0.0.
- C. Water consumption rates and crop pattern are defined randomly for each reach.
- D. The cost (total water consumed) and the constraint violations for each realization are determined.
- E. If the current realization satisfies the constraints (with the given tolerance for this scenario (see section 2.5.2)), the value of RS (Reliability Satisfaction) is modified as follows:
 
$$RS = RS + \frac{1}{R}$$
- F. Steps C through E will be repeated until the end of realizations.
- G. Average objective function value (total water consumed) from all realizations is calculated, considering the effect of convergence for each realization.
- H. If reliability satisfaction is greater than or equal to the reliability target for all constraints, the solution is feasible. Otherwise, the differences between them represent the constraint violations.

- I. Steps B through H are used for each string.
- J. As STS is the constraint-handling technique used here, the objective function value (total water consumed) and the constraint violations are used as the selection criteria.
- K. Other GA operators continue as usual.

## 6.6 Analysis

To determine the required number of realizations, the CCGA model was implemented three times, using 10, 20, and 30 realizations, to evaluate each potential solution using the reliability targets in Table 6.2. For each number of realizations, five different runs with different initial seed values are used. Additionally, the effect of changing the reliability target will be investigated in two steps. In the first step, the reliability target for WS (water shortage) will be increased to from 90% to 95%, and in the second step, the reliability target for RW (required water levels) will be decreased to from 100% to 90%.

## 6.7 Results

Figure 6.6 presents the total water consumed of the best feasible solutions obtained for each number of realizations using different initial seed values, and using uniform probability distribution for water consumption rates and crop pattern. The deterministic total water consumed (obtained using the mean values of water consumption rates and crop pattern) is shown in the figure. The total water consumed for most of the runs is higher than the deterministic value. Also, the average total water consumed values for all number of realizations is higher than the deterministic total water consumed. However, there is no apparent relationship between increasing the number of the realizations and the change in the objective function (total water consumed) values.



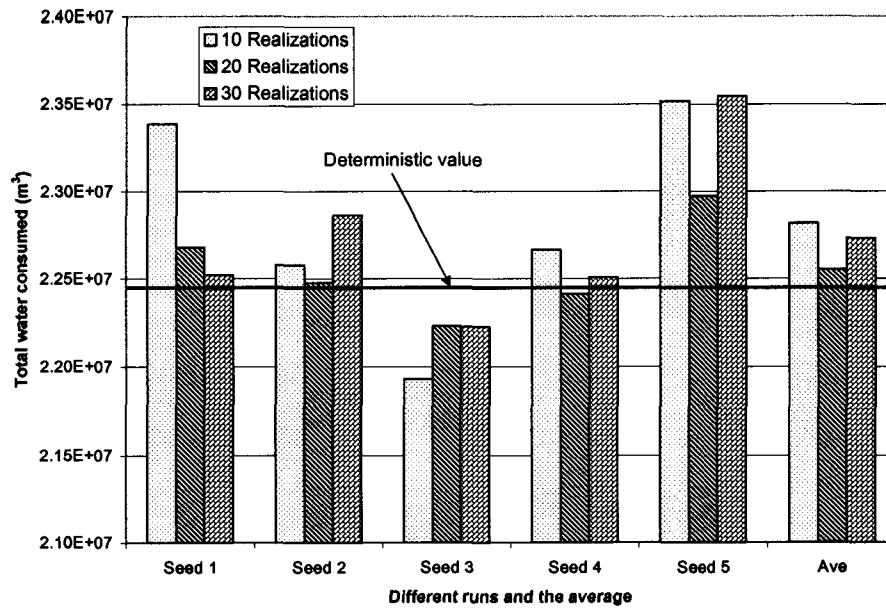


Figure 6.6

Best feasible solution obtained in each run for each number of realizations

Figure 6.7 presents the reliability satisfaction values for the water shortage constraint for the best feasible solution of five different runs and the average for each number of realizations. These values are calculated based on the number of realizations that satisfy the constraints while using CCGA model. Reliability satisfaction for required water levels are 100% for all of these solutions. Figures 6.8 and 6.9 present more accurate estimates of the reliability satisfaction values for the water shortage and required water levels constraints calculated using 200 LHS realizations. From Figure 6.7, average reliability satisfaction for the solutions found by many runs are higher than the target reliability, and for some runs, it is 100%. Also, the highest reliability satisfaction was obtained using 10 realizations, and 20 realizations is the one that got the least reliability satisfaction, with small differences. Figures 6.8 and 6.9 have similar trends, but the reliability satisfactions are smaller in general. From both figures, using 10 realizations satisfies both constraints. Using 20 realizations satisfies only water shortage constraint, and using 30 realizations satisfies none of them. However, the difference from target reliability is small.

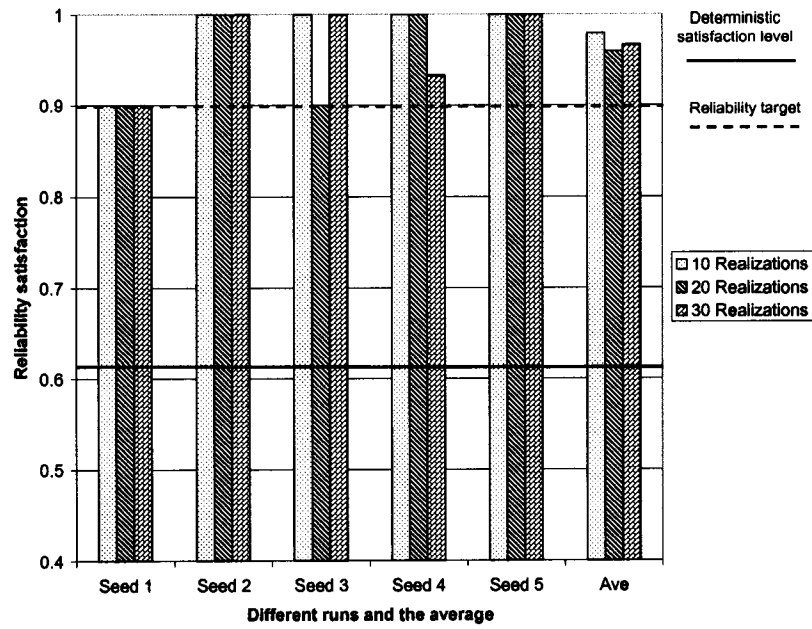


Figure 6.7

Likelihood of satisfying the water shortage constraint in CCGA model for different runs and the average for each number of realizations.

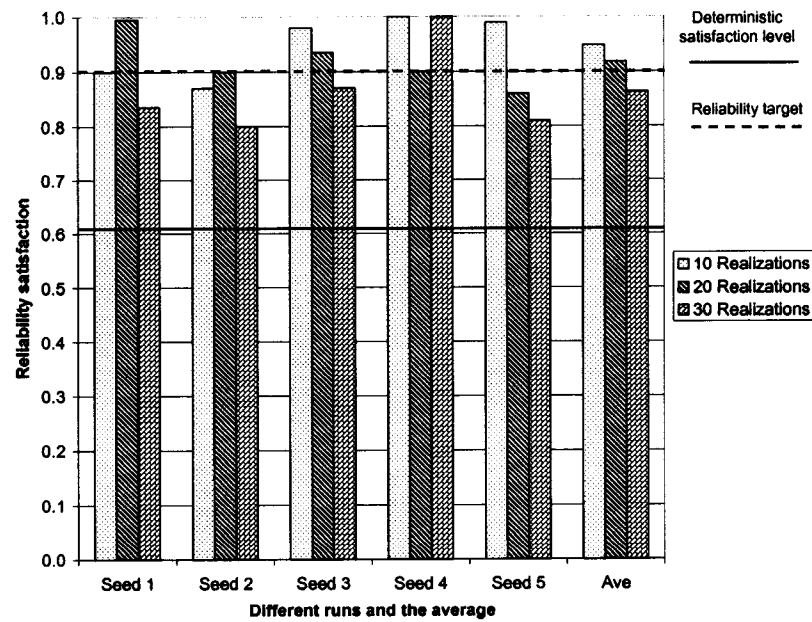


Figure 6.8

Likelihood of satisfying the water shortage constraint in CCGA model for different runs and the average for each number of realizations using 200 realizations

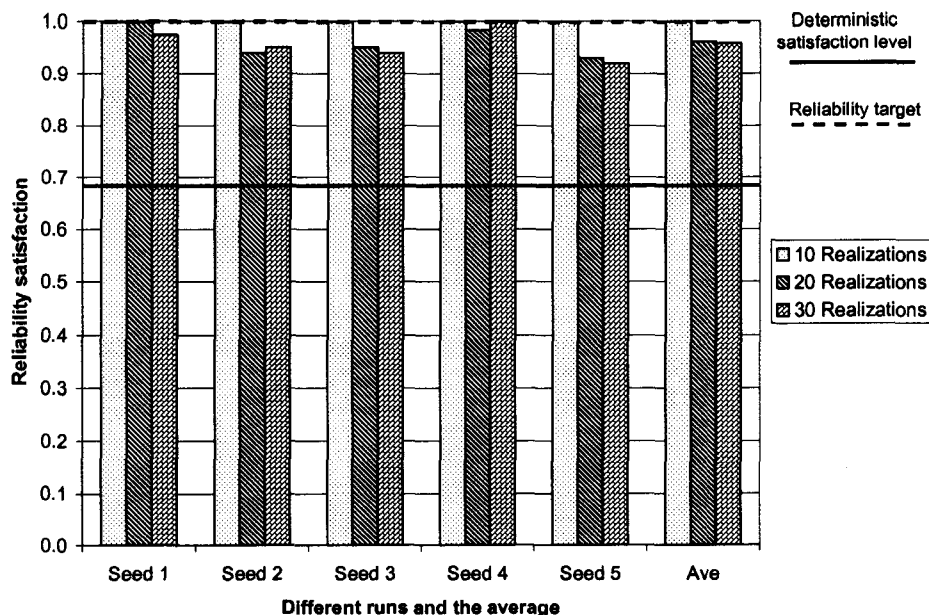


Figure 6.9

Likelihood of satisfying the required water levels constraint in CCGA model for different runs and the average for each number of realizations using 200 realizations

Figure 6.10 displays the average reliability satisfaction for different solutions per generation for the run with the random seed that produced the best solution using 10 realizations. Figure 6.11 displays the average reliability satisfaction for different solutions per generation for the run with the random seed that produced the worst solution using 10 realizations. It can be seen that the average reliabilities for the flood and regulator stability constraints are higher than average reliabilities for other two constraints in both cases. In Figure 6.11, the average reliability for required water levels is much less than the average reliability for all other constraints. Figure 6.12 shows the number of feasible solutions per generation for each of these runs. The worst run fails to find feasible solutions in many generations, and its number of feasible solutions is less in general. It is likely that the difficulty in satisfying the target reliability of required water level is the reason behind this. Changing the values for target reliability of water shortage and required water levels constraints will be tested and the results will be compared with the results from Figures 6.10 and 6.11.

The results (Figure 6.6 to 6.12) show that by incorporating estimates of reliability in the search procedure, solutions with higher reliability can be found with relatively

small increase in the objective function value. The average total water consumed for the best solutions obtained by different runs using different realizations are between 0.42% and 1.59% higher than that of the deterministic solution, and have much less likelihood of causing water shortages or failing to satisfy the required water level at the start of the next irrigation period in the network.

The results indicate that good solutions can be obtained using a sample size of 10 realizations to evaluate each potential solution. The good performance of this small sample is likely due to the fact that over the course of a number of generations, a given solution is tested with a much larger number of realizations.

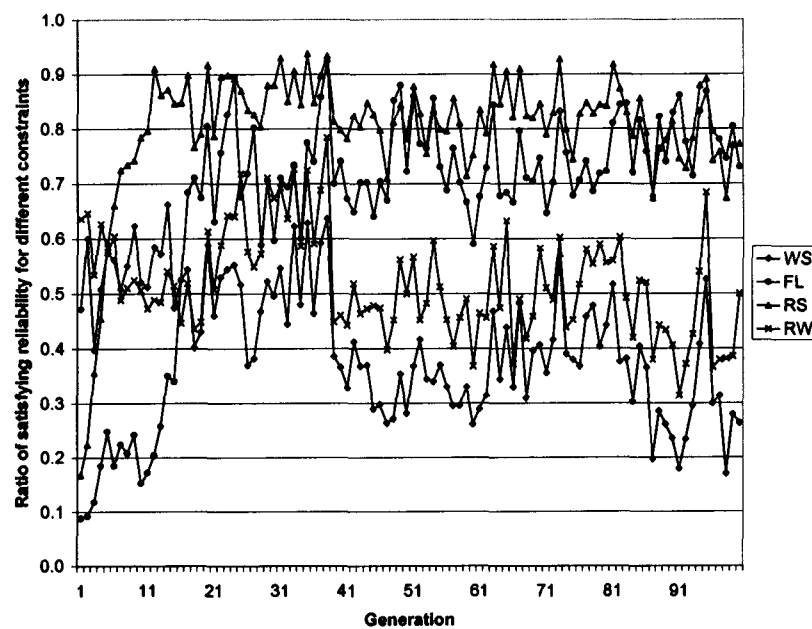


Figure 6.10

Average target satisfactions values for different constraints per generation for the random seed run that resulted in the best solution using 10 realizations

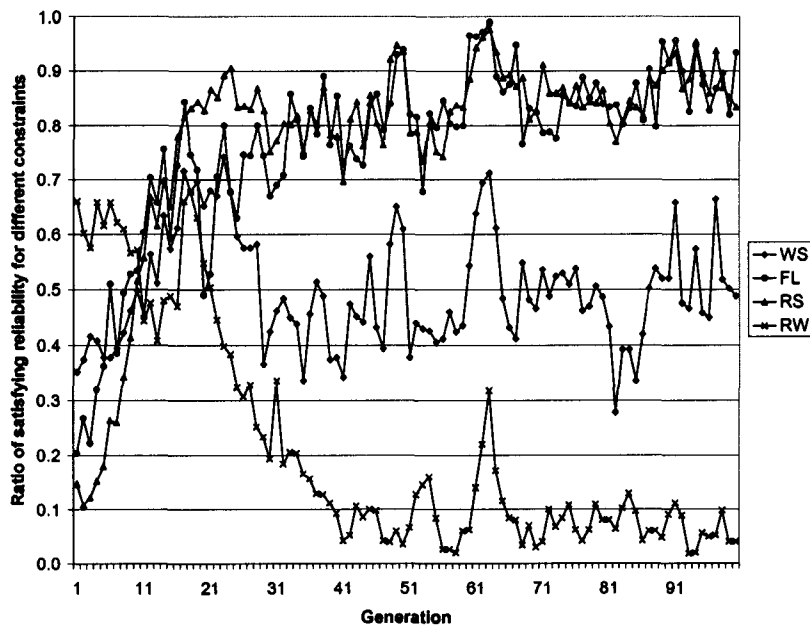


Figure 6.11

Average target satisfactions values for different constraints per generation for the random seed run that resulted in the worst solution using 10 realizations

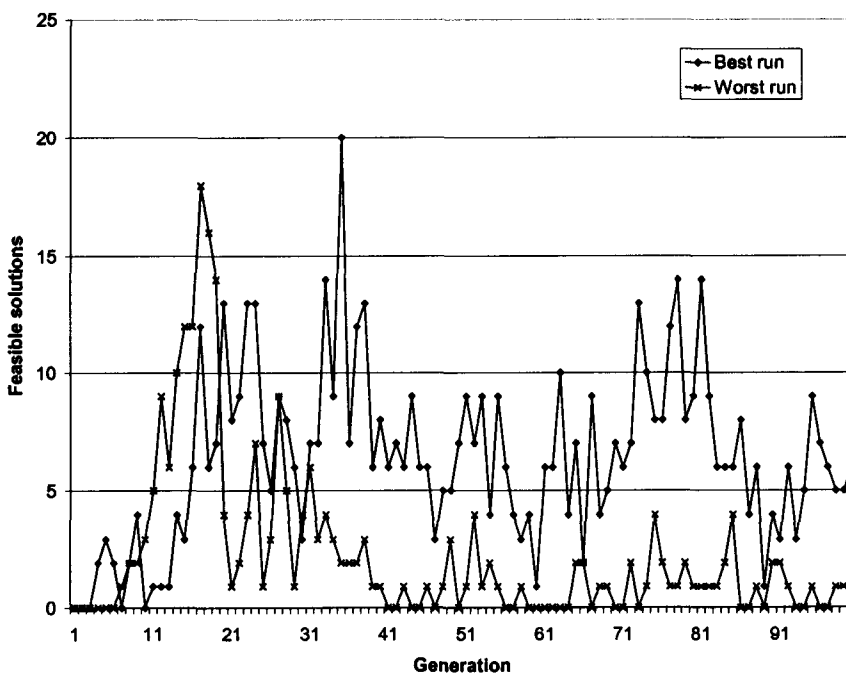


Figure 6.12

Number of feasible solutions per generation for the random seed runs that resulted in the best and worst solutions using 10 realizations

### **6.7.1 Effect of the number of realizations used to evaluate each solution**

From the previous results, using a larger number of realizations (20 and 30) slightly outperform 10 realizations regarding the objective function value (Figure 6.6). Regarding reliability satisfaction (Figures 6.7 through 6.9), using 10 realizations has highest reliability satisfaction value and using 30 realizations has the least reliability satisfaction value, but the differences are small. Also comparing the results of 20 and 30 realizations (Figures 6.6 through 6.9), the results of 20 realizations outperforms the results of 30 realizations regarding the objective function value. Regarding reliability satisfaction, results are different between using the actual number of realization or higher number of realizations. All of the differences are relatively small. Thus, the differences between the results for different number of realizations may be due to the random effect, and using a number of realizations as low as 10 can be adequate for achieving good results.

### **6.7.2 Effect of changing the reliability target**

The effect of changing the reliability target is tested twice. First, the reliability target for the water shortage constraint will be increased from 90% to 95%. This is tested using a new set of 20 realizations to evaluate each potential solution. Second, the reliability target for required water levels constraint is reduced from 100% to 90%, and the reliability target of water shortage is kept as 90%. This is tested using a new set of 10 realizations to evaluate each potential solution. For each case, five runs using different starting random seeds were conducted.

Figures 6.13 and 6.14 present the results of each test. In each figure, the five different runs and the average value are presented. In the first test, increasing the target reliability of satisfying water shortage constraint resulted in an increase of the total water consumed for most of the runs. The increase of the average value of total water consumed is 1.1%. In the second test, relaxing the target reliability for the required water level constraint resulted in a decrease of the average total water

consumed of about 1.1%. Also, most of the five runs got lower total water consumed when the target reliability is relaxed.

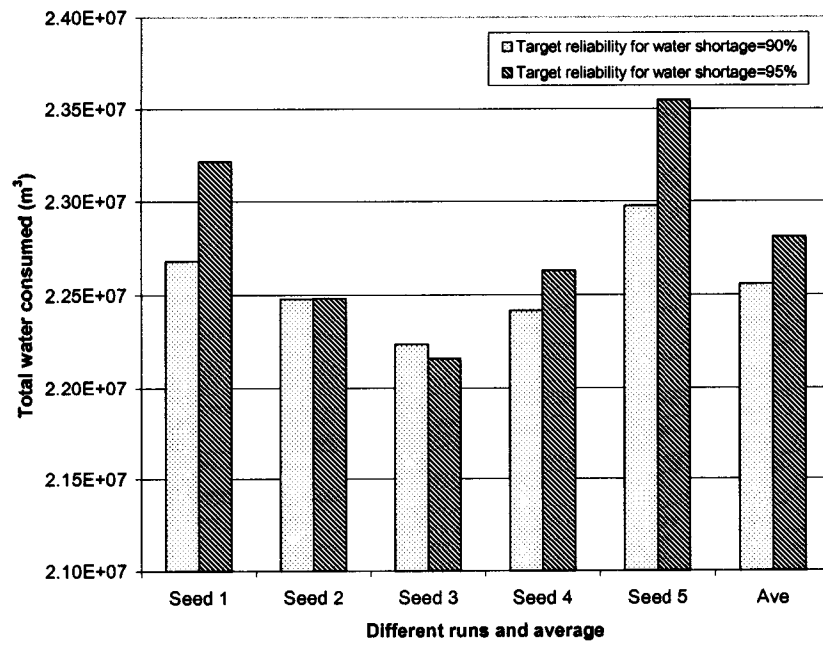


Figure 6.13  
The effect of increasing target reliability of water shortage to 95%

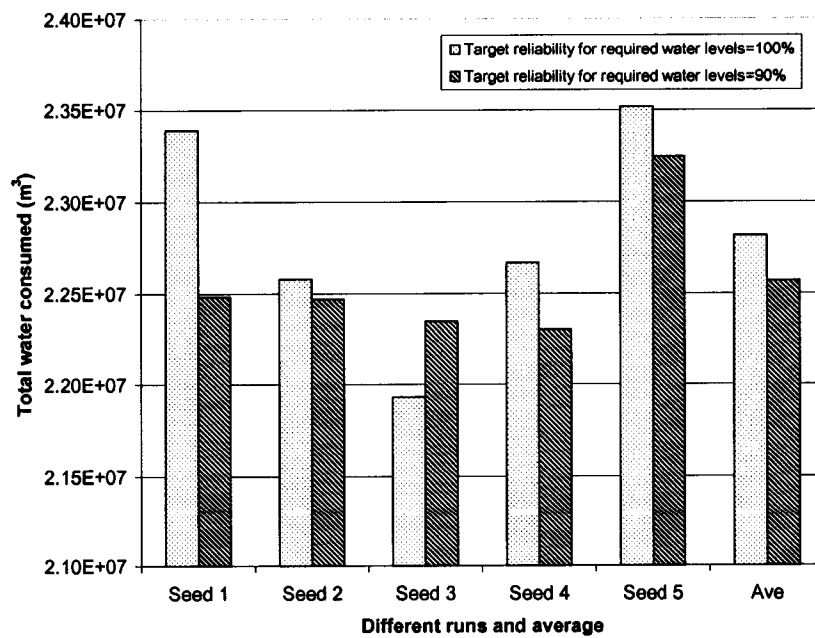


Figure 6.14  
The effect of decreasing target reliability of required water levels to 90%

Regarding reliability satisfaction for water shortage and required water levels constraints, Figures 6.15 to 6.18 display the average reliability satisfaction for both constraints versus generations for the runs that produced the best and the worst solutions for each of the two tests. From Figures 6.15 and 6.16, there is not clear evidence that increasing the reliability target of water shortage constraint affected the average reliability satisfaction. Also, the effects on the best and worst solutions appear to be opposite. While average reliability satisfaction is reduced in the worst scenario as the result of increasing reliability target, it is increased in the best solution, indicating that the difference may be a random effect.

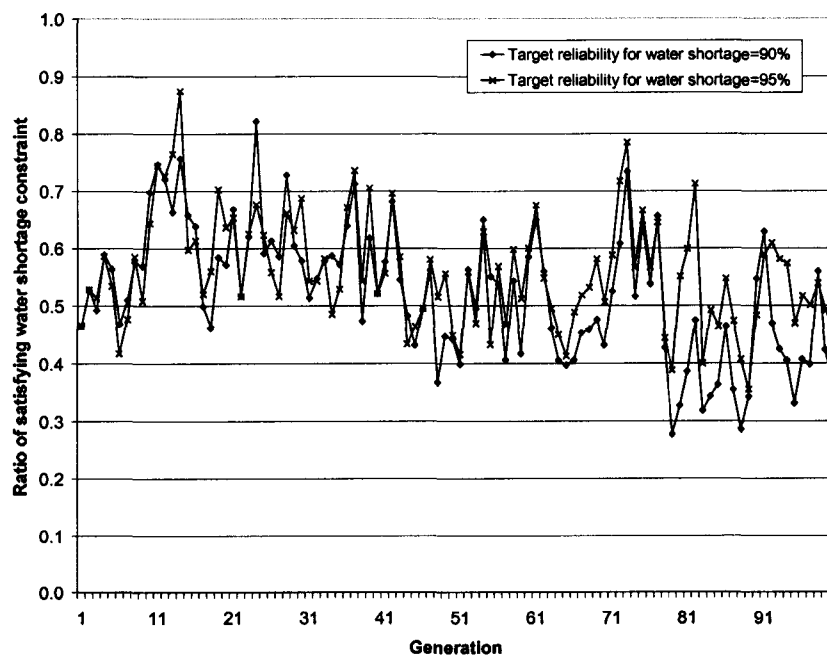


Figure 6.15

Effect of increasing target reliability of water shortage to 95% for the random seed run that resulted in the best solution



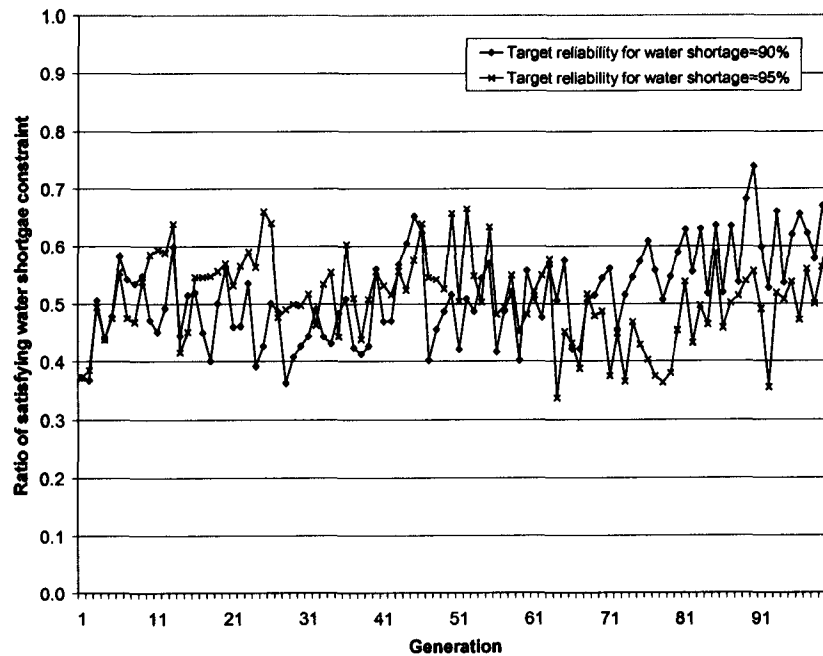


Figure 6.16

Effect of increasing target reliability of water shortage to 95% for the random seed run that resulted in the worst solution

Regarding the required water levels constraint, Figures 6.17 and 6.18 shows that there is an improvement associated with relaxing the reliability target for this constraint for both the best and worst runs.

As expected, results indicate that the model can achieve more reliable solutions at the expense of slightly increasing the objective function value, or it can decrease the objective function value at the expense of decreasing the reliability target.

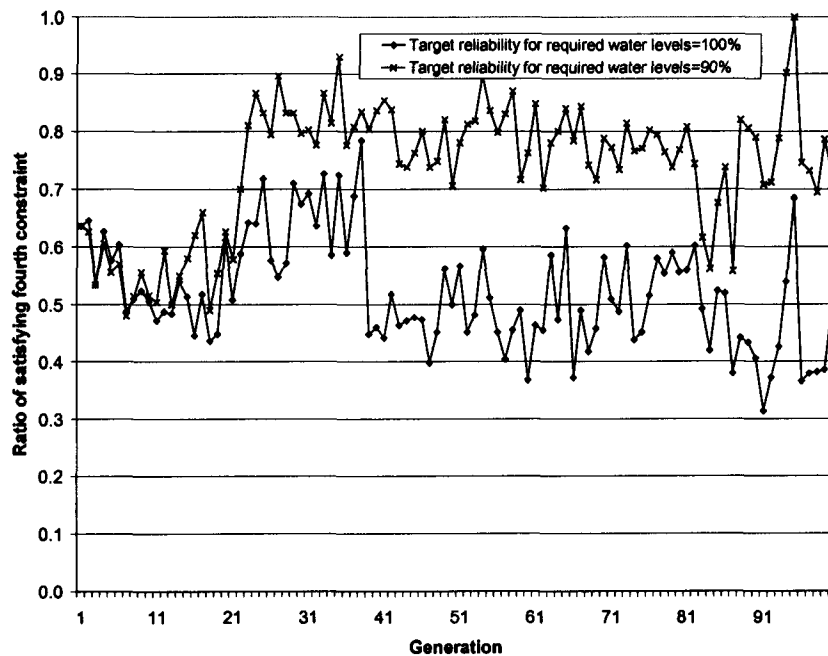


Figure 6.17

Effect of decreasing target reliability of required water levels to 95% for the random seed run that resulted in the best solution

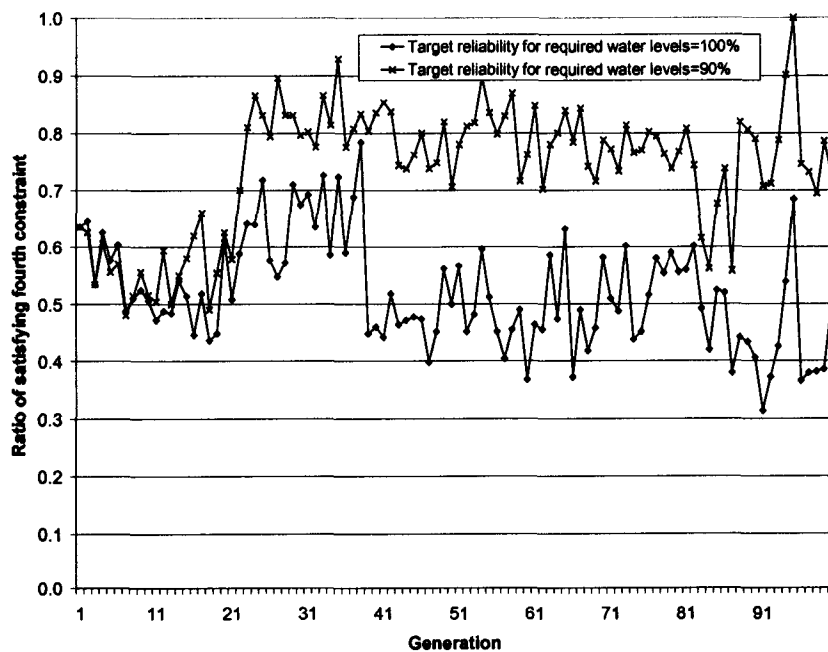


Figure 6.18

Effect of decreasing target reliability of required water levels to 95% for the random seed run that resulted in the worst solution

## 6.8 Conclusions and Recommendations

The deterministic model described in the Chapters 2, 4, and 5 has been extended to incorporate the likelihood of satisfying constraints under conditions of uncertainty in the water consumption rates and crop patterns. A chance-constrained optimization technique was used within the GA search process. Latin Hypercube Sampling was used to generate a relatively small number of realizations to evaluate each potential solution. Uniform probability distribution functions were used to express both uncertain variables (water consumption rates and crop patterns). The results show that the CCGA model can increase the reliability of satisfying constraints at the expense of a small increase in the objective function value.

Also, the results show that using LHS sampling with as few as 10 realizations to evaluate each potential solution can yield good results. From the results (Figures 6.6 through 6.9), there is no clear relationship between increasing the number of the realizations and the improvement in the objective function value or in the reliability satisfaction, indicating that the differences between the results produced by different numbers of realizations may be a random effect. The runs are associated with a very heavy computational effort. A single run using 10 realizations required about 20 hours on average on PC Pentium 4 (2.0 GHz with 512 MB RAM), and it is nearly proportionally longer for larger number of realizations.

Suggested future work:

- ❑ Methods for reducing the computational effort should be investigated.
- ❑ If possible, more information about the uncertain parameters should be collected to express them using more accurate probability distribution functions.
- ❑ The suitable number of realizations that should be used within the model should be investigated again, as no general rule could be obtained for improvement of the results due to increasing the numbers of realizations.
- ❑ The model should be run repeatedly using various target levels of reliability to generate a tradeoff relationship between reliability and objective function value.
- ❑ Various sampling strategies for the CCGA should be tested to determine the most efficient strategy. For example, another strategy that has been shown to perform

well for an air quality management problem is to use the same set of realizations to evaluate all of the strings in the population, with a new set of realizations generated for each generation (Loughlin and Ranjithan, 1999). Also, the number of realizations used for the evaluation of each string should be varied to determine the most efficient size of the set of realizations.

## 6.9 References

- Ali, A. S. "Water productivity in Egyptian Agriculture," *International conference on integrated management of water resources in the 21st century*, Cairo, Egypt, November 21-25, 1999
- Ali, H. M. "Determining optimal crop pattern in Egypt by the use of multicriteria analysis," *Water Science (The scientific magazine published by the National Water Research Center, Ministry of Water Resources and Irrigation, Egypt)*, 28th – 29th Issue, October 2000-April 2001
- Chan Hilton, A. B. and T. B. Culver., "Optimizing Groundwater remediation design under uncertainty," *Proceedings of the joint conference on water resources engineering and water resources planning and management*, July 30-August 2, 2000, Minneapolis, MN: ASCE, 2000
- Cieniawski, S. E., Eheart, J. W., and Ranjithan, S. (1995). "Using genetic algorithms to solve a multiobjective groundwater monitoring problem," *Water Resources Research*, Vol. 31, No 2, pp. 399-409
- El Qusoy, D. "Possibilities of using the treated sewage water in Irrigation" (In Arabic), *Annual conference of the National Water Research Center*, Cairo, Egypt, December, 26-27, 1995
- Fawzy, G. M. "The efficient use of irrigation water in the Egyptian agriculture," *International conference on integrated management of water resources in the 21st century*, Cairo, Egypt, November 21-25, 1999
- Gates, T. K., Alshaikh, A. A., Ahmed, S. I., and Molden, D. J., "Optimal Irrigation Delivery System Design Under Uncertainty," *Journal of Irrigation and Drainage*

- Engineering*, American Society of Civil Engineers, Vol. 118 No. 3, MAY./JUN. 1992, pp. 433-449
- Harrell, L. J., "Evolutionary Algorithm-based Design of a System of Wet Detention Basins Under Uncertainty for Watershed Management," *Proceedings of the ASCE World Water and Environmental Resources Congress*, May 2001.
- Lansey, K., Duan, N., Mays, L., and Tung, Y. "Water Distribution Systems Design Under Uncertainty," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 115 No. 5, SEP./OCT. 1989, pp. 630-645
- Loughlin, D. H. and Ranjithan, S. R. (1999). "Chance-Constrained Genetic Algorithms," *Genetic and Evolutionary Computation Conference (GECCO)*, pp. 369-376.
- Mays, L. W., "Optimal Design of Culverts Under Uncertainty," *Journal of Hydraulic Engineering*, May 1979, pp. 443-459.
- Morgan, M. Granger and Henrion, Max (1990). *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*, Cambridge University Press.
- Ritzel, B. J., Eheart, J. W., and Ranjithan, S. (1994). "Using genetic algorithms to solve multiple objective groundwater pollution containment problem," *Water Resources Research*, Vol. 30, No 5, pp. 1589-1603
- Taha, H. A., *Operations Research: An Introduction*, Pearson Education, Inc., New Jersey, 2003
- Tung, Y. K., and Mays, L. W., "Optimal Risk-Based Hydraulic Design of Bridges," *Journal of Water Resources Planning and Management*, American Society of Civil Engineers, Vol. 108 No. 2, MAR./APR. 1982, pp. 191-203
- Yen, B. C., and Ang, A. H.-S. "Risk Analysis in Design of Hydraulic Projects," *Stochastic Hydraulics, Proceeding of the 1<sup>st</sup> Int. Symp.*, University of Pittsburgh, Pittsburgh, Pa., 694-701.
- Wyss, G. D., and Jorgensen, K. H., 1998. "A User s Guide to LHS: Sandia's Latin Hypercube Sampling Software," *Sandia National Laboratories Technical Report SAND98-0210*, Albuquerque, NM.

## CHAPTER 7

### GRAPHICAL USER INTERFACE FOR THE MODEL

#### 7.1 Introduction

A user-friendly interface has been developed to make the model easier to use. The model can be used as an unsteady flow simulation model, or as an optimization model. In the case of the optimization model, the best solution that is obtained is routed using the unsteady flow model, and the results of this routing are available with the genetic algorithm results.

The interface consists of four categories as shown in Figure 7.1, which are

- Files commands, to help the user work with the projects, such as begin a new project, open an existing project, and other commands.
- Data commands, to help the user enter different types of the data. The data can be categorized into two sub-categories: hydraulics data and settings data.
  - Hydraulics data to describe the irrigation network.
  - Settings dialog to define different parameters, such as genetic algorithm parameters, uncertainty parameters, etc.
- Reports, which are summaries about the data that have been entered. Reports are presented in one of two forms: table form for hydraulics data, and page form for genetic algorithm data.
- Results, which may be genetic algorithm results or hydraulics results. Results can be presented in three forms: table form, chart form, and page form. Page form is used to present the final reports about the whole run. The tables and charts are used to present details.

A brief description about each category with examples of its commands is presented in the following sections.

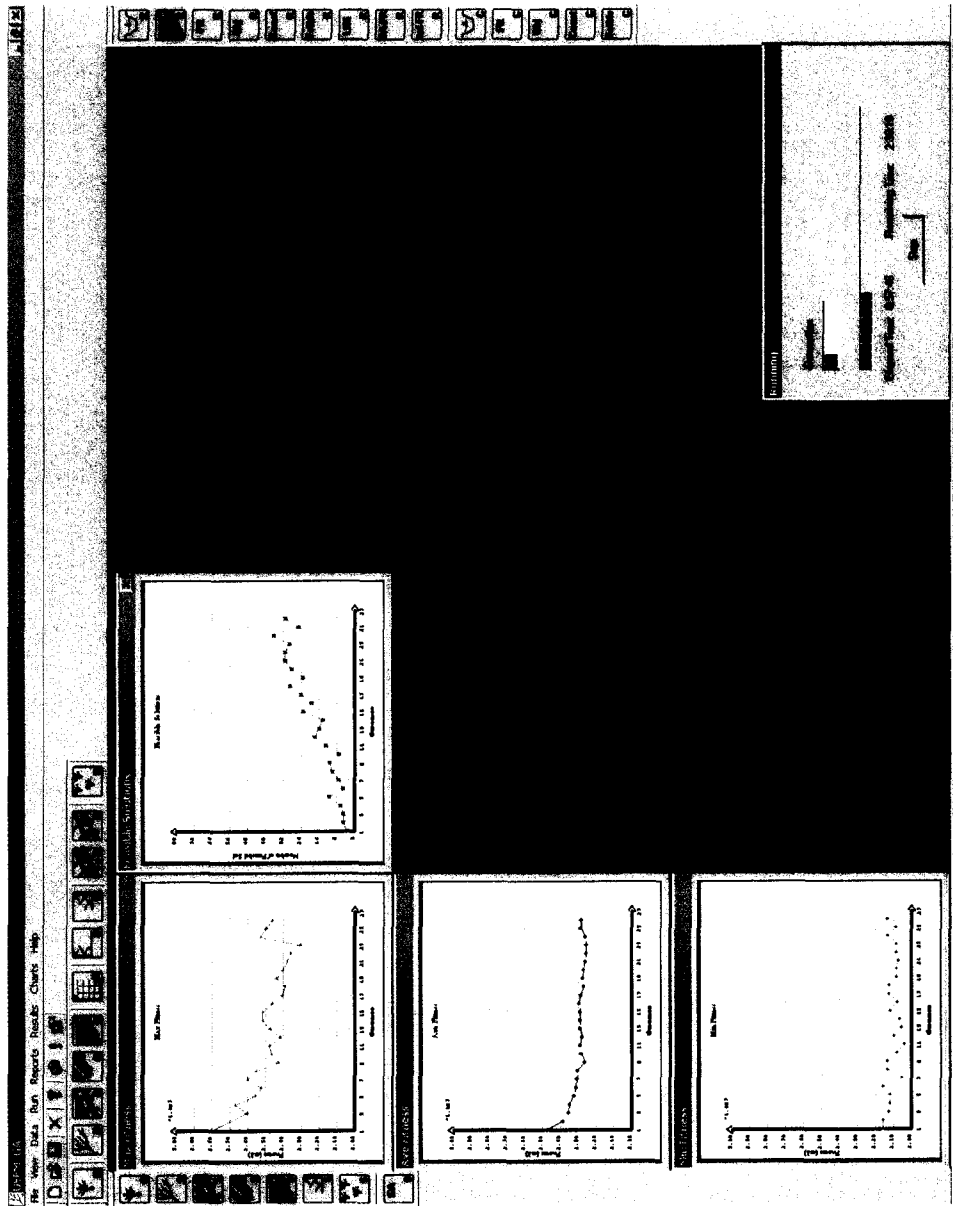


Figure 7.14  
The interface during the run as an optimization model

## 7.2 File commands

This category of commands is used to work with projects, and include commands such as create, open or delete a project. The interface was designed such that all projects are in one folder (projects) under the program folder. The name of the project cannot contain any spaces.

Some examples from this category are as follows:

### 7.2.1 Open Project

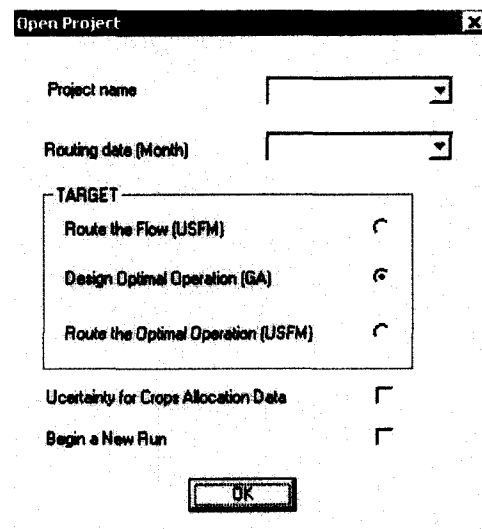


Figure 7.2  
Open project dialog

This command will open an existing project. The dialog, as shown in Figure 7.2, has five types of the data describing different characteristics of the model to be used. When a project is selected, all the characteristics of the project that were previously saved will be retrieved. The user can keep these characteristics as they are or change them. These characteristics are:

- Routing date: this is the month when the flow is routed. This date is used to define the water consumption rate for each crop.
- Target: the model can be used for two purposes. It can be used as unsteady flow simulation model to route the flow or as an optimization model to define the best



schedule for an irrigation network. For the unsteady flow model, there are two options regarding how to enter decision variables data. These data can be entered by the user (choice 1) or it can be the output of the optimization model (choice 3) (the best solution that was found). This third choice was added to give the user the chance to modify the output of the optimization model. For example, one could route the flow for smaller time interval and distance interval to get more details, round the decision variables to more practical units and check the results, or make other desired changes.

- **Uncertainty:** This option is enabled only when the model is used as an optimization model. With this option, the water consumption rate and the allocation of each crop are treated as uncertain data, as described in Chapter 6.
- **New Run:** This option is also enabled only when the model is used as an optimization model. If this option is not chosen, the model will continue the GA run where the previous run left off (after the last generation). If the new run option is chosen, the program will begin from the beginning, discarding the results of previous run. If there is no previous run, choosing this option will have no effect.

### **7.2.2 New project**

This command is used to begin a new project. The dialog is similar to the previous dialog. The difference is that the user should enter the project name instead of selecting it, and must define other characteristics of the project. Also, it does not include the third choice of the target, and “Begin New Run” option.

### **7.2.3 Save Project As**

This command saves a copy of the current project with a new name. The model will save the input data files only. The dialog in Figure 7.3 is used to enter the name of the new project.

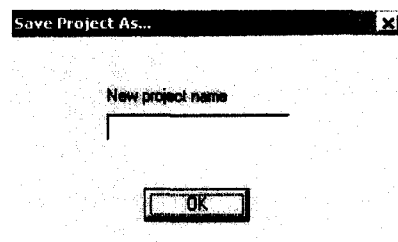


Figure 7.3  
Save project As dialog

### 7.3 Data Commands

The model has two types of the data, hydraulics data, and settings data. The hydraulics data contains the characteristics of the irrigation network. The settings dialog describes several parameters of the model.

There are two points regarding entering the data:

- The sequence: some data should be entered in a particular order. For example, before entering any hydraulics data, the maximum data number in the Settings dialog that is used to allocate the memory should be entered. Also, before entering the regulators and branches for a channel, this channel should be defined.
- Checking the data: the model validates the data at three levels. The first level is performed during data entry. For instance, some data should have positive or non-negative values. Also, in some dialogs that require maximum and minimum limits for a variable, the program will check that the minimum value is smaller than or equal to maximum value. The second level is performed before the program is run. For this level, the data that are related to different dialogs or different records in the dialog will be checked together. For instance, the model will check that the numbering of the canals is acceptable (see section 7.3.1.1 for details about numbering requirements). Another example is that the model will check that the cultivated areas for all reaches and branches of a channel equal the total cultivated land of this channel. The third level is during the calculations.

For hydraulics data, there are 11 commands in five different categories, which are:

- ❑ **Crops.** There are two commands in this category. The first is used to enter the water consumption rate for each crop in each month, and the data that will be used within uncertainty (Maximum and minimum consumption rates, and maximum and minimum allocation ratios). The other command is used to enter the ratio of each reach that is cultivated by each crop.
- ❑ **Geometry data.** This contains the canals data, the regulators data, and reaches data.
- ❑ **Initial data.** This consists of the initial water levels at the upstream end of each channel, and they are found in Canals Data dialog, and the initial data at regulators, which is found in a separate dialog.
- ❑ **Boundary.** These commands define the boundary time, upstream boundary, and downstream boundary.
- ❑ **Operations.** These commands define the operation times and operation data. For the Settings dialog, there are five different pages, which are:
  - ❑ **Genetic Algorithm data.** This part contains all genetic algorithm parameters, such as crossover probability and mutation rate. It is also used for choosing the selection method, constraint-handling technique, and tolerance for constraints.
  - ❑ **Uncertainty data (LHS parameters).** This part is used for defining the data that is used when considering uncertainty with water consumption rate and crop allocation.
  - ❑ **Maximum data.** This dialog defines the maximum expected number of different hydraulic data types such as canals and operations. These numbers are used to allocate the required memory for hydraulics data.
  - ❑ **Routing data.** This part defines distance intervals and time intervals. It also used to define the initial data that is used while opening new gates.
  - ❑ **Convergence.** This part defines values that are used to check the convergence and also the data that is used to penalize un-converged solutions in GA.

For hydraulics data, all dialogs have the same 10 buttons, which are in the following categories:

  - ❑ **Buttons to end the session.** There are two ways to end the dialog, either by saving the data or by the canceling the changes.

- Records buttons. This category includes adding new record, deleting a record and copying a record. Copying a record is used when two records have similar data, so the user can only change the identifier of the record, such as the canal number, then push Copy button to add an identical record with the new identifier.
- Moving buttons. These buttons are used for moving between records, like moving to next, previous, last, or first record. Also, the user can use the GoTo button to move to a specific record by defining the identifier of that record.

### 7.3.1 Hydraulics Data

The hydraulics data are described in the following subsections.

#### 7.3.1.1 Canals Data

Figure 7.4 presents the dialog for the canal data. This dialog defines general characteristics of a channel, and it has the following data:

- Canal definition, which contains the canal number, number of the main canal that this canal diverts from, the location and the side of this diversion. It should be mentioned that there is a specific way to number canals. The main canal has number 1, followed by the canals that divert from it, then canals that divert from second branches, beginning from the first branch, and so on. Figure 7.5 gives an example about the numbering. While working with this dialog, the model will check that the canal number is greater than or equal 1, the main canal that this canal is diverging from is greater than or equal to 0, and the location is greater than or equal to 0.0. A complete validation of the numbering of these data is performed before running the model.
- Canal members, which contains the number of regulators and the number of branches for this canal.
- Total data, which are the total length and the total cultivated area for this channel. This data is used for data checking.

Figure 7.4  
Canals data dialog

- Initial water level, which is used with initial data at regulators as the initial data for the network during the routing.
- There are two options in the dialog which the user can select. The first defines if this canal conveys water outside the network, which is used to calculate total outflow from the network. The second option defines if the water levels of this canal will be included in the constraint on the water levels at the end of the routing.

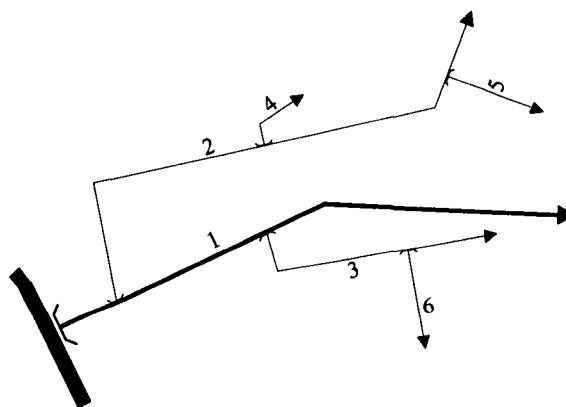


Figure 7.5  
Example of how the canals must be numbered

### 7.3.1.2 Regulators Dialog:

Regulators: Geometry Data

File Goto Record

Regulator Definition

Canal Number  
1

Regulator Number  
1

Regulator Location (Km)  
0.1

Regulator Geometry

Regulator Bed Level (m)  
12

Regulator Thickness (m)  
10

Max Difference (US-DS) (m)  
2.2

Discharge Data

Gate Width (m)  
40

Discharge Coeff  
0.58

GA data

Cultivated Area DS the Regulator (Ac)  
80000

Save  
Cancel

Record  
New  
Copy  
Delete

Goto  
First  
Next  
Previous  
Last  
Go To

Figure 7.6  
Regulators data dialog

The Regulator Geometry Data dialog, as shown in Figure 7.6, contains the following:

- Regulator definition data. This group of data consists of canal number, regulator number, and the location of the regulator.
- Regulator geometry. This includes the bed level, the regulator width, and the maximum allowable difference between the upstream and the downstream water levels.
- Discharge data. This includes the gate width, and the discharge coefficient.
- The cultivated area downstream of the regulator, which is used to penalized a solution that violates the regulator stability in GA. If the model is used as unsteady flow simulation model, this part will be disabled.

### 7.3.1.3 Reaches Dialog

The Reaches Data dialog (Figure 7.7) contains the data for each reach in the irrigation network, which includes:

- The length and the cultivated area of that reach.
- Cross sectional data and Manning coefficient.
- Bank level and longitudinal slope.
- Required water levels at the beginning and at the end of the reach. This part is enabled only if the option “Has required WL data” is selected for this canal in Canals Data dialog.

Figure 7.7  
Reaches data dialog

#### 7.3.1.4 Regulators Initial Data Dialog

The initial data at each regulator (Figure 7.8) consists of:

- Upstream and downstream water levels.
- Gate opening. This value is used with the unsteady flow model.
- Minimum and maximum gate openings. These values are used within genetic algorithm to randomly select the initial gate opening. This part is enabled only if the model is used as an optimization model.

- Initial discharge. This part is used only if the regulator is free opened. Otherwise, the discharge will be calculated using the sluice gate equation, and the data in this dialog and this discharge value will be ignored.

Figure 7.8  
Regulators initial data dialog

### 7.3.1.5 Operation Time and Boundary Time Dialogs

Figure 7.9  
Operation time dialog

This dialog (Figure 7.9) is used to define the allowable times for operations. The data are the operation number and the time step for this operation. A similar dialog is



used to define the time of given boundary conditions. Regarding boundaries, the user can enter the boundaries at any channel at any of these times, and the program will interpolate for other time steps.

### 7.3.1.6 Operations Data Dialog

Figure 7.10  
Operations data dialog

The operations data dialog is used to define the regulator that will have an operation and the time for that operation (operation number). It also defines the value of the operation. This value is positive for opening and negative for closing. The maximum and minimum values for the operation is enabled if the model is used as an optimization model, and they are used to randomly select the operation value. Similar dialogs are used to define the upstream and downstream boundary conditions.

### 7.3.2 Settings Dialog

The settings dialog is used to define different parameters for the model through five different pages. The genetic algorithm parameters page is shown in Figure 7.11. This dialog defines six different parameters for the genetic algorithm. For the

population size, since the model uses binary tournament selection, the number should be even. Otherwise, the model will give a message error and increase the population size by 1.

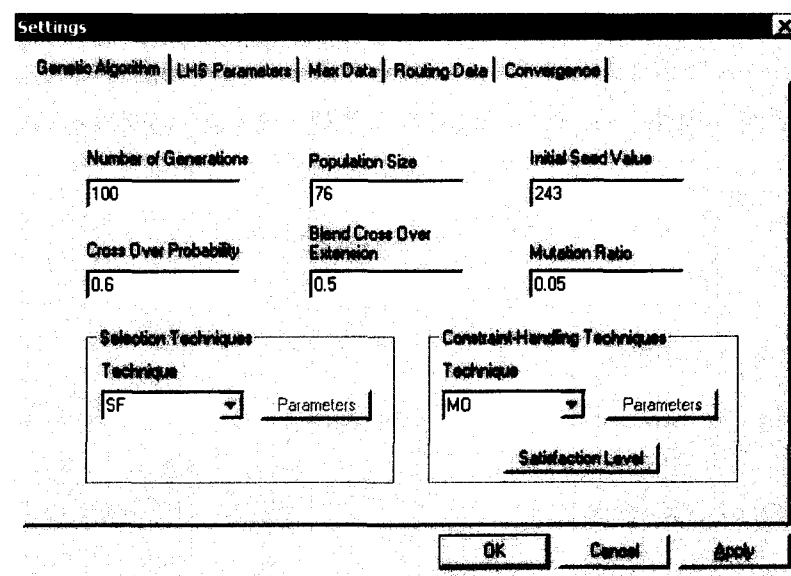


Figure 7.11  
Settings dialog

Also, the dialog defines the selection method, and the constraint-handling technique that will be used. Some of these techniques require defining parameters, and this is done through a popup dialog. Also, the dialog is used to define the satisfaction level for each constraint. If the run is not a new run, the only enabled item in this page is “Number of Generations”, where the user can increase the number of generations and continue the run.

## 7.4 Reports Commands

This category of commands displays some types of data that were entered before. There are two types of reports: hydraulics data reports, which are displayed in table form, and a genetic algorithm report, which is displayed in page form. The tables that are used in reports commands and in results commands have a fixed format, and each has the following options:

- ❑ Show or hide horizontal and vertical grid lines.
- ❑ Resize the columns and rows: these options are performed by pressing the cursor on the line between two rows or two columns and drag it.

Also, there are four common commands in all tables, which are:

- ❑ Change font: To change the font of the table. The font will be changed for the entire table (headers and data).
- ❑ Print Preview: for previewing the printable copy of the data.
- ❑ Print: for printing the table.
- ❑ Close: To close the dialog.

Hydraulics reports include nine different types of data, which are.

- ❑ Crops data.
- ❑ Canals data.
- ❑ Regulators data.
- ❑ Regulators water levels data.
- ❑ Reaches data.
- ❑ Downstream boundary data
- ❑ Crops allocation data.

An example of canals report is presented in Figure 7.12.

Canal No	From Canal	At Point Km	Side	No. of Reg	No of Branches	To
1	0	0.00	Right	3	12	
2	1	6.00	Left	2	3	
3	1	11.10	Left	3	14	
4	1	20.05	Left	2	5	
5	1	22.60	Left	2	6	
6	1	23.00	Left	1	1	
7	1	25.00	Left	3	2	
8	1	27.96	Right	3	6	
9	1	29.96	Right	3	3	
10	1	42.00	Right	4	11	
11	1	52.26	Left	3	5	
12	1	53.49	Left	1	0	
13	1	53.50	Left	2	3	
14	2	1.82	Right	1	0	
15	2	23.42	Right	1	0	
16	2	23.42	Right	1	0	

Show Vertical lines  
 Show Horiz. lines  
 Allow Row resizing  
 Allow Column resizing

Change Font...  
 Print...  
 Print Preview  
 Close

Figure 7.12  
Canals data report

The GA report shown in Figure 7.13, contains the following data:

- ❑ Number of generations.
- ❑ Population size.
- ❑ Initial seed value
- ❑ GA parameters: crossover probability, mutation rate, and blend crossover extension.
- ❑ Selection method, and constraint-handling technique, with their parameters, if applicable.

The report has three buttons to preview, print or quit the dialog.

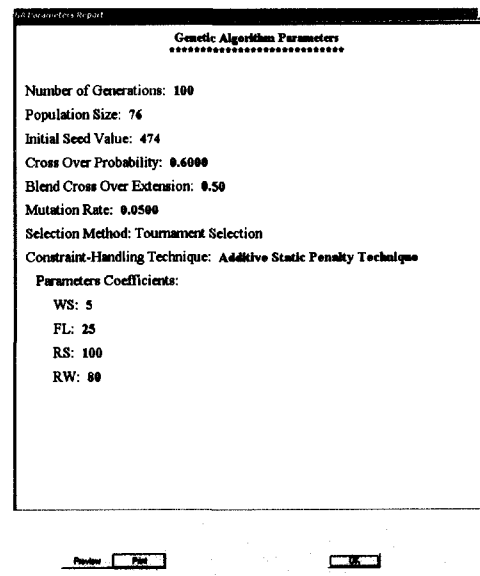


Figure 7.13  
Genetic algorithm data report

## 7.5 Run Menu

The run menu has three commands: define the settings, check the data, and run the model. First command was described in section 7.3.2 and the other two commands are described in the following subsections.

### **7.5.1 Check command**

This command is used to check the unsteady flow model data before running the model. It is also performed automatically when the Run command is used, so the user can simply use Run command directly. If there is an error, the model will give a message and the run command will not work.

### **7.5.2 Run command**

If the model is used as unsteady flow simulation model, there will be a waiting message. If it is used as an optimization model, the screen will look like Figure 7.14. There are four dialogs that present the maximum, average and minimum fitness during the run. They are also present the number of feasible solutions during the run. On the right side, there is a dialog showing the generation number, a progress slider about how much of work has been done, the time, and a button to stop the run. If the user presses this button, the model will give a message that it will stop after the current generation.

It should be mentioned that before the run, all open dialogs will be closed. If the dialog is a results or a report dialog, the model will just close it. If it is a data dialog, the model will give the user the choice to close the dialog or cancel the Run command. Also, during the run, all other commands are disabled.

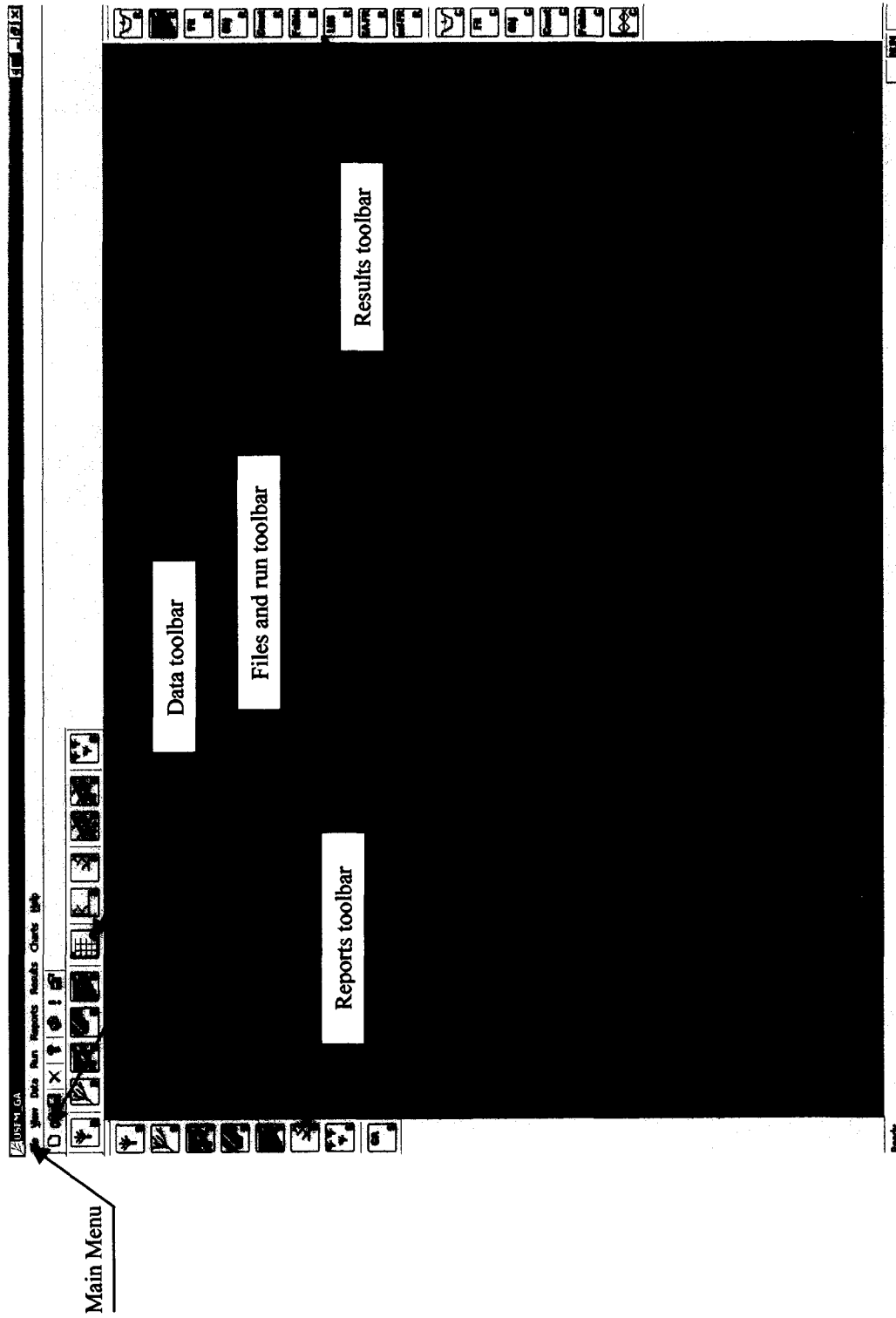


Figure 7.1  
Main page of the interface

## 7.6 Results

The model display results in one of three formats: page form, table form, and chart form. It has three categories of commands, which are:

- Final reports. This category includes the final reports about the whole run, and is in page format. There are two final reports; one about the GA run, and one about the flow routing. The later is available in both cases of the model, either as the result of an unsteady flow model, or as the result of routing the optimal solution of the optimization model.
- Hydraulic results. There are two types of hydraulic results: hydraulics data at each point in the network, and hydraulics data at the regulators. Hydraulics data at each point in the network are available in table and chart format. The data at the Oregulators are available only in table format.
- Genetic Algorithm results. This category includes different commands, which are:
  - Fitness data (as in Figure 7.18).
  - Objective (cost) data (as a figure similar to Figure 7.18).
  - Constraints violations data. (as in Figure 7.19).
  - Number of feasible solutions (as in Figure 7.20)
  - Satisfaction reliability, which is shown in a table, and when uncertainty is considered during the run.
  - STS average probabilities, which is shown in a chart when STS is used as a constraint-handling technique.

Some examples from the results are given in the following subsections.

### 7.6.1 Water level data

The water level data can be presented in a table (as in Figure 7.15) or as a chart (as in Figure 7.16). In both cases, the user can present the data of a channel for a given time step, or the data of a specific point during all time steps. In addition to that charts can present a simulation of the water level during the whole run.

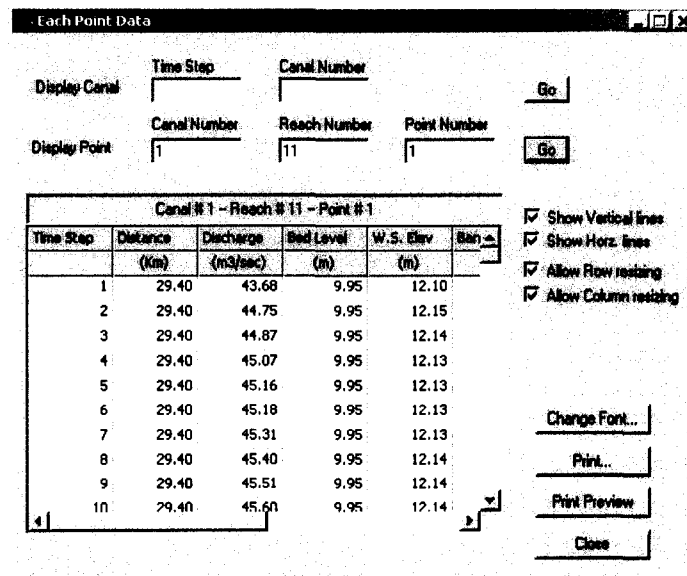


Figure 7.15  
Water level data for a given point over time

The charts present the water surface levels and the energy grade line level, as well as bed levels, bank levels and the gates in the case of presenting the data of a whole channel. The table presents different types of data, including bed level, water surface levels, velocities, and other data.

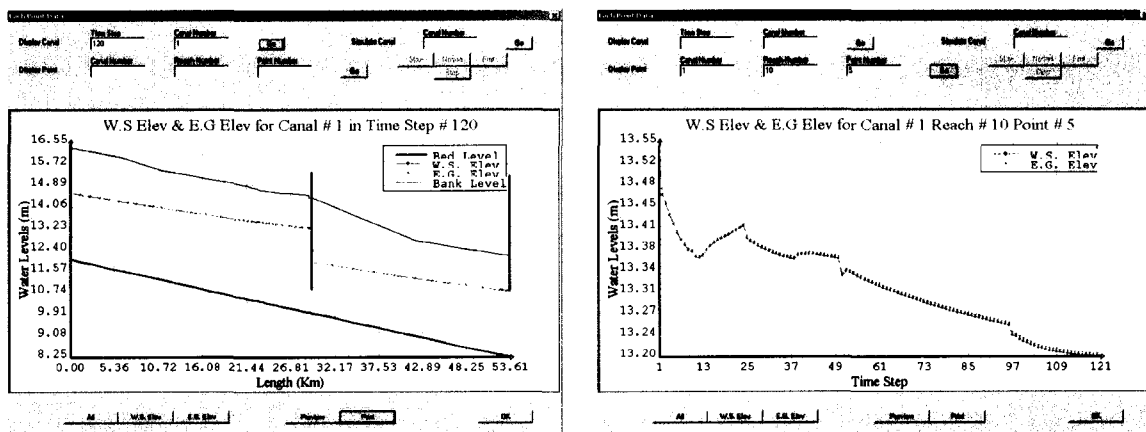


Figure 7.16  
Water levels for a whole channel and for a specific point



### 7.6.2 Regulators Data

The regulators dialog (Figure 7.17) presents the data of all regulators in a channel at a specific time step, or the data of a regulator during all time steps. The presented data are upstream water level, downstream water level, gate opening and discharge.

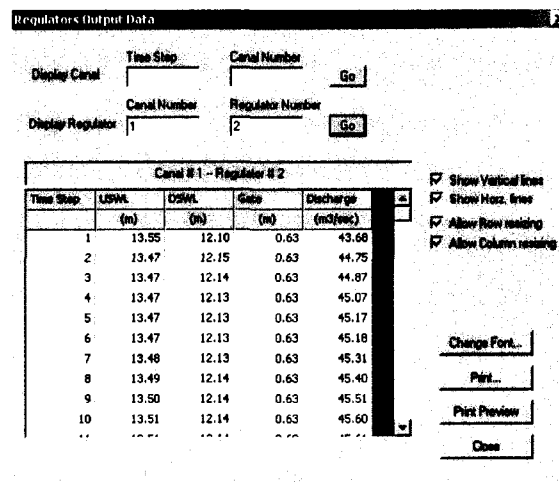


Figure 7.17  
Data of a specific regulator at different time steps

### 7.6.3 Fitness and Cost Data

The fitness values, whether in table or in chart format, are the maximum, average and minimum fitness values. In the chart, these data could be presented together or separately. The objective (cost) data is presented in a similar manner. Figure (7.18) shows an example of the average fitness value per generation.

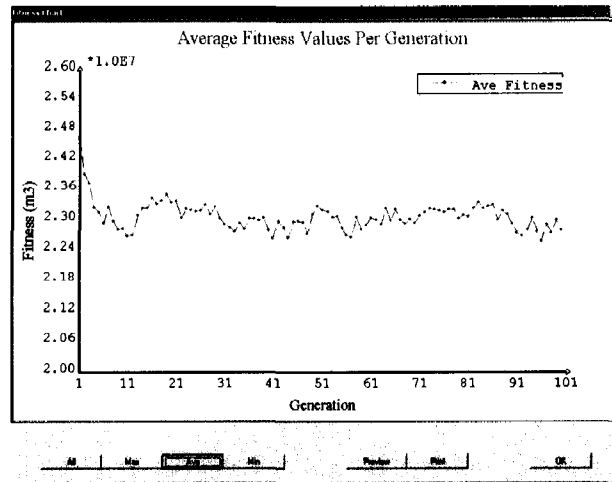


Figure 7.18  
Average fitness values per generation

### 7.6.5 Constraint Violations Data

The constraint violation data table and the chart present the maximum, minimum and average violations per generation for each constraint. In the chart, the value for any constraint could be presented together or separately. If the constraint-handling technique is any additive method (ASP or ALDP), the constraints will be calculated as follows:

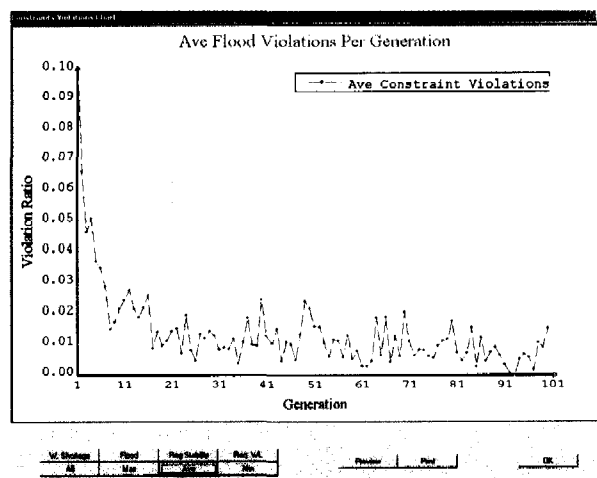


Figure 7.19  
Average flood violation ratio per generation

- Water Shortage (Feddan)
- Flood (m)
- Regulator Stability (Feddan)
- Required Water Level (m<sup>3</sup>)

If any other constraint handling method is used, the constraint violations will be normalized by dividing by the maximum possible violations. Figure (7.19) shows an example of average flood violation ratio per generation.

### 7.6.6 Feasible Solutions

The number of feasible solutions can be presented in table and chart format. The number of solutions that satisfy each constraint, and the number of solutions that satisfy all constraints (feasible solutions) are presented. Regarding chart format, and as in other charts, the data for the various constraints can be presented together or separately. Figure (7.20) shows an example of a chart showing the number of feasible solutions per generation.

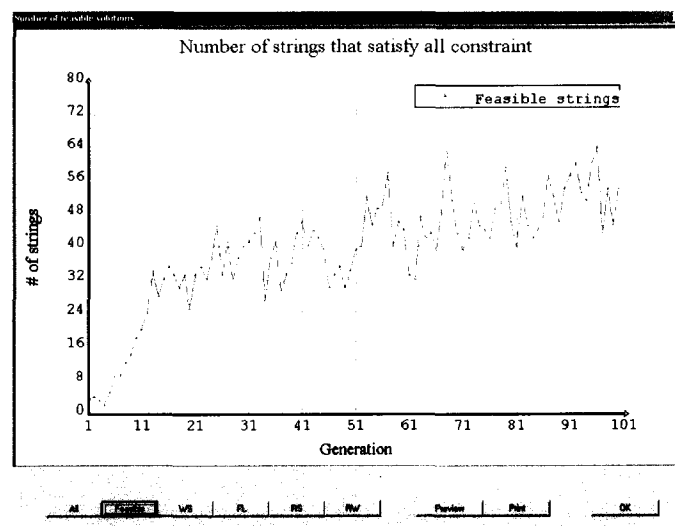


Figure 7.20  
Number of feasible solutions per generation

### 7.6.7 STS average probabilities

As described in Chapter 5, the population average probabilities for selecting based on objective and constraint violation in STS technique can be used as an indicator of the quality of the solution. An example of this chart is shown in Figure 7.21.

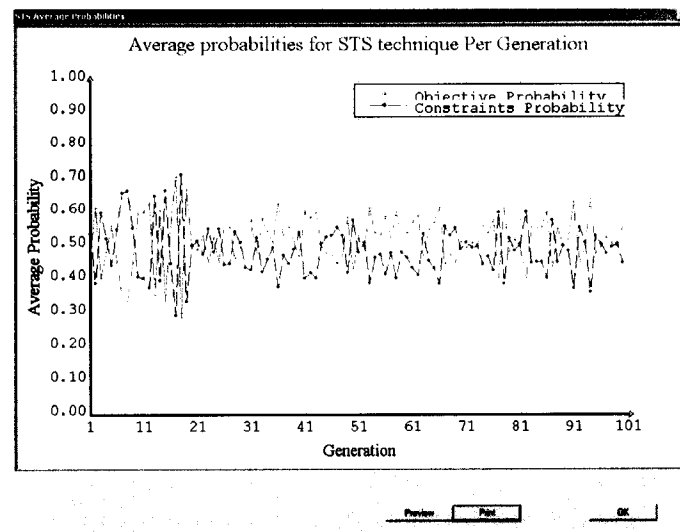


Figure 7.21  
Average probabilities for STS method

### 7.6.8 Final Report

The final report includes a final report about the optimization model, and a final report about the unsteady flow simulation model. Figure 7.22 presents an example of the optimization model final report and unsteady flow simulation model report. The data in the optimization model final report includes:

- The number of feasible solutions in the whole run and in the last generation.
- The best feasible solution in the whole run and in the last generation.
- The average constraint violations in the last generation.

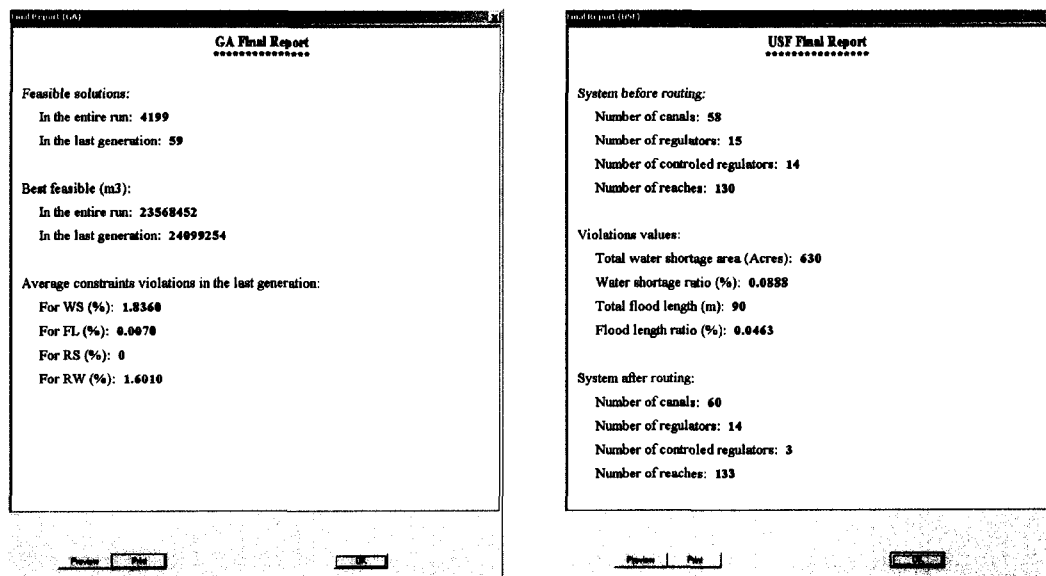


Figure 7.22  
 Final reports of an optimization model and an unsteady flow simulation model

The data in the unsteady slow simulation report includes:

- System data at the beginning and at the end of the routing. This data includes:
  - Number of canals
  - Number of regulators
  - Number of controlled regulators (that are not free opened)
  - Number of reaches
- The amount of water shortage violation and flood violation
- The ratio between water shortage violation and maximum possible water shortage violation
- The ratio between flood violation and maximum possible flood violation

## CHAPTER 8

### CONCLUSIONS AND RECOMMENDATIONS

An optimization model for determination of an efficient management strategy for an irrigation canal network has been developed. The objective of this model is to minimize the total water consumed in the network while satisfying four constraints, which are:

- No water shortage at any point in the network at any time.
- No flood at any point in the network at any time.
- The difference between the upstream water level and the downstream water level of any regulator at any time is less than the maximum allowable difference; to ensure regulator stability.
- The water levels in the network at the end of the routing are sufficient for the start of subsequent irrigation period.

The decision variables for the model are the gate openings and the boundary conditions. Gate openings include initial gate openings at the beginning of the simulation period and gate operations during the simulation. The boundary conditions include the water level at the upstream end of the network and the discharges at the downstream end of each channel. The model is most appropriate for relatively short-term irrigation periods, so the simulation period is typically a few days long, and the constraints are checked at relatively small time intervals (generally one hour or less).

A genetic algorithm (GA) was used to search for efficient solutions to the optimization problem. It is a suitable and efficient optimization tool for this model based on the complexity of the problem. Real representations are used to encode the decision variables. Different versions of binary tournament selection (with and without superiority of feasible solution and a stochastic form) are used in the model. Also, the model uses blend crossover and uniform mutation during GA procedure.

An unsteady flow simulation model was used to evaluate each potential solution in the GA. This model solves the complete Saint Venant equations using an implicit numerical scheme.

The model was applied to a case study in Egypt involving a large-scale irrigation network. Three different scenarios of this case study, representing different levels of difficulty with different number of decision variables were investigated.

GA parameters were tested with the model for the three scenarios. The GA parameters tested were: crossover probability, mutation rate, blend crossover extension, and population size. Based on the results, values for each parameter were recommended for various levels of difficulty of these scenarios.

Different constraint-handling techniques were tested within the current model. Among these techniques, two are newly proposed techniques, while the others are from the literature. The suitable technique that should be used with different scenarios based on the complexity of the scenario, and the number of decision was recommended. The results showed that new proposed techniques outperform the other techniques for two of the three suggested scenarios. A suggestion was made to check the quality of one of the proposed techniques (STS - Stochastic tournament selection) in the absence of information about the actual optimal solution.

The uncertainty in crop distribution and consumption water rates of the crops is incorporated into the search procedure to identify more robust solutions. A Chance-Constrained Genetic Algorithm (CCGA) was used to handle the uncertainty. Latin Hypercube Sampling (LHS) was used within the model. The model shows that CCGA could be used to achieve more reliable solutions at the expense of a small increasing of the objective function value. Also, it proves relatively small LHS samples (10 realization to evaluate each solution) produce good results.

A user-friendly interface is developed to aid the decision maker in using the model.

The computational effort is between two and four hours for 100 generations for different scenarios of the case study using a PC with Pentium 4 processor (2.0 G Hz. with 512 MB RAM). For one of the constraint-handling techniques (self-adaptive) and for the CCGA model, the computational time is expensive (possibly exceed 24 hours depending on the number of realizations or the size of self-adaptive populations).

Recommended future works include the following:

- Some features should be added to the simulation model to make it more applicable to a variety of irrigation canal networks. For example, the program should handle hydraulic structures other than sluice gates. Also, the model should have an option to enter natural the cross sectional data rather than requiring prismatic ones, since the channels may deviate from the design cross-sections in some locations.
- Regarding the GA parameters, adaptive and self-adaptive parameter specifications should be studied with the model, to check if better results can be achieved.
- Regarding the constraint handling techniques, the STS technique should be studied further to ensure it is able to handle different scenarios of the model. Also, new techniques based on maintaining the feasibility of the solutions should be added to the model. The idea of these techniques is to check the situation downstream of each regulator and suggest an operation if the situation is close to violating one of the constraints, and these suggested operations will be added to the basic operation that are defined by the user.
- Methods to reduce the computational effort, especially for the CCGA model should be investigated.



**CURRICULUM VITA**  
for  
**Talaat Taher El Gamel**

**DEGREES:**

- ❖ Master of Science (Civil Engineering – Hydraulics & Water resources), Old Dominion University, Virginia, USA, May 2001
- ❖ Diploma of Higher Studies (Civil Engineering - Hydraulics), Alexandria University, Alexandria, Egypt, June 1991
- ❖ Bachelor of Science (Civil Engineering), Alexandria University, Alexandria, Egypt, May 1987

**PROFESSIONAL CHRONOLOGY:**

Ministry of Public Works & Water Resources, Egypt.  
Engineer, Feb 1988- Present

**COURSES TAUGHT DURING LAST FIVE YEARS:**

Water Resources Management  
Statistical Analysis & Design  
Open Channel Flow  
System Design  
Scientific Computing  
Urban Storm Water Hydrology  
Water Quality Management  
Hydraulic Engineering  
Water Quality Modeling  
Finite Element Analysis  
Topics In Finite Elements  
Stochastic Decision Methods  
Geographic Information Systems  
Engineering Optimization