# Large-scale Asset Renewal Optimization: GAs + Segmentation versus Advanced Mathematical Tools

by

Roozbeh Rashedi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Civil Engineering

Waterloo, Ontario, Canada, 2011

## Author's Declaration

***Roozbeh Rashedi***

## Abstract

Capital renewal is an essential decision in sustaining the serviceability of infrastructure. Effectively allocating limited renewal funds amongst numerous asset components represents a large-scale combinatorial optimization problem that is difficult to solve. While various mathematical optimization techniques have been presented in the published literature, they are not very effective in handling the complexities and huge calculations related to large scale problems. More recently new evolutionary-based techniques, such as genetic algorithms (GA) have been introduced for finding near-optimum solutions to large-scale problems. Experimenting with this technique for asset renewal problems has revealed that GAs performance rapidly degrades with problem size. For instance, a previous research by Hegazy and Elhakeem (2010), could improve fund allocation for only a portion of total existing components (maximum of 8000 asset components) with degradation in optimization performance by increasing number of components. To address larger scale problems, this research investigates both evolutionary and advanced mathematical optimization techniques and seeks a goal of handling models consist of at least 50,000 asset components.

To enhance the performance of GAs for large-scale optimization problems, three aspects were considered: (1) examining different problem formulations such as integer, on-shot-binary, and step-wise binary formulation; (2) experimenting with commercial GA-based tools; and (3) introducing an innovative segmentation method to handle groups of smaller size problems, and then integrating the results. To identify the best segmentation method, similarity-based segmentation is compared to random segmentation and was found to have superior performance. Based on the results of numerous experiments with different problem sizes and comparison with previous results obtained by Hegazy and Elhakeem (2010) from the same prototype used in this study, the GAs + Segmentation approach is found to handle a problem size of 50,000 components, with better solution quality (improved optimum solution), and no noticeable degradation of optimization performance by increasing the problems size.

In addition to evolutionary algorithms, performance of one of the advanced mathematical programming tools (GAMS), and its powerful optimization engine (CPLEX), are investigated. For the mathematical representation of the asset renewal problem, best formulation is selected with regard to the definitions of easy-to-solve integer programming (IP) formulations.  To reduce internal calculations, the GAMS mathematical model is coded to interact with original spreadsheet data by using GAMS data exchange (GDX) files. Based on experimentations, using advanced mathematical tools with strong (easy-to-solve) IP formulations, improved the solution quality even further in compare to GA + Segmentation.

In conclusion, this research investigated both evolutionary and advanced mathematical optimization techniques in handling very large-scale asset renewal problems, and introduced effective models for solving such problems. The developed models represent a major innovative step towards achieving large cost savings, optimizing decisions, and justifying fund allocation decisions for infrastructure asset management. While the focus of this research is on educational buildings, the developed optimization models can be adapted to various large-scale asset management problems.

## Acknowledgements

First and foremost I offer my sincerest gratitude to my supervisor, Dr. Tarek Hegazy, who has supported me throughout the course of my program with his patience and knowledge, and also for his extensive assistance, generous support and close supervision. It has been a great honor to work with him and to learn from his experience.

I would also like to thank my family for their emotional support and encouragement throughout the period of my studies.

***Roozbeh Rashedi***

# Table of Contents

# List of Figures

# List of Tables

# ACRONYMS

| | |
|---|---|
| \$CRC | Total replacement cost |
| $B_{kj}$ | Available budget for segment j in year k |
| $S_i$ | Inspected severity of deficiency i |
| $Y_{jk}$ | Repair action for instance j at year k |
| $w_i$ | Weight of deficiency i |
| $ARDI_k$ | After repair deterioration index (repair at year k) |
| $B/C$ | Benefit to Cost Ratio |
| $CrI_i$ | Criticality index of instance i |
| $DI_N$ | Network Deterioration Index |
| $DI_{jk}$ | Deterioration index of instance j at year k |
| $EP_{j0}$ | Expected performance with no repair |
| $EP_{jk}$ | Expected performance of instance j after repair in year k |
| $IE_{jk}$ | Improvement effect of repair in year k on instance j |
| $IP$ | Integer Programming |
| $LCCA$ | Life Cycle Cost Analysis |

| | |
|---|---|
| $LP$ | Linear Programming |
| $MIP$ | Mixed Integer Programming |
| $NLP$ | Nonlinear Programming |
| $NS$ | Number of segments |
| $RC_j$ | Relative Criticality of segment j |
| $RIF_j$ | Relative importance factor of instance j |
| $RS$ | Repair Scenario |
| $SCr_j$ | Criticality index of segment j |
| $SRC_{jk}$ | Segment j repair cost in year k |
| $SS$ | Segment Size |
| $TB_k$ | Total available budget at year k |
| $TC(\%)$ | Total Cost (percentage of full replacement) |
| $TDSB$ | Toronto District School Board |
| $Z$ | Instance relative area |

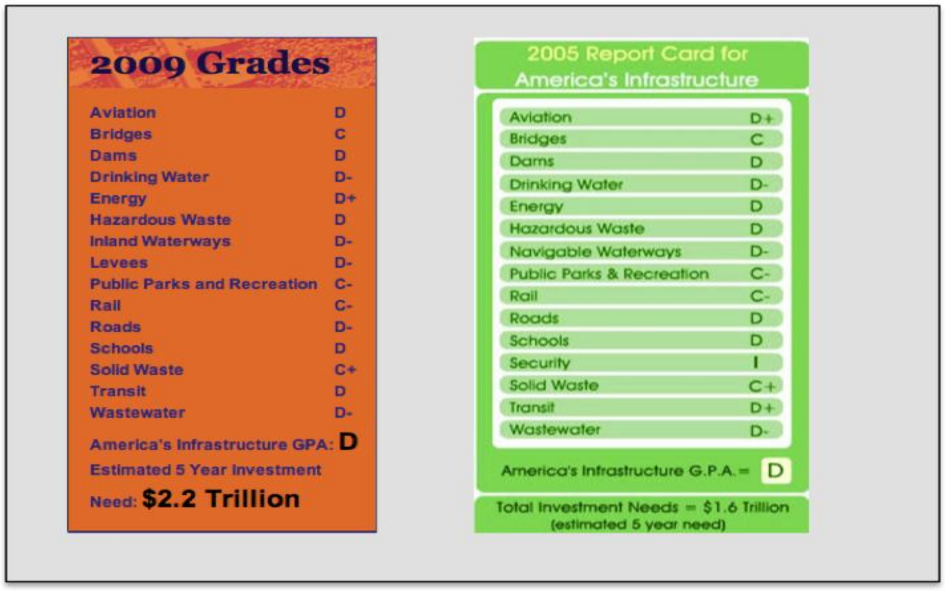
# Chapter 1

# Introduction

## 1.1. General

In modern societies, reliable and efficient public infrastructure is central to achieving high quality of life and prosperous economy. Sustaining the serviceability of infrastructure is a challenging task due to continuous asset deterioration over time, and the lack of renewal funding. Asset management systems (AMSs) have been developed to help decision makers when and how to repair the existing infrastructure cost-effectively.

Asset management systems have been proposed as tools helping asset managers to reduce the maintenance costs and improve serviceability of public assets. Report cards for America's infrastructure estimated the investment needs of $2.2 trillion in 2009, which had increased from $1.6 trillion in 2005 (ASCE 2009; ASCE 2005) (Figure 1.1). In Canada, the municipal infrastructure deficit was estimated to be $120 billion in 2007, which is doubled since 2003. These large backlogs of maintenance needs are accumulated from previous years and are compounded by the demands for new facilities to serve the population growth (Federation of Canadian Municipalities (FCM) 2007). In addition, most of Canada's infrastructure has passed 50% of their useful service lives. Therefore, a large number of aged facilities exist that requires urgent renewal actions (Statistics Canada 2009). Figure 1.2 shows Canada's municipal deficit profile for infrastructure. Accordingly, huge investment demands with municipal infrastructure deficits will result in limited available repair funds for large network of deteriorated assets.



**2009 Grades**

| | |
|---|---|
| Aviation | D |
| Bridges | C |
| Dams | D |
| Drinking Water | D- |
| Energy | D+ |
| Hazardous Waste | D |
| Inland Waterways | D- |
| Levees | D- |
| Public Parks and Recreation | C- |
| Rail | C- |
| Roads | D- |
| Schools | D |
| Solid Waste | C+ |
| Transit | D |
| Wastewater | D- |

America's Infrastructure GPA: **D**

Estimated 5 Year Investment Need: **$2.2 Trillion**

**2005 Report Card for America's Infrastructure**

| | |
|---|---|
| Aviation | D+ |
| Bridges | C |
| Dams | D |
| Drinking Water | D- |
| Energy | D |
| Hazardous Waste | D |
| Navigable Waterways | D- |
| Public Parks & Recreation | C- |
| Rail | C- |
| Roads | D |
| Schools | D |
| Security | I |
| Solid Waste | C+ |
| Transit | D+ |
| Wastewater | D- |

America's Infrastructure G.P.A. = D

Total Investment Needs = $1.6 Trillion (estimated 5 year need)

**Figure 1.1: Report cards for U.S. infrastructure (ASCE 2009; ASCE 2005)**

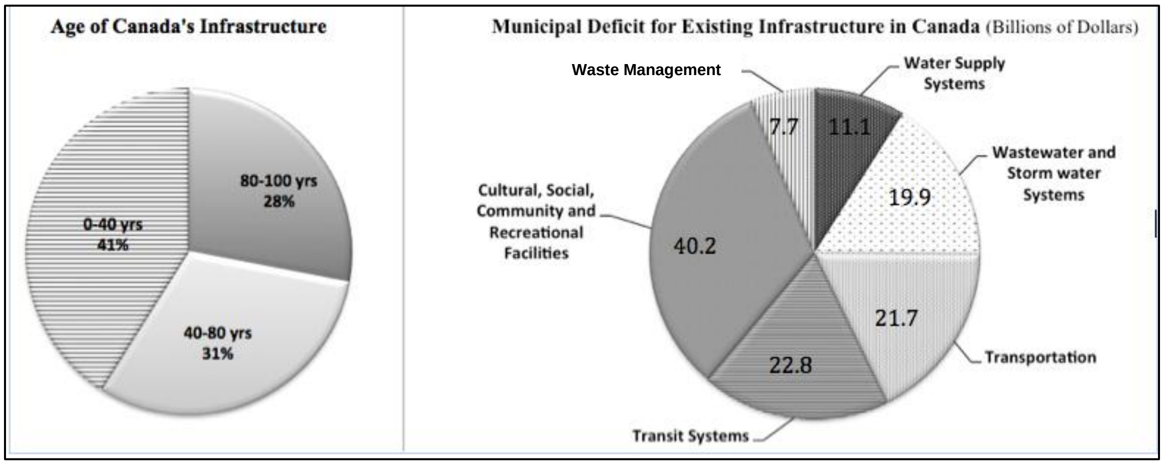


**Figure 1.2: Municipal deficit and the age of existing infrastructure in Canada (Statistics Canada 2009; FCM 2007)**

To manage a large network of assets and their diverse components, an asset management system integrates four functions: 1) condition assessment, 2) deterioration modeling, 3) repair modeling, and 4) life cycle cost analysis (LCCA) (Elhakeem and Hegazy 2010). Each of these components plays an important role in developing an effective AMS. Asset inspection and condition assessment provides data on the current condition of the asset inventory. In addition, deterioration modeling predicts the behavior of an asset and the future conditions of individual components along the planning horizon (Hegazy et al 2004; Elhakeem and Hegazy 2010). Repair modeling is essential to make decisions about suitable repair strategies for each asset. With efficient mechanisms for condition assessment, deterioration modeling, and repair modeling, an AMS must integrate these functions with a detailed life cycle cost analysis. LCCA considers each sub-decision within the planning horizon related to which, when, and how to repair each component cost-effectively. To answer these basic questions and to find the optimum solution to an asset renewal problem, an optimization model needs to be created and solved for all of the components. However, due to the diversity of components in most civil infrastructure systems and the complexity of their deteriorating behaviors and repair needs, it is extremely difficult to handle this large-scale and complex optimization problem.

Optimization is an operation research technique (ORT) used to enhance decision making and to foster effectiveness in solving problems. While different optimization techniques have been published in the literature, few have focused on methods that solve real-world, large-scale asset renewal optimization problems. With deteriorating infrastructure and stringent repair funds development practical methods to handle asset renewal problems is necessary.

## 1.2. Research Motivation

This research has three main motivations: the importance of capital asset renewal; the lack of decision support tools; and the need for practical approaches to optimize renewal decisions for large scale networks of asset. These are briefly described as follows:

**Capital Asset Renewal is a Key Infrastructure Management Function**

Asset managers have two main functions in caring for their asset inventory: preventive/reactive maintenance, and capital asset renewal (Figure 1.3) (Elhakeem and Hegazy 2010). While day-to-day asset operation is supported by the maintenance function, capital asset renewal upgrades the assets to enhance functionality and/or keep the asset in good shape for long-term uninterrupted use. Although capital asset renewal does not involve daily maintenance activities, it has great significance in asset management. A study by Vainer (2001) indicates the importance of capital asset renewal by demonstrating that the expenditure on capital renewal is almost equal to expenditure on maintenance and repair in U.S. and Canada (Table 1.1).

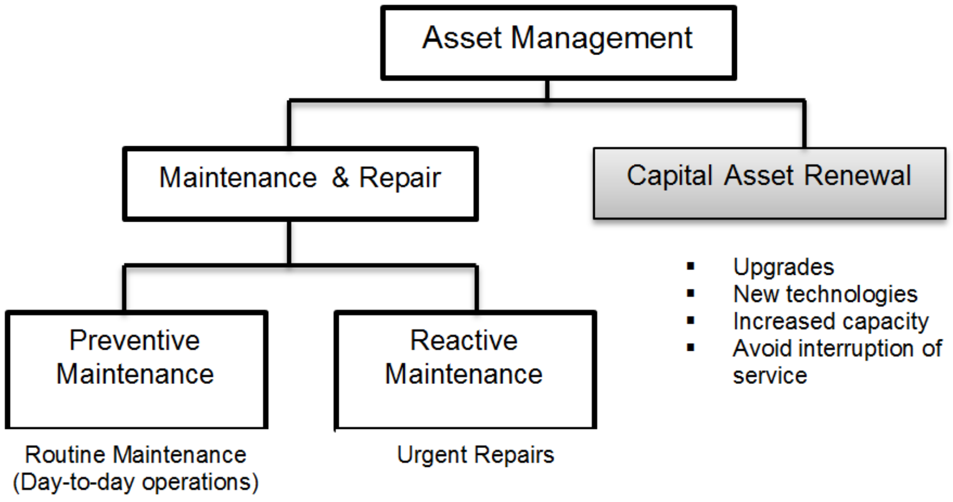


**Figure 1.3: Asset management dimensions**

**Table1.1: Maintenance, Repair, and Capital Renewal in Canada and U.S. (Vainer 2001)**

| Parameter | Canada | USA | Total |
|---|---|---|---|
| Maintenance and Repair | 58.80 | 588.00 | 646.80 |
| Capital Asset Renewal | 45.60 | 456.00 | 501.60 |
| Total | 104.40 | 1,044.00 | 1,148.40 |

Note: All numbers are in billions of dollars

**Lack of Asset Renewal Decision Support Tools (DSTs)**

While capital asset renewal is a key function in asset management, few tools have been developed to support capital renewal decisions, as compared to maintenance decisions. A survey among commercial municipal asset management systems (Halfway et al. 2005) revealed that the majority of such systems focus on supporting day-to-day activities (i.e., maintenance), and only a few offer limited support for long-term renewal planning. Furthermore, most of these systems do not support many fundamental asset management functions such as deterioration and repair modeling or repair prioritization. While existing decision support tools are very limited in scope and involve only partial solutions, there is a huge demand in the industry for an integrated life cycle cost analysis that considers both repair-type decisions and repair-timing decisions within an optimization framework (Kyle et al. 2002; Zhang 2006; Elhakeem and Hegazy 2010).

## The Need to Address Very Large-Scale Asset Renewal Problems

Asset renewal decisions are mostly considered in the category of combinatorial optimization problems, and are extremely difficult to solve (Csiszar 2007). These types of optimization problems exhibit exponential complexity as the number of variables increase. For instance, consider the case of a small asset renewal optimization problem, such as a school board that owns 100 schools (assets). If each school has a list of 100 components (HVAC, boilers, windows, etc.), and even a small number of the building components (say 0.5%) requires renewal actions along a planning horizon (e.g. five years), by having six repair alternatives (no repair, or repair in one of the five years), the possible combination of renewal actions (i.e. solution-space size) becomes $(2 \times 6)^{100 \times 100 \times 0.005} = 9.1E + 53$, which is an extremely large and prohibitive . Finding an optimum or near optimum by trial and error becomes complex and time consuming. For instance, a recent research on asset renewal optimization by Hegazy and Elhakeem (2010), showed degradation in optimization performance by increasing number of components that caused an optimization failure after reaching 8000 components. Since most of the optimization techniques suffer from such performance degradation by increasing the problem size (Csiszar 2007), it is necessary to develop practical mechanisms to address large and complex asset renewal problems.

## 1.3. Research Goal and Objectives

The preliminary goal of this research is to investigate practical approaches in dealing with very large-scale asset renewal optimization problems, involving at least 50,000 components. The research investigates both mathematical and evolutionary optimization methods and the conditions under which they can be applied to large-scale asset renewal optimization problems. Objectives to achieve the research goal are:

- Investigate the performance of different optimization formulations such as integer, one-shot-binary, and step-wise-binary formulations in dealing with large-scale asset renewal optimization problems by using evolutionary algorithms, in particular, genetic algorithms (GAs).
- Investigate the improvements of optimal solutions obtained from evolutionary-based optimization, caused by problem segmentation approaches.
- Develop an automated mechanism for segmenting asset renewal optimization and test this on different sized real-life problems.

- Experiment with advanced mathematical optimization tools, such as GAMS/CPLEX solver, and investigate the performance of integer programming (IP) formulation of asset renewal involving tens of thousands of components.
- Compare and discuss the results of both evolutionary and mathematical optimization approaches.

## 1.4. Research Methodology

The methodology to achieve the aforementioned objectives is illustrated schematically in Figure 1.4. The following are brief descriptions of each step:

1. **Literature Review**: An extensive literature review on existing AMSs to support capital asset renewal is conducted. The LCCA function of existing AMSs and their optimization models are surveyed for asset renewal planning. Based on the literature review, limitations of current optimization models and suggestions for improving their efficiency are identified. The evaluation of existing optimization models is then used to define the best way of modeling the asset renewal problem.

2. **Modeling the Asset Renewal Problem**: After identifying the practical aspects of LCCA function of existing AMSs, a bilevel optimization model is utilized to support decisions regarding educational facilities (i.e., school buildings network). Different parameters such as objective function, budget limits, repair alternatives, decision variables, and planning horizon are defined based on preferences of the decision

makers. After modeling the asset renewal problem, evolutionary-based optimization techniques, mathematical optimization methods, and the conditions under which they can be applied to large-scale asset renewal problems are investigated.

3. **Evolutionary Algorithms (EAs)**: After modeling the asset renewal problem a non-traditional evolutionary-based algorithm, i.e. genetic algorithm (GA), which is one of the most common algorithms to solve large-scale and combinatorial problems (Goldberg 1991; Tong et al. 2001; Elbeltagi et al. 2005; Hegazy and Elhakeem 2010), is investigated. Different formulations and several optimization tools are examined to identify the best formulation and optimization tool to handle the large-scale asset renewal problem. Afterward, a problem segmentation method is introduced and investigated with genetic algorithm to improve the optimization efficiency for very large-scale capital asset renewal problems.

4. **Mathematical Programming (MP)**: In addition to EAs, performance of analytical optimization algorithms (i.e., mathematical optimization) is also investigated. The asset renewal problem is formulated as integer programming (IP) and problems of varying sizes are optimized within GAMS (General Algebraic Modeling System) platform. A powerful optimization engine, GAMS/CPLEX solver, is also used to handle very large-scale asset renewal problems.

5. **Testing, Validation, and Conclusions:** The renewal model for a network of school buildings administrated by Toronto District School Board (TDSB) is selected for the case study. An automated mechanism developed for handling the large-scale asset renewal problems is then tested on different size problems. For validating the final solutions, performances of both evolutionary-based and mathematical optimization techniques are investigated and results are compared with each other and also with previous optimization results obtained from the same prototype by Elhakeem and Hegazy (2010).

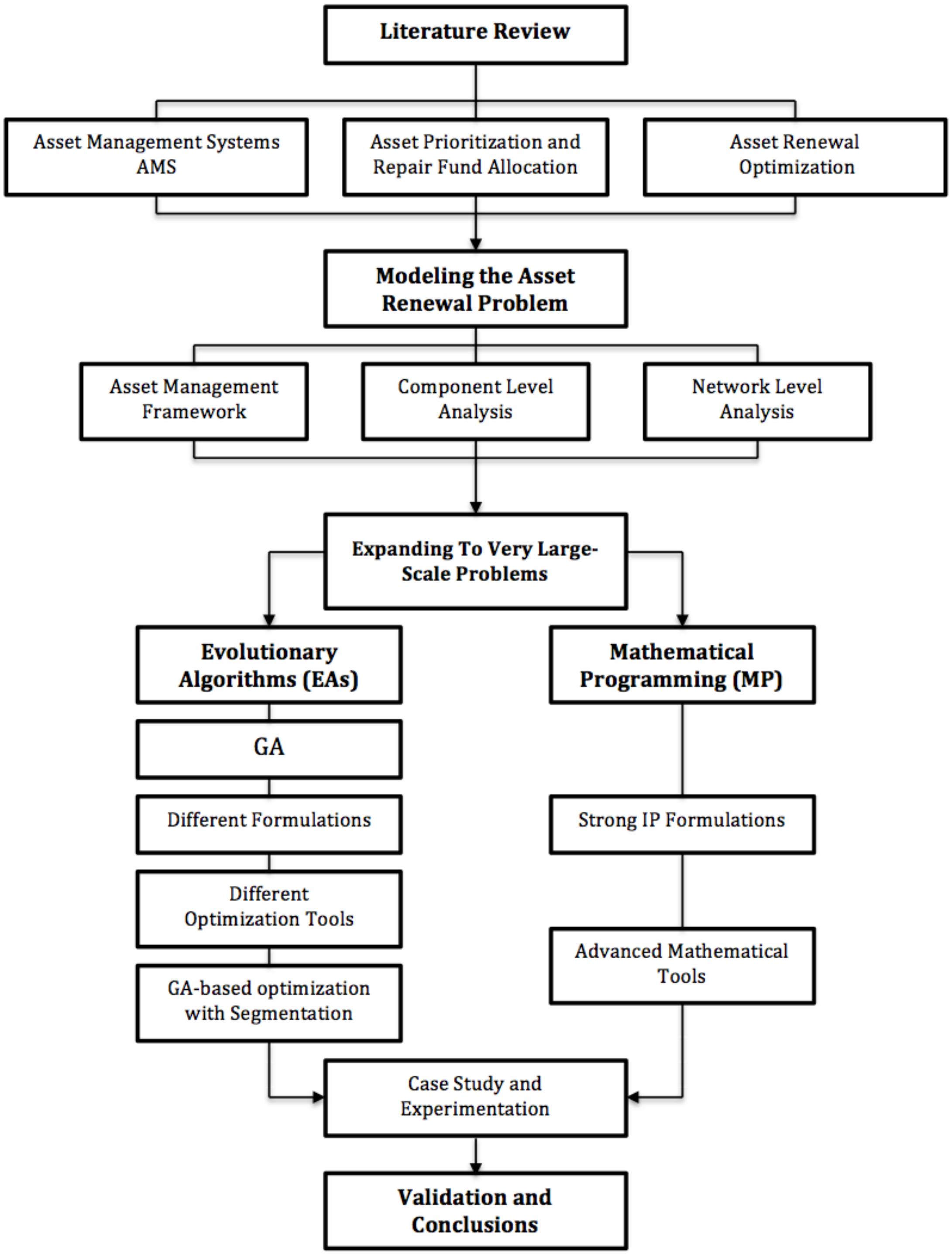

Literature Review
Asset Management Systems AMS
Asset Prioritization and Repair Fund Allocation
Asset Renewal Optimization
Modeling the Asset Renewal Problem
Asset Management Framework
Component Level Analysis
Network Level Analysis
Expanding To Very Large-Scale Problems
Evolutionary Algorithms (EAs)
GA
Different Formulations
Different Optimization Tools
GA-based optimization with Segmentation
Mathematical Programming (MP)
Strong IP Formulations
Advanced Mathematical Tools
Case Study and Experimentation
Validation and Conclusions

**Figure 1.4: Schematic Diagram for Research Methodology**

## 1.5. Thesis Organization

**Chapter 2** presents a literature review of asset management systems, traditional and most recent methods of asset prioritization and repair fund allocation, and recent optimization techniques for handling asset renewal problems.

**Chapter 3** introduces the asset management framework and the life cycle cost analysis model used for TDSB problem and also discusses the previous optimization results obtained by using genetic algorithms. This chapter explores the methodology for expanding to very large-scale real life problems.

**Chapter 4** presents three formulations for the asset renewal problem and investigates the performance of each to determine the best possible formulation. Also, two GA-based optimization tools, Evolver and Risk Solver Platform, are tested and compared to select the best tool for larger size optimizations. Chapter 4 introduces segmentation and investigates its effectiveness of GA-based optimization with segmentation in handling very large-scale asset renewal problems.

**Chapter 5** investigates the performance of advanced mathematical tools, such as GAMS and IBM ILOG optimizer, and compares the results obtained by mathematical approach with segmented GA-based optimization.

**Chapter 6** presents conclusions and future works.

# Chapter 2

# Literature Review

## 2.1. Introduction

Assets management systems (AMSs) are tools to support manager, organization owners, and governments to cost-effectively operate their assets at the highest possible level of performance and level of service. In this chapter a detailed review of the asset management systems, asset prioritization and repair fund allocation, and asset renewal optimization techniques is presented.

## 2.2. Asset Management Systems

In essence, asset management systems are systematic approaches to achieve the highest level of benefit and satisfaction from operating faculties by their owners and users respectively. These benefits can be fiscal and/or functional. Based on Transportation Association of Canada (TAC), asset management is defined as "a comprehensive business strategy employing people, information and technology to effectively allocate available funds amongst valid and competing asset needs" (TAC 1999).

Asset management systems also integrate different areas of science such as engineering, economy, mathematics, business, and computer science. In a more detailed description of asset management, Federal Highway Administration (FHWA 1999) defines AMS as "a systematic process of maintaining, upgrading and operating physical assets cost effectively. It combines engineering and mathematical analysis with sound business practice and economic theory. Asset management systems are goal driven and like the traditional planning process, include components for data collection, strategy evaluation, program selection, and feedback. The asset management model explicitly addresses integration of decisions made across all program areas".

Base on FHWA definition of AMS, an asset management process involves different systems that are integrating within an asset management framework (Figure 2.1). In this AMS framework based on the organizational goals and policies, serviceability expectations, available budgets and resources, and feasible alternatives are developed by using the information obtained from asset inventory, condition assessment, and performance prediction. Next, alternative fund allocation scenarios are evaluated based on their impact on system performance and with regards to financial and serviceability constraints that are defined by the organizational policies and goals. This information is used by asset managers to prepare short-term plans, i.e., day-to-day operations and routine maintenance, and/or long-term plans, i.e., capital asset renewal. After implementing the plans, the performance of the individual assets in monitored and the data from performance monitoring is used as feedback to verify the assumptions, assessment methods, and predictions made at the planning stage (FHWA 1999; Flintsch and Chen 2004).

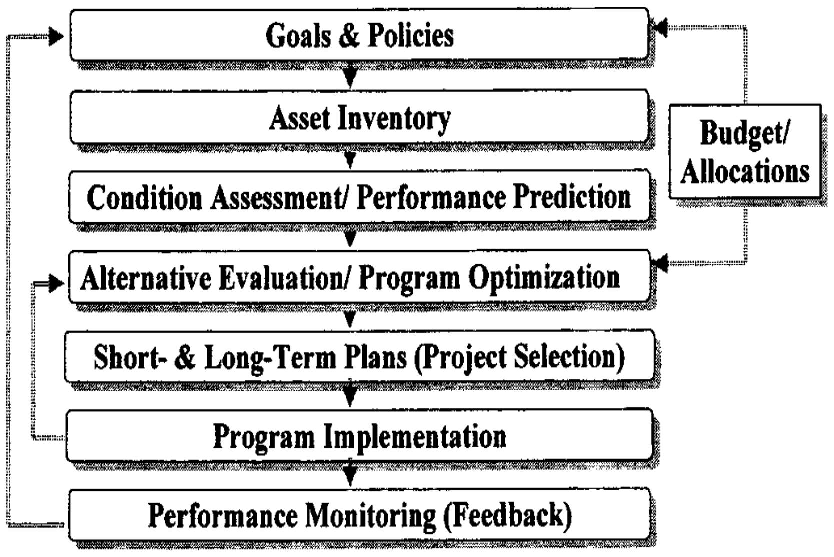

Goals & Policies
Asset Inventory
Condition Assessment/ Performance Prediction
Alternative Evaluation/ Program Optimization
Short- & Long-Term Plans (Project Selection)
Program Implementation
Performance Monitoring (Feedback)
Budget/ Allocations

**Figure 2.1: Generic asset management framework based on FHWA (Flintsch and Chen 2004)**

## 2.3. Asset Prioritization and Repair Fund Allocation

Handling the scarcity of financial resources is a great challenge for managers and decision makers. Accumulated backlog of past maintenance and rehabilitation and the construction of new facilities result in steadily fund deficit problems for federal governments (Hudson et al 1997, Federation of Canadian Municipalities 2007). An ideal infrastructure management system will use all available funds optimally while maximizing the performance of the system (Hudson et al 1997). Accordingly, maintaining the serviceability of the asset and maximizing the benefit over cost ratio (B/C) for complex and large-scale asset renewal problems are big challenges for asset managers (AL-Battaineh et al. 2005). Different techniques have been introduced in literature to support infrastructure management functions, such as condition assessment, performance analysis, need analysis, prioritization, and optimization. Advanced artificial intelligence (AI) techniques such as fuzzy logic, neural networks, and genetic algorithms are some of these techniques that are widely used to support asset management (Halfway 2008; Hegazy 2004; Tong et al. 2001). A summary of AI techniques used for different asset management functions is shown in Table 2.1. Based on this table, the predominant method of optimization for tradeoff analysis is Genetic Algorithm (GA), which is used in different published literatures such as Fwa et al. (1996), Liu et al. (1997), Pilson et al. (1999), and Shekaharan (2000). Also few people worked on developing effective techniques for treatment selection or asset prioritization such as Grivas and Shen (1995), or Martinelli et al. (1995).

To prioritize assets, different methods from simple ranking procedures to complex optimization mechanisms can be used (Elkhakeem and Hegazy 2010). All of the asset prioritization methods seek answers to three main questions related to MR&R strategies, "which, when, and what", during the planning horizon (Hudson et al. 1997).

Ranking methods such as simple subjective ranking, parameter based ranking, or parameter/economic based ranking, are simple heuristic approaches used frequently by asset managers, but are not providing good results in terms of optimality due to the subjective assessment of data. In the process of ranking, various scoring criteria, such as condition/structural integrity, expected lifetime of existing infrastructure, consequence of delay, environmental factors, and/or resource allocation factors are considered for the prioritizing assets (Hudson et al. 1997).

**Table 2.1: Summary of advanced AI tools used for different asset management functions**

| | | Asset performance | | Needs analysis | | Tradeoff analysis | |
|---|---|---|---|---|---|---|---|
| Technology | Reference | Condition assessment | Performance prediction | Project selection | Treatment selection | Prioritization schemes | Optimization techniques |
| Artificial neural networks | | | | | | | |
| | Pant et al. (1993) | √ | | | | | |
| | Kaseko and Ritchie (1993) | √ | | | | | |
| | Hajek and Hurdal (1993) | | | | √ | | |
| | Fwa and Chan (1993) | | | | | √ | |
| | Eldin and Senouci (1995) | √ | | | | | |
| | Flintsch et al. (1996) | | | √ | | | |
| | Razaqpur et al. (1996) | | | | | | √ |
| | Cattan and Mohammadi (1997) | √ | | | | | |
| | Huang and Moore (1997) | | √ | | | | |
| | Alsugair and Al-Qudrah (1998) | | | | √ | | |
| | La Torre et al. (1998) | | √ | | | | |
| | Owusu-Ababia (1998) | | √ | | | | |
| | Shekharan (1998) | | √ | | | | |
| | Wang et al. (1998) | √ | | | | | |
| | Van der Gryp et al. (1998) | √ | | | | | |
| | Martinelli and Shoukry (2000) | √ | | | | | |
| | Lou et al. (2001) | | √ | | | | |
| | Farias et al. (2003) | | √ | | | | |
| | Felker et al. (2003) | | √ | | | | |
| | Fontul et al. (2003) | √ | | | | | |
| | Lee and Lee (2004) | √ | | | | | |
| | Lin et al. (2003) | √ | | | | | |
| | Sadek et al. (2003) | √ | | | | | |
| | Yang et al. (2003) | | √ | | | | |
| Fuzzy logic systems | | | | | | | |
| | Elton and Juang (1988) | √ | | | | | |
| | Zhang et al. (1993) | √ | | | | | |
| | Grivas and Shen (1995) | | | | √ | | |
| | Prechaverakul and Hadipriono (1995) | √ | | √ | | | |
| | Shoukry et al. (1997) | √ | | | | | |
| | Wang and Liu (1997) | | | | | | √ |
| | Fwa & Shanmugam (1998) | √ | | | | | |
| | Cheng et al. (1999) | √ | | | | | |
| | Saitoh and Fukuda (2000) | | | | | | √ |
| | Bandara and Gunaratne (2001) | √ | √ | | | √ | |
| Genetic algorithms | | | | | | | |
| | Fwa et al. (1996) | | | | | | √ |
| | Liu et al. (1997) | | | | | | √ |
| | Pilson et al. (1999) | | | | | | √ |
| | Shekharan (2000) | | √ | | | | |
| | Miyamoto et al. (2000) | | | | | | √ |
| | Chan et al. (2001) | | | | | √ | √ |
| | Hedfi and Stephanos (2001) | | √ | | | | |
| | Ferreira et al. (2002) | | | | | | √ |
| Other hybrid systems | | | | | | | |
| | Ritchie et al. (1991) | √ | | | | | |
| | Chou et al. (1995) | √ | | | | | |
| | Taha and Hanna (1995) | | | | √ | | |
| | Martinelli et al. (1995) | √ | | | | | |
| | Abdelrahim and George (2000) | | √ | | | | |

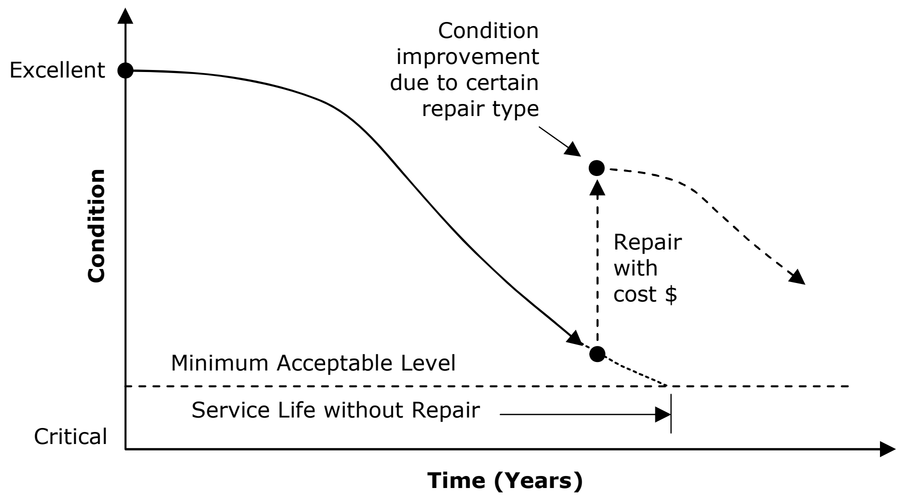


**Figure 2.2: Effect of repair strategy on asset performance (Hudson et al. 1997)**

More complex approaches, such as near-optimization and optimization methods, which are able to have better quality solutions, are finding more and more applications in practice (Cheung et al., 2002: Hegazy 2004; Tong et al. 2001). In addition, near-optimization methods that are mostly based on a marginal cost-effective approach, are finding considerable applications in the road and pavements (Haas 1994). Most of the near-optimum methods select the best repair scenario based on the effective ness of various repairs by calculating the area under the performance curve (Figure2.2) (Hudson et al. 1997). Although ranking and near optimization approaches are practical in some cases, for problems with high level of complexity and importance, optimization is the most promising method. Optimization methods are quantitative techniques of operation research that can enhance the process of decision making by satisfying a specific objective function, such as minimizing cost or maximizing the performance. Optimization methods are used in different areas of infrastructure management such as sewer networks, bridges management, and portfolio management (Halfway 2008; Hegazy 2004; Tong et al. 2001).

A big challenge in optimum allocation of renewal funds is to find the optimum solution amongst numerous feasible solutions. For instance, consider a school board administrates about 100 schools (assets). Even if a small number of the assets requiring renewal actions (say 0.5% of all building components) within 5 years, the number of components for the school board is $100 \times 100 \times 0.005 = 50$. Now, considering each component having six options (no repair, or repair in any of the five years), and assuming three possible techniques to do each repair, then the solution-space size becomes $(2 \times 6)^{100\times100\times0.005} = 9.1E + 53$, which is extremely large and prohibitive. In fact, real-size problems are even much larger due to the fact that many assets are getting older and many components require renewal. As such, optimization size represents a key challenge for life cycle cost analysis and it is necessary to develop optimization mechanisms for handling large-scale problems. The following sections describe the common optimization techniques used in the area of asset management.

## 2.4. Optimization Techniques

Optimization, in general, tries to maximize or minimize an objective function (a goal) by determining the optimum values (quantities) for a set of decision variables respecting a set of constraints. Mathematical and evolutionary-based optimization techniques are two of the most common optimization methods.

### 2.4.1. Mathematical Optimization

Mathematical optimization has three main components: objective function, decision variables, and constraints. An optimization model can be disrobed with regard to the form of decision variables, linearity of objective function or constraint equations, and the amount of uncertainty associated with the data presented in the model. A model can be 'static model' in which the variables are not changes during the optimization process, or a 'dynamic model', which the decision variables change over a multiple period of time during the optimization process. Also, linearity or nonlinearity of a model is based on the way that objective function and constraints (model equations) are defined. Based on the linearity of equations involve in the optimization problem linear programming (LP) or nonlinear programming (NLP) algorithms are used to solves the problem. If one or more variable is in the form of an integer then the model in an integer model. Being all variables equal to integer values results in a pure integer programming problem (IP) and being part of variables equal to integer values results in a mixed integer programming (MIP). In term of certainty, a model can be 'deterministic', which means all values for objective function and decision variables are known with certainty, or it can be 'stochastic', which there is an amount of uncertainty involved in the process of optimization (Thanedar, B.P. 1995; Cook et al. 1997).

One of the most difficult types of optimization problems is the combinatorial optimization problem. These types of problems involve variables with discrete options seeking combinations of these options until an optimum combination is obtained. These problems are very hard to solve because they exhibit exponential complexity as the problem size and the number of variables increase (Csiszar 2007; Elhakeem and Hegazy 2010). Evolutionary Algorithms (EAs) are optimization techniques apply advance search methods to solve combinatorial problems (El Elbeltagi et al. 2005; Hegazy 2004; Tong et al 2001).

### 2.4.2. Evolutionary Algorithms

Evolutionary algorithms are naturally inspired stochastic search methods developed for searching near-optimum solutions to large-scale combinatorial optimization problems (Goldberg 1989). Evolutionary optimization approaches usually mimic the process of biological evolution or the social behavior of species (Elbeltagi et al. 2005). Enhancements in artificial intelligence (AI) lead to development of different evolutionary algorithms, such as genetic algorithm (GA), particle swarm optimization (PSO), ant colony optimization (ACO), and shuffled frog leaping algorithm (SFLA), which are proved to be promising optimization approaches in handling complex engineering problems (Elbeltagi et al. 2005). As discussed in Table 2.1, Genetic Algorithm (GA) is one of the evolutionary-based optimization methods finding different applications in asset management (Hegazy 2004; Tong et al. 2001).

**Genetic Algorithm (GA):** As an effective optimization algorithm with high efficiency in solving different types of problems, GA has been used in different areas of asset management and civil engineering, such as the site-layout optimization of facilities (Cheung et al., 2002; Li and Love, 2000, and Osama et al., 2003), cost optimization and cost trade off problems (Hegazy 1999a), and in resource levelling in construction (Leu et al., 2000; Hegazy, 1999b).

The common conclusion among all the previous researches was the efficiency of implementing GA in solving large-scale and complex problems and arriving at near-optimum solutions. Genetic algorithm was first introduced by Holland in 1975. This type of evolutionary algorithms mimics the process of natural selection based on Darwin's theory of "survival of the fittest". Solution to a given problem is represented in the form of strings called 'chromosomes' and each chromosome consists of a set of elements called 'genes' represent decision variables. Evolution process starts by generating a random population of solutions, i.e., parent chromosomes, and evaluating them based on a fitness function, which is usually defined with respect to objective function. Subsequently, best parent chromosomes exchange their information through the process of "crossover" or "mutation" and create offspring chromosomes. Each offspring chromosome is evaluated based on its fitness value and the fittest chromosomes are selected to repeat the process of evolution until maximizing the fitness function (Goldberg et al 1991). Producing offspring chromosomes based on crossover among parent chromosomes is a natural process of biological evolution, i.e., optimization. Crossover exchanges genes among parent chromosomes in an attempt to produce a fitter offspring (Figure 2.3a). Crossover rate is a number between 0 and 1 that represents the percentage of the selected genes that usually has a range from 0.6 to 1.0. The process of selection is typically based on a uniform crossover routine, which randomly selects a set of elements from an organism (i.e., the chromosome), or a single-point crossover that splits the organism in two parts at a random point (Figure 2.3b). Mutation process resembles sudden changes to a chromosome. A mutation rate from 0 to 1 is usually selected as the percentage of genes to mutate. This can be done by assigning a random number to each variable (say 0.07), and mutating all variables with the number less than or equal to mutation rate (Figure 2.4). Accordingly, by a mutation rate of 1 a completely different solution is generated with changing all variables. Mutation is a rare but useful or even vital process that introduces new genetic materials to the evolutionary process. In optimization, mutation is perhaps the best tool to avoid local optima by randomly searching the solution-space (Figure2.4). The best way of using mutation is by defining a small value for mutation rate in the beginning of optimization and increasing it when there is no improvement for optimal solution (Elbeltagi et al. 2005).

The main parameters that affect the performance of GAs are population size, number of generations, crossover rate, and mutation rate. Increasing the number of population or the number of generations can improve the solution quality but substantially increase processing time. One of the main difficulties in the prioritization of assets involve in an infrastructure network is the large number of components in which optimization performance decreases dramatically by increasing the number of components (Ekhakeem and Hegazy 2010). Accordingly, one of the main objectives of this research is to improve the performance of optimization under such conditions. Following is a pseudo code presented for a GA procedure (Elbeltagi et al. 2005):

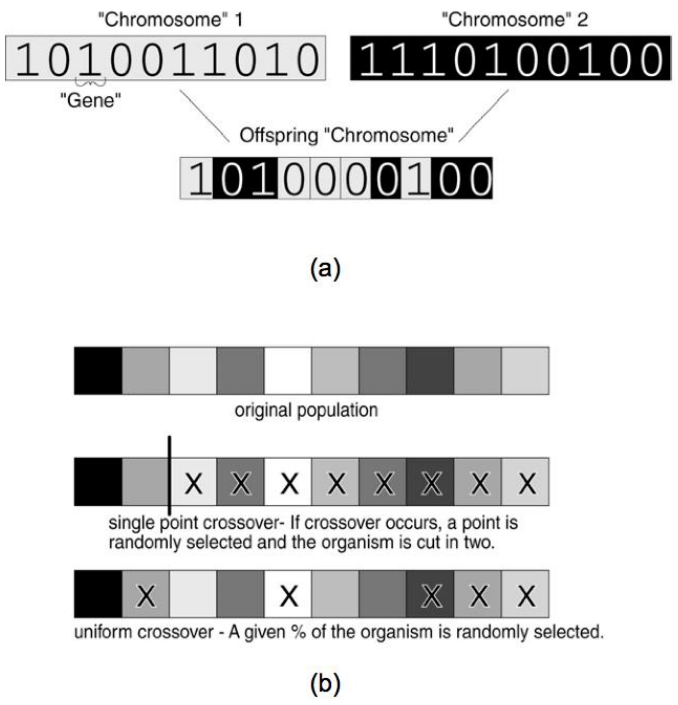

"Chromosome" 1
1010011010
"Chromosome" 2
1110100100
"Gene"
Offspring "Chromosome"
1010000100
(a)
original population
single point crossover- If crossover occurs, a point is randomly selected and the organism is cut in two.
uniform crossover - A given % of the organism is randomly selected.
(b)

**Figure 2.3: a) Crossover procedure; b) Single point and uniform crossover (Elbeltagi et al. 2005)**

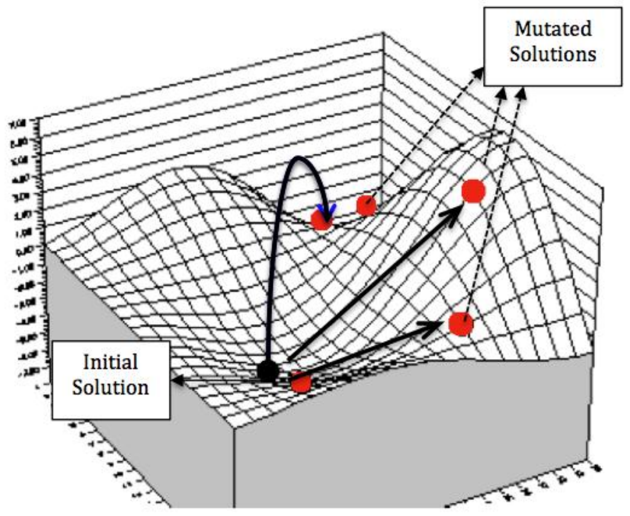


**Figure 2.4: Avoiding local optima by using mutation (Evolver Guide 2010)**

A pseudo code for applying a GA can be written as follow (Elbeltagi et al. 2005):

```
Begin;
    Generate random population of P solutions (chromosomes);
    For each individual i∈P: calculate fitness (i);
         For i=1 to number of generations;
               Randomly select an operation (crossover or mutation);
               If crossover;
                   Select two parents at random $i_a$ and $i_b$;
                   Generate on offspring $i_c$= crossover ($i_a$ and $i_b$);
               Else If mutation;
                   Select one chromosome i at random;
                   Generate an offspring $i_c$= mutate (i);
               End if;
               Calculate the fitness of the offspring $i_c$;
               If $i_c$is better than the worst chromosome then
               replace the worst chromosome by $i_c$;
          Next i;
     Check if termination= true;
End;
```

## 2.5. Summary and Conclusions

In this chapter, a review of asset management systems, asset prioritization and repair fund allocation methods, and asset renewal optimization methods has been presented. The main challenge in the process of asset prioritization is actually how to handle the problem with scarce available budgets such that the satisfactory level of performance is achieved during the planning horizon.  This leads to an asset renewal optimization problem that is usually large-scale and complex in real-life practices. To handle a large-scale combinatorial problem different optimization techniques can be used. Genetic Algorithm (GA) is one of the common tools for dealing with such problems. Also performance of advanced mathematical tools introduced recently requires more investigation in large-scale environments.

The present research focuses on the development of mechanisms for improving the performance of GA-based optimization methods in handling very large-scale asset renewal problems and also investigates the performance of recent advanced mathematical optimization tools on such problems and comparing it with GA-based approach.

# Chapter 3

# Asset Renewal Model for Large-Scale Problems

## 3.1. Introduction

This chapter illustrates the life cycle cost analysis (LCCA) model introduced by Elhakeem and Hegazy (2010), which is the basis for the new developments discussed in chapter 4 to extend the optimization capabilities to larger size problems. Once the basic model is described, the proposed methodology to expand it to large scale problems is highlighted. This methodology is then followed in chapter 4 and 5.

## 3.2. Asset Management Framework

This section presents a brief description of the asset management framework used by Elhakeem and Hegazy (2010). To model an asset renewal problem, different functions of AMS, such as condition assessment, deterioration modeling, repair modeling and LCCA should integrate properly (Figure 3.1) (Hudson et al 1997; Elhakeem and Hegazy 2010). Current conditions of individual asset components are measured by inspecting assets using inspection techniques, such as visual inspection, photographic inspection, non-destructive evaluation, or smart sensors. Evaluating the condition of asset components can be based on a distress survey or direct condition rating (Uzasrki 2002). Also, different scales and linguistic representations can be used for condition evaluation. Some of these condition scales and the corresponding linguistic representation are shown in Table 3.1. A visual direct inspection method using condition scale of 0 to 100 is used for condition assessment of school building networks related to Toronto District school Board (Hegazy 2005). After evaluating the current condition of the asset components, next functions are integrated within a bilevel life cycle optimization. A bilevel or multilevel optimization involves a hierarchy where outer optimization problem is subjected to outcome of several enclosed problems. In the case of bilevel life cycle optimization for TDSB, first level is the component-level that seeks the bests repair scenario for each component. Next level is the network-level that best repair scenarios obtained from component level are used to prioritize assets and allocate available budget to them through a large-scale optimization framework (Figure 3.2). In this chapter, the process of bilevel optimization and current research results using this framework are discussed and the methodology for expanding to larger scale problems is introduced.

**Table 3.1: Condition scales and corresponding linguistic representations**

| **Reference** | **Asset Type** | **Condition Scale** | **Linguistic Representation** |
|---|---|---|---|
| Lee and Aktan 1997 | Buildings | 1 - 4 | Deterioration: (1= no; 2 = slight; 3 = moderate; and 4 = severe). |
| Elhakeem and Hegazy 2005a | Buildings | 0 - 100 | Deterioration: (0-20) = no; (20-40) = slight; (40-60) = moderate; (60-80) = sever; and (80-100) = critical. |
| Greimann et al. 1997 | Locks and Dams | 0 - 100 | Maintenance need: (0-39) = only after further investigation; (40-69) = only if economically feasible; and (70-100) = no action is required. |
| Pontis 1995 | Bridges | 1 - 5 | Deterioration Process: (1 = protected; 2 = exposed; 3 = vulnerable; 4 = attacked; and 5 = damaged). |
| Lounis et al. 1998 | Any Asset | 1-7 | Condition category: (1 = Failed; 2 = V. Poor; 3 = Poor; 4 = Fair; 5 = Good; 6 = V. Good; and 7= Excellent). |

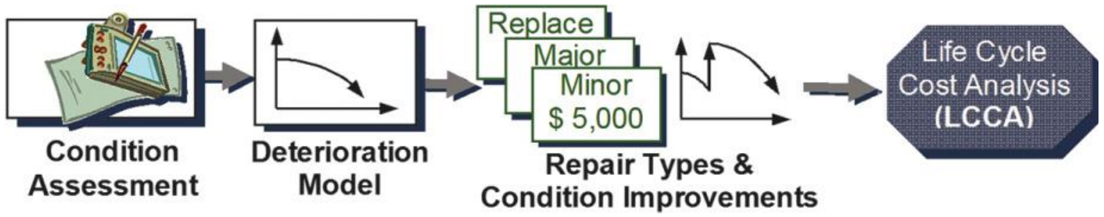

Condition Assessment
Deterioration Model
Replace
Major
Minor
$ 5,000
Repair Types & Condition Improvements
Life Cycle Cost Analysis (LCCA)

**Figure 3.1: Functions of an asset management system (Elhakeem and Hegazy 2010)**

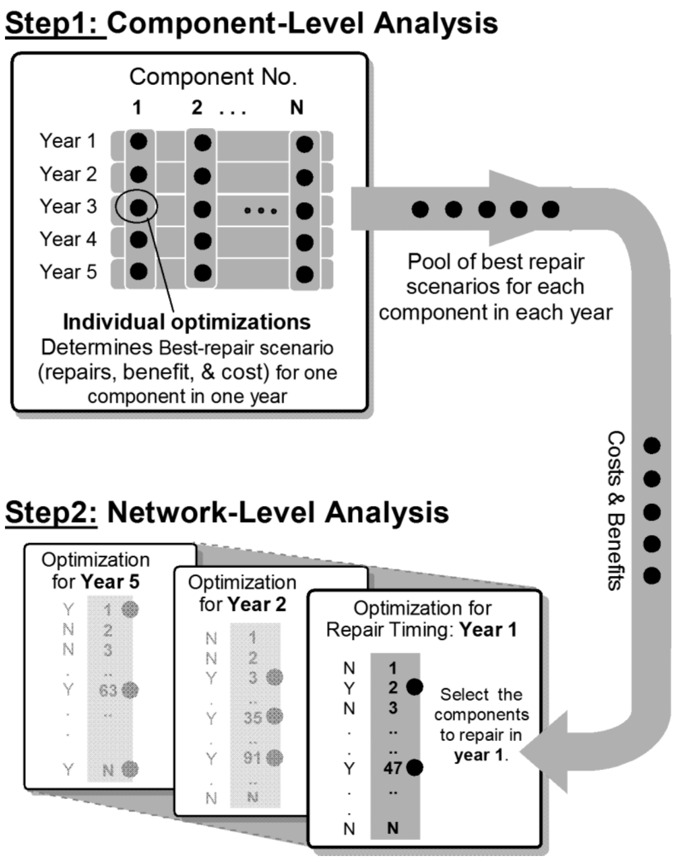

Step1: Component-Level Analysis
Component No.
1
2 . . .
N
Year 1
Year 2
Year 3
Year 4
Year 5
Individual optimizations
Determines Best-repair scenario (repairs, benefit, & cost) for one component in one year
Pool of best repair scenarios for each component in each year
Costs & Benefits
Step2: Network-Level Analysis
Optimization for Year 5
Optimization for Year 2
Optimization for Repair Timing: Year 1
Select the components to repair in year 1.

**Figure 3.2: Bilevel integrated life cycle optimization (Elhakeem and Hegazy 2010)**

## 3.3. Component Level: LCC Model for Building Components

In component level analysis, given a 5-year planning horizon, each individual component is investigated separately to determine the best repair scenario (highest benefit over cost (B/C) ratio) for that component. To do so, first, future deterioration of building components is determined; next, the optimum repair scenario is selected within an optimization process by maximizing the B/C ratio in each year during the planning horizon.

**Future Deteriorations**: In order to determine future deteriorations, deterministic models, such as straight-line extrapolation (linear deterioration modeling), or stochastic methods, such as Markovian deterioration modeling can be used. Linear deterioration modeling is a simple approach and not so accurate in determining future conditions (Hegazy 2005) (Figure 3.3). On the other hand, stochastic Markovian models predict the future behavior based on Markov Decision Process (MDP). Deterioration index for each instance with $d$ number of deficiencies is calculated based on the inspected severities ($S_i$) and weights of various deficiencies ($W_i$) as follow:

$$DI = \frac{\sum_{i=1}^{d} w_i . S_i}{100} \qquad (Elhakeem\ and\ Hegazy\ 2010) \qquad (3.1)$$

For the purpose of deterioration modeling of TDSB school buildings network, the optimized Markov chain process (Hegazy 2005) is used. Optimized Markov chain determines the future conditions of individual components by using the transition probability matrix (TPM) of components and generating a deterioration curve that best corresponds to previously inspected conditions (Figure 3.4).

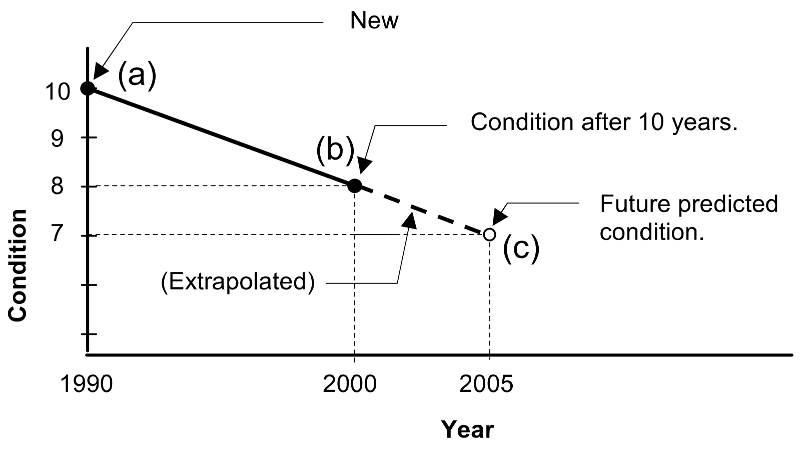
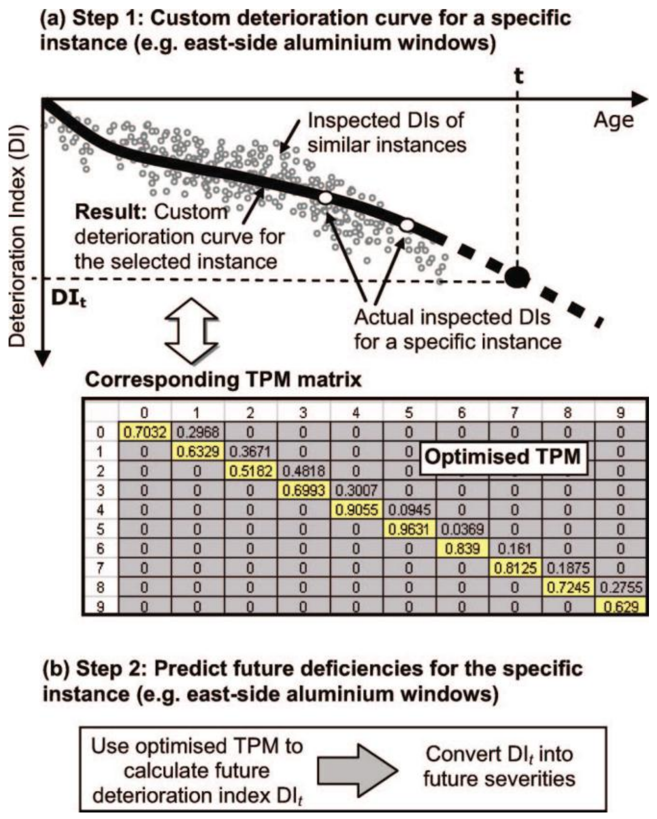


**Figure 3.3: Linear deterioration modeling**


(a) Step 1: Custom deterioration curve for a specific instance (e.g. east-side aluminium windows)
t
Age
Inspected DIs of similar instances
Deterioration Index (DI)
Result: Custom deterioration curve for the selected instance
DI$_t$
Actual inspected DIs for a specific instance
Corresponding TPM matrix
Optimised TPM
(b) Step 2: Predict future deficiencies for the specific instance (e.g. east-side aluminium windows)
Use optimised TPM to calculate future deterioration index DI$_t$
Convert DI$_t$ into future severities

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.7032 | 0.2968 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0.6329 | 0.3671 | 0 | 0 | 0 | [illegible] | [illegible] | [illegible] | 0 |
| 2 | 0 | 0 | 0.5182 | 0.4818 | 0 | 0 | [illegible] | [illegible] | [illegible] | 0 |
| 3 | 0 | 0 | 0 | 0.6993 | 0.3007 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0.9055 | 0.0945 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0.9631 | 0.0369 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0.839 | 0.161 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.8125 | 0.1875 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.7245 | 0.2755 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.629 |

**Figure 3.4: Optimized Markov chain deterioration modeling process (Elhakeem and Hegazy 2010)**

**Determining the Best Repair Scenarios**: After calculating deficiencies for each instance during the planning horizon, a repair scenario (RS) is defined as a combination of actions toward repairing these deficiencies. For instance, if a component has four deficiencies [D1, D2, D3, and D4], one possible repair scenario is represented in the binary form as (1, 1, 0, 1), which implies repairing deficiencies 1, 2, and 4 and keeping repair 3 with no repair. Repair cost for each repair scenario is calculated as a percentage of replacement cost, based on two assumptions:

1. Repair cost for a deficiency is proportional to the weight of that deficiency; and
2. Repairing one deficiency individually costs 25% more than its share of replacement.

Based on these assumptions the total cost (TC) of any repair scenario is calculated by summing the cost of all deficiencies as follow:

$$TC(\%) = \sum_{i=1}^{d} 1.25.W_i.RS_i \qquad (Elhakeem\ and\ Hegazy\ 2010) \qquad (3.2)$$

To convert total repair cost to dollars, following formula is used:

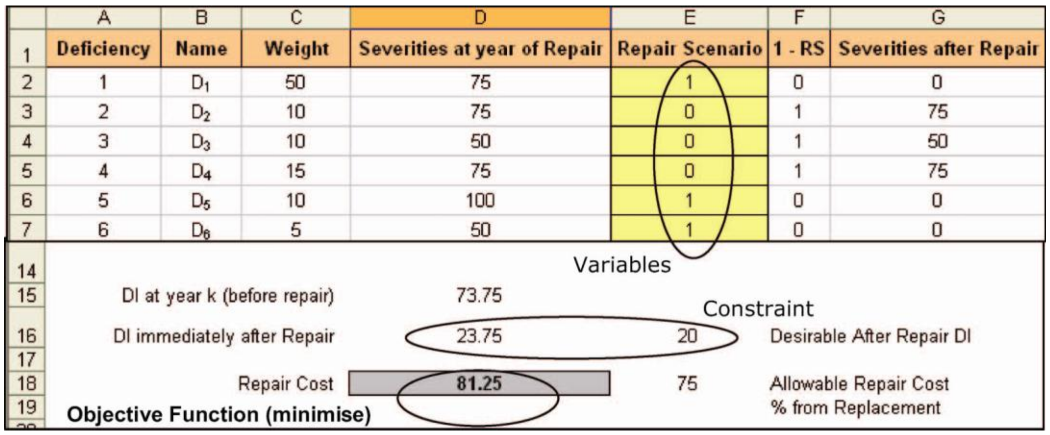
$$\$IRC_j = TC_j \times Z_j \times \$CRC \qquad (Elhakeem\ and\ Hegazy\ 2010) \qquad (3.3)$$

where $\$IRC$ is the instance repair cost in dollars, $TC$ is the total repair cost, $Z$ is the instance relative size, e.g., 20% of roof area, and $\$CRC$ is the total replacement cost for an instance.

In addition to estimating the repair cost, it is possible to determine the after repair deterioration index (ARDI) by assigning 0 serveries to repaired deficiencies. Following formula is used for calculating after repair conditions:

$$ARDI_k = \frac{\sum_{i=1}^{d} W_i \,.\, S_{ik} \,.\, (1 - RS_i)}{100} \quad (Elhakeem\ and\ Hegazy\ 2010) \qquad (3.4)$$

To determine the best repair scenario an optimization for each year to maximize the B/C ratio. The objective function is to minimize the repair cost while after repair deterioration is more that the acceptable level. In this way the cheapest repair that meets the minimum acceptable level of service constraint is selected as the best repair scenario. Figure 3.5 shows the optimization model for determining the best repair scenario.



| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | Deficiency | Name | Weight | Severities at year of Repair | Repair Scenario | 1 - RS | Severities after Repair |
| 2 | 1 | $D_1$ | 50 | 75 | 1 | 0 | 0 |
| 3 | 2 | $D_2$ | 10 | 75 | 0 | 1 | 75 |
| 4 | 3 | $D_3$ | 10 | 50 | 0 | 1 | 50 |
| 5 | 4 | $D_4$ | 15 | 75 | 0 | 1 | 75 |
| 6 | 5 | $D_5$ | 10 | 100 | 1 | 0 | 0 |
| 7 | 6 | $D_6$ | 5 | 50 | 1 | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| 14 | | | Variables | |
| 15 | DI at year k (before repair) | 73.75 | Constraint | |
| 16 | DI immediately after Repair | 23.75 | 20 | Desirable After Repair DI |
| 17 | | | | |
| 18 | Repair Cost | 81.25 | 75 | Allowable Repair Cost % from Replacement |
| 19 | Objective Function (minimise) | | | |

**Figure 3.5: Optimization model to determine the best repair scenario (Elhakeem and Hegazy 2010)**

## 3.4. Network Level: Prioritization and Repair Fund Allocation

The pool of best repair scenarios provided by small component-level optimizations is used as an input of a network-level large-scale optimization to determine the best year to repair each instance. The objective function in this level of optimization is to minimize the network deterioration index ($DI_N$) subjected to yearly budget limits and one time visit during the planning horizon. Network deterioration on a scale from 0 to 100 is calculated by the following formula:

$$DI_N = \frac{\left(\sum_j Average\left(DI_{jk}\right) \times RIF_j\right) + \left(\sum_j \left(EP_{jk} - EP_{j0}\right) \times RIF_j \times Y_{jk}\right)}{\sum_j RIF_j} \tag{3.5}$$

$$\forall\, j \in Network \quad , \quad \forall\, k \in Planning\ Horizon$$

where $EP_{jk}$ is the expected performance of asset j repaired in year k, $EP_{j0}$ is the expected performance of asset j with no repair, $RIF_j$ is the relative importance factor of asset j, and $Y_{jk}$ is the renewal decision for asset j in year k (1 for repair and 0 for no repair).

Instance expected performance is the average of deterioration indices for an instance when the instance is repaired in each of five years. Expected performance is a measure of repair impact over the planning horizon (Figure 3.6).

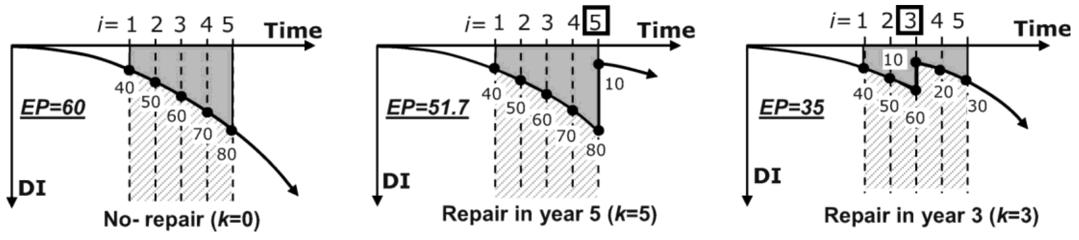


**Figure 3.6: Expected impact of repair over the planning horizon (Elhakeem and Hegazy 2010)**

The difference between the expected performance in any year over the planning horizon and $EP_0$ can be defined as the improvement effect of repairing asset j in year k ($IE_{jk}$):

$$IE_{jk} = EP_0 - EP_{jk} \tag{3.6}$$

Accordingly, the optimization model for network level is defined as follow:

*Objective function*: minimization

$$DI_N = \frac{\left(\sum_j Average\left(DI_{jk}\right) \times RIF_j\right) + \left(\sum_j IE_{jk} \times RIF_j \times Y_{jk}\right)}{\sum_j RIF_j} \tag{3.7}$$

$$\forall\, j \in Network \quad , \quad \forall\, k \in Planning\ Horizon$$

*Subjected to (s.t.)*:

$$\sum_j \$IRC_{jk} \le TB_k \ \ \forall\, k \in planning\ horizon \quad (yearly\ budget\ limits) \tag{3.8}$$

$$\sum_{k} Y_{jk} = 1 \ \forall j \ \in N \ \ (one - time\ visit\ during\ the\ life\ cycle) \tag{3.9}$$

$$Decision\ Variables = \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & Y_{14} & Y_{15} \\ Y_{21} & Y_{22} & Y_{23} & Y_{24} & Y_{25} \\ . & . & . & . & . \\ . & . & Y_{jk} & . & . \\ . & . & . & . & . \\ Y_{j1} & Y_{j2} & Y_{j3} & Y_{j4} & Y_{j5} \end{bmatrix}$$

To check the quality of solutions (optimality), results of simple ranking is used as a reference point. It is expected to have better quality solutions by using optimization in compare to simple ranking (Section 2.3), however, the quality of optimization solutions can be dramatically decrease by increasing the problem size (Figure 3.7).

**Simple Ranking:** Simple ranking or scoring is a subjective mechanism used by owners to prioritize their projects. Scoring projects based on their importance or benefit-to-cost (B/C) ratio and allocating resources to projects with higher scores is a simple approach for prioritizing and allocating funds. In the case of TDSB asset renewal problem, instances can be ranked based on the initial deterioration indices ($DI_0$) weighted by relative importance factor ($IRF_j$) of each individual component. Assuming a \$10 million yearly budget limit, by using simple ranking the overall network deterioration index improved from 54.33 to 44.89. Results of simple ranking is then used to compare and to evaluate the optimization solutions.

**GA-Based Optimization Performance on Large-scale Networks:** Experimentation on different size models by Elhakeem and Hegazy (2010) proved that optimization could lead to better solution quality in compare to simple ranking, but the performance of GA-based optimization approaches is degrading by enlarging the size of the asset renewal problem. Based on these experimentations the GA-based optimization approach is practical until reaching to 8000 instances. Figure 3.6 shows the trend of optimization performance on large-scale asset networks. Comparison with simple ranking indicates 21% improvement on model with 800 instances. However, the improvement is decreased by increasing the scale of the problem and reached to around 10% for 5000 instances.  After reaching 8000 instances, the optimization is failed practically and simple ranking becomes the dominant solution.

To improve the performance of optimization and to solve larger scale problems this research follows a methodology, which investigates both evolutionary-based optimization techniques and advanced mathematical optimization tools.

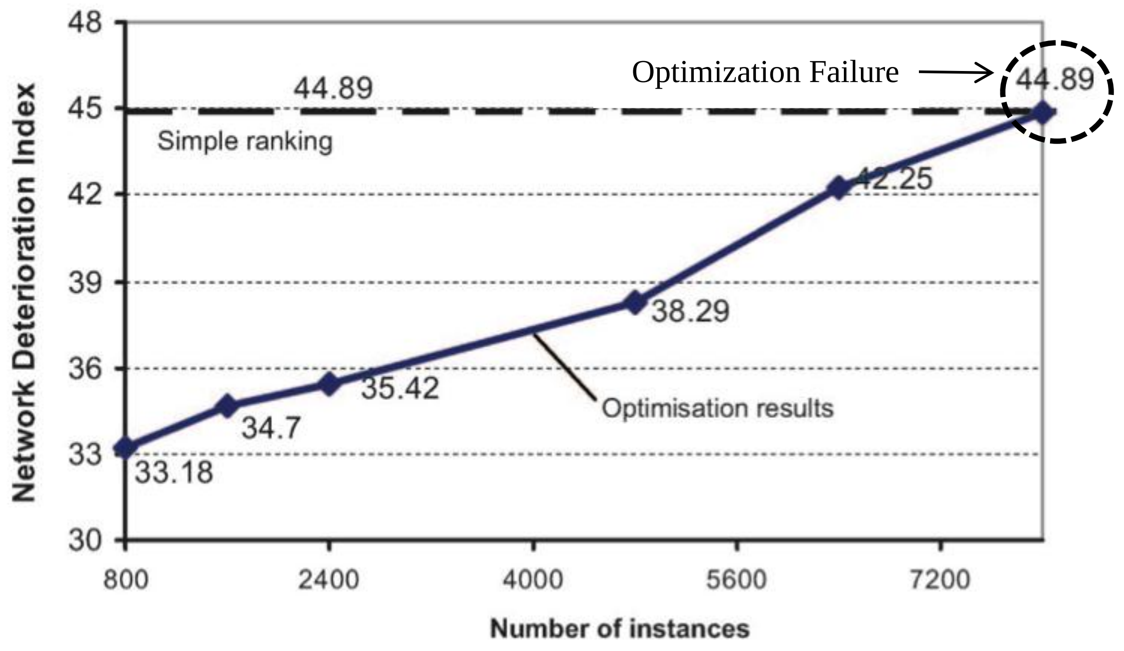


**Figure 3.7: Optimization performance on large-scale networks (Hegazy and Elhakeem, 2010)**

## 3.5. Methodology for Expanding to Very Large-Scale Problems

Developing a mechanism for handling larger-scale asset renewal problems is necessary when real-life problem are usually consist of more than 8000 components. To expand to problems with large number of components, different problem formulations and optimization tools have been investigated to select the best formulation and tool. Also performance of both genetic algorithms and mathematical optimization approaches have been examined and compared to see under which conditions each approach is dominant. Figure 3.8 shows the methodology used for expanding to large-scale problems.

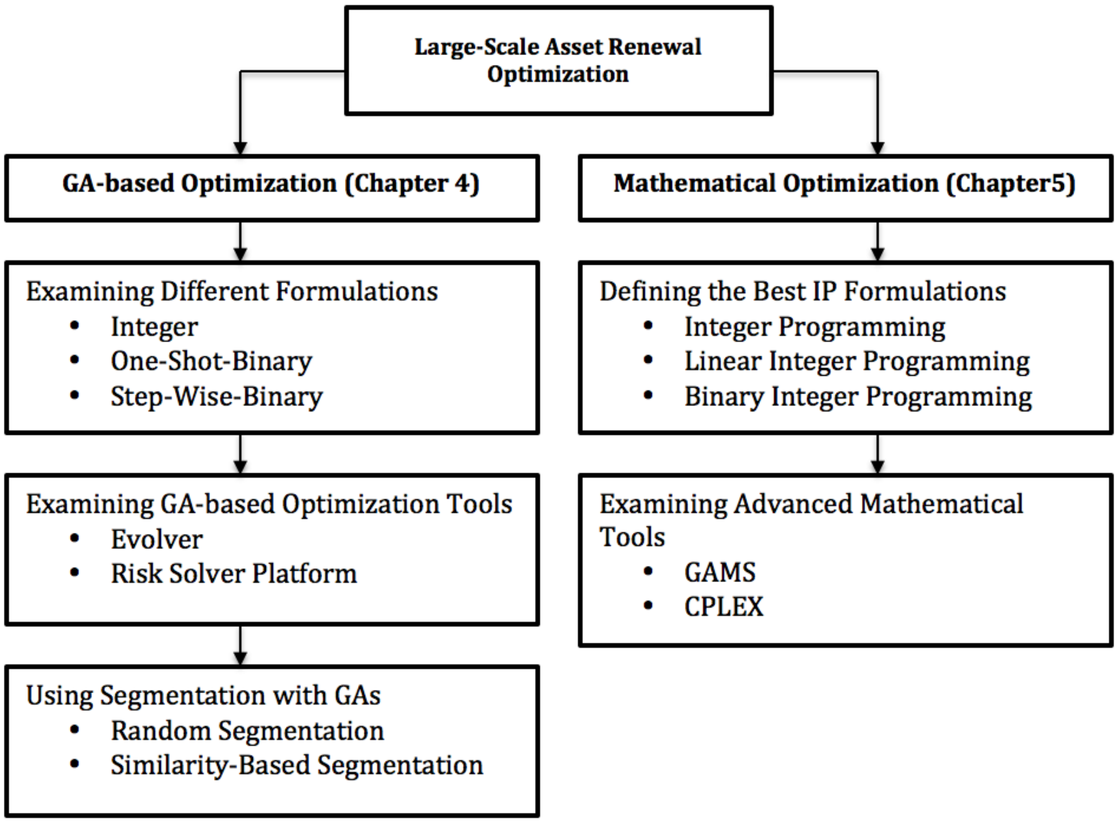

Large-Scale Asset Renewal Optimization
GA-based Optimization (Chapter 4)
Mathematical Optimization (Chapter5)
Examining Different Formulations
Integer
One-Shot-Binary
Step-Wise-Binary
Defining the Best IP Formulations
Integer Programming
Linear Integer Programming
Binary Integer Programming
Examining GA-based Optimization Tools
Evolver
Risk Solver Platform
Examining Advanced Mathematical Tools
GAMS
CPLEX
Using Segmentation with GAs
Random Segmentation
Similarity-Based Segmentation

**Figure 3.8: Methodology for expanding to very large-scale problems**

### 3.5.1. GA-Based Optimization

**Examining Different Formulations**: Different formulations have different solution space size and different number of variables that can affect the performance of optimization significantly (Csiszar 2007; Elhakeem and Hegay 2010). In this research, three different formulations, i.e., integer, one-shot-binary, and step-wise-binary, have been investigated to select the best formulation. Detailed description of each formulation and results of testing them on different size problems is discussed in chapter 4.

**Examining GA-Based Optimization Tools**: With optimization finding applications in almost every perceivable domain, particularly engineering (Maler and Arora 2004), different optimization tools become available to engineers. Evolutionary-based optimization tools are tool with optimization engines that use evolutionary algorithms, such as genetic algorithm, for solving optimization problems. Different evolutionary-based optimization tools have been introduced to engineers by developers, such as Microsoft, Frontline Systems Inc., etc. Table 3.2 shows some of the common evolutionary-based optimization tools and a brief description of their solvers. Two of the commercial optimization software have been tested in this study: 'Evolver' by Palisade Corporation, and 'Risk Solver Platform' by Frontline Systems Inc.

**Table 3.2: Some of the common commercial optimization tools**

| Tool | Developer | Description |
|---|---|---|
| **Excel Solver** | Microsoft Corporation | Easy-to-use mathematical optimization tool capable of handling simple LP and NLP problems |
| **Evolver** | Palisade Corporation | GA-based optimization tool effective in optimizing complex and large-scale models |
| **Risk Solver Platform** | Frontline Systems Inc. | Risk analysis, simulation, and optimization tool having GA-based and Mathematical optimization engines |
| **Microsoft Solver Foundation** | Microsoft Corporation | Mathematical optimization and modeling program supports LP, NLP, and MIP |
| **LINDO 12.0** | LINDO Systems Inc. | Mathematical optimization software supports LP, NLP, and MIP |
| **What's Best** | LINDO Systems Inc. | Excel add-in for LP, NLP, and MIP modeling and optimization |
| **AIMMS** | Paragon Decision Technology | Optimization software for mathematical programming |
| **KNITRO for Mathematica** | Ziena Optimization LLC. | A solver for linear and nonlinear mathematical optimization problems |
| **PPT - Strategic** | VEMAX | Modeling and optimization software supports network-level optimization by linear programing |

**Using Segmentation with GA**: Because the performance of evolutionary systems degrades rapidly with problem size, segmentation methods have been proposed to handle large-scale problems by decomposing them into smaller sub-problems with smaller solution space size. To generate sub-problems, different segmentation techniques have been proposed and tested in chapter 4. Random segmentation that creates segments with randomly sorted input data, or similarity-based segmentation methods, which generate segments with regard to the similarity measures between input data. Segmentation methods are discussed in detail in the next chapter and tested on different size problems to identify the most practical method for handling very large-scale problems.

### 3.5.2. Mathematical Optimization

**Defining the Best IP Formulation**: In mathematical programming, as it is discussed in chapter 5, the structure of the optimization model can influence the performance dramatically. The way that the optimization equations, decision variables, and their relationships are defined plays an important role in a large-scale problem. In chapter 5, it has been tried to formulate TDSB asset renewal problem with regard to specification of strong IP formulations (Wolsey 1989), to increase the efficiency and solution quality.

**Examining Advanced Mathematical Tools**: Mathematical optimization tools are designed to support linear programing (LP), nonlinear programing (NLP), and integer programing (IP) problems (Winston and Venkataramanan 2003). Although mathematical optimization tool are not considered to be effective in handling large-scale combinatorial problems (Elbeltagi 2005; Hegazy et al 2004), recent advances in computer science and mathematical optimization algorithms increased the capabilities of mathematical optimization tools (Winston and Venkataramanan 2003). Table 3.3 shows some of the advanced mathematical tools and a brief description of each. In chapter 5, performance of these advanced tools, i.e., GAMS and IBM ILOG CPLEX solver, are investigated on large-scale problems.

**Table 3.3: Advanced mathematical tools**

| Tool | Developer | Description |
|---|---|---|
| **IBM ILOG CPLEX** | IBM | Powerful mathematical programming and optimization software also capable of conducting constraint programing (CP) |
| **Gurobi Optimizer** | Gurobi Optimization | Mathematical optimization tool capable of solving LP. QP, MILP, and MIQP problems |
| **GAMS** | GAMS Development Corporation | High-level modeling system for mathematical programing and optimization |

## 3.6. Summary and Conclusions

In this chapter, first, the framework of asset management system used for supporting renewal action decisions related to TDSB is described. Next, the integrated bilevel life cycle cost optimization proposed by Hegazy and Elhakeem (2010), is introduced. The procedure for determining the best repair scenario at component-level and also the process of network-level optimization is described in detail. Based on previous optimization results, performance of evolutionary methods degrades by increasing the problem size. Accordingly, for improving the optimization performance, this study proposed a methodology that investigates both GA-based and mathematical optimization approaches.

# Chapter 4

# Large-Scale Optimization Using GAs + Segmentation

## 4.1. Introduction

As discussed in chapter 3, the performance of existing models for building asset renewal is degraded dramatically with the problem size gets larger. As such, performance degradation basically irrationalizes the application of optimization since equally good solutions can be obtained from simpler approaches such as ranking. To improve optimization performance for larger scale problems, different aspects related to formulization of evolutionary-based optimization models are investigated in this chapter. First, different problem formulations are introduced and investigated to select the best formulation in further experiments. Next, different evolutionary-based tools are tested on limited size models to select the most practical one. Subsequently, segmentation methods for handling very large-scale problems are introduced and discussed. Finally, based on the results of various experiments a systematic procedure using segmentation is proposed to handle very large-scale asset renewal problems and its performance on TDSB prototype is compared to previous results.

## 4.2. Problem Formulations

Network-level optimization can be modeled and implemented in different ways each has a unique formulation and corresponding solution quality. Solutions are basically affected by the chromosome size and the variety of genes produce by genetic algotithm (Section 2.4) (Elbeltagi et al. 2005).

### 4.2.1. Integer Formulation

Consider the case of asset renewal problem with a planning horizon of five years and six repair alternatives (no repair or repair in one of the five years). In this case, decision variables can be represented in the form of an integer number between 0 to 5 in which zero means no repair and the number 1 to 5 represents repair in year 1 to 5 (Figure 4.1a). If the asset renewal inventory consists of N instances, the solution-space size is equal to $6^N$ by integer representation of the problem. A sample of a chromosome produced by genetic algorithm and an integer formulated optimization model are shown in Figure 4.1b and 4.2 respectively.

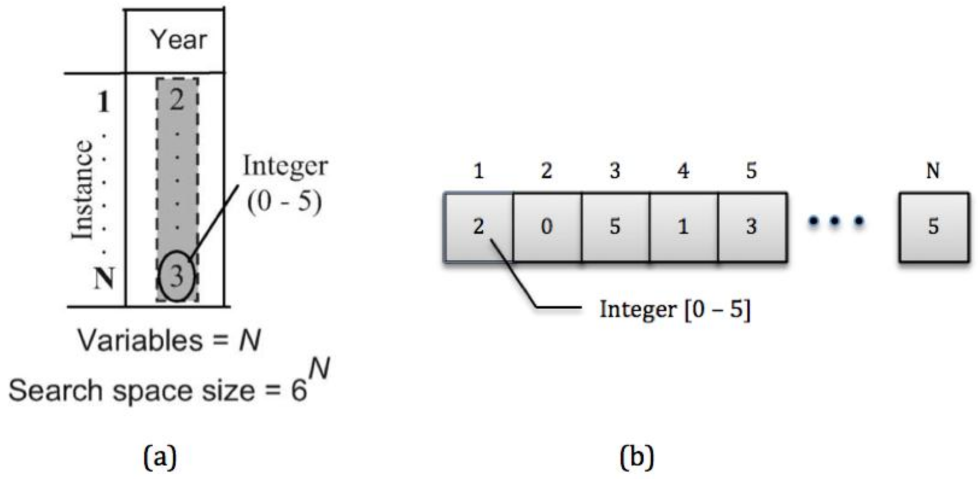
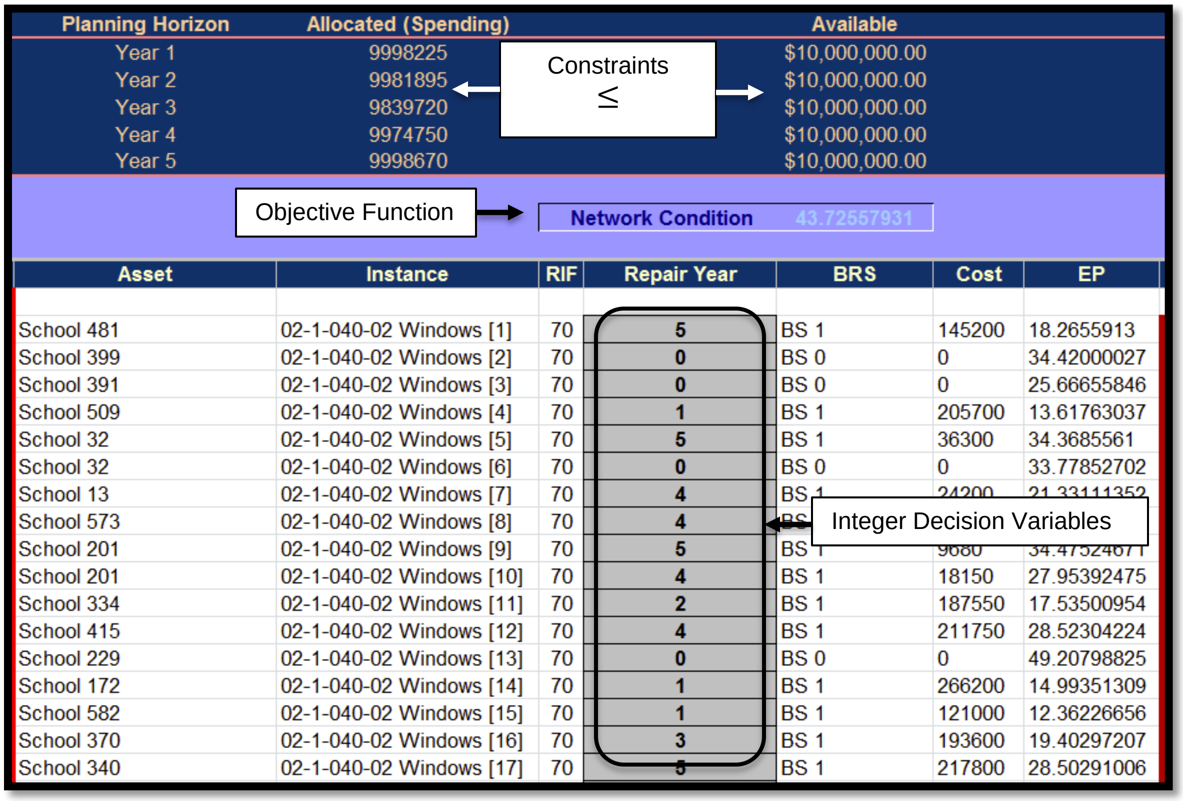


**Figure 4.1: a) Integer formulation; b) Sample chromosome**



| Planning Horizon | Allocated (Spending) | Available |
|---|---|---|
| Year 1 | 9998225 | $10,000,000.00 |
| Year 2 | 9981895 | $10,000,000.00 |
| Year 3 | 9839720 | $10,000,000.00 |
| Year 4 | 9974750 | $10,000,000.00 |
| Year 5 | 9998670 | $10,000,000.00 |

Network Condition 43.72557931

| Asset | Instance | RIF | Repair Year | BRS | Cost | EP |
|---|---|---|---|---|---|---|
| School 481 | 02-1-040-02 Windows [1] | 70 | 5 | BS 1 | 145200 | 18.2655913 |
| School 399 | 02-1-040-02 Windows [2] | 70 | 0 | BS 0 | 0 | 34.42000027 |
| School 391 | 02-1-040-02 Windows [3] | 70 | 0 | BS 0 | 0 | 25.66655846 |
| School 509 | 02-1-040-02 Windows [4] | 70 | 1 | BS 1 | 205700 | 13.61763037 |
| School 32 | 02-1-040-02 Windows [5] | 70 | 5 | BS 1 | 36300 | 34.3685561 |
| School 32 | 02-1-040-02 Windows [6] | 70 | 0 | BS 0 | 0 | 33.77852702 |
| School 13 | 02-1-040-02 Windows [7] | 70 | 4 | BS 1 | 24200 | 21.33111352 |
| School 573 | 02-1-040-02 Windows [8] | 70 | 4 | BS [illegible] | [illegible] | [illegible] |
| School 201 | 02-1-040-02 Windows [9] | 70 | 5 | BS 1 | 9680 | 34.47524671 |
| School 201 | 02-1-040-02 Windows [10] | 70 | 4 | BS 1 | 18150 | 27.95392475 |
| School 334 | 02-1-040-02 Windows [11] | 70 | 2 | BS 1 | 187550 | 17.53500954 |
| School 415 | 02-1-040-02 Windows [12] | 70 | 4 | BS 1 | 211750 | 28.52304224 |
| School 229 | 02-1-040-02 Windows [13] | 70 | 0 | BS 0 | 0 | 49.20798825 |
| School 172 | 02-1-040-02 Windows [14] | 70 | 1 | BS 1 | 266200 | 14.99351309 |
| School 582 | 02-1-040-02 Windows [15] | 70 | 1 | BS 1 | 121000 | 12.36226656 |
| School 370 | 02-1-040-02 Windows [16] | 70 | 3 | BS 1 | 193600 | 19.40297207 |
| School 340 | 02-1-040-02 Windows [17] | 70 | 5 | BS 1 | 217800 | 28.50291006 |

**Figure 4.2: Integer formulated optimization model spreadsheet**

### 4.2.2. One-Shot-Binary Formulation

Another possible way of modeling the network-level optimization is using a binary formulation. In this case, a value of 0 or 1 can be assigned to each decision variable. A set of six variables (no repair one repair in one of the five years) is defined for each instance that represents the renewal action along the planning horizon. Accordingly, total number of variables for an inventory with N instances is equal to $6N$ and the solution-space is equal to $2^{6N}$. In compare to integer formulation, chromosomes produced by one-shot-binary formulation are much longer in size, but the variety of genes is reduced from [0-5] to [0-1]. Figure 4.3 and 4.4 illustrate the one-shot-binary formulation and modeling.

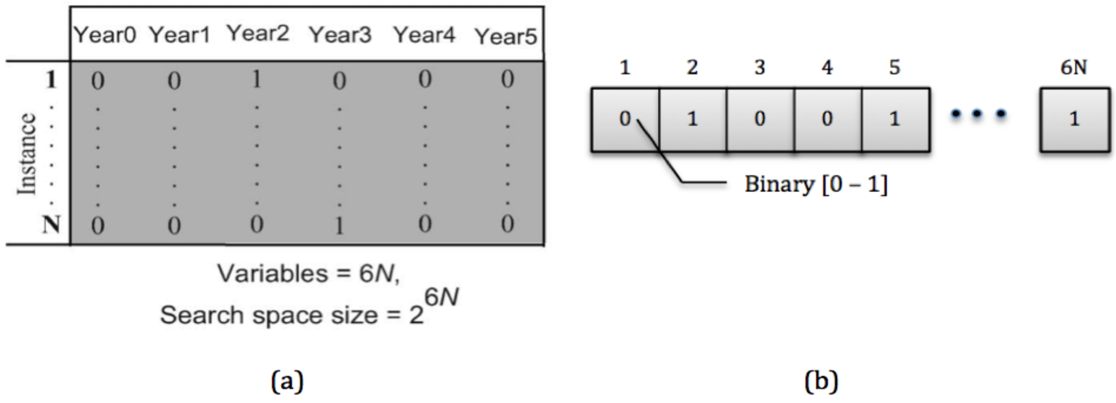
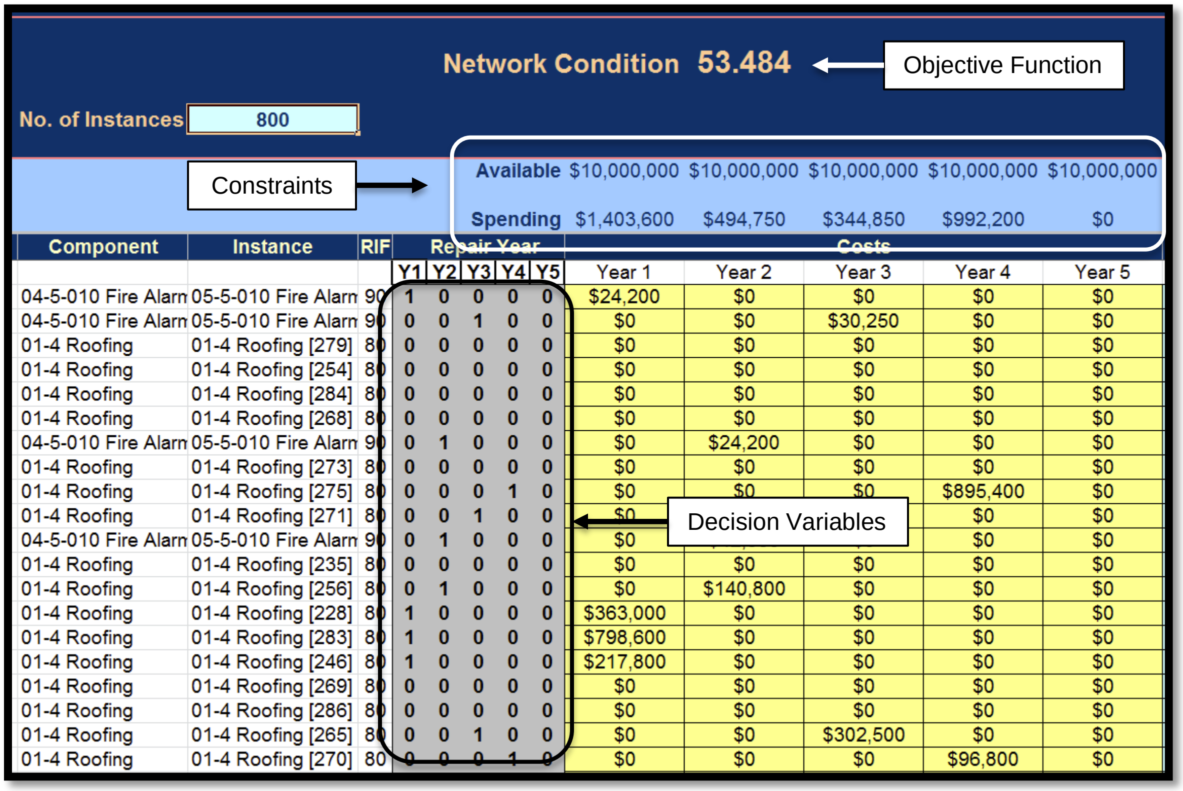


**Figure 4.3: a) One-shot-binary formulation; b) Sample chromosome**



Network Condition 53.484

No. of Instances 800

| | Year 1 | Year 2 | Year 3 | Year 4 | Year 5 |
|---|---|---|---|---|---|
| Available | $10,000,000 | $10,000,000 | $10,000,000 | $10,000,000 | $10,000,000 |
| Spending | $1,403,600 | $494,750 | $344,850 | $992,200 | $0 |

| Component | Instance | RIF | Repair Year | | | | | Costs | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Y1 | Y2 | Y3 | Y4 | Y5 | Year 1 | Year 2 | Year 3 | Year 4 | Year 5 |
| 04-5-010 Fire Alarn | 05-5-010 Fire Alarn | 90 | 1 | 0 | 0 | 0 | 0 | $24,200 | $0 | $0 | $0 | $0 |
| 04-5-010 Fire Alarn | 05-5-010 Fire Alarn | 90 | 0 | 0 | 1 | 0 | 0 | $0 | $0 | $30,250 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [279] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [254] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [284] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [268] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 04-5-010 Fire Alarn | 05-5-010 Fire Alarn | 90 | 0 | 1 | 0 | 0 | 0 | $0 | $24,200 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [273] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [275] | 80 | 0 | 0 | 0 | 1 | 0 | $0 | $0 | $0 | $895,400 | $0 |
| 01-4 Roofing | 01-4 Roofing [271] | 80 | 0 | 0 | 1 | 0 | 0 | $0 | | | $0 | $0 |
| 04-5-010 Fire Alarn | 05-5-010 Fire Alarn | 90 | 0 | 1 | 0 | 0 | 0 | $0 | | | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [235] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [256] | 80 | 0 | 1 | 0 | 0 | 0 | $0 | $140,800 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [228] | 80 | 1 | 0 | 0 | 0 | 0 | $363,000 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [283] | 80 | 1 | 0 | 0 | 0 | 0 | $798,600 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [246] | 80 | 1 | 0 | 0 | 0 | 0 | $217,800 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [269] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [286] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [265] | 80 | 0 | 0 | 1 | 0 | 0 | $0 | $0 | $302,500 | $0 | $0 |
| 01-4 Roofing | 01-4 Roofing [270] | 80 | 0 | 0 | 0 | 1 | 0 | $0 | $0 | $0 | $96,800 | $0 |

**Figure 4.4: One-shot-binary formulated optimization model spreadsheet**

### 4.2.3. Step-Wise-Binary Formulation

To reduce the solution-space size of a one-shot-binary formulation, a mechanism is defined to conduct a year-by-year optimization rather than optimizing all years at the same time (one-sot). In this approach, each year is optimized separately with respect to objective function. After optimizing in the first year, repaired instances are excluded from the next year optimization and the process is iterated until the end of the planning horizon. Using step-wise-binary formulation reduces the solution-space size to $2^N$, and produces smaller chromosomes in compare to integer or one-shot-binary formulation. Figure 4.5 and 4.6 illustrate the step-wise-binary formulation, sample GA chromosomes, and its implementation. All three formulations are investigated more in detail in next section and compared with each other.

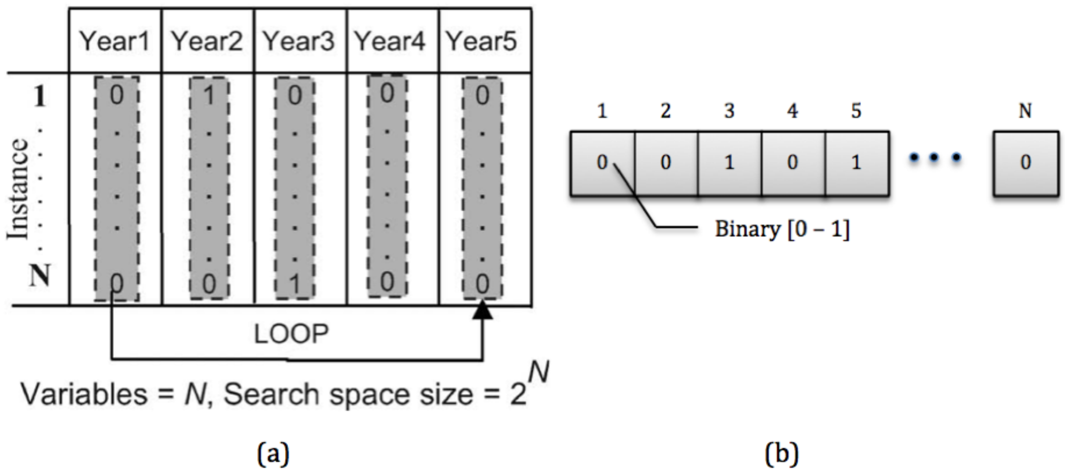
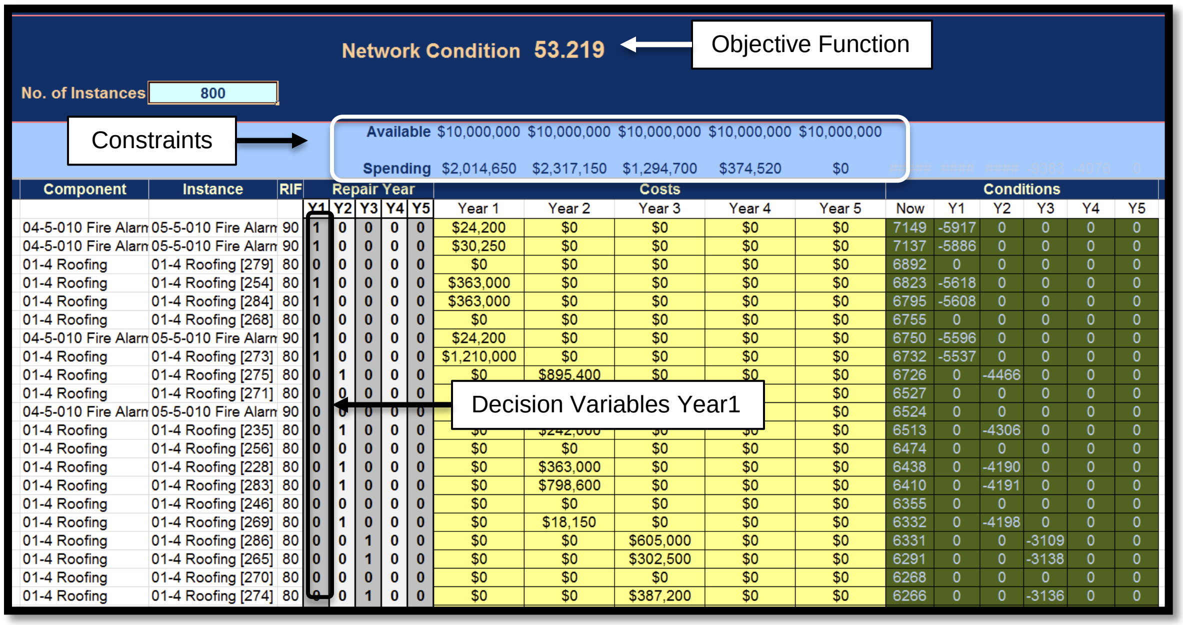


**Figure 4.5: a) Step-wise-formulation; b) Sample chromosome**



Network Condition 53.219

No. of Instances 800

| | Year 1 | Year 2 | Year 3 | Year 4 | Year 5 |
|---|---|---|---|---|---|
| Available | $10,000,000 | $10,000,000 | $10,000,000 | $10,000,000 | $10,000,000 |
| Spending | $2,014,650 | $2,317,150 | $1,294,700 | $374,520 | $0 |

| Component | Instance | RIF | Repair Year | | | | | Costs | | | | | Conditions | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Y1 | Y2 | Y3 | Y4 | Y5 | Year 1 | Year 2 | Year 3 | Year 4 | Year 5 | Now | Y1 | Y2 | Y3 | Y4 | Y5 |
| 04-5-010 Fire Alarm | 05-5-010 Fire Alarm | 90 | 1 | 0 | 0 | 0 | 0 | $24,200 | $0 | $0 | $0 | $0 | 7149 | -5917 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm | 05-5-010 Fire Alarm | 90 | 1 | 0 | 0 | 0 | 0 | $30,250 | $0 | $0 | $0 | $0 | 7137 | -5886 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [279] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6892 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [254] | 80 | 1 | 0 | 0 | 0 | 0 | $363,000 | $0 | $0 | $0 | $0 | 6823 | -5618 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [284] | 80 | 1 | 0 | 0 | 0 | 0 | $363,000 | $0 | $0 | $0 | $0 | 6795 | -5608 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [268] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6755 | 0 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm | 05-5-010 Fire Alarm | 90 | 1 | 0 | 0 | 0 | 0 | $24,200 | $0 | $0 | $0 | $0 | 6750 | -5596 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [273] | 80 | 1 | 0 | 0 | 0 | 0 | $1,210,000 | $0 | $0 | $0 | $0 | 6732 | -5537 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [275] | 80 | 0 | 1 | 0 | 0 | 0 | $0 | $895,400 | $0 | $0 | $0 | 6726 | 0 | -4466 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [271] | 80 | 0 | 0 | 0 | 0 | 0 | [illegible] | [illegible] | [illegible] | [illegible] | $0 | 6527 | 0 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm | 05-5-010 Fire Alarm | 90 | 0 | 0 | 0 | 0 | 0 | [illegible] | [illegible] | [illegible] | [illegible] | $0 | 6524 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [235] | 80 | 0 | 1 | 0 | 0 | 0 | $0 | $242,000 | $0 | $0 | $0 | 6513 | 0 | -4306 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [256] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6474 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [228] | 80 | 0 | 1 | 0 | 0 | 0 | $0 | $363,000 | $0 | $0 | $0 | 6438 | 0 | -4190 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [283] | 80 | 0 | 1 | 0 | 0 | 0 | $0 | $798,600 | $0 | $0 | $0 | 6410 | 0 | -4191 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [246] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6355 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [269] | 80 | 0 | 1 | 0 | 0 | 0 | $0 | $18,150 | $0 | $0 | $0 | 6332 | 0 | -4198 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [286] | 80 | 0 | 0 | 1 | 0 | 0 | $0 | $0 | $605,000 | $0 | $0 | 6331 | 0 | 0 | -3109 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [265] | 80 | 0 | 0 | 1 | 0 | 0 | $0 | $0 | $302,500 | $0 | $0 | 6291 | 0 | 0 | -3138 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [270] | 80 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6268 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [274] | 80 | 0 | 0 | 1 | 0 | 0 | $0 | $0 | $387,200 | $0 | $0 | 6266 | 0 | 0 | -3136 | 0 | 0 |

**Figure 4.6: Step-wise-binary formulated optimization model spreadsheet**

## 4.3. Experimenting with Different Evolutionary Tools

In the previous chapter, different optimization tools have been introduced in both mathematical and evolutionary categories. In this section, performance of two of the evolutionary tools, ‘Evolver’ by Palisade Corporation, and ‘Risk Solver Platform’ by Frontline Systems Inc., is examined.

### 4.3.1. Evolver

Evolver is a GA-based optimization tool working as an add-in to Microsoft Excel spreadsheet programs. User should design, formulate, and build the optimization model in an Excel spreadsheet and then call up Evolver for solving the problem. By using its GA-based engine Evolver is capable of solving different types of optimization models and proved to be one of the effective tools in dealing with different types of problems (Hegazy 2005; Ehakeem and Hegazy 2010).

**Avoid Trapping in Local Optima:** Simple solvers such as Excel Solver usually use a ‘hill climbing’ approach, which starts from a point in the feasible region and searches its neighborhood to find an optimum point. This method of finding optimum solutions can lead to trapping in local optima for problems with large solution space size (Figure 4.7a). Rather than using a hill climbing approach, Evolver is designed to surf the entire solution space by generating a pool of possible solutions and by comparing different local solutions at the same time it refines solutions. This leads to avoiding local optima by jumping from a local solution to other optimum points over the solution space and finding global or near-global answers to combinatorial optimization problem (Figure 4.7b).

**Handling Hard Constraints:** Evolver implements hard constraints by using a backtracking mechanism. After generating parent chromosomes (section 2.4), all of the new offspring chromosomes are checked against the constraints and if an offspring violates any of the hard constraints defined by the user it will be backtracked toward its parents and will be changed until producing a valid offspring (Figure 4.8).

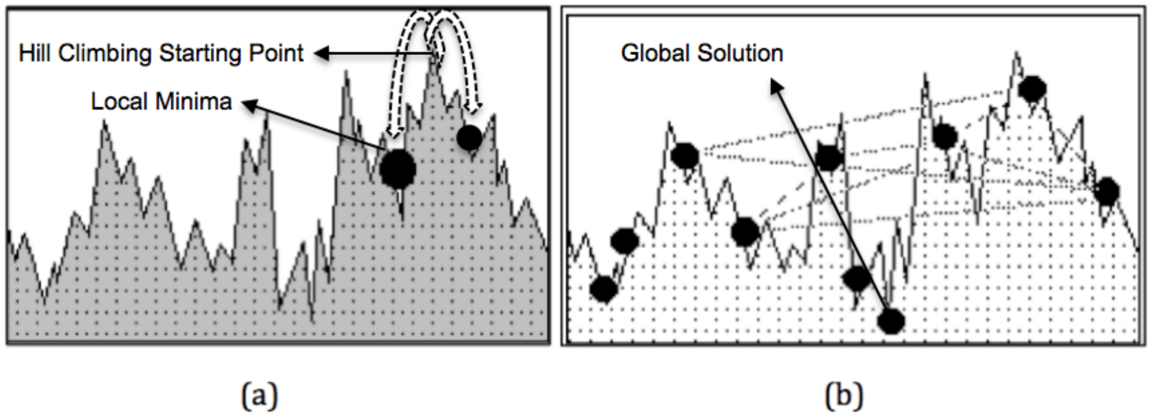


**Figure 4.7: a) Hill climbing approach; b) Avoiding local solution by generating multiple scenarios (Evolver Guide 2010)**

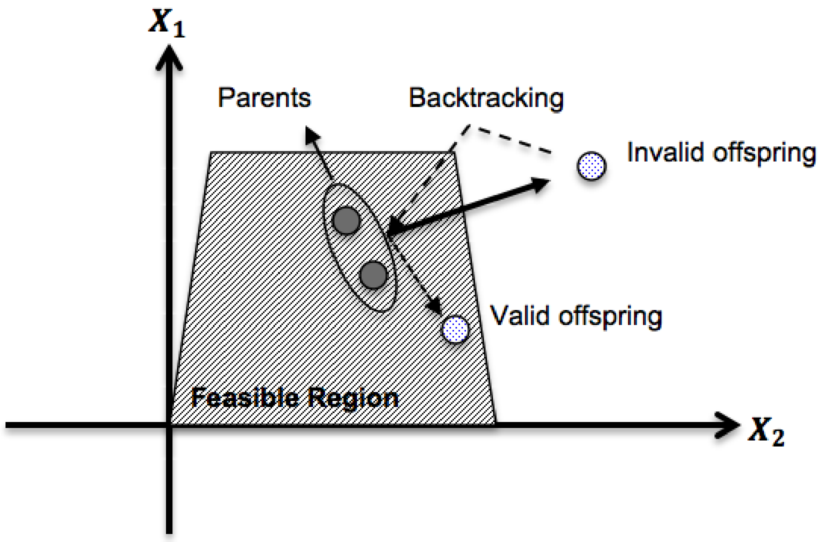
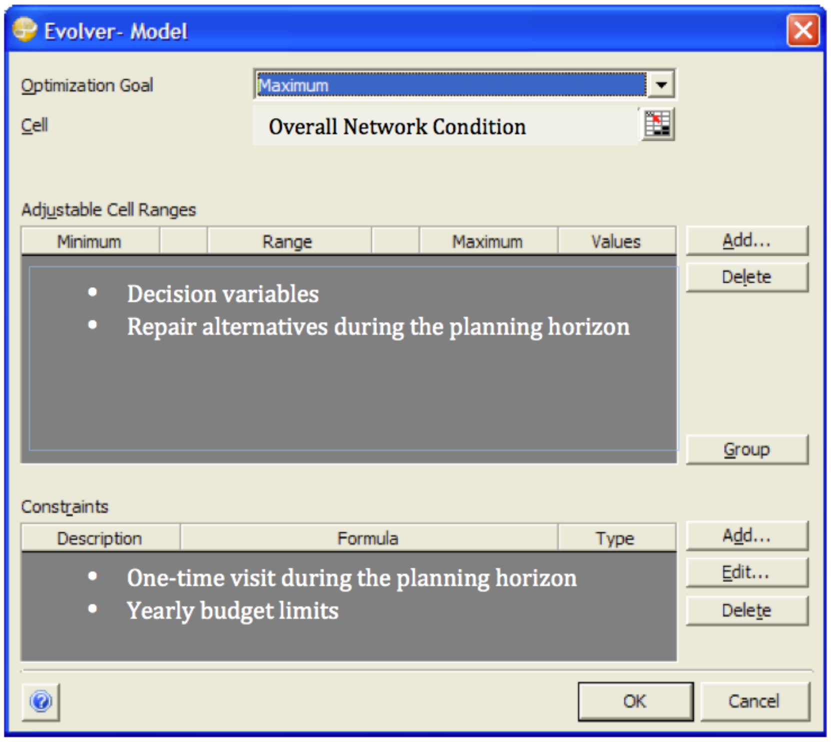


**Figure 4.8: Handling hard Constraints by Evolver (Evolver Guild 2010)**

**Experimentation:** Performance of Evolver is tested by using different size problems and different formulation. Samples with varying scales from 10 to 800 components are generated for this reason. Although these samples are so small in compare to the main goal of this research (50,000 components), they can be used to evaluate the performance of Evolver and different formulations. Figure 4.9 shows Evolver's modeling interface and the corresponding optimization parameters, i.e., objective function, decision variables, and constraints. Using three formulations (integer, one-shot-binary, and step-wise-binary) discussed in previous section, Table 4.1 shows the performance of Evolver in minimizing the overall network deterioration index ($DI_N$). As it can be seen from Table 4.1 and Figure 4.10, there is a trend of performance degradation by increasing the problem size. After examining all formulations, one with less performance degradation and better optimum solutions is selected as the best formulation.



Maximum

Overall Network Condition

- Decision variables
- Repair alternatives during the planning horizon

- One-time visit during the planning horizon
- Yearly budget limits

**Figure 4.9: Model definition in Evolver**

**Table 4.1: Optimization results using Evolver**

| | | Evolver | | |
|---|---|---|---|---|
| Sample | Size | One-shot-binary | Step-wise-binary | Integer |
| 1 | **10** | 29.931 | 29.537 | 28.559 |
| 2 | **10** | 29.456 | 24.288 | 24.288 |
| 3 | **20** | 31.865 | 21.368 | 23.136 |
| 4 | **40** | 38.442 | 25.502 | 26.81 |
| 5 | **50** | 26.687 | 20.935 | 20.55 |
| 6 | **50** | 27.733 | 23.375 | 23.21 |
| 7 | **100** | 34.048 | 27.309 | 27.47 |
| 8 | **100** | 49.3 | 46.901 | 44.41 |
| 9 | **200** | NE* | 32.617 | 30.3827 |
| 10 | **300** | | 35.535 | 32.93 |
| 11 | **400** | | 34.601 | 36.45 |
| 12 | **500** | | 37.833 | 40.3191 |
| 13 | **800** | | 39.079 | 41.6041 |

* NE: Not Efficient = Not able to find an optimum solution within a reasonable amount of time

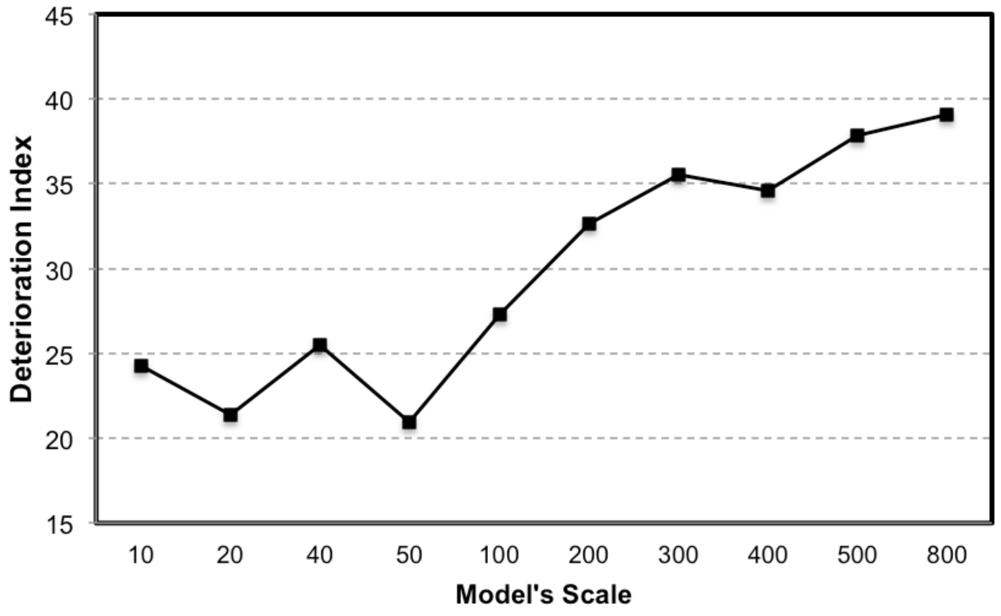


**Figure 4.10: Performance degradation of Evolver by increasing problem size**

**Solution Quality of Different Formulations**: By comparing the results obtained from all three formulations, step-wise-binary formulation found to have better solution quality (smaller values for $DI_N$) and less performance degradation. With regard to description of step-wise-binary formulation (section 4.2), better performance of this formulation is expected to be a result of smaller solution-space and chromosome size in compare to the other formulations.

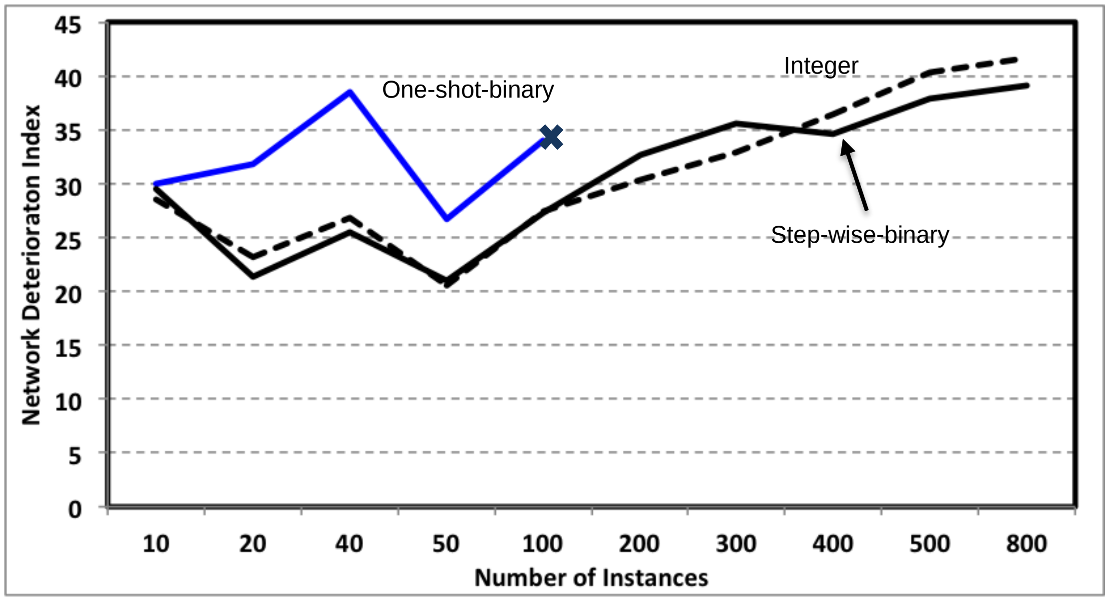


**Figure 4.11: Solution quality of different formulations using Evolver**

### 4.3.2. Risk Solver Platform

Risk Solver Platform is an optimization tool with both mathematical and evolutionary-based optimization engines developed by Frontline Systems. As compared to Excel Solver, Risk Solver Platform is a more advanced optimization capable of conducting simulation and risk analysis in addition to optimization. Table 4.2 presents the results obtained from Risk Solver Platform using the exact samples as used for testing Evolver and Figure 4.12 shows the performance of Risk Solver Platform with different problem formulations.  Based on Figure 4.12, using Risk Solver Platform also indicates that the step-wise-binary is performing better than integer and one-shot-binary formulation.

In this section two of the GA-based optimization tools are examined in terms of their performance in handling asset renewal models with number of components varies from 10 to 800. A comparison between optimum solutions and the capability of solvers in handling problems with larger number of variables shows that Evolver is the dominant optimization tool. Also, step-wise-binary formulation proved to be the best formulation among the three proposed formulations with better quality results and less performance degradation in both Evolver and Risk Solver Platform. Based on the experimentations presented in this section, Evolver is selected as the optimization tool and step-wise-binary as the formulation for handing larger size problems through the rest of this chapter.

**Table 4.2: Optimization results using Risk Solver Platform**

| | | **Risk Solver Platform** | | |
|---|---|---|---|---|
| Sample | Size | One-shot-binary | Step-wise-binary | Integer |
| 1 | **10** | 28.559 | 28.45 | 28.559 |
| 2 | **20** | 21.368 | 21.36 | 23.136 |
| 3 | **40** | 28.025 | 25.5 | 27.5 |
| 4 | **50** | TLTH* | 23.37 | 33.56 |
| 5 | **100** | | 27.3 | NE** |

* TLTH: Too Large To Handle (Not able to solve the problem due to the number of variables and/or constraints
** NE: Not Efficient = Not able to find an optimum solution within a reasonable amount of time

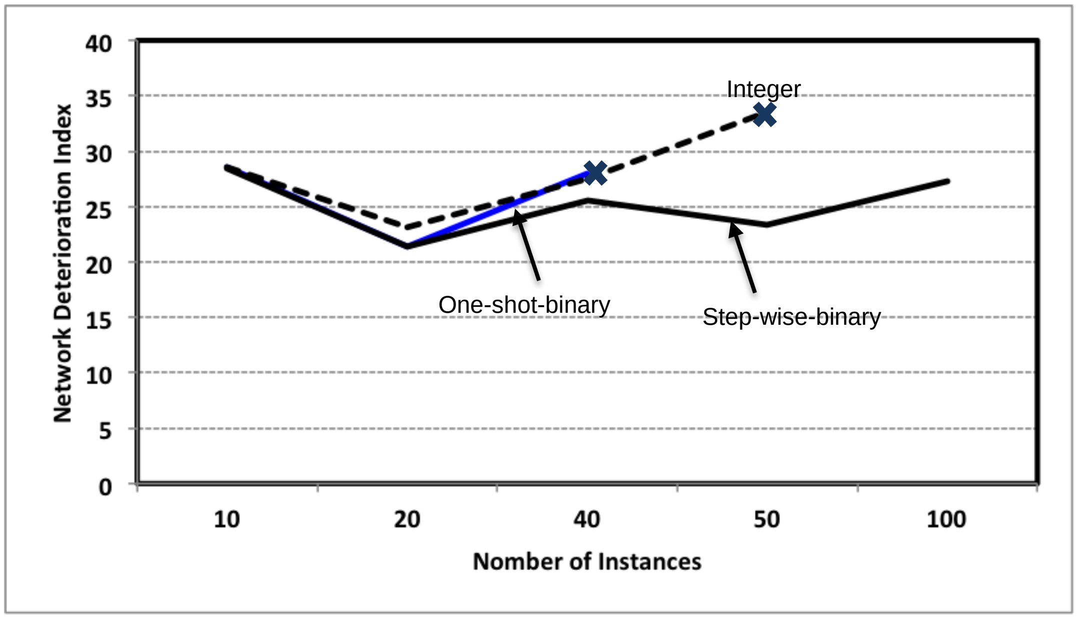


**Figure 4.12: Solution quality of different formulations using Risk Solver Platform**

## 4.4. Optimizing Large-Scale Problems Using Segmentation Methods

After identifying the best optimization tool and formulation, this study proposes segmentation methods as mechanism for handling asset renewal problems with sizes beyond the limits of previous studies (section 3.4). The main idea of segmentation is to decompose a large-scale problem into more optimizable sub-problems with smaller solution-space size. To create sub-models, different segmentation approaches can be used:

- Random Segmentation
- Similarity-based segmentation methods:
  - Data Compression
  - Clustered Segmentation

In the following sections, each of these methods are described and also tested on various size problems to measure the effectiveness of the method in handling large-scale problems.

### 4.4.1. Optimization with Random Segmentation

Random segmentation is a simple procedure for creating sub-models which can be used as a starting point to evaluate the effectiveness of using segmentation with GAs. In this approach, the optimization model is divided into segments with randomly selected internal data, which implies that the assets inside each segment do not have any specific similar characteristics such as same relative importance or initial condition. To distribute total yearly available budget among all segments, the budget for each segment is calculated by dividing the total yearly available budget by the number of created segments. For instance, if an optimization model is divided into four segments, 25% of total yearly available budget (TB) will be allocated to each segment. In the process of optimization each segment is optimized separately and results from each of the optimized segments are merged to resemble the final solution. This can be easily achieved by inserting the final solutions for each segment into the base model. Figure 4.13 shows the procedure of optimization with random segmentation. Based on a pairwise comparison between solution points obtained by random segmentation and GA optimization without segmentation, random segmentation found to be effective in terms of improving the solutions; however, it still suffers from performance degradation by increasing the problem size (Figure 4.14). As an attempt toward increasing the effectiveness of segmentation in handling large-scale problems, similarity-based segmentation methods are introduced and investigated in the next sections.

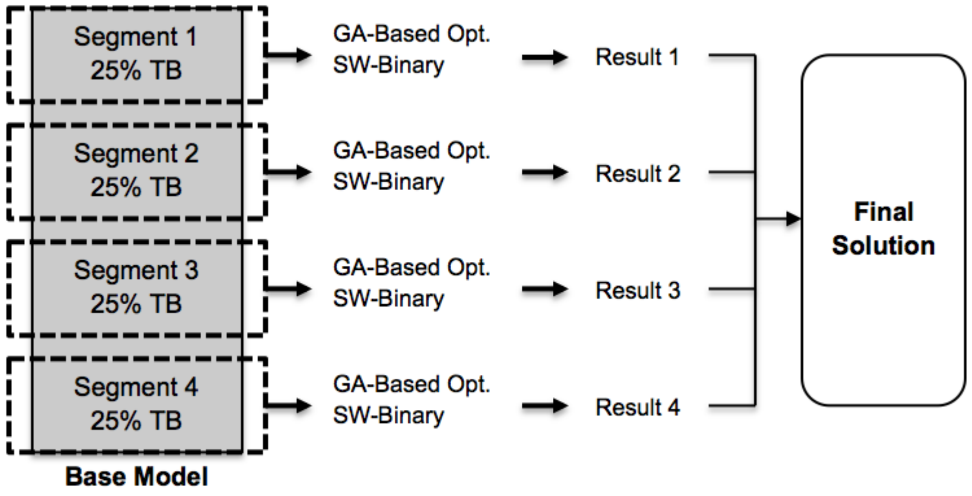


**Figure 4.13: Optimization with random segmentation**

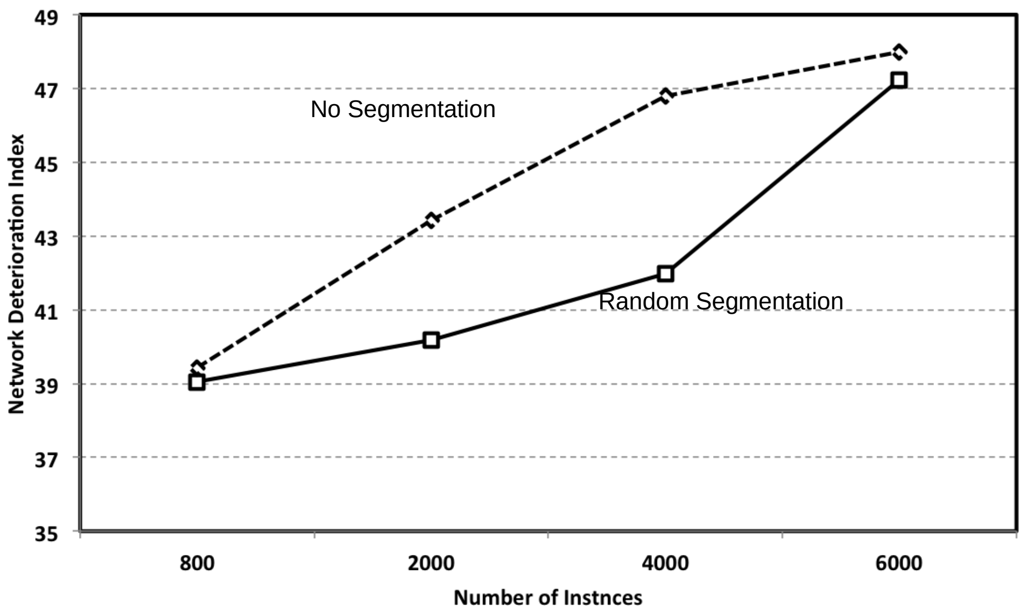


**Figure 4.14: Using random segmentation vs. optimization without segmentation**

### 4.4.2. Optimization with Similarity-Based Segmentation Methods

Basically, the concept of segmentation based on similarity is derived from clustering techniques. Data clustering methods are techniques used to classify datasets or observations into groups (clusters). Clustering is mainly based on pattern recognition and similarities between datasets with regard to different parameters (Steinbach et al. 2000). Considering similarity measures during the process of segmentation can result in creating segments with internal assets having close characteristics (very close or similar RIF, initial condition, deterioration behavior, etc.). Similarity-based segmentation methods can be used in different ways to handle large-scale problems as discussed in the following sections.

#### 4.4.2.1. Similarity/Distance Measures

Similarity (distance) between sets of data can be calculated using different approaches (Steinbach et al. 2000). Pierson Correlation and Euclidian Distance are among common measures that consider correlation or distance as a similarity measure respectively.

**Pierson Correlation**: The Pierson coloration for two sets of data, $x = \{x_1, x_2, \dots, x_n\}$ and $y = \{y_1, y_2, \dots, y_n\}$ is defined as:

$$\rho = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - m_x}{\sigma_x}\right) \left(\frac{y_i - m_y}{\sigma_y}\right) \qquad (T.T.Soong\ 2004) \qquad (4.1)$$

where $m_x$ is the mean of set x and $\sigma_x$ is the standard deviation of $x_i$ values. Pearson correlation coefficient has a value between 1 and -1. A correlation equal to 1 means positively perfectly correlated sets, or in other words, identical datasets. Conversely, a correlation of -1 indicates negatively perfectly correlated set or perfectly opposite sets (Figure 4.15) (T.T Soong 2004).

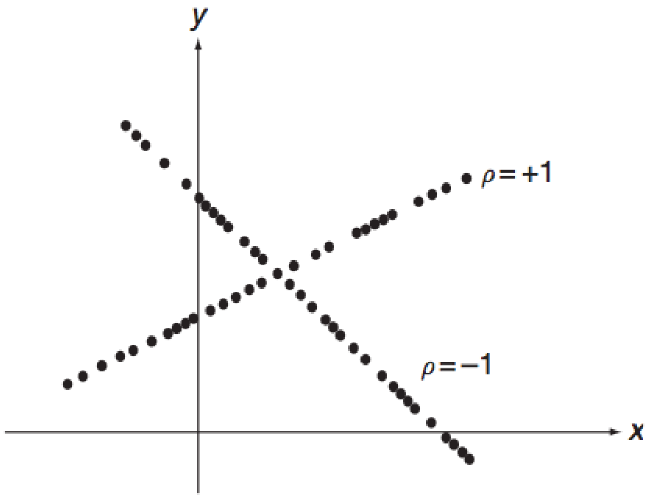


**Figure 4.15: Perfect Pierson correlation (T.T Soong 2004)**

**Euclidian Distance**: Consider two sets of data, $x = \{x_1, x_2, \dots, x_n\}$ and $y = \{y_1, y_2, \dots, y_n\}$, as two points in an n-dimensional Euclidian space. Using Euclidian distance, similarity between two datasets is defined as follow:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2} \qquad (Steinbach\ et\ al. 2000) \quad (4.2)$$

**Similarity Matrix:** Using Pierson Correlation or Euclidean Distance as a similarity measure for a dataset consists on N data element, similarity matrix is an N by N matrix shows a pair-wise similarity comparison between all data elements. Level of similarity between data elements are usually indicated by different colors (Steinbach et al. 2000).

#### 4.4.2.2 Large-Scale Optimization Using Data Compression

The process of data compression is starts with segmenting data related to a large asset inventory based on similarities between instances. Similarity can be defined using one of the aforementioned similarity measures (Eq. 4.1 or Eq. 4.2). Another way for determining similar asset is by grouping them based on the asset hierarchy (assets within the same building system, e.g., mechanical, electrical, architectural) and with regard to parameters that mostly affect the overall network condition, such as relative importance or initial conditions of assets. By applying this approach assets within the same system and with similar or very close importance and initial conditions are grouped in a segment. Each segment will be replaced by only one instance as the representative of all of the instances within that segment. The compressed model contains only the representative instances, is then optimized instead of the initial large-scale model. After optimization is done, solutions obtained for each representative instance is reassigned to all instances in the segment that it represents (Figure 4.16).

**Experimentation**: A model with 800 instances from TDSB asset inventory is used for the purpose of experimentation. The inventory for school buildings consists of several elements defined based on ATSM UNOFORMAT II as shown in Figure 4.17. Based on the previous description of data compression, segments should contain instances from same system, i.e., mechanical, electrical, or architectural. The initial model consists of 541 instances from architectural system, 210 instances from mechanical system, and 49 instances from electrical system. After grouping instances based on building systems, they are segmented with regard to similarity between importance and initial conditions of instances. Following this procedure, 29 segments are created for instances from architectural system, 22 segments for mechanical system, and 21 segments for electrical system.

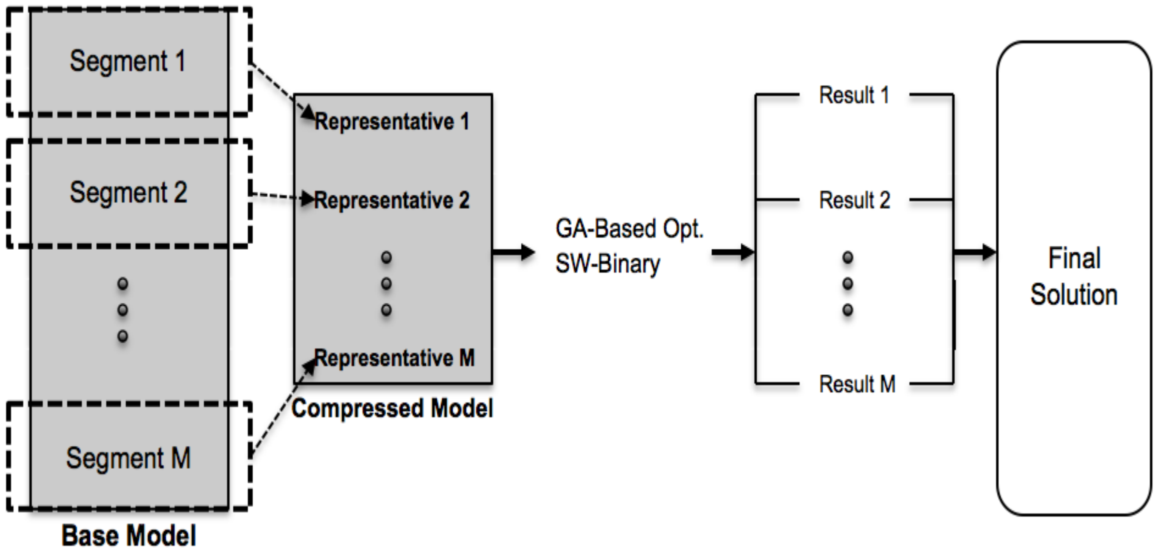
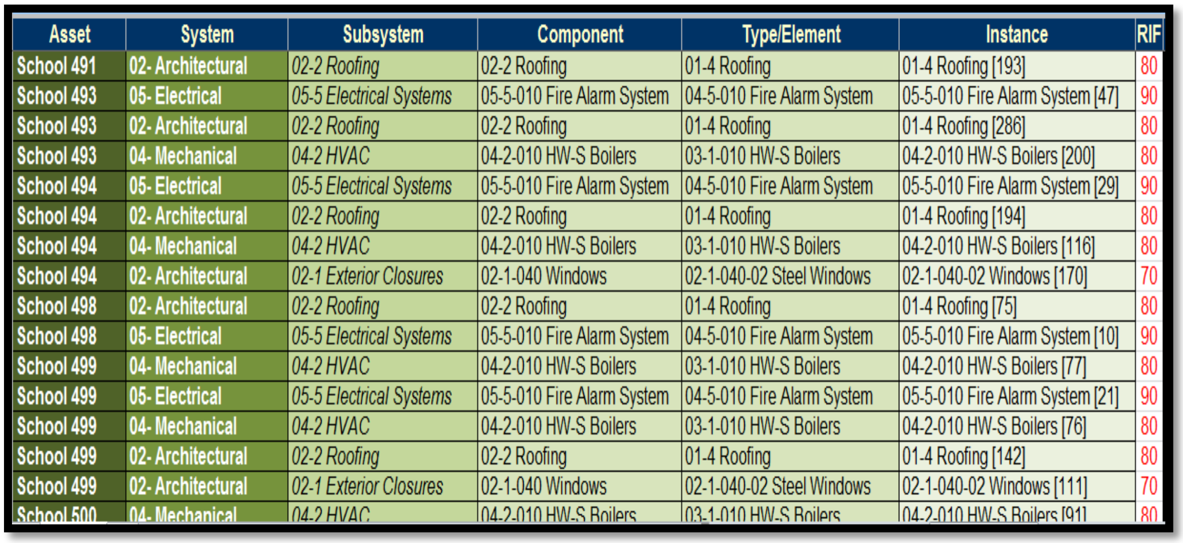


**Figure 4.16: Optimization with data compression**



| Asset | System | Subsystem | Component | Type/Element | Instance | RIF |
|---|---|---|---|---|---|---|
| School 491 | 02- Architectural | 02-2 Roofing | 02-2 Roofing | 01-4 Roofing | 01-4 Roofing [193] | 80 |
| School 493 | 05- Electrical | 05-5 Electrical Systems | 05-5-010 Fire Alarm System | 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [47] | 90 |
| School 493 | 02- Architectural | 02-2 Roofing | 02-2 Roofing | 01-4 Roofing | 01-4 Roofing [286] | 80 |
| School 493 | 04- Mechanical | 04-2 HVAC | 04-2-010 HW-S Boilers | 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [200] | 80 |
| School 494 | 05- Electrical | 05-5 Electrical Systems | 05-5-010 Fire Alarm System | 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [29] | 90 |
| School 494 | 02- Architectural | 02-2 Roofing | 02-2 Roofing | 01-4 Roofing | 01-4 Roofing [194] | 80 |
| School 494 | 04- Mechanical | 04-2 HVAC | 04-2-010 HW-S Boilers | 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [116] | 80 |
| School 494 | 02- Architectural | 02-1 Exterior Closures | 02-1-040 Windows | 02-1-040-02 Steel Windows | 02-1-040-02 Windows [170] | 70 |
| School 498 | 02- Architectural | 02-2 Roofing | 02-2 Roofing | 01-4 Roofing | 01-4 Roofing [75] | 80 |
| School 498 | 05- Electrical | 05-5 Electrical Systems | 05-5-010 Fire Alarm System | 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [10] | 90 |
| School 499 | 04- Mechanical | 04-2 HVAC | 04-2-010 HW-S Boilers | 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [77] | 80 |
| School 499 | 05- Electrical | 05-5 Electrical Systems | 05-5-010 Fire Alarm System | 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [21] | 90 |
| School 499 | 04- Mechanical | 04-2 HVAC | 04-2-010 HW-S Boilers | 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [76] | 80 |
| School 499 | 02- Architectural | 02-2 Roofing | 02-2 Roofing | 01-4 Roofing | 01-4 Roofing [142] | 80 |
| School 499 | 02- Architectural | 02-1 Exterior Closures | 02-1-040 Windows | 02-1-040-02 Steel Windows | 02-1-040-02 Windows [111] | 70 |
| School 500 | 04- Mechanical | 04-2 HVAC | 04-2-010 HW-S Boilers | 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [91] | 80 |

**Figure 4.17: TDSB school buildings network asset inventory**

Representative instance for each segment has the following characteristics:

- Expected performance in each year is the average of expected performances of all instances in the segment, in that year.
- Relative importance factor is the average of relative importance factor of instances in the segment.
- Repair cost for representative instance is equal to sum of repair costs of all instances in the segment.

After determining representative instances for all segments, the initial model with 800 instances is compressed to a model with only 72 representative instances. The compressed model has the same objective function as the base model and also the same budget limits. After optimizing the compressed model and reassigning representative solution to corresponding instances a network deterioration index of 41.043 is obtained. In compare to solutions obtained by using genetic algorithm or GA + random segmentation for the same problem, data compression is found to have lower quality solution. Poor optimality can be attributed to not considering all instances in the process of optimization and only optimizing representative, which can exhibit approximation. Based on model compression experiment, it can be concluded that it is better to utilize segmentation in a way that all of the asset components are considered in the process of optimization since there could be an optimum combinations of variables that is not likely to happen by removing some of the assets from the optimization model.

#### 4.4.2.3. Optimization with Clustered Segments

To achieve higher quality results and better performance, it has been tried to utilize the best aspects of previous methods and experiments in clustered segmentation method. The process of optimization with clustered segments has two phases, 1) segmentation, and 2) optimization. Figure 4.18 shows different steps involve in each phase.

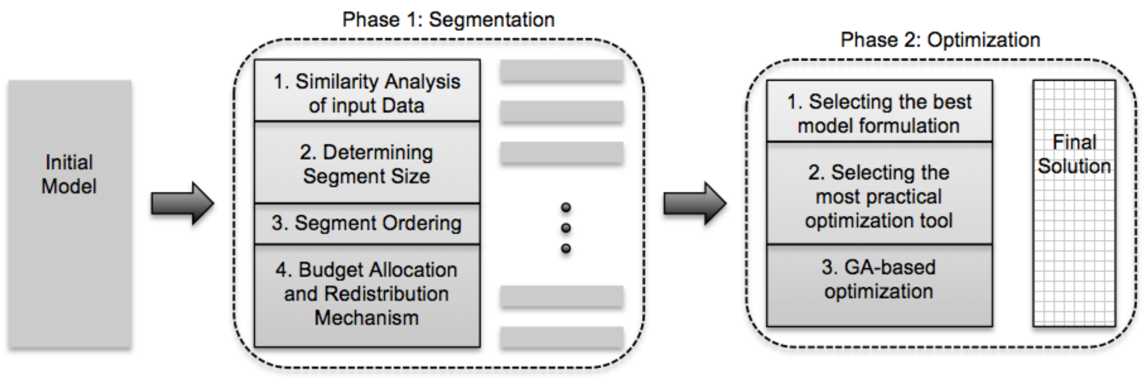

Initial Model
Phase 1: Segmentation
1. Similarity Analysis of input Data
2. Determining Segment Size
3. Segment Ordering
4. Budget Allocation and Redistribution Mechanism
Phase 2: Optimization
1. Selecting the best model formulation
2. Selecting the most practical optimization tool
3. GA-based optimization
Final Solution

**Figure 4.18: Optimization with clustered segments**

**Similarity Analysis of Input Data**: Similarity between instances is determined by using Euclidian distance measures with regard to deterioration behaviors (variations in DIs), and relative importance of individual assets as follow:

$$d(instance_i, instance_j) = \sqrt{\left(Average\left(DI_j\right)_k - Average(DI_i)_k\right)^2 + \left(RIF_j - RIF_i\right)^2}$$
$$\forall\, k \;\in Planning\ Horizon\ [1, 2, 3, 4, 5] \qquad (4.3)$$

where $Average\left(DI_j\right)_k$ is the average of deterioration indices during the planning horizon for instance j, and $RIF_j$ is the relative importance factor for instance j. Using Equation 4.3, similarity matrix of a 200-intance case is generated for with both randomly and similarity-based segmentation (Figures 4.19 and 4.20). As it can be seen from similarity matrices, variations of similarity are much less in similarity-based segments. Creating segments with similar assets makes it possible to characterize segments based on characteristics of their internal assets. For instance, if it is known that the relative importance factors of similar internal assets are low, the segment can be treated as a less important segment in compare to another segment having similar internal assets with high relative importance factors. Accordingly, if the internal assets of segments were not similar (randomly generated segments), it was not possible to assign specific characteristics, because assets with low relative importance factor and high relative importance factor can exist in the same segment (disturbed patterns: Figure 4.19). Creating similarity-based segments can be helpful in the process of segment ordering and budget allocation as described later in this section.

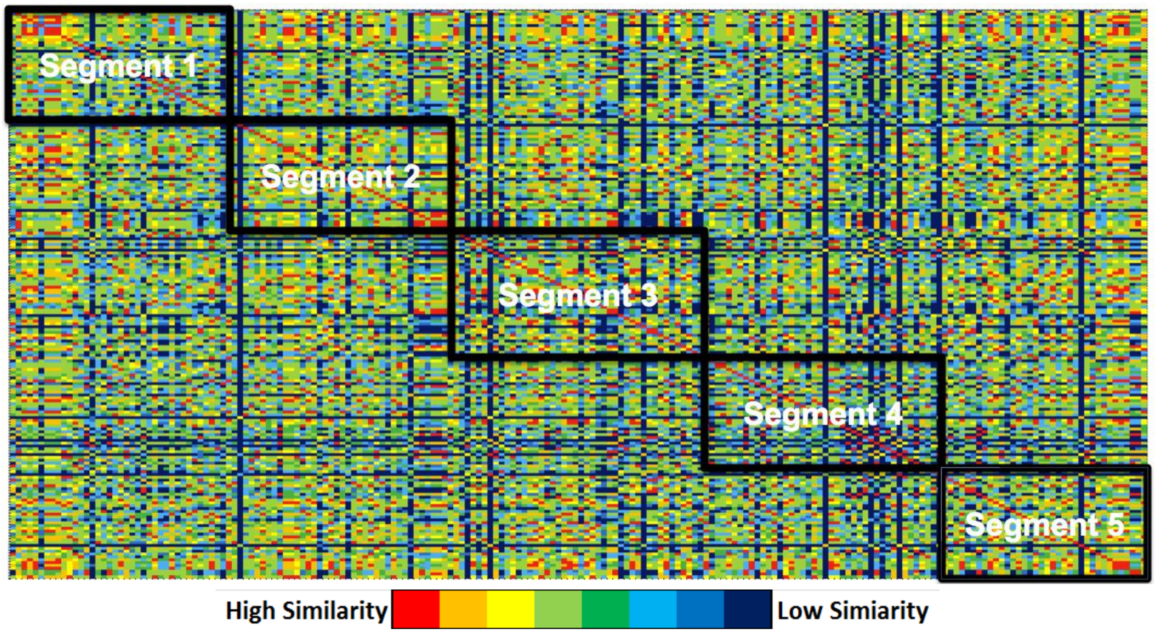


**Figure 4.19: Segments with randomly sorted input data**

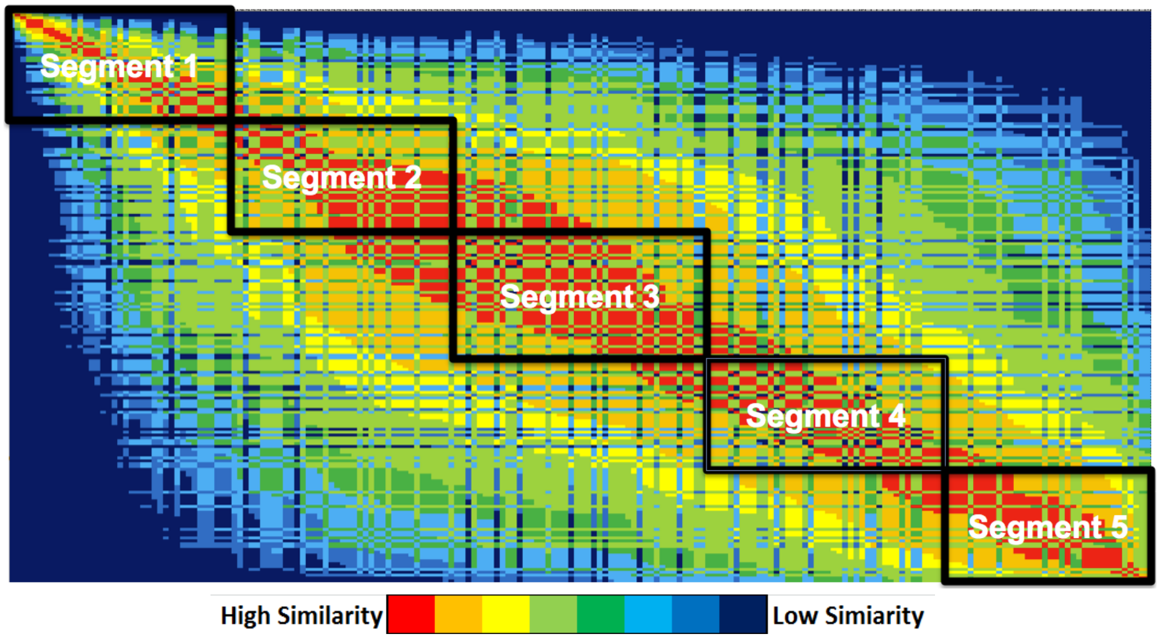


**Figure 4.20: Segments with similarity-based sorted input data**

**Determining Segment Size**: Segment size refers to the number of instances enters each segment. Determining segment size is based on the capability of optimization tools and model formulations to achieve solutions with high level of optimality (near-global or global optimum). Table 4.3 and Figure 4.21 show experimentation results using different segment sizes. Comparison among optimization solutions by different segment sizes shows that the segment size can affect the optimum solution. For instance, network deterioration index $(DI_N)$ of 35.84 is achieved for the case of 4000-instance model with segment size equal to 50. Increasing segment size to 400 resulted in $DI_N$ of 40.168 for the same model. Based on these experiments segments size of 50 to 100 is suggested to use for the TDSB asset renewal problem.

**Table 4.3: Optimization efficiency with different segment size**

| | Model Cases | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 800 instances | | | 2000 instances | | | 4000 instances | | | 6000 instances | | |
| Segment Size | No. of Seg. | Time (min) | CI | No. of Seg. | Time (min) | CI | No. of Seg. | Time (min) | CI | No. of Seg. | Time (min) | CI |
| **50** | 16 | 80 | 35.559 | 40 | 200 | 35.02 | 80 | 400 | 35.843 | 120 | 600 | 37.999 |
| **100** | 8 | 40 | 35.508 | 20 | 100 | 36.887 | 40 | 200 | 36.419 | 60 | 300 | 39.601 |
| **200** | 4 | 20 | 39.05 | 10 | 50 | 39.002 | 20 | 100 | 39.24 | 30 | 150 | 42.585 |
| **400** | 2 | 10 | 40.613 | 5 | 25 | 40.168 | 10 | 50 | 42.038 | 15 | 75 | 45.709 |

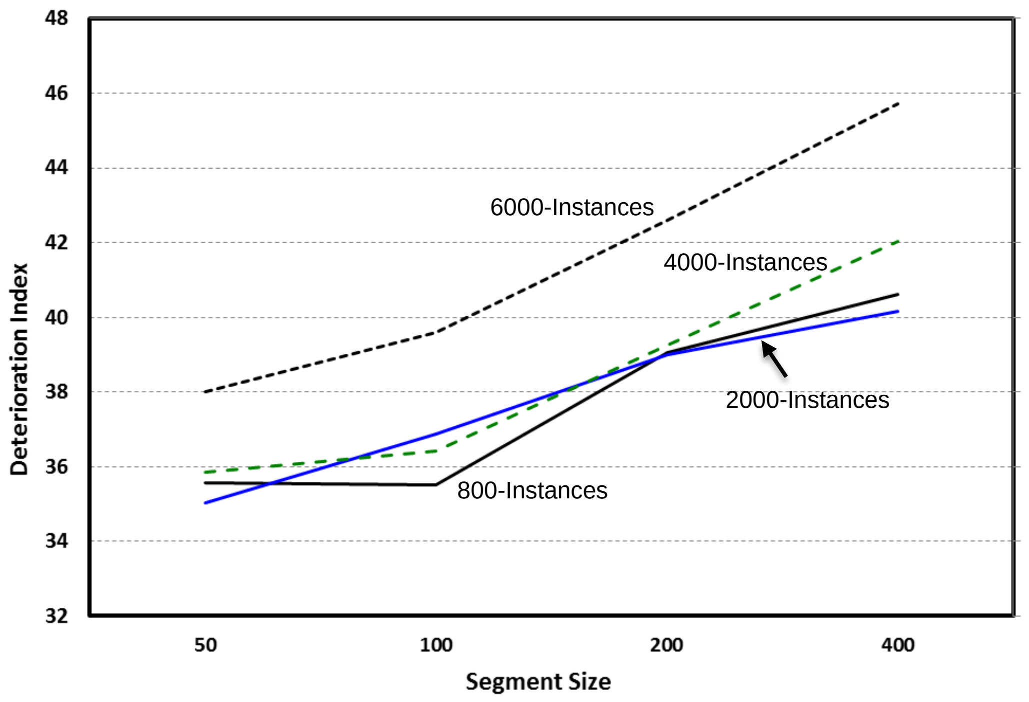


**Figure 4.21: Optimal solutions using different segment size**

**Segment Ordering**: After creating clustered segments (similarity-based generated segments) and determining optimum segment size, an important characteristic of segments called Criticality is defined. Criticality of a segment is the average of criticality indices of internal instances, which represents the effect of that segment on the overall network deterioration index. Segments with higher level of criticality have more negative impact on the overall network deterioration (increasing $DI_N$). Criticality index of an instance is calculated as follow:

$$CrI_i = \frac{RIF_i}{100} \times \frac{Average(EP_i)_k}{100} \qquad \forall\, k \in Planning\ Horizon\ [1, 2, 3, 4, 5] \tag{4.4}$$

where $CrI_i$ is the criticality index of instance i, $RIF_i$ is the relative importance factor for instance i, and $EP_i$ is expected performance. Criticality index has a value from 0 to 1 by which 0 represents low criticality and 1 represents high criticality instances. Criticality of a segment is defined as the average of criticalities of its internal instances.

$$SCr_j = \frac{\sum_{i=1}^{SS} CrI_i}{SS_j} \tag{4.5}$$

where $SCr_j$ is the Criticality of segment j and $SS_j$ is the segment size. To calculate the relative criticality ($RC_j$) of a segment following equation is used:

$$RC_j = \frac{SCr_j}{\sum_{i=1}^{NS} SCr_i} \tag{4.6}$$

*NS = Number of Segments*

After defining the relative criticality of all segments, segments are ordered from low relative criticality to high relative criticality. Sorting segments based on relative criticality facilitates the process of budget redistribution as discussed in the following subsection.

**Budget Allocation and Redistribution**: After ordering segments based on criticality values, available yearly budget is allocated to segments also based on their criticality. Budget allocation function based on segment relative criticality is defined as follow:

$$B_{kj} = TB_k \times RC_j \tag{4.7}$$

where $B_{kj}$ is allocated budget to segment j in year k, and $TB_k$ is the total available budget in year k, and $RC_j$ is the relative criticality of segment j.

Experimenting with three samples of segments with low relative criticality and high relative criticality is shown in Figure 4.22. Experiments shows that increasing the available budget (B1) for high criticality segments can improve the optimum solution.

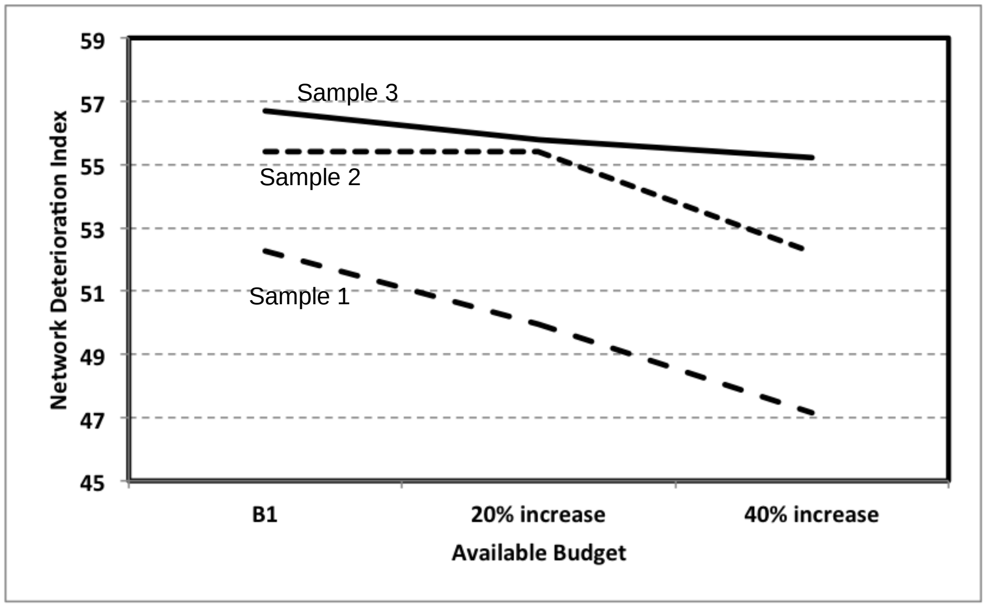


**FIGURE 4.22: Effect of increasing the available budget on network deterioration index for segments with high relative criticality**

After optimizing a segmented large-scale asset renewal problem, at each stage of optimization a small portion of the budget will be remained unallocated since the budget constraint cannot be met to the exact dollar. This remaining unallocated budget will accumulate from many segments and increased during the optimization process that becomes considerable amount of leftover money for a large-scale problem. Three ways are then possible to redistribute the remaining unallocated budget (Figure 4.23). In the first approach (Figure 4.23a) remaining budget from segment j will be added to available budget for segment j+1. Following this order during the process of optimization is resulted in more allocation of budget to segments with higher relative criticality since segments are ordered from low to high relative criticality. Combining sequential budget redistribution with budget allocation based on relative criticality leads to the following budget allocation function:

$$B_{kj} = TB_k \times RC_j + \left(B_{K(j-1)} - SRC_{k(j-1)}\right) \tag{4.8}$$

where $B_{k(j-1)}$ is the allocated budget to segment (j-1) in year k and $SRC_{k(j-1)}$ is segment (j-1) repair cost in year k.

In the second method of budget redistribution (Figure 4.23 b), all the leftover money from segments with low relative criticality is collected and then redistributed to segments with high relative criticality (e.g. $RC_j > 0.7$).

In the third approach, remaining budget is redistributed at the end of optimization process. After optimization is done, unallocated instances are separated from funded instances and all of the leftover money is used for them within a second optimization (Figure 4.23c).

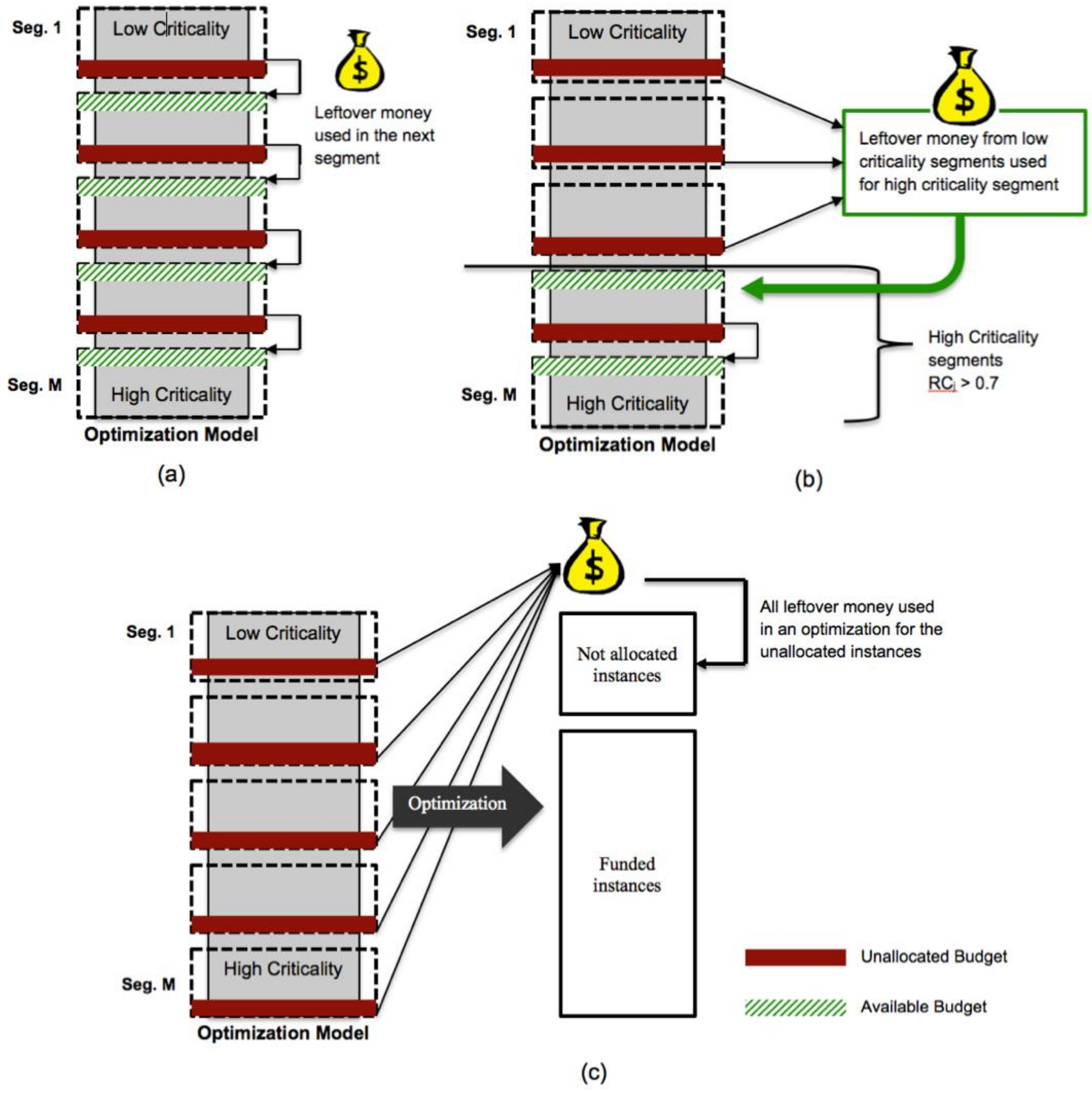


**Figure 4.23: Three methods of budget redistribution for in a segmented optimization process**

In order to meet the purpose of budget redistribution, which is to allocate maximum amount of available budget effectively to repair instances, any of these methods can be used. In the process of optimization with clustered segments, since segments are optimized in a sequential manner, the first method is selected for budget redistribution. Figure 4.24 shows a comparison between optimization with budget redistribution and optimization with no redistribution of leftovers from each segment. As it can be seen from this figure, redistribution of unallocated budgets improves the overall network deterioration index.

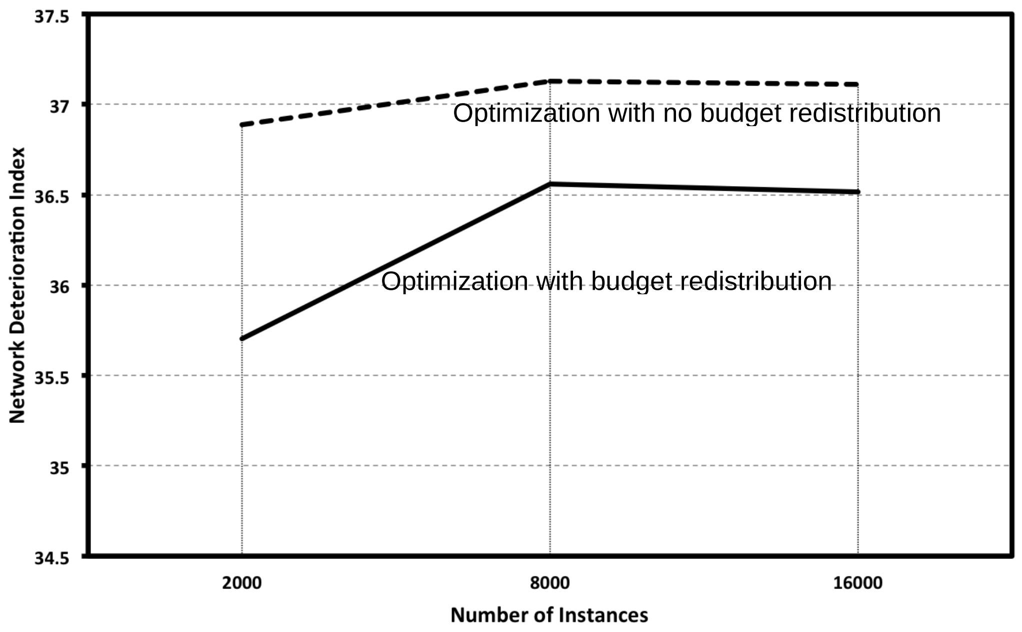


**Figure 4.24: Effect of budget redistribution on network deterioration index**

**Optimization Results using clustered segmentation**: By applying the proposed optimization mechanism on TDSB life cycle cost analysis model, network deterioration index of 36.1 is achieved for model with 50000 instances. Results show a huge improvement in optimization performance in compare to previous studies on the same prototype that achieved $DI_N$ of 48.9 for model with only 8000 instances. In addition, applying this mechanism also solves the problem of performance degradation. Figure 4.25 shows the results obtained from optimization with clustered segments and a comparison with previous studies optimization. As it can be seen from these results, optimization performance is almost constant by increasing the model's scales. Figure 4.26 shows the spreadsheet program developed for optimization with random segmentation.

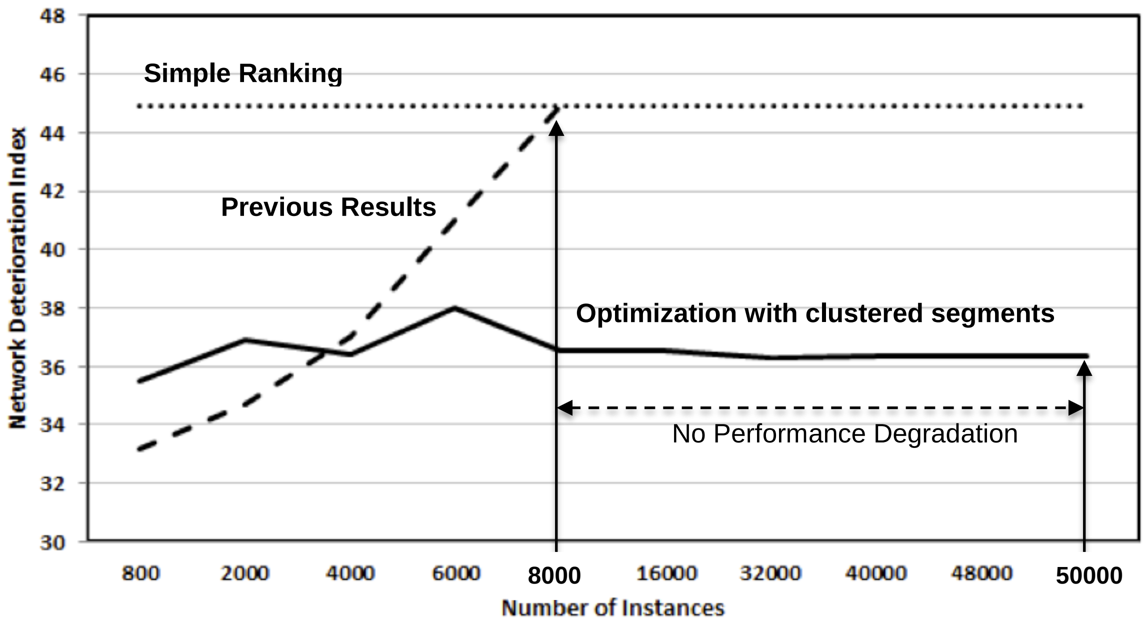
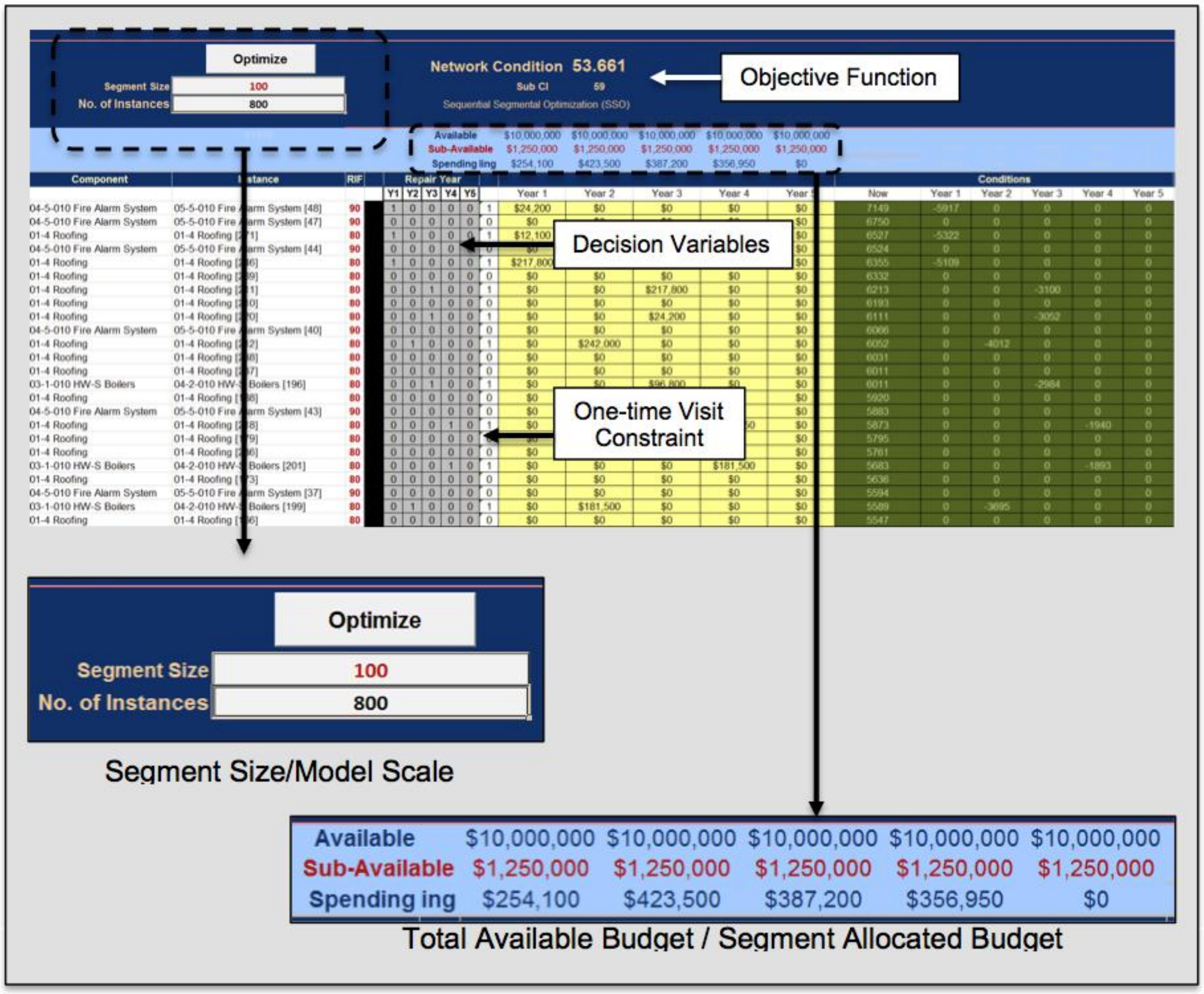


**Figure 4.25: Performance of optimization with clustered segments**


Objective Function
Decision Variables
One-time Visit Constraint
Segment Size/Model Scale
Total Available Budget / Segment Allocated Budget
Optimize

Segment Size 100
No. of Instances 800

Network Condition 53.661
Sub CI 59
Sequential Segmental Optimization (SSO)

| | | | | | |
|---|---|---|---|---|---|
| Available | $10,000,000 | $10,000,000 | $10,000,000 | $10,000,000 | $10,000,000 |
| Sub-Available | $1,250,000 | $1,250,000 | $1,250,000 | $1,250,000 | $1,250,000 |
| Spending ing | $254,100 | $423,500 | $387,200 | $356,950 | $0 |

| Component | Instance | RIF | Y1 | Y2 | Y3 | Y4 | Y5 | | Year 1 | Year 2 | Year 3 | Year 4 | Year 5 | Now | Year 1 | Year 2 | Year 3 | Year 4 | Year 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [48] | 90 | 1 | 0 | 0 | 0 | 0 | 1 | $24,200 | $0 | $0 | $0 | $0 | 7149 | -5917 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [47] | 90 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | [illegible] | [illegible] | [illegible] | $0 | 6750 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]1] | 80 | 1 | 0 | 0 | 0 | [illegible] | 1 | $12,100 | [illegible] | [illegible] | [illegible] | $0 | 6527 | -5322 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [44] | 90 | 0 | 0 | 0 | 0 | [illegible] | [illegible] | [illegible] | [illegible] | [illegible] | [illegible] | $0 | 6524 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]6] | 80 | 1 | 0 | 0 | 0 | 0 | 1 | $217,800 | [illegible] | [illegible] | [illegible] | $0 | 6355 | -5109 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]9] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6332 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]1] | 80 | 0 | 0 | 1 | 0 | 0 | 1 | $0 | $0 | $217,800 | $0 | $0 | 6213 | 0 | 0 | -3100 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]0] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6193 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]0] | 80 | 0 | 0 | 1 | 0 | 0 | 1 | $0 | $0 | $24,200 | $0 | $0 | 6111 | 0 | 0 | -3052 | 0 | 0 |
| 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [40] | 90 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6066 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]2] | 80 | 0 | 1 | 0 | 0 | 0 | 1 | $0 | $242,000 | $0 | $0 | $0 | 6052 | 0 | -4012 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]8] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6031 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]7] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 6011 | 0 | 0 | 0 | 0 | 0 |
| 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [196] | 80 | 0 | 0 | 1 | 0 | 0 | 1 | $0 | $0 | $96,800 | $0 | $0 | 6011 | 0 | 0 | -2984 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [1[illegible]8] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | [illegible] | [illegible] | [illegible] | $0 | 5920 | 0 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [43] | 90 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | [illegible] | [illegible] | [illegible] | $0 | 5883 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]8] | 80 | 0 | 0 | 0 | 1 | 0 | 1 | $0 | [illegible] | [illegible] | [illegible] | $0 | 5873 | 0 | 0 | 0 | -1940 | 0 |
| 01-4 Roofing | 01-4 Roofing [1[illegible]9] | 80 | 0 | 0 | 0 | 0 | 0 | [illegible] | [illegible] | [illegible] | [illegible] | [illegible] | $0 | 5795 | 0 | 0 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [2[illegible]6] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | [illegible] | [illegible] | [illegible] | $0 | 5761 | 0 | 0 | 0 | 0 | 0 |
| 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [201] | 80 | 0 | 0 | 0 | 1 | 0 | 1 | $0 | $0 | $0 | $181,500 | $0 | 5683 | 0 | 0 | 0 | -1893 | 0 |
| 01-4 Roofing | 01-4 Roofing [1[illegible]3] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 5636 | 0 | 0 | 0 | 0 | 0 |
| 04-5-010 Fire Alarm System | 05-5-010 Fire Alarm System [37] | 90 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 5594 | 0 | 0 | 0 | 0 | 0 |
| 03-1-010 HW-S Boilers | 04-2-010 HW-S Boilers [199] | 80 | 0 | 1 | 0 | 0 | 0 | 1 | $0 | $181,500 | $0 | $0 | $0 | 5589 | 0 | -3695 | 0 | 0 | 0 |
| 01-4 Roofing | 01-4 Roofing [1[illegible]6] | 80 | 0 | 0 | 0 | 0 | 0 | 0 | $0 | $0 | $0 | $0 | $0 | 5547 | 0 | 0 | 0 | 0 | 0 |

**Figure 4.26: Spreadsheet for optimization with clustered segments**

## 4.5. Comparison among Different Segmentation Approaches

As discussed in this section, different segmentation methods can be used with genetic algorithms. These methods include: 1) random segmentation, which divides an optimization problem into several segments with random internal data and allocates budget based on segments size; 2) model compression that optimizes a compressed model contains representative instances rather than the large-scale base model; and 3) clustered segmentation, which creates segments based on similarities while considering the best segment size, and allocates budget based on characteristics of segments and redistributes the remaining funds sequentially or at the end of the optimization process. Following table shows a comparison between segmentation methods discussed in this study.

**Table 4.4: Comparison among segmentation methods**

| | | | **Similarity-Based** | |
|---|---|---|---|---|
| | | **Random Segmentation** | **Model Compression** | **Clustered Segmentation** |
| **Performance** | **800-instances** | 39.05 | 41.043 | 35.508 |
| | **Other Cases** | ▪ Better quality results as compared to no segmentation<br>▪ Suffering from performance degradation in larger cases | N/A | ▪ Best optimization results for large-scale cases<br>▪ No performance degradation |
| **Comments** | | ▪ As a good starting point, highlights the effectiveness of segmentation approach<br>▪ Not good enough for using in larger scale situations<br>▪ Using the same segment size and budget allocation methods as used in clustered segmentation approach can improve results further | ▪ Better quality results as compared to simple ranking<br>▪ Poor results as compared to other optimization methods and accordingly is not used for larger size problems | ▪ Selected as the best segmentation approach<br>▪ Optimized 50,000-instances case with huge improvements<br>▪ Applicable to larger size problems with no degradation in performance |

## 4.6. Summary and Conclusions

In this chapter, three possible formulations for a large-scale asset renewal problem are investigated. Also, performance of each formulation is tested on two GA-based optimization tools, Evolver and Risk Solver Platform, to determine the best formulation and tool to use for large-scale optimization. Based on experiments, Evolver was selected as the best tool and step-wise-binary formulation as the best formulation. Next, segmentation introduced as a mechanism for handling very large-scale problems. Two different segmentation methods, Random segmentation and Similarity-based segmentation were discussed in detail. Based on various experiment, clustered segmentation proposed as the best segmentation mechanism. Comparison between optimization results obtained from proposed optimization mechanism of this research and the previous results indicated a significant improvement in optimization performance. Network deterioration index of 36.1 achieved for 50000-instance case with no performance degradation by increasing problem size.

# Chapter 5

# Large-Scale Optimization Using Advanced Mathematical Tools

## 5.1. Introduction

In this chapter, performance of an advanced mathematical modeling and optimization tool, ‘GAMS/CPLX’, is investigated on large-scale asset renewal problems. Modeling the asset renewal problem and the optimization techniques are discussed in detail. In addition, the results of using GAMS/CPLEX are compared with the results obtained by using GA + segmentation as discussed in previous chapter.

## 5.2. Mathematical Programming (MP): Advanced Tools

Mathematical representation and modeling of a real-life problem can be used to study the behavior of a system and to find a solution that enables the system to best meets its goals (W.L. Winston & M. Venkataramanan 2003). Different mathematical optimization tools have therefore been developed to help engineers. Simple optimization tools such as excel-solver or other simple solvers are only capable of handling simple LP problems. Advanced tools such as GAMS or IBM ILOG optimizer are capable of modeling and solving more complex problems using their sophisticated solver engines. To optimize TDSB asset renewal problem, an advanced optimization tool, GAMS/CPLEX, is used in this chapter. Following is a brief description of this tool and its solver engine:

**GAMS**: The General Algebraic Modeling System (GAMS) is a high-level modeling system for mathematical programming and optimization. It consists of a language compiler and a stable of integrated high-performance solvers. GAMS is tailored for complex, large scale modeling applications, and allows users to build large maintainable models that can be adapted quickly to new situations (GAMS user guide 2010). GAMS was mainly motivated and developed to improve the optimization performance in solving large mathematical programming problems by:

- Providing a high-level language for the compact representation of large and complex models
- Allowing changes to be made in model specifications simply and safely
- Allowing unambiguous statements of algebraic relationships
- Permitting model descriptions that are independent of solution algorithms

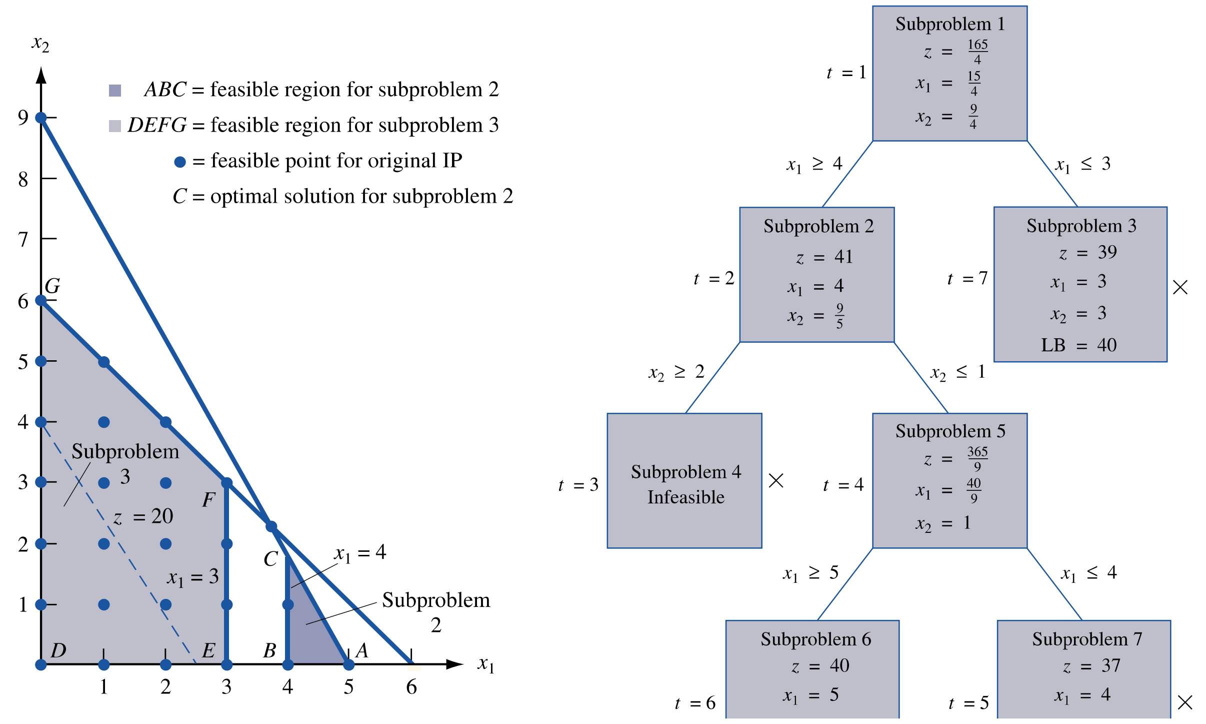
With the GAMS mathematical modeling tool, different solvers can be used. For large-scale asset renewal problems one of the powerful solvers called CPLEX is used.

**CPLEX**: CPLEX optimizer is designed to solve large and difficult linear, quadratically constrained and mixed integer programming problems quickly and with minimal user intervention (CPLEX user guide 2007). CPLEX is also considered as a robust and reliable optimization tool that delivers the power needed to solve very large-scale real world problems (CPLEX user guide 2007). In the next subsection detailed description of CPLEX optimization algorithm is discussed.

### 5.2.1. How GAMS/CPLEX Works

Integer programming (IP) problems require dramatically more amount of processing time and calculations in compare to LP or NLP problems (CPLEX12 User Guide). This fact is basically due to the methods used for solving this type of problems. CPLEX uses a branch-and-bound method, which solves series of LP sub-problems to reach the final solution. Since even a small integer programming problem can generate lots of sub-problems, it requires huge amount of physical memory and calculations to solve the problem (Winston & M. Venkataramanan 2003; CPLEX12 User Guide).

**Branch and Bound Method**: Most integer programming problems are solved by using branch-and-bound techniques. Branch-and-bound methods solve an IP problem by generating LP sub-problems (LP relaxation). If the solution obtained from a relaxed sub-problem is an integer solution, then it is a candidate for being the optimal solution to the problem (W.L. Winston & M. Venkataramanan 2003). Figure 5.1 shows the process of applying branch-and-bound method to an IP problem. After solving all sub-problems, candidate solutions with integer results are compared and the optimal solution is obtained base on the direction of the optimization (minimization or maximization) (Figure 5.1.b).



$x_2$
$ABC$ = feasible region for subproblem 2
$DEFG$ = feasible region for subproblem 3
● = feasible point for original IP
$C$ = optimal solution for subproblem 2
G
F
C
D
E
B
A
Subproblem 3
$z = 20$
$x_1 = 3$
$x_1 = 4$
Subproblem 2
$x_1$
Subproblem 1 $z = \frac{165}{4}$ $x_1 = \frac{15}{4}$ $x_2 = \frac{9}{4}$
$t = 1$
$x_1 \geq 4$
$x_1 \leq 3$
Subproblem 2 $z = 41$ $x_1 = 4$ $x_2 = \frac{9}{5}$
$t = 2$
Subproblem 3 $z = 39$ $x_1 = 3$ $x_2 = 3$ LB = 40
$t = 7$
$x_2 \geq 2$
$x_2 \leq 1$
Subproblem 4 Infeasible
$t = 3$
Subproblem 5 $z = \frac{365}{9}$ $x_1 = \frac{40}{9}$ $x_2 = 1$
$t = 4$
$x_1 \geq 5$
$x_1 \leq 4$
Subproblem 6 $z = 40$ $x_1 = 5$
$t = 6$
Subproblem 7 $z = 37$ $x_1 = 4$
$t = 5$

**Figure 5.1: a) Branching and creating sub-problems, b) Branch-and-bound tree (W.L. Winston & M. Venkataramanan 2003)**

One of the main reasons that IP problems are very tedious to solve is the huge amount of calculations required for large number of sub-problems. For that reason, modelers usually seek for a proper formulation, which requires less calculation and processing time (Wolsey. L. 1989).

### 5.2.2. Characteristics of Easy-to-Solve Models (Called Strong IP Formulations)

Generally, a strong integer programming formulation is a formulation that can lead to optimal solution with less computational effort. The way that an IP problem is formulated can significantly affect the optimization performance (Wolsey. L. 1989). There are different ways to identify an easy-to-solve formulation. Some of the key elements to a good IP formulation are as follow (Wolsey. L. 1989; Winston & M. Venkataramanan 2003):

1. Linearity
2. Decision Variables
3. Simple Calculations and Relationships
4. Strength of the LP Relaxation

**Linearity**: Linearity of objective function and constraint equations enables IP solvers to use simpler and faster algorithms, such as simplex algorithm, to solve sub-problems (W.L. Winston & M. Venkataramanan 2003). In formulating IP problems, defining objective function and constraints as a linear function of decision variables proved to be effective in increasing optimization performance (Wolsey. L. 1989).

**Decision Variables**: Using variables with minimum range of variations can be very helpful in solving IPs (W.L. Winston & M. Venkataramanan 2003). In different applications of integer programming, such as minimum cost network flow problems (MCNFP) or facility location a problem, binary representation of variables (if possible), which is the simplest form of an integer variable, proved to be very effective and is recommended to use in developing a good formulation (W.L. Winston & M. Venkataramanan 2003).

**Simple Calculations and Relationships:** Relationships amongst input data, decision variables, and model equations (objective function, constraints, loops statements, and conditional statements) can significantly affect the process of solving an IP problem (CPLEX12 User Guide) and simplicity of interrelationships and calculations is another key to increase optimization performance (Wolsey. L. 1989). To simplify relationships and calculations one can try to terminate redundancies and complex relationships (if possible), or can use other software parallel to the optimization tool to do some part of calculation by exchanging data between both software.

**The strength of LP relaxation**: The LP relaxation of an IP problem is obtained by relaxing the constraints from discrete to continuous values. For instance, in a binary integer programming (BIP) problem, a variable in the LP relaxation can take a real value between zero to 1 ($0 \leq x \leq 1$). In order to examine the strength of relaxation, the feasible region of the LP relaxation is compared to the convex hull of the problem (Figure 5.2). Given a set of feasible integer solutions (X), the corresponding convex hull (CH(X)) is the smallest polyhedron containing X. Considering $P$ as the feasible region of the LP relaxation, IP formulation is strong when the difference between P and CH is minimized (Wolsey. L. 1989) (Figure 5.2).

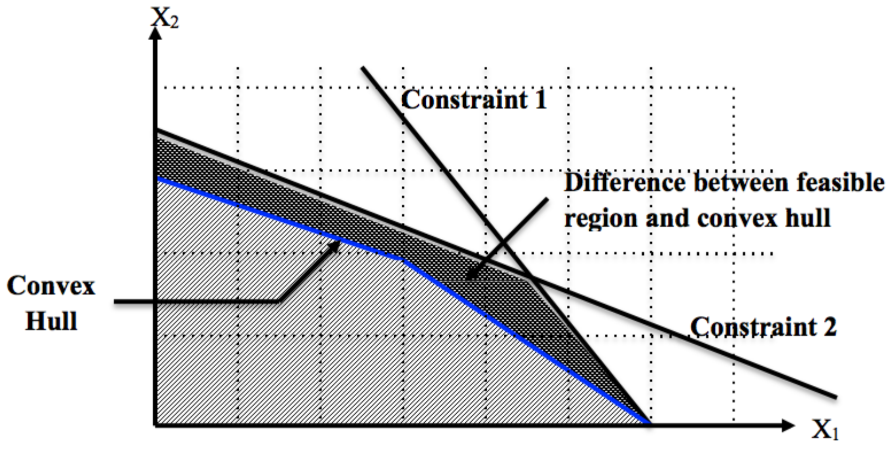


**Figure 5.2: Convex hull and feasible region for the LP relaxation**

## 5.3. Modeling the Asset Renewal Problem for GAMS/CPLEX

Based on previous descriptions in chapter 4, same formulations used for GA-based optimization are also candidates for mathematical programming experimentation. Figure 5.3 shows three formulations used before.

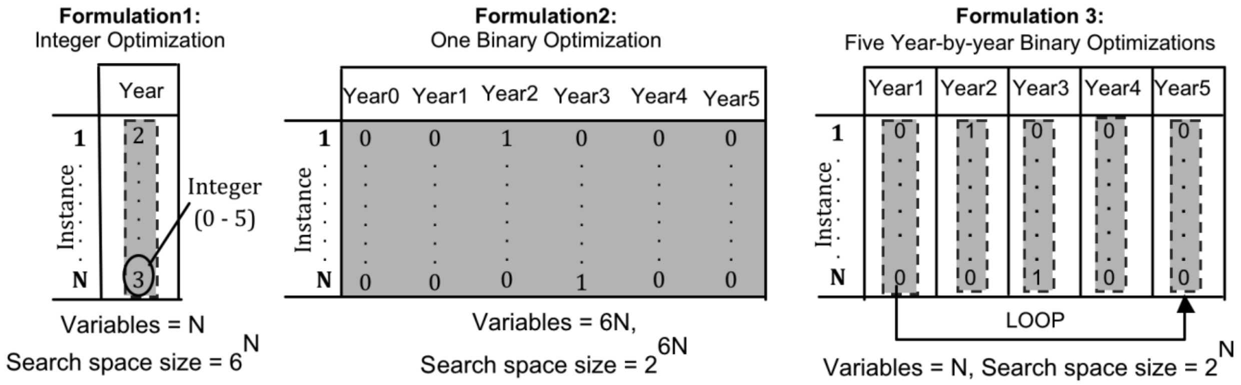


**Figure 5.3: Formulations used for modeling the asset renewal problem**

All of these models share following characteristics (Section 5.2):

- Static Model: Optimization for 5-year planning horizon (no multiple periods of time).
- Linear Model: linear functions re used for defining constraints and objective function.
- Integer Pogromming (IP): Decision variables are allowed to have discrete values.
- Stochastic Model: The values for variables and objective function are also assumed to be known. It is important to note that although the future deterioration indices (DIs) are determined by using deterministic Markovian models, but the model itself assumed to be stochastic.

Comparing these formulations with regard to characteristics of easy-to-solve IP formulation shows that all of them are reasonable formulations in terms of linearity. In terms of decision variables, binary formulations are better candidates since they are using the simplest form of a discrete variable (binary variable), which has the minimum level of variation. Between two binary formulations, the one-shot-binary formulation requires less internal calculations and can be coded using simple relationships, but the step-wise-binary formulation requires more complex statements, such as loop or conditional statements (Figure 5.3). Based on these comparisons one-shot-binary formulation is the best candidate among three suggested IP formulation, however, the amount of calculations and relationships for input data (deterioration indices, expected performance calculations, repair cost calculation) can still affect the optimization performance. To simplify calculations for a large-scale problem, the TDSB asset renewal model is coded in a way that exchanges data between both Microsoft Excel and GAMS/CPLEX as discussed in the next section.

### 5.3.1. Structure of the GAMS Model for the Asset Renewal Problem

The main components of the implemented model in GAMS are as follow:

1. ***Sets***: In the GMS model, the 5-year planning horizon and the network of instances are represented by $j$ and $i$ set declarations respectively.
2. ***Data***: All of the main data including parameters, tables, and scalars are presented in the model by using GDX files and direct connection to the original spreadsheet file contain the base model. These data include: instance repair cost (IRC) for the best repair scenario, improvement effect (IE), initial condition (IC), and yearly budget limit. Calculations related to all input data are as discussed in chapter 3.
3. ***Variables***: Decision variable are in the form of binary as discussed before.
4. ***Equations***: Objective function and two constraints, yearly budget limit and one time visit during the planning horizon, are defined as equation declarations by same equations used in chapter 3.

Since a huge database is used for large-scale problems, the model is defined in a way that extracts input data after calculations are done from the spreadsheet contain model information. Input data are linked to GAMS by using GAMS Data Exchange (GDX) files. As shown in Figure 5.4, in the process of optimization, the GAMS input spreadsheet file is generated from the original optimization model and calculated data (RIC, IE, and IC) are exported to GAMS by using GDX files. CPLEX is selected by GAMS as the solver engine and after optimization is done, results are exported again to excel or saved as a separate text file by using GAMS put writing facility.

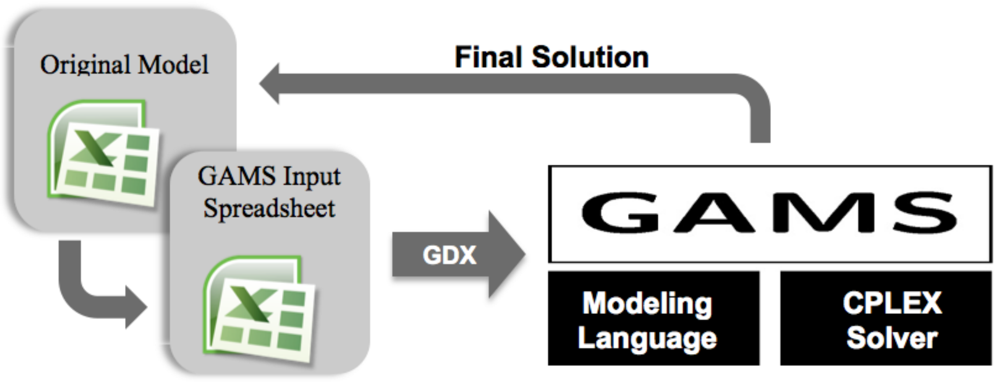
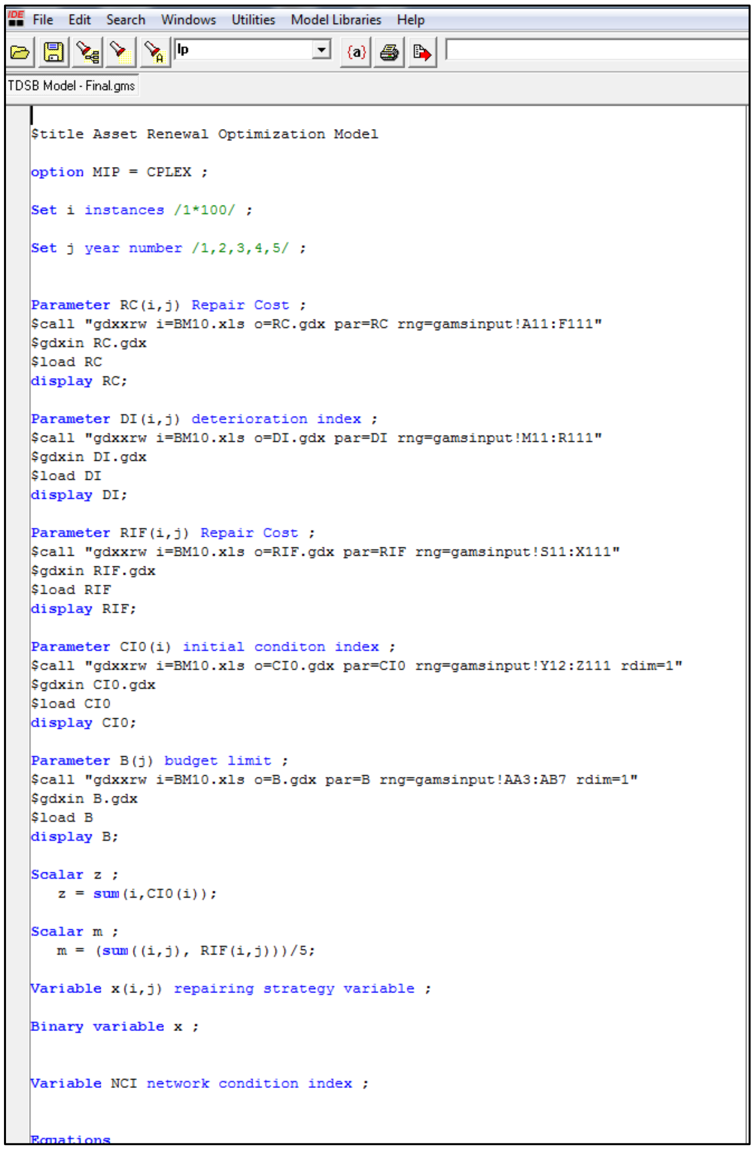


**Figure 5.4: Optimization process with GAMS/CPLEX**

Figure 5.5 shows the GAMS modeling environment and the optimization code for the asset renewal problem. Detailed coding is also presented in appendix A. Although the problem is in the form of IP, model is defined as MIP due to the CPLEX modeling regulations. After identifying that the model is pure integer CPLEX uses IP solution methods automatically. Although this model is dealing with huge amount of data and is very large-scale, it is compacted and simplified by using the interaction between GAMS and Microsoft Excel.



```
$title Asset Renewal Optimization Model

option MIP = CPLEX ;

Set i instances /1*100/ ;

Set j year number /1,2,3,4,5/ ;


Parameter RC(i,j) Repair Cost ;
$call "gdxxrw i=BM10.xls o=RC.gdx par=RC rng=gamsinput!A11:F111"
$gdxin RC.gdx
$load RC
display RC;

Parameter DI(i,j) deterioration index ;
$call "gdxxrw i=BM10.xls o=DI.gdx par=DI rng=gamsinput!M11:R111"
$gdxin DI.gdx
$load DI
display DI;

Parameter RIF(i,j) Repair Cost ;
$call "gdxxrw i=BM10.xls o=RIF.gdx par=RIF rng=gamsinput!S11:X111"
$gdxin RIF.gdx
$load RIF
display RIF;

Parameter CI0(i) initial conditon index ;
$call "gdxxrw i=BM10.xls o=CI0.gdx par=CI0 rng=gamsinput!Y12:Z111 rdim=1"
$gdxin CI0.gdx
$load CI0
display CI0;

Parameter B(j) budget limit ;
$call "gdxxrw i=BM10.xls o=B.gdx par=B rng=gamsinput!AA3:AB7 rdim=1"
$gdxin B.gdx
$load B
display B;

Scalar z ;
   z = sum(i,CI0(i));

Scalar m ;
   m = (sum((i,j), RIF(i,j)))/5;

Variable x(i,j) repairing strategy variable ;

Binary variable x ;


Variable NCI network condition index ;


Equations
```

Figure 5.5: GAMS modeling environment

## 5.4 Mathematical Optimization: Experimentations and Results

As discussed in previous section, one-shot-binary formulation is used to test the performance of GAMS/CPLEX on different size problems. Models with wide variety of sizes, from 10-

instances to 50000-instances case, are coded and optimized using GAMS/CPLEX. Detailed optimization results, model statistics and solution reports for all experiments are presented in appendix B. Table 5.1 also gives brief information about each experiment includes: model size, number of discrete variables, optimization time, number of iterations, and the objective value (network deterioration index). Figure 5.5 shows mathematical optimization results obtained from GAMS/CPLEX.

**Table 5.1: Optimization results using GAMS/CPLEX**

| Model Size (Instances) | Number of Discrete Variables | Generation Time (sec) | Execution Time (sec) | Iteration Count | Objective Value |
|---|---|---|---|---|---|
| **10** | 500 | 0.031 | 0.031 | 10 | 25.5 |
| **20** | 1000 | 0.031 | 0.031 | 25 | 21.387 |
| **40** | 2000 | 0.031 | 0.047 | 62 | 24.633 |
| **50** | 2500 | 0.031 | 0.031 | 64 | 22.743 |
| **100** | 5000 | 0.031 | 0.031 | 131 | 24.046 |
| **200** | 10000 | 0.062 | 0.062 | 238 | 25.132 |
| **400** | 20000 | 0.078 | 0.094 | 447 | 27.051 |
| **500** | 25000 | 0.109 | 0.125 | 505 | 31.646 |
| **800** | 40000 | 0.125 | 2.703 | 713 | 31.732 |
| **8000** | 400000 | 1.062 | 1.408 | 6171 | 31.686 |
| **16000** | 800000 | 2.135 | 2.938 | 12378 | 31.683 |
| **32000** | 1600000 | 4.39 | 6.031 | 24737 | 31.681 |
| **40000** | 2000000 | 5.672 | 7.704 | 31908 | 31.68 |
| **48000** | 2400000 | 6.703 | 9.135 | 38275 | 31.68 |
| **50000** | 2500000 | 6.75 | 9.281 | 41666 | 31.674 |

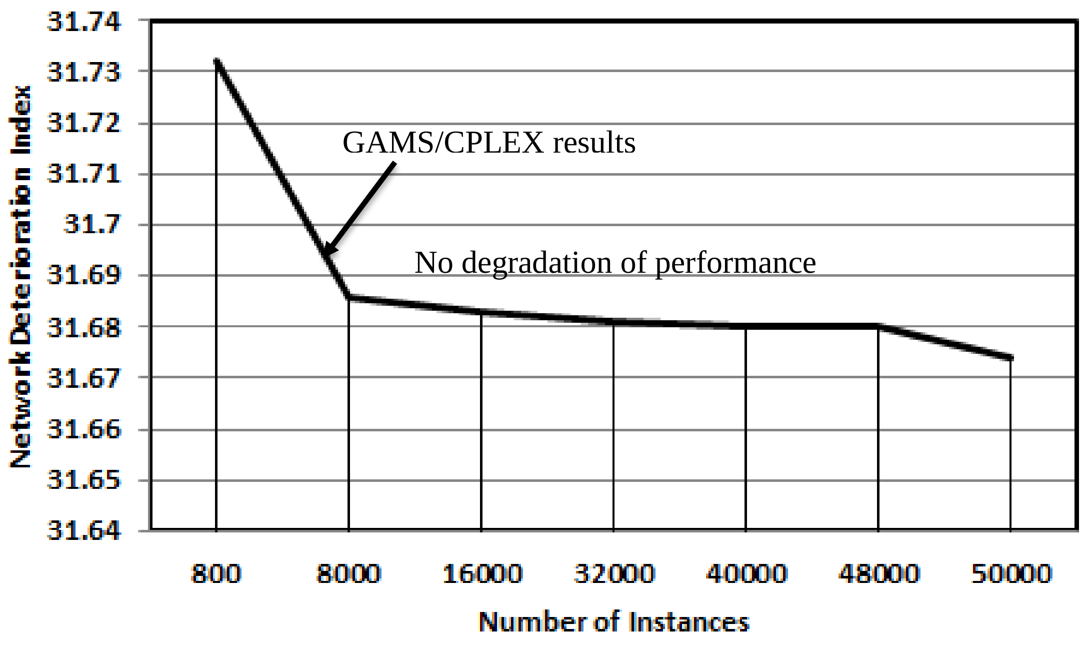


**Figure 5.6: Mathematical Optimization Performance (vertical scale is very narrow)**

In compare to GA-based optimization with segmentation, as shown in Figure 5.7, mathematical approach is more promising. Advanced tools and strong formulation are considered to be the main keys to the good optimization performance. Detailed investigation on the relaxation properties, convex hull and LP feasible region, can be done as an extension to this study (Chapter 6: Future Works) and is not discussed in this chapter. However, experiments with relaxed version of the TDSB problem revealed very close results to the integer programming results. Schematically, the expected CH and P for a simple two-dimensional can be shown as Figure 5.8. Although mathematical approach results are better in compare to genetic algorithms, GA + segmentation approach is still very useful and practical, specifically in the case of more complex and nonlinear large-scale asset renewal problems.

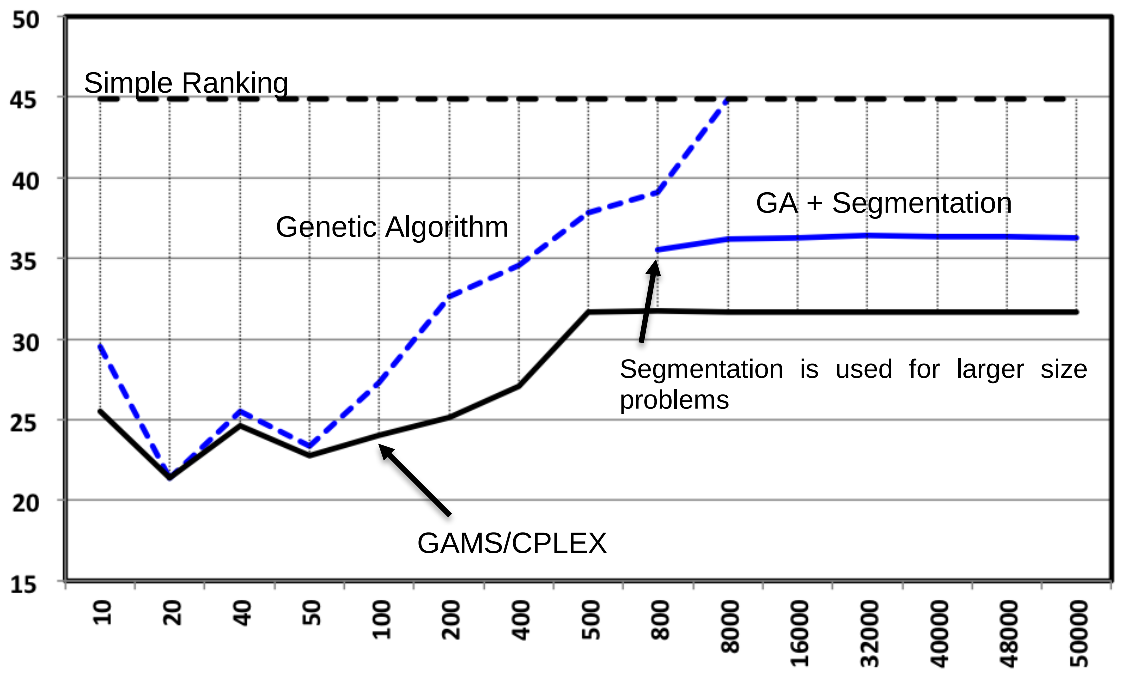


**Figure 5.7: Comparison between GAMS/CPLEX and GA + segmentation**

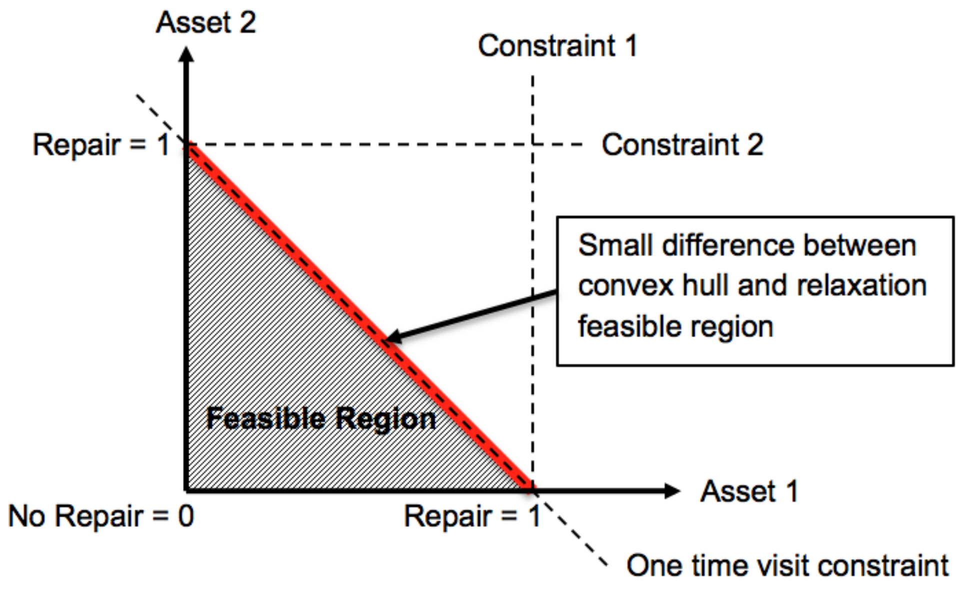

Asset 2
Constraint 1
Repair = 1
Constraint 2
Small difference between convex hull and relaxation feasible region
Feasible Region
No Repair = 0
Repair = 1
Asset 1
One time visit constraint

**Figure 5.8: Convex hull and feasible region for a two-dimensional simple case**

## 5.5. Summary and Conclusions

In this chapter, the performance of an advanced mathematical optimization tool (GAMS/CPLEX) discussed on large-scale asset renewal problems. The mathematical model is defined in a way that the initial spreadsheet contains asset inventory information, the mathematical modeling software (GAMS), and the optimization tool (CPLEX) collaborate to increase the efficiency of optimization.

Solution quality of mathematical approached was better as compared to GA-based optimization. The promising results are attributed to high-level computational capabilities of the optimization tools and also to the simple and easy-to-solve IP formulation of the asset renewal problem. In the case of the TDSB school renewal program, the mathematical approach proved to be successful, however, GA-based optimization, particularly segmentation, is still an effective alternative to handle more complex and non-linear large-scale problems.

# Chapter 6

# Conclusions and Future Research

## 6.1. Conclusions

Reliable and efficient public infrastructure is indispensible for the existence of a prosperous and modern society. Preserving the serviceability of public infrastructure is a highly challenging task due to the deteriorations caused by aging and lack of asset renewal funding. Consequently, asset management systems (AMSs), involving four main functions: condition assessment, deterioration modeling, repair modeling, and life cycle cost analysis (LCCA), have been introduced to support decision makers in handling renewal actions cost-effectively. During the life cycle cost analysis process all information provided by other functions are utilized to find the optimum solution of an asset renewal problem. One of the main difficulties of most AMSs is that they are not able to provide a good life cycle cost analysis for real-life problems due to the complexities and large number of components. Therefore, the main goal of this research was to investigate practical approaches in dealing with very large-scale asset renewal optimization problems involving at least 50,000 components.

An asset management framework developed by Hegazy and Elhakeem (2010) suggested an integrated bilevel life cycle optimization for handling the renewal planning problem related to Toronto District School Board (TDSB), which was able to optimize the problem for only

8000 components. For expanding to larger size problems, this study proposed a methodology that investigates both evolutionary-based optimization techniques and also advanced mathematical tools.

On the evolutionary side, three possible formulations of the asset renewal problem were investigated. In addition, performances of each formulation were tested on two GA-based optimization tools, Evolver and Risk Solver Platform, to determine the best formulation and suitable tools to use for large-scale optimization. Based on the experiments, Evolver with a step-wise-binary formulation was selected. Next, segmentation was introduced as a possible mechanism for handling very large-scale problems. Two different segmentation methods, Random segmentation and Similarity-based segmentation, were discussed in detail. Based on various experiments, the best segmentation approaches, effective budget redistribution mechanism, best segment size and segment ordering were selected and tested on problems of varying sizes. A comparison between optimization results obtained by the proposed segmentation methodology and previous results showed significant improvement in optimization results. A network deterioration index of 36.1 was achieved for the 50000-instance case with no performance degradation.

In addition to genetic algorithms, performance of an advanced mathematical optimization tool was discussed and investigated on large-scale asset renewal problems. The mathematical model was defined in a way that the initial spreadsheet contains asset inventory information, and the mathematical optimization tool (GAMS/CPLEX) collaborated to increase the efficiency of optimization. Based on the experimentations, mathematical optimization approach leaded to better quality results as compared to GA-based optimization. Good performance of mathematical approach is attributable to the high-level computational ability of GAMS and to the simplicity of the IP formulation (easy-to-sole fotmulation). The study, in essence, achieved its objective of optimizing up to 50,000 assets simultaneously, using either the segmented GA approach or using the developed GAMS model.

Based on the current experiments and results, this research contributes the following:

- **Developed a mechanism for handling very large-scale real-life asset renewal problems using GA with segmentation**: By applying the GA-based optimization mechanism introduced in this research, asset managers are able to use optimization for even very large-scale problems and allocate the available budget more cost-effectively in comparison to using ranking methods, which are common for large size problems.

- **Introduced an effective optimization model using an advanced mathematical tool for handling large-scale problems**: By investigating advanced mathematical tools, this research demonstrated that utilizing appropriate mathematical tools and strong formulations can lead to high quality results even for large-scale asset renewal problems.

- **Better understanding of problem formulations**: This research investigated various problem formulations through numerous experiments on several optimization tools and problems. These experiments and results provided a clearer insight and better understanding of good formulation for solving large-scale problems.

## 6.2. Future Research

Future extensions of this research could cover various improvements related to asset management and life cycle cost, including:

- Experiment with larger sized problems involving mixed assets, larger planning horizons, and more complex situations which model more than a single visit to each asset;
- Examine alternative evolutionary algorithms and test their performance with segmentation;
- Investigate further the potential benefits of using clustering techniques during the segmentation process;
- Study in detail the mathematical optimization approaches used by advanced solvers and relaxation properties of the model;
- Use the proposed optimization methodologies in other large-scale asset renewal problems, such as bridge management, sewer networks, or pipe line systems;
- Develop a post-optimization module for checking the acceptability of final solutions with regard to variation of acceptability levels;

- Link the optimization results to other systems for day-to-day maintenance planning;
- Investigate parallel computing methods and cloud computing tools that could speed the optimization process.

# Appendix A

# GAMS Model for TDSB Asset Renewal Problem

Following is the GAMS code for the TDSB asset renewal problem. Model is coded in such way that integrates with excel spreadsheets contain the original model used in chapter 4 with GA-based optimization process, and also the GAMS input data spreadsheet (`BM10.xls`). Using GDX files leads to create more compact and efficient mathematical model. As discussed in chapter 5, model has four main components including: 1) sets, 2) data, 3) variables, and 4) equations.

```
$title TDSB Asset Renewal Optimization Model (800 instances)
* using GAMS/CPLEX as the solver
option MIP = CPLEX ;

Set i instances /1*800/ ;
Set j year number /1,2,3,4,5/ ;

* input data imported from BM10.xls (GAMS input spreadsheet)
Parameter IRC(i,j) Instance Repair Cost ;
$call "gdxxrw i=BM10.xls o=IRC.gdx par=IRC rng=gamsinput!A11:F811"
$gdxin IRC.gdx
$load IRC
display IRC;

Parameter IE(i,j) Improvement Effect ;
$call "gdxxrw i=BM10.xls o=IE.gdx par=IE rng=gamsinput!M11:R811"
$gdxin IE.gdx
$load IE
display IE;

Parameter RIF(i,j) Repair Cost ;
$call "gdxxrw i=BM10.xls o=RIF.gdx par=RIF rng=gamsinput!S11:X811"
$gdxin RIF.gdx
$load RIF
```

```
display RIF;

Parameter CI0(i) initial conditon Index ;
$call "gdxxrw i=BM10.xls o=CI0.gdx par=CI0 rng=gamsinput!Y12:Z811
rdim=1"
$gdxin CI0.gdx
$load CI0
display CI0;

Parameter B(j) Yearly Budget Limit ;
$call "gdxxrw i=BM10.xls o=B.gdx par=B rng=gamsinput!AA3:AB7 rdim=1"
$gdxin B.gdx
$load B
display B;

Scalar z ;
   z = sum(i,CI0(i));
Scalar m ;
   m = (sum((i,j), RIF(i,j)))/5;

* binary decision variables (BIP formulation)
Variable x(i,j) Decision Variable for Renewal Action ;
Binary variable x ;

Variable DIN Network Deterioration Index ;

* objective and constraints
Equations
        cost(j)      total cost in year j
        SRC(i)       one time visit during the planning horizon
        Condition    objective function ;

  cost(j)..       sum(i, x(i,j)*IRC(i,j)) =l= B(j) ;
  SRC(i)..        sum(j, x(i,j)) =l= 1  ;
  Condition.. DIN =e= (z + sum((i,j), x(i,j)*IE(i,j)*RIF(i,j)))/m ;

* solving asset renewal model using GAMS/CPLEX
Model TDSB /all/ ;

solve TDSB using mip minimizing DIN ;
```

```
Display x.l, DIN.l ;

solve TDSB using mip minimizing DIN ;
execute_unload "result.gdx" x.l
execute 'gdxxrw.exe result.gdx o=BM10.xls var=x.l rng=gamsresult!'

file results /results.txt/ ;
put results;
loop((i,j), put x.l(i,j)/);
```

# Appendix B

## GAMS/CPLEX Optimization Output Data

Using GAMS modeling software and its powerful CPLEX solver following results are obtained for different size problems. Due to the large scale of the models, complete GAMS output is not presented here. Instead, the solution reports and model statistics for cases from 800 to 50000 instances are presented here:

**800-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS           806
BLOCKS OF VARIABLES           2     SINGLE VARIABLES         4,001
NON ZERO ELEMENTS        11,861     DISCRETE VARIABLES       4,000


GENERATION TIME      =        0.125 SECONDS      5 Mb
EXECUTION TIME       =        0.156 SECONDS      5 Mb


Solution Report     SOLVE TDSB Using MIP From line 65


               S O L V E      S U M M A R Y

     MODEL   TDSB                OBJECTIVE  NCI
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  65
```

```
**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE               31.7319

 RESOURCE USAGE, LIMIT          0.306      1000.000
 ITERATION COUNT, LIMIT       713    2000000000
```

**8000-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS        8,006
BLOCKS OF VARIABLES           2     SINGLE VARIABLES       40,001
NON ZERO ELEMENTS       118,601     DISCRETE VARIABLES     40,000


GENERATION TIME      =        1.062 SECONDS     18 Mb
EXECUTION TIME       =        1.468 SECONDS     18 Mb


Solution Report     SOLVE TDSB Using MIP From line 65


               S O L V E      S U M M A R Y

     MODEL   TDSB                OBJECTIVE  NCI
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  65


**** SOLVER STATUS     1 Normal Completion
```

```
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE               31.6861


 RESOURCE USAGE, LIMIT          2.013      1000.000
 ITERATION COUNT, LIMIT      6171    2000000000
```

**16000-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS       16,006
BLOCKS OF VARIABLES           2     SINGLE VARIABLES       80,001
NON ZERO ELEMENTS       237,201     DISCRETE VARIABLES     80,000


GENERATION TIME      =        2.125 SECONDS     33 Mb
EXECUTION TIME       =        2.938 SECONDS     33 Mb


Solution Report     SOLVE TDSB Using MIP From line 66


               S O L V E     S U M M A R Y

     MODEL   TDSB                OBJECTIVE  NCI
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  66

**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE               31.6829
```

```
 RESOURCE USAGE, LIMIT          5.228      1000.000
 ITERATION COUNT, LIMIT     12378    2000000000
```

**32000-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS       32,006
BLOCKS OF VARIABLES           2     SINGLE VARIABLES      160,001
NON ZERO ELEMENTS       474,401     DISCRETE VARIABLES    160,000


GENERATION TIME      =        4.390 SECONDS     61 Mb
EXECUTION TIME       =        6.031 SECONDS     61 Mb


Solution Report     SOLVE TDSB Using MIP From line 66


               S O L V E     S U M M A R Y

     MODEL   TDSB                OBJECTIVE  NCI
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  66


**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE               31.6811


 RESOURCE USAGE, LIMIT         16.865      1000.000
 ITERATION COUNT, LIMIT     24737    2000000000
```

**40000-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS       40,006
BLOCKS OF VARIABLES           2     SINGLE VARIABLES      200,001
NON ZERO ELEMENTS       593,001     DISCRETE VARIABLES    200,000


GENERATION TIME      =        5.672 SECONDS     75 Mb
EXECUTION TIME       =        7.734 SECONDS     75 Mb


Solution Report     SOLVE TDSB Using MIP From line 66


               S O L V E     S U M M A R Y

     MODEL   TDSB                OBJECTIVE  NCI
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  66


**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE               31.6801


 RESOURCE USAGE, LIMIT         27.772      1000.000
 ITERATION COUNT, LIMIT     31908    2000000000
```

**48000-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS       48,006
BLOCKS OF VARIABLES           2     SINGLE VARIABLES      240,001
NON ZERO ELEMENTS       711,601     DISCRETE VARIABLES    240,000


GENERATION TIME      =        6.703 SECONDS     90 Mb
EXECUTION TIME       =        9.125 SECONDS     90 Mb


Solution Report     SOLVE TDSB Using MIP From line 66

               S O L V E      S U M M A R Y

     MODEL   TDSB              OBJECTIVE  NCI
     TYPE    MIP               DIRECTION  MINIMIZE
     SOLVER  CPLEX             FROM LINE  66


**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE              31.6798


 RESOURCE USAGE, LIMIT         37.666      1000.000
 ITERATION COUNT, LIMIT     38277    2000000000
```

**50000-instances:**

```
MODEL STATISTICS

BLOCKS OF EQUATIONS           3     SINGLE EQUATIONS       50,006
BLOCKS OF VARIABLES           2     SINGLE VARIABLES      250,001
NON ZERO ELEMENTS       741,243     DISCRETE VARIABLES    250,000



GENERATION TIME      =        6.750 SECONDS     93 Mb
EXECUTION TIME       =        9.281 SECONDS     93 Mb



Solution Report     SOLVE TDSB Using MIP From line 66

               S O L V E     S U M M A R Y

     MODEL   TDSB                OBJECTIVE  NCI
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  66



**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE               31.6740

 RESOURCE USAGE, LIMIT         35.271      1000.000
 ITERATION COUNT, LIMIT     41666    2000000000
```

# References

Al-Battaineh, H. T., AbouRizk, S. M., Siu, K. L., and Allouch, M., (2005). "*The Optimization of Infrastructure Budget Allocation Using Genetic Algorithms*." Proceedings of 1st CSCE Specialty Conference on Infrastructure Technologies, Management, and Policies, CSCE, Toronto, Ontario, Canada, June 2-4, FR-173.

ASCE (2009). "*http://www.infrastructurereportcard.org/"* Accessed January 14, 2011.

ASCE (2005), "*http://apps.asce.org/reportcard/2005/page.cfm?id=103"* Accessed January 14, 2011.

Cheung, S. -., Tong, T. K. -., & Tam, C. -. (2002). "*Site pre-cast yard layout arrangement through genetic algorithms*." Automation in Construction, 11(1), 35-46.

Csiszár, S. (2007). "*Optimization algorithms (survey and analysis)*." LINDI 2007 - International Symposium on Logistics and Industrial Informatics 2007, Wildau. 185-188.

Elbeltagi, E., Hegazy, T., & Grierson, D. (2005). "*Comparison among five evolutionary-based optimization algorithms*." Advanced Engineering Informatics, 19(1), 43-53.

Elhakeem, Ahmed and Hegazy, Tarek (2010) "*Building asset management with deficiency tracking and integrated life cycle optimisation', Structure and Infrastructure Engineering*," First published on: 19 May 2010 (iFirst) Engineering, Taipei, Taiwan, April 3-5, 2002, v. 1 of 2, pp. 701-706

Evolver Guide to Using (2010), "*A genetic algorithm solver for Microsoft Excel*".

Federal Highway Administration FHWA (1999). "*Asset Management Primer*." A Report prepared by the U.S. Department of Transportation.

Federation of Canadian Municipalities (2007). "*The Coming Collapse of Canada's Infrastructure*." FCM, Ottawa, Ontario, Canada.

Flintsch, G. W., & Chen, C. (2004). "*Soft computing applications in infrastructure management*." Journal of Infrastructure Systems, 10(4), 157-166.

GAMS A User's Guide. GAMS Development Corporation, Washington, DC, USA, 2010

Goldberg, D. (1989), "*Genetic Algorithms in search, optimization, and Machine Learning*", Addison-Wesley publishing company, Inc., New York.

Goldberg, D., and Deb, K. (1991) "*A comparison analysis of selection schemes used in genetic algorithms*", Foundations of Genetic Algorithms, pp.69-93.

Grussing, M. N., Uzarski, D. R., & Marrano, L. I. (2006). "*Optimizing facility component maintenance, repair, and restoration investment strategies using financial ROI metrics and consequence analysis*." Applications of Advanced Technology in Transportation - Proceedings of the Ninth International Conference on Applications of Advanced Technology in Transportation, Chicago, IL. 81-86.

Halfawy, M. M. R., Newton, L. A., & Vanier, D. J. (2006). "*Review of commercial municipal infrastructure asset management systems*." Electronic Journal of Information Technology in Construction, 11, 211-224.

Halfawy, M. R., Dridi, L., & Baker, S. (2008*). "Integrated decision support system for optimal renewal planning of sewer networks."* Journal of Computing in Civil Engineering, 22(6), 360-372.

Haas, R., Hudson, W.R., and Zaniewski, J.P. (1994). "*Modern Pavement Management*." Krieger Publishing Company, Malabar, Fla.

Hegazy, T. (1999). "*Optimization of construction time - cost trade-off analysis using genetic algorithms"*. Canadian Journal of Civil Engineering, 26(6), 685-697.

Hegazy, T., Elbeltagi, E., & El-Behairy, H. (2004). "*Bridge deck management system with integrated life-cycle cost optimization".*

Holland, J. (1975), "*Adaptation in Natural and Artificial Systems*", The University of Michigan Press, Ann Arbor, MI.

Hudson, W.R., Hass, R., and Uddin, W. (1997). "*Infrastructure Management."* McGraw-Hill, New York, USA.

IBM ILOG CPLEX User's Manual. International Business Machines Corporation 1987, 2009

Kyle, B.R.; Lounis, Z.; Vanier, D.J. (2002)."*Multi-objective optimisation of asset maintenance management*." 9th International Conference on Computing in Civil and Building

Leu, S., Yang, C., and Huang, J. (2000), "*Resource Levelling in Construction by Genetic Algorithm-Based Optimization and Its Decision Support System Application*", Automation in Construction, 10 (1), Nov., pp. 27-41.

Li, H., & Love, P. E. D. (2000). "*Genetic search for solving construction site-level unequal-area facility layout problems*." Automation in Construction, 9(2), 217-226.

Marler, R. T., & Arora, J. S. (2004). *"Survey of multi-objective optimization methods for engineering."* Structural and Multidisciplinary Optimization, 26(6), 369-395.

Osman, H. M., Georgy, M. E., & Ibrahim, M. E. (2003). "*A hybrid CAD-based construction site layout planning system using genetic algorithms*." Automation in Construction, 12(6), 749-764.

Shapiro, J. F. "*A survey of lagrangean techniques for discrete optimization"*, 1979.

Stainbach M., Karpis G., & Kumar V. (2000), "*A Comparison of Document Clustering Techniques"* Department of Conmuter Science and Engineering, University of Manitoba, Technical Report #00-034.

Statistics Canada (2009). "*Age of Public Infrastructure: A Provincial Perspective*."

Thanedar, P. B., & Vanderplaats, G. N. (1995). "*Survey of discrete variable optimization for structural design."* Journal of Structural Engineering New York, N.Y., 121(2), 301-305.

Tong, T. K. L., Tam, C. M., & Chan, A. P. C. (2001). *"Genetic algorithm optimization in building portfolio management*." Construction Management and Economics, 19(6), 601-609.

Transportation Association of Canada TAC (1999). "*Primer on Highway Asset Management Systems*." Ottawa, March 1999.

UNIFORMAT II (2005). “*UNIFORMAT II Element Classification for Building Specifications*,” Cost Estimating, and Cost Analysis.”

Uzarski, D. R. (2002). “*Condition assessment Manual for Building Components for use with BUILDER Version 2.1.”* U.S. Army, Engineering Research and Development Centre - Construction Engineering Research Laboratory (ERDC-CERL), Champaign, IL, USA, March 2002.

Vanier, D. J. D. (2001). “*Why industry needs asset management tools.”* Journal of Computing in Civil Engineering, 15(1), 35-43.

W.L. Winston & M Venkataramanan, (2003). *“Introduction to mathematical programming,”* Duxbury Press, ISBN: 0-534-35964

Wolsey, L. (1989). “*Strong formulations for mixed integer programming: A survey. Mathematical Programming*,” 45(1-3), 173-191.