# ABSTRACT

KUMAR, JITENDRA. Asynchronous Hierarchical Parallel Evolutionary Algorithm-Based Framework for Water Distribution Systems Analysis. (Under the direction of G. Mahinthakumar and S. Ranji Ranjithan.)

Water distribution systems (WDSs) are vulnerable to accidental and intentional contamination that can have serious effect on the public health. Accurate characterization of the contaminant source is usually the first step in designing a response strategy for control and containment of the contaminant during an event. Contaminants can spread quickly in the network due to complex hydraulic conditions requiring a procedure for near real-time identification of the source characteristics and design of response strategy. Limitations associated with the quality and quantity of network monitoring data, unknown nature of the contaminant and its interaction with other chemicals in the water, complexity of the network, and inherent uncertainties make contaminant source characterization a difficult and challenging problem to solve. Solution of source characterization problems requires iterative simulations of flow and transport, which are highly computing intensive posing a computational challenge for near real-time solution of the problem.

This research focuses on the development and demonstration of a computational framework for real-time contamination threat management in water distribution systems. Methodologies for contaminant source identification and characterization are developed. An array of contamination scenarios is explored and simulation-optimization-based methodologies are developed to address the source characterization problems. Effects of quality and quantity of available continuous as well as filtered water quality sensor data are studied. The methodologies are tested and evaluated under various conditions of noise and uncertainties in the data. Contamination scenarios involving reactive contaminants are also studied to characterize the source using routinely monitored chlorine levels as surrogate for detection and identification. The uncertainties in the problem due to unknown reaction kinetics of the contaminant in the system are investigated and methodologies for source

characterization under the conditions of such uncertainties and noise in the system are examined. These methods are applied and tested for an array of contamination scenarios in two different example networks. To enable a near real-time solution, a massively parallel computational framework is developed for modern shared/distributed memory architecture-based parallel computers. This includes the development of a new Asynchronous Hierarchical Parallel Evolutionary Algorithm (AHPEA) to solve complex large-scale global optimization problems. Performance and robustness of AHPEA algorithm is tested using a suite of benchmark function optimization problems and its application to address water distribution contaminant threat management problems is demonstrated. A WDS simulation model and the optimization methodologies are fully integrated into the computational framework for real-time analysis of the system. The computational framework is tested for parallel performance and scaling on different state-of-the-art parallel computers.

The simulation-optimization framework developed in this research is successfully able to address the WDS contamination threat management problem by narrowing down the potential source location to a small set of concentrated nodes in the network. This is shown for different conditions of data availability, noise and uncertainties, as well as for reactive as well as non-reactive contaminants. Issue of non-uniqueness in the solution is addressed by identifying all possible solutions. A near real-time solution of the problem is enabled by this framework through efficient use of computational resources. The efficient and scalable framework developed in this dissertation research provides a robust tool that can be applied to solve large-scale complex engineering design problems in general.

Asynchronous Hierarchical Parallel Evolutionary Algorithm-Based Framework for Water Distribution Systems Analysis

by
Jitendra Kumar

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Civil Engineering

Raleigh, North Carolina

2010

APPROVED BY:

-------------------------------------- --------------------------------------

Dr. E. Downey Brill, Jr. Dr. Jeff A. Joines

-------------------------------------- --------------------------------------

Dr. G. Mahinthakumar
(Co-chair of Advisory Committee)

Dr. S. Ranji Ranjithan
(Co-chair of Advisory Committee)

# DEDICATION

*Dedicated to*

*my parents, sisters and brother*

*for their never ending love, blessings and support*

# BIOGRAPHY

Jitendra Kumar was born on March 15, 1981 in Munger, India. He graduated from Muzaffarpur Institute of Technology, Muzaffarpur, Bihar in 2004 with a Bachelor of Science in Civil Engineering. In the fall of 2004, he joined Indian Institute of Technology (IIT) Kanpur for Master of Technology in Civil Engineering. He worked under the guidance of Dr. Ashu Jain in Hydraulics and Water Resources Engineering group. In 2005 he was awarded Indo-German Academic Exchange (DAAD) scholarship to pursue his Masters thesis research at Technische Universität Darmstadt, Germany under the guidance of Dr. Manfred Ostrowski. His Masters research focused on conceptual and data-driven modeling of hydrologic processes. Jitendra joined North Carolina State University for PhD in Civil Engineering in fall of 2006. He worked under the guidance of Dr. G. Mahinthakumar and Dr. Ranji Ranjithan for his doctoral dissertation research. Upon completion of PhD, he will be joining as a Postdoctoral Research Associate in Computational Earth Sciences Group at Oak Ridge National Laboratory, Oak Ridge, TN.

# ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my advisors Dr. G. Mahinthakumar and Dr. Ranji Ranjithan for their guidance, support and encouragement throughout this work. It was a privilege to work with and learn from two great teachers, the skills for problem solving and critical thinking. In the weekly breakfast meetings with Dr. Ranjithan over the last few years, I have learnt a lot more than just research, which would help me a long way in my career. I would like to thank Dr. Downey Brill for his constant support and encouragement. He is undoubtedly the best teacher I ever had. I would also like to thank my committee member Dr. Jeff Joines for his invaluable suggestions and time. I have had an opportunity to interact, learn and share ideas with many other professors in the department and I am very thankful to all of them.

I would also like to acknowledge my officemates and colleagues for the thoughtful discussions, research collaborations, feedback and support. Special thanks to Sarat Sreepathi and Li Liu for their support and collaboration as a team on NSF DDDAS project. I would like to thank the wonderful group of friends in Mann 431 and the department including, Naresh Devineni, Vamsi Sripathi, Jason Simon, Danielle Touma, Brandon Graver, Silesh Amujuri, Ali Mehran and many others for the great fun filled days and amazing 'Fridays' at NC State. I also want to thank my friends Salil Wadhavkar and Pradeep Sharma for their help and support.

I thank my parents, sisters and brother for their love, affection, support and encouragement. They have always been a source of strength and inspiration to me.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1. Introduction

Water distribution systems (WDS) consist of large network of pipes and utilities to enable the supply of fresh drinking water supply to every household. Because of the large spatial extent of these systems, it is not possible to ensure the physical security of the system. WDS are highly interconnected in nature with numerous possible access points in the system making it vulnerable to accidental or intentional contamination. WDS networks are designed purposely for a high degree of redundancy for mixing at the nodes of the network so that the disinfectant (i.e., free chlorine) reaches all parts of the network. Thus, any external contaminant can spread quickly through mixing and transport and might be consumed by consumers, possibly leading to severe public health emergency. During a contamination event, large number of consumers can potentially be affected unless control measures are taken immediately. Accurate knowledge of the possible contamination source characteristics is needed for the design of a control and response strategy in the event of a contamination.

Water distribution system contaminant source characterization problem is an ill-posed problem because of the complexity of the network structure, complex hydraulics, and inadequate quality and quantity of observation data. Observation data from the water quality sensors installed in the network can be used for characterization of the source. However, there are limitations over the quantity and quality of the available observations from the water quality monitoring sensors in any network. Characteristics of the contaminant injected in the network during a contamination event can be non-reactive or reactive chemical, biological or radioactive which remains unknown during an event and may go undetected by the available water quality monitoring sensor network. Also, during a contamination event contaminants can be injected at more than one location. A source identification methodology to identify all possible contaminant sources under a range of different conditions is required for WDS. To minimize the impact of a WDS contamination on public and to restore the system to normal in shortest possible time, a near real-time solution of the problem is required.

The objective of this research is to develop new algorithms, methodologies and computational tools to address the contaminant source identification in a WDS for a wide range of possible contamination scenarios discussed above.

Source characterization problem can be posed as inverse problem and can be solved using a simulation-optimization approach. Optimization model is used to identify the characteristics of the contaminant source and simulation model is used to predict the contaminant concentration distribution in the network for a given network configuration and demand condition. Optimization problem for source identification involve several discrete and continuous variables. Also, non-uniqueness in the solution is a major issue with more than one set of source characteristics possible to match the observations at the sensors equally well. Determining a single solution might lead to misleading assessment of the contamination event. Thus, the optimization problem needs to be solved not just for a single best solution but all possible solutions which fit the observations. An evolutionary algorithm based simulation-optimization methodology has been developed in this research for the identification of contaminant source characteristics in WDS. The multiple-population optimization algorithm employs a method for generating alternative solutions to address the issue of non-uniqueness in source characterization problems.

The source identification problem was studied using the data from various kinds of sensors providing continuous and accurate observations to filtered threshold based binary signals. Effect of quality and quantity of water quality sensor data on the accuracy and non-uniqueness of source characterization problem was studied in detail. Effect of observation data noise and uncertainties were also studied in detail.

Methodology was developed for the identification of source characteristics for the reactive chemical, biological or radioactive contaminants.  It is not possible to monitor a system for all of possible reactive contaminant using contaminant specific sensor. A methodology was developed to use routinely recorded chlorine measurements as surrogate to detect the presence of a reactive contaminant in the system and to identify the source characteristics. The issues of uncertainty about the reaction kinetics and parameters and noise in the observation data were studied.

To minimize the impact of a contamination and to develop a response strategy, real time solution of the source identification is needed. However, simulation of a water distribution system is computationally intensive requiring enormous computation for the solution of the inverse problem. A new parallel evolutionary algorithm was developed in this research to enable the solution of large scale optimization problems in near real-time. An asynchronous hierarchical parallel evolutionary algorithm was developed to address the class of large scale global optimization problems in general and was successfully applied for the near real-time solution of WDS source identification problem.

All the algorithms and methods were incorporated into an integrated parallel simulation-optimization framework and were tested on a range of parallel platforms from small clusters to massively parallel supercomputers. An extensive series of parallel performance studies were carried out to optimize the computational performance of the framework.

## 1.1 Organization of the dissertation

The research conducted in this study has been organized in the dissertation as follow.

Chapter 2 describes the development of contaminant source identification method for WDS using imperfect binary sensors. Analysis of the effect of quality and quantity of water quality observation data on accuracy and non-uniqueness of source identification problem has also been reported.

Chapter 3 presents an analysis of effect of data noise and uncertainty in imperfect binary sensors on the accuracy and non-uniqueness of source identification problem.

Chapter 4 describes the methodology for source identification in WDS involving reactive contaminants.

Chapter 5 presents a new Asynchronous Hierarchical Parallel Evolutionary Algorithm (AHPEA) framework for large scale global optimization problems.

Chapter 6 describes the design of massively parallel computational framework for the AHPEA and results of the parallel performance analysis conducted on several different parallel platforms.

Chapter 7 describes the development of a computational cyberinfrastructure for contaminant threat management in WDS. Computational framework for coupling of simulation and optimization model has been presented and computational performance results have been reported.

Chapter 8 discusses the findings of the research and future work planned.

# CHAPTER 2. Contaminant source characterization in water distribution systems using filtered sensor data

## 2.1 Introduction

Water distribution systems are vital for supplying safe drinking water to the public, and they are vulnerable to contamination that can be introduced into the system either deliberately or accidentally. The fate and transport of a contaminant and the extent of the contaminant spread through the network depend on the characteristics of the network topology and the resulting hydraulic conditions of the network.

Networks of sensors can be used as an early warning system to detect sudden deterioration in water quality. They are meant to supplement conventional water quality monitoring by quickly providing timely information on unusual threats to a water supply system. The goal of an early warning system is to reliably identify a contamination event in the distribution systems in time to allow an effective targeted response that reduces or avoids entirely the adverse effects of contamination on the system. The contaminant source characterization (i.e., identifying the contaminant source location and injection pattern) problem has been approached as an inverse problem by many researchers (e.g., Waanders et al. 2003; Laird et al. 2005, 2006; Guan et al. 2006). These approaches attempt to reconstruct a contamination injection event by matching the estimated contaminant concentration profiles with the observations at contaminant sensor locations.

Most of the investigations reported in the literature on contaminant source characterization assume the availability of ideal contaminant-specific sensors to provide concentration measurements. It may be more economical to use instead sensors that provide a binary signal to indicate the presence or absence of contamination according to a specified contaminant detection threshold concentration. The sensitivity of such a binary sensor is defined here to be determined by the detection threshold level; i.e., a more sensitive (therefore more expensive) binary sensor will have a lower detection threshold. A contamination event may even go undetected if the installed sensors have a low sensitivity. Thus, in addition to lowering of information quality due to inherent data filtering in a binary sensor, the sensitivity of the binary sensor also affects the quality of the information used to estimate the source characteristics. Such sensor data quality issues along with errors and uncertainties in model and data associated with a water distribution system results in potential ambiguities in the predicted contaminant source characteristics. One implication is that there may be non-unique solutions to the inverse problem such that a set of different source characteristics match the observations at the sensors. This limits the ability to precisely identify the true source location and the contaminant injection pattern.

The work presented in this paper describes a methodology for characterizing the contaminant source in a particular event using detection signals from a set of binary sensors. This work also includes an investigation of the effect of the quality (i.e., sensitivity of the sensors) and quantity (i.e., number of sensors in the network) of the sensor data on the accuracy in identifying the source characteristics and on resolving the degree of non-uniqueness. The methodology is applied and tested on realistic water distribution networks to assess its applicability and robustness.

## 2.2 Data filtering effects in binary sensors

This section illustrates the effects of binary sensors on filtering continuous concentration signals that would have been observed at an ideal sensor. Figure 2.1.a shows the continuous concentration profile at a sensor. The filtered signals (shown by the vertical bars) at different levels of sensor sensitivity are shown in Figure 2.1.b-Figure 2.1.d that correspond to threshold values of 0.01 mg/l, 0.1 mg/l, and 0.2 mg/l, respectively. As evident in the graphs, the duration over which the contaminant is detected decreases as the sensor sensitivity decreases. Another effect of data filtering in a binary sensor is the potential lack of discrimination of the magnitude of the contaminant concentration at the sensor. For example,

Figure 2.2.a shows the continuous concentration profile with relatively low concentration values, and Figure 2.2.b-Figure 2.2.d show the filtered binary signals for sensors with detection threshold of 0.01 mg/l, 0.1 mg/l, and 0.2 mg/l, respectively. Considering Figure 2.1.b and Figure 2.2.b, there is no discernible difference when comparing the binary signal profiles for the 0.01 mg/l threshold even though the real concentration profiles are different and may have resulted from different contaminant injection patterns. Such effects of data filtering at binary sensors can potentially contribute to non-uniqueness and lack of precision in the source characteristics being identified.

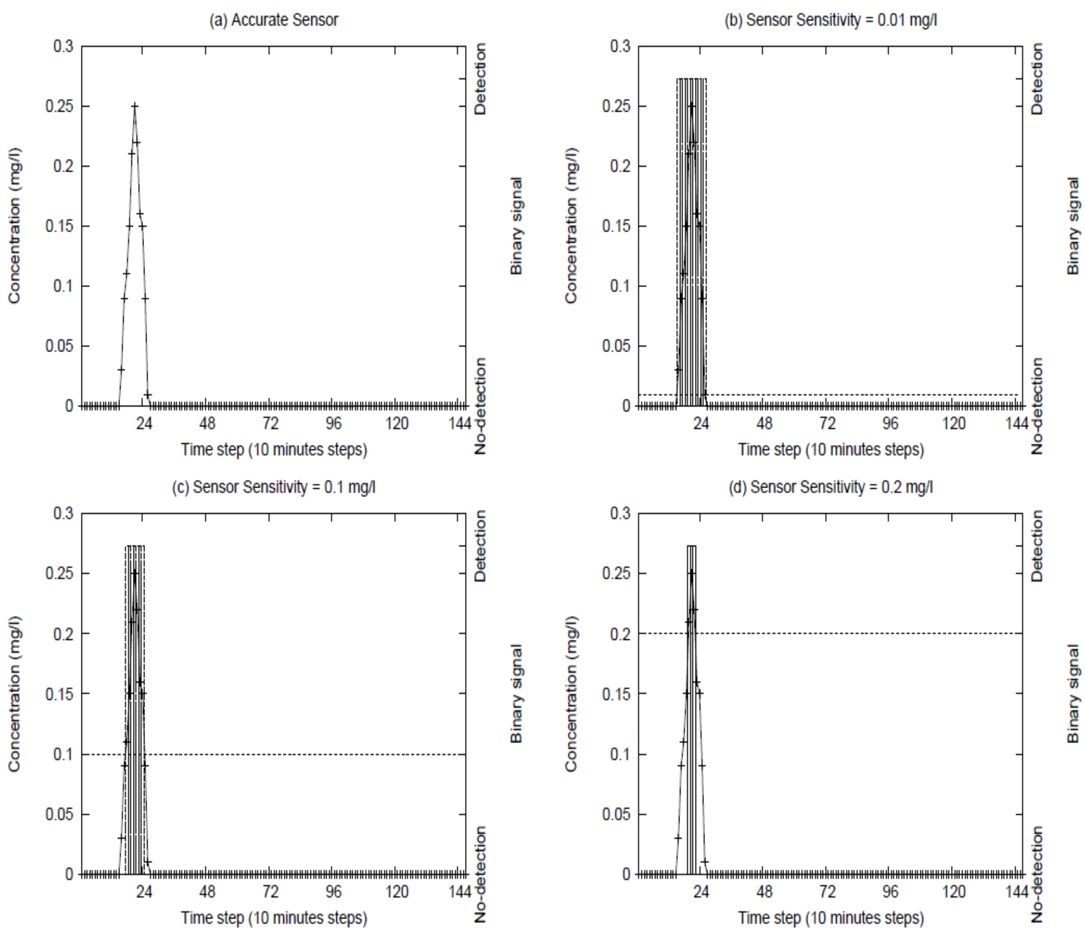



**Figure 2.1 Effect of sensor threshold on the filtering of data to produce a binary signal: Example 1**

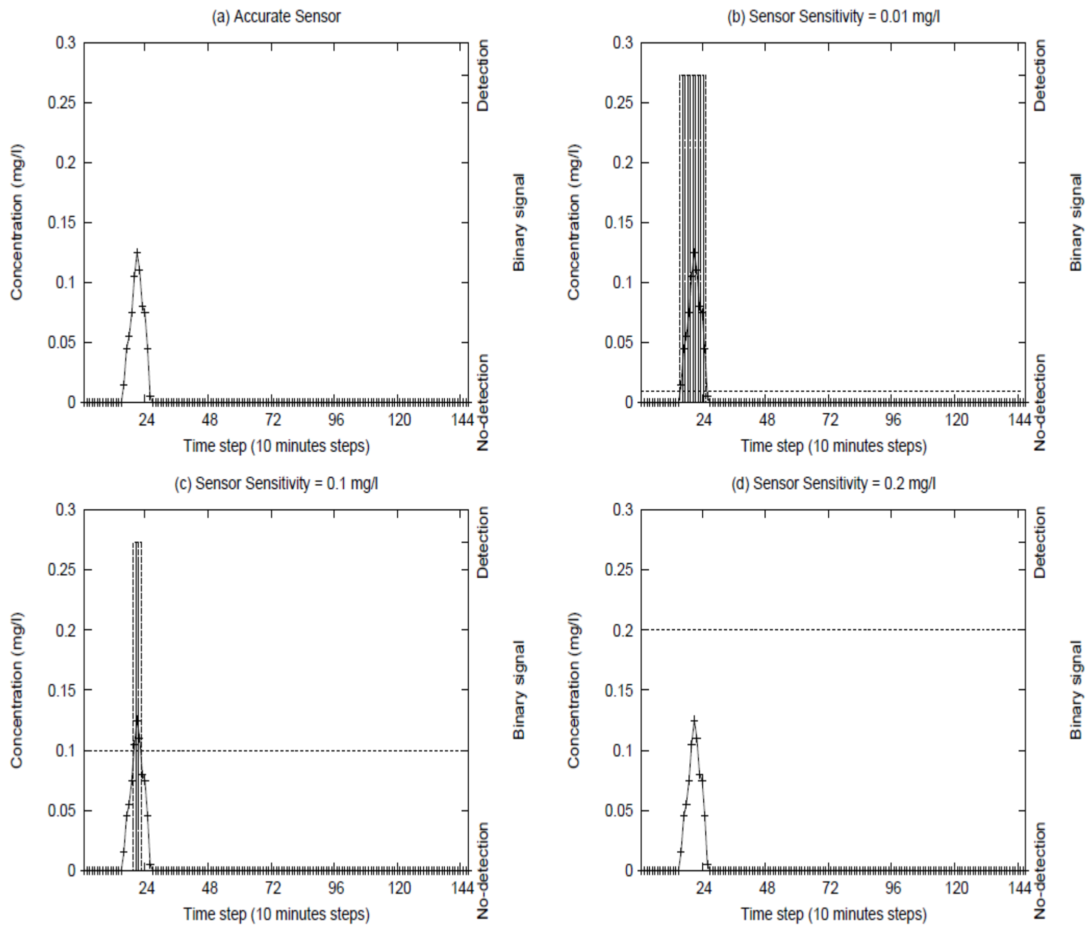



**Figure 2.2 Effect of sensor threshold on the filtering of data to produce a binary signal: Example 2**

## 2.3 Characterization of the contaminant source using binary sensor data

This paper focuses on developing and testing an approach that uses the filtered binary sensor data to identify the location of a contaminant source and the injection pattern. The contaminant source characterization problem is posed as an inverse problem where the source characteristics are treated as unknowns. For each potential set of source characteristics (i.e., the location and injection pattern), the binary sensor signals are simulated and compared to the observations. The prediction error is then used as a measure of goodness of fit to direct the search for a set of source characteristics that best fits the observations. The resulting search problem is solved using a coupled simulation-optimization approach.

The following characteristics of the contaminant source are decision variables for the optimization problem:

(a) Location ($L$) of the contaminant source in the network. Location of the source is assumed to be at one or more of the nodes in the WDS network.

(b) Start time of the contaminant injection into the system ($T_0$). Start time is measured as the time elapsed from the start time of the simulation.

(c) The pattern of contaminant injection ($C_0$) into the system measured in terms of milligrams of contaminant injected per minute into the system. Injection rate is assumed to be constant during each water quality time step of the simulation. No assumptions are made regarding the magnitude, pattern or duration of the contaminant injection.

The optimization model for this inverse problem is defined below (Equations 1-3) assuming that the contamination scenario involves only a single source; however, it could be extended to scenarios involving simultaneous contamination at multiple sources.

$$Minimize \quad E = \sum_{i=1}^{N_S} \sum_{t=1}^{T_S} \left| BS_{i,t}^{s} - BS_{i,t}^{a} \right| \tag{1}$$

$$subject\ to \quad BS_{i,t}^{s} = \begin{cases} 0, & if\ C_{i,t}^{s}(L, T_0, C_0) < C_{th} \\ 1, & if\ C_{i,t}^{s}(L, T_0, C_0) \geq C_{th} \end{cases} \tag{2}$$

$$where \quad BS_{i,t}^{a} = \begin{cases} 0, & if\ C_{i,t}^{a} < C_{th} \\ 1, & if\ C_{i,t}^{a} \geq C_{th} \end{cases} \tag{3}$$

The objective function (Equation 1) is a measurement of error represented as the total number of mishits at the sensors. A simulated expected signal for the given values of $L$, $T_0$, and $C_0$, will be considered a mishit if it does not match the actual signal. Specifically, there is a mishit when there is a simulated detection signal when the actual signal was of no-detection or vice versa. The optimization problem is to obtain the values of $L$, $T_0$, and $C_0$ that minimize the total number of mishits between the simulated and actual sensor signals. Equation 2 filters the simulated concentration levels for a given set of source characteristics at the sensors into detection/no-detection binary signals depending on the sensitivity of the sensor for the particular contaminant. $BS_{i,t}^{s}$ and $BS_{i,t}^{a}$ are respectively, the simulated binary signal and the actual binary signal for given source characteristics at sensor $i$ at any time $t$; $N_s$ is the number of sensors in the network; $T_S$ is the total time for which contaminant mixing and transport in the network is being simulated. The simulated concentrations of the contaminant are given by $C_{i,t}^{s}(L, T_0, C_0)$ for a given solution as a function of location ($L$), time of measurement ($t$), start time of the contamination injection ($T_0$), and the contaminant injection pattern ($C_0$), respectively.

$C_{th}$ is a constant defining the threshold concentration level for determining detection and generating a binary signal by the sensors. The constants $C_{i,t}^{a}$ are the actual measured concentrations of the contaminant being monitored.

## 2.4 Methodology

The solution to the optimization model represented by Equations 1-3 would ideally identify the true contaminant source location and the injection pattern. Because of data limitations, however, the solution to Equations 1-3 is likely to yield non-unique source characteristics to fit the observations equally well. Specifically, there may be data quality limitation resulting from, for example, measurement errors and filtering effects, as well as quantity limitations resulting from, for example, discrete measurements and limited number of observations.. Therefore, the solution approach should be able to conduct a global search and be able to identify the set of potential source characteristics that provide equally good fit to the given observations. The niched co-evolution strategies (NCES) search procedure (Zechman and Ranjithan, 2004, 2007) was used to solve the above model. Also, given the possibility of error in modeling of the physical distribution network, it is also important to identify near optimal model solutions that might be the solution to the real problem. The NCES procedure also provides this capability.

NCES conducts a global search using evolution strategies (ES), a generalized heuristic optimization approach. A ES works in a continuous space and has the capability to self adapt its major algorithmic parameters such as selection and mutation. ES starts with a randomly initialized population of $\mu$ individuals. A probabilistic mutation operator is applied to produce $\lambda$ new solutions each generation. The next set of individuals is selected from either a combined set of parent and offspring solutions (denoted as ($\mu + \lambda$) selection) or from the set of offspring alone (denoted as ($\mu, \lambda$) selection).

The standard ES is extended in NCES to generate a set of alternative solutions that performs equally well in terms of fitness, i.e., the objective function being optimized. The search for a set of solutions in NCES is based on the concept of cooperative co-evolution to evolve simultaneously a set of subpopulations for exploring the decision space. Each subpopulation is driven to a region in the solution space that has good fitness and is far away from other solutions with good fitness. This helps to identify different contamination source locations and mass injection patterns, if any, that can produce the same or similar observations at the sensors (i.e., within comparable prediction error values), leading to insights about potential non-uniqueness or imprecision about the estimate of the true source characteristics. The simple Euclidean distance between the location of an individual and the location of the centroids of all subpopulations are calculated and the distance from the closest centroid is used as the distance metric. For the contaminant source identification problem, identifying sources at different possible locations was identified as a key problem, and thus for all reported scenarios the distance measures in terms of location are used during the selection step. However, the distance measure can easily be extended to include time and contaminant injection pattern also.

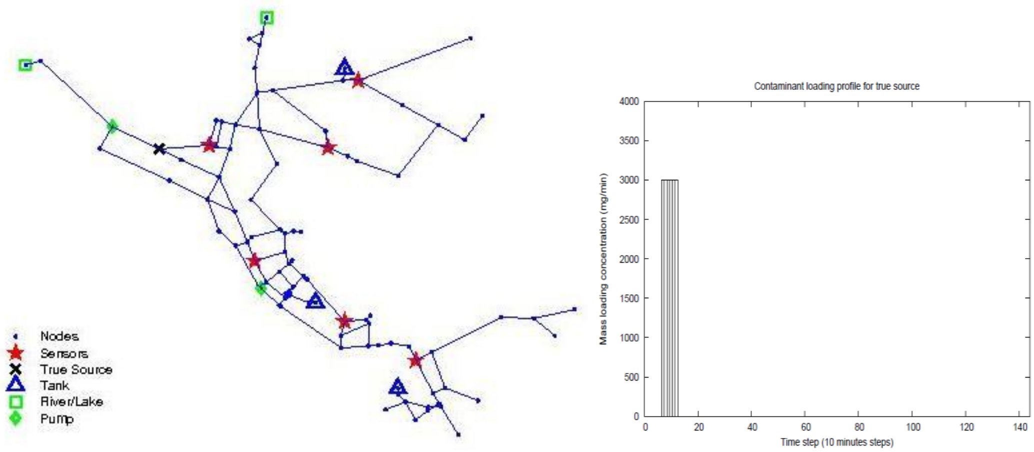
While one subpopulation focuses its search on finding the optimal solution based on the prediction error (*E* in Eqn. 1), the other subpopulations search for alternatives based on the current best prediction error value as well as the degree of difference among the alternative solutions. In the results presented in this paper, every solution that is within one miss-hit from the best prediction error value (i.e., $\Delta E = 1$) is considered as a viable alternative solution.

NCES was applied and tested on two water distribution networks to study: 1) the effect of the sensitivity of binary sensors (quality of data) on the accuracy and non-uniqueness of the solutions to such source identification problems; 2) the effect of the quantity of available data (i.e., number of installed sensors) on the solutions to source identification problems; and 3) the robustness of the method when applied to a set of different contamination scenarios and networks.

## 2.5 Illustrative case studies

### 2.5.1 Networks analyzed

Two networks were modeled and studied to demonstrate the proposed simulation-optimization method. The first case Network 1 is a test network included with the standard EPANET distribution. Network 1 (Figure 2.3.a) consists of 97 nodes, including two sources, three tanks and 117 pipes. The network is assumed to have a maximum of six water quality sensors. EPANET was used for simulating the hydraulic and contaminant transport in the network for a period of 24 hours. Hydraulic simulations were carried out after every hour and were considered to be at steady-state during each hour. Water quality transport simulations were carried out at 10-minute intervals and the measurements were recorded every 10 minutes.




(a) Network (b) Injection pattern

**Figure 2.3 Layout of the Network 1 water distribution network**

The second test network considered is for the virtual city of Micropolis (Brumbelow et. al. 2007), which represents a realistic municipality capturing the evolution of the infrastructure over many decades. This is a publicly available water distribution system dataset for research studies without compromising public security (Figure 2.4). Micropolis mimics a small city of approximately 5,000 people in a typically rural region that is served by a single elevated storage tank with a capacity of 440,000 gallons (1,670,000 liters). The water distribution system consists of 1236 nodes, 575 mains (for a total length of 10.1 miles), 486 service and hydrant connections (total length of 7.1 miles), 197 valves, 28 hydrants, 8 pumps, 2 reservoirs, and 1 tank. The 458 demand nodes are composed of 434 residential, 15 industrial and 9 commercial/institutional users with diurnal demand patterns that are defined at hourly time intervals.

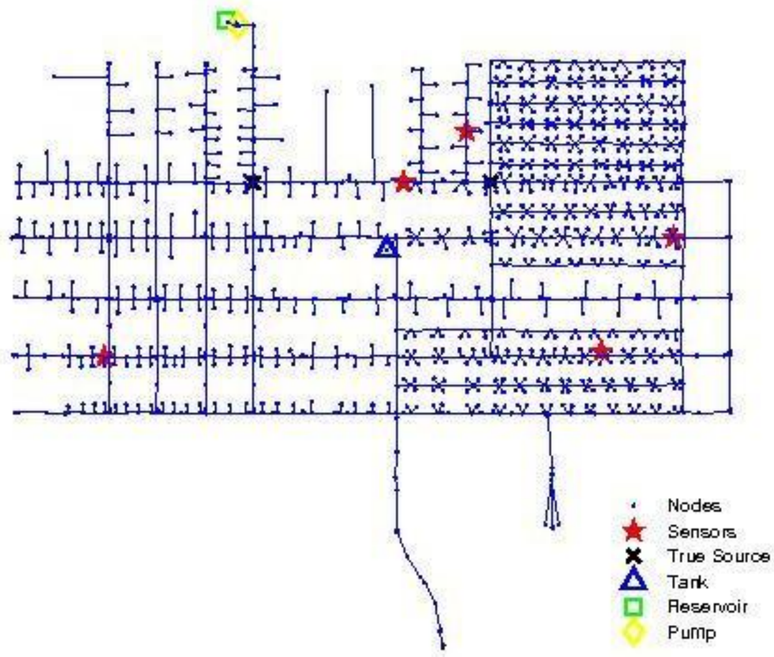



**Figure 2.4 Layout of the Micropolis water distribution network**

### 2.5.2 Scenarios analyzed

Network 1 was used to construct several hypothetical contamination scenarios (Table 2.1) designed to investigate various aspects of the source characterization problem that is to be solved based on observations from a set of threshold-based binary sensors.

*Scenario 1* was designed to serve as a benchmark for an ideal scenario in which continuous and accurate concentration measurements of the contaminants are assumed to be available. The sum of the squared error between the actual and simulated concentrations at the sensors was used as the metric to be minimized in the optimization problem. Equation 4 represents the mathematical formulation of the objective function for this scenario.

$$Minimize\ E = \sum_{i=1}^{N_S} \sum_{t=1}^{T_s} \left( C^{s}_{i,t}(L, T_0, C_0) - C^{a}_{i,t} \right)^2 \quad mg^2/\min.^2 \tag{4}$$

In *Scenarios 2, 3 and 4*, the sensitivity of the binary sensor was set at 0.2 mg/l, 0.1 mg/l and 0.01 mg/l, respectively, to examine the impact of increasing sensor sensitivity on the accuracy and degree of non-uniqueness of the solutions to the source characterization problem. The effect of quantity of available data was studied in *Scenarios 5, 6 and 7* using observation data from 6 sensors, 3 sensors and 1 sensor, respectively. *Scenarios 8-11* were designed to test the methodology for a range of different contaminant injection instances and to assess the robustness and general applicability of the method. While a continuous and uniform contaminant injection (i.e., a constant mass injection pattern active for one hour) scenario was used for *Scenarios* 2-7, the ability to solve the problem for non-uniform and discontinuous contaminant injection scenarios were analyzed in *Scenarios 8 and 9*, respectively. To study the performance of the procedure when the contamination source location varies, *Scenarios 10 and 11* were analyzed with sources located in different regions of Network 1. For *Scenarios 1-11*, the contamination injection was assumed to start at 1.00 AM (i.e., at time step 7) (Figure 2.3.b) and was active for one hour period. To evaluate the applicability of the proposed procedure for other networks, *Scenarios 13 and 14* (described later) were analyzed using a contamination event in the Micropolis network.

**Table 2.1 Details of the scenarios analyzed**

| Scenario | True source location<br>Node No. (Network) | Sensor type | Sensor sensitivity | No. of sensors | Contaminant injection profile |
|---|---|---|---|---|---|
| 1 | 12 (Network 1) | Ideal | - | 6 | Continuous (1 hr.) |
| 2 | 12 (Network 1) | Binary | 0.2 mg/l | 6 | Continuous (1 hr.) |
| 3 | 12 (Network 1) | Binary | 0.1 mg/l | 6 | Continuous (1 hr.) |
| 4 | 12 (Network 1) | Binary | 0.01 mg/l | 6 | Continuous (1 hr.) |
| 5 | 12 (Network 1) | Binary | 0.1 mg/l | 6 | Continuous (1 hr.) |
| 6 | 12 (Network 1) | Binary | 0.1 mg/l | 3 | Continuous (1 hr.) |
| 7 | 12 (Network 1) | Binary | 0.1 mg/l | 1 | Continuous (1 hr.) |
| 8 | 12 (Network 1) | Binary | 0.1 mg/l | 6 | Intermittent |
| 9 | 12 (Network 1) | Binary | 0.1 mg/l | 6 | Intermittent |
| 10 | 20 (Network 1) | Binary | 0.1 mg/l | 6 | Continuous (1 hr.) |
| 11 | 54 (Network 1) | Binary | 0.1 mg/l | 6 | Continuous (1 hr.) |
| 12 | 787 (Micropolis) | Binary | 0.1 mg/l | 5 | Continuous (1 hr.) |
| 13 | 653 (Micropolis) | Binary | 0.1 mg/l | 5 | Continuous (1 hr.) |

For a potential solution (i.e., the source characteristics), the EPANET model was run to predict the concentration values at the sensors. These concentration values were then filtered and converted to a simulated binary signal indicating the presence or absence of contamination at each sensor. The actual binary sensor signal was generated for the true contamination event using EPANET simulations; however, in real applications, these observations would be the sensor readings obtained from measurements. When the simulated signal and the actual binary signal at each sensor and each observation time point do not match, the mishit count was incremented by one.

In all scenarios, the search was conducted to identify up to five alternative potential solutions by including five subpopulations in NCES. Subpopulation 1 was designed to search for the solution with the minimum number of mishits, while the other subpopulations were set to search for alternative solutions that are maximally different from each other and have no more than one mishit more compared to the lowest number of mishits (i.e., the minimum objective function value for subpopulation 1). The NCES algorithmic parameters used in this study are shown in Table 2.2.

**Table 2.2 Algorithmic parameter settings for the NCES algorithm**

| Parameter | Setting |
|---|---|
| Population size (μ) | 200 |
| Individuals created by mutation (λ) | 200 |
| Number of sub-populations | 5 |
| Mutation strength for self adaptive mutation | 0.5 |
| Number of generations | 100 |
| Feasibility criteria | Objective function value of best individual of first subpopulation + 1 |
| Number of random trials | 50 |

## 2.6 Results

For each scenario, 50 random trials were carried out to assess the robustness of the solution approach. The summary statistics as well as a detailed description of a representative solution are presented, and the following are also reported: 1) mean and standard deviation of the prediction error in subpopulation 1 (which is tasked to search for minimum error solution); 2) mean and standard deviation of the prediction error for the alternative solutions; 3) mean and standard deviation of the number of feasible alternative (non-unique) solutions identified; and 4) locations of all the feasible non-unique solutions identified in the 50 random trials.

### 2.6.1 Ideal water quality sensor measurements (based on Scenario 1)

The results for Scenario 1 described below are based on the sets of five alternative solutions obtained for each of the 50 trials. In every trial, at least one of the five alternative solutions identified the true source location (i.e., Node 12). The average prediction error (sum of squared error) was 0.229 $mg^2/min^2$ and the standard deviation was 0.062 $mg^2/min^2$. The source location and the contaminant injection start time for the set of feasible alternative solutions obtained for a typical trial are presented in Table 2.3, and the injection profiles are shown in Figure 2.5. On average, 0.1 alternative solutions (with a standard deviation of 0.364) were identified over all 50 trials. The average prediction error for the alternative solutions was 0.349 $mg^2/min^2$ with a standard deviation of 0.120 $mg^2/min^2$. Solutions with a prediction error that is no more than 110% of the best prediction error were accepted as feasible alternative solutions. In the 50 trials, feasible solutions were identified at locations 10, 12 and 86 with an average error of 0.476 $mg^2/min^2$, 0.228 $mg^2/min^2$ and 0.309 $mg^2/min^2$, respectively. Even in the case of ideal sensors with error-free continuous measurements, the number of sensors (six in this scenario) and their placements in the network are insufficient to precisely identify the source characteristics, resulting in non-unique solutions at three locations in the network(with 0.1 alternatives on average). The two alternative locations (Nodes 10 and 86) (Figure 2.6.a) identified in the 50 trials are adjacent to the true source location in the network.

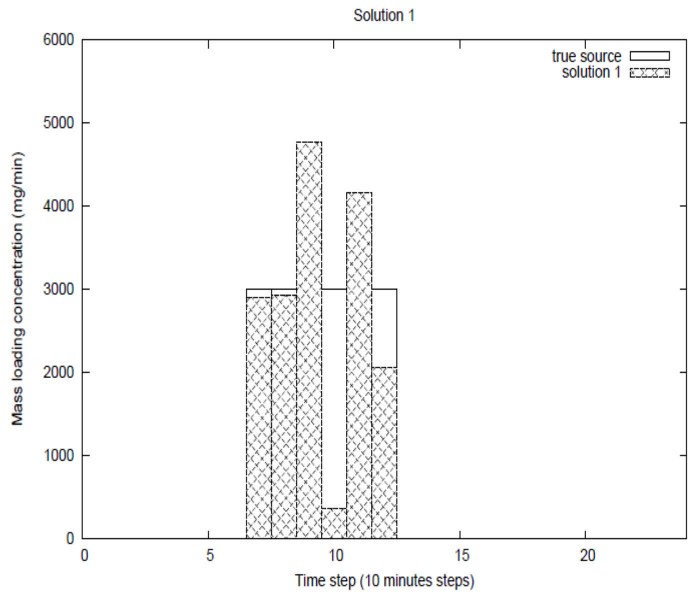



**Figure 2.5 Contamination injection profiles for a typical solution for Scenario 1 and the true source**

**Table 2.3 The source location and injection start time of the non-unique solutions obtained for a typical trial for Scenarios 1-4**

| *Scenario* | *Solutions (Location, start time)* | *No. of mishits* |
|---|---|---|
| 1 | (12, 7) | 0.195290[1] |
| 2 | (12, 7) | 0 |
| | (12, 7) | 1 |
| | (86, 10) | 0 |
| 3 | (12, 8) | 0 |
| | (86, 9) | 1 |
| | (12, 8) | 1 |
| 4 | (12, 7) | 0 |
| | (12, 6) | 1 |

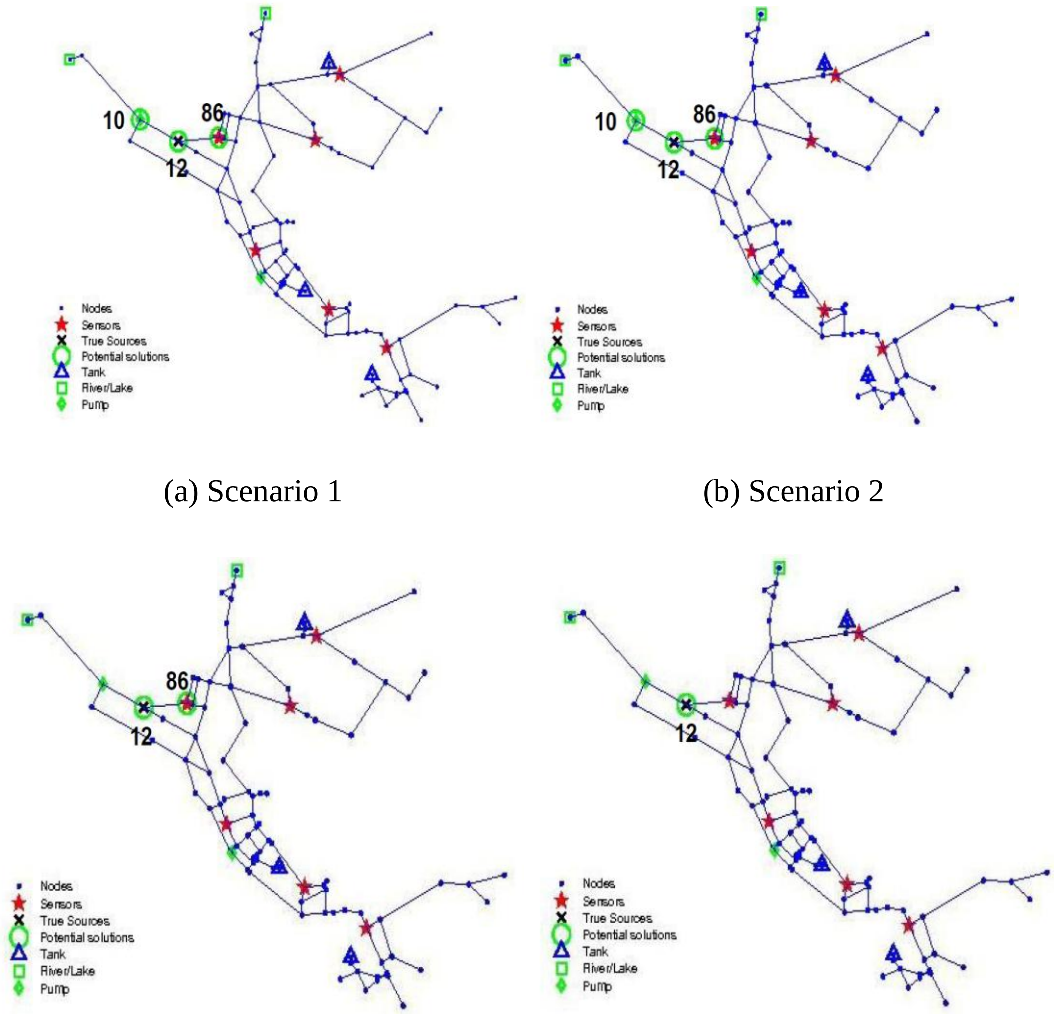
[1] sum of squared error $mg^2/min^2$ since binary sensors were not used in this scenario

### 2.6.2 Effect of quality of data (based on Scenarios 2, 3, and 4)

Scenarios 2-4 were run with different values of sensor sensitivity to study the effect of data quality (due to different degrees of data filtering in the binary sensor) on level of non-uniqueness. In each case, 50 trials were run to assess the number of alternative solutions and the relative locations of the source nodes associated with the non-unique solutions.

#### 2.6.2.1 Scenario 2: Sensor sensitivity = 0.2 mg/l

In each of the 50 trials, at least one solution among the set of non-unique solutions matched the observation data perfectly. Table 2.3 shows a representative result for this scenario, and the set of predicted source nodes are shown in Figure 2.6.b. The location and start time of the best solution were identical to those of the true source. The source location and start time of another non-unique solution was the same as those of the true source, but had a different contaminant injection pattern, resulting in one mishit in the observation data. The source location (Node 86) of the other non-unique solution is adjacent to the true source location, but with a perfect prediction of the observation data. Based on 50 trials, on average 2.72 alternative solutions were identified (with a standard deviation of 1.030). Alternative solutions were obtained with an average prediction error of 0.837 mishits and a standard deviation of 0.258 mishits.




(a) Scenario 1 (b) Scenario 2

(c) Scenario 3 (d) Scenario 4

**Figure 2.6 Potential contamination source locations identified by the set of non-unique solutions for Scenario 1-4**

### 2.6.2.2 Scenario 3: Sensor sensitivity = 0.1 mg/l

The key aspects of the solutions for a representative trial for this scenario with improved sensor sensitivity are the shown in Table 2.3. A solution identified at true source location (Node 12) had a start time of 8 (instead of true start time of 7) but perfectly matched the observation data. The second solution was identified at an adjacent location (Node 86) and matched the observation data with a single mishit. Third solution that was identified at the true source location matched the observation data with a single mishit. A solution perfectly matching the observed data was always identified among the set of feasible solutions obtained for each of the 50 trials. Locations of these non-unique solutions are shown in Figure 2.1.c. Based on the 50 trials, on average 2.12 alternative solutions (with a standard deviation of 0.961) were identified with average prediction error of 0.887 mishits and standard deviation 0.238 mishits.

#### 2.6.2.3 Scenario 4: Sensor sensitivity = 0.01 mg/l

Table 2.3 shows a summary of the key characteristics of a representative solution obtained for this scenario that had the most sensor sensitivity. The best solution in the first subpopulation perfectly matched the source location and start time of the true source. Only one other non-unique solution was identified which had the same source location as the true source but with a start time of 6 (instead of 7). In the 50 trials, the best solution was found with an average prediction error of 0.12 mishits and a standard deviation of 0.385 mishits. On average 0.56 alternative solutions (standard deviation 0.884) were obtained with an average prediction error of 0.358 mishits (standard deviation 0.687 mishits). In all solutions the source location was identified to be the same as the true source location Node 12 (Figure 2.6.d), indicating a reduced level of non-uniqueness due to the increased sensitivity of the sensor.

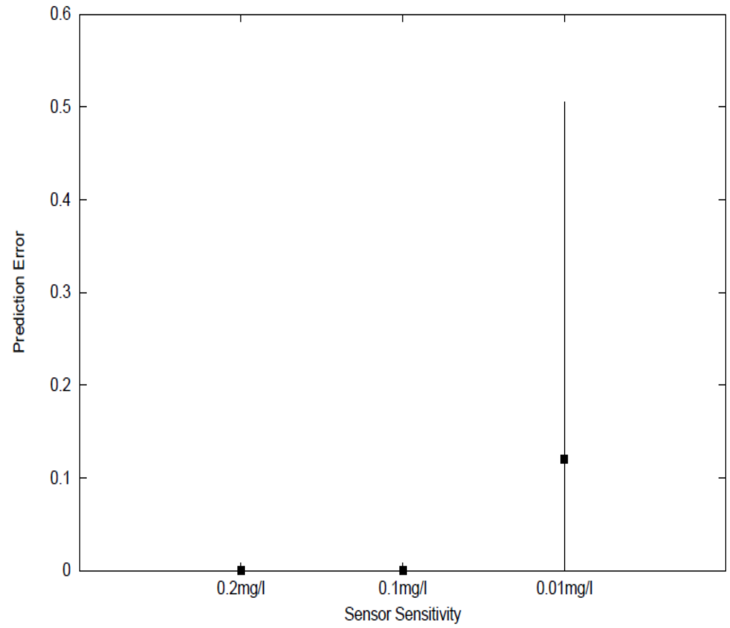



**Figure 2.7 For Scenarios 2, 3 and 4, the average (horizontal bar) and one-standard deviation (vertical bar) of prediction error of the best solutions obtained for the 50 trials**

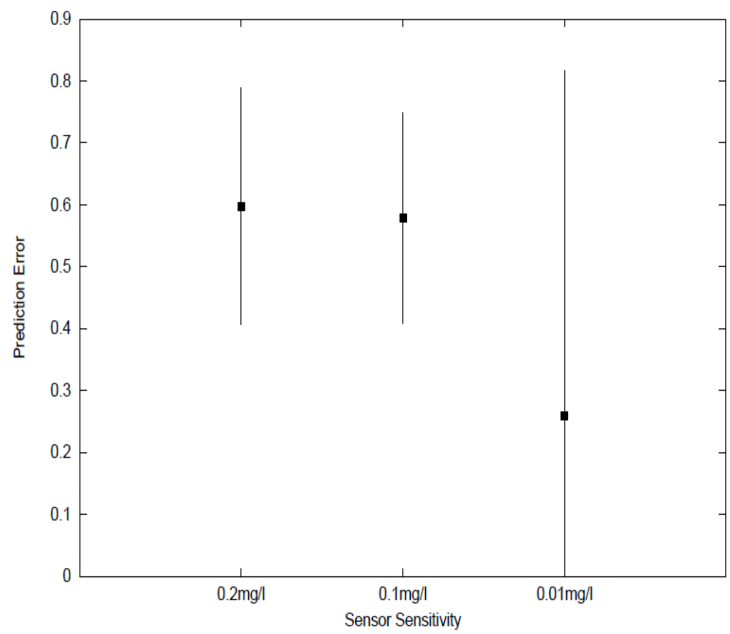



**Figure 2.8 For Scenarios 2, 3 and 4, the average (horizontal bar) and one-standard deviation (vertical bar) of prediction error of the non-unique solutions obtained for the 50 trials**

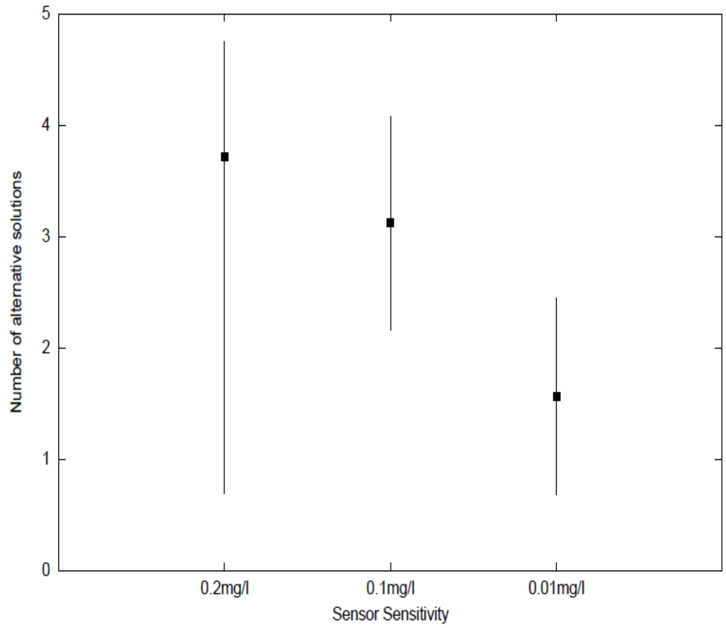



**Figure 2.9 For Scenarios 2, 3 and 4, the average (horizontal bar) and one-standard deviation (vertical bar) of the number of non-unique solutions obtained for the 50 trials**

The Scenario 2, 3 and 4 results show the trends in the accuracy of solutions and degree of non-uniqueness as the level of sensitivity of the sensors increases. The first subpopulation, which was dedicated to search for the best solution, was able to obtain a perfectly matching solution in almost all scenarios except Scenario 4 with the highest sensitivity for which an average prediction error of 0.12 was obtained. A similar trend of increasing accuracy with improved sensitivity was also observed for the alternative solutions (Figure 2.8). As the sensor sensitivity increases, the number of non-unique solutions (i.e., the number of alternative solutions) decreases (Figure 2.9). Correspondingly, the prediction error at the true source decreases while at the other potential sources the prediction error increases (Table 2.4), helping to rule out the non-unique solutions that are not the true source location.

**Table 2.4 The average prediction error (number of mishits) for non-unique sources located at nodes 10, 12, and 86**

| Location | Scenario 2 | Scenario 3 | Scenario 4 |
|---|---|---|---|
| Node 10 | 1.0 | N/A[1] | N/A[1] |
| Node12 | 0.671053 | 0.523438 | 0.347222 |
| Node 86 | 0.55 | 1.0 | N/A[1] |

[1] no feasible solution with no more than one mishit was found

### 2.6.3 Effect of quantity of data (based on Scenarios 5, 6, and 7)

#### 2.6.3.1 Scenario 5, 6, 7

The amount of data available for characterizing the source has a significant impact on the source identification problem, which was studied using several scenarios with different numbers of water quality sensors. Scenarios 5, 6, and 7 include 6, 3, and 1 binary sensor, respectively, in the network, and the sensitivity of the sensors was assumed to be 0.1 mg/l. It was observed that as the amount of data decreases with the decreasing number of sensors, the number of alternative solutions that fit the observations (i.e., non-uniqueness) increases. Figure 2.10, Figure 2.11 and Figure 2.12 show the alternative source locations in the network for 6, 3 and 1 sensor scenarios, respectively, and the number of locations where alternative sources were found was 2, 13, and 13 for Scenario 5, 6, and 7, respectively.

In the 3-sensor scenario (Scenario 6), contamination was detected at only one sensor, which was the same location as where the sensor was placed in the 1-sensor scenario (Scenario 7). Thus, active detection observations were same in both the cases. However, no detection at the other sensors in Scenario 6 provides useful information when reconstructing the source characteristics. This leads to some differences in the overall results of the two scenarios, resulting in slightly different sets of nodes for the 13 locations.

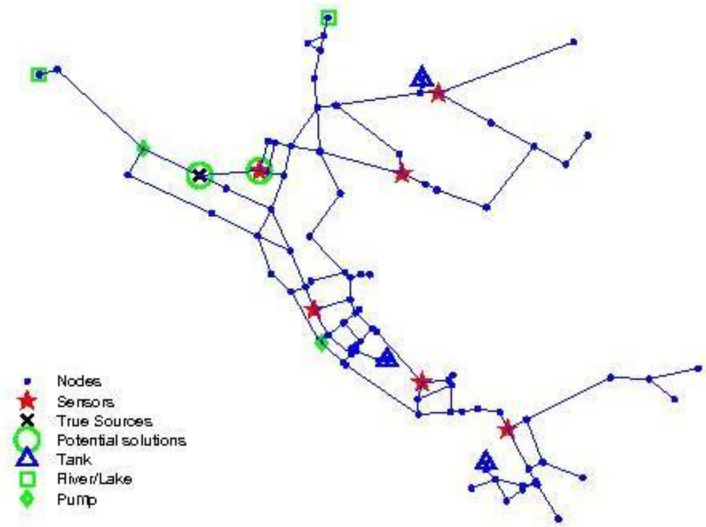



**Figure 2.10 Potential source locations identified for the 6-sensor case (Scenario 5)**

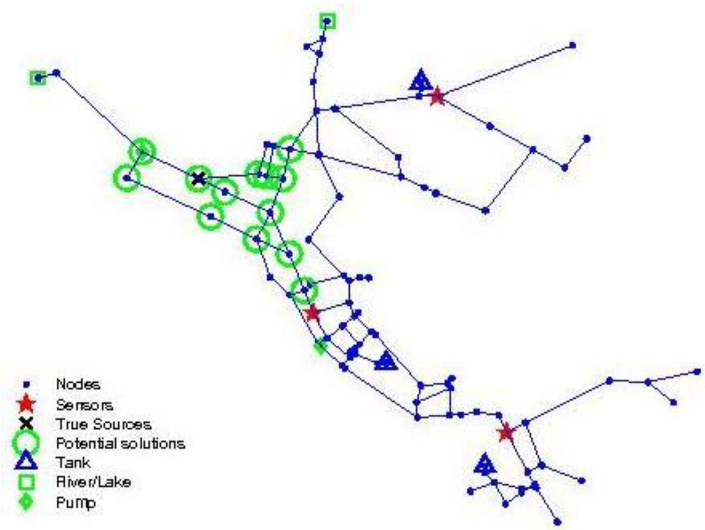



**Figure 2.11 Potential source locations identified for the 3-sensor case (Scenario 6)**

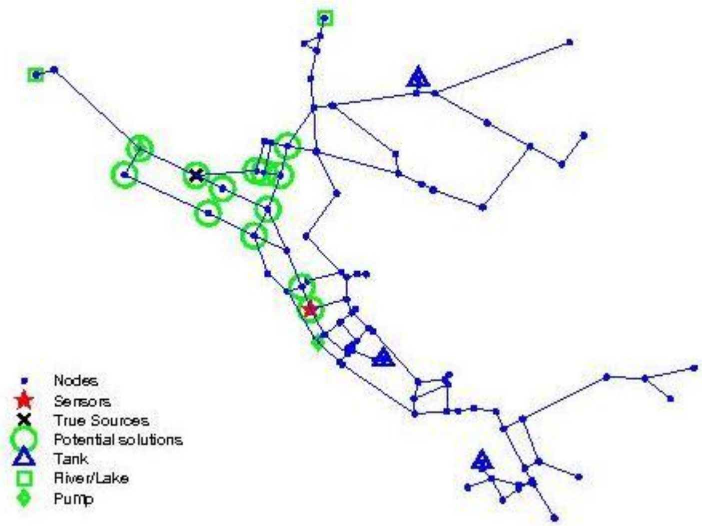
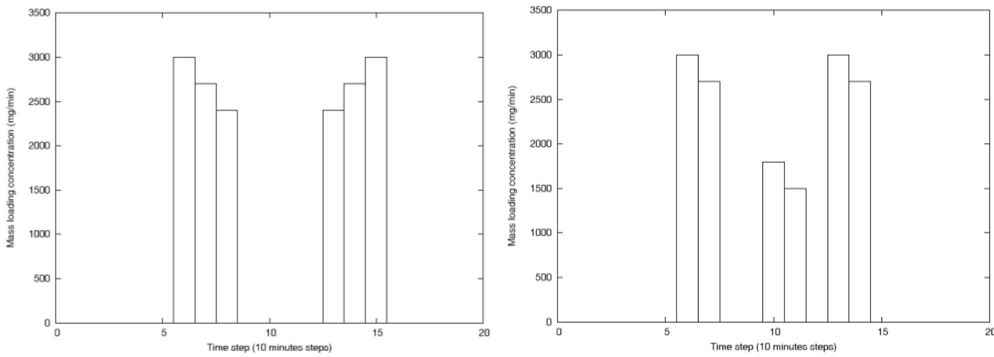



**Figure 2.12 Potential source locations identified for the 1-sensor case (Scenario 7)**

## 2.6.4 Testing under different contamination scenarios (based on Scenarios 8, 9, 10, and 11)

### 2.6.4.1 Scenario 8, 9: Intermittent contaminant injection

To test the applicability of the method to different source characteristics, different mass injection pattern were considered in Scenarios 8 and 9, where two different intermittent contaminant injection scenarios were used (Figure 2.13). For both scenarios, Node 12 was chosen as the true source location and six binary sensors with a sensitivity of 0.1 mg/l were assumed to be present in the network (Table 2.1).




**(a)** **(b)**

**Figure 2.13 Intermittent contaminant injection pattern used in Scenario 8 (a) and Scenario 9 (b)**

For both the scenarios, feasible solutions were identified at the true source location (Node 12) and an adjacent node (Node 86). For Scenario 8, the best solution in the first subpopulation was identified with an average error of 1.0 mishit and standard deviation of 1.069 mishits, and on average 1.24 (with a standard deviation of 1.238) alternative solutions were identified. The average prediction error for alternative solutions was found to be 1.398 mishits with a standard deviation of 1.299 mishits. Similarly, for Scenario 9, the average error of the best solution was 0.64 mishit (standard deviation 0.942 mishit), and on average 0.86 (with a standard deviation of 0.989) alternative solutions were identified with an average prediction error of 0.945 mishit (and standard deviation of 1.1612 mishits).

#### 2.6.4.2 Scenario 10, 11: Different source locations

To test the ability of the source identification method to identify a source existing at different locations in the network, in scenarios 10 and 11 the source location was placed at Node 20 and Node 54, respectively (Figure 2.14, Figure 2.15). The contaminant injection pattern was the same as in Scenarios 1-7. Six binary sensors with a sensitivity of 0.1 mg/l were included.

For scenario 10 with a source at location Node 20, feasible solutions were found at the true source location and five other locations, 13, 16, 17, 18, and 85, all of which are close to the true source location (Table 2.5, Figure 2.14). A perfect fit (i.e., zero prediction error) solution was identified by the first subpopulation during each of the 50 trials. On average 3.84 (with a standard deviation of 0.370) alternative solutions were identified. The average error for the alternative solutions was 0.83 mishit (with a standard deviation of 0.222 mishit).

**Table 2.5 The set of solutions obtained for a typical trial for Scenario 10**

| Solution | Location | Start time | No. of mishits |
|---|---|---|---|
| 1 | 18 | 11 | 0 |
| 2 | 17 | 12 | 1 |
| 3 | 13 | 11 | 1 |
| 4 | 85 | 11 | 1 |
| 5 | 20 | 8 | 1 |

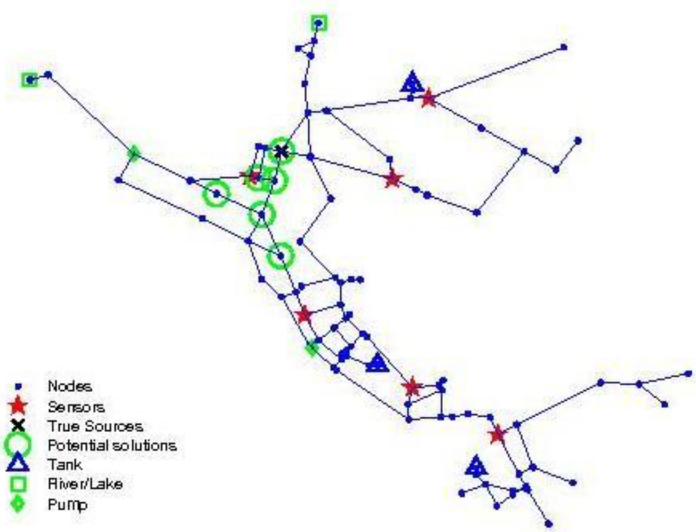



**Figure 2.14 Potential locations for contamination source for Scenario 10**

In Scenario 11 where the true source was located at location Node 54, feasible solutions were identified at the true location along with four other locations (i.e., nodes 16, 17, 18, and 88) (Table 2.6, Figure 2.15). A perfect fit solution was identified by the first subpopulation in all 50 random trials. In all cases, four alternative solutions were identified with a mean error of 0.925 mishit (with a standard deviation of 0.1450 mishit).

**Table 2.6 The set of solutions obtained for a typical trial for Scenario 11**

| Solution | Location | Start time | No. of miss-hits |
|---|---|---|---|
| 1 | 54 | 7 | 0 |
| 2 | 18 | 2 | 1 |
| 3 | 17 | 2 | 1 |
| 4 | 17 | 5 | 1 |
| 5 | 17 | 1 | 1 |

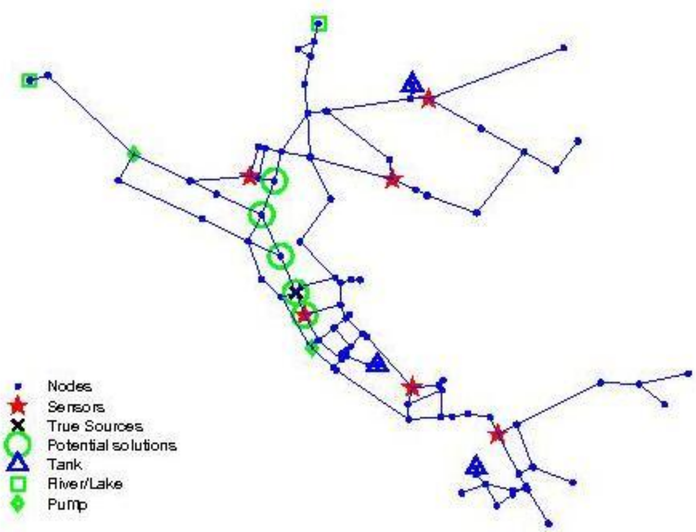



**Figure 2.15 Potential locations for contamination source for Scenario 11**

### 2.6.5 Testing for a different network (based on Scenarios 12 and 13)

For simulating the contamination scenario in the Micropolis network, water quality simulations were carried out at a 5-minute interval using EPANET. Five trigger-based binary sensors with a sensitivity of 0.1mg/l were installed in the network (Figure 2.16). Presented here are two scenarios (12 and 13) with the contamination source located at Node 787 and Node 653, respectively. In both scenarios the contamination source was assumed to be active for one hour starting at 12.00am (i.e., at time step 1) with a injection pattern as shown in Figure 17. For each scenario, the source characterization method was applied with 50 subpopulations to search for a maximum of 50 alternative solutions. Alternative solutions within a single mishit from the best solution were accepted as a feasible solution to the problem. The statistics of the solutions obtained for ten random trials are reported for each scenario.

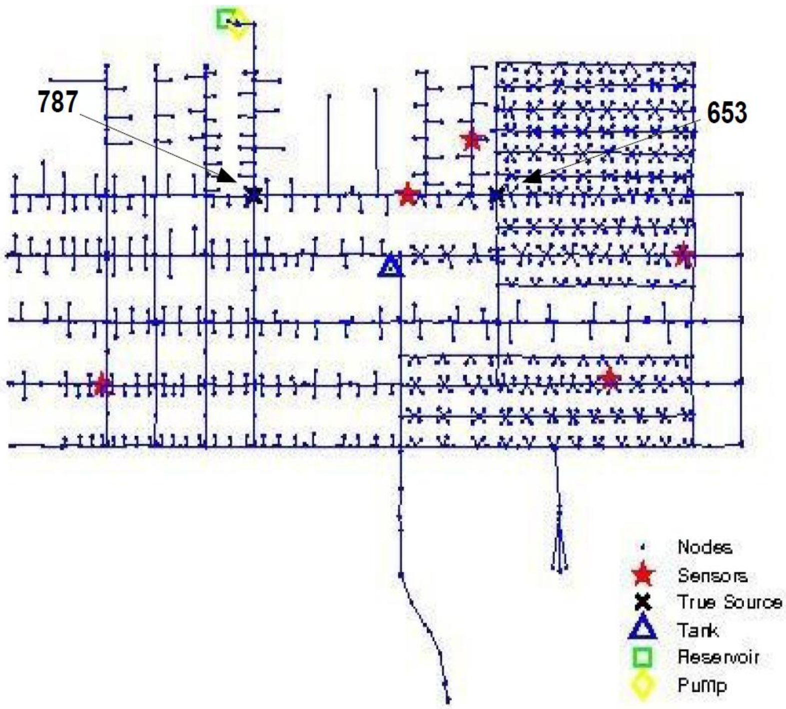



**Figure 2.16 Layout of the Micropolis water distribution network**

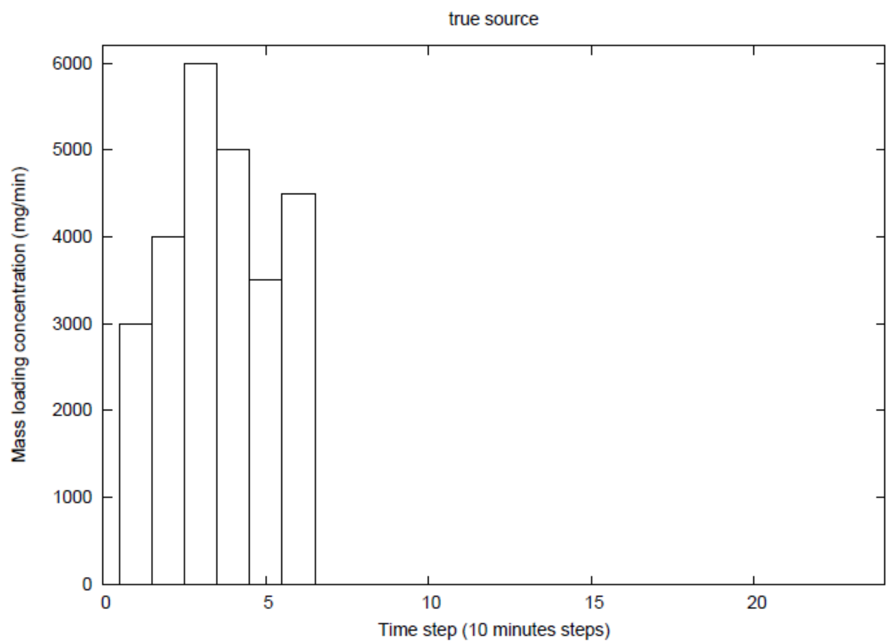



**Figure 2.17 Contamination injection pattern for the true source in Scenarios 12 and 13**

### 2.6.5.1 Scenario 12

In this scenario, the contaminant source was placed at Node 787 with a contaminant injection pattern as shown in Figure 2.17. Because the contaminant source (Node 787) is close to the source of water in the network, the contaminant spreads out to a large section of the system and is detected at all five water quality sensors. The average error of the solutions identified in the first subpopulation was 0.5 mishit with a standard deviation of 0.547 mishit. An average of 40.3 feasible alternative solutions (with a standard deviation of 3.226) was identified with an average error of 0.534 mishit (with a standard deviation of 0.0737 mishit). In each trial, the true source location and 13 other locations were identified (Figure 2.18).

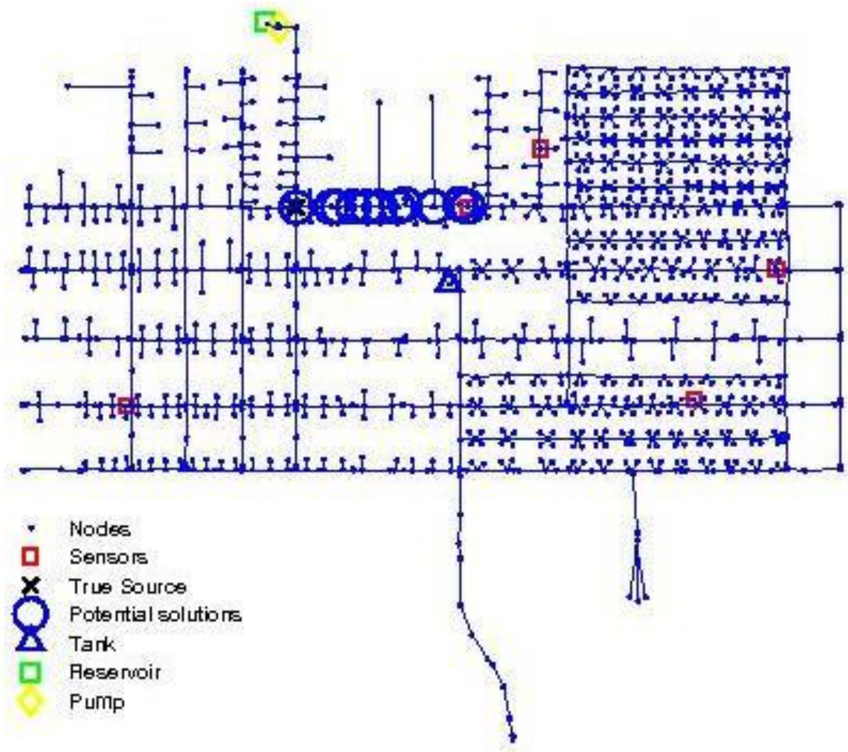



**Figure 2.18 Potential source locations identified in the ten random trials for Scenario 12**

#### 2.6.5.2 Scenario 13

In this scenario, the contaminant source was placed at Node 653 with a contamination injection pattern as shown in Figure 2.17. Again, ten random trials were carried out. The true source location (Node 653) was identified by the solutions for all ten trials. The best solution in the first subpopulation in all trials had an average prediction error of 0.5 mishit with a standard deviation of 0.577 mishit. In all trials, 36 alternative solutions were found to be feasible with an average prediction error of 0.75 mishit. In each trial, the true source location and seven other locations were identified (Figure 2.19).

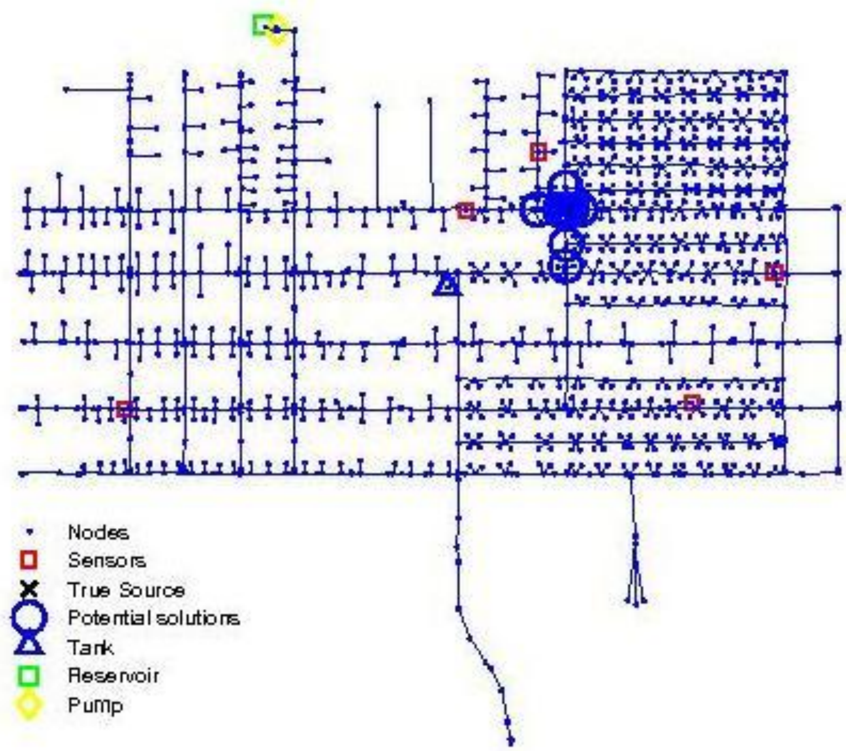



**Figure 2.19 source locations identified in the ten random trials for Scenario 13**

Results for Scenarios 12 and 13 demonstrate that the methodology described in this paper is able to solve the contaminant source characterization problem for the Micropolis example, which represents a medium-sized network. Besides consistently identifying the true source characteristics, it was also able to identify a set of non-unique solutions that is expected to fit the limited observations. Although the prediction error is relatively higher, the set of alternative source locations is able to pinpoint the potential source location to an area around the true source location.

## 2.7 Conclusions

This paper presents an evolutionary algorithm-based methodology for characterizing a pollutant source during a contamination event in a water distribution system that is being monitored using a set of sensors that report a binary signal based on the presence of contamination. The source characteristics were reconstructed using these binary signals.

The accuracy of contaminant source characterization depends primarily on the amount and quality of the data available. The effect of sensor sensitivity on the accuracy and non-uniqueness in source identification was studied. It was observed that the decrease in sensitivity of the sensors leads to a reduction in accuracy and an increase in the level of non-uniqueness in source prediction. Further, a decrease in the quantity of data resulted in an increase in the level of non-uniqueness in source identification. Extremely low quantity and quality of available data may make it impossible to reasonably resolve the contamination source characteristics. The methodology was tested for a range of different contamination scenarios to establish its robustness. The methodology was able to resolve the contaminant source to a small subset of possible locations while identifying the true source in all cases.

## References

Brumbelow, K., Torres, J., Guikema, S., Bristow, E., and Kanta, L. (2007). “Virtual Cities for Water Distribution and Infrastructure System Research.” World Environmental and Water Resources Congress 2007, Tampa, FL.

Dandy, G. C., Simpson, A. R., and Murphy, L. J. (1996). “An Improved Genetic Algorithm for Pipe Network Optimization.” Water Resources Research, 32(2), 449-458.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Cambridge, MA.

ILSI (1999). “Early warning monitoring to detect hazardous events in water supplies.” International Life Sciences Institute-Risk Science Institute.

Kumar, J., Zechman, E. M., Brill, E. D., Mahinthakumar, G., Ranjithan, S., and Uber, J. (2007). “Evaluation of Non-Uniqueness in Contaminant Source Characterization Based on Sensors with Event Detection Methods.” World Environmental and Water Resources Congress 2007, Tampa, FL.

Laird, Carl D., Biegler, Lorenz T., van Boleman Waanders, Bart G., Barlett, and Roscoe A. (2005). “Contamination source determination for water networks.” Journal of Water Resources Planning and Management, Vol. 131(2), 125-134.

Laird, Carl D., Biegler, Lorenz T., van Boleman Waanders, Bart G. (2006). “A Mixed-integer approach for obtaining unique solutions in source inversion of drinking water networks.” Journal of Water Resources Planning and Management, 132(4), 242-251.

Lingireddy, S. and Ormsbee, L. (2002). “Hydraulic Network Calibration Using Genetic Optimization.” Civil Engineering and Environmental Systems, 19(1), 13-39.

McKenna, Sean. A., Hart, David., Klise, Katherine., Cruz, Victoria., and Wilson, Mark. (2007). “Event detection from water quality time series.” World Water and Environmental Congress 2007, Tampa, FL.

USEPA (2005), “Technologies and techniques for early warning systems to monitor and evaluate drinking water quality: A state-of-art review.”

Savic, D. A. and Walters, G. A. (1997). “Genetic Algorithms for Least-Cost Design of Water Distribution Networks.” Journal of Water Resources Planning and Management, 123(2), 67-77.

Schwefel, H.-P. (1995). Evolution and Optimum Seeking, Wiley & Sons, New York.

van Bloemen Waanders, B. G., Bartlett, R. A., Bigler, L. T., and Laird, C. D. (2003). “Nonlinear Programming Strategies for Source Detection of Municipal Water Networks.” World Water and Environmental Congress 2003, Philadelphia, PA.

Vitkosvsky, J. P., Simpson, A. R., and Lambert, M. F. (2000). “Leak Detection and Calibration using Transients and Genetic Algorithms.” Journal of Water Resources Planning and Management, 126(4), 262-265.

Zechman, E. M., Brill, E. D., Mahinthakumar, G., Ranjithan, S., and Uber, James. (2006). “Addressing non-uniqueness in a water distribution contamination source identification problem.” Proceedings of 8th Annual Water Distribution System Analysis Symposium, Cincinnati, OH.

Zechman, E. M., and Ranjithan, S. (2004). “An evolutionary algorithm to generate alternatives (EAGA) for engineering optimization problems.” Engineering Optimization, Vol. 36(5), 539-553.

Zechman, E. M. and Ranjithan, S. (2007) "Generating Alternatives using Evolutionary Algorithms for Water Resources and Environmental Management Problems." Journal of Water Resources Planning and Management, 133(2), 156-165.

# CHAPTER 3. Water supply contamination source identification using noisy filtered sensor data

## 3.1 Introduction

Contamination threat management in a water distribution system (WDS) has received considerable research attention in the wake of recent terrorist attacks on critical infrastructure. A WDS consists of a network of a large number of pipes for supplying water at sufficient pressure for safe drinking as well as for fire fighting. The highly interconnected nature of a typical network topology makes a WDS vulnerable to accidental and intentional contamination. WDS networks are designed purposely for a high degree of redundancy for mixing at the nodes of the network so that the disinfectant (e.g., free chlorine) reaches all parts of the network. Any external contamination could rapidly spread within the network through mixing and transport. One defense against this security threat is to use a set of sensors in the network to monitor the system and to detect a contamination event. The sensor information can potentially be used to accurately and quickly identify the source characteristics (e.g., location and time of injection, and the contaminant mass loading pattern). Subsequently, the source characteristics could be used to help identify appropriate strategies to control the spread of the contaminant while maintaining the services, especially the fire fighting capacity, as much as possible.

The contaminant source identification problem has been approached using different methodologies by various researchers. Waanders et al. (2003) used a gradient-based nonlinear programming approach for the identification of a contaminant source in a WDS. Laird et al. (2005) developed an origin tracking algorithm and used nonlinear programming for solution of the source identification problem. Preis and Ostfeld (2006) used a hybrid model trees-linear programming approach for contamination source identification in WDS. Evolutionary algorithms (EAs) have emerged as an efficient and powerful tool for solutions of complex optimization problems and EAs have been successfully demonstrated for solving water distribution systems design problems (e.g., Dandy et al. 1996; Savic and Walters 1997). Preis and Ostfeld (2007) used a genetic algorithm for contaminant source identification. The source identification problem can be posed as an inverse problem with the source characteristics as the variables to be identified. In the present work, a simulation-optimization approach was used for solution of the inverse problem. The time series of observations recorded at sensors at spatially distributed monitoring stations in the network were used to reconstruct the source characteristics, i.e., the location of the contaminant source, time of the contaminant injection, and the contaminant injection pattern. The accuracy of source characterization depends on the amount and quality of the observations available. The quantity of observation is relatively invariant once the number of sensors and sampling frequency are predefined.

The quality of the data from the sensors depends upon the type of sensors installed in the system. The accuracy and resolution of the data may vary from a continuous and accurate measurement from an ideal sensor to detection or no-detection binary alarms based on the filtered data. The quality of data from binary sensors in turn depends on their sensitivity (i.e., the lowest detectable concentration) and accuracy. Malfunctions in sensor instruments may introduce additional uncertainty in the recorded data. The quality of data significantly influences the uncertainty in the source identification and limits the ability to uniquely estimate the source characteristics.

The challenge in solving the source identification problem is twofold: first, the solution should provide estimates of the source characteristics that match the observations closely; and second, the solution should be as unique as possible (i.e., only one or a few possible solutions should fit the observations within some acceptable degree of fit). These issues continue to be the focus of ongoing research. For example, Zechman et al. (2006) studied the issue of non-uniqueness in source identification problems, and Kumar et al. (2007; 2010) analyzed the problem of non-uniqueness arising due to data filtering when binary sensors are used.

This paper presents the results from an investigation using an evolutionary algorithm-based source identification procedure (extending the approach reported by Zechman et al. 2006 and Kumar et al. 2007; 2010), with the focus on assessing the effects of sensor errors on the solution quality. Specifically, this study evaluates the EA-based procedure to identify the source characteristics in the presence of noise in the data caused by malfunctioning and lack of precision of binary sensors.

## 3.2 Problem formulation

Source identification during a contamination event is assumed in this work to be based on water quality observations at sensors with binary signals, indicating detection or non-detection according to a predefined sensitivity (e.g., detection is signaled when the measured concentration is above a threshold concentration). The following optimization problem is formulated mathematically to obtain the source characteristics that yield estimates of contaminant concentrations that closely match the sensor observations. The following characteristics of the contaminant source are decision variables for the optimization problem:

(a) Location ($L$) of the contaminant source in the network. Location of the source is assumed to be at one or more of the nodes in the WDS network.

(b) Start time of the contaminant injection into the system ($T_0$). Start time is measured as the time elapsed from the start time of the simulation.

(c) The pattern of contaminant injection ($C_0$) into the system measured in terms of milligrams of contaminant injected per minute into the system. Injection rate is assumed to be constant during each water quality time step of the simulation. No assumptions are made regarding the magnitude, pattern or duration of the contaminant injection.

The optimization problem is formulated as:

$$Minimize \quad E = \sum_{i=1}^{N_S} \sum_{t=1}^{T_S} \left| BS_{i,t}^{s} - BS_{i,t}^{a} \right| \qquad (1)$$

$$subject\,to \quad BS_{i,t}^{s} = \begin{cases} 0, \; if \; C_{i,t}^{s}(L,T_0,C_0) < C_{th} \\ 1, \; if \; C_{i,t}^{s}(L,T_0,C_0) \geq C_{th} \end{cases} \qquad (2)$$

$$where \quad BS_{i,t}^{a} = \begin{cases} 0, \; if \; C_{i,t}^{a} < C_{th} \\ 1, \; if \; C_{i,t}^{a} \geq C_{th} \end{cases} \qquad (3)$$

The objective function (Equation 1) is a measurement of error represented as the total number of mishits at the sensors. A simulated expected signal for the given values of $L$, $T_0$, and $C_0$, assuming a perfect functioning sensor, will be considered a mishit if it does not match the actual signal. Specifically, there is a mishit when there is a simulated detection signal when the actual signal was of no-detection or vice versa. The optimization problem is to obtain the values of $L$, $T_0$, and $C_0$ that minimize the total number of mishits between the simulated and actual sensor signals. Equation 2 filters the simulated concentration levels for a given set of source characteristics at the sensors into detection/no-detection binary signals depending on the sensitivity of the sensor for the particular contaminant. $BS_{i,t}^{s}$ is the simulated binary signal for the given source characteristics at sensor $i$ at any time $t$; $BS_{i,t}^{a}$ is the actual binary signal for the event at sensor $i$ at time $t$, recognizing that there may be errors associated with it. $N_s$ is the number of sensors in the network; $T_S$ is the total time for which contaminant mixing and transport in the network is being simulated. The simulated concentrations of the contaminant are given by $C_{i,t}^{s}(L,T_0,C_0)$ for a given solution as a function of location ($L$), time of measurement ($t$), start time of the contamination injection ($T_0$), and the contaminant injection pattern ($C_0$), respectively.

$C_{th}$ is a constant defining the threshold concentration level for determining detection and generating a binary signal by the sensors. The constants $C_{i,t}^{a}$ are the actual measured concentrations of the contaminant being monitored, again recognizing that there may be errors associated with these measurements.

In this study, the sensitivity of a solution to a source identification problem is investigated when there is noise due to uncertainties or errors in the values for $C_{i,t}^{a}$ and $BS_{i,t}^{a}$. Different scenarios (described later in Section 3.4) with different types and degrees of noise in the observations are examined using the following methodology.

## 3.3 Methodology

The optimization problem described by Equations 1-3 was solved using an evolutionary strategy (ES) based search as described below. As discussed above, the solution to this inverse problem may not necessarily be unique when the observations are insufficient or erroneous. Thus, the search for the solutions to this problem should not only identify the source characteristics that best fit the noisy observations, but also identify the set of all other possible source characteristics, each of which fit those observations within some acceptable level of prediction error. The ES-based search was extended, as describe below, to simultaneously identify such a set of alternative solutions.

### 3.3.1 Evolutionary algorithm for generating alternatives

Zechman and Ranjithan (2004) designed an evolutionary algorithm for generating alternatives (EAGA) based on concepts of niched co-evolution to generate a small number of good but maximally different alternatives. In this algorithm, subpopulations collectively search for different alternative solutions. Each solution is represented by one subpopulation that undergoes an evolutionary search. The survival of solutions in each subpopulation depends upon their performance with respect to the objective function as well as how far they are from the other solutions in the decision space (to identify a solution that is as different, or non-unique, as possible). Thus, the evolution of solutions in a subpopulation is influenced by solutions in the other subpopulations, forcing the migration of each subpopulation towards good but distant regions of the decision space. The ES-based search was extended in this work by adapting the concepts of EAGA. A schematic of the ES-based search to identify a set of alternative solutions is shown in Figure 3.1 and the key steps of the algorithm are:

*Step 1.* Initialization – create an initial population with $P$ subpopulations (each of population size $\mu$), where $P$ is the number of alternative solutions being sought. Let $SP_p$ ($p$=1, . . , $P$) represent the index for subpopulation $p$. The first subpopulation ($SP_1$) is dedicated to the search for the solution with the best objective function (i.e., minimum prediction error) value, while the others search for maximally different alternatives.

*Step 2.* In each subpopulation $SP_p$ ($p$=1, . . ., P), apply an adaptive mutation operator to generate $\lambda$ offspring.

*Step 3.* In $SP_1$, evaluate the fitness (Equation 1) of each solution and identify the solution in the subpopulation with the best fitness (i.e., minimum prediction error). This solution will serve as the benchmark for determining the relaxed target for solutions in other subpopulations $SP_p$, ($p$=2, . . ., $P$).

*Step 4.* In $SP_p$, ($p$=2, . . ., $P$), evaluate all solutions with the respect to the objective function. If the value of the objective function of a solution is within the relaxed target based on the best fitness in $SP_1$, it is flagged as 'feasible' (else 'infeasible').

*Step 5.* Calculate the centroid of each subpopulation $SP_p$ (p=1, . . ., P). The centroid is calculated in terms of the location decision variable. Spatial coordinates of the locations associated with each individual are used for the calculation of a centroid.

*Step 6.* In subpopulations $SP_p$, ($p$=2, . . .,$P$), calculate the distance of each individual from the centroid of each of the subpopulations. The distance from the closest centroid is used as the distance metric.

The simple Euclidean distance between the location of an individual and the location of the centroid of a subpopulation was used as the distance metric. For the contaminant source identification problem, identifying sources at different possible locations was identified as a key problem, and thus for all reported scenarios the distance measures in terms of location were used during the selection step (step 7). However, the distance measure can easily be extended to include time and contaminant injection pattern also.

*Step 7*. In each subpopulation $SP_p$, apply a selection operator. In $SP_1$, the selection is based on the objective function value only. The solutions ($\mu$+$\lambda$) are ranked based on objective function value, and the top $\mu$ solutions survive to the next generation.

In subpopulations $SP_p$, ($p$=2, . . ., $P$), the selection of an individual is based on its feasibility (determined in Step 4) and its distance from the other subpopulations. Specifically, solutions flagged as 'feasible' are ranked from highest to lowest with respect to their distance values. Among the pool of feasible solutions, the individuals with higher distance values are preferred for selection. If the number of feasible solutions is less than the population size, infeasible solutions with best objective function values are selected to be added to the population.

Step 8. Stop if the termination criterion is satisfied, else go to Step 2. A limit on the maximum number of generations was used as the termination criterion in this study.

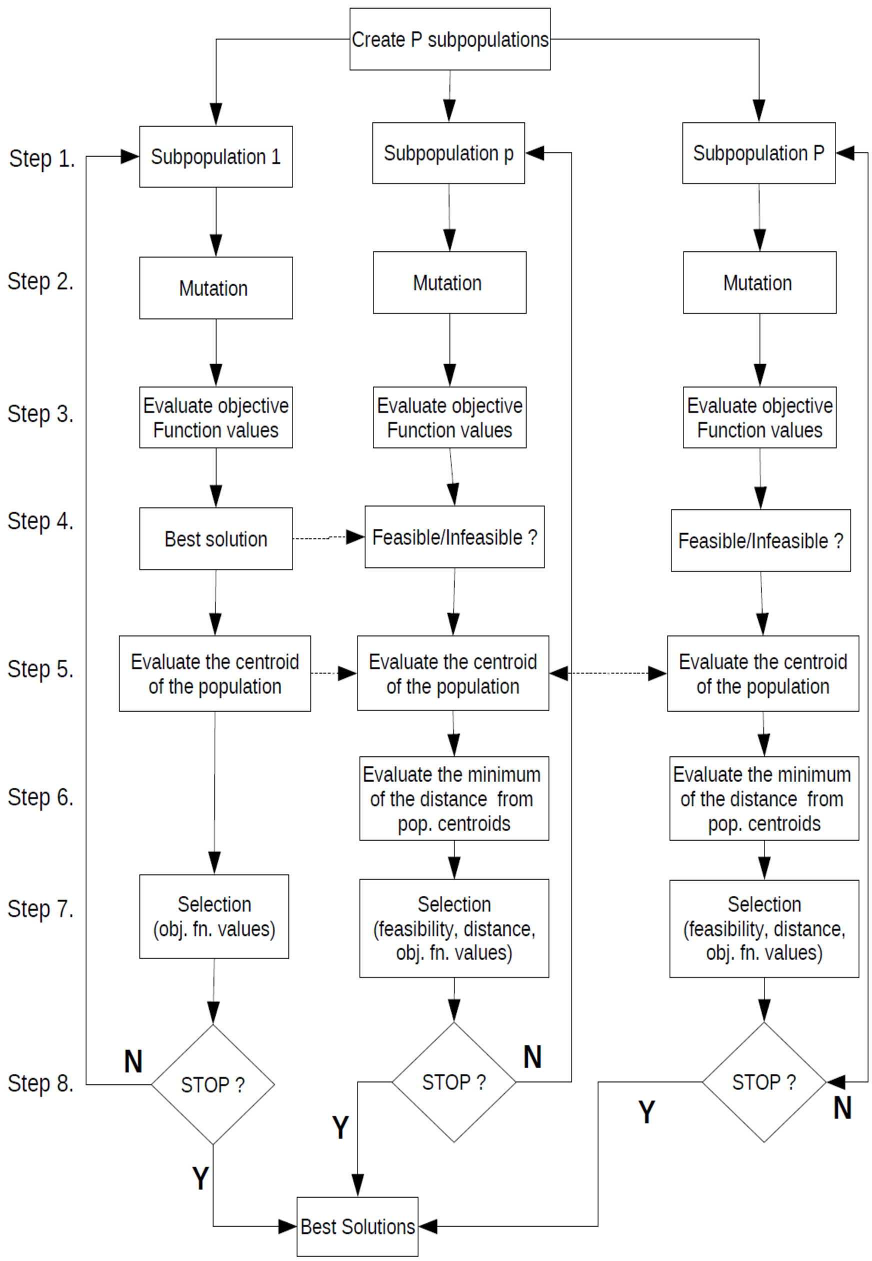



**Figure 3.1 Evolutionary algorithm for generating alternatives**

### 3.3.2 Representation of variables

A contamination event in a water distribution system can be characterized by three types of decision variables: location of the contaminant source, start time of the contaminant injection, and the rate and pattern of contaminant injection. The following representations schemes were used to encode them:

1. Location of contaminant source: A water distribution system includes a network of pipes and joints. The spatial distribution of these pipes and joints depends on the specific network. Every joint (referred to as a node) in a pipe network is assumed to be a potential location for contaminant injection. The nodes are sequentially numbered, and an integer variable was used to encode the node number, bounded by 1 and the total number of nodes in the network.

2. Start time: An integer variable was used to represent the time when the contaminant injection began. This number refers to the discrete time step in the dynamic simulation of the contaminant mixing and transport. The value of the integer variable is bounded by 1 (i.e., start time of the simulation) and the time step when the first sensor was triggered.

3. Contaminant injection pattern: The contaminant injection pattern is represented as an array of real values denoting the mass loading values, where each value indicates the mass loading during a single time step. As the duration of injection is not predefined, the length of this array varies, enabling each solution to represent contaminant injection patterns of different durations. The injection pattern is encoded as a linked-list, along with a special feasibility preserving mutation operator as described below.

### 3.3.3 Genetic operators

Except for the integer variable representing the start time, the variables for location and contaminant injection pattern represent problem-specific characteristics that are best addressed using special mutation operators as described below. All mutation is conducted via an adaptive mutation strategy.

1. Location of contaminant source: When mutating an integer value that represents a node in the network, the new values are chosen using a Gaussian-like adaptive mutation. In this network problem, a pseudo-Gaussian mutation is achieved by a special operation where the new node number is selected based on topology adjacency and node connectivity

information. For each node, a list of one-node-away connected neighbors, two-node-away connected neighbors, and so on was first generated to capture the connectivity and adjacency of the network. Then the mutation parameter, set adaptively, specifies the distance between the current node and the child node. A child node at that distance away is then selected randomly from the connectivity/adjacency preserving list.

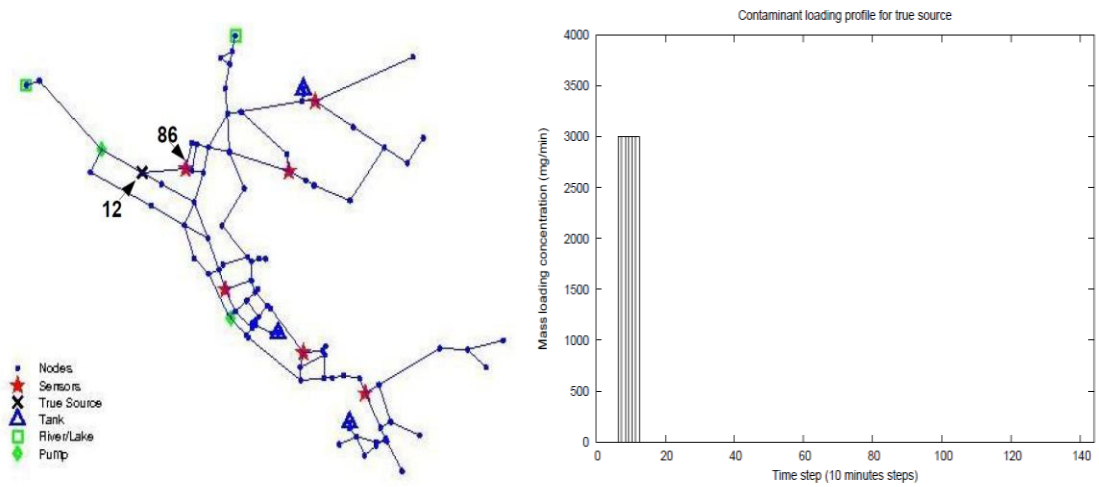
2. Start time: A simple adaptive Gaussian mutation operator for an integer variable is used, and the new value is restricted to be selected within the bounds for this variable.

3. Contamination injection pattern: A special mutation operator was designed to operate on this linked-list of real values representing the injection pattern. To allow the length of this linked-list to be flexible, three different mutation operations were defined. The first two are addition of link-items and deletion of link-items, both of which change the duration of the profile, and the third is a standard Gaussian mutation that changes the real value in the linked-list to modify the contaminant injection rate at the corresponding time step. While no restrictions are applied to the length of injection pattern, the operator does not allow the injection loading to continue beyond the end of simulation time, thus ensuring the feasibility of the solution. The value of the injection rate during any time step has a lower bound of zero and a specified upper bound. The mutation operator allows the injection rate to take the value of zero, thus making it possible to explore any possible continuous or intermittent contaminant injection pattern.

## 3.4 Case studies

### 3.4.1 The example water distribution networks

The ES-based search for detecting potential source characteristics is demonstrated using case studies based on two WDS networks. The first example, Network 1, is shown in Figure 3.2 and is available as a part of the standard EPANET distribution. Network 1 consists of 97 nodes including two water supply sources, three tanks and 117 pipes.




(a) Network (b) Injection pattern

**Figure 3.2 Network 1 layout and contamination scenario**

To generate a set of synthetic observations for an illustrative hypothetical contamination event, a non-reactive contaminant source was introduced into the network. Six sensors (shown by stars in Figure 3.2) were placed in the network based on engineering judgment to monitor the concentration levels as a contaminant spreads through the network. The pipe flows in the network were simulated hourly over a 24-hour time period and were assumed to be at steady-state within each hour of the simulation. For each hourly hydraulic condition, the contaminant transport was simulated at 5-minute intervals, and the concentration values at the sensors were observed at the end of each 10-minute increment. These concentration observations were then processed to trigger a binary signal representing the presence of contamination based on a fixed threshold value. For this study the sensor threshold was assumed to be 0.1mg/l.

To establish the robustness and general applicability of the methodology, testing was also carried out using the larger network shown in Figure 3.3. This WDS network is part of the virtual city called Micropolis developed by Brumbelow et al. (2007) to provide realistic, publicly-available WDS datasets for research studies without compromising public security.

Micropolis mimics a small city of approximately 5,000 people in a historically rural region. It is served by a single 440,000 gallon (1,670,000 liter) elevated storage tank. The WDS consists of 1236 nodes, 575 mains (16.25 km total length), 486 service and hydrant connections (11.42 km total length), 197 valves, 28 hydrants, 8 pumps, 2 reservoirs, and one tank. The 458 demand nodes are composed of 434 residential, 15 industrial, and 9 commercial/institutional users. Diurnal demand patterns are defined with hourly time intervals. Total daily demand of the system is 1.20 MGD (4.54 Ml/day) with minimum and maximum hourly demands of 0.68 MGD (2.57 Ml/day) and 1.66 MGD (6.28 Ml/day). For simulating the contamination scenarios in the Micropolis network, water quality simulations were carried out at 5 minutes intervals using EPANET (Rossman 2000). Five threshold based binary sensors of sensitivity 0.1 mg/l were placed in the network based on engineering judgment. All simulations in both the networks were assumed to start at 12:00AM.

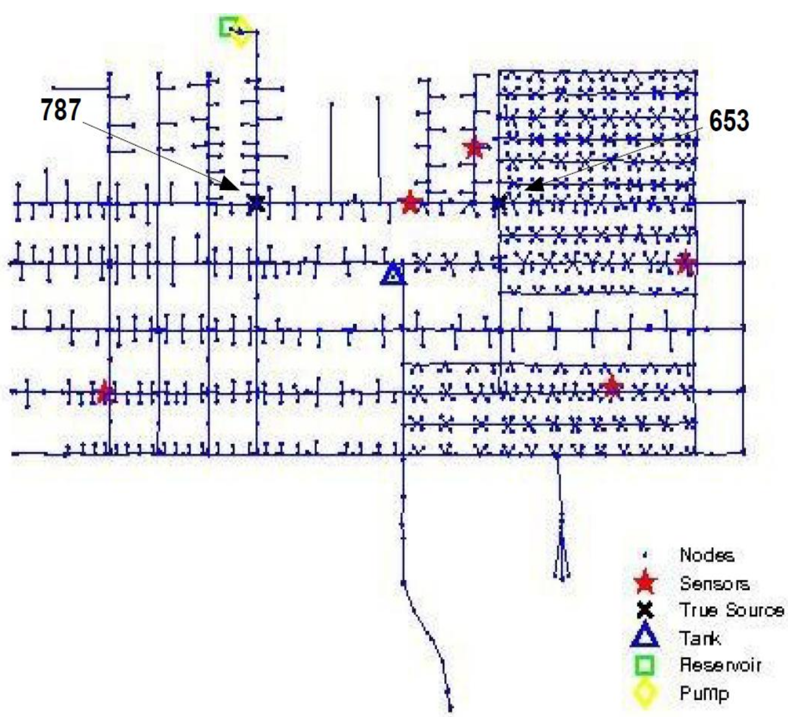



**Figure 3.3 Micropolis water distribution network**

### 3.4.2 Scenarios for Network 1

Scenarios 1-4 were constructed to study the effects of different types and levels of noise in the sensor observations on the solution to the source identification problem for Network 1. The contamination event was assumed to occur at location 12 (see Figure 3.2a) starting at 1:00 AM (i.e., time step 7) with the contaminant injected at a uniform rate of 3000 mg/min for one hour.

*Scenario 1*: This scenario is the base case representing a perfectly functioning sensor network and perfect measurements. It is used to provide a benchmark for evaluation and comparison of the noisy scenarios.

*Scenario 2*: The precision of a sensor instrument in detecting contaminant concentration introduces uncertainty in the data. This scenario studies the effect of noise in data due to inaccuracies of the threshold-based trigger mechanism of a binary sensor. A random fluctuation in the threshold value leads to potential noise in the binary signals observed at these sensors. The result could be a false-positive signal (i.e., be turned on when it should be off) or a false-negative signal (i.e., be turned off when it should be on). To simulate the error scenario, a normally distributed error was introduced in all of the sensors in the network. A maximum error of ±10% was introduced as follows.

$$Threshold = Threshold_{true} + 0.10 \times Threshold_{true} \times N(0,1) \tag{4}$$

where $N(0,1)$ is a normal variate with mean zero and variance 1.

*Scenario 3*: Sensors installed in a distribution network continuously monitor the system and signal an alarm in the event of contaminant detection. These sensors may malfunction intermittently and randomly and trigger either a false-positive or a false-negative signal. A false-positive situation would typically be easier to recognize and correct since an isolated malfunctioning sensor being on when all others are not would get the attention of the water distribution system operators. On the other hand, a false-negative condition may tend to go unnoticed since no attention is drawn to that malfunctioning sensor unless other sensors in the vicinity are signaling otherwise. A false-positive case that is noticed and corrected can still add noise, however, to the overall pattern of sensor observations since the contamination information at that sensor location during the period of its malfunction is lost and remains unknown. A subset of the sensors (two out of the six) is assumed to malfunction intermittently and randomly in this scenario. Five random failure events were introduced at the sensors, each lasting for an hour before it is restored to normal. Sensors were assumed to issue false-positive alarms during failures.

*Scenario 4*: This is a scenario with conditions similar to those in Scenario 3 with all six sensors malfunctioning in an intermittent and random manner.

These four scenarios were designed to the test the efficacy and robustness of the methodology under the conditions of increasing level of noise. They were used to test the methodology when applied to Network 1.

### 3.4.3 Scenarios for Micropolis network

Scenarios 5-8 were used to test the methodology for the Micropolis network.

*Scenarios 5 and 6*: These scenarios were designed as base cases with contaminant sources located at nodes 787 and 653 respectively. Perfectly working threshold based binary sensors at the five sensor locations shown in Figure 3 were assumed to be present.

*Scenarios 7 and 8*: Random and intermittent sensor failure scenarios (similar to Scenario 3 above) were studied in these scenarios where the true sources are nodes 787 and 653, respectively. In each scenario two out of five sensors were assumed to experience five failure events during the simulation period of 24 hours. Each failure event lasts for an hour after which the system was restored to normal. Sensors were assumed to issue false positive alarms during failures.

### 3.4.4 Scenario results for Network 1

Analysis was carried out under different scenarios of perfect and erroneous sensor signals in Network 1. In all optimization runs of the ES-based search for generating alternatives, population size ($\mu$) was set equal to 200 and the number of offspring ($\lambda$) was set equal to 200. The total number of generations was limited to 200. For all analysis, five subpopulations were used to search for a maximum of five alternative solutions. The first subpopulation was used to search for the best possible solution, while the other subpopulations searched for alternative solutions that are maximally different from each other while being within an allowable range of prediction error compared to the best solution obtained by the first subpopulation. The allowable range (i.e., the feasibility criterion in Step 4 of the EAGA procedure) for the alternative solutions was defined as one mishit more than the prediction error of the best solution in the first subpopulation. This means that the alternative solutions match the observations of binary signals with at most one mishit more than the best fit solution.

In the following subsections, the results for each scenario are described. While for each of these scenarios a representative solution is reported, the statistics of results for the first subpopulation and feasible alternative solutions are also reported. All reported statistics are based on 50 random trials for each scenario. These statistics are summarized and compared for all four scenarios after the scenario results are described.

#### 3.4.4.1 Scenario 1

Table 3.1 shows the decision variable values for location (L) and start time ($T_0$) for the set of alternative solutions obtained for a typical run. The injection profile ($C_0$) is shown in Figure 4 for each solution.

**Table 3.1 The set of solutions from a representative trial for Scenario 1**

| Solution | Location | Start time | No of mishits |
|---|---|---|---|
| 1 | 12 | 7 | 0 |
| 2 | 86 | 10 | 1 |
| 3 | 12 | 8 | 1 |
| 4 | 12 | 9 | 1 |

Solution 1 indicates that the procedure was able to identify the true source location (Node 12) and start time (7) exactly, with zero mishits (i.e., the solution fit all observed binary signals exactly). While the identified solution accurately matched the true source in terms of location and start time, the injection pattern identified is different than that of the true source (Figure 3.4). In addition, three other solutions were identified with only one mishit, indicating several sets of source characteristics other than the true set that are able to fit the observations within the range of prediction error of only one mishit. These solutions, however, are not too different from the true source. Solutions 3 and 4 point to the same source node (Node 12), and solution 2 points to Node 86 that is only one link away from the true source node (see Figure 3.2). Although the ES-based search method searched for solutions that are as different as possible from each other, the alternative solutions appear very close to each other in decision space, indicating that the available sensor information was sufficient to narrow down the source characteristics with only a little ambiguity (or small degree of non-uniqueness).

In each of the 50 random trials, at least one of the solutions always identified the source characteristics with a perfect fit to the sensor observations. The best solution in the first subpopulation was always a perfect fit with the actual sensor data, while the alternative solutions were identified with an average error of 0.75 mishits (standard deviation 0.279). On average, 2.6 alternative solutions (with a standard deviation of 0.76) were found during the

50 random trials. At least one of the solutions was identified at the true location (Node 12) during every trial. Collectively, the node numbers for the sources identified in the solutions over the 50 trials were Node 12 (the true source location) and Node 86, a node close to Node 12 (see Figure 3.1).

While values of the location and start time for the feasible solutions were always found to be close to those of the ‘true source,’ widely different injection patterns were obtained. Multiple solutions were found at the same location with similar (even exactly the same in some instances) start times but different injection patterns which resulted in exactly the same fit to the observed sensor signals. For example, solutions 3 and 4 (Table 3.1) were at the same location with the start times differing by a single time step and exactly the same prediction error, but with different injection profiles (Figure 3.4). Since similar observations were made for all the scenarios presented in the paper, and since the location and start time were considered to be the most important variables, the results and discussions presented below omit the details of the injection profiles. Injection patterns, however, do have an important role in determining the non-uniqueness of the solution and may be important in some specific cases such as the design of contaminant containment strategies.

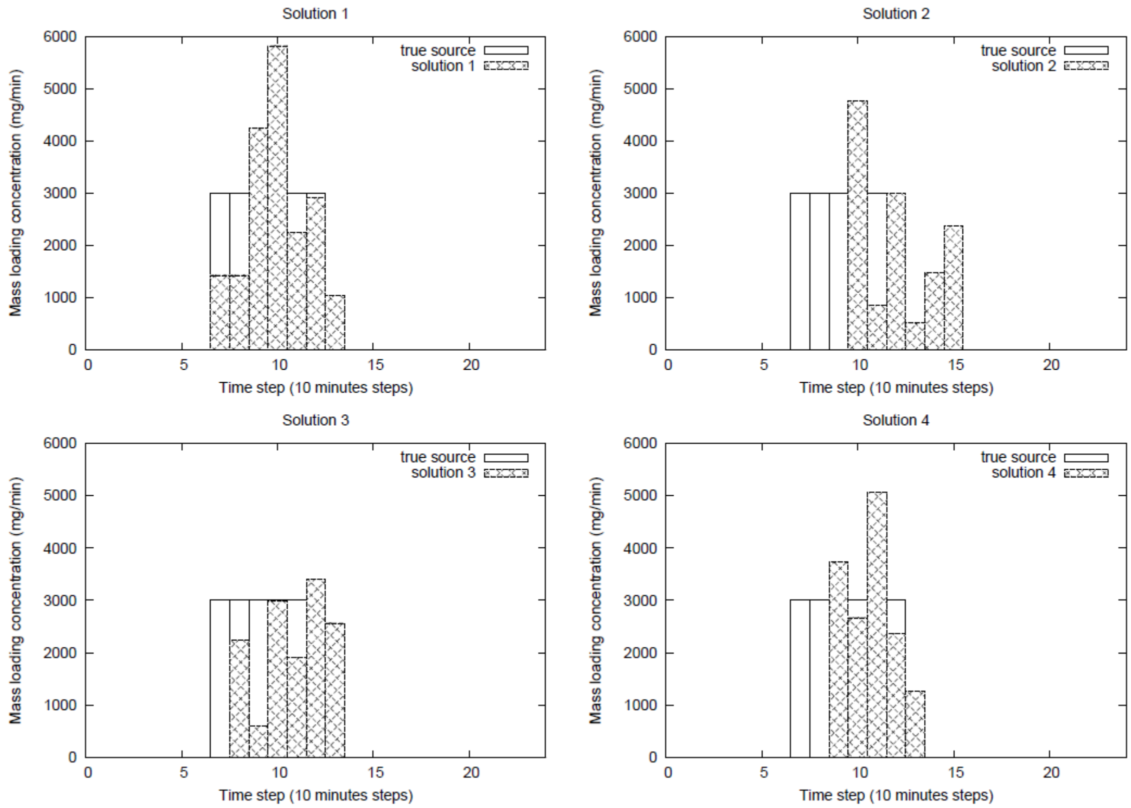



**Figure 3.4 Injection loading profiles for four solutions of scenario 1**

Figure 3.5 shows the convergence of the ES-based search in terms of average fitness of the five subpopulations. The first subpopulation, which was dedicated to search for the best fit solution, converged faster than the other subpopulations that were exploring for different solutions. The fitness value and the decision vector of the best solution of the first subpopulation are used as input to the other subpopulations which are searching for maximally different solutions in the decision space that are within one mishit of the best fitness value. This target on the fitness is imposed as a dynamically tightened constraint, resulting in the lag in convergence as seen in Figure 3.5.

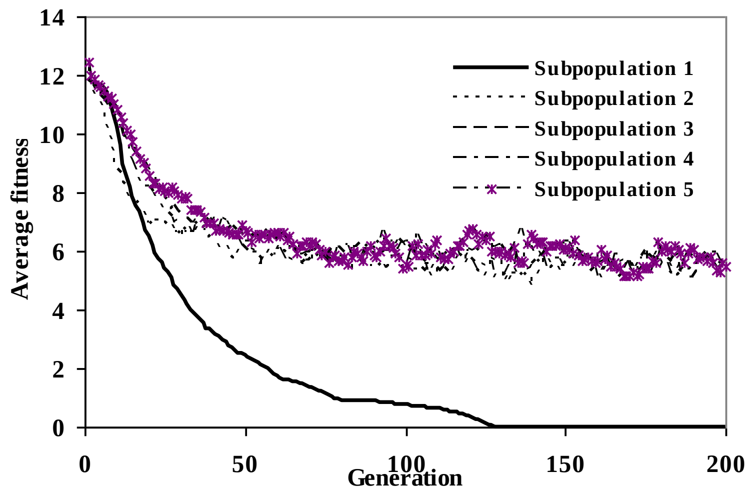



**Figure 3.5 Convergence of the ES-based search for subpopulation**

#### 3.4.4.2 Scenario 2

Table 3.2 shows the decision variable values for the set of alternative solutions obtained for a typical run of this scenario. The solution that yielded the best fit (with a single mishit) was identified to be located at Node 12 with a start time of 7 (same location and start time as of the true source). A second solution was identified at Node 86, which is adjacent to the true source Node 12 with a start time and injection pattern different from that of true source but matching the observation with only two mishits. Compared to the perfect observation data case (Scenario 1), the fit at the sensors is slightly poor because of the noise in the observations due to the erroneous trigger mechanism of the threshold based sensors. The best solution for the first subpopulation was identified with an average error of 1.70 mishits (standard deviation of 0.462). On average, 2.74 alternative solutions (with a standard deviation of 0.985) were obtained with average error of 2.63 mishits (standard deviation of 0.441). During all 50 trials, at least one alternative solution pointed to the true source node location (Node 12). Based on the 50 trials, solutions were identified at Nodes 12 and 86 (as in scenario 1), and in addition solutions were also identified at Node 10 due to the noise in the data. The injection profile for each solution in Table 3.2 is shown in Figure 3.6.

**Table 3.2 The set of solutions from a representative trial for Scenario 2**

| Solution | Location | Start time | No. of mishits |
|---|---|---|---|
| 1 | 12 | 7 | 1 |
| 2 | 86 | 10 | 2 |

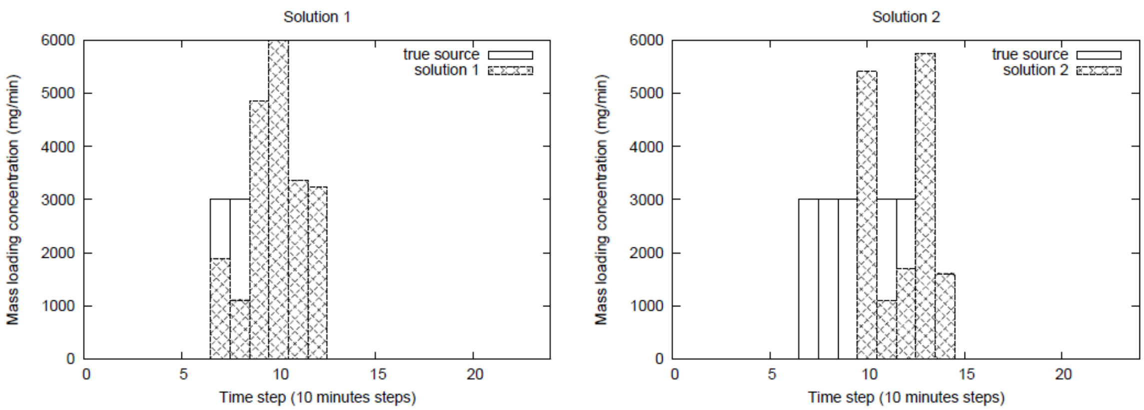



**Figure 3.6 Injection profiles for two solutions of Scenario 2**

### 3.4.4.3 Scenario 3

Table 3.3 shows the decision variable values for the set of alternative solutions for a typical run of this scenario. The injection profile for each solution is shown in Figure 3.7.

**Table 3.3 The set of solutions from a representative trial for Scenario 3**

| Solution | Location | Start time | No. of mishits |
|---|---|---|---|
| 1 | 12 | 8 | 52 |
| 2 | 10 | 6 | 57 |
| 3 | 86 | 10 | 57 |
| 4 | 86 | 9 | 55 |

The best solution obtained for this run identified the correct location and a start time near to the true time (8 as compared to 7 for the true source) with 52 mishits. Due to the higher level of noise in the data, solutions within a range of 10% greater error than that of best solution in the first subpopulation were accepted as alternative solutions. Two alternative solutions with the prediction error of 57 mishits were identified at Nodes 10 and 86. A fourth alternative solution was also identified at node 86 with an error of 55 mishits. During 50 trials, the best solution was identified in the first subpopulation with an average error of 52.16 mishits and standard deviation of 0.421. On average, 3.48 alternative solutions (with a standard deviation of 0.735) were identified over the 50 random trials with an average error of 56.79 mishits (standard deviation of 0.528). As in the previous two scenarios, the true source node was always among the set of solution locations for all the trials. Solutions were identified at five locations (10, 12, 18, 85 and 86) for this scenario.

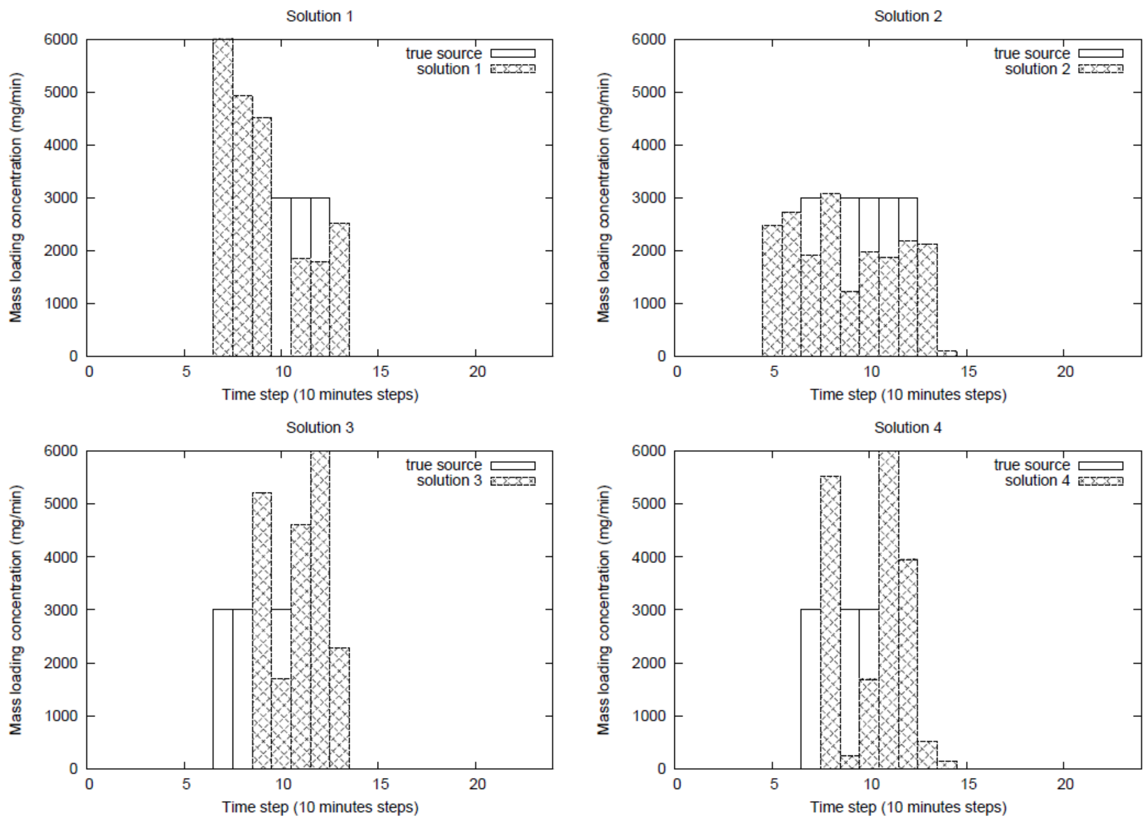



**Figure 3.7 Injection profiles for four solutions of Scenario 3**

### 3.4.4.4 Scenario 4

Table 3.4 shows the decision variable values for the set of solutions for a typical run for this scenario. Figure 3.8 shows the injection profile for each solution.

**Table 3.4 The set of solutions from a representative trial for Scenario 4**

| Solution | Location | Start time | No. of mishits |
|---|---|---|---|
| 1 | 12 | 8 | 146 |
| 2 | 13 | 10 | 153 |
| 3 | 17 | 12 | 153 |
| 4 | 18 | 10 | 153 |
| 5 | 10 | 7 | 153 |
| 6 | 86 | 12 | 153 |
| 7 | 13 | 12 | 153 |

The best solution identified was at the correct source location and with a start time near to the true time (9 instead of 7). This solution had 146 mishits (Table 3.4) at the sensors. Six alternative solutions were also identified with 153 mishits, but they point to different nodes close to the true source node. In this scenario, which has the highest degree of noise in the observations, the prediction errors were the highest among all scenarios; however, the true source node was still identified among the set of solution locations in each of 50 trials. The best solution was identified in the first subpopulation with an average error of 145.84 mishits (standard deviation of 0.738). On average, 7.46 alternative solutions (with a standard deviation of 1.50) were identified during the 50 trials with an average error of 152.27 mishits (standard deviation of 0.960). Solutions were identified at nine different locations (nodes 12, 86, 10, 18, 85, 1, 13, 17, and 20), all close to the true source location.

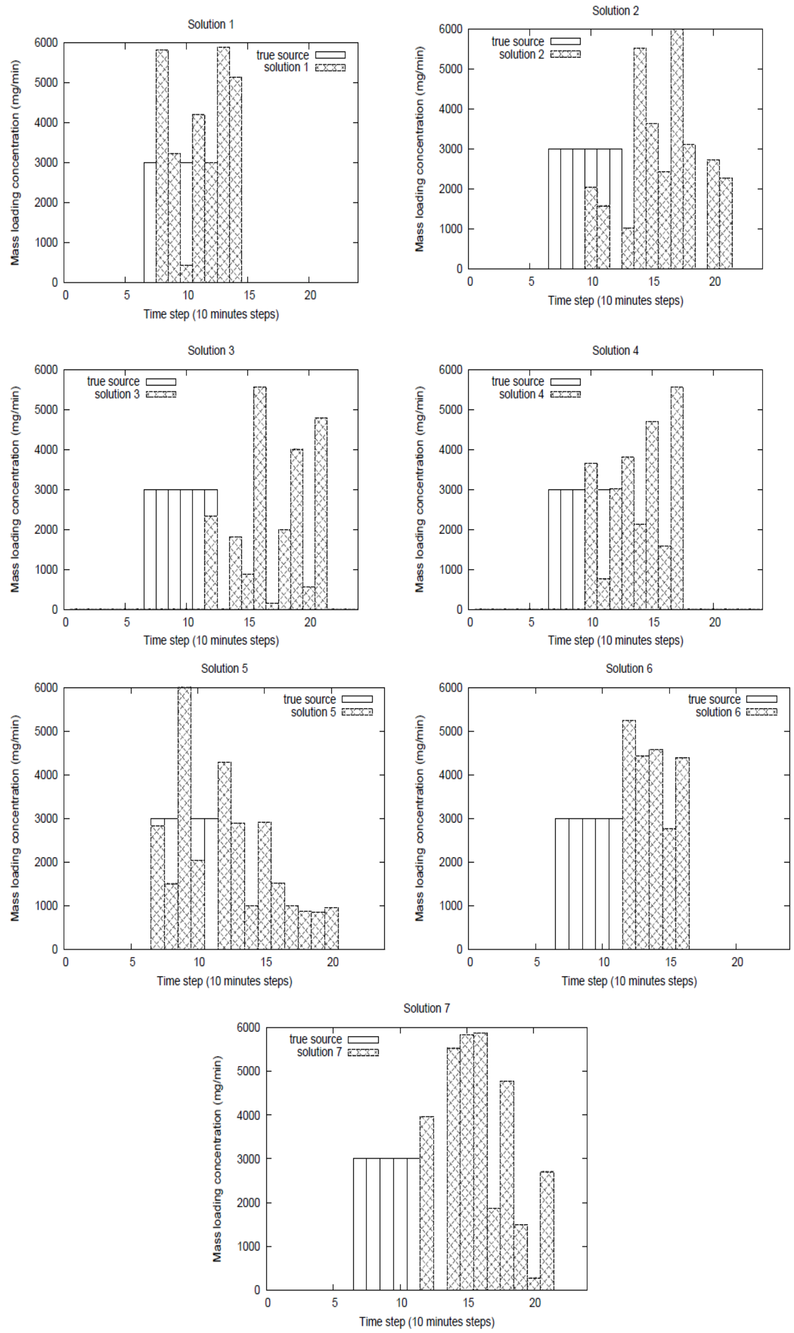



**Figure 3.8 Injection profiles for seven solutions of Scenario 4**

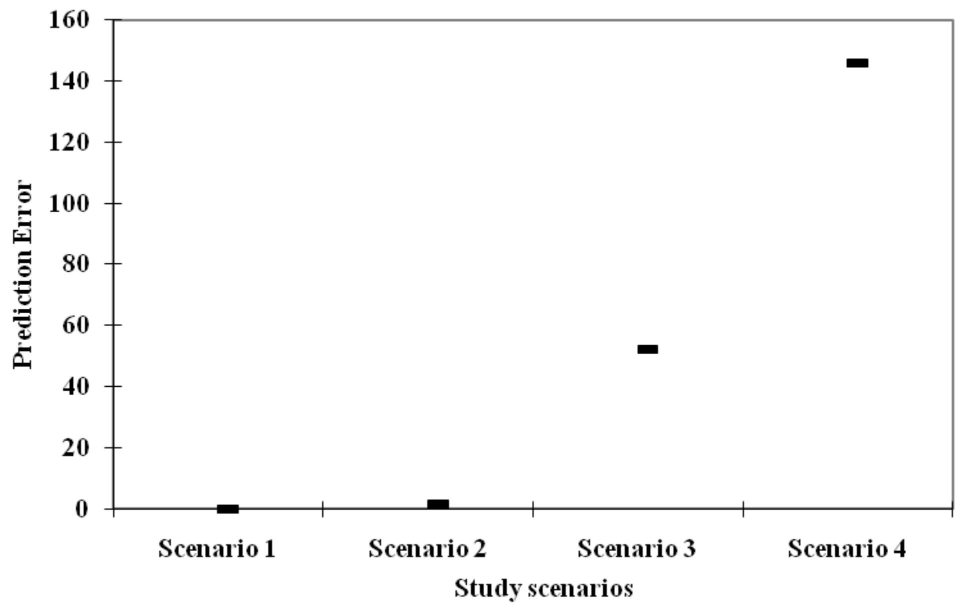



**Figure 3.9 Average prediction error for the best-fit solutions for each scenario over 50 random trials**

### 3.4.5 Summary of results from the scenarios for Network 1

All results from the 50 trials for each scenario were combined to examine the performance of the ES-based search for the four scenarios for Network 1. Figure 3.9 shows the mean of the prediction error (i.e., number of mishits as estimated using Eqn. 1) of the best-fit solution from 50 trials of each scenario. Similar statistics for the alternative solutions are shown in Figure 3.10. As expected, higher average prediction errors are obtained as the degree of noise increases from Scenario 1 to Scenario 4. Although the average and standard deviation of the prediction error are high for the scenarios (3 and 4) with higher levels of noise, the set of solutions in all trials included the true source location. But these solutions were different in terms of other source characteristics, contaminant injection time and pattern. Identifying just the source location under such noisy conditions, however, is still highly useful in solving this complex problem.

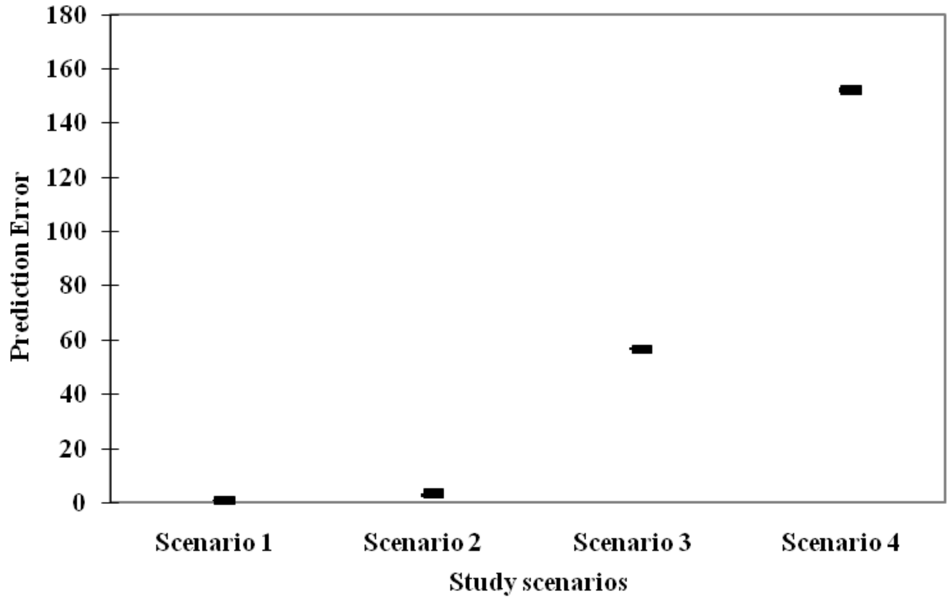



**Figure 3.10 Average prediction error for the alternative solutions for each scenario over 50 random trials**

It was observed (see Table 3.5) that the degree of non-uniqueness was low when accurate sensor data were available. With the increase in the error in the sensors, however, the degree of non-uniqueness increased, resulting in an increased number of alternative solutions (Figure 3.11) that include a larger set of potential source locations (Table 3.5). As error increases, the potential source locations identified are also farther away from the true location and are spread out in the network, as shown in Figure 3.12.

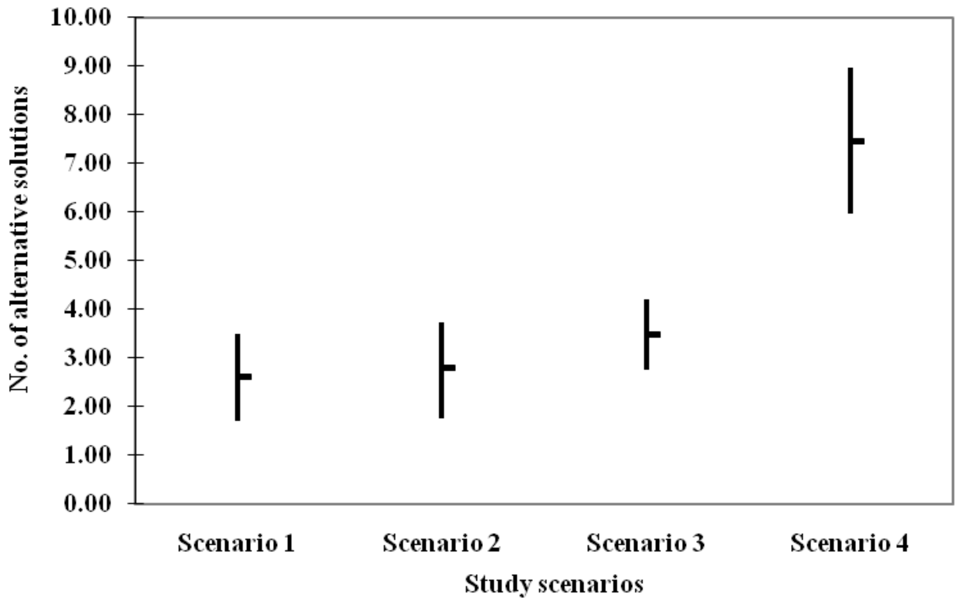



**Figure 3.11 Average (horizontal bar) and one-standard deviation (vertical bar) of the number of solutions obtained for each scenario over 50 random trials**

**Table 3.5 Potential locations for alternative solutions for different scenarios**

| Scenario | Potential source locations for alternative solutions |
| --- | --- |
| Scenario 1 | 12, 86 |
| Scenario 2 | 12, 86, 10 |
| Scenario 3 | 12, 86, 10, 18, 85 |
| Scenario 4 | 12, 86, 10, 18, 85, 1, 13, 17, 20 |

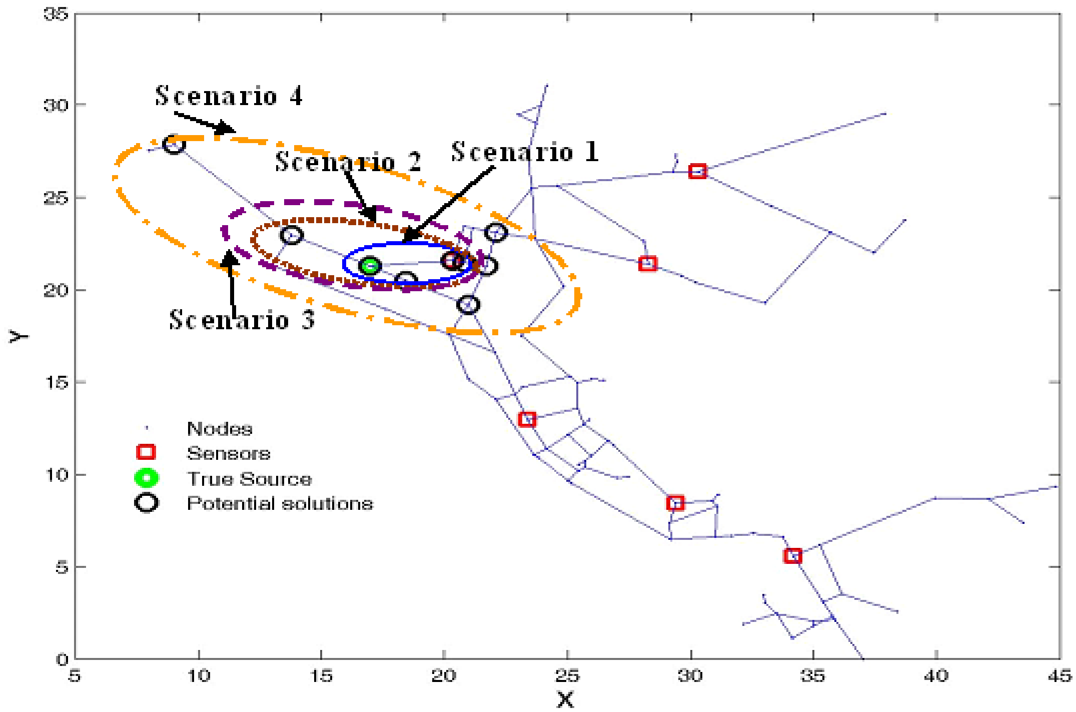



**Figure 3.12 Potential contamination source locations for Scenarios 1-4**

### 3.4.6 Scenario results for Micropolis network

A set of analyses was carried out in a similar fashion using the Micropolis network. The ES-based method described above was solved using a population size (μ) of 800 and 800 offspring (λ) for 100 generations. Compared to Network 1, the Micropolis network is larger in size, potentially allowing a higher number of non-unique solutions. Therefore, 50 subpopulations were used in the ES-based search allowing for up to 50 non-unique solutions. For scenarios 5 and 6, which assume perfectly working sensors, solutions within one mishit from the best solution were accepted as feasible solutions. However, due to noise in data for scenarios 7 and 8, solutions within the range of five mishits from the best solution were accepted as feasible. Summary statistics for all runs for the Micropolis network are based on 10 random trials.

#### 3.4.6.1 Scenario 5

Using the observations available from the perfectly working binary sensors, the ES-based search was used to identify the characteristics of the contaminant source. During each of the ten random trials solutions that perfectly matched the observation data identified the true source location, Node 787. The best solution in the first subpopulation was obtained with an average error of 1.6 mishits (standard deviation of 1.959). On average, 40.3 alternative solutions were obtained (standard deviation of 3.226) with an average error of 0.053 mishits (standard deviation 0.073). In addition to the correct source location at Node 787, alternative source locations were also identified at 13 other nodes in a close region around Node 787 (Figure 3.13).

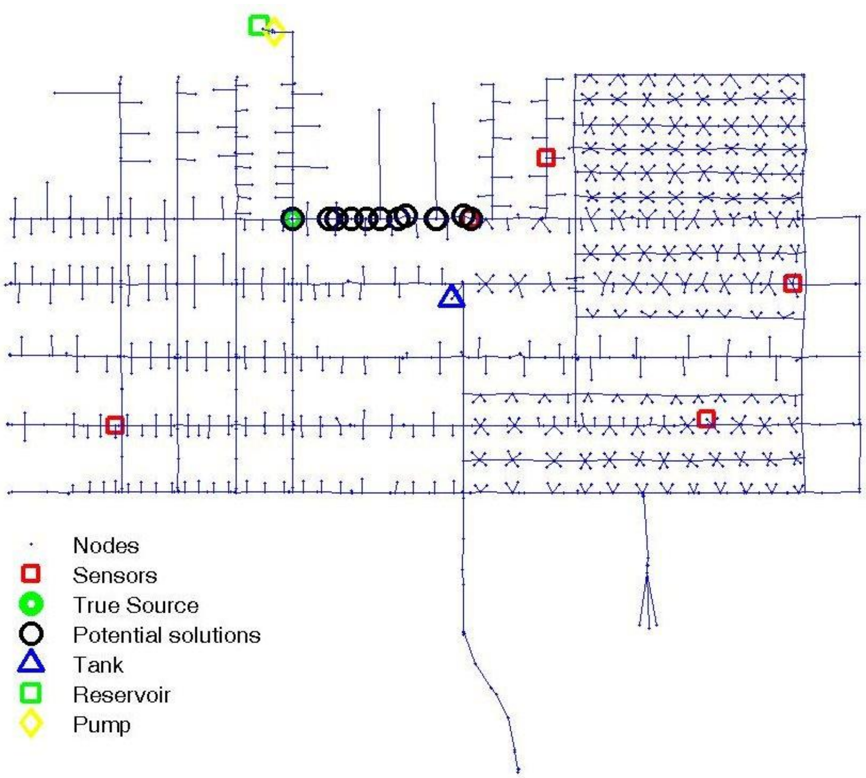



**Figure 3.13 Potential sorce locations for Scenario 5**

### 3.4.6.2 Scenario 6

Under this scenario also, for each set of ten random trials at least one of the feasible alternative solutions identified the true source location, Node 653. These solutions with the true source location had an average error of 0.568 mishits. Feasible solutions were also identified at seven other potential source locations in a close region. On average, 32.4 alternative solutions were identified (standard deviation 9.013) for this scenario. The best solution in the first subpopulation was identified with an average error of 3.70 mishits (standard deviation 3.318). Alternative solutions were identified with an average error of 0.786 mishits (standard deviation 0.064). Figure 3.14 shows the locations in the network where feasible sources were identified.

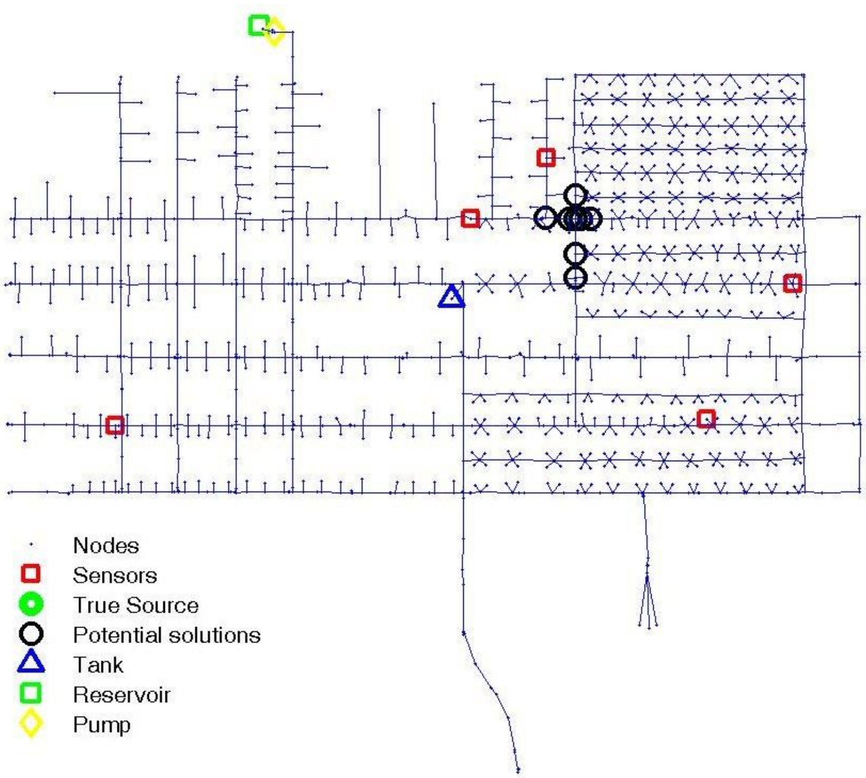



**Figure 3.14 Potential source locations for Scenario 6**

### 3.4.6.3 Scenario 7

The observation data in this scenario includes noise due to the random and intermittent failures of the binary water quality sensors. During each of the 10 random trials, at least one of the solutions in the set of feasible alternative solutions identified the true source location, Node 787. These solutions were obtained with an average error of 54.0 mishits. Feasible solutions were also identified at 14 additional potential source locations in a close region around Node 787 (Figure 3.15). In comparison, 13 additional source locations were obtained in scenario 5. The best solution in the first subpopulation was obtained with an average error of 55.0 mishits (standard deviation 2.049). On average, 41.10 alternative solutions were identified (standard deviation 2.426) with an average error of 54.534 mishits (standard deviation 0.074).

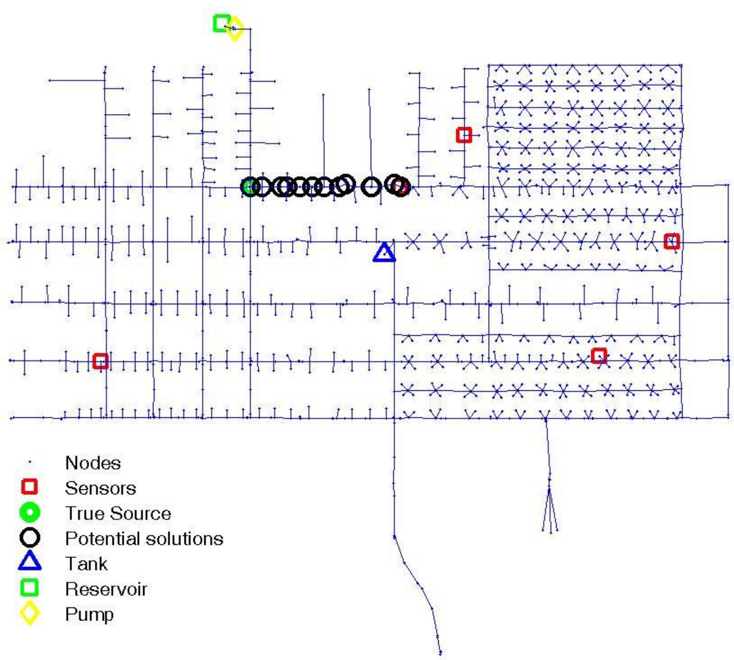



**Figure 3.15 Potential source locations for Scenario 7**

### 3.4.6.4 Scenario 8

The observation data available under the scenario also include noise due to random and intermittent sensor failures. During each of the ten trials, at least one of the feasible alternative solutions identified the true source location, Node 653, with an average error of 60.704 mishits. The best solution in the first subpopulation was identified with an average error of 61.7 mishits (standard deviation 1.345). On average, 30.5 alternative solutions (standard deviation 2.872) were obtained with an average error of 60.862 mishits (standard deviation 0.080). A high degree of non-uniqueness was observed in the solutions due to the presence of noise in the data, and the number of potential source locations increased from eight in scenario 6 to 14 (Figure 3.16). While most of the potential solutions were at locations close to the true source, a few solutions were found in a different region of network. These solutions matched the observations equally well because of the noise in the data.

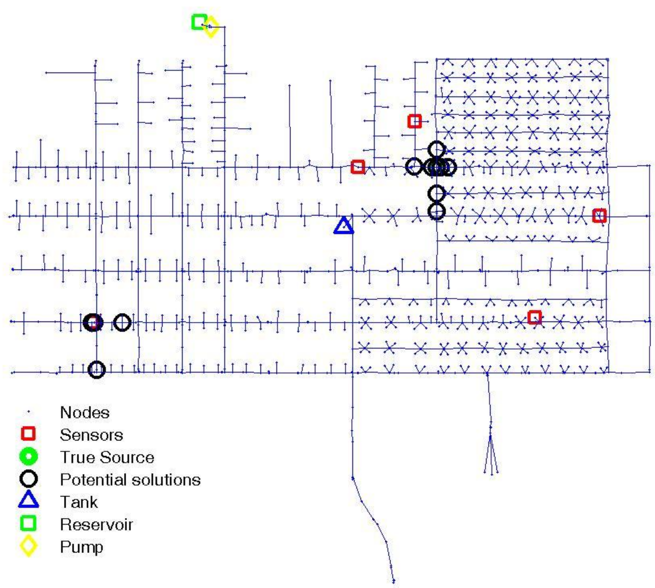



**Figure 3.16 Potential source locations for Scenario 8**

### 3.4.7 Summary of results from the scenarios for Micropolis network

Similar observations can be made about the results for Micropolis (scenarios 5-8) as for Network 1 (scenarios 1-4). The ES-based search provides a good estimate of the potential contaminant source characteristics. The presence of noise in the observation data adds to the difficulty of uniquely identifying the source characteristics. However, the method was designed to identify alternative solutions to the problem, and it was demonstrated to narrow down the source location to a region close to the actual source, even under conditions of limited data availability and uncertainties.

## 3.5 Summary and final remarks

An ES-based approach is presented for solving a contamination source identification problem in water distribution systems. The approach is demonstrated to identify a set of variable values that represent a contamination event with a uniform or non-uniform injection rates. Also, the problem specific variable encoding and a set of corresponding mutation operators are described for the ES-based search. In addition to providing a single best solution, the method also identifies good alternative solutions that are within an acceptable range of prediction error, and that are different in terms of decision variables.

Various scenarios were investigated to test the proposed approach under varying degrees of noise in the sensor information. The proposed approach performed robustly and consistently to identify the unknown contaminant source characteristics. Even though increased prediction errors were observed as the degree of noise in the sensor information increased, the method was able to identify a set of possible sources, including the true source, such that the different solutions yielded a comparable fit to the sensor observations. Despite the presence of non-uniqueness in the problem, the methodology did converge to a small set of possible source locations in close proximity to the true contaminant source location. The method was demonstrated through detailed case studies for two different networks. These results show promise in that even under relatively noisy conditions the method could potentially help water distribution network system managers resolve a contaminant source identification problem.

While the current work was carried out for a conservative contaminant with a single contaminant source, the methodology can potentially be extended for considering non-conservative chemical or biological contaminants and for identifying multiple contaminant sources. Several investigations are underway to extend the methodology to address these issues. The methodology is also being extended for addressing contamination scenarios involving reactive contaminants (Kumar et al. 2008; 2009). The goal of these investigations is to provide a robust set of tools to address the source identification problem for a wide range of water distribution contamination problem.

## Acknowledgements

This work was supported by the National Science Foundation (NSF) under Grant No. CMS-0540316 from the Dynamic Data Driven Analysis System (DDDAS) program.

## References

Brumbelow, K., Torres, J., Guikema, S., Bristow, E. and Kanta, L. (2007). "Virtual Cities for Water Distribution and Infrastructure System Research." World Environmental and Water Resources Congress, *2007*, Tampa, FL.

Dandy, G. C., Simpson, A. R., and Murphy, L. J. (1996). "An Improved Genetic Algorithm for Pipe Network Optimization." Water Resources Research, 32(2), 449-458.

Kumar, J., Zechman, E.M., Brill, E.D., Mahinthakumar, G., Ranjithan, S., and Uber, J. (2007). "Evaluation of Non-Uniqueness in Contaminant Source Characterization Based on Sensors with Event Detection Methods." World Environmental and Water Resources Congress 2007, Tampa, FL.

Kumar, J., Brill, E. D., Mahinthakumar, G., Uber, J., and Ranjithan, S. R. (2008). "Source identification for contamination events involving reactive contaminant." World Environmental and Water Resources Congress 2008, Honolulu, Hawaii.

Kumar, J., Brill, E. D., Mahinthakumar, G., and Ranjithan, S. R. (2009). "Characterizing reactive contaminant sources in water distribution system." World Environmental and Water Resources Congress 2009, Kansas City, Missouri.

Laird, Carl D., Biegler, Lorenz T., van Boleman Waanders, Bart G., Barlett, and Roscoe A. (2005). "Contamination source determination for water networks." Journal of Water Res. Planning and Management, 131(2), 125-134.

Preis, A., and Ostfeld, A. (2006). "Contamination source characterization in Water Systems: A Hybrid Model Trees-Linear Programming Scheme." Journal of Water Resources Planning and management, 132(4), 263-273.

Rossman, L. A. (2000). "EPANET 2 Users Manual." National Risk Management Research Laboratory, Environmental Protection Agency, Cincinnati.

Savic, D. A., and Walters, G. A. (1997). “Genetic Algorithms for Least-Cost Design of Water Distribution Networks.” Journal of Water Resources Planning and Management, 123(2), 67-77.

van Bloemen Waanders, B. G., Bartlett, R. A., Bigler, L. T., and Laird, C. D. (2003). “Nonlinear Programming Strategies for Source Detection of Municipal Water Networks.” World Environmental and Water ResourcesCongress 2007, Philadelphia, PA.

Zechman, E. M., and Ranjithan, S. (2004). “An evolutionary algorithm to generate alternatives (EAGA) for engineering optimization problems.” Engineering Optimization, 36(5), 539-553.

Zechman, E. M., Brill, E. D., Mahinthakumar, G., Ranjithan, S., and Uber, J. (2006). “Addressing non-uniqueness in a water distribution contamination source identification problem.” Proceedings of 8th Annual Water Distr. Sys. Anal. Symposium, Cincinnati, OH.

# CHAPTER 4. Characterizing reactive contaminant sources in a water distribution system

## 4.1 Introduction

Water distribution systems (WDSs) are vital for the supply of safe drinking water to the public. They are prone to accidental or intentional contamination that can endanger the public health. A robust drinking water monitoring program should be able to provide early indication of contamination and help minimize the public health consequences (USEPA 2005). Also, an early warning system should be able to reliably identify chemical, biological, or radioactive contamination events in source water or distribution systems in time to allow an effective local response to reduce or avoid the potential adverse impacts (Brosnan 1999).

Water distribution system contamination source identification problem has traditionally been approached as an inverse problem with the source characteristics as the variables to be identified using water quality observation data from sensors installed in the network. van Bloeman Waanders et al. (2003) used a standard successive quadratic programming approach. An origin-tracking algorithm based on a Lagrangian non-linear programming approach was proposed by Laird et al. (2005). Laird et al. (2006) proposed a mixed-integer programming-based method for the identification of unique injection scenarios. Zechman et al. (2006) studied the issues related to non-uniqueness in source identification problem, which arises due to the limited availability of the observed water quality data from the sensors in the network. Most of the studies for contamination source identification in WDS reported in the literature assume the pollutants to be conservative and transported through the network by the bulk velocity with mixing at the junctions. Availability of contaminant-specific probes to sense each contaminant is also assumed. In a realistic scenario, the contaminant introduced in the system can be a reactive chemical or biological species that may react with chemical components present in the bulk water, or attach to the pipe walls. In addition, sensors installed in a system might not be capable of detecting and reporting the presence of every possible reactive pollutant. This situation poses a challenge when detecting a contamination event in a system and characterizing the source in case of a contamination event.

This paper presents a study of WDS contamination source identification problems involving reactive contaminants. A methodology has been developed to characterize, without the need for contaminant-specific probes, a contamination event involving reactive contaminants. This method uses available water quality data and a simulation-optimization method for identification of the source characteristics. The issue of non-uniqueness in the problem has also been addressed when solving this problem using the proposed method.

## 4.2 Reactive contaminants

A reactive contaminant injected into a WDS would be transported by the flow in the pipe network and would react with the substances present in the water. Chlorine is a very commonly used disinfectant in most municipal water distribution systems. Doses of free chlorine are injected in the source water and at other booster injection nodes to maintain a minimum level of chlorine concentration required to provide the necessary disinfection capability in the water distribution system. Chlorine concentration is routinely monitored in every water distribution system. This study uses the chlorine measurements as surrogate to detect a contamination event and identify the contaminant source characteristics.

When a contaminant that is reactive with the chlorine enters the water distribution system, the chlorine concentration in the system would be affected according to the contaminant concentration present. Anomaly in the chlorine concentration in the system can be statistically analyzed to issue a contamination event alarm (McKenna et al. 2007) and can be used to identify the source characteristics (Kumar et al. 2008). The kinetics of reaction between the contaminant and chlorine in the water remain unknown, however, during an event. Flow and hydraulic conditions in a water distribution network are highly dynamic, introducing additional uncertainties about the interaction of the contaminant with chlorine.

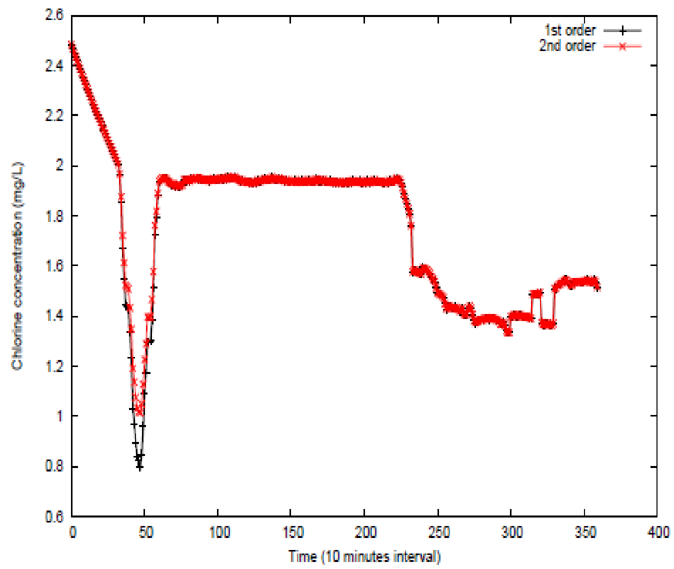



**Figure 4.1 Observed chlorine concentrations at a sensor under 1st/2nd order reaction conditions**

Kumar et al. (2008) studied the effect of the order-of-reaction between arsenic and chlorine on the chlorine concentration level in the system. First- and second-order reaction kinetics were studied. As shown in Figure 4.1, a different response in the chlorine concentrations in the system is observed depending upon the order of the reaction between arsenic and chlorine. The order of the reaction is unknown, however, during an event as the type and reactive properties of the contaminant are unknown. This study extends the previous work by Kumar et al. 2008, and develops a methodology for identifying source characteristics for reactive contaminants using chlorine measurements under uncertain conditions.

Contamination of water by arsenic (As) is considered for the illustrative case studies in this paper. Chlorine is assumed to be present in the form of monochloramine residuals. Arsenic may react with monochloramine leading to the decay of monocloramine as follows.

$$As^{+3} + NH_2Cl \rightarrow AsCl_3 + 3NH_2^{+} \qquad (1)$$

The reaction system can be solved using the following differential equations:

$$\frac{dAs^{+3}}{dt} = -K_a\,(As^{+3})^x\;(NH_2Cl)^y \qquad (2)$$

$$\frac{dNH_2Cl}{dt} = -K_b(As^{+3})^x(NH_2Cl)^y - K_c(NH_2Cl)^z \qquad (3)$$

where $K_a$ is the decay rate of arsenic ($As^{+3}$), $K_b$ is the decay rate of monochloramine when arsenic reacts with chlorine in water, $K_c$ is the self decay rate of chlorine, and $x$, $y$, and z are the order of reaction of arsenic, monochloramine and self-decay of monochloramine, respectively, in the assumed reaction system. Self-decay of monochloramine was assumed to be of first-order in the current analysis while the reaction between arsenic and monochloramine was considered to be under first and second order conditions.

## 4.3 Formulation of the optimization problem

Anomaly in the chlorine concentration levels in the system due to contaminant injection can be used for the characterization of contamination source. An optimization problem can formulated to find the sources characteristics to match the observed chlorine measurements as accurately as possible. The objective function for the optimization problem can be expressed mathematically as following:

$$\text{Minimize Error} = \sum_{i=1}^{N_S}\sum_{t=1}^{T_S} C_{i,t}^{s}(L,T_0,C_0) - C_{i,t}^{a}{}^{2} \qquad (4)$$

The objective function represents the sum of squared deviations in the predicted chlorine concentration over the chlorine observations at all the sensors over a monitoring period. $C_{i,t}^{a}$ is the actual chlorine concentration at sensor $i$ at time $t$, $C_{i,t}^{s}(L,T_0,C_0)$ is the simulated chlorine concentration at sensor $i$ at time $t$ due to a predicted source characteristics. $T_S$ is the total time over which the flows and concentrations in the network are being simulated, and $N_S$ is the total number of chlorine monitoring locations in the distribution system. The simulated chlorine concentrations are a function of the location of the source ($L$), start time of contaminant injection ($T_0$) and contaminant injection pattern ($C_0$). The goal of the optimization problem is to identify the source characteristics (represented by the node location ($L$) where the source was introduced, the start time of the contamination ($T_0$), and the contaminant injection pattern ($C_0$)) that minimize the prediction error by closely matching the predicted time series of anomalous drop in chlorine concentration to a set of chlorine concentration observations in the network. In general, the order of the reaction kinetics between chlorine and the contaminant is unknown. Instead of posing this kinetic parameter as a variable in the optimization problem, separate optimization problems were solved assuming 1$^{st}$ order and 2$^{nd}$ order kinetics. This was done primarily to make the simulations of the WDS flow and reactive transport using EPANET-MSX easier and to reduce the complexity of the optimization problem.

## 4.4 Methodology

The problem of source identification in water distribution system can be posed as inverse problem and solved using a simulation-optimization method. Evolutionary algorithms (EAs) (Holland 1975) are a class of heuristic methods that provide a global search mechanism to identify efficiently near-optimal solutions for large optimization problems. Dandy et al. (1996) and Savic and Walters (1997) used genetic algorithms for the optimal design of water distribution systems. Vitkosvsky et al. (2000) and Lingireddy and Ormsbee (2002) solved using EAs the inverse problem of water distribution network calibration. In the present study, an EA-based procedure is employed for the source identification problem.

Evolution strategies (ES) (Schwefel, 1995) is an EA-based search method that has been explored as a solution approach for the source identification problem. ES works with a population of solutions. The first generation starts with a random population of μ individuals. A set of λ new solutions is created in each generation using a random mutation process. From the combined set of parent and offspring solutions (($\mu + \lambda$)-selection) or from the set of offspring solutions alone (($\mu, \lambda$)-selection), a set of $\mu$ individuals are selected for the next generation. Zechman and Ranjithan (2009) used an ES-based search approach to solve the source identification problem in water distribution systems. The inverse problem was defined in terms of continuous concentration observations assuming the availability of perfect sensors for the specific contaminant. In the study presented in this paper, an ES based methodology will be used for addressing the source identification problem using chlorine observations.

## 4.5 Illustrative case studies

### 4.5.1 The example water distribution networks

The proposed study is demonstrated through illustrative case studies involving two test networks. First is an example network (Network 1, Figure 4.2) that is included with the standard EPANET distribution. This network consists of 97 nodes including two sources, three tanks and 117 pipes, and the water distribution system is simulated using EPANET. The second network used in the study is the virtual city of Micropolis (

Figure 4.3) that was developed by Brumbelow et al. (2007). It consists of 1236 nodes, 575 mains, 486 service and hydrant connections, 197 valves, 28 hydrants, 8 pumps, 2 reservoirs, and 1 tank. Multiple-species extension toolkit (MSX) for EPANET, which provides the capability to model the chemical interaction among multiple reactive species, was used for simulating the contamination scenarios in this study.

For the scenarios analyzed (described below), both the networks (Network 1 and Micropolis network) were simulated for a period of 24 hours. Hourly hydraulic conditions in network were simulated and were assumed to remain constant during each hour, while the water quality and transport processes were simulated every 5 minutes. A sample network of sensors/monitoring stations was chosen for each network to uniformly cover the entire network. Water quality observations were recorded every 10 minutes at all the monitoring stations.

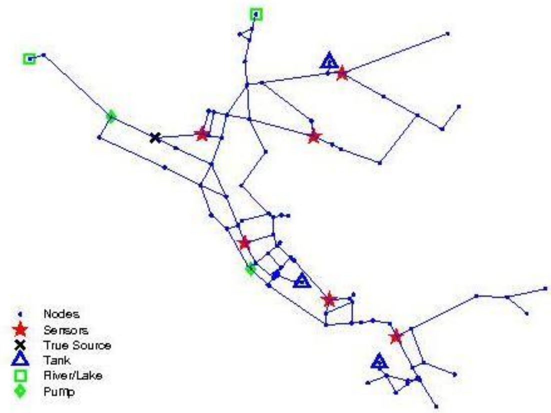



**Figure 4.2 Network 1 water distribution system**

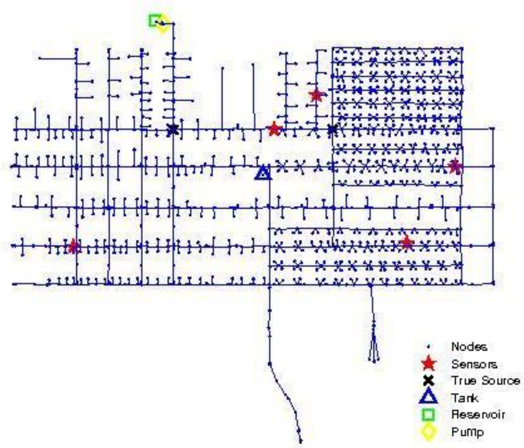



**Figure 4.3 Micropolis water distribution system**

### 4.5.2 Scenarios analyzed

Several scenarios were designed for the two example WDS networks to test the ability of the proposed methodology for identification of contaminant source characteristics. These scenarios were investigated for the contamination of the system from a single source of arsenic pollutant. As the kinetics of the reaction between arsenic and chlorine in the system is assumed to be unknown during each contamination event, one of the goals of the source characterization problem is to address this uncertainty. In this study, arsenic was assumed to undergo either a first order or a second order reaction with monochloramine in the system.

#### 4.5.2.1 Scenarios for Network 1

Since the kinetics of the reaction between the arsenic and chlorine in the system is unknown and can be either of first or second order, the optimization problem for source identification was solved twice; once assuming first order reaction kinetics and another instance with second order assumptions. In Scenarios 1 and 2, the actual arsenic contamination was set to be first order and second order, respectively. When solving each scenario, the optimization model was solved once assuming first order conditions and again assuming second order conditions. Six water quality sensors, distributed uniformly in the network, were assumed to be available in both scenarios.

The accuracy and non-uniqueness of the solutions to the source identification problem depend on the amount of available water quality observation data (Kumar et al. 2007). Scenarios 3-6 were designed to test the performance of the source identification methodology under different conditions of data availability. Similar to Scenarios 1 and 2, Scenarios 3 and 4 were solved using data from three water quality sensors, and Scenarios 5 and 6 were solved using data from a single sensor. These scenarios are summarized in Table 4.1.

#### 4.5.2.2 Scenarios for the Micropolis network

To test the robustness and wider applicability of the proposed approach, the methodology was applied to several scenarios involving the Micropolis network. Five water quality sensors were assumed to be available in the network. In Scenarios 7 and 8, the true contaminant source was assumed to be at node 653, and it was assumed to be at node 787 for Scenarios 9 and 10. These scenarios also are summarized in Table 4.1. For these scenarios, the source identification solution approach was applied assuming that the order of the reaction kinetics was known.

**Table 4.1 Scenarios analyzed for the two example networks**

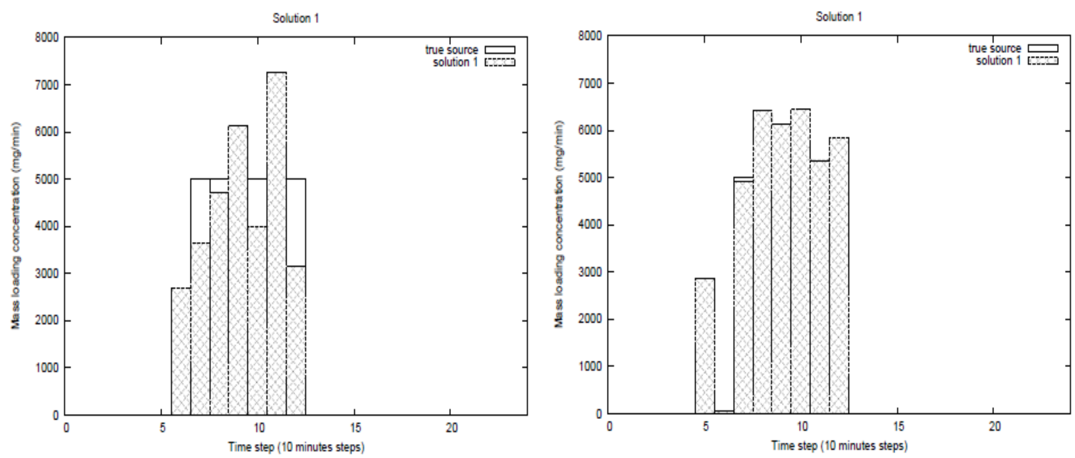
| Scenario | Network | No. of sensors | True injection source | | Order of reaction for solution |
|---|---|---|---|---|---|
| | | | Location | Order of reaction | |
| 1 | Network 1 | 6 | 12 | 1st order | 1st order / 2nd order |
| 2 | Network 1 | 6 | 12 | 2nd order | 1st order / 2nd order |
| 3 | Network 1 | 3 | 12 | 1st order | 1st order / 2nd order |
| 4 | Network 1 | 3 | 12 | 2nd order | 1st order / 2nd order |
| 5 | Network 1 | 1 | 12 | 1st order | 1st order / 2nd order |
| 6 | Network 1 | 1 | 12 | 2nd order | 1st order / 2nd order |
| 7 | Micropolis | 5 | 653 | 1st order | 1st order |
| 8 | Micropolis | 5 | 653 | 2nd order | 2nd order |
| 9 | Micropolis | 5 | 787 | 1st order | 1st order |
| 10 | Micropolis | 5 | 787 | 2nd order | 2nd order |

## 4.5.3 Results for the Network 1 scenarios

### 4.5.3.1 Scenario 1

An arsenic injection event at node 12 in Network 1 with first order reaction kinetics between chlorine and arsenic was considered for Scenarios 1a and 1b. As the order of the reaction kinetics was considered unknown, Scenario 1a assumed first order conditions and Scenario 1b assumed second order conditions. In the ten random trials carried out for this scenario, all the solutions obtained were found to be at the correct contaminant source location (Node 12). The solutions at Node 12 were found with an average error (mean squared error) of 0.334 $mg^2/L^2$ and a standard deviation of 0.202 $mg^2/L^2$. Figure 4.4.a shows the arsenic injection pattern, with an injection start time of 6 (close to the true start time of 7), for a representative solution for Scenario 1.

In the ten random trials conducted for Scenario 1b, all the solutions were found to be at the correct source location (Node 12) even though the order of the reaction was different (second order) from the one that was used (first order) to model the true source. The average error of the solutions, however, was higher, with an average error (mean squared error) of 0.455 $mg^2/L^2$ and a standard deviation of 0.195 $mg^2/L^2$). Figure 4.4.b shows the arsenic injection pattern for a representative solution.



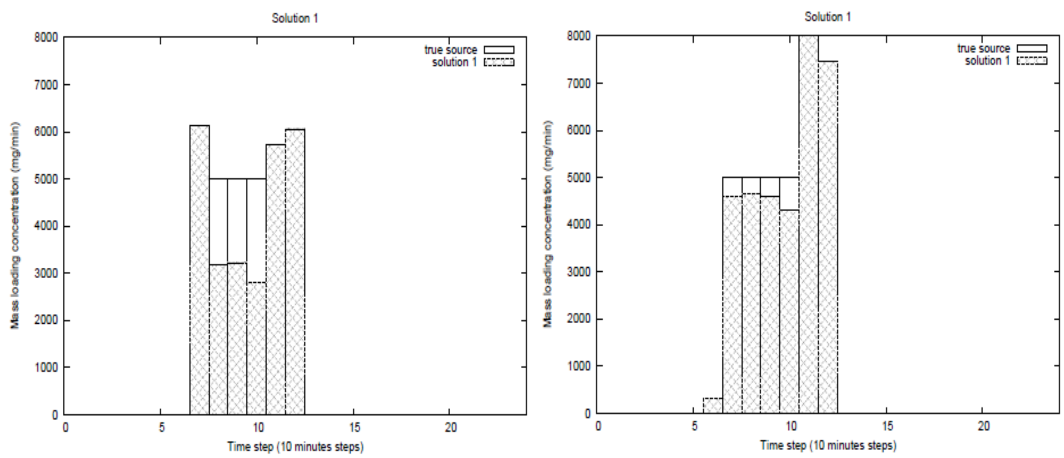

(a) First order reaction kinetics (b) Second order reaction kinetics

**Figure 4.4 Arsenic injection pattern for a typical solution for Scenario 1**

### 4.5.3.2 Scenario 2

In Scenarios 2a and 2b, arsenic injection at Node 12 in the network with second order reaction kinetics between chlorine and arsenic was considered. The order of the reaction kinetics was assumed to be first order in Scenario 2a. In ten random trials conducted for this scenario, all solutions obtained were found to be located at the true source location (Node 12). The solutions were obtained with an average error (mean squared error) of 0.596 $mg^2/L^2$ and a standard deviation of 0.182 $mg^2/L^2$. Figure 4.5.a shows the arsenic injection pattern, with an injection start time of 7 (same as the true start time), for a representative solution for Scenario 2.

The order of the reaction kinetics was assumed to be second order in Scenario 2b. The average error (mean squared error) of solutions was 0.327 $mg^2/L^2$ with a standard deviation of 0.156 $mg^2/L^2$. The average prediction error for solutions for Scenario 2b, in which reaction kinetics was assumed to be the same as that used to model the true source, was found on average to be lower than that of Scenario 2a. Figure 4.5.b shows the arsenic injection pattern for a representative solution for Scenario 2b.



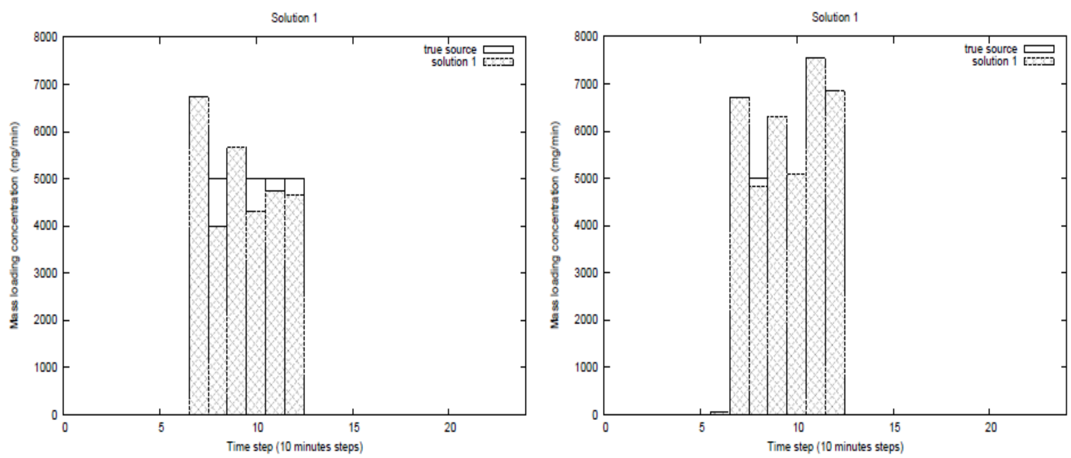

(a) First order reaction kinetics (b) Second order reaction kinetics

**Figure 4.5 Arsenic injection pattern for a typical solution for Scenario 2**

### 4.5.3.3 Scenario 3

Scenarios 3a and 3b were designed similar to Scenarios 1a and 1b with only three water quality sensors in the network. The arsenic injection at Node 12 with first order reaction kinetics between arsenic and chlorine was considered for these scenarios. Scenarios 3a and 3b assumed first order and second order reaction kinetics, respectively. All solutions obtained for ten random trials for Scenario 3a were found to be at the actual source location (Node 12). The solutions were obtained with an average error (mean squared error) of 0.079 $mg^2/L^2$ and a standard deviation of 0.060 $mg^2/L^2$. Figure 4.6.a shows the arsenic injection pattern for a representative solution for this scenario.

All the solutions obtained for the ten random trials for Scenario 3b were found to be at the true source location (Node 12). The solutions were obtained with an average error of 0.144 $mg^2/L^2$ and a standard deviation of 0.111 $mg^2/L^2$. The average prediction error for solutions for Scenario 3a, in which reaction kinetics was assumed to be the same as that used to model the true source, was found on average to be lower than that of Scenario 3b. The arsenic injection pattern for a representative solution for Scenario 3b is shown in Figure 4.6.b.



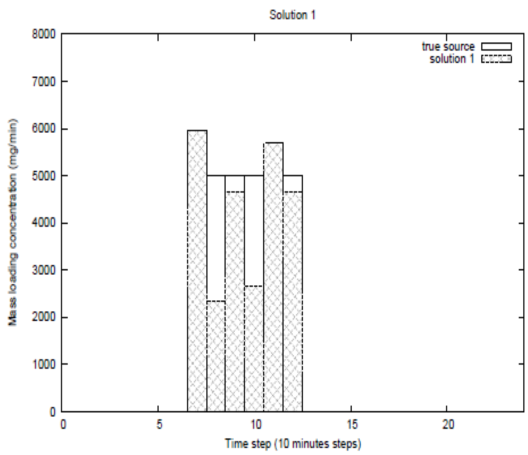

(a) First order reaction kinetics (b) Second order reaction kinetics

**Figure 4.6 Arsenic injection pattern for a typical solution for Scenario 3**

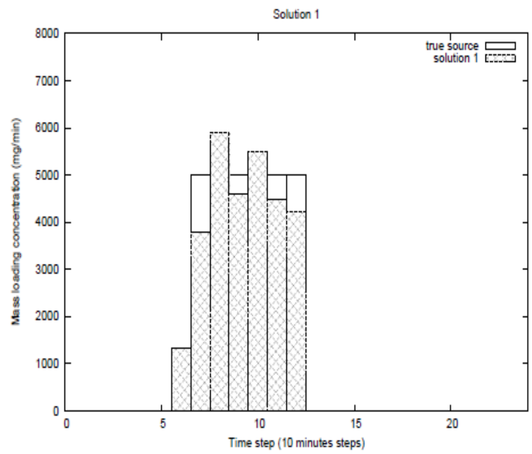
### 4.5.3.4 Scenario 4

Scenarios 4a and 4b are similar to Scenarios 2a and 2b with only three water quality sensors in the network. A contamination scenario with arsenic injection at Node 12 and was assumed with a second order reaction between arsenic and chlorine in the system. Scenarios 4a and 4b assumed first order and second order reaction kinetics, respectively. All solutions obtained for ten random trials for Scenario 4a were found to be at the actual source location (Node 12). These solutions were obtained with an average error (sum squared error) of 0.059 $mg^2/L^2$ and standard deviation of 0.033 $mg^2/L^2$. Figure 4.7.a shows the arsenic injection pattern for a representative solution under this scenario.

All solutions obtained for ten trials were found at the actual source location (Node 12). The solutions for Scenario 4b were obtained with an average error (sum squared error) of 0.036 $mg^2/L^2$ and standard deviation of 0.050 $mg^2/L^2$. Figure 4.7.b shows the arsenic injection pattern for a representative solution for this scenario. The average prediction error for solutions for Scenario 4b, in which reaction kinetics was assumed to be the same as that used to model the true source, was found on average to be lower than that of Scenario 4a.




(a) First order reaction kinetics



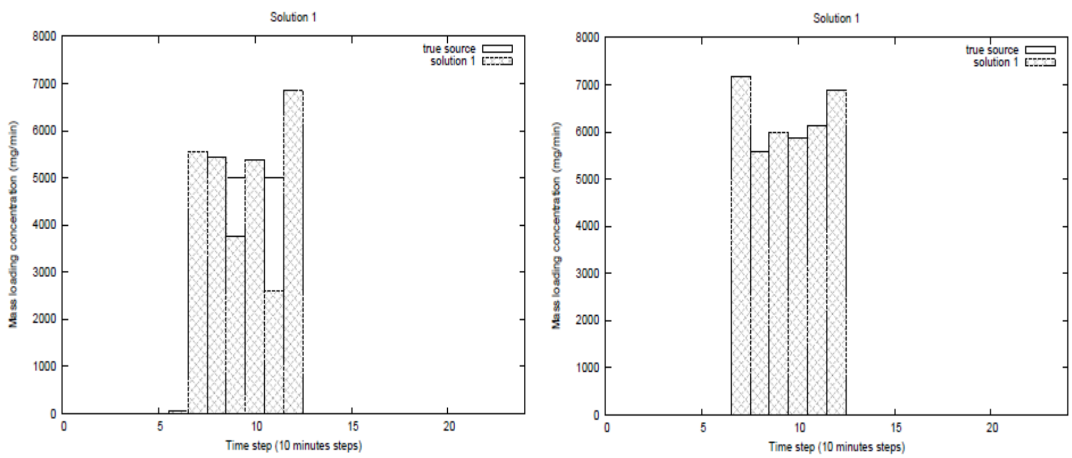

(b) Second order reaction kinetics

**Figure 4.7 Arsenic injection pattern for a typical solution for Scenario 4**

#### 4.5.3.5 Scenario 5

Scenario 5a and 5b are similar to the Scenarios 1a, 1b and 3a, 3b but with a single water quality sensor in the network. A contamination scenario with arsenic injection at Node 12 with first order reaction kinetics between arsenic and chlorine was considered for these scenarios. Scenarios 5 and 5b assumed first order and second order reaction kinetics, respectively. Even with a single sensor in the network, the all solutions obtained for the ten random trials of Scenario 5a were found to be at the actual source location (Node 12). The average prediction error (mean squared error) of the obtained solutions was 0.060 $mg^2/L^2$ and standard deviation of 0.039 mg$^2$/L$^2$. Figure 4.8.a shows the arsenic injection pattern for a representative solution under this scenario.

All solutions obtained for the ten random trials of Scenario 5b were found at the true location (node 12). Solutions for Scenario 5b were obtained with an average prediction error (mean squared error) of 0.085 mg$^2$/L$^2$ and standard deviation of 0.147 mg$^2$/L$^2$. Figure 4.8.b shows the arsenic injection pattern for a representative solution under this scenario. The average prediction error for solutions for Scenario 5a, in which reaction kinetics was assumed to be the same as that used to model the true source, was found on average to be lower than that of Scenario 5b.



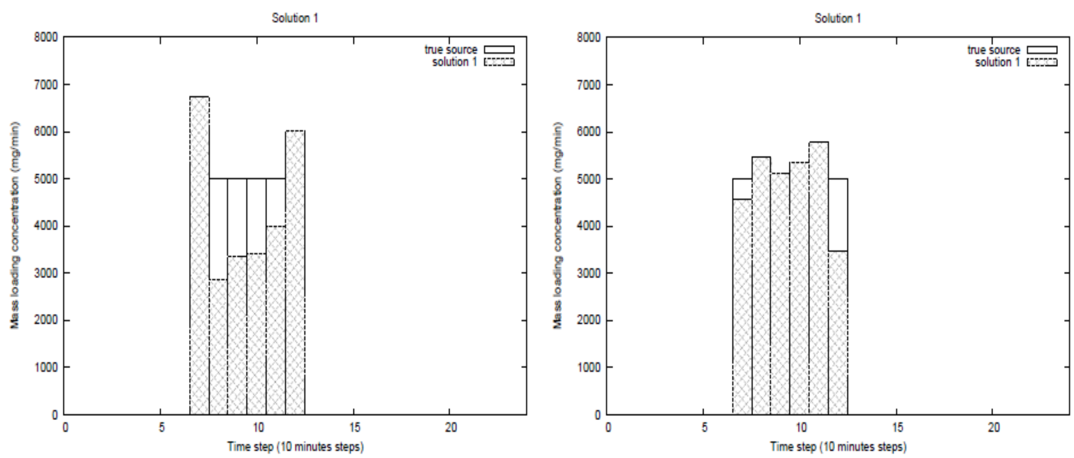

(a) First order reaction kinetics (b) Second order reaction kinetics

**Figure 4.8 Arsenic injection pattern for a typical solution for Scenario 5**

#### 4.5.3.6 Scenario 6

Scenario 6a and 6b are similar to the Scenarios 2a, 2b and 4a, 4b with only one water quality sensor present in the network. A contamination scenario with arsenic injection at Node 12 with second order reaction kinetics between chlorine and arsenic was considered for these scenarios. Scenarios 6a and 6b assumed first order and second order reaction kinetics, respectively. All solutions obtained for the ten random trials for Scenario 6a were found to be at the actual source location (Node 12). The solutions were obtained with an average prediction error (mean squared error) of 0.048 $mg^2/L^2$ and standard deviation of 0.034 $mg^2/L^2$. Figure 4.9.a shows the arsenic injection pattern for a representative solution under this scenario.

All solutions obtained for ten random trials for Scenario 6b were found to be at the actual source location (Node 12). The solutions were obtained with an average prediction error (mean squared error) of 0.037 $mg^2/L^2$ and standard deviation of 0.023 $mg^2/L^2$. Figure 4.9.b shows the arsenic injection pattern for a representative solution under this scenario. The average prediction error for solutions for Scenario 6b, in which reaction kinetics was assumed to be the same as that used to model the true source, was found on average to be lower than that of Scenario 6a.




(a) First order reaction kinetics (b) Second order reaction kinetics

**Figure 4.9 Arsenic injection pattern for a typical solution for Scenario 6**

#### 4.5.3.7 Summary of results for Network 1 scenarios

Results from Scenario 1-6 show that the methodology was able to resolve the contaminant source location accurately in all instances. Start time and pattern of injection that were identified matched closely with the true values. As all the solutions identified the same source node, which was the true source node, no non-unique source location in the network was identified in any of the scenarios. Although the scenarios used varying numbers of water quality sensors in the network, the solutions consistently identified the true source location even when the amount of observations was minimal.

In all the scenarios, the average error (mean squared error) of the solutions obtained was observed to be higher if the order of reaction kinetics was assumed incorrectly. While this observation was consistent for Scenarios 1-6, the high standard deviation associated with each scenario limits the confidence with which the unknown reaction kinetics could be resolved. The proposed methodology is able to identify the possible source characteristics sufficiently accurately for different contamination scenarios.

### 4.5.4 Results for Micropolis network scenarios

#### 4.5.4.1 Scenario 7

In Scenario 7, arsenic injection at Node 653 in the Micropolis water distribution network was considered with first order reactions between arsenic and chlorine. Order of reaction kinetics was assumed to be known correctly for the source identification problem in this scenario. All the solutions obtained for the ten random trials for Scenario 7 were found to be at the actual source location of Node 653. The solutions were obtained with an average prediction error (mean squared error) of 3.0E-07 $mg^2/L^2$ and a standard deviation of 6.749E-07 $mg^2/L^2$. Figure 4.10 shows the injection pattern for a representative solution for this scenario.

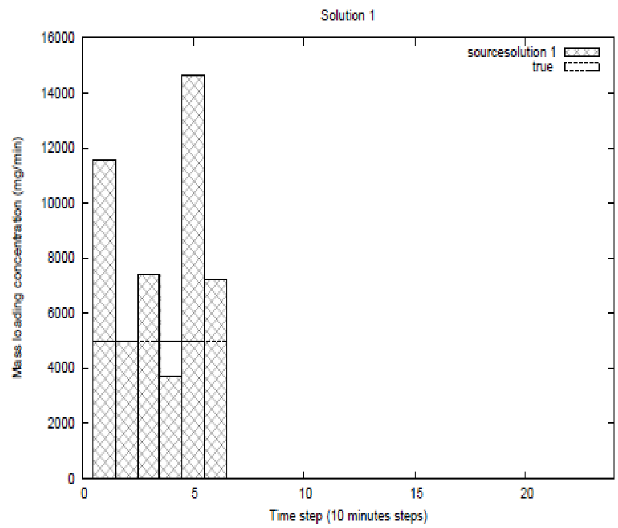



**Figure 4.10 Asenic injection pattern for a typical solution for Scenario 7**

### 4.5.4.2 Scenario 8

In Scenario 8, an arsenic injection at the Node 653 in the Micropolis network was considered with second order reactions between arsenic and chlorine. The order of reaction kinetics was again assumed to be known correctly when solving the source identification problem. All solutions obtained for the ten random trials for Scenario 8 were found to be at the actual source location Node 653. The solutions were obtained with an average prediction error (means squared error) of 0.0 $mg^2/L^2$ and standard deviation of 0.0 $mg^2/L^2$. Figure 4.11 shows the injection pattern for a representative solution under this scenario.

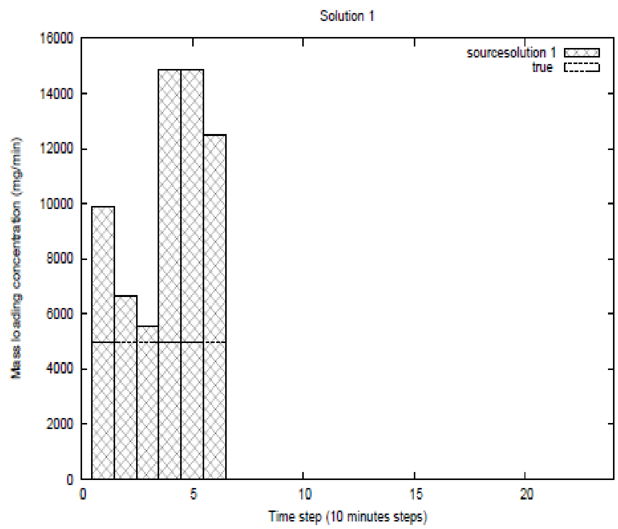



**Figure 4.11 Arsenic injection pattern for a typical solution for Scenario 8**

#### 4.5.4.3 Scenario 9

In Scenario 9, arsenic injection at the Node 787 in the Micropolis network was considered with first order reactions between arsenic and chlorine. The order of reaction kinetics between arsenic and chlorine was assumed to be known correctly when solving the source identification problem. All solutions obtained for the ten random trials were found to be located at the actual source location Node 787. The solutions were obtained with an average prediction error (mean squared error) of 1.177 $mg^2/L^2$ and a standard deviation 1.371 $mg^2/L^2$. Figure 4.12 shows the injection pattern for a representative solution under this scenario.

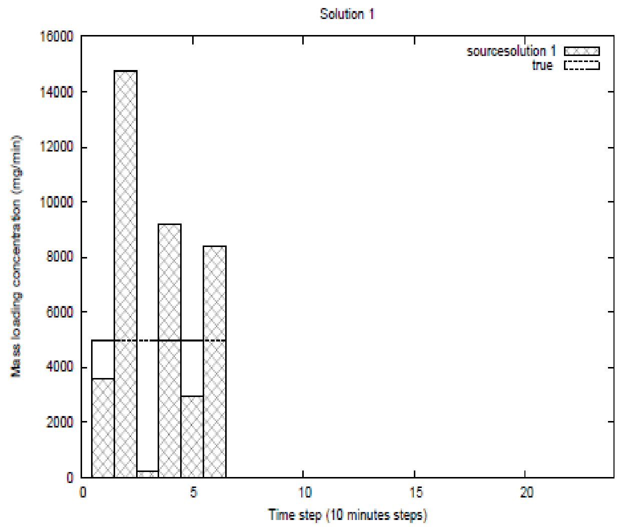



**Figure 4.12 Arsenic injection pattern for a typical solution for Scenario 9**

### 4.5.4.4 Scenario 10

In Scenario 10, arsenic injection at Node 787 in the Micropolis network was considered with second order reactions between arsenic and chlorine. The order of reaction was assumed to be known correctly. All the solutions obtained for the ten random trials for Scenario 10 were found to be at the actual source location Node 787. The solutions were identified with an average prediction error (mean squared error) of 3.644 $mg^2/L^2$ and a standard deviation of 3.621 $mg^2/L^2$. Figure 4.13 shows the injection pattern for a representative solution under this scenario.

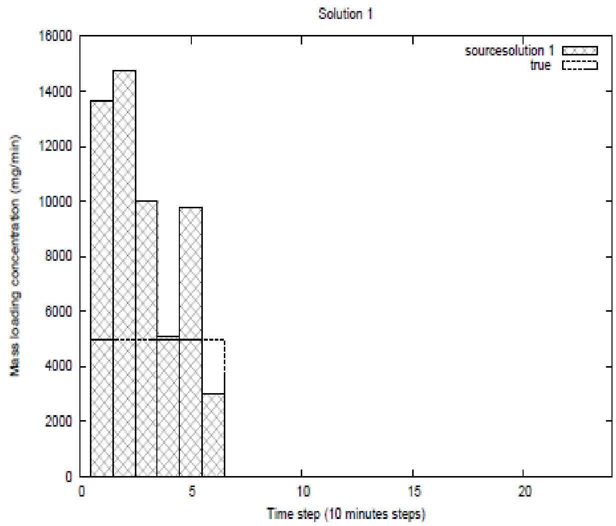



**Figure 4.13 Arsenic injection pattern for a typical solution for Scenario 10**

#### 4.5.4.5 Summary of results for Micropolis network scenarios

Scenarios 7-10 were designed to test the general applicability of the source identification methodology for a contamination event involving reactive contaminants. These scenarios involved contamination injections scenarios at two different locations in the Micropolis network while the observation data from the same set of five water quality sensors were used when solving the source identification problem. For all the instances, the source location was correctly identified with an injection start time and pattern close to that of actual contamination event. While a good agreement was obtained between simulated and actual sensor observations for Scenario 7 and 8 (actual source location Node 653), comparatively higher errors were obtained for solutions for Scenarios 9 and 10 (actual source location Node 787). This indicates that the accuracy of source identification problem largely depends upon the available data from the sensors. Provided a set of sensors in the network, the ability to identify a contaminant source will vary depending upon the location of the actual source and anomalies recorded by the installed water quality sensors. If a contamination event is not detected by the observations at the installed sensors, it will be impossible to identify the contamination source characteristics.

The results discussed above show that the source characterization method is able to identify the contaminant source characteristics accurately and robustly for two different networks. An array of scenarios was designed to test the effectiveness of the approach. While the method was able to narrow down to the correct source characteristics, the results also highlight the effect of uncertainties associated with the contamination scenario and reaction kinetics on the quality of the solutions to the source identification problem. If the available observation data is insufficient, it may not be possible to resolve issues such as unknown order of reaction kinetics. However, even under those conditions the solutions provide useful information to help the utility managers narrow down their efforts to a few potential source locations and characteristics. Such information would help in focusing necessary field operations to a smaller potential region that is suspected to contain the actual source and taking appropriate measures to mitigate the contaminant.

## 4.6 Summary and conclusions

This study describes the development and investigation of a simulation-optimization methodology to identify and characterize a contaminant source involving reactive chemicals. In this study, the unknown source characteristics are described by the source location, the start time of contamination, the contaminant injection patter and the order of reaction of the contaminant with chlorine present in the system. Any reactive contaminant injected into a system would interact with chlorine to alter its rate of decay. Anomalies in chlorine observations are used as a surrogate to characterize a contamination event and to identify the source characteristics. A simulation-optimization methodology is developed in this study to recreate the source characteristics during a contamination event based on its impact on the chlorine concentrations in the system. Source identification problem was studied under different data availability conditions. The study explores and addresses the issue of uncertainties about the reaction kinetics between contaminant and chlorine. The source characterization approach was applied to a variety of scenarios for two different water distributions networks to establish its accuracy and robustness.

This method was able to accurately identify under different conditions of data availability and uncertainties. Even under conditions of limited data availability and uncertainties, the solutions identified by the method was able to generate information potentially useful to utility mangers for carrying out ground operations for contamination containment and control operations.

While this study was conducted using arsenic as the reactive contaminant, the simulation-optimization approach is independent of the nature of contaminant being studied. Thus it can be extended to address a wide range of contamination scenarios involving reactive chemical, biological or radioactive contaminants. Efforts are underway to apply the methodology for a range of contaminants.

## References

Brosnan, Thomas M. (1999). “Early warning monitoring to detect hazardous events in water supplies.” International Life Sciences Institute-Risk Science Institute.

Brumbelow, K.; Torres, J.; Guikema, S.; Bristow, E. & Kanta, L. (2007). “Virtual Cities for Water Distribution and Infrastructure System Research.” World Environmental and Water Resources Congress, 2007, Tampa, FL.

Dandy, G. C., Simpson, A. R., and Murphy, L. J. (1996). “An Improved Genetic Algorithm for Pipe Network Optimization.” Water Resources Research, 32(2), 449-458.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Cambridge, MA.

Kumar, J., Zechman, E.M., Brill, E.D., Mahinthakumar, G., Ranjithan, S., Uber, J. (2007). “Evaluation of Non-Uniqueness in Contaminant Source Characterization Based on Sensors with Event Detection Methods.” World Environmental and Water Resources Congress 2007, Tampa, FL.

Kumar, J., Brill, E.D., Mahinthakumar, G., Ranjithan, S., Uber, J. (2008). “Source identification for contamination events involving reacting contaminants.” World Environmental and Water Resources Congress 2008, Honolulu, Hawaii.

Laird, Carl D., Biegler, Lorenz T., van Boleman Waanders, Bart G., Barlett, Roscoe A. (2005). “Contamination source determination for water networks.” Journal of Water Res. Planning and Management, 131(2).

Laird, Carl D., Biegler, Lorenz T., van Boleman Waanders, Bart G. (2006). “A Mixed-integer approach for obtaining unique solutions in source inversion of drinking water networks.” World Water Congress 2005, 173(47).

Lingireddy, S. and Ormsbee, L. (2002). “Hydraulic Network Calibration Using Genetic Optimization.” Civil Engineering and Environmental Systems, 19(1).

McKenna, Sean.A., Hart, David., Klise, Katherine., Cruz, Victoria., Wilson, Mark. (2007). “Event detection from water quality time series.” World Water and Environmental Congress 2007, Tampa, FL.

USEPA (2005). “Technologies and techniques for early warning systems to monitor and evaluate drinking water quality: A state-of-art review.”

Savic, D. A. and Walters, G. A. (1997). “Genetic Algorithms for Least-Cost Design of Water Distribution Networks.” Journal of Water Resources Planning and Management, 123(2), 67-77.

Schwefel, H.-P. (1995) Evolution and Optimum Seeking, Wiley & Sons, New York.

van Bloemen Waanders, B. G., Bartlett, R. A., Bigler, L. T., and Laird, C. D. (2003). “Nonlinear Programming Strategies for Source Detection of Municipal Water Networks.” World Water and Environmental Congress 2003, Philadelphia, PA.

Vitkosvsky, J. P., Simpson, A. R., and Lambert, M. F. (2000). “Leak Detection and Calibration using Transients and Genetic Algorithms.” Journal of Water Resources Planning and Management, 126(4), 262-265.

Zechman, Emily M., Brill, E. Downey., Mahinthakumar, G., Ranjithan, S., Uber, James. (2006). “Addressing non-uniqueness in a water distribution contamination source identification problem.” Proceedings of 8th Annual Water Distr. Sys. Anal. Symposium, Cincinnati, OH.

Zechman, E. M. and Ranjithan, S. (2009). "Evolutionary Computation-Based Methods for Characterizing Contaminant Sources in a Water Distribution System." Journal of Water Resources Planning and Management, 135(5), 334-343.

# CHAPTER 5. Asynchronous hierarchical parallel evolutionary algorithm for large-scale global optimization

## 5.1 Introduction

Evolutionary algorithms (EA) are efficient search methods that are structured based on the principles of natural selection. EAs have been established to be able to find good solutions to a wide range of complex optimization problems. However, solving complex optimization problems are often associated with high computational cost, which grows with the size and complexity of the problem. Computational requirements are even higher when using simulation-optimization to solve typical engineering problems where each function evaluation of the search algorithm requires simulation of some complex numerical models. A large number of engineering optimization problems in practice that is associated with real-time systems demands near-real-time solution, increasing the need for computationally faster solution schemes. This has led to several studies on developing parallel evolutionary algorithms.

Evolutionary algorithms are inherently parallel in nature, allowing many of the steps in the algorithm to be readily structured for computing them simultaneously on multiple processors. Several different approaches have been developed and reported in literature for parallelizing evolutionary algorithms, and some representative studies reported in the literature by various researchers are outlined below.

Global parallel EA models maintain a single population while the fitness evaluations for the individuals are done in parallel. They are synchronous in structure as all evaluations must be completed before proceeding to the next generation. Global parallel EAs have been developed and tested for different shared and distributed memory architectures by various researchers (e.g., Cantu-Paz 1997; Fogarty and Huang 1991; Branke et al. 1996).

Alternatively, coarse-grained island-based parallel EAs are structured as multiple populations that allow occasional exchange (migration) of individuals among the populations. Migration strategies are the primary operators in island-based models and have been studied in detail by Petley et al. (1987), Tanese (1987), and Braun (1990). An empirical study about the effect of communication topologies and migration rates on parallel EAs was reported by Cantu-Paz and Goldberg (1997). Another approach for parallelism in EAs is the use of fine-grained models where the population is divided into small demes with intensive communication among them. The increased communication among the demes provides a way to disseminate across the entire population the good solutions within each deme. Recombination (i.e., selection and mating) operations occur only among the individuals of the same deme. Several hybrid methods combining these different parallelization schemes have also been proposed in the literature (Lim et al. 1997).

## 5.2 Background

Coarse-grained island-based implementations are the type of parallel evolutionary algorithms most widely studied and used. Quality of solution from an island-based parallel EA depends on key factors such as: 1) number of islands of population; 2) population size; 3) migration rate; and 4) communication topologies. Studies reported in literature have established the effectiveness of parallel island-based models for different applications. Most of the algorithms offer, however, poor computational scalability, limiting their application for solving complex engineering design problems where real-time solutions are desired. The poor scalability can be attributed to two major issues:

- Communication overhead: Often engineering design problems involve a large number of variables, which increases the storage requirements, consequently limiting the population size that can be used. Though increasing the number of islands can compensate for the population size limitation, it in turn leads to increased communication overhead during migration operations.
- Synchronization overhead and load imbalance: Every population in an island-based

model needs to synchronize during the migration step. In a heterogeneous computing environment, faster processors would have to wait at each migration step for the slower processors to complete their computations.

An island-based model scales poorly with the number of processors due to high communication overhead involved and the need for generational synchronization. These issues limit the applicability of the island-based models for real-time solution of large complex engineering design problems even when abundant computational resources are available.

The present research develops a new asynchronous hierarchical parallel evolutionary algorithm (AHPEA) framework that addresses some of these limitations of standard island-based models.

## 5.3 Asynchronous Hierarchical Parallel Evolutionary Algorithms (AHPEA)

### 5.3.1 Motivation

AHPEA is designed for the solution of large-scale global optimization problems while achieving enhanced parallel performance and scalability. The algorithm design was motivated by two key ideas for algorithmic and computational efficiency, which are described below.

In natural systems, communities of a species co-exist and grow in different places and in different environments, inherently increasing intra-species diversity. Being subjected to different environments, individuals of the same species adapt themselves to fit best into their own environment. Through this process of adaptation, each community develops unique traits of strength for survival. Members of the communities interact with each other as well as migrate from one community to another. An individual after migrating to a new community interacts with the members of the new community to share its traits and adapts in the new environment. Through exchange of the strong traits of the incoming member and the members of the host population, better individuals are expected to be formed and evolved.

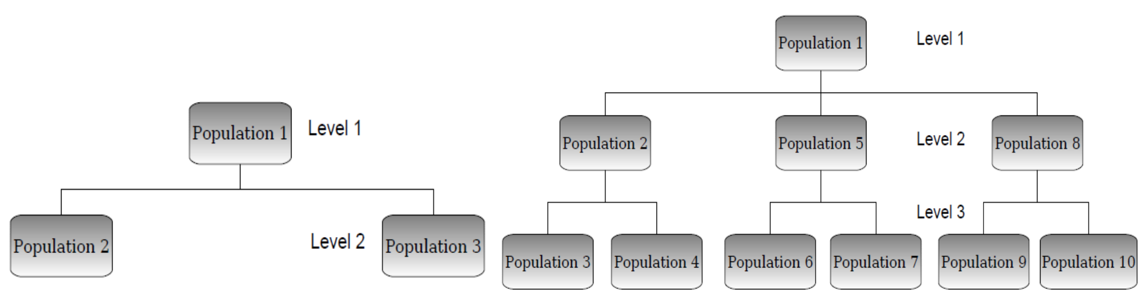
Architecture of modern high performance computing (HPC) systems is the other motivation for the design of AHPEA. Many modern HPC systems inherently have hierarchical memory architecture. Thus, algorithms with a hierarchical communication structure may map well on to these systems.  For example, a modern supercomputer with thousands of processor cores may have several loosely coupled cabinets, where each cabinet contains several blades that contain several nodes. Each node contains several processors, and each processor contains several cores. A hierarchical communication cost structure exists in these complex distributed memory architectures. Data access or transfer cost between the cores of NUMA (non uniform memory access) architecture-based multi-core processors on a node may vary depending on the location of the processors. Communication between two processing units located on two different physical nodes has to be established through the network, and is consequently associated with a higher cost in terms of time. In a parallel application, time required for communication between any pair of processes would depend upon their physical location on the machine. While this complex communication structure poses a problem, with proper alignment of algorithms and computer architecture the application can be designed to match the underlying communication pattern to achieve maximum efficiency.

The proposed AHPEA framework in this study builds upon these two ideas. It uses a coarse-grained multi-population approach, similar to island-based procedures, with the populations arranged in a hierarchical structure (Figure 5.1). The migration schemes in AHPEA are designed to localize the migration patterns to efficiently exploit the underlying communication pattern.

### 5.3.2 Conceptual framework

#### 5.3.2.1 Hierarchical population topology

Populations in AHPEA are arranged in a multi-layered *n-ary* tree structure with hierarchical relationships where larger numbers of populations are assigned to the lower levels of the hierarchical population layout (Figure 5.1). Each population in the hierarchical structure performs an evolutionary search and shares information with the other connected populations to collectively evolve a set of good solutions. The populations at the bottom-most layer collectively enable a search for good solutions in diverse regions of the search space using a diversity preserving mechanism. The diverse set of good solutions identified by the populations at the bottom layer is shared with the populations in the upper-layer. While the populations in the upper-layer also perform their own evolutionary search, they benefit from the good solutions being passed on by the connected populations at the lower-layer; this is analogous to continuously seeding the upper-layer populations with good diverse solutions found so far by the lower layer populations. Depending on the number of layers, this structure is repeated at the upper layers. This hierarchical structure distributes the search by employing the lower-layer populations to conduct, in parallel, localized search for diverse solutions, and refines the search at the upper layers by building upon the diverse set of locally good solutions.




(a) Three-populations structure (AHPEA-3) (b) Ten-populations structure (AHPEA-10)

**Figure 5.1 Examples of hierarchical population topologies**

#### 5.3.2.2 Computational design

Hierarchical population topology requires decentralized and localized communication among the populations. Every population primarily interacts with the other populations within its hierarchical group. For example, in Figure 5.1b, Populations 2, 3 and 4 form a hierarchical group. Information exchanges with all upper-layer populations could also be allowed (e.g., solutions from Population 3 could be shared with Populations 2, 5 and 8), which implicitly may work against finding and maintaining diverse solutions at upper layers. Significant computational benefits can be achieved by properly mapping the distribution of these populations over the underlying computing hardware. For example, the populations within a hierarchical group (Populations 2-3-4 in Figure 5.1b) that interact frequently among each other can be mapped on the multiple cores of a cluster node (say Node 1) to take advantage of faster communication in a shared-memory environment, while the populations within another hierarchical group (such as Populations 5-6-7 in Figure 5.1b) can be mapped on a different cluster node (say Node 2). Communications between the populations on Node 1 and Node 2 are routed through the network using message passing (with higher overhead than the shared-memory communication). But by properly designing the migration strategies, the faster (shared-memory) communication can be exploited for frequent communications and the slower over-the-network (distributed-memory) communication can be used for infrequent communications.

The computational framework for AHPEA was developed in C language using Message Passing Interface (MPI). To achieve fully asynchronous communication in AHPEA, MPI Remote Memory Access routines were used. Every population shares, at an interval prescribed by the migration strategy, its data with the other populations by saving the data in a shared-memory window, from which other populations needing to acquire the data can read it whenever necessary. With such a data-pull strategy, sender and receiver populations are not required to be available simultaneously to interact during data exchange, avoiding any need for synchronization. A low overhead is experienced due to localized communication schemes in the algorithm, which helps in executing efficiently a fully asynchronous algorithm.

### 5.3.3 Algorithm design

#### 5.3.3.1 Base algorithm

Development of AHPEA is carried out to achieve algorithmic efficiency for the solution of large-scale global optimization problems and most importantly to achieve high computational improvements over the underlying base evolutionary algorithm. A simple genetic algorithm (Figure 5.2) is used as the underlying search procedure when testing the AHPEA framework.

```
Initialize population
Evaluate initial population
for i=1 to maximum number of generations
            Perform competitive selection (tournament selection)
            Generate new solutions: Apply cross-over operator (simulated binary cross-over)
 and mutation operators (polynomial mutation)
Evaluate solutions in the population
end for
```

**Figure 5.2 Base serial genetic algorithm**

For the purpose of a fair comparison, the base algorithm was solved for all the problems reported in this paper and the results were used for the evaluation of the performance of the APEA algorithm. Since the purpose of AHPEA is to provide a conceptual framework for the design of parallel evolutionary algorithms, effort was focused on assessing algorithmic and computational efficiency in comparison to the serial version of the base algorithm. Although a generic version of a genetic algorithm was implemented in AHPEA and tested, any version of evolutionary algorithms can be easily implemented in AHPEA to improve upon its serial version by utilizing the concepts and parallel efficiency of AHPEA.

#### 5.3.3.2 Migration

Strategy for the migration of individuals between the populations has an important effect on the algorithmic as well as the computational performance of any island-based parallel algorithm (Cantu-Paz 1999). In AHPEA, a bidirectional migration scheme is used between population pairs in adjacent layers. Populations in a lower layer send some of their current best individuals to the populations in the upper layer (upward migration) at some predetermined interval. This helps to seed the upper-layer populations. Upper-layer populations also broadcast their good solutions to the lower-layer populations (downward migration). The downward migration helps to nudge a lower-layer population out of potential premature convergence to local optima. Since the upper-layer populations are meant to fine-tune the solutions identified and migrated up by the lower-layer populations, upward migrations are done more frequently than downward migration. Frequency and number of individuals to be migrated during upward/downward migration affect the convergence of the algorithm.

#### 5.3.3.3 Diversity preservation

While the bidirectional migration strategy adopted in the algorithm helps improve the convergence and exploits the solutions from the diverse search conducted by multiple populations at lower layers, it is prone to loosing diversity among the populations. If a good solution that is migrated downward spreads over all the lower-layer populations, it may steer the search in all lower-layer populations towards the same region of the search space. In such situations, diversity among the lower-layer populations could be lost, resulting in redundant search by the populations. To minimize this from occurring, features to maintain diversity among the populations at the same level of hierarchy as well as between different levels are included in the algorithm. This is achieved in two different ways.

Each population calculates the centroid of its location in the search space and shares that information with other populations. To maintain diversity among the populations, an individual solution is selected based not only on its fitness value, but also on the distances of that individual from the centroids of the other populations in the same layer. This attempts to keep populations in a layer far away from each other.

The solutions received by a population through downward migration helps to improve the convergence of that population, but it may also lead to loss of diversity among populations at the vertically adjacent layers. To minimize this effect, the incoming downwardly migrated solutions are included in the selection process, which is based, as described above, on fitness and diversity considerations. Thus, an incoming solution with superior fitness may be rejected by a population if the incoming solution is located closer to another population in the same layer. This ensures that the good solutions broadcasted by the upper-layer populations are used by the appropriate lower-layer populations without the loss of diversity among the same hierarchical group of populations. Selection pressure may be adaptively varied during the search process to maintain a balance between fitness and diversity considerations.

```
Initialize population
Evaluate initial population
for i=1 to maximum number of generations
    Perform competitive selection (tournament selection)
    Generate new solutions: Apply cross-over operator (simulated binary cross-over)
                    and mutation operators (polynomial mutation)
    Evaluate solutions in the population
    At upward migration frequency:
        Share 'n' best individuals for migration to the parent population
        Receive the 'n' best individuals from all the children populations (through asynchronous remote memory
    access)
        Add received individuals to the selection pool
    At downward migration frequency:
        Share 'm' best individuals for migration to the children populations
        Receive the 'm' best individuals from the parent populations (through asynchronous remote memory
    access)
        Add received individuals to the selection pool
    At centroid update frequency:
        Calculate population centroid and share through the shared memory window
        Get the updated centroids of all the populations at the same level (through asynchronous remote
    memory access)
end for
```

**Figure 5.3 Asynchronous Hierarchical Parallel Evolutionary Algorithm (AHPEA)**

## 5.4 Experimental results

### 5.4.1 Benchmark test functions

A detailed validation study is carried out for AHPEA to test its performance and general applicability for a class of large-scale global optimization problems. A suite of standard benchmark minimization problems from the Competition on Large-Scale Global Optimization held as a part of CEC'2008 (Tang et al. 2008) is adopted in this study for testing and validation of the AHPEA algorithm (http://nical.ustc.edu.cn/conferences/cec2008/lsgo/LSGO.CEC08.Benchmark.zip). These problems are designed to test the performance and robustness of the algorithms for solving high-dimensional global optimization problems. The mathematical formulas and properties of these problems are summarized here. All test functions were solved for 100, 500 and 1000 dimensions in this study.

***Shifted Sphere Function:***

$$F_1(x) = \sum_{i=1}^{D} Z_i^2 \ , \ z = x - o, x = [X_1, X_2, ...., X_D] \qquad (1)$$

D: dimensions. $o = [o_1, o_2, ...., o_D]$ : shifted global optimum

$$x \in [-100,100]^D, Global\ minimum : x^* = o, \quad F_1(x^*) = 0$$

Properties: Unimodal, shifted, separable, scalable

***Shifted Schwefel's Problems:***

$$F_2(x) = \max \ \left|Z_i\right|, 1 \le i \le D \ , \quad z = x - o, x = [X_1, X_2, ...., X_D] \qquad (2)$$

D: dimensions. $o = [o_1, o_2, ...., o_D]$ : shifted global optimum

$$x \in [-100,100]^D, Global\ minimum : x^* = o, \quad F_2(x^*) = 0$$

Properties: Unimodal, shifted, non-separable, scalable

***Shifted Rosenbrock's Function:***

$$F_3(x)=\sum_{i=1}^{D-1} 100\left(Z_i^2-Z_{i+1}^2\right)^2+\left(Z_i-1\right)^2 \quad , \quad z=x-o+1, x=[X_1, X_2,...., X_D] \tag{3}$$

D: dimensions. $o=[o_1,o_2,....,o_D]$ : shifted global optimum

$x\in[-100,100]^D, Global\ minimum : x^*=o, \quad F_3(x^*)=0$

Properties: Multi-modal, shifted, non-separable, scalable

***Shifted Rastrigin's Function:***

$$F_4(x)=\sum_{i=1}^{D} \left(Z_i^2=10\cos(2\pi Z_i)+10\right) \quad , \quad z=x-o+1, x=[X_1, X_2,...., X_D] \tag{4}$$

D: dimensions. $o=[o_1,o_2,....,o_D]$ : shifted global optimum

$x\in[-5,5]^D, Global\ minimum : x^*=o, \ F_4(x^*)=0$

Properties: Multi-modal, shifted, separable, scalable

***Shifted Griewank's Function:***

$$F_5(x)=\sum_{i=1}^{D}\frac{Z_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{Z_i}{\sqrt{i}}\right)+1, \quad z=x-o+1, x=[X_1, X_2,...., X_D] \tag{5}$$

D: dimensions. $o=[o_1,o_2,....,o_D]$ : shifted global optimum

$x\in[-600,600]^D, Global\ minimum : x^*=o, \ F_5(x^*)=0$

Properties: Multi-modal, shifted, non-separable, scalable

***Shifted Ackley's Function:***

$$F_6(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} Z_i^2}\right) \quad -\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\ 2\pi Z_i\right) + 20 + e, \tag{6}$$

$$z = x - o + 1, x = [X_1, X_2, ...., X_D]$$

D: dimensions. $o = [o_1, o_2, ...., o_D]$ : shifted global optimum

$$x \in [-32, 32]^D, Global\ minimum : x^* = o,\ \ F_6(x^*) = 0$$

Properties: Multi-modal, shifted, separable, scalable

### 5.4.2 Results

The purpose of AHPEA is to achieve high algorithmic and as well as computational performance compared to a serial version of the same algorithm. As described before, a genetic algorithm (Figure 5.2) was implemented in AHPEA framework and was used for all the analyses reported here. Summary statistics (mean and standard deviation) of the solutions for 25 random trials for each problem instance are presented to show the average and variability of performance of the algorithm.

The serial version of the algorithm was solved for all the test functions and those results were used to evaluate the performance of AHPEA. Table 5.1 shows the results of the serial algorithm for 100 dimensional versions of all the test functions described above. The problems are solved for increasing population sizes of 100 (1 x number of variables), 300 (3 x number of variables) and 1000 (10 x number of variables). Evolutionary search was conducted for 1000 generations for each problem instance. The genetic operators and parameter values used for the serial algorithm are shown in Table 5.2. Improved solutions are obtained for all test functions as population size was increased.

**Table 5.1 Results obtained using serial algorithm for the 100 dimensional (D=100) versions of the test problems. The results are based on 25 random trials.**

| Test functions | Population size = 100 | | Population size = 300 | | Population size = 1000 | |
|---|---|---|---|---|---|---|
| | Mean | Std. dev. | Mean | Std. dev. | Mean | Std. dev. |
| Shifted Sphere | 1.647E+02 | 3.896E+01 | 3.885E+01 | 6.606E+00 | 6.831E+00 | 1.601E+00 |
| Shifted Schwefel | 9.753E+01 | 8.028E+00 | 6.943E+01 | 6.412E+00 | 4.041E+01 | 5.702E+00 |
| Shifted Rosenbrock | 3.087E+05 | 9.947E+04 | 2.929E+04 | 8.513E+03 | 6.925E+03 | 4.704E+03 |
| Shifted Rastrigin | 3.089E+02 | 5.363E+01 | 1.265E+02 | 2.054E+01 | 3.435E+01 | 7.715E+00 |
| Shifted Griewank | 2.534E+00 | 2.687E-01 | 1.369E+00 | 8.033E-02 | 1.058E+00 | 4.520E-02 |
| Shifted Ackley | 1.610E+01 | 3.404E+00 | 3.260E+00 | 3.020E+00 | 7.533E-01 | 2.793E-01 |

**Table 5.2 Parameter values for the implementation of the algorithms used in this study**

| *Parameter values used for the serial algorithm* | *Parameter values used for the parallel algorithm in AHPEA* |
|---|---|
| Cross over operator: Simulated Binary Crossover<br>Crossover probability = 0.60<br>Mutation operator: Polynomial mutation<br>Mutation probability = 0.05<br>Selection operator = Tournament selection | Cross over operator: Simulated Binary Crossover<br>Crossover probability = 0.60<br>Mutation operator: Polynomial mutation<br>Mutation probability = 0.05<br>Selection operator = Tournament selection<br>Upward migration starting at 125th generation<br>Upward migration frequency: Every other generation<br>Upward migration size: Top two individuals<br>Downward migration starting at 250th generation<br>Downward migration frequency: Every other generation<br>Downward migration size: Top two individuals |

AHPEA is applied to solve the same instances of the 100-dimensional test functions. Population sizes (300 and 1000) used for the serial algorithm were split into multiple populations of 100 individuals each and they were arranged in a hierarchical structure using a three-population (AHPEA-3) and a ten-population (AHPEA-10) topology as shown in Figure 5.1. The multiple populations in AHPEA are assigned to parallel processors such that the total evaluations at each population to be equivalent to that of the serial implementation with a population of 100 individuals. Assuming a perfect speed up, time-to-solution for the AHPEA-3 and AHPEA-10 topologies can be expected to be the same as that for the serial implementation with a population size of 100. Table 5.3 summarizes the results obtained using AHPEA for the different test functions with a dimension of 100. Comparison of results from the AHPEA algorithm with those for the serial algorithm shows that in equal number of generations for the two algorithms, AHPEA identifies improved results for all the test functions. Thus, instead of increasing the population size of a serial genetic algorithm, organizing the same number of individuals into the hierarchical structure in AHPEA yielded improved solutions for all the benchmark test functions in a significantly reduced computational time. Table 5.2 shows the parameter values used for the parallel algorithm in AHPEA framework.

**Table 5.3 Results obtained using the parallel algorithm in AHPEA for the 100 dimensional (D=100) versions of the test problems. The results are based on 25 random trials.**

| Test functions | Number of populations | Population size = 100 | | Population size = 200 | |
|---|---|---|---|---|---|
| | | Mean | Std. dev. | Mean | Std. dev. |
| Shifted Sphere | 1 (serial algorithm) | 1.647E+02 | 3.896E+01 | 6.569E+01 | 9.761E+00 |
| | 3 | 8.557E-01 | 1.801E-01 | 2.199E-01 | 5.692E-02 |
| | 10 | 1.042E-01 | 2.881E-02 | 2.134E-02 | 2.460E-02 |
| Shifted Schwefel | 1 (serial algorithm) | 9.753E+01 | 8.028E+00 | 7.484E+01 | 7.647E+00 |
| | 3 | 1.388E+01 | 3.605E+00 | 6.850E+00 | 1.963E+00 |
| | 10 | 1.036E+01 | 5.830E+00 | 5.557E+00 | 2.168E+00 |
| Shifted Rosenbrock | 1 (serial algorithm) | 3.087E+05 | 9.947E+04 | 6.685E+04 | 1.809E+04 |
| | 3 | 2.514E+03 | 3.853E+03 | 1.309E+03 | 2.286E+03 |
| | 10 | 5.230E+02 | 7.024E+01 | 4.470E+02 | 7.374E+01 |
| Shifted Rastrigin | 1 (serial algorithm) | 3.089E+02 | 5.363E+01 | 1.789E+02 | 4.110E+01 |
| | 3 | 1.733E+01 | 4.213E+00 | 1.899E+00 | 1.282E+00 |
| | 10 | 1.103E+00 | 9.783E-01 | 9.681E-03 | 3.908E-03 |
| Shifted Griewank | 1 (serial algorithm) | 2.534E+00 | 2.687E-01 | 1.617E+00 | 1.162E-01 |
| | 3 | 3.951E-01 | 6.288E-02 | 1.211E-01 | 2.673E-02 |
| | 10 | 9.101E-02 | 6.481E-02 | 1.443E-02 | 7.906E-03 |
| Shifted Ackley | 1 (serial algorithm) | 1.610E+01 | 3.404E+00 | 4.719E+00 | 3.609E+00 |
| | 3 | 1.668E-01 | 2.220E-02 | 5.089E-02 | 8.527E-03 |
| | 10 | 4.198E-02 | 6.606E-03 | 1.452E-02 | 4.146E-03 |

Figure 5.4 and Figure 5.5 show the convergence of the populations of AHPEA-3 and AHPEA-10, respectively, for the 100-dimensional Shifted Ackley's function. In AHPEA-3, faster convergence is achieved by the Population 1 (at topmost Level 1 as in Figure 5.1a), which is seeded by the solutions from the diverse search conducted by Populations 2 and 3 at the lower layer (Level 2 as in Figure 5.1a). Relatively independent search is carried out by each of the lower-level population, which explains the relatively slower convergence. Occasionally, Population 1 broadcasts its good solutions to Populations 2 and 3 through the downward migration process, which helps improve the search by the lower-layer populations and finally converge to near-optimal solutions.

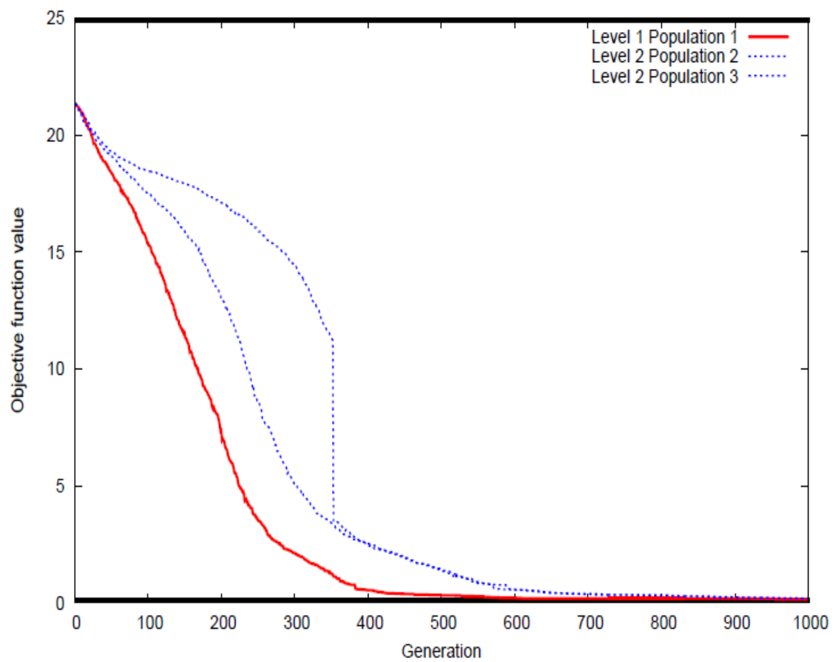



**Figure 5.4 Typical convergence of each population in the three-population AHPEA (AHPEA-3) for the 100-dimensional (D=100) Shifted Ackley's function**

Similar observations are also made for AHPEA-10 (Figure 5.5), where slower convergence was observed for the populations at the lowest layer (Level 3, as in Figure 5.1b) which carries out relatively independent searches. Comparatively better convergence is achieved by the populations at the middle layer (Level 2, as in Figure 5.1b) as good solutions from the lower–layer populations are used to seed the search. Similarly, the top-layer population (population 1 in Level 1, as in Figure 5.1b) benefits from the good solutions migrating up from the middle-layer populations and achieved the best convergence. It can be observed that some of the populations at Levels 2 and 3 had relatively weaker convergence during the early stages of the search. The convergence of the middle-layer populations in Level 2 improves as the upward migration begins at Iteration 125. Similarly, convergence of some of the lowest-layer populations in Level 3 improved after the downward migration began at Iteration 250 when good solutions from the upper layers are used to seed the search at the lower-layer populations. Similar convergence patterns for AHPEA were also observed for the other test problems. These results and observations show the nature and efficiency of the hierarchical evolution in AHPEA.

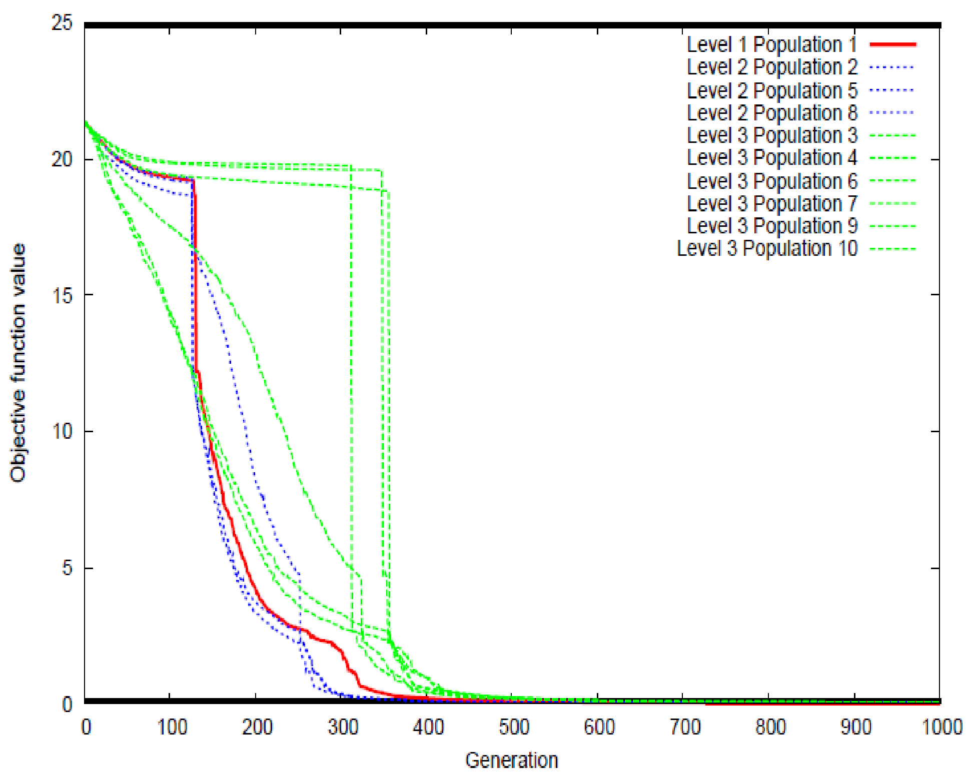



**Figure 5.5 Typical convergence of each population in the ten-population AHPEA (AHPEA-10) for the 100-dimension Shifted Ackley's function**

The convergence of AHPEA can be further improved either by increasing the population size of each populations or by increasing the number of populations in the hierarchical topology. To analyze the effect of increasing population size on the performance of AHPEA, all the test problems (for dimension size of 100) were solved using increased population size of 200 individuals for the AHPEA-3 and AHPEA-10 topologies. In each case, the search was carried out for 1000 generations. Summary statistics of solutions obtained for 25 random trials are shown in Table 5.3. In most cases better results are obtained by increasing the number of populations in the hierarchical topology instead of increasing the population size. In situations where the computational resources are limited, increase in population size can help improve the solutions, though at the cost of increased time for convergence. Application of AHPEA with increased number of populations can efficiently utilize the high performance computing resources, if available, and significantly reduce the time-to-solution.

AHPEA framework was also applied for all the benchmark problems in the test suite for 500 and 1000 dimensions to test its applicability to solve high dimensional global optimization problems. Table 5.4 and Table 5.5 show the results for the 500 dimension and 1000 dimension problems, respectively. Compared to the serial algorithm, significant improvement in the quality of solutions is achieved by the parallel algorithm within the AHPEA framework in comparable or better time to converge.

**Table 5.4 Results obtained using the parallel algorithm in AHPEA for the 500 dimensional (D=500) versions of the test problems. The results are based on 25 random trials.**

| Test functions | Number of populations | Population size = 500 | | Population size = 1000 | |
|---|---|---|---|---|---|
| | | Mean | Std. dev. | Mean | Std. dev. |
| Shifted Sphere | 1 (serial algorithm) | 1.150E+04 | 1.121E+03 | 7.817E+03 | 6.579E+02 |
| | 3 | 5.898E+02 | 2.311E+02 | 2.626E+01 | 1.054E+01 |
| | 10 | 3.248E+01 | 2.294E+01 | 2.622E-01 | 4.931E-01 |
| Shifted Schwefel | 1 (serial algorithm) | 1.234E+02 | 3.007E+00 | 1.142E+02 | 3.288E+00 |
| | 3 | 8.103E+01 | 2.765E+00 | 7.002E+01 | 3.013E+00 |
| | 10 | 7.265E+01 | 2.735E+00 | 5.877E+01 | 4.064E+00 |
| Rosenbrock | 1 (serial algorithm) | 2.263E+08 | 5.185E+07 | 8.451E+07 | 1.417E+07 |
| | 3 | 6.183E+06 | 2.323E+06 | 7.773E+05 | 6.421E+05 |
| | 10 | 7.117E+05 | 6.458E+05 | 2.435E+04 | 2.200E+04 |
| Shifted Rastrigin | 1 (serial algorithm) | 2.036E+03 | 1.823E+02 | 1.100E+03 | 1.127E+02 |
| | 3 | 5.832E+02 | 1.701E+02 | 2.898E+02 | 6.819E+01 |
| | 10 | 3.541E+02 | 1.051E+02 | 1.322E+02 | 3.374E+01 |
| Shifted Griewank | 1 (serial algorithm) | 1.082E+02 | 6.690E+00 | 7.213E+01 | 7.048E+00 |
| | 3 | 6.252E+00 | 1.818E+00 | 1.341E+00 | 1.393E-01 |
| | 10 | 2.220E+00 | 7.612E-01 | 4.327E-01 | 3.102E-01 |
| Shifted Ackley | 1 (serial algorithm) | 1.959E+01 | 5.556E-01 | 1.784E+01 | 1.887E+00 |
| | 3 | 5.837E+00 | 5.614E+00 | 1.795E+00 | 5.722E-01 |
| | 10 | 2.356E+00 | 4.634E-01 | 4.837E-01 | 4.826E-01 |

**Table 5.5 Results obtained using the parallel algorithm in AHPEA for the 1000 dimensional (D=1000) versions of the test problems. The results are based on 25 random trials.**

| Test functions | Number of populations | Population size = 1000 | Population size = 2000 |
|---|---|---|---|
| | | Mean | Mean |
| Shifted Sphere | 1 (serial algorithm) | 5.393E+04 | 3.595E+04 |
| | 3 | 3.895E+02 | 1.143E+02 |
| | 10 | 2.956E+01 | 1.037E-01 |
| Shifted Schwefel | 1 (serial algorithm) | 1.357E+02 | 1.258E+02 |
| | 3 | 1.060E+02 | 9.145E+01 |
| | 10 | 9.180E+01 | 8.169E+01 |
| Shifted Rosenbrock | 1 (serial algorithm) | 1.686E+09 | 6.477E+08 |
| | 3 | 3.787E+07 | 2.876E+06 |
| | 10 | 2.990E+06 | 1.585E+04 |
| Shifted Rastrigin | 1 (serial algorithm) | 3.352E+03 | 1.805E+03 |
| | 3 | 1.427E+03 | 7.363E+02 |
| | 10 | 7.021E+02 | 5.308E+02 |
| Shifted Griewank | 1 (serial algorithm) | 4.736E+02 | 2.893E+02 |
| | 3 | 7.043E+00 | 2.094E+00 |
| | 10 | 3.519E+00 | 3.525E-01 |
| Shifted Ackley | 1 (serial algorithm) | 2.020E+01 | 1.975E+01 |
| | 3 | 1.427E+01 | 2.266E+00 |
| | 10 | 2.238E+00 | 8.317E-02 |

Results obtained using the parallel algorithm in AHPEA for the set of benchmark mathematical test problems (for 100, 500 and 1000 dimensions) show the efficiency and robustness of the algorithm in solving a general class of large-scale global optimization problems. For the purpose of comparison, the values of various parameters (e.g., migration rates and frequencies, and crossover and mutation probabilities) were kept constant for the all the problems reported in this study. However, significantly better results can be obtained from the algorithm by fine-tuning these parameters for each problem. Effect of different population sizes, migration schemes and population topologies were also explored.

## 5.5 Conclusions

This study proposed a new approach for an asynchronous hierarchical parallel evolutionary algorithm. A hierarchical population topology is used in AHPEA to obtain good quality solutions to large-scale global optimization problems by enabling hierarchical evolution and to achieve fast convergence through high computational efficiency by utilizing localized communication schemes. The algorithm was applied to solve an array of standard benchmarks test problems for large-scale global optimization and was established to obtain robustly high quality solutions. Performance of the parallel algorithm in AHPEA was found to be significantly better than that of the serial algorithm with equivalent population sizes. AHPEA yielded significant reduction in the time-to-solution for all the test problems. As the AHPEA framework is developed to map efficiently to the high performance computational architecture on which AHPEA is executed, this close synergy between the design of the algorithm and modern computer architecture provides excellent parallel computing performance and scalability on high performance computing clusters.

AHPEA is developed primarily for the solutions of real-world engineering design problems, which demand real-time solutions. It was successfully applied for the solution of several large-scale complex engineering problems to provide near real-time solution using state-of-the-art supercomputers. Results of these studies are reported in a separate chapter.

## References


Branke, J., Andersen, H. , Schmeck, H. (1996). "Global selection methods for massively parallel computers." Proceedings of AISB'96 Workshop on Evolutionary Computing, LNCS 1143, Springer, 175-188.

Braun, H. (1990). "On solving travelling salesman problem by genetic algorithm." Parallel Problem Solving from Nature, 129-133.

Cantú-Paz, E. (1997). "Designing efficient master-slave parallel genetic algorithms." Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

Cantú-Paz, E. (1997). "A survey of parallel genetic algorithms." Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

Cantu-Paz, E. and Goldberg, D. (1997). "Modeling idealized bounding cases of parallel genetic algorithms." Genetic Programming, 353-361.

Cantú-Paz, E. (1999). "Topologies, migration rates and multi-population parallel genetic algorithms." Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

Fogarty, T. and Huang, R. (1991). "Implementing the genetic algorithm on transputer based parallel processing systems." Parallel Problem Solving from Nature, 145-149.

Petley, C., Leuze, M., Grefenstette J. (1987). "A parallel genetic algorithm." Proceedings of the 2nd International Conference on Genetic Algorithms, 155-161.

Tanese, R. (1987). "Parallel genetic algorithm for a hypercube." Proceedings of the 2nd International Conference on Genetic Algorithms, 177-183.

Tang, K., Yao, X., Suganthan, P.N., MacNish, C., Chen, Y. P., Chen, C. M., Yang, Z. (2008). "Benchmark functions for the CEC'2008 Special session and competition on large scale global optimization."

http://nical.ustc.edu.cn/conferences/cec2008/lsgo/LSGO.CEC08.Benchmark.zip

# CHAPTER 6. Computational framework for Asynchronous Hierarchical Parallel Evolutionary Algorithm

## 6.1 Introduction

Parallel evolutionary algorithms (EAs) are designed to partition the tasks in an evolutionary search which can then be assigned to multiple processors for execution in parallel. Efficiency of a parallel algorithm greatly depends on its computational implementation and design. Several studies have been reported in the literature for design of parallel EAs for different target computing platforms. Master-slave genetic algorithms (GAs) were studied on shared and distributed memory computers by Cantu-Paz (1997). An asynchronous genetic algorithm was developed by Liu et al. (2002) using MPI and PVM. Branke et al. (2004) developed an island-based model for a heterogeneous cluster of computers. Lim et al. (2007) presented a hierarchical genetic algorithm with a multi-population GA at the first level and a master-slave GA at second level. Each population was mapped to a different cluster and further evaluations of a master-slave GA were distributed on processors available on that cluster. They presented an implementation of the algorithm for grid computing environment. They also developed some middleware and meta-scheduler for recovery of the available resources on the grid. Computing tools, such as Globus ToolKit, the Commodity Grid Kit (CogKit), Ganglia monitoring tool and Netsolve, were used. In addition, they presented case studies for solving real world problems in a grid environment.

Architecture and the design of the target parallel computing platforms primarily dictate the effectiveness of the designed algorithm in terms of its computational performance. While solving a complex engineering problem, given enough computational resources, the time to solution (or turnaround time) is of immense importance. The design of parallel algorithms should be carried out to efficiently utilize the available computational resources to minimize turnaround time. The hierarchical structure of most modern parallel platforms consisting of

GPUs (Graphical Processing Units), cores, processors, nodes, blades, and cabinets makes the design of efficient parallel algorithms for these machines a non-trivial task.. In many cases, the parallel algorithm should be designed such that an efficient mapping is achieved with the underlying hardware. The goal of this research is to develop a parallel evolutionary algorithm, Asynchronous Hierarchical Parallel Evolutionary Algorithm (AHPEA), which can effectively utilize the large number of processors on a modern parallel supercomputer to conduct an accurate and efficient search for the global optima.

This chapter describes the design of a computational framework for AHPEA and parallel performance results from the application of the framework on various state-of-the-art supercomputers. The AHPEA framework was also successfully applied to solve the water distribution system (WDS) contamination source identification and leak detection problems (Kumar et al. 2007; 2008; 2009; 2010a; 2010b).

## 6.2 Computational issues with parallel EAs

Coarse grained island based parallel EAs are the most widely used and well studied in the literature. Quality of solution from an island-based parallel EA depends on major factors such as: 1) number of population islands; 2) population size; 3) migration rate; and 4) communication topologies. These aspects parallel evolutionary algorithms have been studied in details by various researchers (Alba and Tomassini 2002). However, the computational scalability of most of the parallel EA models is limited due to two key computational design issues as described below.

Communication overhead: Often engineering design problems involve large number of variables that increases the storage requirements for the individuals, thus limiting the population size that can be used. Though the limitation in population size can be compensated by increasing the number of islands, it in turn leads to increased communication overhead during migration operations.

Synchronization overhead and load imbalance: Every population in an island based model needs to synchronize during the migration step. In a heterogeneous computing environment, faster processors would have to wait for the slower processors to finish during the migration step.

High communication overhead and generational synchronization limits the scalability of a parallel EA to large number of processors, thus limiting the ability to use the available computational resources to solve complex and critical real world problems in near-real-time. A fully asynchronous and massively parallel computational framework for hierarchical evolutionary algorithm has been designed in this study to avoid these computational difficulties. Design of AHPEA was motivated by the underlying hierarchical architecture of the modern parallel computers.

## 6.3 Design of computational framework

The computational implementation targets a distributed memory architecture that may consist of shared memory sub systems. For example, a modern supercomputer with thousands of processor cores may have several loosely coupled cabinets with each cabinet containing several blades, each blade containing several nodes, each node containing several processors, and each processor containing several cores. While primarily a distributed memory environment exists in the system, processors located on the same physical nodes are connected to each other in a shared memory environment.

### 6.3.1 Computational tools

The current framework was developed in C language using Message Passing Interface (MPI) and OpenMP that are most widely accepted and supported standards for distributed and shared memory programming. Development and testing of all the implementations were done in a Linux/Unix environment which is the most common operating system for high performance computers.

### 6.3.2 Hierarchical topology

As mentioned above, modern parallel computers consist of large number of commodity hardware connected together in a hierarchical fashion. A combination of the shared and distributed memory environments present in these systems. Data access or transfer cost between the cores of NUMA (non uniform memory access) architecture based multi-core processors on a node may vary depending on their locations. Any communication between two processing units located on two different physical nodes has to be established through the network, which is associated with a higher cost in terms of time. In a parallel application, time required for communication between any pair of processes would depend upon their physical location on the machine. While this complex communication structure poses a problem, a proper understanding of algorithms and computer architecture can help design the application to match the underlying communication pattern for achieving maximum efficiency. Hierarchical population topology in AHPEA was specially designed to exploit the underlying hierarchical memory architecture of these computers.

The migration schemes in the AHPEA framework are designed to optimize the communications among the populations in multiple layers. To minimize the communication overhead in AHPEA, the populations are appropriately mapped to the physical processors on the system using a graph partitioning program. The population topology information is preprocessed to partition the graph based on the architecture of the target computer. Figure 6.1 shows an example mapping scheme used for a ten population EA in the AHPEA framework on three quad-core processors.

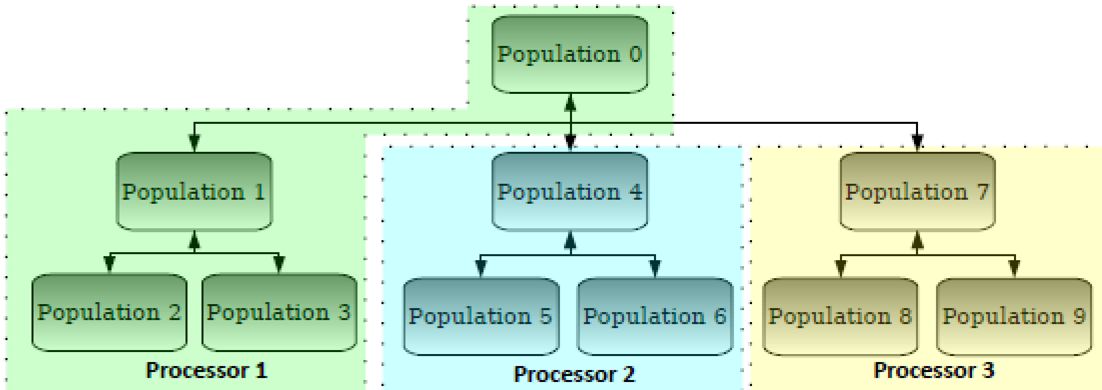


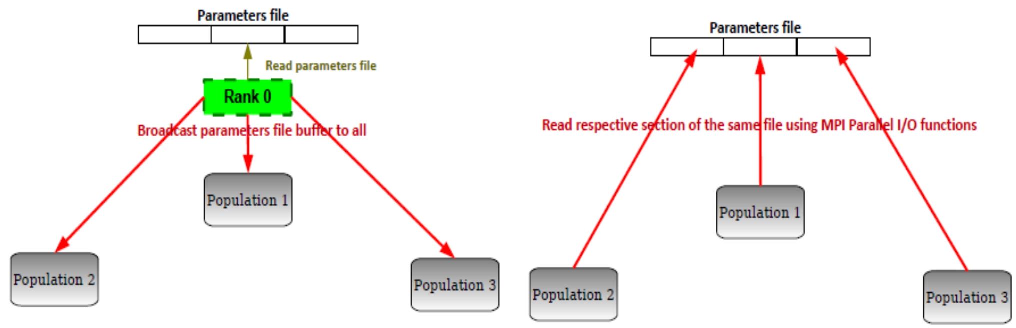

**Figure 6.1 Population to processor mapping in AHPEA for a 10 population EA**

Since migration schemes in AHPEA are designed to limit the primary data exchanges to only among the populations in a multi-layer group (e.g., Populations 1, 2 and 3 in Figure 6.1) that are mapped onto the cores of same processor, these communications are carried out using the faster shared memory environment. Over-the-network communications are required for data exchanges between the populations in different multi-layer groups, which are minimized in the migration schemes used in AHPEA.

### 6.3.3 File I/O

The computational framework for AHPEA allows the multiple populations of the algorithm to use the same or different sets of algorithm-specific parameters (such as population size, cross-over probability, and mutation probability) based on the problem definition and solution requirements. Thus, all populations need to read the input parameters from a user-provided input file at the start of execution. Support for two modes of file I/O is provided in the current framework.

In the first mode, a single processor reads the input data file into a buffer and sends the buffer to the respective processor, which then parses the data and assigns the parameters to the target populations hosted on that processor (Figure 6.2.a).




(a) File read and broadcast by a single processor (b) Each processor reads from the same file

**Figure 6.2 File I/O modes in the AHPEA framework**

In the second mode (Figure 6.2.b), all processors read their input parameters from a single file using MPI parallel file I/O routines (Gropp et al. 1999b). This mode is not preferred when using a small number of processors or when a common set of parameters is used for all populations due to the high startup overhead. However, when using a large number of processors on platforms with dedicated parallel I/O systems, using this mode may result in improved performance depending on the underlying MPI-IO implementation. These two I/O modes in the framework provide flexibility to use the program on a wide range of parallel platforms.

### 6.3.4 Asynchronous communication

Migration is one of the most important steps and a primary driving force for the evolution of good solutions in island-based parallel EAs. Populations share some of their individuals with each other at some frequency determined by the migration scheme to help improve their own search as well as that of others. Explicit message passing using two-sided, and often synchronous point-to-point communication protocols (e.g., MPI_Send/MPI_Recv, Gropp et al. 1999a) are commonly used for implementing parallel EAs (Liu at al. 2002; Branke at al. 2004; Alba and Tomassini 202). Such two-sided handshake-based communications require the involvement of both the sending and the receiving processes for the completion of the data transfer. This leads to a generational synchronization where a faster process may have to wait for a slower process to complete the exchange of individuals through migration before it can continue with the next generation. Schemes for asynchronous communications in parallel EAs have been developed by several researchers. An asynchronous model in MPI using functions MPI_Iprobe(), MPI_Recv() and MPI_Isend() was developed and reported in the literature by Liu et al. (2002). MPI_Iprobe() and MPI_Isend() are non-blocking functions in MPI. In their application, a MPI_Iprobe() was posted to check for any message and a blocking MPI_Recv() was used only after MPI_Iprobe() returns "true." While this model enables asynchronous communication, it may not provide good performance if there is load imbalance among the processors as is often the case in heterogeneous computing environments. While MPI asynchronous communication functions such as MPI_Isend(), MPI_Irecv() and MPI_Iprobe() are non-blocking, they are two-sided communication routines and still require the involvement of both the sending and the receiving processes to complete the communication. While a population (e.g., Population 1 in Figure 6.1) may continue with its computation by sending the individuals to be shared by another population (e.g., Population 2 in Figure 6.1) using non-blocking MPI_Isend() functions, at some point they are required to wait for all the MPI_Isend() calls to complete by issuing MPI_Wait()/MPI_Waitall() calls. When there is load imbalance in the system (for example, in a heterogeneous platform or in the event of uneven computations), a population on a faster processor (e.g., Population 1) may be several generations ahead of Population 2 in an evolutionary search. Suppose a scenario where Population 1 sends a message (message 1) to Pop 2, and before that message was received by Population 2, it reaches the next migration step and sends a second message (message 2). When Population 2 issues an MPI_Iprobe() to look for any incoming message, it will find two messages matching the request (message 1 and message 2). While at least one of the messages will be received by Population 2, it cannot be guaranteed whether the message received was message 1 or message 2. In systems with significant load imbalance, a large number of such incomplete communications may arise at any given time, potentially incurring a significant amount of overhead due to message contention; a large number of outstanding messages can sometimes result in a program-hang. While such redundant messages can be avoided by the sending process by checking for the status (using MPI_Staus()) of last sent message and cancelling it (using MPI_Cancel()), significant overhead may be incurred due to repeated status checks and message cancellations. In this research, such an implementation of AHPEA was first attempted but later abandoned due to the aforementioned problems.

To overcome these limitations, the communications in the current AHPEA framework is designed using MPI-2 one-sided communication functions (Gropp et al. 1999b). A 'pull' strategy where a host process can readily access the migration data from another process using a one-sided 'Get' operation was used. A virtual shared memory buffer is created by each processor (using MPI_Win_create()) where the host populations post the migration data to be shared with the other populations. The populations continue with their computations after pushing the data to be migrated into this shared memory buffer without waiting for others to access this data. They also access (using MPI_Get()) asynchronously the remote shared memory window of the other populations to obtain migration data. To avoid any data corruption, an exclusive memory lock (MPI_Win_lock()) is used on the shared memory window during write operations and a shared lock otherwise. Figure 6.3 shows a schematic of asynchronous migration in an instance of AHPEA problem using MPI-2 one-sided routines. Since the involvement of both the sending and the receiving processes is not required, one-sided communication scheme enables a fully asynchronous parallel algorithm.

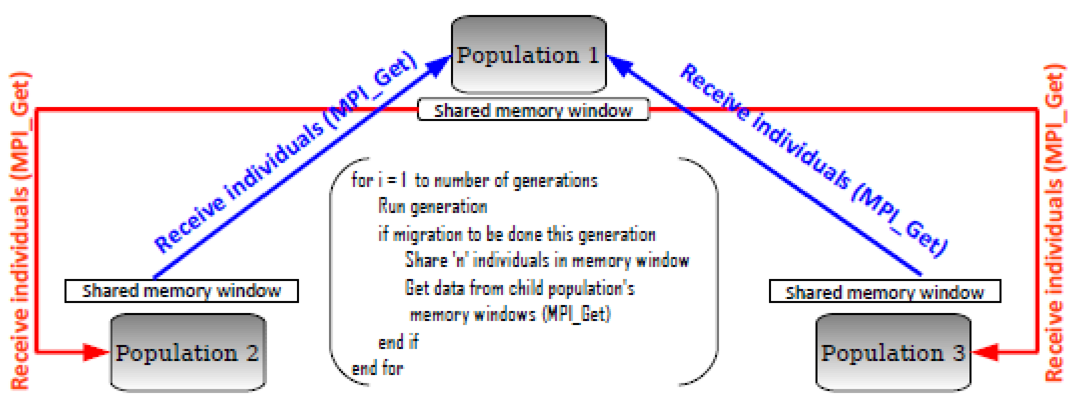



**Figure 6.3 Migration of individuals using MPI-2 one sided communication**

### 6.3.5 Sharing of centroids

Unlike the migrations that are designed to be within a multi-layer group of populations, some information, such as current best solution and population centroid, needs to be shared among all the populations or at least among the populations at the same level of hierarchy. While an asynchronous one-sided communication scheme similar to that used for migration (as described above) can be used for this purpose, significant overhead is associated with such all-to-all memory accesses. Also, since the amount of centroid data to be shared is relatively small, a global memory table was used in the current design for sharing centroid information. A memory window was created on one processor (Rank 0) that has sufficient space allocated for storing centroid information of all populations. At the end of a prescribed number of generations (typically 1-10), every population updates (using MPI_Put()) the corresponding values in the global table with its latest centroid data. It also receives (using MPI_Get()) a copy of the global table for updated information about the other populations.

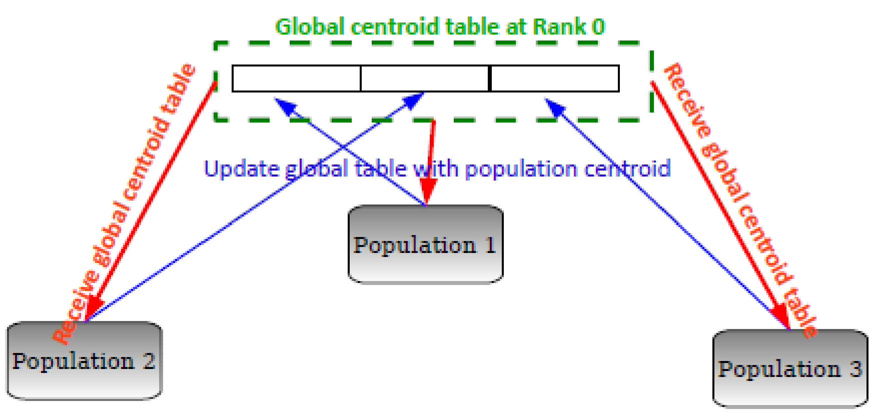



**Figure 6.4 Sharing of centroid information among the populations**

### 6.3.6 Fine grained parallelism using OpenMP

Processors physically located on the same node of a cluster shares a common memory. OpenMP is a portable and efficient protocol for implementing thread-based parallelism in shared memory environments (Chapman et al. 2002). To exploit the faster communications in a shared memory environment, OpenMP was used to add another level of parallelism in the model. OpenMP-based parallelism was used for the function evaluations for the individuals of a population in parallel using the multiple cores/processors residing in a shared memory environment.

### 6.3.7 Simulation-optimization model coupling for water distribution system problem

The AHPEA framework was used to model and solve an inverse problem to identify the contaminant source characteristics of a contamination event in a water distribution system (WDS). The simulation-optimization approach requires the simulation of contaminant movement in the WDS to evaluate the fitness of each potential solution during the search using AHPEA. In this implementation, the EPANET toolkit (Rossman 2000) was used for the WDS simulation. EPANET is an extended period hydraulic and water-quality simulation model developed at EPA. It was originally developed for the Windows platform and provides a C language library with a well-defined API (Sreepathi et al. 2006). Since WDS simulations may be computationally intensive for large networks involving contaminant transport, coarse-grained parallelism feature was incorporated into EPANET by writing an MPI-based wrapper interface around EPANET (Sreepathi et al. 2006). This feature allows execution of multiple EPANET simulations in parallel using a single executable (called ‘pEPANET’) with different decision variables (e.g., source characteristics). Tremendous savings are achieved by not performing repeated operations that are common to all simulations or needed only once (e.g., loading of EPANET program into memory, reading of base input files, and hydraulic simulation). The objective function values are also calculated within this wrapper, avoiding costly communication of the entire output file.

The pEPANET simulation model was integrated into the AHPEA framework using MPI-2 dynamic process creation and management functions. For every population in AHPEA, an instance of simulation model (pEPANET) was started (using MPI_Com_spawn) with specified number of processors and an intercommunicator was established between the optimization (parent process) and simulation (child process) models. The individuals to be evaluated are sent by AHPEA to pEPANET using MPI_Send()/MPI_Recv() via the intercommunicator. The fitness of the individuals are calculated by pEPANET and sent back to AHPEA in the same manner via the intercommunicator. Figure 6.5 shows the data flow and communication pattern in the coupled AHPEA-pEPANET model.

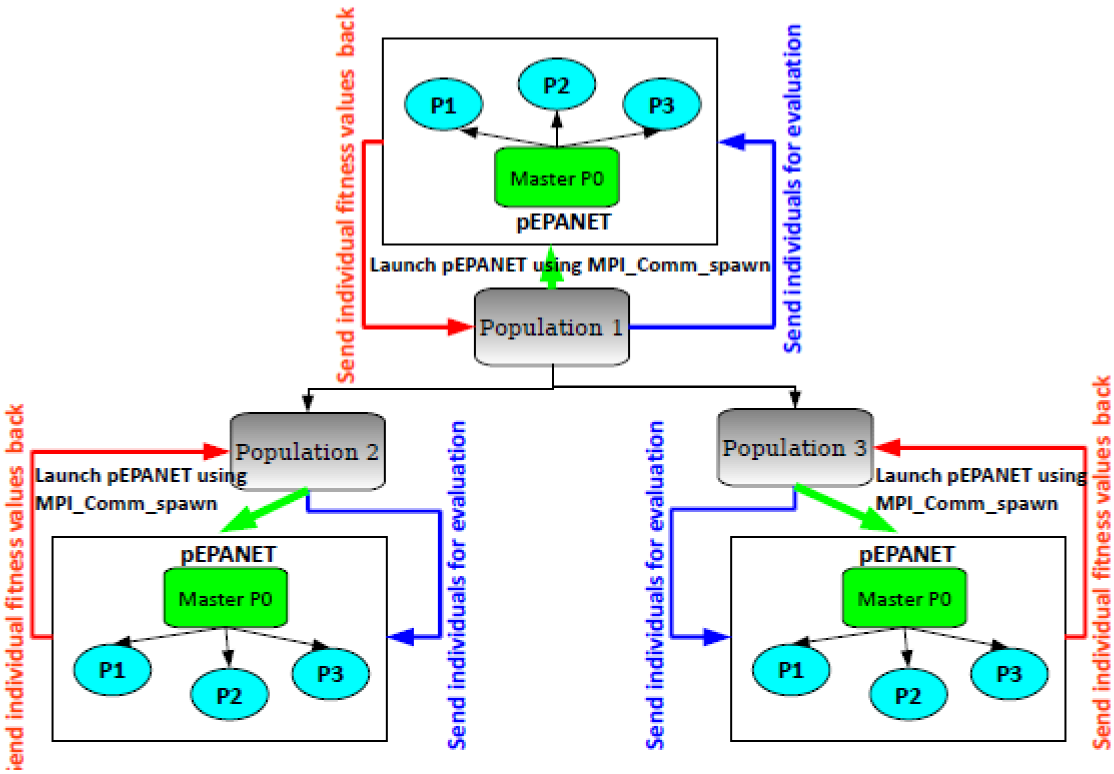



**Figure 6.5 Simulation-optimization model integration into AHPEA using MPI-2 dynamic process creations**

## 6.4 Parallel performance results

### 6.4.1 Application to a test function

Performance analysis of AHPEA was carried out on the 88-core Cygnus cluster (hardware details shown in Table 6.1) at NC State University using 100, 500 and 1000 dimensional Shifted Ackley's function (Tang et al. 2008).

***Shifted Ackley's Function***

$$F_6(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} Z_i^2}\right) \quad -\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\ 2\pi Z_i\right) + 20 + e + f\_bias_6,$$

$$z = x - o + 1, x = [X_1, X_2, ...., X_D]$$

D: dimensions. $o = [o_1, o_2, ...., o_D]$ : shifted global optimum

$$x \in [-32, 32]^D, Global\ minimum : x^* = o,\ \ F_6(x^*) = f\_bias_6 = -140$$

Properties: Multi-modal, shifted, separable, scalable

**Table 6.1 Computational resource used for the study**

| | |
|---|---|
| Host Name | Cygnus |
| Architecture | AMD Quad Core Opteron 2350 |
| Number of nodes | 11 |
| Processor | Two 2.0 GHz Quad Core<br>AMD Opteron |
| Memory | 2GB per core |
| Network interconnect | GigE |
| Filesystem | Network File System (NFS) |

Figure 6.6 shows ‘weak’ scaling of AHPEA with increasing number of processor cores on the Cygnus cluster for problems of different sizes. In weak scaling, an approximate amount of computation per processor core was assigned and was achieved by increasing the population size proportionally with increasing number of processor cores. For perfect scaling, the line should remain nearly horizontal with increasing processor cores. As indicated by the nearly flat lines in Figure 6.6, near perfect scaling was achieved for the Ackley’s function with dimensions of 100 and 500. A slight increase in wall clock time was observed for the 1000-dimensional problem. This can be attributed to the higher communication overhead incurred in the migration steps due to the larger message sizes (80 Kilobytes for each individual). A migration scheme with exchange of two individuals every other generation between each multi-layer group of populations was used for generating the results reported in Figure 6.6. Reducing the size and frequency of migration will lead to improved scalability.

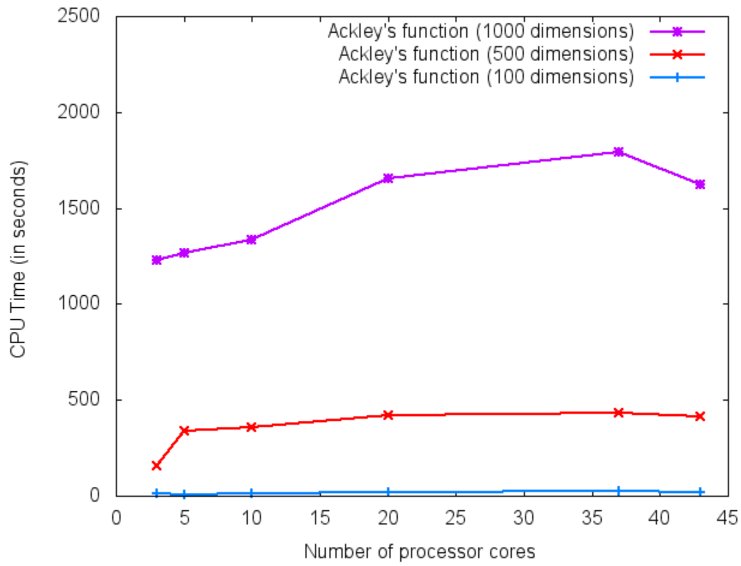
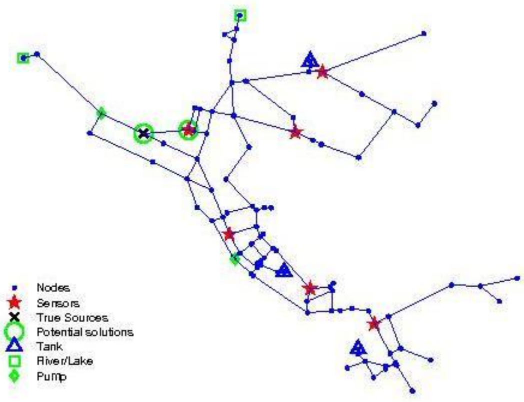


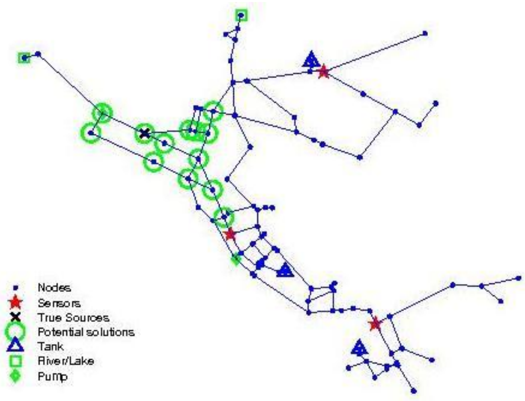

**Figure 6.6 Scaling of AHPEA for different problem sizes solved on the Cygnus system at NCSU)**

### 6.4.2 Application to water distribution system contamination threat management problem

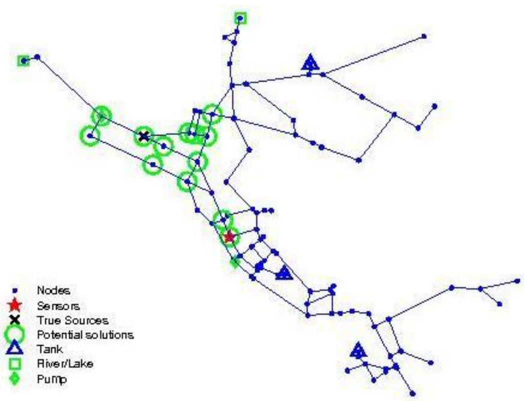
The coupled AHPEA-pEPANET model was applied to solve a contamination source identification problem in a small 97-node WDS network (Network 1). A hypothetical scenario was constructed for this study by synthetically generating observations with the true source located at node 12 of the network. The inverse problem for source identification was solved using six, three, and one threshold-based binary water quality sensors (Kumar et al. 2007) installed in the network. Figure 6.7 shows the solutions of potential source locations obtained using observation data from six sensors, three sensors, and one sensor. All these solutions obtained matched the observed binary sensor signals either perfectly or with a maximum of one mishit. As evident from the results in Figure 6.7, in all cases the potential solutions included the true source as well as other non-unique solutions that matched the data equally well.




(a) Using six sensor




(b) Using three sensors




(c) Using one sensor

**Figure 6.7. Potential contaminant source locations obtained using different numbers of sensors for Network 1**

The model was also applied for the solution of source identification problem in a medium sized network with 1575 nodes (Micropolis network). A non-uniform contaminant injection at a single node lasting one hour was considered. Besides identifying the actual source location, feasible alternative solutions were also identified at 14 other locations in the network in close vicinity of the true source location (Figure 6.8).

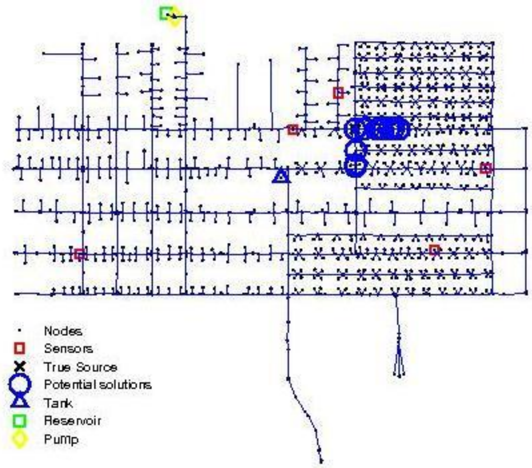
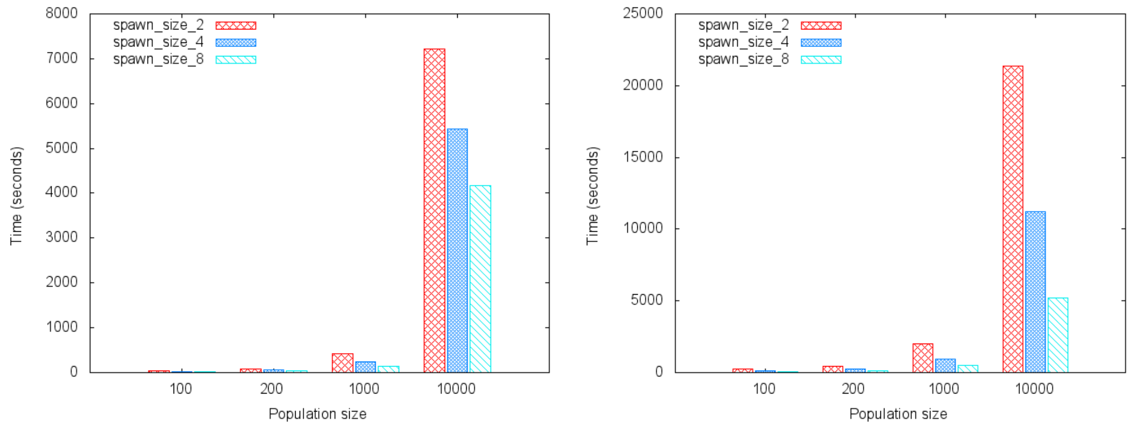



**Figure 6.8 Potential source locations for a contamination scenario for Micropolis network**

Figure 6.9 shows the parallel performance of AHPEA-pEPANET with increasing size of the EA populations when solving the WDS contaminant threat source identification problem in Network 1 and Micropolis network. For this problem majority of the total computation time is spent on pEPANET simulation required for the evaluation of fitness of potential solutions. Therefore, while the resources allocated for AHPEA was kept constant at a single processor core per population, increasing number of processors (2, 4 and 8) were allocated for the simulation model (pEPANET) spawned by each population. Figure 6.9 shows that a decrease in the time to solution was achieved with increasing processor cores for pEPANET. While a nearly perfect scaling was achieved for Micropolis network (Figure 6.9.b), the scaling was relatively poor for Network 1 (Figure 6.9.a). The computation time for Network 1 which is smaller in size is relatively smaller than Micropolis network. However, for any given population size, same amount of communication is required for both networks. Thus Network 1 has a higher communication/computation ratio leading to its poor scaling.




(a) Network 1

(b) Micropolis network

**Figure 6.9 AHPEA-pEPANET scaling for increasing population size when solved on Cygnus system at NCSU**

## 6.5 Summary and conclusions

This research developed and demonstrated an efficient parallel computing framework for Asynchronous Hierarchical Evolutionary Algorithm. AHPEA facilitates an efficient mapping of the populations in the hierarchical topology of the algorithm to an underlying parallel platform with hierarchical memory/communication architecture. A remote memory access based communication scheme using MPI-2 was employed to exploit the inherent asynchronous nature of the algorithm. With these design features, a highly scalable computational framework was developed and the same was demonstrated through deployment on a high performance computing cluster.

The framework was successfully applied for the solution of a high dimensional benchmark test problem and WDS contamination source identification problems for two different networks. Parallel performance results show that the AHPEA framework exhibits reasonable weak scaling for the benchmark test problem and nearly perfect strong scaling for the larger WDS problem.

## Acknowledgment

This work is supported by the National Science Foundation (NSF) under Grant Nos. CMS-0540316. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

## References

Alba, E., Tomassini, M. (2002). "Parallelism and evolutionary algorithms." IEEE Transactions on Evolutionary Computation, Vol. 6, No. 5, 443-462.

Branke, J., Andersen, H., Schmeck, H. (1996). "Global selection methods for massively parallel computers." Proceedings of AISB'96 Workshop on Evolutionary Computing, LNCS 1143, Springer**,** 175-188.

Branke, J., Andersen, H. , Schmeck, H. (2004). "Distribution of Evolutionary Algorithms in Heterogeneous Networks." Genetic and Evolutionary Computation Conference, LNCS 3102, Springer, 923-934.

Chapman, B., Jost, G., van der Pas, R., Kuck, D. J. (2007), Using OpenMP: Portable Shared Memory Parallel Programming. The MIT Press in Scientific and Engineering Computation Series, Cambridge, MA, USA.

Cantú-Paz, E. (1997). "Designing efficient master-slave parallel genetic algorithms." Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign.

Gropp, W., Lusk, E., Skjellum, A. (1999a) Using MPI, 2nd Edition: Portable Parallel Programming with the Message Passing Interface. MIT Press in Scientific and Engineering Computation Series, Cambridge, MA, USA.

Gropp, W., Thakur, R., Lusk, E. (1999b) Using MPI-2: Advanced Features of the Message Passing Interface. MIT Press in Scientific and Engineering Computation Series, Cambridge, MA, USA

Kumar, J., Zechman, E.M., Brill, E.D., Mahinthakumar, G., Ranjithan, S., Uber, J. (2007). "Evaluation of Non-Uniqueness in Contaminant Source Characterization Based on Sensors with Event Detection Methods." *World* Environmental and Water Resources Congress 2007, Tampa, FL.

Kumar, J., Brill, E.D., Mahinthakumar, G., Ranjithan, S. (2008). "Source Identification for Contamination Events Involving Reactive Contaminant." World Environmental and Water Resources Congress 2008, Honolulu, Hawaii.

Kumar, J., Brill, E.D., Mahinthakumar, G., Ranjithan, S. (2009). "Characterizing Reactive Contaminant Sources in a Water Distribution System." World Environmental and Water Resources Congress 2009, Kansas City, Missouri.

Kumar, J., Brill, E.D., Mahinthakumar, G., Ranjithan, S. (2010a). "Identification of reactive contaminant sources in a water distribution system under the conditions of data uncertainties." World Environmental and Water Resources Congress 2010, Providence, Rhode Island.

Kumar, J., Sreepathi, S., Brill, E.D., Ranjithan, S., Mahinthakumar, G. (2010b). "Detection of leaks in water distribution system using routine water quality measurements." World Environmental and Water Resources Congress 2010, Providence, Rhode Island.

Lim, D., Ong, Y.-S., Jin, Y., sendhoff, B., Lee, B.-S. (2007). "Efficient hierarchical Parallel genetic Algorithms using Grid Computing." Future Generation Computer Systems, 23, 658-670.

Liu, P., Lau, F., Lewis, M. J., Wang, C.-l. (2002). "A New Asynchronous Parallel Evolutionary Algorithm for Function Optimization." Parallel Problem Solving from Nature - PPSN VII: 7th International Conference, Granada, Spain, September 7-11, Proceedings, 2002, 401-410.

Rossman, L. A. (2000). "EPANET 2 Users Manual." National Risk Management Research Laboratory, Environmental Protection Agency, Cincinnati.

Sreepathi, S. (2006). "Simulation-Optimization for Threat Management in Urban Water Systems." Fall 2006 Internet2 Meeting.

Sreepathi, S., Mahinthakumar, G., Zechman, E. M., Ranjithan, S., Brill, D., Ma, X., von Laszewski, G. (2007). “Cyberinfrastructure for Contamination Source Characterization in Water Distribution Systems.” Lecture Notes in Computer Science, ICCS2007 Part I, *4487*, 1058-1065.

Tang, K., Yao, X., Suganthan, P.N., MacNish, C., Chen, Y. P., Chen, C. M., Yang, Z. (2008). “Benchmark functions for the CEC’2008 Special session and competition on large scale global optimization.”

http://nical.ustc.edu.cn/conferences/cec2008/lsgo/LSGO.CEC08.Benchmark.zip

# CHAPTER 7. Cyberinfrastructure for contamination threat management in water distribution systems

(A modified version of this chapter is under review for publication as a contributed chapter in the book Dynamic Data Driven Analysis System, Springer-Verlag. Authors: Jitendra Kumar, Sarat Sreepathi, S. Ranji Ranjithan, Downey Brill, Li Liu, G. Mahinthakumar)

## 7.1 Introduction

Urban water distribution systems (WDSs) are vulnerable to accidental and intentional contamination incidents that could result in adverse human health and safety impacts. Identifying the source and extent of contamination (source characterization problem) is usually the first step in devising an appropriate response strategy in a contamination incident. This paper develops and tests a preliminary grid cyberinfrastructure for solving this problem as part of a larger multidisciplinary dynamic data driven application systems project (DDDAS, Darema 2006). The overall goal of the DDDAS project is to develop algorithms and associated middleware tools leading to a full fledged cyberinfrastructure for threat management in WDSs (Mahinthakumar et al. 2006a).

The source characterization problem involves finding the contaminant source location (typically a node in a water distribution system) and its temporal mass loading history (release history) from observed concentrations at several sensor locations in the network. The release history includes start time of the contaminant release in the WDS, duration of release, and the contaminant injection pattern during this time. Assuming that we have a forward simulation model that can simulate concentrations at various sensor locations in the WDS for given source characteristics, the source characterization problem, which is an inverse problem, can be formulated as an optimization problem with the goal of finding a source that can minimize the difference between the simulated and observed concentrations at the sensor nodes. This approach is commonly termed simulation-optimization as the optimization algorithm drives a simulation model to solve the problem. Population based search methods such as evolutionary algorithms (EAs) are popular methods to solve this problem owing to their exploratory nature, ease of formulation, flexibility in handling different types of decision variables, and inherent parallelism. Despite their many advantages, EAs can be computationally intensive as they may require a large number of forward simulations to solve an inverse problem such as source characterization. As the DDDAS project relies heavily on EA based methods (Zechman and Ranjithan 2009; Liu et al. 2006) for solving source characterization and sensor placement problems, an end-to-end cyberinfrastructure is needed to couple the optimization engine to the simulation engine, launch the simulations seamlessly on the grid, and track the solution progress in real-time.

## 7.2 Description of the overall cyberinfrastructure

Solution of simulation-optimization problem requires a close interaction between the simulation and optimization components. Optimization model searches for the decision variables to minimize the error between observed and simulated sensor responses and sends them to the simulation model for fitness evaluation. Simulation model simulates the system for the given decision variables to simulate the sensor responses, calculates and sends back the fitness value. This iterative process is repeated until a near optimal solution is found within the specified termination criteria. Figure 7.1 shows a simplified architecture of the simulation-optimization framework.

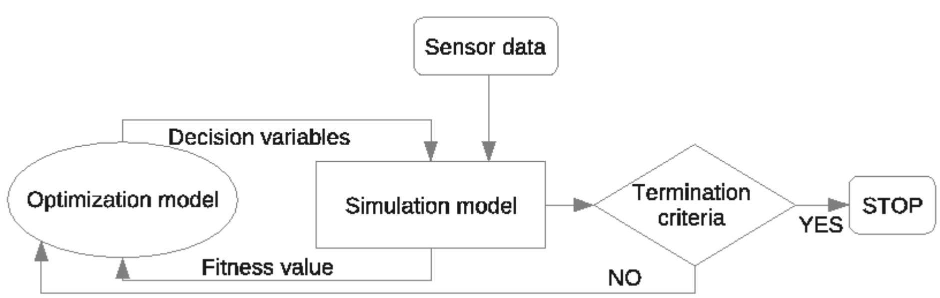



**Figure 7.1 Simulation-optimization framework**

In the present work, two approaches have been developed for the coupling of simulation and optimization components: (a) a file-based communication system to interoperate with existing EA based optimization tools developed in diverse development platforms such as Java (Zechman and Ranjithan 2006) and Matlab (Liu et al. 2006), and (b) an MPI-2 based communication system designed specifically for communication with the new C-based optimization engine. The focus of this paper is on the development of the MPI-2 based communication subsystem and its performance.

### 7.2.1 Simulation module

The simulation engine, EPANET (Rossman 2000) is an extended period hydraulic and water-quality simulation toolkit developed at Environmental Protection Agency (EPA). It is originally developed for the Windows platform and provides a C language library with a well defined API (Sreepathi 2006). The original EPANET was ported to Linux environments and customized to solve simulation-optimization optimization problems by building a wrapper around it. For testing purposes, limited amount of customization was built into the wrapper to solve source identification problems. The wrapper aggregates the EPANET simulations into a single parallel execution for multiple sets of source characteristics to amortize the startup costs and minimize redundant computation.

The parallel version of the wrapper is developed using MPI and referred to as "pEPANET". In the file-based implementation, middleware scripts are designed to invoke multiple pEPANET instantiations depending on resource availability. Within each MPI program, the master process reads the base EPANET input file (WDS network information, boundary conditions etc.). The master process also obtains the source characteristics (i.e., decision variables) from the optimization component via files or MPI-2 communication. The master process then populates data structures for storing simulation parameters as well as the multiple sets of contamination source parameters via MPI_Scatter calls. The contamination source parameter sets are then divided among all the processes equally ensuring static load balancing. Each process then simulates its assigned group of contamination sources successively. At the completion of assigned simulations, the master process collects results from all the member processes using an MPI_Gather and sends it to the optimization component via MPI-2 Send/Recv or a file.

### 7.2.2 Optimization module

Unique nature of the WDS system and insufficient water quality sensor data availability poses several challenges for the optimization problem. An evolutionary algorithm based approach was developed to address these issues in source identification problem.

Zechman and Ranjithan (2004) designed an evolutionary algorithm for generating alternatives (EAGA) based on concepts of co-evolution to generate small number of good but maximally different alternatives. In this algorithm, subpopulations collectively searches for different alternative solutions (Figure 7.2). Each solution is represented by one subpopulation that undergoes an evolutionary search. The survival of solutions in each subpopulation depends upon their performance with respect to the objective function as well as how far they are from the other solutions in the decision space (to identify a solution that is as different, or non-unique, as possible). Thus, the evolution of solutions in a subpopulation is influenced by solutions in the other subpopulations, forcing the migration of each subpopulation towards good but distant regions of the decision space. The concepts of the EAGA were adopted and extended for the solution of WDS source contamination problem.

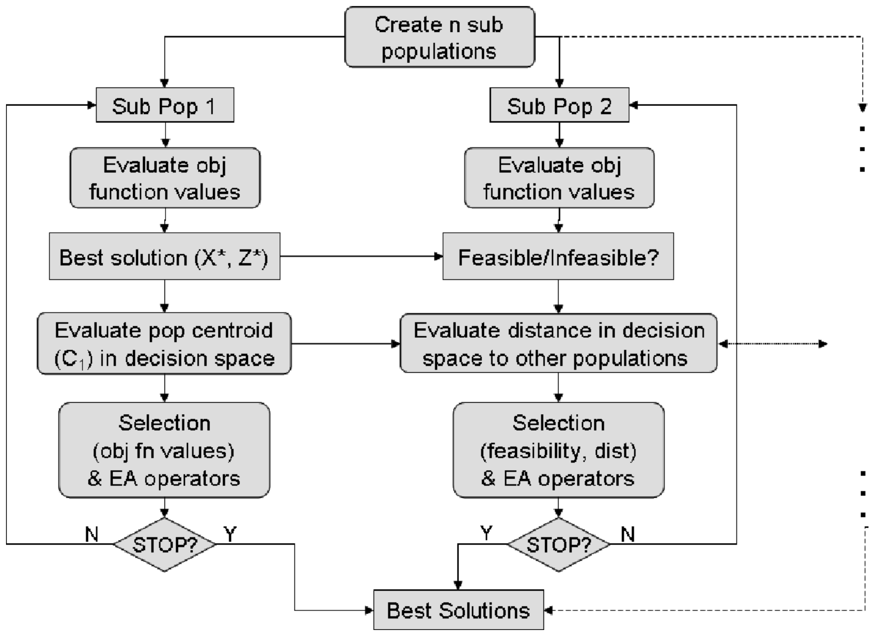



**Figure 7.2 Evolutionary algorithm based optimization model**

The optimization model was originally implemented in Java. However, the poor computational performance of Java (Sreepathi et al. 2007) prompted us to switch to a C-based implementation. Since the simulation model is also written in C, this version allowed us to take advantage of MPI-2 functionalities to carry out a more efficient coupling with the simulation model. A series of tests were performed to validate the algorithmic efficiency of this implementation with the Java version. A MATLAB version of the model was also developed for the purpose of rapid prototyping and simple analysis. In the current cyberinfrastructure all three versions (C, Java, MATLAB) are available for use. For performance reasons, however, the C-based optimization module is generally used for the production runs.

### 7.2.3 Coupling the simulation and optimization components

#### 7.2.3.1 File based coupling

In the file-based coupling, interactions between the simulation and optimization components which are independent executable programs are established through files. During every generation of the evolutionary search, the optimization models writes the decision variables of the candidate solutions to a file and launches the simulation model by the means of system call in the new C-based optimization module or via invocation of shell scripts for Java and Matlab optimization modules. The file is then moved to the location of the simulation component via a shell command. The master process in the simulation component then reads this file and distributes the decision variable sets to the slave processors for simulating the hydraulics and transport in the WDS in parallel. The master collects the evaluated fitness values from all the slaves and writes it to another file. This file is then moved to the location of the optimization component which reads this file and prepares the decision variable set for the next generation. The file-based communication system is schematically illustrated in Figure 7.3.

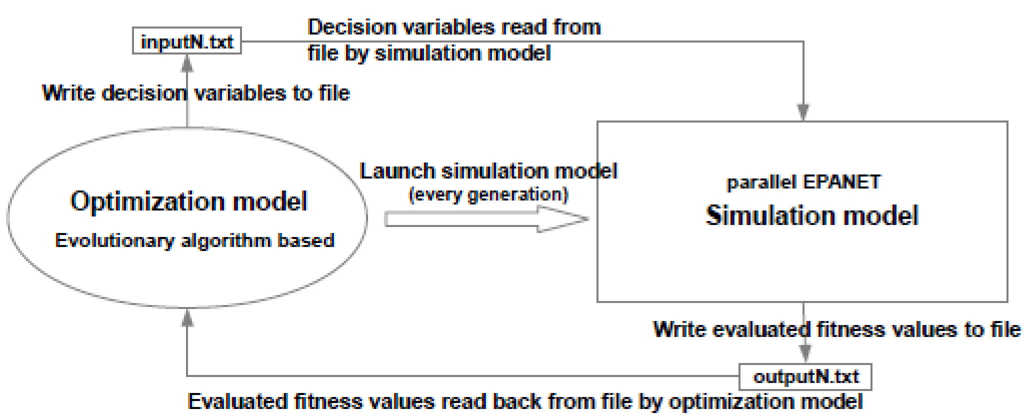



**Figure 7.3 File based coupling of simulation-optimization models**

*Persistency Mechanism:* The evolutionary computing based optimization methods that are currently in use within the project exhibit the following behavior. The optimization method submits some evaluations to be computed (generation), waits for the results and then generates the next set of evaluations that need to be computed. If the simulation program were to be separately invoked every generation, it needs to wait in a batch environment for acquiring the requisite computational resources. In the MPI-2 based communication system this is not a problem as the intercommunicator already allows the simulation component to remain alive while the optimization component is working on the next set of evaluations. However, in the file based system, a special persistency mechanism needs to be built in for the simulation component to remain active. In this case, the program needs to wait in the queue just once when it is first started. Hence pEPANET was enhanced to remain persistent across generations. In addition to amortizing the startup costs, the persistent wrapper significantly reduces the wait time in the job scheduling system. The persistent wrapper achieves this by eliminating some redundant computation across generations. One all evaluations are completed for a given generation (or evaluation set) the wrapper maintains a wait state by polling periodically for a sequence of input files whose filenames follow a pattern. The polling frequency can be tuned to improve performance. This pattern for the input and output file names can be specified as command line arguments facilitating flexibility in placement of the files as well as standardization of the directory structure for easy archival.

*Job submission middleware:* Consider the scenario when the optimization toolkit is running on a client workstation and the simulation code is running at a remote supercomputing center. Communication between the optimization and simulation programs is difficult due to the security restrictions placed at current supercomputing centers. The compute nodes on the supercomputers cannot be directly reached from an external network. The job submission interfaces also differ from site to site. In light of these obstacles, a middleware framework based on Python has been developed to facilitate the interaction between the optimization and simulation components and to appropriately allocate resources. The middleware component utilizes public key cryptography to authenticate to the remote supercomputing center from the client site. The middleware then transfers the file generated by the optimization component to the simulation component on the remote site using available file transfer protocols. It then waits for the computations to be completed at the remote sites and then fetches the output file back to the client site. This process is repeated until the termination signal is received from the client side (in the event of solution convergence or reaching iteration limit). The middleware script also polls for resource availability on the remote sites to allocate appropriate number of processors to minimize queue wait time by effectively utilizing the backfill window of the resource scheduler. When more than one supercomputer site is involved, the middleware divides the simulations proportionally among the sites based on processor availability and processor speed. A simple static allocation protocol is currently employed.

#### 7.2.3.2 MPI-2 based coupling

This approach uses the MPI-2 dynamic process creation and management routines for the coupling of the simulation and optimization models. A user starts the optimization model with the information regarding the available resources for the computation. The optimization model spawns the simulation model as a child process using MPI-2 process creation calls. An intercommunicator is established between the parent process (optimization model) and the child process (simulation model). Optimization model sends the decision variables to the simulation model using MPI_Send/MPI_Recv via the intercommunicator. The decision variables are received by the master process in the simulation model which in turn distributes them among the slave processes for evaluation. The evaluated fitness values are collected by the master and sent back to the parent process (optimization model) using MPI_Sends/Recvs. This process is shown schematically in Figure 7.4.

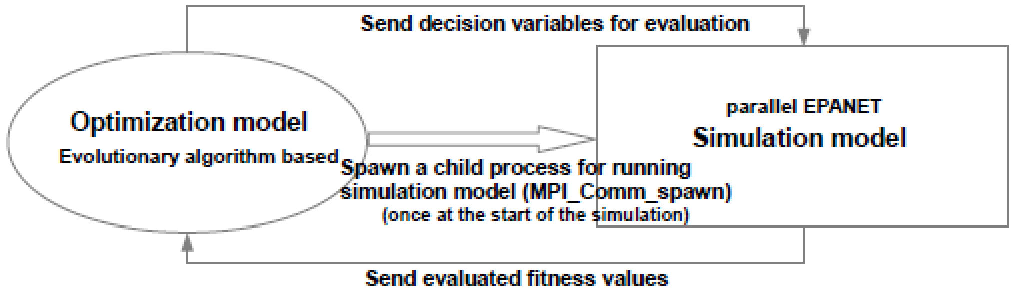



**Figure 7.4 MPI-2 dynamic process based simulation-optimization coupling**

## 7.3 Application case study

Simulation-optimization framework developed in this study was applied for the source reconstruction during a contamination event. Observed water quality data from the installed sensors are used for the identification of the source characteristics. In ideal condition, accurate and continuous observation of the contaminant concentration is desired from the sensors. However, with current sensor technology, many sensors are only able to provide a contamination or no-contamination binary response during an event. Kumar et al. (2007) developed a methodology for source identification using binary sensor data. The probe of a binary sensor detects the concentration of a contaminant in the system and triggers a contamination alarm if the contaminant is present. The accuracy of detection depends on the sensitivity of the sensor probe i.e., the minimum concentration of the contaminant it can detect. Sensor issues a detection signal as *1* if the concentration of contaminant if above the trigger value and a no-detection signal of *0* otherwise. Optimization model searches for the source characteristics which produces a response at sensors similar to the observed data. A predicted sensor signal is termed as miss-hit if it is not the same as observed i.e., detection instead of no-detection or vice versa. Optimization model is formulated to minimize the number of mishits in the predicted sensor response.

The optimization problem for the source characterization problem using binary sensors can be formulated as:

$$Minimize\ E = \sum_{i=1}^{N_S} \sum_{t=1}^{T_S} \left| BS_{i,t}^{s} - BS_{i,t}^{a} \right|$$

$$subject\ to\ \ BS_{i,t}^{s}\ (simulated\ signal) = \begin{cases} 0\ if\ \ C_{i,t}^{s}(L, T_0, C_0) < C_{th} \\ 1\ if\ C_{i,t}^{s}(L, T_0, C_0) \geq C_{th} \end{cases}$$

$$where \qquad BS_{i,t}^{a}(actual\ signal) = \begin{cases} 0\ \ if\ C_{i,t}^{a}\ < C_{th} \\ 1\ \ if\ \ C_{i,t}^{a}\ \geq C_{th} \end{cases}$$

Where, $BS_{i,t}^{a}, BS_{i,t}^{s}(L, T_0, C_0)$ are actual and simulated binary signals respectively at sensor $i$ at time $t$. $C_{i,t}^{a}, C_{i,t}^{s}(L, T_0, C_0)$ are actual and simulated continuous concentration values, respectively at sensor $i$ at time $t$. $C_{th}$ is the sensor threshold concentration above which it issues detection signal and no-detection signal otherwise. $T_S$ is the total time for which network is being simulated and $N_S$ the number of sensors installed in the network. $L$ is the location of the source in the network, $T_0$ the time of start of contaminant injection and $C_0$ the contaminant injection pattern. $E$ is the error between the actual and simulated binary signals at the sensors. Error to be minimized is calculated as the sum of absolute errors between actual and simulated signals at each time step during the simulation period over all the installed sensors in the network.

### 7.3.1 Study network

Micropolis is a virtual city network (Figure 7.5) developed by Brumbelow et al. (2007) as a realistic, publicly available water distribution system datasets for research studies without compromising public security. Micropolis mimics a small city of approximately 5,000 populations in a historically rural region. It is served by a single 440,000 gallon (1,670,000 liters) elevated storage tank. The water distribution system consists of 1236 nodes, 575 mains, 486 service and hydrant connections, 197 valves, 28 hydrants, 8 pumps, 2 reservoirs, and 1 tank. 458 demand nodes are composed of 434 residential, 15 industrial, and 9 commercial/institutional users. Diurnal demand pattern were defined with hourly time interval. Total daily demand of the system was 1.20 MGD (4.54 Ml/day) with minimum and maximum hourly demand of 0.68 MGD (2.57 Ml/day) and 1.66 MGD (6.28 Ml/day). For simulating the contamination scenario in the Micropolis network, water quality simulations were carried out at 5 minutes interval using EPANET. Five threshold based binary sensors were placed in the network based on engineering judgment.

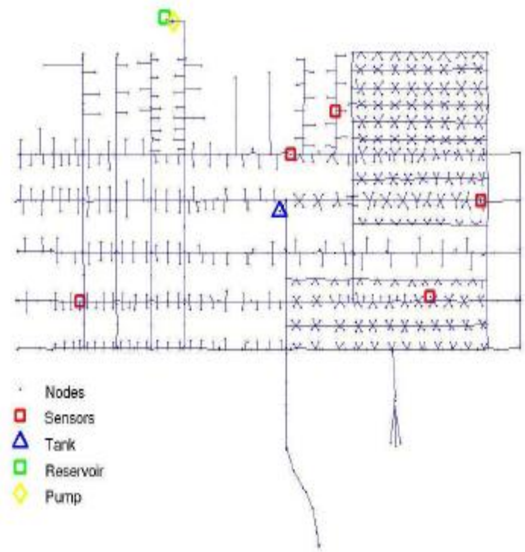
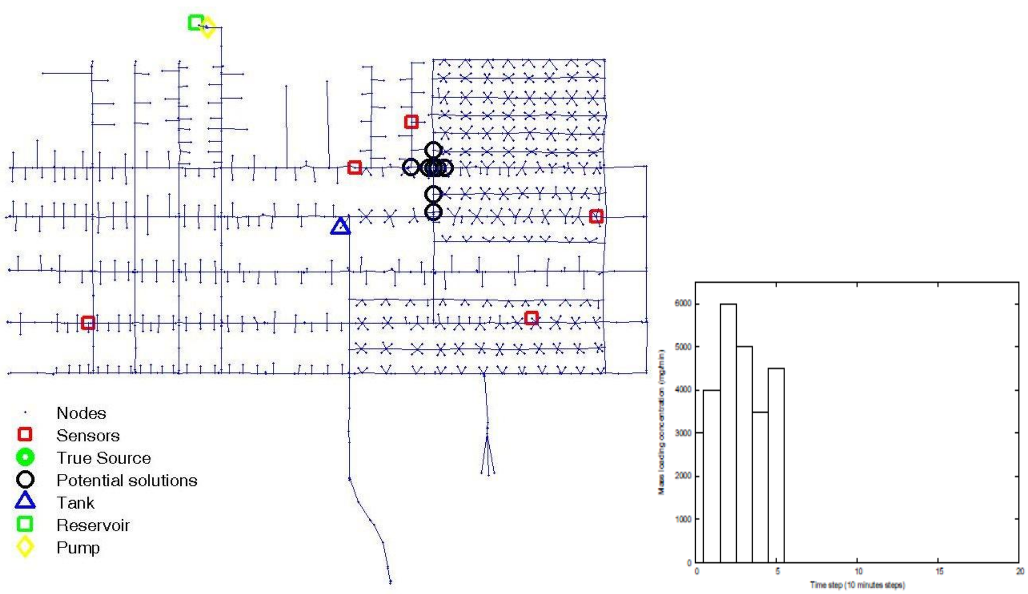



**Figure 7.5 Micropolis water distribution network**

### 7.3.2 Application results

The developed framework was employed for the solution of contamination source identification problem in the Micropolis network. The hypothetical study was carried out using a nonreactive conservative pollutant. Observations from five water quality sensors installed in the network were used for characterizing the contamination source and was solved using the developed simulation-optimization framework. Sensors of sensitivity 0.1 mg/l were used in the current study. Evolutionary algorithm based model with fifty subpopulations was employed with first sub population searching for the best possible solution while other devoted to search for all alternative solutions possible within a 10% error range.

In this scenario contaminant source was placed at node 653 with contamination loading profile as shown in Figure 7.6b. 10 random trials were carried out using the developed framework for identifying the contamination source characteristics. Solutions were identified at true source location of 653 during all trials. Best solution in first subpopulation was identified with a mean error of 2.0 mishits (standard deviation of 0.0 mishits). While searching for maximum of 50 alternatives, on average 36 alternatives (standard deviation 0.0) were found during 10 random trials to be feasible with average error of 0.75 mishit (standard deviation 0.0 mishit). Besides the true locations of 653, the solutions were also found at 7 other locations close to the true source location (Figure 7.6a). Limitation over the quantity and quality of data available from the sensors introduces some uncertainty in source identification; however the methodology was able to narrow down to a small region very close to the source which would be helpful for the utility managers in devising contaminant containment strategies.




(a) Contamination scenario and potential source locations (b) Contaminant injection pattern

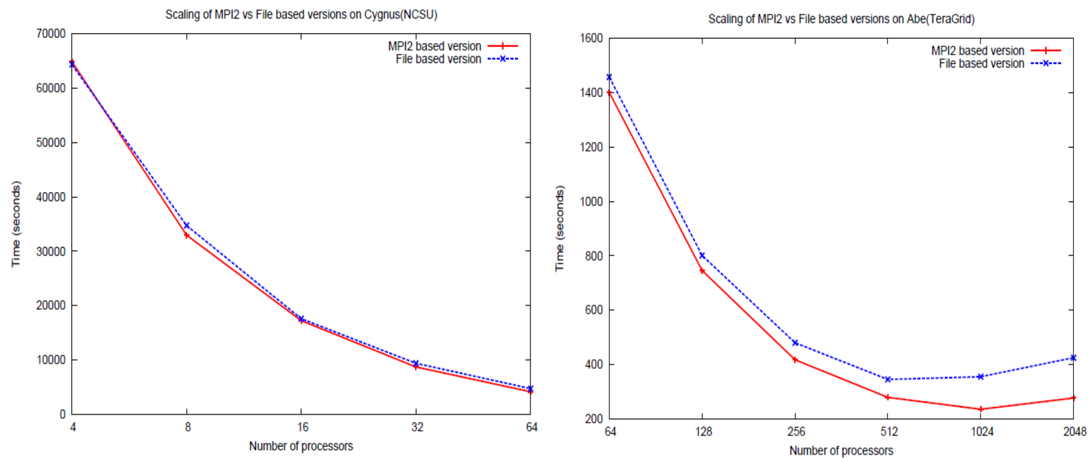
**Figure 7.6 Study contamination scenario and results**

## 7.4 Parallel performance results

Performance analysis of the two versions of the simulation-optimization framework was carried out on two different computing platforms available at NC State University and National Center for Supercomputing Applications (NCSA), TeraGrid (Table 7.1). Comparable performance of the two versions were observed on 'Cygnus' at NC State (Figure 7.7a) for up to 64 processors. MPI2 dynamic process based version avoids the file I/O overheads involved in the file based coupling by direct message passing (communication) between simulation and optimization components. For smaller number of processors, the communication cost of message passing between the two components almost equals the I/O overhead of the file based version. However, significant improvements in turnaround time was achieved for the MPI2 dynamic process based version at higher processor counts on the 'Abe' cluster at TeraGrid NCSA (Figure 7.7b).

**Table 7.1 Computational resources used in the study**

| | TeraGrid NCSA Abe | Cygnus, NC State University |
|---|---|---|
| Architecture | Blades (Dell PowerEdge 1955) | AMD Quad Core Opteron 2350 |
| Number of nodes | 1600 | 11 |
| Processor | Intel 64 (Clovertown)<br>2.33 GHz dual socket Quad Core | Two 2.0 GHz Quad Core<br>AMD Opteron |
| Memory | 600 nodes with 8GB (1GB per core)<br>600 nodes with 16GB (2Gb per core) | 2Gb per core |
| Network interconnect | InfiniBand | GigE |
| Filesystem | Lustre | Network File System (NFS) |




(a) Cygnus (NCSU) (b) NCSA 'Abe' (TeraGrid)

**Figure 7.7 Scaling study of file based and MPI-2 based versions**

The MPI2 dynamic process based framework was studied further for the time spent in computation in the simulation and optimization model, communication, and I/O operations.

*Computation time in simulation and optimization models:* The optimization model used in the current framework works in a serial mode and thus the time spent in the optimization component remains constant. Percentage contribution of time spent in optimization model thus increases with decreased total simulation time as we scale to larger number of processors (Figure 7.9). Computation time in simulation model decreases as the scenarios to be evaluated are distributed among a larger number of processes, each with smaller number of evaluations to process.

*Communication time:* Communication cost in the application was very small for up to 256 processors Figure 7.9. However, the overhead due to communication increases at larger processor counts.

*I/O time:* A decreasing trend for the time spent in I/O operations was observed with increase in the number of processors for the MPI2 based version (Figure 7.8, Figure 7.9). This is due to the fact that each process of simulation model (pEPANET) uses a temporary file created in the temp file space to read/write hydraulics data at each simulation time step. As the number of slave processors grows, each processor is assigned a smaller number of evaluations and thus decreased number of accesses (read/write) to the temporary file system reducing the overall I/O overhead. The same holds true even for the file based version. However, in file based version interaction between the simulation and optimization models is achieved via files which are to be read and written from the parallel file system. The time spent in the file I/O in file based coupling model increases as the number of generation in the optimization model increases, limiting the scalability of the application.

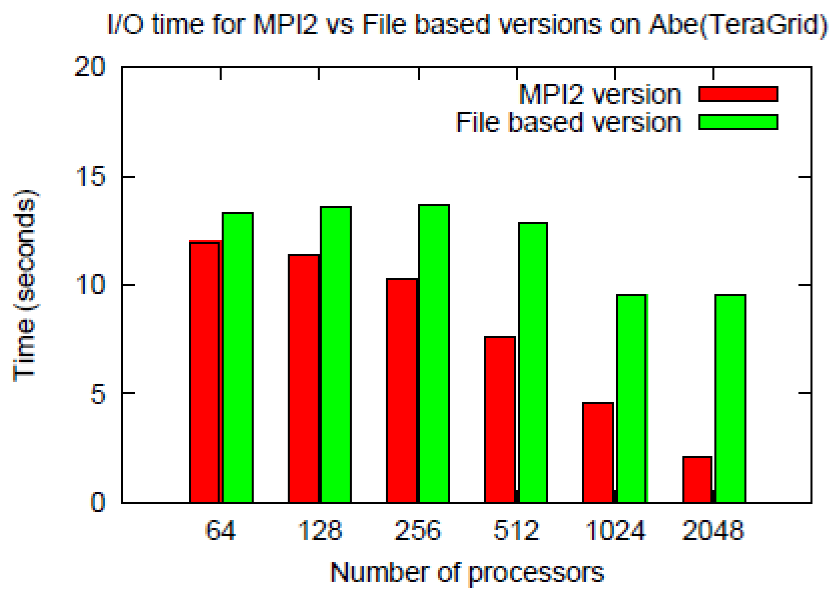



**Figure 7.8 Percentage of time spent in I/O operations for the two versions (Abe, NCSA TeraGrid)**

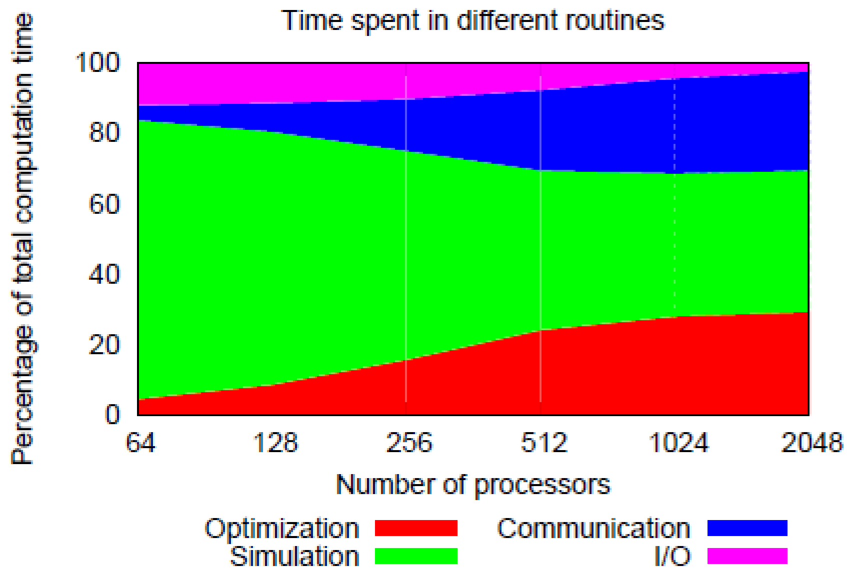



**Figure 7.9 Percentage time spent in different routines for MPI-2 version**

## 7.5 Conclusion and future work

An end-to-end solution for solving WDS contamination source characterization problems in parallel and grid environments has been developed as part of this DDDAS project. This involved coarse-grained parallelization of simulation module and middleware for seamless grid deployment. In addition, a number of algorithmic and problem related advancements were carried out including a new dynamic optimization procedure, investigation of non-uniqueness in the source identification problem, and a number of problem investigations targeting reactive contaminants and binary sensors.

Various performance optimizations such as improving processor placements, minimizing file system overheads, eliminating redundant computations, and amortizing queue wait times were carried out to improve turnaround time. Even with these optimizations, the file movement overheads were significant when the client and server sites were geographically distributed as in the case of TeraGrid. To address this problem, the original Java based optimization module was rewritten in C. In addition to achieving a nearly a factor of 10 improvement in the optimization computation time, this also permitted us to optionally use MPI-2 communication calls to connect to the simulation component. Strong scaling studies up to 2048 processors of the TeraGrid cluster at NCSA indicated that the use of MPI-2 communication significantly improved the scalability of the application beyond 1024 processors when compared to the file based version. Several future improvements are planned including optimization algorithm changes that can allow for overlapping file movements and optimization calculations with simulation calculations, localizing optimization calculations on remote sites by partitioning techniques, and minimizing file transfer overhead using grid communication libraries.

## Acknowledgment

This work is supported by the National Science Foundation (NSF) under Grant Nos. CMS-0540316. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

## References

Brumbelow, K., Torres, J., Guikema, S., Bristow, E., Kanta, L. (2007). “Virtual Cities for Water Distribution and Infrastructure System Research.” World Environmental and Water Resources Congress, 2007, Tampa, FL.

Darema, F. (2006). “Introduction to the ICCS 2006 Workshop on Dynamic data driven applications systems.” Lecture Notes in Computer Science, 2006, *3993*, 375-383

Kumar, J., Zechman, E.M., Brill, E.D., Mahinthakumar, G., Ranjithan, S., Uber, J. (2007). “Evaluation of Non-Uniqueness in Contaminant Source Characterization Based on Sensors with Event Detection Methods.” World Environmental and Water Resources Congress 2007, Tampa, FL.

Liu, L., Zechman, E. M., Brill, E. D., Mahinthakumar, G., Ranjithan, S., Uber, J. (2006). “Adaptive Contamination Source Identification in Water Distribution Systems Using an Evolutionary Algorithm-based Dynamic Optimization Procedure.” Water Distribution Systems Analysis Symposium, Cincinnati, OH.

Mahinthakumar, G., von Laszewski, G., Ranjithan, S., Brill, E. D., Uber, J., Harrison, K. W., Sreepathi, S., Zechman, E. M. (2006). “An Adaptive Cyberinfrastructure for Threat Management in Urban Water Distribution Systems.” Lecture Notes in Computer Science, International Conference on Computational Science, 401-408.

Rossman, L. A. (2000). “EPANET 2 Users Manual.” National Risk Management Research Laboratory, Environmental Protection Agency, Cincinnati.

Sreepathi, S. (2006). “Simulation-Optimization for Threat Management in Urban Water Systems.” Fall 2006 Internet2 Meeting**.**

Sreepathi, S., Mahinthakumar, G., Zechman, E., Ranjithan, S., Brill, E. D., Ma, X., von Laszewski, G. (2007), “Cyberinfrastructure for Contamination Source Characterization in

Water Distribution Systems." Lecture Notes in Computer Science, ICCS2007 Part I, *4487*, 1058-1065.

Zechman, E. M. and Ranjithan, S. (2004). "An evolutionary algorithm to generate alternatives (EAGA) for engineering optimization problems." Engineering Optimization, 36(5), 539-553.

Zechman, E. M. and Ranjithan, S. (2009). "Evolutionary Computation-Based Methods for Characterizing Contaminant Sources in a Water Distribution System." Journal of Water Resources Planning and Management, 135, 334-343.

# CHAPTER 8. Summary and conclusions

This research developed a comprehensive simulation-optimization framework and associated methodologies for contamination threat management in water distribution systems (WDSs). New methodologies were developed for contaminant source identification in WDSs for a wide range of scenarios including imperfect sensors, reactive contaminants, and multiple sources. In addition, a new parallel evolutionary optimization algorithm, AHPEA, was developed to address the inverse problem of contaminant source characterization and the general class of large scale global optimization problems. The methods and test cases were implemented into a massively parallel computational framework to enable a real-time contaminant threat management for WDSs. Activities and findings for different aspects of this research are summarized below.

## 8.1 Methodologies for imperfect sensors

Contaminant source characterization is typically carried out using the observation data collected at the various water quality sensors installed in the network. However, number and type of installed water quality sensors in any real WDS widely varies and is driven by cost associated with the installation and maintenance of these sensors. The effect of varying quantity and quality of data on the source characterization problem was studied in detail in this research. Methodologies for source characterization was developed using the data from continuous as well as filtered threshold based binary sensors. These methodologies were used to identify the location of source, time of start and pattern of contaminant injection for a contamination event. Issue of non-uniqueness caused due to the limited observation was addressed. Testing of the methodologies was carried out using a number of hypothetical scenarios using several test networks. Detailed studies were also carried out to account for the sensor measurement errors, data noise and uncertainties experienced in the field. A wide range of scenarios were designed to test the developed methodology for accuracy and robustness. The developed simulation-optimization methodology was successfully demonstrated to address the source characterization problem including identifying all potential non-unique solutions based on the available data. Even under the conditions of noise and uncertainties, the approach was able to locate the source to a small region in the network; this information could be subsequently used by utility personnel to carry out ground operations.

## 8.2 Reactive contaminants

The external contaminant injected in the system during a contamination event remains unknown and can be reactive chemical, biological or radioactive substance. However, if a contaminant which is reactive with chlorine is injected in the system, it would interact with the chlorine which is used as a common disinfectant in the system and cause its increased decay. A methodology was developed in his research to use the chlorine monitoring data as surrogate to identify the presence of a contaminant in the system and for the inversion of the source characteristics. However, kinetics of the reaction between the contaminant and chlorine remains unknown and may vary due to highly dynamic flows in the system. An approach was developed to address the source contamination under the uncertain reaction conditions. Issue of data noise and model/data uncertainties were studied in detail. Developed approach was demonstrated to be effective in indentifying the source characteristics and addressing the issue of non-uniqueness even under conditions of noise and uncertainties for reactive contaminants in the system.

## 8.3 Parallel optimization method

The source inversion problem is solved as an optimization problem in the simulation-optimization approach used in this research. However, the optimization problem becomes increasingly complex and computationally intensive as the size of the network grows. To address complex large-scale optimization problems such as this, a new Asynchronous Hierarchical Parallel Evolutionary Algorithm (AHPEA) was developed. Design of hierarchical population topology used in AHPEA was motivated by the hierarchical evolution process occurring in nature and also by the hierarchical memory architecture of modern parallel computers. AHPEA was designed for the solution of large-scale global optimization problems with a special focus on parallel performance and scalability. AHPEA was first demonstrated for a suite of large-scale global optimization benchmark test problems to provide high quality of solutions. It was then extended to the WDS application by successfully applying it to solve source identification problems.

## 8.4 Parallel implementation

AHPEA was integrated into the simulation-optimization framework resulting in a fully asynchronous system suitable for execution on massively parallel architectures. . The framework uses the information about the underlying computer architectures to enable a localized communication/migration scheme in the AHPEA. Asynchronocity in the algorithm was implemented using the MPI-2 one sided communication protocols. The WDS simulation model (pEPANET) and the methodologies for source characterization were seamlessly coupled with the AHPEA optimization model using MPI-2 dynamic process creation functions. The framework was deployed and tested on a variety of cluster resources and supercomputers showing excellent performance and scalability for WDS source identification problems.

## 8.5 Broader impact

Since the methodologies were successfully demonstrated for a wide range of contamination scenarios on several test water distribution networks, they can be easily extended to problems involving real water distribution networks. The AHPEA framework was designed and tested for general class of large scale global optimization problems and thus can applied to address ay complex engineering design and management problems. The computational framework was developed using industry standard tools and libraries for shared and distributed memory programming in UNIX/LINUX environment. And thus can be readily ported to any cluster or supercomputer to address large-scale global optimization problems. The highly efficient and parallel simulation-optimization framework developed in this research would be able to address a wide range of complex real world engineering problems.