SECURE DATA PROVENANCE FOR IOT DEVICES

EBELECHUKWU NWAFOR

Department of Computer Science

APPROVED:

Gedare Bloom, Chair, Ph.D.

Wayne Patterson, Ph.D.

Gloria Washington, Ph.D.

Robert RubenGuira, Ph.D.

Pablo Arenaz, Ph.D.
Dean of the Graduate School

©Copyright

by

Ebelechukwu Nwafor

2016

to my

MOTHER and FATHER

thanks for all the love and support, i am forever grateful

SECURE DATA PROVENANCE FOR IOT DEVICES

by

EBELECHUKWU NWAFOR, M.S.

DISSERTATION PROPOSAL

Presented to the Faculty of the Graduate School of

Howard University

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

Department of Computer Science

HOWARD UNIVERSITY

FEB 2016

Acknowledgements

First and foremost, I would like to thank God for making this possible, without him i will not be where i am today. I would also like to thank my parents, Benneth and Chinwe for their constant encouragement, love and support.

I would like to thank my advisors Dr. Gedare Bloom and Dr. Legand Burge for their guidiance and encouragement.Also for believing in me and my research ideas. I would like to thank my lab partner Habbeeb Olufowobi and fellow graduate student Marlon Mejias for their constant constructive criticism and valuable input in various drafts revisions of my proposal.

Abstract

The concept of Internet of Things (IoT) offers immense benefits by enabling devices to leverage networked resources thereby making more intelligent decisions. The numerous heterogeneous connected devices that exist throughout the IoT system creates new security and privacy concerns. Some of these concerns can be overcome through data trust, transparency, and integrity, which can be achieved with data provenance. Data provenance also known as data lineage provides a history of transactions that occurs on a data object from the time it was created to its current state.Data provenance has immense benefits in detecting and mitigating current and future vulnerability attacks and has applications in anomaly detection, access control, and digital forensics.

This dissertation looks at provenance with a focus on IoT devices. It takes a holistic approach in the creation, security and applications of provenance data. We create a provenance aware system for IoT devices. This ensure trust and helps establish causality for decisions and actions taken by an IoT connected system. As a result of the amounts of data that is generated from our data provenance system, there arises an issue of running out of memory.To address this issue,we look at prior work done in the area of data compression and graph summarization. We address this by proposing a novel data pruning technique.We conclude by looking at the applications of provenance for security of malicious attacks and anomaly detection of malicious threats.

Table of Contents

Chapter 1

Introduction

According to the oxford English dictionary, provenance can be defined as the place of origin or earliest known history of something. An example of provenance can be seen with a college transcript. A transcript can be defined as the provenance of a college degree because it outlines all of the courses satisfied in order to attain the degree.

In the field of computing, Data provenance also known as data lineage is the history of transformations on a data object from the beginning of time to its current state. An example of provenance for software systems is...Provenance denotes the who, where and why of data.Provenance data is represented as an acyclic graph which denotes casual relationship and dependencies between entities.Provenance ensures trust and integrity of data. The information in which provenance offers can be used in preventing malicious attacks. The internet of things(IoT) can be defined as a network of heterogeneous devices communicating together. With the recent data explosion,information is disseminated at every communication level that exists. From mobile devices to desktop computers and servers.Making IoT systems provenance aware is of the essence because it ensures trust and integrity of systems. Enabling IoT device provenance aware allows devices to be able to capture information such as the who, where and how transformations occur on a data object enabling us to be able to trace back in the vent of a malicious attack.

1.1 Motivation

According to a report released by Cisco [4], it is estimated that a total of 50 million devices will be connected to the internet by 2020. With these heterogeneous devices connected,

security and privacy risks increase. Rapid7 [5] discovered that vulnerabilities exist in baby monitors that allowed attackers unauthorized remote access to these devices whereby an attacker can remotely view live video feeds. Having a provenance aware system will be beneficial in this situation since we have a record of input and output operations performed on the device, we can be able to look back on operations performed on the device to determine who, where, and how a malicious activity occurred. Devices (things) connected in an IoT network are embedded systems, which require lightweight and efficient solutions compared with general purpose systems. This requirement is attributed to the constrained memory and computing of such devices. A major issue arises in ensuring that data is properly secured and disseminated across the IoT network. The vast amount of data generated from IoT devices requires stronger levels of trust which can be achieved through data provenance. Provenance has immense benefits in the IoT. Data provenance ensures authenticity, integrity and transparency between information disseminated across an IoT network. Security applications such as anomaly detection, digital forensics, and access control can be further enhanced by incorporating data provenance in IoT devices. The goal of data provenance is to determine causality and effect of actions or operations performed on data. Provenance ensures transparency between things connected in IoT systems. By creating data transparency, we can trace information to determine where, if and when a malicious attack occurs. To achieve transparency, we propose a secure provenance aware system that provides a detailed record of all data transactions performed on devices connected in an IoT network.

1.2 Research Questions

collecting provenance data in IoT devices raises some research questions. Some of these questions are outlined below:

- Memory constraints,

- what approach to use. access control with IoT.
- Collecting system level provenance in embedded systems.
- Do we use the OPM approach or something else. Querying provenance data.

1.3 Research Contribution

This dissertation proposes the following contributions:

- A provenance collection framework which denotes causality and dependencies between entities in an IoT system.This system creates groundwork for capturing and storing provenance data.
- A novel framework for Data pruning.This addresses the memory overflow of the memory constrained IoT devices.
- A framework for providing anomaly detection using provenance data in an IoT system.

1.4 Organization of Dissertation proposal

The remaining portion of the dissertation proposal is organized as follows. Chapter 2 talks about background information on data provenance, some of the techniques of collecting system level provenance, Data pruning techniques and also provenance based access control, Provenance data model. chapter 3 discusses our proposed provenance collection system and focuses specifically on preliminary work done in creating a provenance aware system. Chapter 5 concludes the proposal and discusses the proposed framework and projected timeline for completion.

Chapter 2

Background and Related Work

This section outlines some of the work done in the area of data provenance for file systems and also data compression techniques for data provenance, and providing access control for using data provenance.

2.1 Overview of Provenance Aware systems

There have been a considerable amount of work done on collecting provenance data. Some of which has been focused on databases, sensore networks and system level provenance but so far little attention has been given to provenance in the IoT. Some the previous work done on data provenance collection are outlined below:

2.1.1 Provenance Aware Storage System(PASS)

This was developed at Harvard by MuniswamyReddy et al. There are two versions PASS v1 and PASS v2. v1 allows... while version 2...Provenance information is stored in the same location as the file system for easy accessibility, backup, restoration, and data management. Provenance information is collected and stored in the kernel space. The system also provides provenance query capabilities and cycle detection. PASS systems recognizes data pruning for efficient storage, which is in line with our research goal but data pruning was not fully explored in PASS.

The collector keeps track of system level provenance. It intercepts system calls which are translated into provenance data and is stored in an in kernel database. This database maps key value pairs for provenance data for fast index look up. It also provides functionalities

for querying provenance data in the database. PASS detects and eliminates cycles that might occur in provenance dependencies as a result of version avoidance.cycles violates the dependency relationships between entities. For example, a child node could depend on a parent node and also be an ancestor of a parent node. It achieves this by merging processes that might have led to the cycles.

2.1.2 HiFi

Bates et al. [2] developed system level provenance information for the Linux kernel using a Linux Provenance Modules, which tracks whole system provenance including interprocess communication, networking, and kernel activities. This is achieved by mediating access to kernel objects. Linuex Security Model is a framework that was designed for providing custom access control into the Linux kernel.It consists of a set of hooks which is executed before access decision is made.

HiFi contains three components, provenance collector, provenance log and provenance handler. collector and log are in the kernel space while the handler is in the user space.The log is a storage medium which transmits the provenance data to the user space.The collector contains the LSM which resides the kernel space. The collector records provenance data and writes it to the provenance log.The handler intercepts the provenance record fromt he log processes the data and stores it to the provenance record. LSM was designed to avoid problem created by direct system call interception. The provenance information from the kernel space is securely transmitted to the provenance recorder in the user space. This approach to collecting provenance data differs from our work since we focus on embedded systems and are concerned with input and output (I/O) data, which primarily involve sensor and actuator readings.

2.1.3 RecProv

creates a provenance system which records user level provenance thereby avoiding the overhead incurred by kernel level provenance recording.It uses mozilla rr to perform deterministic record and replay by capturing system calls and non deterministic input. It also ensure the integrity of provenance data up till the point that the host is compromised by trace isolation. Mozilla rr relies on ptrace which intercepts system call during context switch.Mozilla rr is a debugging tool for linux browser. It is developed for the deterministic recording and replaying of firefox browser in linux. System calls such as execve, clone, fork, open, read, write, clode, dup, mmap, socket, connect, accept are recorded. the provenance information generated in converted into PROV-JSON a W3C standard for relaying provenacne information and also stores provenance data in Neo4j a graph database for visualization of provenance graphs. It does not require changes to the kernel like most provenance monitoring systems (include citations of other provenance monitoring systems that require kernel modification). it generates 20 percent overhead which is acceptable according to the authors.

Recprov uses PTRACE_PEEKDATA from PTRACE to access the derefrenced address of the traced process from the registers.

2.1.4 StoryBook

Spillance et al developed a user space provenance collection system, Storybook [14] that allows the collection of provenance data from the user space thereby reducing performance oververhead from kernel space provenance collection. This system is modular.It allows the use of application specific extensions allowing additions such as database provenance, system provenance, and web and email servers. It achieves provenance capture by using FUSE for system level provenance and MYSQL for database level provenance capture. Story Book allows developers to implememtn provenance inspectors. these are custom provenance models which captures the provenance of applications which are often modified

by different application(e.g web servers, databases). When an operation is performed on a data object, the appropriate provenance model is triggered and provenance data for that data object is captured. Sotrybook stores provenance information such as open, close, read or write, application specific provenance, causality relationship between entities contained in the provenance system(?). Provrenance is stored in key value pairs and It uses Fable as the storage backend. Storybook allows for provenance query.It achieves this by looking up inode in the file, ino hashtable.

2.1.5 Trustworthy whole system provenance for Linux kernel

2.1.6 Towards Automated Collection of Application-Level Data Provenance

2.1.7 user space provenance with Sandboxing

2.1.8 Provenance for Sensors

Lim et al. [3] developed a model for calculating the trust of nodes contained in a sensor network by using data provenance and data similarity as deciding factors to calculate trust. The value of provenance signifies that the more similar a data value is, the higher the trust score. Also, the more the provenance of similar values differ, the higher their trust score. This work differs from our approach since the authors focus on creating a trust score of nodes connected in a sensor network using data provenance and do not emphasize how the provenance data is collected. We are focused on creating a secure provenance aware system for I/O operations which is used to ensure trust of connected devices.

2.2 Model for representing provenance for IoT

In order to generate provenance, we have to satisfy the who, where, how, and what of data transformations.provenance data is represented using a provenance model which is serialized as JSON output. This model contains information such as sensor readings, device name,, device information, provenance information. This information will be converted into provenance data model and in order to allow for interoperability and visualization.

2.2.1 Open Provenance Model(OPM)

Open provenance model was a specification derived as a result of a meeting at the International Provenance and Annotation Workshop (IPAW) workshop in may 2006. OPM was created to address the need of allowing a unified way of representing provenance data among various applications. It allows for interchangeability between various provenance models that might exist. The goal of OPM is to develop a digital representation of provenance for entities regardless of whether it is produced by a computer system. An example of such is depicted in Figure 7. This OPM graphs represents a process of driving a car. It is represented as a directed acyclic graph which denotes causal dependencies between entities. The edges in the graph denotes dependencies with its source denoting effect and its destination denoting cause.The edges and their relationships are denoted below:

When muliple process has been used by an artifact, roles(denopted by R) should be defined.

- wasGeneratedBy: Shows relationship in which an entity(e,g artifact) is utilized by one or more entities(e.g process). An entity can use multiple enities so it is important to define the role of the
- wasControlledBy: This showsa the relationship in which an entity caused the creation of another entity.
- used(Role): denotes an enity requires the services of another enity in order to execute.

- wasTriggeredBy:
- wasDerrivedFrom:

There are three entities contained in the OPM model: artifact, process, agent.

- artifact: This represents the state of an entity.An artifact is graphically represented by a circle.
- Process: This denotes an event which is taking place.A process is represented by a square object.
- Agent: These are actors that facilitate the execution of a process.An agent is represented by a hexagon.

OPM denotes all previous and current actions that have been performed on an entity and the relationship between each entities contained in the graph. Figure 2 represents an example of an OPM acyclic graph with all of its causal dependencies. The goal of OPM is to be able to model the state of how things both digital or physical are at a given state.

2.2.2 Provenance Data Model(ProvDM)

PROV-DM is an extension of OPM. It was created in an effort to standardize OPM.Prov-DM is a model that is used for depicting entities, activities and , and agents digital or physical. It contains similar yet subtle differences between OPM.Some of the difference between OPM and PROV-DM are outlined below:

- the main components Artifact, process and agent in the OPM model are changed to Entity, Action, and agent.
- additional causal dependencies such as wasAttributedTo and actedOnBelafOf are included to represent direct and indirect causal dependencies respectively between agents and entity.

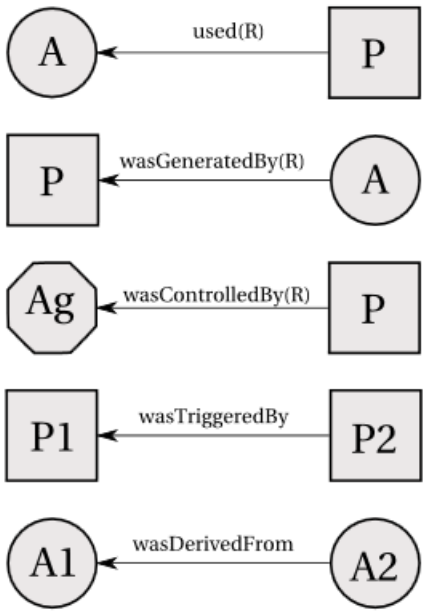

Figure 2.1: Edges and entities in OPM

Since PROV-DM is built on OPM and contains easy to understand constructs of enities, It we choose to use this instead of OPM.

2.3 Overview of Data pruning techniques

One important notion about the class NP is the existence of *complete* problems. Crudely speaking, complete problems for a class are the hardest problems in the class. Thus, NP-complete problems are the hardest problems in the class NP. Also of importance is the notion of NP-hardness. Informally, NP-hard problems are problems that are at least as hard as NP-complete problems. Cook (see [1]) proved that the problem of satisfiability of Boolean formulas (known as SAT) is an NP-complete problem, and Karp (see [3]) showed that many other important problems are also NP-complete by use of *reductions* (generically denoted as $\leq_r$). Completeness is formally defined in terms of reductions. We will now define

two types of reductions, both dealing with the notion of polynomial time. The *time* that an algorithm (e.g., a reduction) requires in order to solve a problem is defined to be the number of steps as a function of the size of the input. The size here, of course, depends on the encoding scheme chosen. Recall that polynomial time means that the number of steps is bounded by some polynomial of the input size. As an example, if we say that an n^3 algorithm $\mathcal{U}$ exists for solving a problem Π of size n, we mean that $\mathcal{U}$ will take no more than n^3 steps in order to solve Π. Therefore, we say that $\mathcal{U}$ is a polynomial-time algorithm. Though not explicitly stated as being algorithms, the following two definitions refer to polynomial-time algorithms.

Definition 1 A *polynomial-time many-one reduction* $\leq_m^p$ (or *polynomial transformation*) from a language $L_1 \subseteq \Sigma_1^*$ to a language $L_2 \subseteq \Sigma_2^*$ (denoted $L_1 \leq_m^p L_2$) is a polynomial-time computable function $f : \Sigma_1^* \to \Sigma_2^*$ such that

$$\forall x \in \Sigma_1^* \ (x \in L_1 \iff f(x) \in L_2).$$

Definition 2 A *polynomial-time Turing reduction* $\leq_T^p$ from a language $L_1 \subseteq \Sigma_1^*$ to a language $L_2 \subseteq \Sigma_2^*$ (denoted $L_1 \leq_T^p L_2$) is a polynomial-time oracle Turing machine that accepts L_1 using L_2 as its oracle.

We now are ready to formally define NP-hardness and NP-completeness. In the following, $\leq_r^p$ stands for either many-one or Turing polynomial-time reductions.

Definition 3 A language L is NP-*hard under* $\leq_r^p$ *reductions* (alternatively, $\leq_r^p$-*hard for* NP) if

$$\forall L' \in \text{NP} \ (L' \leq_r^p L).$$

Definition 4 A language L is NP-*complete under* $\leq_r^p$ *reductions* (alternatively, $\leq_r^p$-*complete for* NP) if $L \in$ NP and L is NP-hard under $\leq_r^p$ reductions.

The following theorem says that unless P=NP, no one will ever be able to find a polynomial-time algorithm to solve any NP-hard problem.

Theorem 1 *If a language L is* NP*-hard under Turing reductions and $L \in$ P, then* P=NP*.*

Corollary *If a language L is* NP*-hard under many-one reductions and $L \in$ P, then* P=NP*.*

Thus, if someone some day finds a polynomial-time algorithm for any NP-hard problem, then this algorithm could be used to solve any NP-complete problem in polynomial time, which would solve many important open problems (and would also be very useful). However, the current wisdom is that this will never be possible. This means that proving that a problem is NP-hard is strong evidence that no polynomial-time algorithm exists for that problem. For this reason, NP-hard problems are said to be *computationally intractable*.

Once we have a collection of problems that are proven to be NP-hard, these problems can be used to prove other problems are NP-hard as well via reductions. This is the method most used to prove NP-hardness, and it is precisely how we prove our main result. This method is formalized by the following theorem:

Theorem 2 *If L is* NP*-hard under $\leq_r^p$ reductions and $L \leq_r^p L'$ then L' is* NP*-hard under $\leq_r^p$ reductions.*

It should be noted that NP-completeness is usually defined using many-one reductions, but philosophically, it may make more sense to define NP-completeness using Turing reductions. This distinction is important because definitions based on different reductions are believed not to be identical; however, since many-one reductions are a special case of Turing reductions, proving that a language is many-one-hard for NP implies that it is Turing-hard for NP.

Chapter 3

Solving Interval Linear Systems Is NP-Hard: Known Result

In this chapter we will define formally what it means to solve a system of interval linear equations and present a negative result related to such systems that was first discovered by Kreinovich, Lakeyev and Noskov in the early 1990's.

3.1 Definitions

Definition 5 An expression of the form

$$\sum_{j=1}^{n}[a_j^-, a_j^+]x_j = [b^-, b^+] \tag{3.1}$$

is called an *interval linear equation* with n unknowns, $x_1, \ldots, x_n$.

Definition 6 Let

$$\sum_{j=1}^{n}[a_{ij}^-, a_{ij}^+]x_j = [b_i^-, b_i^+] \tag{3.2}$$

for $i = 1, \ldots, m$ be a *system of interval linear equations*. The tuple $(x_1, \ldots, x_n)$ is called a *solution* of the system (3.2) if there exist values $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$ such that

$$\sum_{j=1}^{n} a_{ij}x_j = b_i$$

for all $i = 1, \ldots, m$.

Definition 7 By *solving* a system (3.2) of interval linear equations, we mean computing the bounds x_j^- and x_j^+ where

$$x_j^- = \min\{x_j | (x_1, \ldots, x_n) \text{ is a solution to system (3.2)}\}$$

and

$$x_j^+ = \max\{x_j | (x_1, \ldots, x_n) \text{ is a solution to system (3.2)}\}$$

for all $j = 1, \ldots, n$.

To clarify the problem, consider a (hypothetical) feasible algorithm that can be used to solve any system of interval linear equations. Such an algorithm would require the following input to be given:

- the number of equations m,
- the number of unknowns n,
- the endpoints a_{ij}^- and a_{ij}^+ for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$, and
- the endpoints b_i^- and b_i^+ for all $i = 1, \ldots, m$.

Such input would define a particular system of the form (3.2). For the algorithm to be feasible it must be able, given any particular system defined by the input, to compute in polynomial time the following output:

- the endpoints x_j^- and x_j^+ such that

  $$x_j^- = \min\{x_j | (x_1, \ldots, x_n) \text{ is a solution to the given system}\}$$

  and

  $$x_j^+ = \max\{x_j | (x_1, \ldots, x_n) \text{ is a solution to the given system}\}$$

  for all $j = 1, \ldots, n$.

3.2 Theorem

Theorem 3 *The problem of solving systems of interval linear equations is computationally intractable (*NP*-hard).*

This theorem was proven (see [5]) in 1993 by Kreinovich, Lakeyev and Noskov by a reduction from SAT[1].

3.3 What If Intervals Are Narrow?

The proof of theorem 3 uses intervals that are of type $[0, 1]$ that correspond, in measurement terms, to measuring a value of magnitude 0.5 with an absolute accuracy of 0.5, or a relative accuracy of 100%. Such measurements, though possible, are not very accurate. Most measuring devices of practical use in science and industry are far more accurate, and hence, intervals are much narrower. The natural question is:

> *If we restrict ourselves to systems with narrow intervals only, will the problem of solving systems of interval linear equations still be* NP*-hard?*

This question was first analyzed by Lakeyev and Kreinovich in 1995 (see [9]). In the next chapter we will present the result of that analysis.

[1]Recall from chapter **??** that SAT is the problem of satisfiability of Boolean formulas.

Chapter 4

Solving Most Narrow-Interval Linear Systems Is Easy: Known Result

In this chapter we will define what we mean by *narrow intervals* and present a positive result concerning systems of interval linear equations having such narrow intervals. This result was first discovered by Lakeyev and Kreinovich in the mid 1990's.

4.1 Definitions

Definition 8 We say that the number $\tilde{x}$ represents a number x with *absolute accuracy* $\Delta > 0$ if $|x - \tilde{x}| \leq \Delta$.

Definition 9 By *absolute half-width* of the interval $\mathbf{x} = [x^-, x^+]$, we mean the smallest number $\Delta > 0$ for which every number in $\mathbf{x}$ is represented with an absolute accuracy Δ by $\tilde{x} = \dfrac{x^- + x^+}{2}$.

Proposition It can be seen that the absolute half-width of $\mathbf{x}$ is

$$\max_{x \in [x^-, x^+]} |x - \tilde{x}| = \frac{x^+ - x^-}{2}.$$

Definition 10 By *absolute width* W of an interval $\mathbf{x} = [x^-, x^+]$, we mean twice the absolute half-width of $\mathbf{x}$, i.e.,

$$W([x^-, x^+]) = x^+ - x^-.$$

Definition 11 We say that an interval $\mathbf{x}$ is Δ-*narrow in the sense of absolute accuracy* if $W(\mathbf{x}) \leq \Delta$.

Definition 12 Let $\varepsilon > 0$ be a real number, let $D \subseteq R^N$ be a closed and bounded set of positive N-dimension volume $V(D) > 0$, and let $P(x)$ be a property that is true for some points $x \in D$. We say that $P(x)$ is true for (D, ε)-*almost all* x if

$$\frac{V(\{x \in D | \neg P(x)\})}{V(D)} \leq \varepsilon.$$

Definition 13 Let $\eta > 0$ be a real number. We say that intervals

$$[r_1 - d_1, r_1 + d_1], \ldots, [r_n - d_n, r_n + d_n]$$

are η-close to intervals

$$[\tilde{x}_1 - \Delta_1, \tilde{x}_1 + \Delta_1], \ldots, [\tilde{x}_n - \Delta_n, \tilde{x}_n + \Delta_n]$$

if $|r_i - \tilde{x}_i| \leq \eta$ and $|d_i - \Delta_i| \leq \eta$ for all i.

Definition 14 Let $\mathcal{U}$ be an algorithm that solves some systems of interval linear equations, and let $\eta > 0$ be a real number. We say that an algorithm $\mathcal{U}$ is η-*exact* for the interval matrices $\mathbf{a}_{ij}$ and $\mathbf{b}_i$ if for every interval matrix $\mathbf{a}'_{ij}$ and $\mathbf{b}'_i$ that are η-close to $\mathbf{a}_{ij}$ and $\mathbf{b}_i$, the algorithm $\mathcal{U}$ returns the exact solution to the system

$$\sum_{j=1}^{n} \mathbf{a}'_{ij} x_j = \mathbf{b}'_i.$$

Definition 15 We say that an algorithm $\mathcal{U}$ is *almost always exact for narrow input intervals* if for every closed and bounded set $D \subseteq R^N (N = n \cdot m + n)$ there exist $\varepsilon > 0$, $\Delta > 0$ and $\eta > 0$ such that, for (D, ε)-almost all $\tilde{a}_{ij}$ and $\tilde{b}_i$, if all input intervals $\mathbf{a}_{ij}$ and $\mathbf{b}_i$ (containing $\tilde{a}_{ij}$ and $\tilde{b}_i$ respectively) are Δ-narrow (in the sense of absolute accuracy), then the algorithm $\mathcal{U}$ is η-exact for $\mathbf{a}_{ij}$ and $\mathbf{b}_i$.

4.2 Theorem

Theorem 4 *There exists a feasible (polynomial-time) algorithm $\mathcal{U}$ that is almost always exact for narrow input intervals.*

This theorem was proven in 1995 by Lakeyev and Kreinovich (see [9]).

4.3 Open Problem

Theorem 4 says that we can have a feasible algorithm that solves *almost all* narrow-interval linear equation systems, but it does not say whether we can solve *all* of them in reasonable time. Thus, there still remains an open question:

> *Can a feasible algorithm be developed for the general problem of solving systems of linear equations with narrow-interval coefficients?*

The answer to this open question is the main concern of this thesis.

We will show that the problem of solving all narrow-interval linear equation systems is NP-hard; moreover, we will show that the problem is NP-hard not only for intervals that are narrow in the sense of absolute accuracy, but also in the sense of relative accuracy.

Chapter 5

Solving Narrow-Interval Linear Systems Is NP-Hard: New Result

In this chapter we present the main result upon which this thesis is centered—a new theorem and a proof for it based upon the reduction[1] of a system of (arbitrary) interval linear equations to a system of narrow-interval linear equations.

5.1 Definitions

Definition 16 We say that the (non-zero) number $\tilde{x}$ represents a number x with a *relative accuracy* $\delta > 0$ if $\dfrac{|x - \tilde{x}|}{|\tilde{x}|} \leq \delta$.

Definition 17 By *relative half-width* of the interval $\mathbf{x} = [x^-, x^+]$ where $0 \notin [x^-, x^+]$, we mean the smallest number $\delta > 0$ for which every number in $\mathbf{x}$ is represented with a relative accuracy δ by $\tilde{x} = \dfrac{x^- + x^+}{2}$.

Proposition It can be seen that the relative half-width of $\mathbf{x}$ where $0 \notin [x^-, x^+]$ is

$$\max_{x \in [x^-, x^+]} \frac{|x - \tilde{x}|}{|\tilde{x}|} = \frac{x^+ - x^-}{2|\tilde{x}|}.$$

Definition 18 By *relative width* W^{rel} of an interval $\mathbf{x} = [x^-, x^+]$ where $0 \notin [x^-, x^+]$, we mean twice the relative half-width of $\mathbf{x}$, i.e.,

$$W^{rel}([x^-, x^+]) = \frac{x^+ - x^-}{|\tilde{x}|}.$$

[1] The reduction used is called a *polynomial-time one-one reduction*, a special case of polynomial-time many-one reductions.

Definition 19 We say that an interval $\mathbf{x}$ is *δ-narrow in the sense of relative accuracy* if $W^{rel}(\mathbf{x}) \leq \delta$.

Definition 20 We say that an interval $\mathbf{x}$ is *δ-narrow* if it is both δ-narrow in the sense of absolute accuracy and δ-narrow in the sense of relative accuracy.

5.2 Theorem

Theorem 5 *For every $\delta > 0$, the problem of solving systems of interval linear equations with δ-narrow intervals is computationally intractable (*NP*-hard).*

Corollary 1 *For every $\Delta > 0$, the problem of solving systems of interval linear equations with intervals Δ-narrow in the sense of absolute accuracy is computationally intractable (*NP*-hard).*

Corollary 2 *For every $\delta > 0$, the problem of solving systems of interval linear equations with intervals δ-narrow in the sense of relative accuracy is computationally intractable (*NP*-hard).*

5.3 Proof

5.3.1 Part I: Reduction of Interval Linear System to Narrow-Interval Linear System

We have seen that the general problem of solving interval linear equation systems (3.2) is NP-hard. We now intend to show that this general problem can be reduced to the problem of solving a system of interval linear equations with δ-narrow intervals.

Let us start with an arbitrary interval linear equation system (3.2). To reduce it to a δ-narrow interval linear equation system, we introduce new variables w_i, y_{ij} and z_{ij} for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$. For the enlarged list of $m + n + 2mn$ variables (i.e., x_j, w_i, y_{ij} and z_{ij}) we introduce the following new system of $m + n + 2mn$ δ-narrow interval linear equations[2]:

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] z_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] w_i = c_i, \tag{5.1}$$

$$w_i = \gamma_i, \tag{5.2}$$

$$y_{ij} - \mu_{ij} x_j = 0, \tag{5.3}$$

$$z_{ij} - \nu_{ij} x_j = 0, \tag{5.4}$$

where $\delta > 0$ and the numerical (non-interval) coefficients c_i, $\gamma_i \geq 0$, μ_{ij} and $\nu_{ij} \geq 0$ will be chosen in such a way that for every solution of this δ-narrow interval linear equation system (5.1)–(5.4) we have

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] w_i = [b_i^-, b_i^+] \tag{5.5}$$

and

$$y_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] z_{ij} = [a_{ij}^-, a_{ij}^+] x_j \tag{5.6}$$

for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

[2]For clarity, non-interval coefficients can be thought of as intervals of zero width; e.g., the coefficient for each variable y_{ij} is $[1, 1]$ and each coefficient c_i is equivalent to the interval $[c_i, c_i]$.

Let us first find the values for c_i and γ_i. From equation (5.2) we know that $w_i = \gamma_i$. By substituting this expression into equation (5.5) we get

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]\gamma_i = [b_i^-, b_i^+].$$

Intervals are equal *if and only if* their endpoints coincide. Since we are looking for a solution with $\gamma_i \geq 0$, the equations for the endpoints take the following forms:

$$c_i - (1 - \frac{\delta}{2})\gamma_i = b_i^+ \tag{5.7}$$

and

$$c_i - (1 + \frac{\delta}{2})\gamma_i = b_i^-. \tag{5.8}$$

Subtracting the second equation from the first we get

$$\delta \cdot \gamma_i = b_i^+ - b_i^-,$$

and hence,

$$\boxed{\gamma_i = \frac{1}{\delta}(b_i^+ - b_i^-).} \tag{5.9}$$

Substituting this value into equation (5.7), we get

$$c_i - (1 - \frac{\delta}{2})\frac{1}{\delta}(b_i^+ - b_i^-) = b_i^+$$

from which we get

$$c_i - (\frac{1}{\delta} - \frac{1}{2})(b_i^+ - b_i^-) = b_i^+,$$

and hence,

$$\boxed{c_i = \frac{1}{2}(b_i^+ + b_i^-) + \frac{1}{\delta}(b_i^+ - b_i^-).} \tag{5.10}$$

Now let us find the values for μ_{ij} and ν_{ij}. From equations (5.3) and (5.4) we conclude that $y_{ij} = \mu_{ij}x_j$ and $z_{ij} = \nu_{ij}x_j$. Substituting these expressions into equation (5.6) we get

$$\mu_{ij}x_j + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]\nu_{ij}x_j = [a_{ij}^-, a_{ij}^+]x_j.$$

For this equality to be true for all x_j, coefficients for x_j on both sides of the equation must coincide. Thus, we conclude that

$$\mu_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]\nu_{ij} = [a_{ij}^-, a_{ij}^+].$$

As stated before, for intervals to be equal their endpoints must coincide, and since we are looking for a solution for $\nu_{ij} \geq 0$, the equations for the endpoints take the following forms:

$$\mu_{ij} + (1 + \frac{\delta}{2})\nu_{ij} = a_{ij}^+ \tag{5.11}$$

and

$$\mu_{ij} + (1 - \frac{\delta}{2})\nu_{ij} = a_{ij}^-. \tag{5.12}$$

Subtracting the second equation from the first we get

$$\delta \cdot \nu_{ij} = a_{ij}^+ - a_{ij}^-,$$

and hence,

$$\boxed{\nu_{ij} = \frac{1}{\delta}(a_{ij}^+ - a_{ij}^-).} \tag{5.13}$$

Substituting this value back into equation (5.11) we get

$$\mu_{ij} + (1 + \frac{\delta}{2})\frac{1}{\delta}(a_{ij}^+ - a_{ij}^-) = a_{ij}^+$$

which leads to

$$\mu_{ij} + (\frac{1}{\delta} + \frac{1}{2})(a_{ij}^+ - a_{ij}^-) = a_{ij}^+,$$

and hence,

$$\boxed{\mu_{ij} = \frac{1}{2}(a_{ij}^+ + a_{ij}^-) - \frac{1}{\delta}(a_{ij}^+ - a_{ij}^-).} \tag{5.14}$$

5.3.2 Part II: The Two Systems Are "Equivalent"

Let us show that every system (3.2) of interval linear equations is "equivalent" to the δ-narrow interval linear equation system (5.1)–(5.4) in the following sense:

Claim 1 If $(x_1, \ldots, x_n)$ is a solution of the original system (3.2), then for the same values $x_1, \ldots, x_n$, and for some values $w_1, \ldots, w_m$, $y_{11}, \ldots, y_{mn}$ and $z_{11}, \ldots, z_{mn}$, the extended tuple

$$(x_1, \ldots, x_n, w_1, \ldots, w_m, y_{11}, \ldots, y_{mn}, z_{11}, \ldots, z_{mn}) \tag{5.15}$$

is a solution of the δ-narrow interval system (5.1)–(5.4).

Claim 2 Conversely, if tuple (5.15) is a solution of the δ-narrow interval linear equation system (5.1)–(5.4), then $(x_1, \ldots, x_n)$ is a solution of the original system (3.2) for the same values $x_1, \ldots, x_n$.

We will now prove that the two systems are indeed "equivalent" in the above sense.

Proof of Claim 1

Let us assume that $(x_1, \ldots, x_n)$ is a solution of the original system (3.2). By definition, if $(x_1, \ldots, x_n)$ is a solution to the original system (3.2), then there must exist values $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$ such that

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \tag{5.16}$$

for all $i = 1, \ldots, m$. With this in mind, let us introduce new parameters $\alpha_{ij} = a_{ij} - a_{ij}^-$ and $\beta_i = b_i - b_i^-$; thus, $a_{ij} = a_{ij}^- + \alpha_{ij}$ and $b_i = b_i^- + \beta_i$. Substituting these expressions into (5.16) we get

$$\sum_{j=1}^{n} (a_{ij}^- + \alpha_{ij}) x_j = b_i^- + \beta_i. \tag{5.17}$$

Since $a_{ij} \in [a_{ij}^-, a_{ij}^+]$, we conclude that $\alpha_{ij} \in [0, a_{ij}^+ - a_{ij}^-]$. Similarly, since $b_i \in [b_i^-, b_i^+]$, we conclude that $\beta_i \in [0, b_i^+ - b_i^-]$.

Now let us show that there exists an extended tuple (5.15) that is a solution to system (5.1)–(5.4) for the same values $x_1, \ldots, x_n$ and for appropriately chosen values of $w_1, \ldots, w_m$, $y_{11}, \ldots, y_{mn}$ and $z_{11}, \ldots, z_{mn}$. In order to have equations (5.2)–(5.4) automatically satisfied, we will take $w_i = \gamma_i$, $y_{ij} = \mu_{ij} x_j$ and $z_{ij} = \nu_{ij} x_j$. It is thus sufficient to show that equation (5.1) is satisfied for all $i = 1, \ldots, m$; in other words, we need to show that there exist values $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ and $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ such that

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \lambda_{ij} z_{ij} + \kappa_i w_i = c_i \tag{5.18}$$

for all $i = 1, \ldots, m$.

Note that for any given i, if $\gamma_i = 0$ (and thus $w_i = 0$), then $\kappa_i w_i = 0$, implying that κ_i can be *any value* as long as $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ (e.g., $\kappa_i = 1$). Likewise, for any given i and j, if $\nu_{ij} = 0$ (and thus $z_{ij} = 0$), then $\lambda_{ij} z_{ij} = 0$, implying that λ_{ij} can be *any value* as long as $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ (e.g., $\lambda_{ij} = 1$). Therefore, for the remainder of the proof we will be choosing κ_i only for those i for which $\gamma_i > 0$ (i.e., $b_i^- < b_i^+$), and we will be choosing λ_{ij} only for those i and j for which $\nu_{ij} > 0$ (i.e., $a_{ij}^- < a_{ij}^+$).

We have chosen w_i and c_i for which the equation (5.5) is true, i.e., for which

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] w_i = [b_i^-, b_i^+].$$

Crudely speaking, this equality means that when we take different values for κ_i such that $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, the set

$$c_i - [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}] w_i$$

of possible values of $c_i - \kappa_i w_i$ coincides with the interval $[b_i^-, b_i^+]$. In particular, since $b_i \in [b_i^-, b_i^+]$, there must exist a value $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ for which

$$c_i - \kappa_i w_i = b_i = b_i^- + \beta_i.$$

From this equation we can find the exact value of κ_i.

We know from equation (5.2) that $w_i = \gamma_i$, so by substitution we get

$$c_i - \kappa_i \gamma_i = b_i^- + \beta_i,$$

and hence,

$$\kappa_i = \frac{1}{\gamma_i}(c_i - b_i^- - \beta_i).$$

Substituting the values of γ_i and c_i from equations (5.9) and (5.10) we get

$$\kappa_i = \frac{\delta}{b_i^+ - b_i^-}(\frac{1}{2}(b_i^+ + b_i^-) + \frac{1}{\delta}(b_i^+ - b_i^-) - b_i^- - \beta_i).$$

Simplifying, we get

$$\kappa_i = 1 + \frac{\delta}{b_i^+ - b_i^-}(\frac{1}{2}(b_i^+ + b_i^-) - b_i^- - \beta_i)$$

from which we get

$$\kappa_i = 1 + \frac{\delta}{b_i^+ - b_i^-}(\frac{1}{2}(b_i^+ - b_i^-) - \beta_i),$$

and hence,

$$\boxed{\kappa_i = 1 + \frac{\delta}{2} - \frac{\delta \cdot \beta_i}{b_i^+ - b_i^-}.} \tag{5.19}$$

We have also chosen y_{ij} and z_{ij} for which the equation (5.6) is true, i.e., for which

$$y_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]z_{ij} = [a_{ij}^-, a_{ij}^+]x_j$$

Crudely speaking, this equality means that when we take different values for λ_{ij} such that $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, the set

$$y_{ij} + [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]z_{ij}$$

of possible values of $y_{ij} + \lambda_{ij}z_{ij}$ coincides with the interval $[a_{ij}^-, a_{ij}^+]x_j$. In particular, since $a_{ij} \in [a_{ij}^-, a_{ij}^+]$, there must exist a value $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ for which

$$y_{ij} + \lambda_{ij}z_{ij} = (a_{ij}^- + \alpha_{ij})x_j.$$

From this equation we can find the exact value of λ_{ij}.

We conclude from equations (5.3) and (5.4) that $y_{ij} = \mu_{ij}x_j$ and $z_{ij} = \nu_{ij}x_j$, so by substitution we get

$$\mu_{ij}x_j + \lambda_{ij}\nu_{ij}x_j = (a_{ij}^- + \alpha_{ij})x_j,$$

and hence,

$$\lambda_{ij} = \frac{1}{\nu_{ij}}(a_{ij}^- + \alpha_{ij} - \mu_{ij}).$$

Substituting the values of the coefficients ν_{ij} and μ_{ij} from equations (5.13) and (5.14) we get

$$\lambda_{ij} = \frac{\delta}{a_{ij}^+ - a_{ij}^-}(a_{ij}^- + \alpha_{ij} - \frac{1}{2}(a_{ij}^+ + a_{ij}^-) + \frac{1}{\delta}(a_{ij}^+ - a_{ij}^-)).$$

Simplifying, we get

$$\lambda_{ij} = 1 + \frac{\delta}{a_{ij}^+ - a_{ij}^-}(a_{ij}^- + \alpha_{ij} - \frac{1}{2}(a_{ij}^+ + a_{ij}^-))$$

from which we get

$$\lambda_{ij} = 1 - \frac{\delta}{a_{ij}^+ - a_{ij}^-}(\frac{1}{2}(a_{ij}^+ - a_{ij}^-) - \alpha_{ij}),$$

and hence,

$$\boxed{\lambda_{ij} = 1 - \frac{\delta}{2} + \frac{\delta \cdot \alpha_{ij}}{a_{ij}^+ - a_{ij}^-}.} \tag{5.20}$$

To show that these values for κ_i and λ_{ij} are indeed the ones desired, we first note that, since $\beta_i \in [0, b_i^+ - b_i^-]$ and $\alpha_{ij} \in [0, a_{ij}^+ - a_{ij}^-]$, we can conclude that

$$\frac{\beta_i}{b_i^+ - b_i^-} \in [0, 1]$$

and

$$\frac{\alpha_{ij}}{a_{ij}^+ - a_{ij}^-} \in [0, 1].$$

Thus,

$$\frac{\delta \cdot \beta_i}{b_i^+ - b_i^-} \in [0, \delta]$$

and

$$\frac{\delta \cdot \alpha_{ij}}{a_{ij}^+ - a_{ij}^-} \in [0, \delta]$$

from which it follows that

$$1 + \frac{\delta}{2} - \frac{\delta \cdot \beta_i}{b_i^+ - b_i^-} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$$

and

$$1 - \frac{\delta}{2} + \frac{\delta \cdot \alpha_{ij}}{a_{ij}^{+} - a_{ij}^{-}} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}],$$

and hence,

$$\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$$

and

$$\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}].$$

Our goal is to prove that equation (5.1) is satisfied. By showing that we have found appropriate values for κ_i and λ_{ij} such that equation (5.18) is satisfied, we can do just that. We have already chosen the values for y_{ij} and z_{ij}. Substituting these values into equation (5.18) we get an equivalent equation

$$\sum_{j=1}^{n} \mu_{ij} x_j + \sum_{j=1}^{n} \lambda_{ij} \nu_{ij} x_j + \kappa_i w_i = c_i.$$

This equation, in its turn, is equivalent to the equation

$$\sum_{j=1}^{n} (\mu_{ij} + \lambda_{ij} \nu_{ij}) x_j = c_i - \kappa_i w_i. \tag{5.21}$$

Let us now show that this equation is satisfied and thus prove that the equivalent equation (5.18) is satisfied as well.

- According to our choice of κ_i, the right-hand side of equation (5.21) is equal to b_i.
- According to our choice of λ_{ij}, we have

  $$\mu_{ij} + \lambda_{ij} \nu_{ij} = a_{ij},$$

  and hence, the left-hand side of equation (5.21) is equal to

  $$\sum_{j=1}^{n} a_{ij} x_j.$$

We started with $x_1, \ldots, x_n$ for which $\sum_{j=1}^{n} a_{ij} x_j = b_i$; hence, the left-hand side and the right-hand side of equation (5.21) coincide. Thus, we have proven that equation (5.18) is satisfied.

Since we are able to show that there do indeed exist values $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ and $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ such that equation (5.18) is satisfied for all $i = 1, \ldots, m$, it follows that if $(x_1, \ldots, x_n)$ is a solution of system (3.2), then for the same values $x_1, \ldots, x_n$ and for some values $w_1, \ldots, w_m$, $y_{11}, \ldots, y_{mn}$ and $z_{11}, \ldots, z_{mn}$, there exists an extended tuple (5.15) that is a solution of system (5.1)–(5.4). This proves Claim 1.

Proof of Claim 2

Now we must show that the converse is true, namely, that if tuple (5.15) is a solution of the δ-narrow interval linear equation system (5.1)–(5.4), then $(x_1, \ldots, x_n)$ is a solution of the interval linear equation system (3.2) for the same values $x_1, \ldots, x_n$.

Let us assume tuple (5.15) is a solution of system (5.1)–(5.4). For this to be true, equations (5.2)–(5.4) must be satisfied, and thus we can conclude that $w_i = \gamma_i$, $y_{ij} = \mu_{ij} x_j$ and $z_{ij} = \nu_{ij} x_j$. The fact that equation (5.1) is satisfied by a given tuple means that

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \lambda_{ij} z_{ij} + \kappa_i w_i = c_i \tag{5.22}$$

for some $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$ and $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$. Let us show that

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \tag{5.23}$$

for some values $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$.

From equation (5.22) we conclude that

$$\sum_{j=1}^{n} y_{ij} + \sum_{j=1}^{n} \lambda_{ij} z_{ij} = c_i - \kappa_i w_i.$$

Since $w_i = \gamma_i$, $y_{ij} = \mu_{ij} x_j$ and $z_{ij} = \nu_{ij} x_j$, by substitution we conclude that

$$\sum_{j=1}^{n} (\mu_{ij} + \lambda_{ij} \nu_{ij}) x_j = c_i - \kappa_i \gamma_i.$$

Note that this equation has the desired form (5.23) where

$$a_{ij} = \mu_{ij} + \lambda_{ij} \nu_{ij} \tag{5.24}$$

and

$$b_i = c_i - \kappa_i \gamma_i. \tag{5.25}$$

Thus, to complete the proof, we need only show that $a_{ij} \in [a_{ij}^-, a_{ij}^+]$ and $b_i \in [b_i^-, b_i^+]$.

Let us first show that $b_i \in [b_i^-, b_i^+]$. Since $\kappa_i \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, we have that

$$1 + \frac{\delta}{2} \geq \kappa_i \geq 1 - \frac{\delta}{2}.$$

By multiplying all sides of the inequality by $\gamma_i \geq 0$, we conclude that

$$(1 + \frac{\delta}{2})\gamma_i \geq \kappa_i \gamma_i \geq (1 - \frac{\delta}{2})\gamma_i.$$

By subtracting all parts of this inequality from c_i, we conclude that

$$c_i - (1 + \frac{\delta}{2})\gamma_i \leq c_i - \kappa_i \gamma_i \leq c_i - (1 - \frac{\delta}{2})\gamma_i.$$

According to our choice of c_i and γ_i (which led to equations (5.7) and (5.8)), this is equivalent to

$$b_i^- \leq c_i - \kappa_i \gamma_i \leq b_i^+.$$

Thus, the value $b_i = c_i - \kappa_i \gamma_i$ we have chosen indeed belongs to the interval $[b_i^-, b_i^+]$.

Let us now show, in like fashion, that $a_{ij} \in [a_{ij}^-, a_{ij}^+]$. Since $\lambda_{ij} \in [1 - \frac{\delta}{2}, 1 + \frac{\delta}{2}]$, we have that

$$1 - \frac{\delta}{2} \leq \lambda_{ij} \leq 1 + \frac{\delta}{2}.$$

By multiplying all sides of the inequality by $\nu_{ij} \geq 0$, we conclude that

$$(1 - \frac{\delta}{2})\nu_{ij} \leq \lambda_{ij}\nu_{ij} \leq (1 + \frac{\delta}{2})\nu_{ij}.$$

By adding all parts of this inequality to μ_{ij}, we conclude that

$$\mu_{ij} + (1 - \frac{\delta}{2})\nu_{ij} \leq \mu_{ij} + \lambda_{ij}\nu_{ij} \leq \mu_{ij} + (1 + \frac{\delta}{2})\nu_{ij}.$$

According to our choice of μ_{ij} and ν_{ij} (which led to equations (5.11) and (5.12)), this is equivalent to

$$a_{ij}^- \leq \mu_{ij} - \lambda_{ij}\nu_{ij} \leq a_{ij}^+,$$

i.e., the value $a_{ij} = \mu_{ij} - \lambda_{ij}\nu_{ij}$ that we have chosen indeed belongs to the interval $[a_{ij}^-, a_{ij}^+]$.

Thus, we have shown that if tuple (5.15) is a solution of system (5.1)–(5.4), then $(x_1, \ldots, x_n)$ is a solution of system (3.2) for the same values $x_1, \ldots, x_n$. This proves Claim 2.

5.3.3 Conclusion

We saw in part II that the two systems, namely, the interval linear equation system (3.2) and the δ-narrow interval linear equation system (5.1)–(5.4), are "equivalent" in the sense described therein. In part I we saw that the number of equations and unknowns of system (5.1)–(5.4) is bounded by a polynomial of the number of equations and unknowns of system (3.2). Further, the number of steps required for reducing an interval linear equation system (3.2) to a δ-narrow interval linear equation system (5.1)–(5.4) is bounded by a polynomial of the number of equations and unknowns of system (3.2). Thus, it follows from parts I and II that if we have an algorithm $\mathcal{U}$ that can solve δ-narrow interval linear equation systems in polynomial time, then we can solve an arbitrary system (3.2) of interval linear equations in polynomial time by applying this hypothetical algorithm $\mathcal{U}$ to the "equivalent" δ-narrow interval linear equation system (5.1)–(5.4). But, as stated in part I, solving interval linear equation systems is an NP-hard problem.

So, if we can (in general) solve interval linear equation systems in polynomial time, we can solve any problem from the class NP in polynomial time. Therefore, if we can (in general) solve δ-narrow interval linear equation systems in polynomial time, we can solve any problem from the class NP in polynomial time. Thus, we conclude that *the problem of solving δ-narrow interval linear equation systems is* NP*-hard*. Q.E.D.

Chapter 6

Concluding Remarks

6.1 Significance of the Result

Now that we have shown that we cannot solve narrow-interval linear equation systems in general, what does this mean to the computing and mathematical communities? Unless P=NP, attempts at developing a feasible algorithm to solve this problem in general will assuredly fail; however, the very fact that there exists an algorithm for solving a large number of these systems (see [9]) shows that work on solving a subclass of the class of all narrow-interval linear equation systems is certainly a worthwhile endeavor.

6.2 Future Work

The problem now shifts to identifying new subclasses of the class of all narrow-interval linear equation systems for which the problem of solving them is possible with the development of new algorithms. Also, if the general problem (or any problem in the class NP) shows up often enough in industry, science, research, etc., work on improving existing and/or creating new approximation methods (including heuristic and/or statistical methods, where applicable) is certainly warranted. Since we cannot compute the exact bounds for the general case, good approximation methodologies are the most we can hope for or expect.

Chapter 7

Extra Chapter

This is an extra chapter added into Patrick's thesis to illustrate the use of figures and tables and the inclusion of a pdf file.

First the tables are shown as Table 7.1 and Table 7.2. Then the figure is shown in Figure 7.1.

Table 7.1: A long caption. In this table example, we have a very long caption that does not fit on one line. The text of the caption should line up on the left. The text inside the square brackets will be included in the List of Tables.

2	3	300
3	4	400
4	5	500
5	6	600

Table 7.2: Example of a table

2	3	300
3	4	400
4	5	500
5	6	600

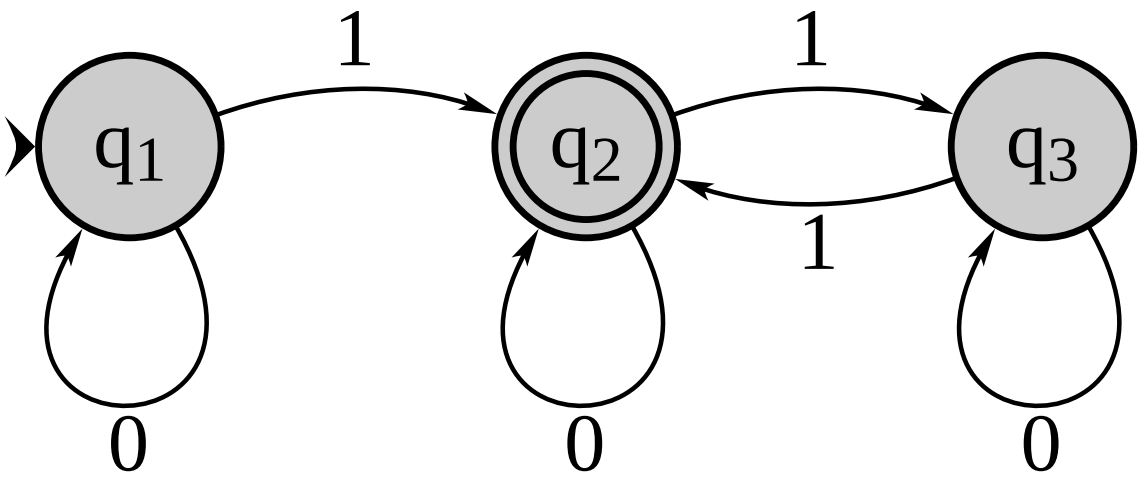

Figure 7.1: A simple finite automaton

References

[1] S. Cook, "The complexity of theorem-proving procedures," *Proceedings of the 3rd ACM Symposium on Theory of Computing*, Shaker Heights, Ohio, 1971, pp. 151–158.

[2] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W. H. Freeman, San Francisco, 1979.

[3] R. Karp, "Reducibility among combinatorial problems," in: R. Miller and J. Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York, 1972, pp. 85–103.

[4] R. B. Kearfott and V. Kreinovich (eds.), *Applications of Interval Computations*, Kluwer Academic Publishers, Norwell, MA, 1996.

[5] V. Kreinovich, A. V. Lakeyev and S. I. Noskov, "Optimal solution of interval linear systems is intractable (NP-hard)," *Interval Computations*, 1993, No. 1, pp. 6–14.

[6] V. Kreinovich, A. V. Lakeyev and J. Rohn , "Computational complexity of interval algebraic problems: some are feasible and some are computationally intractable: a survey," in: G. Alefeld and A. Frommer (eds.), *Scientific Computing and Validated Numerics*, Akademie-Verlag, Berlin, 1996, pp. 293–306.

[7] V. Kreinovich, A. V. Lakeyev, J. Rohn and P. Kahl, *Feasible? Intractable? On Computational Complexity of Data Processing and Interval Computations*, Kluwer Academic Publishers, Norwell, MA, 1996 (to appear).

[8] U. Kulisch and W. L. Miranker, *Computer Arithmetic in Theory and Practice*, Academic Press, NY, 1981.

[9] A. V. Lakeyev and V. Kreinovich, "If input intervals are small enough, then interval computations are almost always easy," *Reliable Computing*, Supplement (Extended

Abstracts of APIC'95: International Workshop on Applications of Interval Computations), 1995, pp. 134–139.

[10] L. Levin, "Universal sequential search problems," *Problems of Information Transmission*, 1973, Vol. 9, No. 3, pp. 265–266.

[11] R. E. Moore, "Automatic error analysis in digital computation," *Technical Report LMSD-48421*, Lockheed Missiles and Space Co., Palo Alto, CA, January 1959.

[12] R. E. Moore, *Interval Analysis*, Prentice Hall, Englewood Cliffs, NJ, 1966.

[13] A. Neumaier, *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, 1990.

[14] S. G. Rabinovich, *Measurement Errors: Theory and Practice*, American Institute of Physics, NY, 1995.

Curriculum Vitae

Patrick Thor Kahl was born on July 12, 1961. The first son of Ulf Thor Gustav Kahl and Carolyn Kahl, he graduated from Coronado High School, El Paso, Texas, in the spring of 1979. He entered Auburn University in the fall of 1979, and, in the spring of 1982, The University of Texas at El Paso. In 1985 he joined the United States Navy where he served for eight years, most of it aboard the submarine USS Narwhal (SSN671). In the fall of 1993, after being honorably discharged from the navy, Patrick resumed his studies at The University of Texas at El Paso. While pursuing his bachelor's degree in Computer Science he worked as a Teaching Assistant, and as a programmer at the National Solar Observatory at Sunspot, New Mexico. He received his bachelor's degree in Computer Science in the summer of 1994.

In the fall of 1994, he entered the Graduate School of The University of Texas at El Paso. While pursuing a master's degree in Computer Science he worked as a Teaching and Research Assistant, and as the Laboratory Instructor for the 1995 Real-Time Programming Seminar at the University of Puerto Rico, Mayagüez Campus. He was a member of the Knowledge Representation Group and the Rio Grande Chapter of the Association for Computing Machinery.

Permanent address: 6216 Sylvania Way
El Paso, Texas 79912-4927