Informal Workshop on Formal Methods at Koç University

Monday, May 31, 2010	
9:00 am to 10:00 am	Eliminating Web Software Vulnerabilities Using Automated Verification **Tevfik Bültan, UCSB**
10:00 am to 10:30 am	Break
10:30 am to 12:30 pm	Overview: Runtime and Static Verification of Concurrent Software Systems **Serdar Taşıran, Koç University** Static Verification of Concurrent Programs using Reduction and Abstraction **Tayfun Elmas, Koç University** Static Verification with Reduction - The Case for Optimistic Concurrency **Ali Sezgin, Koç University** Defect-aware Logic Mapping Based on Bipartite Subgraph Isomorphism and Canonization **Sezer Gören Uğurdağ, Bahçeşehir University**
12:30 pm to 2:00 pm	**LUNCH**
2:00 pm to 4:00 pm	FSM-Based Testing **Hüsnü Yenigün, Sabancı University** Combining Hardware and Software Instrumentation as Abstraction Mechanisms for Program Executions **Cemal Yılmaz, Sabancı University** Applications of Automated Reasoning **Esra Erdem, Sabancı University** A Formal Look at Program Generation **Barış Aktemur, Özyeğin University**
4:00 pm to 4:30 pm	Break
4:30 pm to 6:30 pm	A Framework for Formal Specification and Verification of Security Policies **Devrim Ünal, Boğaziçi University** Formal Modeling and Analysis of Trust Based on Securities of Services **Şerif Bahtiyar, Boğaziçi University** Formal Security Analysis of Mobile Ad-Hoc Network Routing Protocols using Model Checking **A. Burak Gürdağ, Boğaziçi University** Security aspects of green computing **Murat Cihan, Boğaziçi University**

Eliminating Web Software Vulnerabilities Using Automated Verification

Tevfik Bültan, UCSB

ABSTRACT: Most important Web application vulnerabilities, such as SQL Injection, Cross Site Scripting and Malicious File Execution, are due to inadequate manipulation of string variables. We present sound automata-based symbolic string analysis techniques for automatic verification of string manipulating programs. String analysis is a static analysis technique that determines the values that a string expression can take during program execution at a given program point. This information can be used to detect security vulnerabilities and program errors, and to verify that program inputs are sanitized properly. We use our automata-based string analysis techniques to detect and prevent string related vulnerabilities in Web applications.

Our approach consists of three phases: Given an attack pattern, we first conduct a vulnerability analysis to identify if strings that match the attack pattern can reach security-sensitive functions. Next, we compute the vulnerability signature which characterizes all input strings that can exploit the discovered vulnerability. Given the vulnerability signature, we then construct sanitization statements that 1) check if a given input matches the vulnerability signature and 2) modify the input in a minimal way so that the modified input does not match the vulnerability signature.

We have developed a tool called Stranger that implements our automata-based symbolic string analysis approach. Stranger can be used to find and eliminate string-related security vulnerabilities in PHP applications.

Overview: Runtime and Static Verification of Concurrent Software Systems

Serdar Taşıran, Koç University

ABSTRACT: Single processor performance has peaked. Partly due to this, almost all general-purpose and embedded computing is carried out on multi-core or multi-processor systems. As a result, all software has to be concurrent or concurrency-aware. Verification of multithreaded concurrent programs is a difficult problem and an area of recently intensified practical interest and research activity. In this talk, I will first discuss the challenges unique to specifying and validating concurrent software. I will then present highlights from the research we have been carrying out on concurrent program specification, validation and verification at Koc University for the past seven years.

Static Verification of Concurrent Programs using Reduction and Abstraction

Tayfun Elmas, Koç University

ABSTRACT: Concurrency-related bugs are notoriously difficult to discover by code review and testing. By doing a formal proof on the program text, one can statically verify that no execution of the program leads to an error. In this talk, we present a new proof method that simplifies verifying assertions in concurrent programs. The key feature of our method is the use of atomicity as the main proof tool. A proof is done by rewriting the program with larger atomic blocks in a number of steps. In order to reach the desired level of atomicity, we alternate proof steps that apply reduction and abstraction, each of which improves the outcome of the other in a following step. Then, we check assertions sequentially within the atomic blocks of the resulting program and declare the original program correct when we discharge all the assertions. Our publicly available software tool, QED, mechanizes proofs using the Z3 SMT solver. We demonstrated the simplicity of our proof strategy by verifying well-known programs using fine-grained locking and non-blocking data algorithms.

Static Verification with Reduction - The Case for Optimistic Concurrency

Ali Sezgin, Koç University

ABSTRACT: Optimistic concurrency is a programming paradigm aimed at improving the performance of concurrent programs by minimizing the interval of exclusive ownership of shared resources under the assumption that few threads simultaneously compete for the same resources. Reduction is a well-known complexity relief technique applied to concurrent program verification. Unfortunately, reduction cannot be directly applied to optimistic concurrency. In this talk, I will briefly describe these terms, why they seem to be a poor match and propose a technique, developed in our group, as a possible conciliation between them.

Defect-aware Logic Mapping Based on Bipartite Subgraph Isomorphism and Canonization

Sezer Gören Uğurdağ, Bahçeşehir University

ABSTRACT: Today CMOS technology is facing serious challenges at the device level as Moore's law reaches its end. While research on the issues of CMOS technology such as manufacturing variability, sub-threshold leakage, power dissipation, performance, and cost improvement is continuing, research on new devices based on quantum physics phenomena is progressing and maturing. Nanoelectronic devices have been proposed as an alternative to CMOS or for integration within CMOS. Nanoelectronic devices such as carbon nanotubes and silicon nanowires can be chemically self-assembled on a molecule-by-molecule basis due to their very regular structures. Since the regular and chemically self-assembled nature of these devices can support implementation of regular arrays similar to FPGAs, reprogrammable nanoarchitectures such as crossbars are currently being investigated. Nanocrossbars consist of several perpendicular nanowires and two molecular diodes at the crosspoints. The horizontal nanowires are the inputs whereas the vertical nanowires are the outputs. The molecular diodes at the crosspoints act like programmable switches and contain only a few tens of atoms. Obviously, these devices exhibit higher defect rates, since with such small contact areas they can get damaged very easily and random breaks can occur during manufacturing.

In this work, we address the NP-complete problem of mapping a logic function on to a nanocrossbar with a known defect map. We first show that this problem can be transformed into a Bipartite SubGraph Isomorphism (BSGI) problem. Then we present our proposed KNS-2DS algorithm, which canonizes both graphs in N2 time (N being the number of nodes) and then matches them in N3 time in the worst case. KNS-2DS uses a K-Neighbor Sort to initialize our main contribution "2D-Sort." 2D-Sort is an iterative rough canonizer that lets a straight-forward matching algorithm complete the job. Our algorithm offers very short run-times (due to canonization) compared to previous work and has success on all benchmarks. KNS-2DS is also novel from the perspective of the BSGI problem in the sense that it is based on canonization but not on a search tree with backtracking.

FSM-Based Testing

Hüsnü Yenigün, Sabancı University

ABSTRACT: In the context of finite state machine based testing, where the specifications are given in the form of a Mealy machine, Checking Sequences are special test sequences with full fault coverage guarantee under certain assumptions on specifications and implementations. The problem of generating checking sequences and finding short ones has an history of about half a century. In this talk, we will go over some of the relatively new methods for generating checking sequences.

Combining Hardware and Software Instrumentation as Abstraction Mechanisms for Program Executions

Cemal Yılmaz, Sabancı University

ABSTRACT: In recent years, numerous researchers have proposed data-driven techniques to improve the quality of deployed software. At a high level these techniques typically follow the general approach of lightly instrumenting the software system, monitoring its execution, analyzing the resulting data, and then acting on the analysis results. Some example applications of this general approach include identifying likely fault locations, anticipating resource exhaustion, and categorizing crash reports as instances of previously reported bugs. A fundamental assumption these and similar approaches have is that there are identifiable and repeatable patterns in the behavior of program executions and that similarities and deviations from these patterns can help shape future software development efforts. One less well-understood issue, however, is how best to limit the total costs of implementing these approaches and whether and how tradeoffs can be made between cost and analysis accuracy. This issue is important because these approaches have been targeted at deployed software systems, excessive runtime overhead is generally undesirable. Therefore, it is important to limit instrumentation overhead as much as possible while still supporting the highest levels of analysis accuracy. In general, previous efforts have tended to either ignore this problem or have appealed to various sampling strategies for a solution. One potential drawback of sampling however is that aggressive sampling schemes greatly increase the numbers of observations that must be made in order to have confidence in the data.

In this talk, I will present an improved approach, which aim to push substantial parts of the data collection work onto the CPU by leveraging fast hardware performance counters. The data is augmented with further data collected by a minimal amount of software instrumentation that is added to the system's software. I will also discuss three example applications of the proposed approach: Failure detection, failure classification, and fault localization.

Applications of Automated Reasoning

Esra Erdem, Sabancı University

ABSTRACT: I will give an overview of several applications of automated reasoning studied by Knowledge Representation and Reasoning Group at Sabanci University.

Our group focuses on research in the areas of knowledge representation and automated reasoning along two lines: on the mathematical foundations (e.g., monotonic/nonmonotonic formalisms, formulations of various tasks like diagnosis and planning in these formalisms, algorithms for automating reasoning over these formulations), and their applications to computer sciences and other sciences (e.g., robotics, VLSI Design, historical linguistics, medical informatics, and computational biology).

A Formal Look at Program Generation
Barış Aktemur, Özyeğin University

ABSTRACT: Program Generation (PG) is the technique of composing various pieces of code to produce a program. PG can improve performance of a program by specializing the program according to parameters that become available only at runtime. A major research problem in program generation is to formally guarantee that a generator generates "sensible" programs. This talk will introduce the state-of-the-art, future directions, and challenges regarding this problem.

A Framework for Formal Specification and Verification of Security Policies
Devrim Ünal, Boğaziçi University

ABSTRACT: Formal methods for specification and verification of security policies ensure that the security policy is consistent and satisfied by the network elements in a given network configuration. We present a framework for specification and verification of security policies based on formal methods. The formal languages used for specification are Predicate Logic and Ambient Calculus. The framework is directly usable by system administration personnel. Formal specifications are developed and verified in an incremental and automatic manner without the need of formal methods knowledge by the user.

Formal Modeling and Analysis of Trust Based on Securities of Services
Şerif Bahtiyar, Boğaziçi University

ABSTRACT: Trust based systems are complementary solutions to existing traditional security mechanisms in computer science. Therefore, trust is a significant topic of the security research. It has been recognized that the trust assessment of security systems in dynamic environments with multiple entities, each with its own changing needs from the security systems, is a complex task. Formal representation and analysis of trust is a one way to accomplish with this complex task. In this presentation, the relationship between trust and security will be presented to clarify the problem in our work. Then our proposed architectural approach to assess system trust based on the security of a service will be expressed briefly. Additionally, I will talk about our initial work of a formal model for trust assessment of a service. Moreover, the presentation will include the definition of trust and types of trust to clarify the meaning of different trust definitions in literature. The properties and general models of trust in networks will be also explained. Finally, some open problems in the area of trust research are presented.

Formal Security Analysis of Mobile Ad-Hoc Network Routing Protocols
using Model Checking
A. Burak Gürdağ, Boğaziçi University

ABSTRACT: Secure routing in Mobile Ad hoc Networks (MANETs) is a challenging research area. There are many protocols and enhancements proposed in this field. However, there are not much research on providing formal security analysis for such protocols. In this talk, we will summarize and discuss our research on using model checking to analyze security properties of routing protocols for MANETs.

Security aspects of green computing

Murat Cihan, Boğaziçi University

ABSTRACT: Information and communication technologies industry is one of the major carbon emission sources, which affects environmental changes and causes additional infrastructural expenditures. Green computing approaches try to reduce undesired outcomes of high energy demand of computing facilities in micro and macro levels. A body of literature is being built on measuring and optimizing energy efficiency of computing resources, resulting international standardization efforts and recommended actions. However, security issues are not the first concern of these efforts, due to urgent need for functionality. In our study, we introduce reasons why we need security in green computing. In addition, we give some attack models regarding energy efficiency and optimization efforts, followed by discussion on open problems in this area.