The IPSI BgD Transactions on Internet Research

Multi-, Inter-, and Trans-disciplinary Issues in Computer Science and Engineering

A publication of

IPSI Bgd Internet Research Society
New York, Frankfurt, Tokyo, Belgrade
January 2009 Volume 5 Number 1 (ISSN 1820-4503)

Table of Contents:

The IPSI BgD Internet Research Society

The Internet Research Society is an association of people with professional interest in the field of the Internet. All members will receive this TRANSACTIONS upon payment of the annual Society membership fee of €100 plus an annual subscription fee of €1000 (air mail printed matters delivery).

Member copies of Transactions are for personal use only

IPSI BGD TRANSACTIONS ON INTERNET RESEARCH

www.internetjournals.net

Interview: “HPS, a New Microarchitecture”

Patt, Yale

1. *What is the essence of the contribution for which you received the Eckert Mauchly Prize?*

One can never be sure what it is that makes a Jury of Award decide to honor one's achievements. The citation says it is for my contributions to instruction level parallelism and superscalar processor design. If I take this at face value, then the award was for HPS, a new microarchitecture for obtaining high performance processing of irregular applications, and for my branch predictor which enhanced the capability of HPS.

That is, the HPS microarchitecture with the branch predictor formed a major part of the implementation of the microprocessor which is at the heart of every computer system. Improving the capabilities of microprocessors make computer systems execute applications faster.

However, there are many people's work that deserve strong recognition. So one must humbly ask why my work was singled out for extreme praise, rather than someone else's. I do not have an easy answer for that.

2. *What are the impacts of this contribution?*

Prior to HPS and the branch predictor, many of the pseudo-gurus in the computer architecture community were putting an artificial ceiling on how much performance one could obtain from a microprocessor. Our work showed that we have a long way to go before we approach the real ceiling. One immediate result of our work was that some important applications were able to be carried out faster by computer systems that used the results of our work than that ceiling suggested. The more important long-term value, I think, was that we showed that one should not be so quick to put an upper bound on what human ingenuity can produce.

3. *What are the applications of your contribution that may change the everyday life?*

Two things. First the obvious, that tasks can be performed by a computer faster, perhaps twice as fast. What used to take two days takes one day. But far more important is the mind set that we helped to create: that the limit is a long way off. Today, what took a day, takes less than an hour. This impacts many new uses for computers, from predicting weather earlier to allow ships to navigate seas better, to displaying medical data in a way that allows physicians to save lives, to someday providing computer-controlled automobile operation that will eliminate auto accidents. Hardly any part of human life is not touched and helped by higher capability computers.

5. *We learned a lot from your lectures in L Aquila. Can you tell us, what are the issues that we have to teach our kids, so they become creative when they finish studies?*

Tell them to ignore the fads, and master the fundamentals. Fads come into fashion and then go out of fashion. Fundamentals last forever. Fifteen years ago, the programming language of choice was C, which gave way to C++. Today it seems to be Java. Soon, perhaps C#. Perhaps in five years it will be D-flat. The point is that the fads change. One prepares much better with a solid foundation, with lots of math, physics, statistics.

Second I would tell them that to be creative, they need to really understand the fundamentals, not simply memorize some set of equations. Certainly, we need to memorize some things, but the fewer things the better in my view. They need to be able to use what they learn if they are to be creative, and memorizing does not really help accomplish that. Students often complain about exams with "trick" questions,

when in fact the questions were not "trick" questions at all. They were geared to test whether the student really
understood the concept or simply memorized some formulas he hoped to apply.
Deep understanding of fundamentals, is in my view, the key to creative research downstream.

6. *What are the major things to keep in mind, when you form a team for a scientific experiment, or similar?*

That you pick members of the team that are individually strong, both in their intellect and in their resolve. But it is also necessary that they each have respect for the other team members and are willing to listen as well as talk.

7. *What are the people to avoid, when trying to generate a break-through achievement?*

People who are mired in yesterday, unafraid to take risks, afraid to fail. People who never listen, who totally know it all -- or at least think they do. People who spend a lot of time posturing.

8. *What is your opinion about the impact of math?*

I already answered that in one of my answers above. Incredibly important. For several reasons. It obviously provides you with a set of tools, and that is important even though you may not use these tools every day. But more important is the reasoning ability one develops from studying math.

9. *When targeting a major breakthrough, how sensitive one has to be about the direct interests of tax-payers?*

I guess you are asking me how much one should consider the interests of those who are paying for the research in selecting what research problems to work on. That should depend on how the funding arrived. If it came with no strings attached, then the researcher is free to work on whatever
he finds fascinating. If the source of funding is very explicitly to do x,y,and z, then I believe the researcher is compelled to work on x,y,and z. The problem arises when the funding is not accompanied by explicit instructions, but rather implicit intentions. In that case, one can not give a blanket answer. Each situation needs to be dealt with on its own. I do believe the researcher does share the responsibility of determining how unconstrained the donor's intentions are.

10. *What is the major driving force that motivates a person like you to continue to create and generate results after he-she receives such a big prize?*

I guess I do not spend a lot of time looking in the mirror being pleased with myself, and that frees up time to do other things.

Seriously, there is a thrill that one gets when one comes up with new knowledge that changes how the world sees things. In fact, as you perform the experiment or prove the theorem, and are traveling on seas heretofore uncharted, the heart pounds harder as you get closer to the result, and when you get there, all your senses reach a crescendo that is indescribable. It does not happen often, but when it does, wow! It makes you come back for more.

I should also say that research is only part of my life as a professor. I also get to teach. And teaching is really my first love. I get to walk into a classroom and explain things to students, and I get to see their eyes light up when they understand. That is number one with me.

The good news is that research and teaching go hand in hand. They are mutually very symbiotic. I am not sure I would be any good at either if I could not do both.

11. *For small nations like Serbian, what is your advice, which road to take, when it comes to science?*

There are many ways to make a difference, some of them are capital intensive, others require very little capital investment. I would say a country that is not laden with research funds to spend should concentrate on those topics which do not require huge capital investment. In fact, I would go one step further. If a small nation can demonstrate its creative prowess in science that does not require large amounts of capital, it is more likely to attract partnerships with nations that have more capital than creative talent.

12. *What road to take, when it comes to its general future development plans?*

That depends on where that country wants to be in 20 years, and I would be very presumptuous to tell anyone where they should be in 20 years. I do think that an educated work force would be helpful regardless of a nation's goals. So, I would certainly argue for that. But beyond that, I think I will pass.

Future Microprocessors: What Must We do Differently if We Are to Effectively Utilize Multi-core and Many-core Chips?

Patt, N., Yale

Abstract—*The microprocessor f rst surfaced in 1971 with 2300 transistors. Today, some microprocessors have more than a billion transistors on a single chip, and Moore's Law predicts 50 billion transistors in a very few years. Yet this continuing exponential growth in on-chip resources has not resulted in a corresponding improvement in performance. I believe we need to do things differently if we are to take advantage of what process technology has provided. This paper looks at multi-core chips, describes how we got to where we are, exposes some of the myths being promulgated, and explores some of the ways we might do things differently if we are to exploit these increased resources.*

1 Introduction

The microprocessor first surfaced in 1971 with 2300 transistors. Today, some microprocessors have more than a billion transistors on a single silicon die, and Moore's Law predicts 50 billion transistors in a very few years. This continuing exponential growth has unsurprisingly already resulted in the multi-core chip consisting of two, four, or eight cores (processors) on a single chip, and we will soon see the many-core chip consisting of 64 and more cores. Conventional wisdom predicts thousands of cores on a single die in less than a decade. Yet, we have not seen corresponding improvement in performance.

The pseudo-gurus blame it on the "energy wall," claiming that the sum of the energies dissipated by all the cores at even current frequencies is too high and unsustainable. I agree that if we continue to do things as we have, that is true. But I also believe that if we make some fundamental changes to what we provide in the hardware and what we expect of the software, and what we expect of the programmer, we can continue to improve performance at our previous high rate.

First we need to understand how we got to where we are. Why are the most recent offerings from chip manufacturers multi-core and not single core chips? It is also worth exploring some of the multi-core nonsense being advanced. We also need to understand the real benefits and costs of abstraction, and not simply accept abstraction as a fundamental good. Finally, we need to ask whether parallel programming is possible, or whether we should throw up our hands and give up.

• *Y. Patt is Professor of Electrical and Computer Engineering, Professor of Computer Sciences, and the Ernest Cockrell, Jr. Centennial Chair in Engineering at The University of Texas at Austin, Austin, TX 78712-0240.*
E-mail: patt@ece.utexas.edu

2 Why Multi-core Happened

The first microprocessor showed up in 1971 as the Intel 4004, consisting of 2300 transistors, running at a clock frequency of 106 KHz. Process technology continued to improve in two dimensions: our understanding of the process of growing transistors which resulted in larger and larger fault-free dies, and our understanding of photolithography allowing smaller and smaller geometries which resulted in smaller and smaller transistors. Smaller geometries meant higher frequencies. Smaller geometries and larger dies meant more transistors on each die. Gordon Moore initially predicted the number of transistors on each chip would double every year. Ten years later, he revised his prediction to doubling every two years. No matter, the exponential growth in transistor count per die was in place, and has remained true for the past 40 years. Pat Gelsinger, a high-level VP at Intel says the pace will continue to geometries of 10 nanometers – Moore's Law is alive and well!

Initially, this growth in transistor count went to improving the performance of the single-processor core. Motorola introduced a 256 byte Instruction cache on their MC 68020 in the early 1980s. Pipelining enabled intra-instruction concurrency, which quickly resulted in separate on-chip first level instruction and data caches. It also demonstrated the need for branch prediction, which in turn showed the value of speculative execution. Functional units became more substantial, with the x86 floating point unit being installed on the processor chip in the Intel 486. As the number of transistors on a chip grew to tens of millions, we saw more and more sophisticated branch predictors, a larger and larger second level cache, more and more on-chip functional units, the arrival of the multi-media instruction set, trace caches, and simultaneous multithreading.

However, as the turn of the century saw the number of transistors on a chip pass into the hundreds of millions, it just got too hard to design more and more improvement into that uniprocessor core. Some engineers tried and failed. A much easier solution – put the extra transistors into the second level (L2) cache. This resulted in the next round of microprocessors having larger and larger L2 caches until the size of the L2 cache dwarfed the processor it was supporting. The first Pentium M chip had 77 million transistors, 50 million of which was allocated to the L2 cache. The next Pentium M chip had 140 million transistors, with 117 million allocated to the L2 cache. Bigger L2 caches provide some benefit, but is it enough to justify the use of these enormous resources?

The next step was the obvious and easiest next step: use the transistors to replicate the core. A larger, more effective branch predictor or trace cache would take a lot of thought. Replicating the core takes comparatively little thought. So, Intel came out with Core2Duo, Core2Quad, AMD produced the four core Barcelona, IBM gave us P4, P5, P6, and will soon give us P7.

My reason for describing multi-core as such is not to take a cheap shot at processor designers. On the contrary, doing anything other than what they have done would have been very, very hard. And, it is not clear that the corporate world would have allowed them to do it. But, because we have today multi-cores of their current type is no reason we should accept them as the correct multi-cores. I would rather we adopt the position that we can do better. In section 4, I will suggest how.

3 Multi-core begets Multi-nonsense

I think the biggest problem with multi-core is that, given its existence, too many people have come up with nonsensical pronouncements justifying its existence instead of critically looking at current multi-core chips, and asking how we can do better.

3.1 ILP is dead

The first justification for multi-core is that the benefits of single-core performance have been exhausted. As evidence, one is usually shown performance figures for single core performance as a function of the number of transistors on the chip. We double the number of transistors; performance goes up by 5%. ”Hardly worth it,” the naysayers claim. A closer look, as stated in Section 2, provides the insight that those extra transistors did not go into the core, they went into the L2 cache. So, we use the doubled transistor count to increase the size of the L2 cache and then blame the processor for not doing any better. Perhaps we have the wrong culprit.

Studies have shown that better branch predictors get much better performance. Microarchitecture research has come a long way from the 2 bit saturating counter of 1981 which showed up on the Intel Pentium chip ten years later to some of the sophisticated two-level predictors like TAGE [1] that have appeared in the literature in the past few years.

Tailored accelerators also improve performance. Even the anemic AMD 29000 had a Find First instruction in the mid-1980s that used a simple priority encoder as an accelerator to speed up the Find First operation from a code fragment taking tens of instruction cycles to a single cycle instruction.

I submit that ILP is far from dead if one wants to seriously address what can be accomplished with a sophisticated single core.

3.2 Make the cores simple

Pseudo-gurus are touting multi-cores with thousands of identical simple cores. The mantra seems to be: the more cores the better, and the way to get more is to keep them simple. We actually played that tune in the 1980s where designers attempted to design many, very simple cores into a single chip. The transistor count was much smaller then – barely a million, so the simple cores were really simple. But the mentality remains.

I would argue for two things: accelerators and an asymmetric chip with many simple cores and a very few (perhaps only one) heavy-weight cores. I call such a chip an Asymmetric Chip MultiProcessor (ACMP). The many simple cores are for the embarrassingly parallel parts of parallel algorithms. The heavyweight core is for the serial bottleneck first identified by Gene Amdahl (Amdahl’s Law), and for executing code in critical sections where executing fast and thereby leaving the critical section more quickly provides great performance benefit [2], [3].

As to accelerators, there is no limit to what we can provide on the chip for handling important functions, if we have the ability to power off those accelerators when not in use. Transistor count is freely available. Transistors that are allowed to sit by idly draining leakage current are not freely available. The bottom line: we identify, like the AMD engineers did 20 years ago with Find First, a set of accelerators that are really useful when they are needed, and we design them into the chip. When not needed, they sit there powered off, doing no harm. When needed, they provide great benefit.

Even the classical processing elements can be made to perform greater benefit if we invest more of the transistor budget in making them more sophisticated. Subordinate simultaneous multithreading [4] provides an opportunity for improving the performance of on-chip resources, Runahead execution [5] provides a mechanism for concurrently satisfying off-chip memory accesses, DIP [6] provides a better cache replacement policy.

The point is that if we assign the transistor budget to improving the performance of a heavyweight core, rather than adopting the mantra that more simple cores are better, there are lots of places where engineering ingenuity may prevail.

3.3 Make the interface high

Another mantra we hear constantly now that we are in the era of multi-core is that the interface to the software needs to be high. Otherwise, most programmers will be overwhelmed. It is not my place to indict most programmers. However, there are programmers – perhaps 5 to 10 percent of them, who can take advantage of the underlying hardware. For them, not providing a low level interface denies them the ability to practice their craft.

The answer, I think, is quite straightforward: multicore demands multiple interfaces, each tuned to the skill set of a set of programmers. Certainly we need to provide a high level interface for those programmers who need it. But, can't we do that while also providing the low level interface for those programmers who can use it? And, while we are at it, we need that middle layer of software to bridge the gap between what the high level programmers require to be effective and what the low level programmers can effectively use.

4 ABSTRACTION: A DOUBLE-EDGED SWORD

Conventional wisdom dictates that abstraction is an absolute good. The higher the level of abstraction one works at, the more productive one is. I would add: True if you do not care about performance.

I have lots of examples from technology and from non-technology (i.e., life) to demonstrate this. My favorite example was a taxi ride I once had in the financial district of Manhattan, in the heart of New York City. I needed to get to JFK airport. I got into a cab, and announced, "Take me to JFK airport." What could be a higher level of abstraction than that. I did not tell the driver how to drive the automobile. At a higher level, I did not tell him what route to take. I just gave him five words – take me to JFK airport. In fact, I probably could have reduced my instructions to one word: "JFK"! The problem was that since I did not deal with any lower level of abstraction, the driver was free to take me along whichever route he wished. In this instance, it was over the Triboro Bridge, which substantially increased my fare, and almost made me miss my plane.

A second example comes from the early VLSI design days of the late 1970s, when designers first started designing circuits while being oblivious to how transistors actually worked. In one instance that I am very familiar with, the result was a disaster. Then the engineers looked at each other, noted that they really understood how transistors worked, small-signal models and all, and set out to redesign the chip, putting their lower level knowledge to work. The result: a working chip.

The point is that while it is true that raising the level of abstraction increases productivity, it puts one at the mercy of everything below the level one is working at. That is, everything below the level must work perfectly. It also prevents one from taking advantage of what goes on at the lower levels.

Raising the level of abstraction and improving performance are fundamentally at odds with each other. To show how this is relevant to the chip multiprocessor, I need to introduce the transformation hierarchy.

4.1 The transformation hierarchy

The transformation hierarchy (see figure 1) is the name I gave 25 years ago to the mechanism that converts problems stated in natural language (English, Serbian, Hindi, Japanese, etc.) to the electronic circuits of the computer that actually do the work of producing a solution. The problem is first transformed from a natural language description into an algorithm, and then to a program in some mechanical language, then compiled to the ISA of the particular processor, which is implemented in a microarchitecture, built out of circuits. At each step of the transformation hierarchy, there are choices. These choices enable one to optimize the process to accommodate some optimization criterion. Usually, that criterion is microprocessor performance.

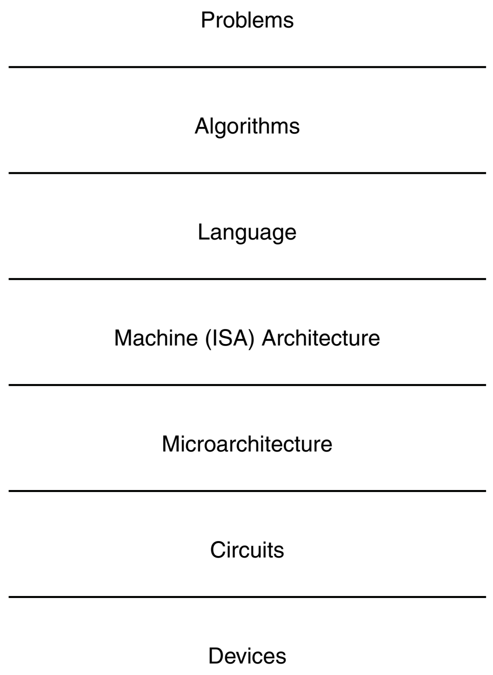

Fig. 1. Levels of Transformation

Up to now, optimizations have been done mostly within each of the layers, with artificial barriers in place between the layers. It has not been the case (with a few exceptions) that knowledge at one layer has been leveraged to impact optimization of other layers. One could argue that such is as it should be, since this approach, working within one's layer of abstraction allows one to become an expert at that layer. It also provides great comfort since most people know very little outside their layer.

Ask computer science graduates if their choice of sorting algorithm would depend on whether all the data to be sorted can reside in memory at the same time or whether it has to be piecemeal brought in from the disk. Their answers more often than not would be that it does not matter. Even a casual reading of Knuth, vol 3 identifies the differences between ”internal sorts” and ”external sorts.”

4.2 What if we break the layers?

If we break the layers, and introduce discomfort to those who have operated within their own level of abstraction, we open up new opportunities to improve performance.

We have already seen this with respect to the Compiler and Microarchitecture. Predicated execution can get rid of conditional branches if the microarchitecture supports conditional execution. The Block-structured ISA [7] compiled programs into instructions having the granularity of a basic block. With microarchitecture support for this instruction, operations within an instruction can be reordered, resulting in decreasing the number of bypass networks. With explicit linkages between operations within a basic block, register pressure can be decreased dramatically. The net effect is increased performance.

In a larger scope, if those working at the algorithm layer talked more to those working at the microarchitecture layer, accelerators could be more effectively identified. The algorithm people know what they want, the microarchitects know whether they can provide it.

If people working at the circuit layer talked to the microarchitects, things like the ill effects of soft errors due to increased frequencies perhaps could be eliminated.

5 Parallel programming

Everyone says, ”Thinking in parallel is hard.” Perhaps ”thinking is hard.” Is thinking in parallel hard, or are people told it is hard often enough that it is a self-fulling prophesy.

I tried an experiment in the first year course ”Intro to Computing” that I teach. I gave the class – on the spot, with no warning, the following problem: Suppose it takes a computer five units of time to perform a multiply. How many units of time (approximately) to compute 10 factorial? Answer: approximately 40 units, since eight multiplies are required. Unsurprisingly, everyone got it correct. Then I said, suppose you have two processors working together, how many units approximately? More than half, with very little effort came up with 25 units, noting that if processor A directs processor B to compute 5!, processor A, after multiplying 10x9x8x7x6 can multiply its result by 5! supplied by processor B.

My point is a simple one. Certainly, considering all asynchronous actions that can go wrong in parallel processing is not easy, but I wonder how much better we can do if we introduce parallel thinking early in the computer scientists’ education before they are convinced that it is hard.

6 Some final observations

My point in this paper has been to call attention to this new world of microprocessors we have recently entered, called multi-core, and ask how we are to continue to exploit the enormous potential it can provide. First, I argue that the existing multi-core paradigm – lots of identical cores – is not the right answer. I submit that a better solution would be ACMP with serious accelerators for doing serious work.

Second, I would argue that our current approach to provide high level interfaces where all programmers can retain their current level of abstraction will not best utilize the chips that we could produce. I recognize that people’s desire to remain within the comfort zones provided by their respective levels of abstraction and their hesitancy to embrace parallel thinking is real and pervasive. However, I believe this provides a tremendous opportunity for education. Education is never a short term solution, but it is a solution we need to embrace immediately.

Finally, I believe that if we break with current trends, the multiprocessor of the future will be a many core processor with lots of simple cores to handle the embarrassingly parallel parts of parallel algorithms, but with at least one heavyweight core to handle the serial bottleneck called Amdahl’s Law and the execution of critical sections. I believe that this heavyweight core will continue to acquire structures needed to continue to improve the performance of single instruction stream programs. I think the chip will have accelerators identified by algorithms people, and support for structures identified by compiler people. I believe the chip will provide multiple interfaces for multiple types of programmers. But I believe none of this can happen unless we break with the past and at least some of us do things differently.

Acknowledgments

I don’t know about other professors, but for me, my abilities are constantly challenged and improved by my interaction with very bright, very focused graduate students, starting with my one PhD student at NC State in the early 1970s, my PhD students at Berkeley in the 1980s, at Michigan in the 1990s, and at Texas since 1999. Since space does not allow me to name them all, I must name none of them since each in some unique way is my ”best student,” and I gratefully acknowledge them all.

References

[1] A. Seznec, “A 256 kbits l-tage branch predictor,” Journal of Instruction-Level Parallelism (JILP) Special Issue: The Second Championship Branch Prediction Competition (CBP-2), vol. 9, 2007

[2] M. A. Suleman, O. Mutlu, M. K. Qureshi, and Y. N. Patt, “Accelerating critical section execution with asymmetric multi-core architectures,” In Proceeding of the 14th international Conference on Architectural Support For Programming Languages and Operating Systems (ASPLOS’09), 2009.

[3] M. A. Suleman, Y. N. Patt, E. A. Sprangle, A. Rohillah, A. Ghuloum, and D. Carmean, “ACMP: Balancing Hardware Efficiency and Programmer Efficiency,” HPS Technical Report, TR-HPS- 2007-001, The University of Texas at Austin, 2007.

[4] R. S. Chappell, J. Stark, S. K. Reinhardt, Y. N. Patt, S. P. Kim, ”Simultaneous Subordinate Microthreading (SSMT),” In Proceedings of the 26th Annual International Symposium on Computer Architecture (ISCA’99), 1999.

[5] O. Mutlu; H. Kim; Y. N. Patt, ”Techniques for efficient processing in runahead execution engines,” In Proceedings of the 32nd International Symposium on Computer Architecture (ISCA ’05), 2005.

[6] M. K. Qureshi, A. Jaleel, Y. N. Patt, S. C. Steely, and J. Emer, “Adaptive insertion policies for high performance caching,” In Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA ’07), 2007.

[7] S. Melvin, and Y. N. Patt, “Exploiting fine-grained parallelism through a combination of hardware and software techniques,” In Proceedings of the 18th Annual International Symposium on Computer architecture (ISCA ’91), 1991.

Performance Analysis of WSN Clustering Algorithms using Discrete Power Control

Aslam, Nauman; Robertson, William; Phillips, William

Abstract—*Research efforts in the area of Wireless Sensor Networks (WSNs) are heavily dependent on the development and evaluations of protocols through simulations. Therefore, realistic simulations are vital for the development of viable protocols. However, most of the research efforts that depend on simulations tend to use non-realistic parameters and assumptions. Such examples include use of infinite transmit power levels and no consideration of radio propagation loss and irregularities. These assumptions are common among many clustering protocols, which lead to incorrect estimation of performance metrics such as network lifetime, energy consumed per bit, and connectivity. In this paper we modify clustering protocols by incorporating a model compliant with Crossbow MICAz motes. The energy consumption model takes into account the discrete transmit power levels of the CC2420 radio ship used by MICAz sensor nodes. The radio propagation path loss is modeled by using the Lognormal Shadowing Model. We evaluate a number of clustering protocols including LEACH, HEED, EECS and MOECS. We also present results that demonstrate how realistic assumptions can effect the system behavior in comparison with the results obtained by assuming ideal conditions.*

Index Terms—*Wireless Sensor Networks, Clustering, Discrete power, Energy conservation*

1. INTRODUCTION

WIRELESS sensor network (WSNs) have emerged as the-state-of-the-art technology in data gathering from remote locations by interacting with physical phenomena and relying on collaborative efforts by a large number of low cost resource constrained devices [1]. Each sensor node has an embedded processor, a wireless transceiver for communication, a non replenish-able source of energy, and one or more onboard sensors such as temperature, humidity, motion, speed, photo, and piezoelectric detectors [2]. Once deployed, sensor nodes collect the information of interest from their on board sensors, perform local processing of these data including quantization and compression, and forward the data to a base station (BS) either directly or through a neighboring node.

Manuscript received July 10, 2008, accepted January 2009.
Nauman Aslam is with the Engineering Mathematics and Internetworking Department, Dalhousie University, NS, Canada.

In recent years, clustering has emerged as a popular approach for organizing the network into a connected hierarchy [3]. By using clustering, nodes are organized into small disjoint groups called clusters. Each cluster has a coordinator, referred to as a Cluster Head (CH), and a number of member nodes. Clustering results in a hierarchical network in which the CHs form the upper level and member nodes form the lower level. In contrast to flat architectures, clustering provides distinct advantages with respect to energy conservation by facilitating localized control and reducing the volume of inter-node communication. Moreover, the coordination provided by the CH allows sensor nodes to sleep for an extended period thus allowing significant energy savings. By adapting to a clustered topology, WSNs can share many advantages that result in a direct or indirect impact on the energy efficiency. Some of the advantages as the result of using clustering include network scalability, local route set up, bandwidth management, minimizing communication overheads and data aggregation. As a result of the advantages offered by clustered topology, a number of clustering protocols [4-12] were developed for WSNs. However, most of these protocols contained simplistic assumptions that are far from reality. For example, the energy model originally proposed by LEACH or its slight variation is used. Moreover, assumption like infinite transmit power levels and no consideration of path loss further biases the performance results in an optimistic manner.

This paper makes three contributions which are summarized as follows. Firstly, we take into account the fact that sensor nodes are limited to a few discrete power levels. As an example, Crossbow MICAz [13] motes that use Chipcon CC2240 radio chip. CC2240 chip is limited to only seven transmit power levels. Secondly, a Lognormal Shadowing model is used for calculating radio propagation path loss. Earlier studies [14] have shown that this model to be more accurate for cellular system, making it a favorable and realistic option as compared to the free space model. Thirdly, we propose a simple scheme that allows sensor nodes to adapt to correct output power level based on distance from the transmitter and path loss.

The remainder of this paper is organized as follows. In Section 2, we describe the network model and assumptions. Section 3 summarizes the radio propagation path loss model and Section 4 presents the discrete power control algorithm. In Section 5, we present simulation results that apply both conventional and discrete power model to clustering protocols including MOECS [11] EECS [8] LEACH [4] and HEED. [7]. Section 6 presents a summary of related work. Finally, conclusions and future work is discussed in Section 7.

2. Network Model and Assumptions

The following assumptions are made for the sensor network under consideration:

1. Nodes are dispersed randomly following a Uniform distribution in a 2-dimensional space.
2. The location of the BS is known to all sensors. The BS is considered a powerful node having enhanced communication and computation capabilities with no energy constraints.
3. All nodes remain stationary after deployment. All nodes are homogeneous in terms of energy, communication and processing capabilities.
4. Nodes are location unaware i.e. they are not equipped with any global positioning system (GPS) device.
5. The nodes are capable of transmitting at variable power levels depending on the distance to the receiver as in [15]. For instance, MICAz Motes use the MSP430 [16] [17] series micro controller which can be programmed to 7 different power levels.
6. The nodes can estimate the approximate distance by the received signal strength, given that the transmit power level is known, and the communication between nodes is not subject to multi-path fading.
7. A network operation model similar to that of [4, 7, 8] is adopted here, which consists of rounds. Each round consists of a clustering phase followed by a data collection phase.

3. Radio Model

This section presents the radio propagation model adopted in simulations. Equation (1) provides a generalized expression for the strength of a signal at the receiver considering the path loss and fading effects.

$$P_{Rx} = P_{Tx} - PL - Fading \qquad (1)$$

Where P_{Rx} is received power in dBm, P_{Tx} is the power at the transmitter in dBm and PL is the path loss. For path loss calculations we use a Log-Normal Shadowing model to provide the value of signal loss $L(d)$ between a transmitting node and the receiving node located at a distance 'd' from each other. The value of $L(d)$ is given by;

$$L(d) = L(d_o) + 10\beta \log\left(\frac{d}{d_o}\right) \qquad (2)$$

Where β is the path loss exponent and $L(d_o)$ is the path loss measured at distance d_o. These parameters can experimentally determined or taken from sources such as [14]. The fading effect, as mentioned in equation (1) can be modeled by adding a Gaussian random variable X_σ (with standard deviation σ^2) in equation (2). Hence the received power is given by;

$$P_{Rx} = P_{Tx} - L(d_o) - 10\beta \log\left(\frac{d}{d_o}\right) - X_\sigma \quad (3)$$

Figure 1 presents the typical values of path loss at a receiver plotted against distance from the transmitter.

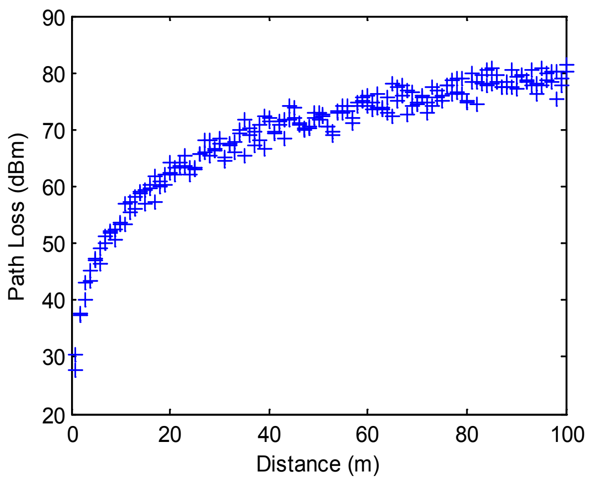

Figure 1: Path Loss Vs. Distance(d_o=87 m , β = 2.5)

4. Discrete Power Level Selection Algorithm

Most of the clustering protocols described assume that sensor nodes can adjust their power according to the exact distance. Hence the corresponding energy consumption computed in this manner will always be different for two different measurements of '*d*'. In reality, the transmit power level of the sensor node can only be adjusted to discrete values which may result in one power level for multiple values of distance. Therefore the resulting energy consumption for the two different distance would be same. Table 1 provides energy consumed in transmission of a 100 byte packet considering the different power levels used by CC2420. The values of current drawn I_x (3rd column) are taken from the data sheet [16]. It is also worthwhile to note that receiving a packet draws a fixed amount of current, hence the energy consumed remains constant for a given packet size. Table 2 provides the value for energy consumed in receiving a packet of 100 bytes.

Table 1: Energy consumed per packet for different power levels used in CC2420 . (Packet size = 100 bytes, channel rate 250 Kbps, V_{DD}= 1.5 V)

Power Level (k)	P_{Out} [dBm]	I_x [mA]	P_{Tx} [mW]	E_{Tx} /packet [μJ]
1	0.00	17.04	30.67	98.14
2	-1.00	15.78	28.40	90.88
3	-3.00	14.63	26.33	84.26
4	-5.00	12.27	22.08	70.66
5	-10.00	10.91	19.62	62.78
6	-15.00	9.71	17.47	55.90
7	-25.00	8.42	15.15	48.48

Table 2: Energy consumed in reception (Packet size = 100 bytes, channel rate 250 Kbps, V_{DD}= 1.5 V)

I_x [mA]	P_{Rx} [mW]	E_{Rx} /packet [μJ]	E_{Rx} /bit [μJ]
19.60	35.28	112.90	0.1411

In the operational model used for most clustering protocols the advertisement messages from CHs are sent at the fixed (known) power levels. The cluster membership phase involves sensor nodes to adjust their power levels according to their distance from the CH. We propose an algorithm that allows sensor nodes to select the appropriate power level for communicating to the CH. Figure 2 presents the flowchart for power level selection algorithm. It can be noted from Table 1 that power level 7 corresponds to the lowest and power level 0 corresponds to the highest power output.

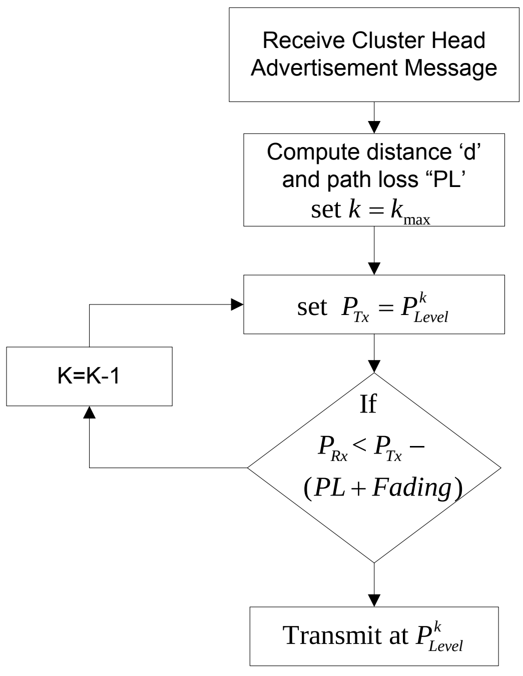
Receive Cluster Head Advertisement Message
Compute distance 'd' and path loss "PL" set $k = k_{max}$
set $P_{Tx} = P^k_{Level}$
K=K-1
If $P_{Rx} < P_{Tx} - (PL + Fading)$
Transmit at P^k_{Level}

Figure 2: Flow Chart for Discrete Power Level Selection Algorithm

Once sensor nodes have estimated the distance to the CH, it computes path loss and fading. It then sets its transmit power level to the lowest value. Based on the computed path loss and fading it calculates the projected received power at the CH. If the projected received power is greater than the receiver sensitivity, the current power level is used for transmission, otherwise the transmit power level is incremented and same procedure is repeated until the appropriate level is found or highest power level is reached.
Figure 3 presents a comparison of energy consumed in transmission using the discrete power level model and conventional (LEACH) energy model. Measurement for both models is obtained using a packet size of 100 bytes. The discrete power model assumes path loss exponent equal to 2.5 and d_0 equal to 87 m. It can be observed that the discrete power model render more energy consumption at 75 m. It is worth noting the conventional model results in a false optimistic performance. Moreover, a typical round data transmission rounds consists of a large number of transmission from sensor node to the sink and an aggregated packet (single transmission) from the cluster head to the sink. In the conventional LEACH model, the former follows the energy consumption is proportional to the square of distance and in the latter the energy consumption is proportional to forth power of distance [4]. Results presented later in Section 5 demonstrate that the conventional model results are significantly deviated from the discrete power model.

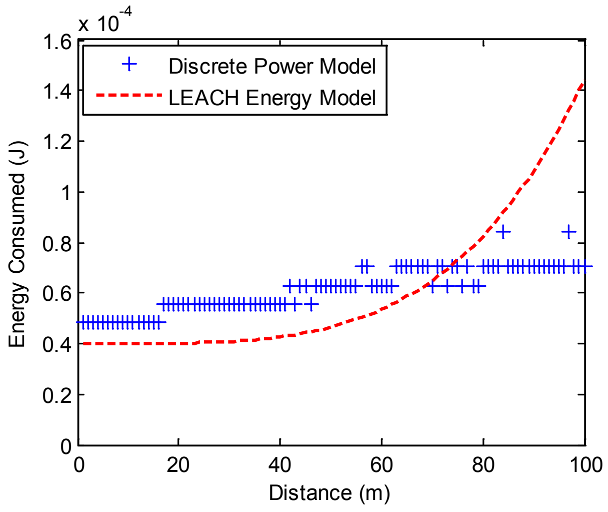

Figure 3: Comparison of Energy Consumed in Transmission with Conventional LEACH Energy Model and Discrete Power Model (Packet Size =100 bytes)

5. Simulation Results

This section presents the performance analysis of clustering protocols namely LEACH, HEED, EECS and MOECS. We used MATLAB as the simulation environment. As outlined in Section 2 we used network of 100 nodes placed in an area of 100 x 100 m. The BS location is taken as (50,150) m. All clustering protocols are evaluated with and without (conventional) the discrete power selection scheme presented in Section 3. Each simulation experiment is performed on a unique topology and consists of several rounds of cluster set up phase and data transmission phase. In each round a set of new cluster heads is elected and the non-cluster head nodes send five data packets to their associated cluster head. We also assume that the cluster head is capable of data aggregation and data received from member nodes is therefore sent in aggregated form. In performance analysis of different clustering schemes using both conventional and discrete power model remains on metrics related to network life and energy conservation. The network life time is measured in data collection rounds till the first node runs out of its energy. The network lifetime measured to the death of first node is extensively used in the literature including [4, 7, 8, 11]. Figure 4 presents a performance comparison of network life using both conventional and discrete power model for various clustering protocols mentioned earlier. It is clearly evident that the realistic discrete model results in a significant decrease (in excess of approximately 100% for all protocols) in network life as compared to the conventional model. Based on the discussion in Section 4, a large number of transmissions at higher energy consumption in the discrete model contribute to a quick depletion of sensor nodes' energy.

Figure 5 illustrates results for the random topology where y-axis indicates the mean residual energy of the system normalized to number of nodes and x-axis denotes the number of rounds. It can be observed that the mean residual energy of the system in case of discrete power model is lower than that of the conventional model for all protocols. A sharp slope in case of the discrete power model is indicative of the sensor nodes losing their energy at a much faster rate as compared to the conventional model. These results are also corroborated with the network life results presented in Figure 4.

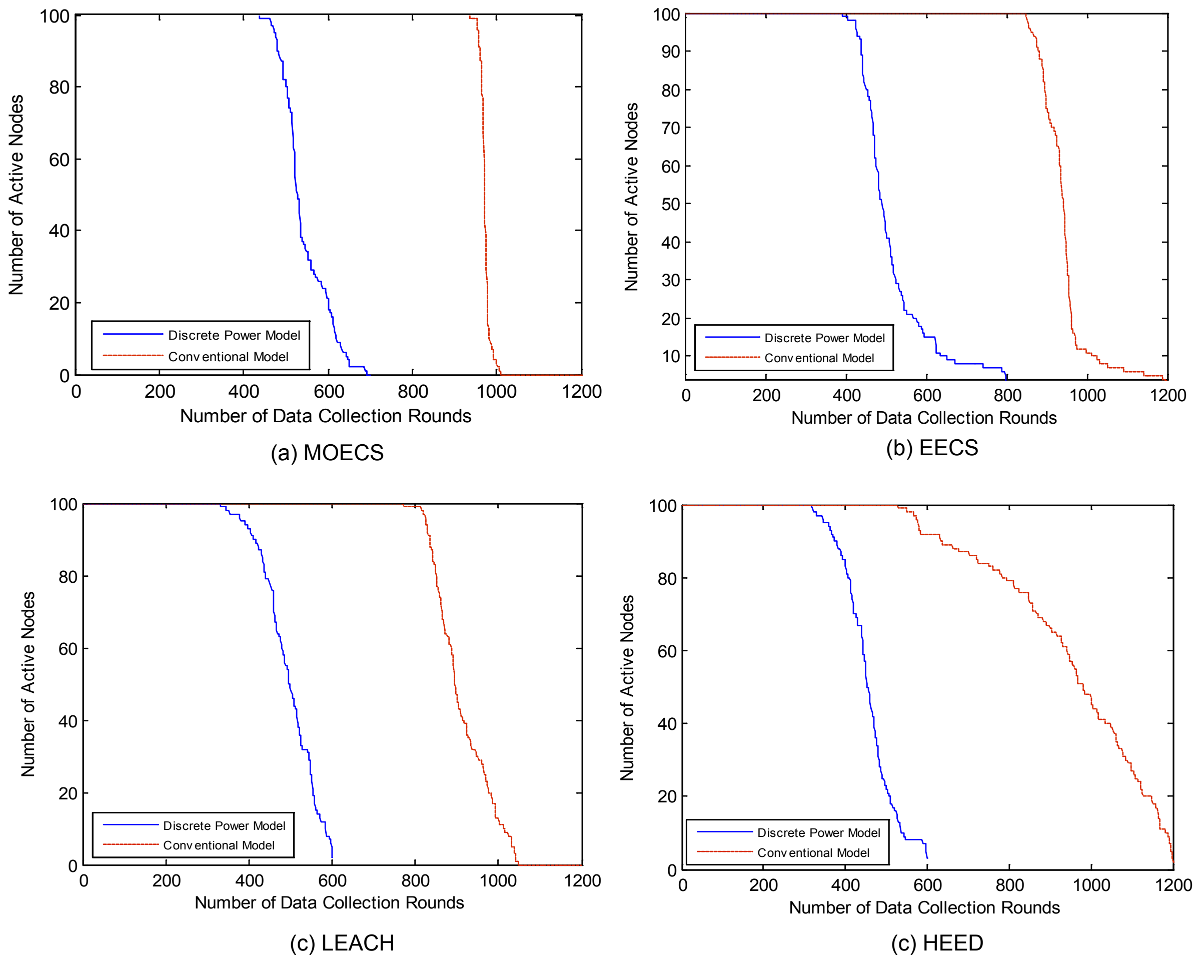

Figure 4: Network Life in Rounds for a) Multi-Objective Energy-efficient Clustering Scheme-MOECS, b) Energy Efficient Clustering Scheme-EECS, c) Low Energy Adaptive Clustering Hierarchy-LEACH, d) Hybrid Energy Efficient Distributed Clustering- HEED

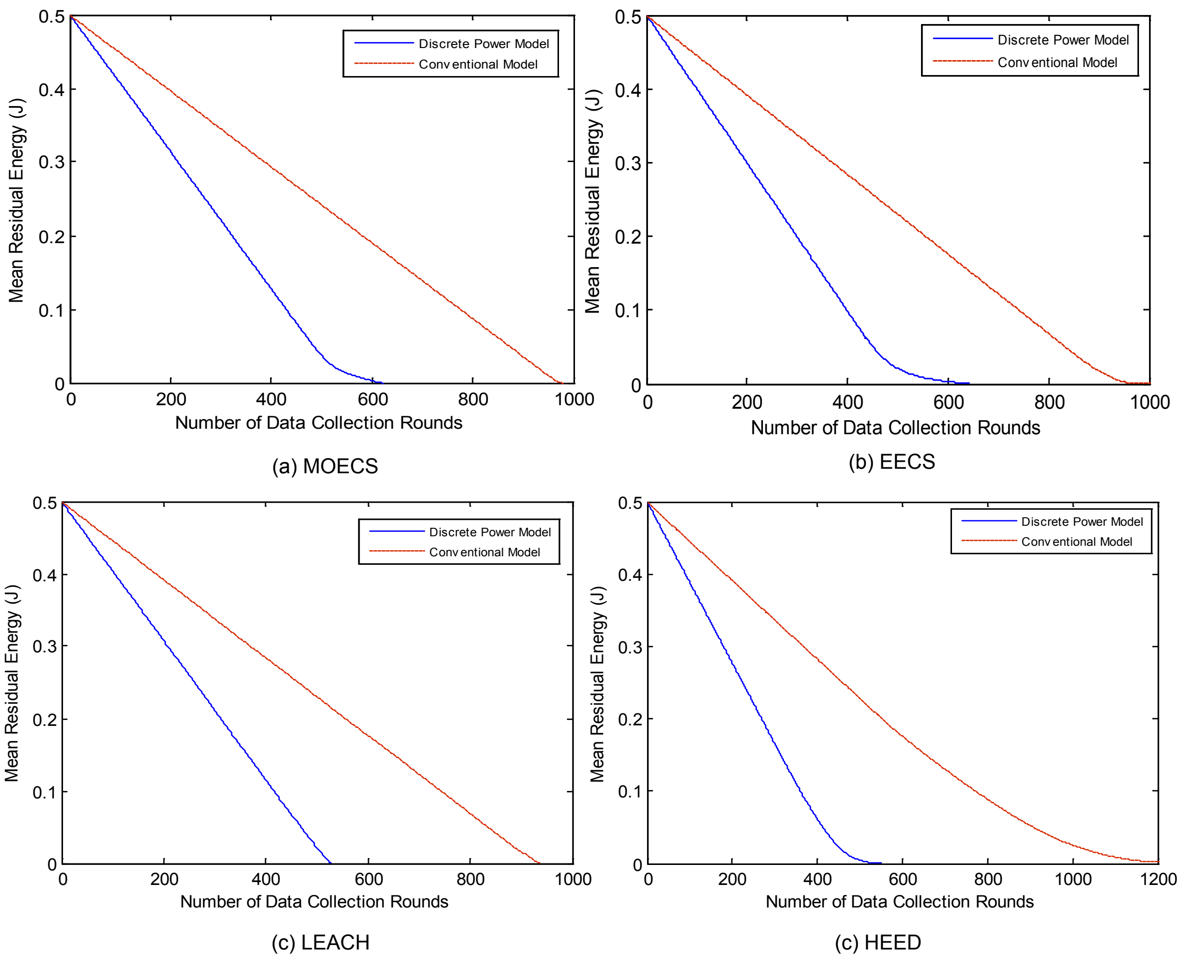

Figure 5: Mean Residual Energy Vs. Number of Rounds for a) Multi-Objective Energy-efficient Clustering Scheme-MOECS, b) Energy Efficient Clustering Scheme-EECS, c) Low Energy Adaptive Clustering Hierarchy-LEACH, d) Hybrid Energy Efficient Distributed Clustering- HEED

We also investigate the energy consumed in transmission (end-to-end, i.e. from sensor node to the BS) on per packet basis. Results in Figure 6 demonstrate the measurements for this statistics for both models. Again it is evident that the discrete power model results in excess of 100% extra energy consumption as compared to the conventional model.

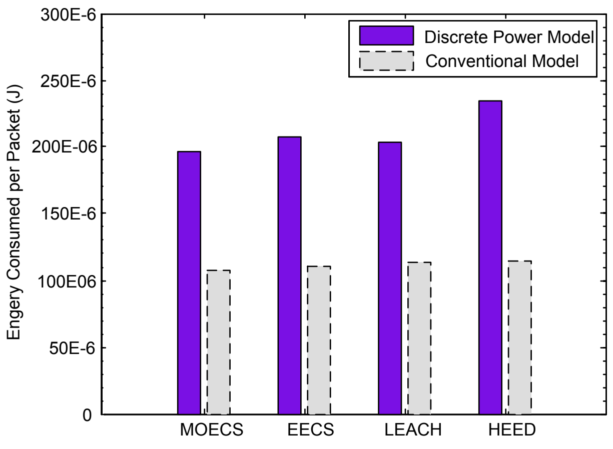

Figure 6: Mean Energy Consumed in Transmission per Packet from Sensor Node to the BS

6. Related Work

Recent relevant studies have [18] [19, 20] have focused on incorporating the realistic radio and energy consumption models in WSNs. The authors in [18] presented an energy model for calculating the transmission and reception cost. This model is based on the first order model presented in [15]. Research results presented in [19] are collected from field experiments involving few sensor motes. These experiments were performed to collect link quality index (LQI) and received signal strength indication (RSSI). The investigations show that transmission power costs do not always increase as the distance increases. Our work in closely related to [19, 20] which are also focused on the CC2420 radio chip. In [20], a similar model as proposed in this paper is implemented in the PROWLER [21] simulator. Our work focuses on the clustering protocols.

7. Conclusions

In this paper we investigated the effects of incorporating realistic radio models in the simulation of cluster-based WSNs. Most sensor hardware platforms make use of discrete power levels. Therefore incorporating such models into simulations would bridge the gap between simulation and experimentation results. We proposed a simple algorithm that allows sensor nodes to choose appropriate power level based on inducted path loss and distance. We evaluated four clustering protocols using metric relevant to the energy conservation following both discrete power and conventional models. We showed that estimates of network life using discrete power model differ significantly than that of conventional model. Our future work will address effects of controlled mobility, and realistic deployment strategies.

References

[1] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*: Wiley, 2005.

[2] F. Zhao and L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach*: Morgan Kaufmann, 2004.

[3] O. Younis, M. Krunz, and S. Ramasubramanian, "Node clustering in wireless sensor networks: recent developments and deployment challenges," *Network, IEEE*, vol. 20, pp. 20-25, 2006.

[4] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, " An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 660-670, 2002.

[5] S. Bandyopadhyay and E. Coyle, "An Energy-Efficient Hierarchical Clustering Algorithm for Wireless sensor Networks," *Proceedings of IEEE INFOCOM*, 2003.

[6] P. Ding, J. Holliday, and A. Celik, "Distributed energy-efficient hierarchical clustering for wireless sensor networks," *Lecture notes in computer science*, pp. 322-339, 2005.

[7] O. Younis and S. Fahmy, " HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks," *Transactions on Mobile Computing*, vol. 3, pp. 660-669, 2004.

[8] M. Ye, C. F. Li, G. H. Chen, and J. Wu, "EECS: An Energy Efficient Clustering Scheme in Wireless Sensor Networks," *IEEE Int'l Performance Computing and Communications Conference (IPCCC)*, pp. 535-540, 2005.

[9] C. Li, M. Ye, G. Chen, and J. Wu, "An energy-efficient unequal clustering mechanism for wireless sensor networks," *Proceedings of the 2nd IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS'05)*, pp. 8.

[10] M. Demirbas, A. Arora, and V. Mittal, "FLOC: A fast local clustering service for wireless sensor networks," *Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS/DSN 2004)*, 2004.

[11] N. Aslam, W. Robertson, S. C. Sivakumar, and W. Phillips, "Energy Efficient Cluster Formation Using Multi-Criterion Optimization for Wireless Sensor Networks," in Proc. 4th IEEE Consumer Communications and Networking Conference (CCNC), 2007.

[12] S. Yi, J. Heo, Y. Cho, and J. Hong, "PEACH: Power-efficient and adaptive clustering hierarchy protocol for wireless sensor networks," *Computer Communications*, vol. 30, pp. 2842-2852, 2007.

[13] C. T. Inc., "http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf."

[14] T. S. Rappaport, *Wireless Communications: Principles and Practice*: IEEE Press Piscataway, NJ, USA, 1996.

[15] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for," *IEEE Transactions on Wireless Communications*, vol. 1, 2002.

[16] Chipcon, "CC2420, 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver Data Sheet," 2004.

[17] I. Texas Instruments, "MSP430x13x, MSP430x14x Mixed Signal Microcontroller. Datasheet, ." 2001.

[18] K. Dasgupta, K. Kalpakis, and P. Namjoshi, "An efficient clustering-based heuristic for data gathering and aggregation in sensor networks," *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, vol. 3, 2003.

[19] M. Mallinson, P. Drane, and S. Hussain, "Discrete Radio Power Level Consumption Model in Wireless Sensor Networks," *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, pp. 1-6, 2007.

[20] A. Barberis, L. Barboni, and M. Valle, "Evaluating Energy Consumption in Wireless Sensor Networks Applications," *Proceedings of the 10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)-Volume 00*, pp. 455-462, 2007.

[21] G. Simon, P. Volgyesi, M. Maroti, and A. Ledeczi, "Simulation-based optimization of communication protocols for large-scale wireless sensor networks," *Aerospace Conference, 2003. Proceedings. 2003 IEEE*, vol. 3, 2003.

Nauman Aslam received the B.Sc. (Eng.) degree from the University of Engineering and Technology, Lahore, Pakistan, in 1994 and the Master's of Engineering degree in Internetworking (2003) and Ph.D. degree from the Dalhousie University, Halifax, Nova Scotia, Canada, in 2008. He is currently an Assistant Professor at the Engineering Mathematics and Internetworking Department at Dalhousie University. His research interests include cluster-based wireless sensor networks, wireless ad hoc networks, IP communications, and Quality of Service.

William J. Phillips received the B.Sc. degree in Engineering Mathematics and the M.Sc. degree in Mathematics from Queen's University at Kingston and the Ph.D in Mathematics from the University of British Columbia. Dr. Phillips held visiting positions at the University of Guelph, Queen's University, Dalhousie University, and Saint Mary's University before joining the Department of Applied Mathematics at Technical University of Nova Scotia. He is currently Professor and Head of the Department of Engineering Mathematics at Dalhousie University (merged with the Technical University of Nova Scotia in 1997).

William Robertson received the B.Sc. (Eng. Hons.) degree and the M.Sc. (Eng.) degree from Aberdeen University, Scotland, U.K., in 1967, and 1969 respectively, and the Ph.D. degree from the Technical University of Nova Scotia [(TUNS), now Dalhousie University], Halifax, NS, Canada, in 1986. Since 1983, he has held various positions at TUNS and Dalhousie University. He is currently the Director of the Internetworking Program—a program leading to the Master's of Engineering in Internetworking. His research interests include signal processing and networking, QoS, routing, and wireless applications. Dr. Robertson is a member of the Association of Professional Engineers of Nova Scotia.

From Fraud Vulnerabilities and Threats to Fraud Avoidance and Tolerance

Lilien, Leszek; Bhargava, Akhil; Bhargava, Bharat

Abstract—*Fraud vulnerability and fraud threat assessments can improve the level of trustworthiness in computer systems by reducing fraud attacks. A deep knowledge of fraud vulnerabilities and fraud threats can be utilized to avoid/tolerate fraud attacks. A swindler is a legitimate user who intentionally benefits from the system or other users by deception. We describe an architecture for swindler detection consisting of four components. One of them runs the deceiving intention predictor algorithm to foresee fraudulent intentions. It is effective in uncovering different fraud attack strategies, from naive to smart ones*

Index Terms—*Computer security, fraud, threats, trusted computing, vulnerabilities*

1 Introduction

EVER-GROWING dependence of our civilization on ubiquitous computer systems brings with it an increase in vulnerabilities and threats, including fraud vulnerabilities and fraud threats, experienced by individuals, systems, and enterprises. The need to ameliorate these negative side effects of information technology, ranging from accidental failures to terrorist security attacks, becomes more urgent every day.

1.1 Basic Ideas and Terminology

Vulnerability is defined as a flaw or weakness in system security procedures, design, implementation or internal controls that could be exercised (accidentally triggered or intentionally exploited), and results in a penetration: a security breach or a violation of the security policy [34].

We define *threats* against systems as entities that can cause security breaches [20, 34]. An *attack* is an intentional exploitation of vulnerabilities, and an *accident* is an inadvertent triggering of vulnerabilities. A threat by itself causes no harm. To cause harm, its potentiality must *materialize* as an attack or an accident.

Trust can be defined as a firm reliance on the integrity, ability, or character of a person or thing [36]. This definition extends trust to things.

Fraud (a *fraud attack*) is a deception deliberately practiced in order to secure unfair or unlawful gain [36], or an intentional perversion of truth in order to induce another to part with something of value or to surrender a legal right [25]. *Fraud attacks* are a special class of security attacks, defined by attacker's goal of gaining a financial or other tangible reward.

Entities (human or artificial) that commit fraud are known as *fraudsters*. They can be classified into two categories: impersonators and swindlers [11]. An *impersonator* is an *illegitimate* user who steals resources from victims, for instance by taking over their accounts. A *swindler* is a *legitimate* user who intentionally benefits from the system or other users by deception. For instance, she obtains legitimate accounts and uses the services without intention to pay the bills.

We extend the notion of a fraudster to cover not only humans but also artificial entities (such as programs, *computers*, etc.).

1.2 Related Work on Fraud

Fraud detection systems are widely used in telecommunication, online transactions, computer and network security, and insurance. A taxonomy of computer fraud is

Manuscript received on November 17, 2007, accepted on March 15, 2009.

L. Lilien is with the Department of Computer Science, Western Michigan University, Kalamazoo, MI, USA. A. Bhargava is with the School of Electrical and Computer Engineering; and B. Bhargava is with the Department of Computer Sciences, Purdue University, West Lafayette, IN, USA

presented by Vasius [37]. Many research efforts address fraud detection in telecommunications (e.g., [2, 13, 35]). Examples of other common fraud research address identity fraud [19], and fraud in e-commerce and e-business [29, 24, 38]. Data mining [1, 12, 16], machine learning [14, 27], and statistical methods [13, 35] have been developed to detect fraud.

Different fraud detection approaches include an adaptive rule-based detection framework for superimposition fraud [16], a neural network technique [13], solving the incompatible schema problem caused by sharing a distributed fraud-detection database [15].

Due to the skewed distribution of fraud, one challenge in fraud detection is a very high *false alarm* rate. Also several other criteria are used to evaluate performance of fraud detection engines. *Receiver Operating Characteristics (ROC)* [17, 25, 35] is a common one. Rosset *et al.* [31] use accuracy and fraud coverage as criteria.

A cost-based metric can be used in commercial fraud detection systems [33].

Security techniques such as cryptography, separation of duty policies, biometric authentication, and state transition analysis can prevent fraud [21]. Also general techniques are applicable, e.g., use of adaptability [9, 16, 23].

Impersonators can be forestalled by utilizing *cryptographic techniques* that provide strong protection to users' authentication information.

The idea of *separation of duty* [20] may be applied to reduce the impact of a fraudster. An example of this idea is the *Chinese wall policy* arising in the commercial sector of consulting services is an example [11].

For more on related work cf. Reference [6].

2 Towards Trusted Computing

In this and the following sections, we discuss one approach for reaching the goal of trusted computing. We proceed from analysis of fraud vulnerabilities, via fraud threats, and via providing fraud attacks to establishing an environment of mutual trust among human and technological entities involved in computing, thus providing a *trustworthy* computing environment.

We concentrate on the integrity aspects of security (pretty much ignoring the other two elements of classically defined security: availability and confidentiality) [18, 30].

3 Fundamental Principles

We start by presenting the fundamental principles of our approach. We propose that the following principles be employed for creating integrity-improving algorithms or procedures, methods and methodologies:

1. Apply research principles from the areas of reliability, integrity and fault tolerance [5, 8, 7].
2. Apply non-deterministic system behavior, with the ideas of uncertainty, unpredict-ability, and deception. E.g., use honeypots [26, 32].
3. Employ adaptability, diversity, flexibility [4, 9], so a system can adaptively decide which of the alternative executables it runs.
4. Use appropriate military principles for defensive or even offensive actions [22]. Among the candidate principles are: defense in depth; use of spies, deception, surprise; and counter-attacking (when allowed by law).
5. Apply biologically-inspired principles, such as autonomic computing [3], including self-configuring, self-healing, self-optimizing, and self-protecting.

4 Analyzing Fraud Vulnerabilities

4.1 Fraud and Trust

Fraud involves *abuse of trust* [10, 39]. A fraudster strives to present himself as a trustworthy individual and friend. In a clear way, the more trust one places in others, the more open or even careless one tends to become in relationships with them. This results in becoming much more vulnerable (with the possibility of experiencing more harm). Fortunately, vulnerabilities are not automatically exploited but create only a *potential* for fraud. Only misplaced trust may end up being exploited.

4.2 The Nature of Fraud Vulnerabilities

Vulnerabilities create an *opportunity* for fraud. E.g., if Bob learns that Alice has a new account but is not using it, he can break in and use it.

While stealing is a singular act, fraud might be a *series* of acts being repeated over and over again as long as the victim is *unaware* of these acts or of their perpetrator.

Summarizing, fraud has *opportunistic*, *serial* and *covert* nature. (The serial nature encourages to model fraud as *nested transactions* [28].)

4.3 Categorization of Fraud Vulnerabilities

Fraud vulnerabilities can lead to recurring attacks and increasing damage in the enterprise. They appear during the design, implementation, and deployment stages of a system's lifetime.

Taxonomies of fraud vulnerabilities for each stage facilitate gaining a deep insight into the realm of vulnerabilities. The taxonomy can be used by a formal model to reason about characteristics of vulnerabilities, and to eliminate some of them. Good fraud vulnerability taxonomy can guide system design and implementation, and facilitate avoiding and tolerating fraud attacks in deployed systems. In particular, a proper vulnerability categorization underlies analytical and experimental methods for assessing the system-wide impact of vulnerabilities.

Causes of fraud vulnerabilities include the following categories of causes: (a) location-related, (b) rush-related, (c) mobility-related, and (d) software-related (cf. [6] for more details).

4.4 Elimination of Fraud Vulnerabilities

Elimination of fraud vulnerabilities is an effective way of dealing with fraud. However, it is *not* necessarily the most efficient way. Hence, we may need to accept even some known fraud vulnerabilities and instead try to ameliorate them by analyzing potential threats that could exploit them. This is the topic of the next section.

5 ANALYZING FRAUD THREATS

Efficient methods to assess threats resulting from system flaws allow for analysis of all kinds of threats, as well as evaluation of their potential impacts. The assessment precedes building a robust threat avoidance/tolerance mechanism for handling fraud threats due to known or even unknown fraud vulnerabilities.

Also in this case proper fraud threat categorization facilitates the analysis.

5.1 Fraud Threats as Subcategory of Security Threats

Fraud threats can be viewed as a special subcategory of general security or privacy threats that have a few salient features. First, a fraudster is usually *well known* to the victim whose trust he first gains and then abuses. In contrast, a "generic" attacker is usually not as closely related to a victimized system as a fraudster to his victim. This is true even in case of perpetrators of "generic" insider attacks.

Second, the goal of a fraudster is to *benefit himself* and not to hurt the system. Unlike a malicious attacker, fraudster does not plan to disrupt or destroy the system that he exploits. This is similar to the relationship between a clever parasite and the organism on which it feeds. Negative consequences of fraudster's actions are only unavoidable side effects.

Third, once a suspicion of a fraud threat arises and becomes known to the fraudster who is still not identified, the fraudster usually moves to prey on another victim. In contrast, when a suspicion of a malicious attack threat arises and becomes known to the attacker, the threat does not vanish: the attacker will probably just use different venues of attack without changing its object.

5.2 Fraud Attacks as Subcategory of Security Attacks

Similarly, materialized fraud threats, that is fraud attacks, can be viewed as a special subcategory of general security or privacy attacks, distinguished by a few salient features. First, fraud is often committed *over a period of time*, if not detected, even over months or years. Many other security attacks occur over a relatively short time span even though attack preparation can also be surprisingly patient and slow.

Second, a successful fraud requires *keeping secrecy*. The longer it is kept the better for the fraudster, who can repeat or escalate his fraud. In contrast, an attacker often wants her exploits to be known to the general public (for fame, intimidation, etc.). Even if she does not want it, due to very visible impacts of her exploits, she often cannot conceal the attack effects.

5.3 Other Fraud Threat Features and Types

Fraud often occurs as a malicious *opportunistic reaction*, triggered by a careless action. In other words, quite often people do not plan fraud but commit it when tempted by a vulnerability revealed to them. When the vulnerabilities become known to recidivist fraudsters, the risk of becoming a fraud victim rises sharply.

Fraud escalation seems to be a natural phenomenon. A fraudster, maybe reacting to an opportunity, defrauds five dollars. Next time, encouraged by the ease of the act, he might defraud a much more significant amount.

Gang fraud threats can be especially damaging since they involve a group of collaborating fraudsters (cf. [6] for more details).

There exist environments contributing to fraud threats, such as the ones with fuzzy assignment of responsibilities between participating entities, be they human or artificial (cf. [6]).

5.4 Need for Fraud Threat Assessment

Fraud threats resulting from system flaws should be assessed at each stage of system lifetime by efficient methods. A good taxonomy facilitates this analysis and further investigation of threats. It is important to notice that a comprehensive assessment of fraud threats may identify even unknown fraud vulnerabilities, not identified by fraud vulnerability analysis.

A threat analysis precedes developing robust threat avoidance/tolerance methods, mechanisms, and tools. Designers and implementers of computer systems need them at the development time stages of system life cycle. Similarly, system administrators and maintenance staff need them at the post-deployment time stages (after the system is deployed).

6 Fraud Threat Avoidance and Tolerance

Detection of fraud *vulnerabilities* and fraud *threats* is a prerequisite for or a component of fraud avoidance and tolerance. In addition, detection of fraud *attacks* might be a prerequisite for or a component of fraud tolerance (but not for fraud avoidance since preventing fraud attacks is assumed for such an approach).

6.1 Fraud Threat Avoidance

The problem of threat *avoidance* is most important at the development time for computer systems. The situation is exacerbated by the inherent fraudsters' advantage: once a typical system is deployed, its overall structure and its many functionalities are pretty much frozen (at least until the next major system release). Fraudsters have a lot of time to study and probe the system to discover its vulnerabilities, and then to exploit them. Unless the system has built-in capabilities for fault and intrusion tolerance, the only weapons in developers' hands are security patches and point releases fixing bugs (non-structural and of limited scope).

The designers should therefore be very careful what they let out of their door. Yet, too often designs of even critical systems consider threat avoidance to an insufficient degree, making them just too vulnerable to even relatively unsophisticated attacks.

The promising known fraud prevention techniques include cryptography, separation of duty policies, and state transition approaches.

6.2 Fraud Threat Tolerance

The goals for fraud threat *tolerance* are analogous. In addition to arming the system for threat tolerance at the development time, good algorithms and tools must be provided for post-deployment support. This includes the techniques analogous to the ones used for fault tolerance in the field of computer reliability.

6.3 Post-deployment Detection of Fraud Vulnerabilities and Threats

In addition to fraud vulnerability and threat analysis done at the system development time, we must ensure that deployed systems are sub-jected to detection *of fraud* vulnerabilities and threats also after deployment. This can be done by system testing, analogous to reliability testing.

Costs of testing should be weighed against the expected fraud losses and future benefits, since testing should enhance future system performance. Unfortunately, irrationally extensive fraud protection measures might be an irrational—albeit psychologically explainable—reaction by an entity just recently defrauded.

Exhaustive fraud vulnerability or threat testing might not be a practical approach, so sampling could be used (cf. [6] for more details).

6.4 Detection of Fraud Attacks

After system deployment, in addition to identifying fraud *vulnerabilities* or fraud *threats* (discussed above), fraud threat tolerance may require tests for detecting fraud *attacks*.

The two main approaches for fraud detection are: profile-based anomaly detection and rule-based fraud detection.

7 AN ARCHITECTURE FOR SWINDLER DETECTION

In the preceding sections we proposed an approach to analyzing fraud vulnerabilities, fraud threats and fraud attacks. In this and the next section, we present a system for swindler detection, and an analysis of its critical algorithm.

The major challenge for swindler detection is to react to a suspicious action or cooperation that may lead to a fraud. Three approaches were considered: (1) detecting an entity's activities that deviate from legitimate patterns; (2) constructing state transition graphs for existing fraud scenarios and detecting frauds similar to the known ones; and (3) discovering an entity's intention based on its past behaviors. The first two approaches are also used to detect frauds conducted by impersonators. The last one is applicable only for swindler detection. An architecture utilizing all three approaches, briefly described in the next section, was proposed in Reference [11].

7.1 Design Considerations and Top-level Architecture

The design of the architecture was based on the following assumptions:

- A deviation from the usual pattern of an entity behavior may indicate a fraud.
- A similarity between an entity's current activity and a known fraud scenario warns that the same fraud may be occurring again.
- Analysis of an entity's behaviors in a relatively long period may reveal entity's bad intentions that it tries to mask by blameless activities.

Our swindler detection architecture consists of four components:

1. Profile-based *anomaly detector* monitors current activities for suspicious actions based upon the established patterns of an entity's behavior. It outputs the *fraud confidence* indicator showing the probability of a fraud.
2. *State transition analysis* component builds for current activities a state transition graph that provides so called *state descriptions* when a current activity results in entering a danger state, which may lead to a fraud.
3. *Deceiving intention (DI) predictor* discovers deceiving intention of an entity based on entity's history and *satisfaction ratings,* that is, in contrast to the previous two components it investigates only entity's past behaviors. *DI-confidence* indicator is a measure that characterizes the belief that the target entity has a deceiving intention. It is a real number ranging over [0,1], with the higher values indicating stronger beliefs.
4. *Decision-making* component takes as its inputs the outputs of the three preceding components, namely, fraud confidence, state description, and DI-confidence. It assists system administrators in reaching fraud alarm decisions, based on the predefined policies.

7.2 Anomaly Detector Component

Profile-based anomaly detector monitors for a target entity's activities that deviate from established patterns. It consists of three major subcomponents:

1. The *rule generation and weighing* subcomponent applies data mining techniques to existing massive amounts of entity activity records. From this information, fraud rules are generated and assigned weights according to their frequency of occurrence. Both entity and behavior attributes are used in mining and weighing fraud rules.
2. The *user profiling* subcomponent produces the profiling information characterizing entities, such as age, location and financial status. It also captures an entity's behavior patterns, such as how often she buys or sells, the price preferences and product choices.

   There are two sets of profiling data, one for *current profiles* and the other for *historical profiles*. In order to reflect an entity's current behavior patterns, the current profile set is dynamically updated according to behaviors. As behavior data grows larger, the decay process is used to reduce the data volume.

   This subcomponent also involves rule selection for a specific entity, based on profiling results and rules. When combined with profiling information, a set of rules is selected as fraud indicators for monitoring a specific entity.
3. The *online detection* subcomponent retrieves the related fraud rules when an activity occurs. It may also need to retrieve the entity's current and historical behavior patterns. Each rule is checked and a weight for it is produced.

   If a behavior deviation reaches the defined threshold, the offending entity will be caught. A weight will be output according to the rules. The results are combined to determine fraud detection confidence.

7.3 State Transition Analysis Component

State transition analysis models fraud scenarios as series of states changing from an initial secure state to a final compromised state. The *initial sta-te* precedes actions that lead to a fraud. The *final state* is the resulting state of committing a fraud. There may be several *intermediate states* between them. Actions that cause transition from one state to another are called *signature actions*. They are the smallest actions required to transition closer to the final state. A fraud scenario will not be completed without them.

This model requires collecting fraud scenarios at the beginning and identifies the initial and final states. Then, the signature actions for that scenario are identified in the backward direction. A *fraud scenario* is represented as a state transi-tion graph of states and signature actions.

A *danger factor* is associated with each state. It is defined as the distance from the current state to the final state that indicates a fraud. If one sta-te leads to several final states, the minimum dis-tance is used. For each activity, state transition analysis checks the *potential next states.* If the maximum value of the danger factors associated with these potential next states exceeds a thre-shold, a warning is raised and a state description is sent to the decision-making component.

7.4 Deceiving Intention Predictor Component

The kernel of deceiving intention predictor is the *deceiving intention prediction (DIP) algorithm* [11] that views a belief in a deceiving intention as complementary to a trust belief. The trust belief for an entity is evaluated based on the satisfaction sequence $< S_1, S_2, \ldots, S_n>$. S_n is the most recent satisfaction rating, which contributes the α portion to the trust belief. The remaining portion, $(1 – \alpha)$, comes from the previous trust belief and is determined recursively.

For each entity, DIP maintains a pair of factors: the *construction factor* W_c and the *destruction factor* W_d. If integrating current satisfaction rating increases trust belief, then $\alpha = W_c$; otherwise $\alpha = W_d$. W_c and W_d are initialized by the system administrator. They satisfy the constraint $W_c < W_d$ so that the property of *easy-destruction-hard-construction* is assured. This property stems from the fact that more effort is needed to gain the same amount of trust than to lose it.

W_c and W_d are modified when a foul event occurs. A *foul event* is triggered when a satisfaction rating is lower than the established threshold. Upon a foul event, the target entity is put under supervision in the sense that entity's W_c is decreased and W_d is increased.

If the entity does not conduct any foul event during a *supervision period,* entity's W_c and W_d are restored to the initial values. Otherwise, entity's W_c and W_d are further decreased and increased, respectively. The supervision period associated with an entity increases each time when the entity is put under supervision, so that this punishment lasts longer the next time. In this way, an entity with a worse history is treated harsher. The *DI-confidence* is computed as *1 – current_trust_belief.*

7.5 Decision-making Component

The decision-making component takes fraud confidence, state description, and DI-confidence as its inputs. It passes warnings from state transition analysis to system administrators, and displays the description of a next potential state in a readable format. It computes the *expected fraud risk*, raising a fraud alarm when this risk ex-ceeds the corresponding fraud investigation cost.

8 EXPERIMENTAL STUDY OF DIP ALGORITHM FOR DECEIVING INTENTION PREDICTOR

8.1 Three Types of Deceiving Behaviors

Experimental evaluation of the DIP algorithm [11] investigates its performance for three types of deceiving behavior identified by us:

1. Behavior with *uncovered deceiving intentions,* where swindler's satisfaction ratings are stably low and vary in a small range over time.
2. Behavior with *trapping intentions,* where a swindler first exhibits intentionally blameless behavior to gain trustworthiness. This is only a preparation for a fraud, which follows.
3. Behavior with *illusive intentions,* where a swindler exhibits cycles of intentionally blameless behavior followed by intervals fraudulent actions. The periods of apparently blameless behavior are meant to cover the dishonest activities and to confuse the fraud detection mechanisms. This results in *cycles* of preparation and entrapment (in contrast to the previous case when one preparation interval precedes one entrapment period).

Representations of these behaviors are inputs to the DIP algorithm, which calculates for them the value of the DI-confidence indicator (which is a real number ranging over [0,1] with the higher values indicating stronger beliefs).

8.2 Results

The experimental results can be summarized as follows [11]:

1. For a *swindler with uncovered deceiving intentions:* Since the possibility that the swindler conducts foul events is high, he is under supervision most of the time. The construction and destruction factors become close to 0 and 1, respectively, due to repetitive punishment for foul events. The trust values at 0.1 are close to the minimum. The DI-confidence is around 0.9.
2. For a *swindler with trapping intentions:* The DIP algorithm responds quickly to the sharp drop of the satisfaction rating when swindler ends the preparation phase and enters the entrapment phase. Increasing DI-confidence from 0.22 to 0.76 takes only 6 ratings.
3. For a s*windler with illusive intentions*: DI-confidence increases when the swindler ends the preparation phase of a cycle and starts an entrapment. Similarly, DI-confidence decreases when the swindler ends the entrapment phase and enters the "honest" phase of a cycle. Still, the DIP algorithm is able to catch this smart swindler because her DI-confidence eventually increases to about 0.9. This demonstrates that an effort to cover periods of fraudulent activities with periods of good behaviors is less and less effective with each repetition of the preparation-entrapment cycle.

9 Conclusion

We discussed relationships between fraud vulnerabilities, fraud threats, fraud attacks, and trust. We analyzed fraud, identifying the salient features of fraud attacks, which make them a separate category within the realm of security attacks. We proposed architecture for swindler detection consisting of four components: profile-based anomaly detector, state transition analysis, deceiving intention predictor, and decision-making component. The deceiving intention predictor (DIP) algorithm is the critical element of the architecture. The DIP algorithm predicts fraudulent intentions. Its experimental evaluation shows that it is effective in uncovering different fraud strategies, from naive to smart ones.

Acknowledgements

This research was supported in part by the NSF grants IIS-0242840 and IIS-0209059.

References

[1] D. W. Abbott, I. P. Matkovsky, and J. F. Elder, "An evaluation of high-end data mining tools for fraud detection," *Proc. 1998 IEEE Intl. Conf. on Systems, Man, and Cybernetics*, 1998.

[2] R. Alves, P. Ferreira, O. Belo and J. Lopes, "Discovering telecom fraud situations through mining anomalous behavior patterns," *Proc. DMBA Workshop, 12th ACM SIGKDD*, 2006.

[3] Autonomic Computing, *IBM Systems J*. (special issue), Vol. 42(1), 2003.

[4] C. Bain, D. B. Faatz, A. Fayad, D. Williams, "Diversity as a defense strategy in information systems. Does evidence from previous events support such an approach?" *Proc. IICIS 2001*.

[5] Anjali Bhargava and B. Bhargava, "Applying fault-tolerance principles to security research," *Proc. IEEE Symposium on Reliable Distributed Systems*, New Orleans, LA, Oct. 2001.

[6] B. Bhargava, "Vulnerabilities and Fraud in Computing Systems," *Proc. International Conf. on Internet, Processing, Systems, Interdisciplinaries* (*IPSI*), *VIP Scientific Forum*, Sv. Stefan, Serbia and Montenegro, Oct. 2003.

[7] B. Bhargava and L. Lilien, "A Review of Concurrency and Reliability Issues in Distributed Database Systems," pp. 1-84 in: *Concurrency Control and Reliability in Distributed Systems,* ed. B. Bhargava, Van Nostrand Reinhold, New York, New York, 1987.

[8] B. Bhargava and L. Lilien, "Expert Systems for Fault Tolerant Distributed Database Systems," pp. 41-182 in: *Essays in Computer Vision and Other Topics*, ed. J. Tou, Academia Sinica, Republic of China, 1990.

[9] B. Bhargava and J. Riedl, "A Formal Model for Adaptable Systems for Transaction Processing," *IEEE Transactions on Knowledge and Data Engineering,* Vol. 4, No. 1, 1989.

[10] B. Bhargava and Y. Zhong, "Authorization Based on Evidence and Trust," *Proc 4th Intl. Conf. on Data Warehousing and Knowledge Discovery (DaWaK-2002),* Aix-en-Provence, France, Sep. 2002.

[11] B. Bhargava, Y. Zhong, and Y. Lu, "Fraud Formalization and Detection," *Proc. 5th Intl. Conf. on Data Warehousing and Knowledge Discovery (DaWaK-2003),* Prague, Czechia, Sep. 2003.

[12] R. Brause, T. Langsdorf, and M. Hepp, "Neural data mining for credit card fraud detection," *Proc. 11th IEEE Intl. Conf. on Tools with Artificial Intelligence*, 1999.

[13] P. Burge and J. Shawe-Taylor, "Detecting cellular fraud using adaptive prototypes," *Proc. AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Mgmt*, 1997.

[14] P. Burge, J. Shawe-Taylor, Y. Moreau, B. Preneel, C. Stoermann, and C. Cooke, "Fraud detection and management in mobile telecommunications networks," *Proc. European Conf. on Security and Detection*, 1997.

[15] P. Chan, W. Fan, A. Prodromidis, and S. Stolfo, "Distributed data mining in credit card fraud detection," *IEEE Intelligent Systems*, 1999.

[16] T. Fawcett and F. Provost, "Adaptive Fraud Detection," *Data Mining and Knowledge Discovery,* Vol. 1(3), 1997.

[17] J. Hollmén, J. and V. Tresp, "Call-based fraud detection in mobile communication networks using a hierarchical regime-switching model," *Advances in Neural Information Processing Systems 11: Proc. 1998 Conf.*, 1998.

[18] Information Technology Security Evaluation Criteria (IT-SEC): Provisional Harmonized Criteria, Dec. 1993.

[19] R. Jamieson, D. Winchester, G. Stephens, S. Smith, "Developing a Conceptual Framework for Identity Fraud Profiling," *Proc. European Conf. on Information Systems*, Galway, Ireland, 2008.

[20] E. Jonsson, L. Strömberg, and S. Lindskog, "On the Functional Relation between Security and Dependability Impairments," *Proc. Workshop on New Security Paradigm*s, Sep. 1999.

[21] E. Kotsakis *et al.*, "Advancing Security & Anti-fraud by Means of Info-mobility and Navigation Technologies (SAINT). System Preliminary Design," European Commission, June 2001.

[22] R. R. Leonhard, *The Principles of War for the Information Age,* Presidio Press, Novato, CA, 1998.

[23] L. Lilien and Anu Bhargava, "From Vulnerabilities to Trust: A Road to Trusted Computing," *Proc. International Conf. on Internet, Processing, Systems, Interdisciplinaries* (*IPSI*), *VIP Scientific Forum*, Sv. Stefan, Serbia and Montenegro, October 2003.

[24] I. MacInnes, D. Musgrave, and J. Laska, “Electronic Commerce Fraud: Towards an Understanding of the Phenomenon,” *Proc. 38th Hawaii Intl. Conf. on System Sciences*, 2005.

[25] *Merriam-Webster's Collegiate Dictionary*, *Eleventh Edition*, 2003.

[26] D. B. Moran, “Trapping and Tracking Hackers: Collective security for survival in the Internet Age,” *Proc. Third Information Survivability Workshop,* Boston, MA, Oct. 2000.

[27] Y. Moreau, H. Verrelst and J. Vandewalle, “Detection of mobile phone fraud using supervised neural networks: a first prototype,” *Proc. Intl. Conf. on Artificial Neural Networks*, 1997.

[28] J. E. B. Moss, “Nested transactions: an approach to reliable distributed computing,” Ph.D. Thesis, MIT, 1981. Available as Technical Report MIT/LCS/TR-260.

[29] S. Pandit, D.H. Chau, S. Wang, and C. Faloutsos, “NetProbe: A Fast and Scalable System for Fraud Detection in Online Auction Networks,” *Proc. 16th Intl. Conf. on World Wide Web (WWW'07)*, Banff, Alberta, Canada, May 2007.

[30] C. P. Pfleeger and S.L. Pfleeger, *Security in Computing. Fourth Edition*, Prentice Hall PTR, 2007.

[31] S. Rosset, U. Murad, E. Neumann, Y. Idan, and G. Pinkas. “Discovery of fraud rules for telecommunications - challenges and solutions,” *Proc. 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 1999.

[32] L. Spitzner, “Honey pots: Definition and value of honey pots,” <http://www.enteract.com/~lspitz/honeypot.html>.

[33] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan, “Cost-based modeling for fraud and intrusion detection: results from the JAM project,” *Proc. DARPA Information Survivability Conf. and Exposition*, 2000.

[34] G. Stoneburner, A. Goguen, and A. Feringa, “Risk Management Guide for Information Technology Systems,” NIST Special Publication 800-30, Washington, DC, 2001.

[35] M. Taniguchi, M. Haft, J. Hollmén, and V. Tresp, “Fraud detection in communications networks using neural and probabilistic methods,” *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, 1998.

[36] *The American Heritage Dictionary of the English Language*, Fourth Edition, Houghton Mifflin, 2000.

[37] L. Vasiu and I. Vasiu, “Dissecting Computer Fraud: From Definitional Issues to a Taxonomy,” *Proc. 37th Hawaii Intl. Conf. on System Sciences*, 2004.

[38] L. Vasiu, “A Conceptual Framework of E-Fraud Control in an Integrated Supply Chain,” *Proc. European Conf. on Information Systems*, Turku, Finland, 2004.

[39] Y. Zhong, Y. Lu, and B. Bhargava, “Dynamic Trust Production Based on Interaction Sequence,” Technical Report CSD-TR 03-006, Department of Computer Sciences, Purdue University, Mar. 2003.

Prism-Spectrometer as Demultiplexer for WDM over POF

Lutz, Daniela; Haupt, Matthias and Fischer, H.P., Ulrich

Abstract—*Polymer Optical Fibres (POFs) show clear advantages compared to copper and glass fibres (GOFs). In essence, POFs are inexpensive, space-saving and not susceptible to electromagnetic interference. Thus, the usage of POFs has become a reasonable alternative in short distance data communication. Today, POFs are used in a wide number of applications due to these specific advantages. These applications include automotive communication systems and In-House-Networking. The currently used transmission technology via POF is based only on one channel (or rather on one wavelength), making the usable bandwidth the limiting factor of this technology. One potential solution to expand the usable bandwidth of POF-based systems is wavelength division multiplexing (WDM). Because of the attenuation behaviour of POF, the only transmission window is situated in the visible spectrum. The solution proposed in this paper allows the transfer of several signals on different wavelengths through a single polymeric fibre. In order to separate the transmitted signals, special separators – called demultiplexers (DEMUX) – are utilized. These DEMUX are realized by employing the principle of the prism-spectrometer. In the set-up described in this paper, the light emitted by the polymeric fibre is collimated via an off-axis parabolic mirror. Then it is led to a dispersion prism and there divided into its monochromatic parts.*

Key words - *WDM, POF, demultiplexer*

1. Introduction

1.1 Advantages and Applications of POF

Polymer Optical Fibres offer essential advantages compared with GOF technology, copper cable and wireless communication. In comparison to GOFs, POFs show a greater mechanical flexibility making them uncomplicated in handling because of the smaller bend radius.

Also, polymer optical fibres can be stressed mechanically much stronger because of their geometrical dimensions. In comparison to copper cables, which are still standard in industry and technology, optical polymer fibres save more space and weight. They allow an easy connector assembly as well. Additionally, they cannot be influenced by electromagnetic fields [1-4]. The wireless data communication technology has two basic disadvantages compared to fiber technology. First of all, the electromagnetic fields can lead to interferences. Additionally, the radio technology is not secure from interference by third persons. As a result of the mentioned features, POFs offer an attractive alternative.

Nevertheless, POFs have only one transmission window, which is allocated in the visible spectrum of light. The attenuation of POF is too high for the remaining electromagnetic spectrum and therefore not acceptable for data transfer.

POFs are therefore best suited for the use in short distance data communication. Today, POFs are applied in in-house-networks and widely in the automotive industry [5]. It can thus be concluded, that POF technology is widely applicable.

1.2 WDM over POF

The transmission with standard POF is realised with only one wavelength [1-2]. The only possibility to increase the bandwidth is to raise the data rate. This reduces the signal-to-noise-ratio and can be changed therefore only in strong limitations. In this paper, an established technique for GOF technology is introduced and applied for POF. Though, it concerns the wavelength multiplex technology. There are different multiplex technologies available, e.g. time division multiplex (TDM) [6]. Recently WDM (wavelength division multiplex) and code division multiplex (CDM) have been used in GOF-infrared systems and in mobile RF communication [7-10]. The principle of WDM is transferred to the visible spectrum for POF communication, which means several wavelengths are used which are transmitted at the same time over one fibre. For more than 15 years, WDM widely expanded the overall transmission bit rate in GOF-long-range systems, because of its easy expandable system approach: adding one new source with different transmission wavelength in combination with a MUX/DEMUX-element expands the usable transmission rate directly by this source. Due to the attenuation of POF, the wavelengths from 400nm to 700nm [1] are used for the WDM, as shown in fig. 1.

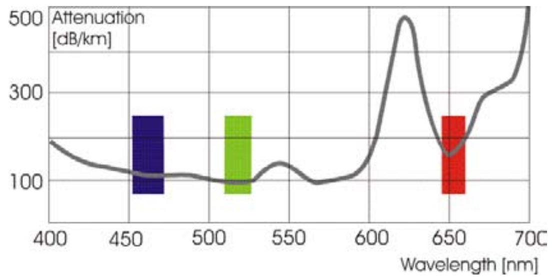

Fig. 1 Attenuation behaviour of a POF in the area of the visible spectrum

For the use of WDM technology in POF-systems, the same key elements are needed: the multiplexer and the demultiplexer. A complete redesign of these components is necessary, because a different spectrum is used. The light has to be combined by the multiplexer and split by means of a demultiplexer. This can be realised by means of different techniques. An optical phased array can be applied to change the phases of every different channel and therefore to divide the light in different channels [11]. Another possible technique are interference filters, which are well-known in the infrared range but also available for the visible spectrum [1, 12]. But key elements which are already available on the market use the infrared wavelength range or are cost-intensive solutions and thereby not suitable for mass market POF applications.

2. Basic concept of the multiplexer

For WDM over POF, a complete new-developed demultiplexer is required, as described in the previous chapter. The principle of the demultiplexer is schematically illustrated in fig. 2 and is already pending patent [13-15].

A standard SI-POF with a core diameter of 980µm and a cladding thickness of 10µm is used. The refractive index of the core is about the whole cross section equal 1.49 and the numerical aperture is 0.5.

The light emitted from the POF is focused and divided by the DEMUX. The POF is situated in the focus point of an off-axis parabolic mirror on which the light is reflected. To reduce the aberrations of the dispersion prism, the beams of light are leaded collimated. The prism separates the light in its different wavelengths. At the end of the DEMUX, a plano-convex lens focuses the separated wavelengths onto a detection layer.

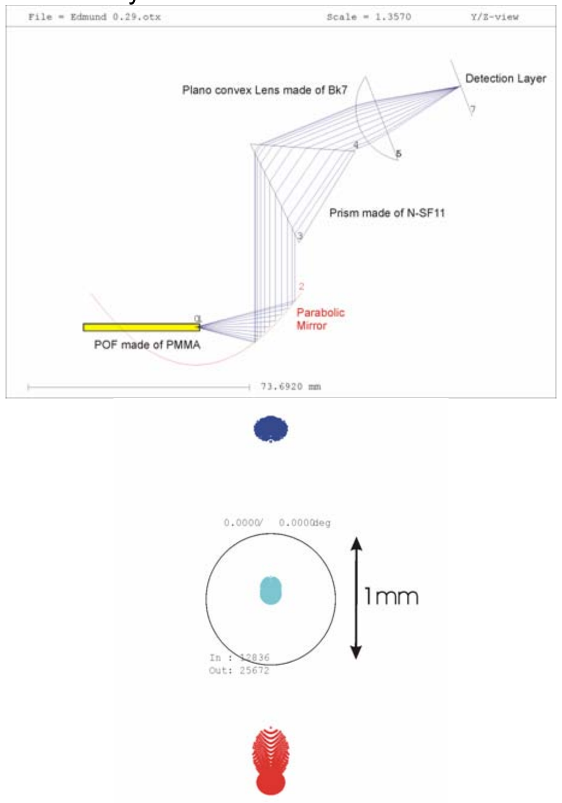

Fig. 2 Setup of the demultiplexer and the focus points at the detection layer in the simulation

The setup uses only three colours, blue (480nm), green (530nm) and red (660nm). This is not a limitation for possible future developments, but rather an experimental basis to run the various simulations described below.

3. Simulation and results of the simulation

In the following step, a simulation program is used to design a demultiplexer based on the general concept outlined above. For the current task, the software OpTaliX provides all needed functionalities [16]. This approach offers different advantages, it is easy to design, analyze and evaluate the simulated results. Also effective improvements of the configuration can be simulated fast. The simulated design was planned and developed with available standard components.

In the simulation of the DEMUX, all three colours are detectable on the image layer, as shown in fig. 2. The different focus points show a diameter lower than 1mm. The cross talk is below -30dB, because at the detection layer the different channels do not overlap each other.

4. Lab setup of the POF-DEMUX

4.1 Assembly

To verify the simulation results, the DEMUX is realised with commercially available standard components under lab conditions. The components are chosen, because they are inexpensive and the geometrical dimensions are close to the optimal design [17]. The complete construction is shown in fig. 3.

The various optical components effect the size of the focus point in the image layer differently. Especially the positioning of the POF and the off-axis parabolic mirror have to be assembled precisely, because a divergence of a few micrometers in any direction affects the focus size considerably at the detection layer. So for both of these optical elements, micrometer stages are used to guarantee a precise adjustment.

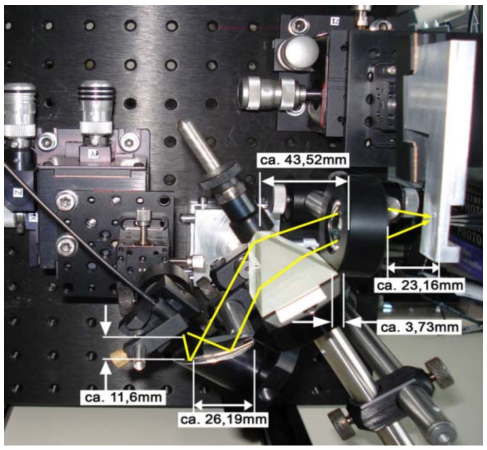

Fig. 3 Lab setup including the path of rays

A separation without viewable overlap of the three channels is achieved with an optimal alignment of the optical elements and a concurrent transfer of the wavelengths over POF used for this lab setup. The intensity dispersion is shown in fig. 4. It is demonstrated, that a clear separation of the different channels is achieved. However, the spectral width of the different channels avoids a clear determination.

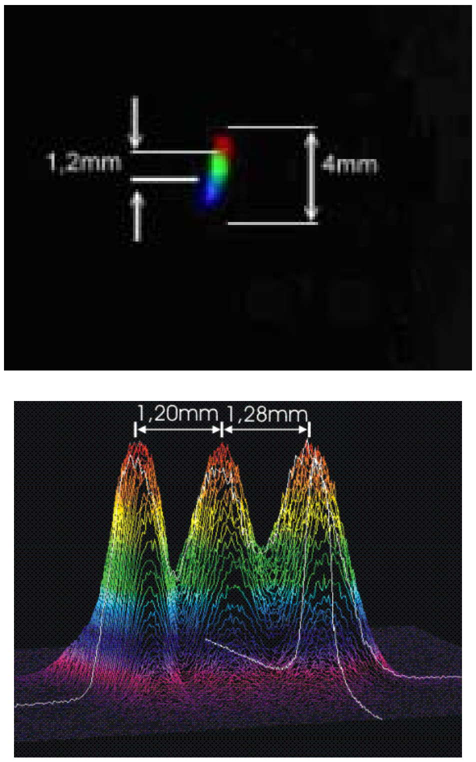

Fig. 4 Detection layer of the lab setup and measured intensity of the focus points

4.2 Characterisation of the DEMUX-Setup

In the first step, a characterisation of the lab setup is demonstrated for the attenuation behaviour and channel bandwidth. For the measurement of the general attenuation of the DEMUX, the colours blue, green and red will be transferred without multiplexing. For the channels blue and red media converters from DieMount [18] and for the green channel the OPTOTEACH teaching system of HarzOptics [19] are used.

Furthermore, a white light source (YOKOGAWA AQ 4305) is used for the estimation of the available bandwidth for every channel, see fig. 5.

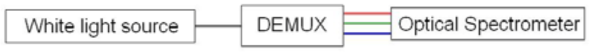

Fig. 5 Schematic setup with white light source

A connection is built up with all three WDM-channels (470nm, 520nm, 650nm), see fig 6.

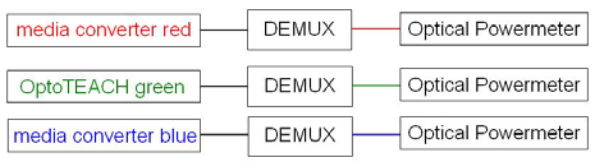

Fig. 6 Schematic setup for 3 channel attenuation measurement

The attenuation caused through the data transfer by the system is at blue 19,3dB, at green 12,1dB and at red 14dB. A coupler of the company HarzOptics for the combination of two channels causes an additional attenuation of about 5-6dB, see table 1. Fig. 7 demonstrates the existing cross talk of the different channels.

By means of a white light source, the spectrum is measured to verify the received signals and to estimate the channel bandwidth, as shown in fig. 7. The cross talk between the blue channel and the green channel is 4,6dB respectively 16,1dB. Also, the cross talk of the green channel to the red channel is rather high with 13,3dB. It can be recognised, that the spectral width of the red channel is larger in comparison to the blue channel. An explanation can be found in fig. 8, the steeper the curve falls the larger the gap between the colours.

Optical component \ Channel	λ = 470nm	λ = 520nm	λ = 650nm
Attenuation DEMUX (1 channel) [dB]	19,3	12,1	14
Attenuation DEMUX (with coupler & connector) [dB]	23,9	18,9	19,8

TABLE 1. ATTENUATION RESULTS OF THE DEMULTIPLEXER

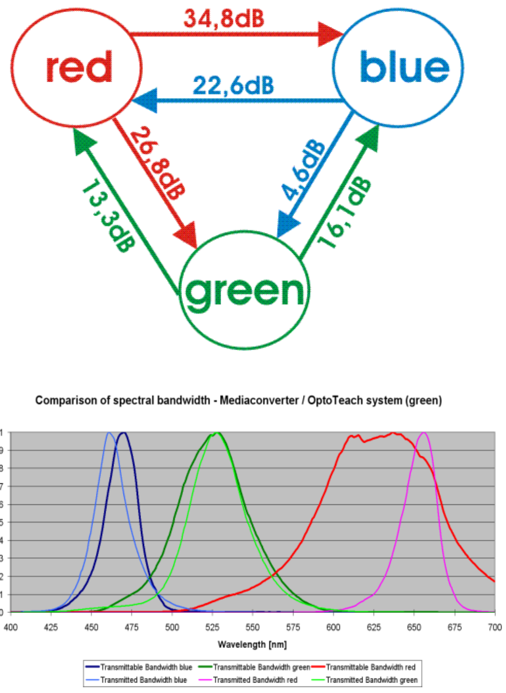

Fig. 7 Cross-talk of the 3 channels and comparison of the spectra

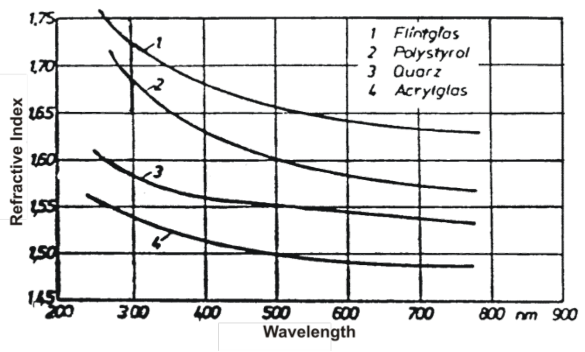

Fig. 8 Refractive index in dependence of wavelength

Therefore, the dispersion of the prism is non-linear. Furthermore the attenuation trait is also to view for short POF-lengths. This can be seen in the curve progression of the red channel. It shows an easy reduction of the signal power located for the wavelength area about 620nm, shown in fig. 1.

4.3 Data Transfer

The first Fast Ethernet data transfer was designed and tested with the colours red and blue, see fig. 9. An error-free transfer of data was realised. For the Ethernet data transfer the DEMUX behaves transparent. The functionality of the WDM-system is thus verified.

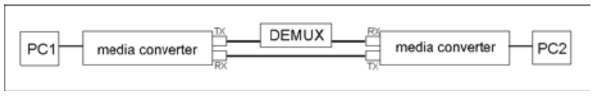

Fig. 9 Schematic setup Ethernet data link

However, because of the attenuation, which is still too high, the data transfer is possible with only one channel. For a data transfer via WDM, the transmitted signals are too weak considering their appearing attenuation. With the media converters used in combination with couplers, a combined signal transfer is currently not possible, but it may become possible with a more precise adjustment of the focus length.

4.4 Conclusion and Outlook

This paper demonstrates, that it is principally possible to design and use a demultiplexer for polymer optical fibres and also to separate the different channels.

The adjustment of the optical components used in the lab setup exhibits a very high attenuation of the transmission link.

At the moment, it is only possible to transmit signals with high optical power to receive and use the transmitted signals with sufficient optical power at the end of the transfer distance, because of the high appearing attenuation. The focused signals at the detection layer are separated clearly as such and are detectable in comparison to the simulated spot diagram.

We still believe, that WDM over POF has a good potential to substitute the classical transfer technology in many areas of the short distance communication. But it is necessary to manufacture special components for this setup to increase the gaps between the channels. With standard components, an online three channel data communication assembly was not possible.

To make the WDM over POF solution attractive for the mass market, a possible inexpensive production must be implemented with improved components. One possible solution would be the moulding technology, which is a low-cost production with high amount of pieces.

REFERENCES

[1] W. Daum, J. Krauser, P. E. Zamzow, O. Ziemann, „ POF Handbook: Optical Short Range Transmission Systems “, Springer-Verlag, 2008

[2] H. S. Nalwa (Ed.), “Polymer Optical Fibers”, American Scientific Publishers, California 2004Last name1, Fn1., Last name2, Fn2., “Paper title,” http://www.(URL), Day. Month. Year, pp. 15–64.

[3] Club des Fibres Optiques Plastiques (CFOP) France, “Plastic Optical Fibres – Practical Applications”, Edited by J. Marcou, John Wiley & Sons, Masson, 1997

[4] J. Brandrup, E. H. Immergut, E. A. Grulke, “Polymer Handbook” 4th Edition, Wiley-Interscience, 1999…

[5] http://www.mostcooperation.com/

[6] E. Voges, K. Petermann, „Optische Kommunikationstechnik“, Springer-Verlag, 2002

[7] R. T. Chen and G. F. Lipscomb, Eds, “WDM and Photonic Switching Devices for Network Applications”, Proceedings of SPIE, vol. 3949, 2000

[8] J. Colachino, “Mux/DeMux Optical Specifications and Measurements”, Lightchip Inc. white paper, Lightreading, July 2001Remember that each paper must have at least 10 references!

[9] A. H. Gnauck, A. R. Chraplyvy, R. W. Tkach, J. L. Zyskind, J. W. Sulhoff, A.J. Lucero, et. al., “One terabit/s transmission experiment”, in Proc. OFC’96, PD 20, San Jose, CA, 1996

[10] C. R. Batchellor, B. T. Debney, A. M. Thorley, T. J. B. Swanenburg, G. Heydt, F. Auracher,et. al., “A coherent multichannel demonstrator”, Electr. & Comm. Engineer. J., 235- 242 (1992)J. Brandrup, E. H. Immergut, E. A. Grulke, “Polymer Handbook” 4th Edition, Wiley-Interscience, 1999…

[11] U. H. P. Fischer, “Optoelectronic Packaging”, Vde-Verlag, 2002

[12] Fraunhofer Institut für Integrierte Schaltungen, “Optical multiplexer for short range communication” http://www.iis.fraunhofer.de/ec/oc/download/demux.pdf

[13] Multiplex-Sender für Polymerfaserübertragung und Verfahren zu dessen Herstellung, 10 2005 050 747.6 (Tx) 22.10.2005

[14] Demultiplex-Empfänger für Polymerfaserübertragung und Verfahren zu dessen Herstellung, 10 2005 050 739.5 (Rx), 22.10.2005

[15] Multiplex-Transceiver für Polymerfaserübertragung und Verfahren zu dessen Herstellung, 10 2006 009 365.8 (Trx)

[16] http://www.optenso.com

[17] U. H. P. Fischer, M. Haupt “WDM over POF – The inexpensive way to breakthrough the limitation of bandwidth of standard POF communication” SPIE Photonics West 2007, San Jose, USA

[18] http://www.diemount.de

[19] http://www.harzoptics.de

Software Testing Methods and Techniques

Jovanović, Irena

Abstract—*In this paper main testing methods and techniques are shortly described. General classification is outlined: two testing methods – black box testing and white box testing, and their frequently used techniques:*

- *Black Box techniques: Equivalent Partitioning, Boundary Value Analysis, Cause-Effect Graphing Techniques, and Comparison Testing;*
- *White Box techniques: Basis Path Testing, Loop Testing, and Control Structure Testing.*

Also, the classification of the IEEE Computer Society is illustrated.

1. Definition and the Goal of Testing

PROCESS of creating a program consists of the following phases (see [8]): 1. defining a problem; 2. designing a program; 3. building a program; 4. analyzing performances of a program, and 5. final arranging of a product. According to this classification, software testing is a component of the third phase, and means checking if a program for specified inputs gives correctly and expected results.

Software testing (Figure 1) is an important component of software quality assurance, and many software organizations are spending up to 40% of their resources on testing. For life-critical software (e.g., flight control) testing can be highly expensive. Because of that, many studies about ***risk analysis*** have been made. This term means *the probability that a software project will experience undesirable events, such as schedule delays, cost overruns, or outright cancellation* (see [9]), and more about this in [10].

There are a many definitions of ***software testing***, but one can shortly define that as:[1]

A process of executing a program with the goal of finding errors (see [3]). So, testing means that one inspects behavior of a program on a finite set of ***test cases*** (*a set of inputs, execution preconditions, and expected outcomes developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement*, see [11]) for which valued inputs always exist. In practice, the whole set of test cases is considered as infinite, therefore theoretically there are too many test cases even for the simplest programs. In this case, testing could require months and months to execute. So, how to select the most proper set of test cases? In practice, various techniques are used for that, and some of them are correlated with risk analysis, while others with test engineering expertise.

Testing is an activity performed for evaluating software quality and for improving it. Hence, the goal of testing is systematical detection of different classes of errors (***error*** *can be defined as a human action that produces an incorrect result*, see [12]) in a minimum amount of time and with a minimum amount of effort. We distinguish (see [2]):

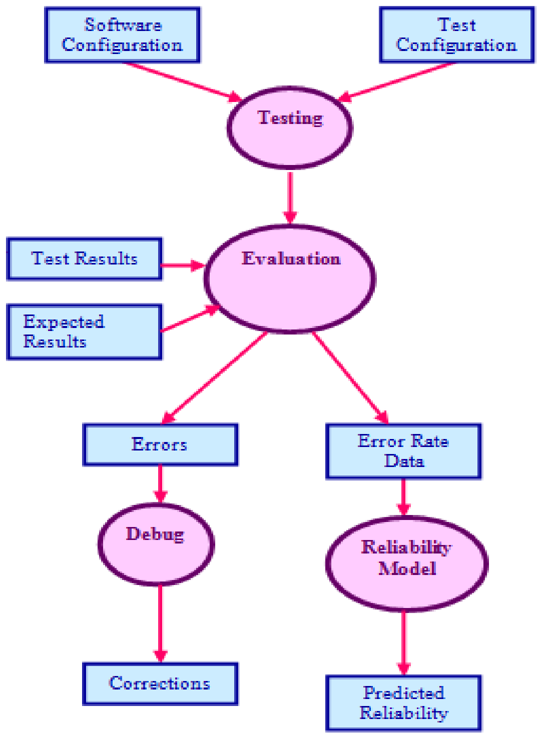

Figure 1: Test Information Flow

[1] Manuscript received May 26, 2008.
I. M. Jovanovic is with the DIV Inzenjering, d.o.o., Belgrade

- *Good test cases* - have a good chance of finding an yet undiscovered error; and
- *Successful test cases* - uncovers a new error.

Anyway, a *good test case* is one which:

- Has a high probability of finding an error;

Is not redundant;

- Should be “best of breed”;
- Should not be too simple or too complex.

2. Testing Methods

Test cases are developed using various test techniques to achieve more effective testing. By this, software completeness is provided and conditions of testing which get the greatest probability of finding errors are chosen. So, testers do not guess which test cases to chose, and test techniques enable them to design testing conditions in a systematic way. Also, if one combines all sorts of existing test techniques, one will obtain better results rather if one uses just one test technique.

Software can be tested in two ways, in another words, one can distinguish two different methods:

1. Black box testing, and
2. White box testing.

White box testing is highly effective in detecting and resolving problems, because bugs (***bug*** *or* ***fault*** *is a manifestation of an error in a software*, see [12]) can often be found before they cause trouble. We can shortly define this method as *testing software with the knowledge of the internal structure and coding inside the program* (see [13]). White box testing is also called white box analysis, clear box testing or clear box analysis. It is a strategy for software ***debugging*** (*it is the process of locating and fixing bugs in computer program code or the engineering of a hardware device*, see [14]) in which the tester has excellent knowledge of how the program components interact. This method can be used for Web services applications, and is rarely practical for debugging in large systems and networks (see [14]). Besides, in [15], white box testing is considered as a ***security testing*** (*the process to determine that an information system protects data and maintains functionality as intended,* see [6]) method that can be used to validate whether code implementation follows intended design, to validate implemented security functionality, and to uncover exploitable vulnerabilities (see [15]).

Black box testing *is testing software based on output requirements and without any knowledge of the internal structure or coding in the program* (see [16]). In another words, a black box is any device whose workings are not understood by or accessible to its user. For example, in telecommunications, it is a resistor connected to a phone line that makes it impossible for the telephone company’s equipment to detect when a call has been answered. In data mining, a black box is an algorithm that doesn’t provide an explanation of how it works. In film–making, a black box is a dedicated hardware device: equipment that is specifically used for a particular function, but in the financial world, it is a computerized trading system that doesn’t make its rules easily available.

In recent years, the third testing method has been also considered – ***gray box testing***. *It is defined as testing software while already having some knowledge of its underlying code or logic* (see [17]). It is based on the internal data structures and algorithms for designing the test cases more than black box testing but less than white box testing. This method is important when conducting integration testing between two modules of code written by two different developers, where only interfaces are exposed for test. Also, this method can include reverse engineering to determine boundary values. Gray box testing is non-intrusive and unbiased because it doesn’t require that the tester have access to the source code.

The main characteristics and comparison between white box testing and black box testing are follows.

2.1. Black Box Testing Versus White Box Testing

Black Box Testing:

- Performing the tests which exercise all functional requirements of a program;
- Finding the following errors:
  1. Incorrect or missing functions;
  2. Interface errors;
  3. Errors in data structures or external database access;
  4. Performance errors;

5. Initialization and termination errors.

- Advantages of this method:
  - ✓ The number of test cases are reduced to achieve reasonable testing;
  - ✓ The test cases can show presence or absence of classes of errors.

White Box Testing:

- Considering the internal logical arrangement of software;
- The test cases exercise certain sets of conditions and loops;
- Advantages of this method:
  - ✓ All independent paths in a module will be exercised at least once;
  - ✓ All logical decisions will be exercised;
  - ✓ All loops at their boundaries will be executed;
  - ✓ Internal data structures will be exercised to maintain their validity.

3. GENERAL CLASSIFICATION OF TEST TECHNIQUES

In this paper, the most important test techniques are shortly described, as it is shown in Figure 2.

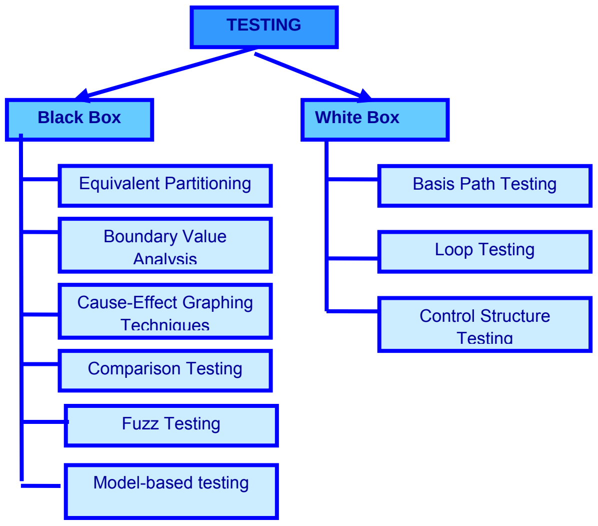

Figure 2: General Classification of Test Techniques

3.1. Equivalence Partitioning

Summary: *equivalence class*

This technique divides the input domain of a program onto equivalence classes.

Equivalence classes – set of valid or invalid states for input conditions, and can be defined in the following way:

1. An input condition specifies a range → one valid and two invalid equivalence classes are defined;
2. An input condition needs a specific value → one valid and two invalid equivalence classes are defined;
3. An input condition specifies a member of a set→ one valid and one invalid equivalence class are defined;
4. An input condition is Boolea~~n~~ one valid and one invalid equivalence class are defined.

Well, using this technique, one can get test cases which identify the classes of errors.

3.2. Boundary Value Analysis

Summary: *complement equivalence partitioning*

This technique is like the technique Equivalence Partitioning, except that for creating the test cases beside input domain use output domain.

One can form the test cases in the following way:

1. An input condition specifies a range bounded by values a and→ btest cases should be made with values just above and just below a and b, respectively;
2. An input condition specifies various values → test cases should be produced to exercise the minimum and maximum numbers;
3. Rules 1 and 2 apply to output conditions;

If internal program data structures have prescribed boundaries, produce test cases to exercise that data structure at its boundary.

3.3. Cause-Effect Graphing Techniques

Summary: *translate*

One uses this technique when one wants to translate a policy or procedure specified in a natural language into software's language.

This technique means:

Input conditions and actions are listed for a module $\Rightarrow$ an identifier is allocated for each one of them $\Rightarrow$ cause-effect graph is created $\Rightarrow$ this graph is changed into a decision table $\Rightarrow$ the rules of this table are modified to test cases.

3.4. Comparison Testing

Summary: *independent versions of an application*

In situations when reliability of software is critical, redundant software is produced. In that case one uses this technique.

This technique means:

Software engineering teams produce independent versions of an application→ each version can be tested with the same test data→ so the same output can be ensured.

Residual black box test techniques are executing on the separate versions.

3.5. Fuzz Testing

Summary: *random input*

Fuzz testing is often called fuzzing, robustness testing or negative testing. It is developed by Barton Miller at the University of Wisconsin in 1989. This technique feeds random input to application. The main characteristic of fuzz testing, according to the [26] are:

- the input is random;
- the reliability criteria: if the application crashes or hangs, the test is failed;
- fuzz testing can be automated to a high degree.

A tool called fuzz tester which indicates causes of founded vulnerability, works best for problems that can cause a program to crash such as buffer overflow, cross-site scripting, denial of service attacks, format bug and SQL injection. Fuzzing is less effective for spyware, some viruses, worms, Trojans, and keyloggers. However, fuzzers are most effective when are used together with extensive black box testing techniques.

3.6. Model-based testing

Model-based testing is automatic generation of efficient test procedures/vectors using models of system requirements and specified functionality (see [27]).

In this method, test cases are derived in whole or in part from a model that describes some aspects of the system under test. These test cases are known as the abstract test suite, and for their selection different techniques have been used:

- generation by theorem proving;
- generation by constraint logic programming;
- generation by model checking;
- generation by symbolic execution;
- generation by using an event-flow model;
- generation by using an Markov chains model.

Model-based testing has a lot of benefits (according [28]):

- forces detailed understanding of the system behavior;
- early bug detection;
- test suite grows with the product;
- manage the model instead of the cases;
- can generate endless tests;
- resistant to pesticide paradox;
- find crashing and non-crashing bugs;
- automation is cheaper and more effective;
- one implementation per model, then all cases free;
- gain automated exploratory testing;
- testers can address bigger test issues.

3.7. Basis Path Testing

Summary: *basis set, independent path, flow graph, cyclomatic complexity, graph matrix, link weight*

If one uses this technique, one can evaluate logical complexity of procedural design. After that, one can employ this measure for description basic set of execution paths.

For obtaining the basis set and for

presentation control flow in the program, one uses ***flow graphs*** (Figure 3 and Figure 4). Main components of that graphs are:

- *Node* – it represents one or more procedural statements. Node which contains a condition is called *predicate node*.
- *Edges* between nodes – represent flow of control. Each node must be bounded by at least one edge, even if it does not contain any useful information.

Region – an area bounded by nodes and edges.

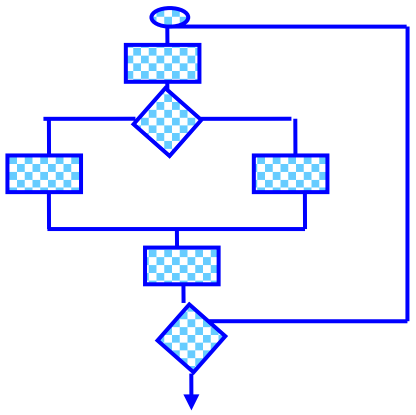

Figure 3: Flow Graph

Cyclomatic Complexity is software metric. The value evaluated for cyclomatic complexity defines the number of independent paths in the basis set of a program.

Independent path is any path through a program that introduces at least one new set of processing statements.

For the given graph G, cyclomatic complexity V(G) is equal to:

1. The number of regions in the flow graph;
2. **V(G) = E - N + 2**, where E is the number of edges, and N is the number of nodes;

V(G) = P + 1, where P is the number of predicate nodes.

So, the core of this technique is: one draws the flow graph according to the design or code like the basis ⇒ one determines its cyclomatic complexity; cyclomatic complexity can be determined without a flow graph→ in that case one computes the number of conditional statements in the code ⇒ after that, one determines a basis set of the linearly independent paths; the predicate nodes are useful when necessary paths must be determined ⇒ at the end, one prepares test cases by which each path in the basis set will be executed. Each test case will be executed and compared to the expected results.

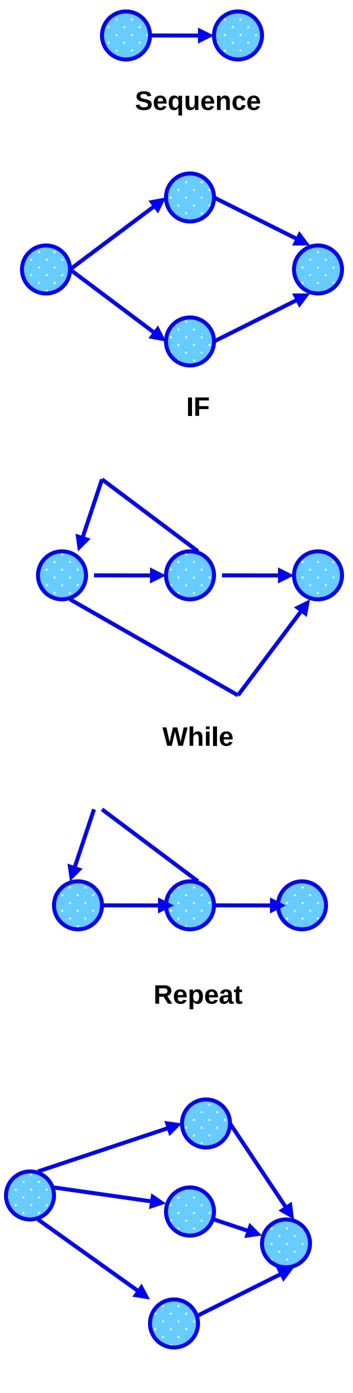

Figure 4: Different Versions of Flow Graphs

Example1. (cyclomatic complexity)

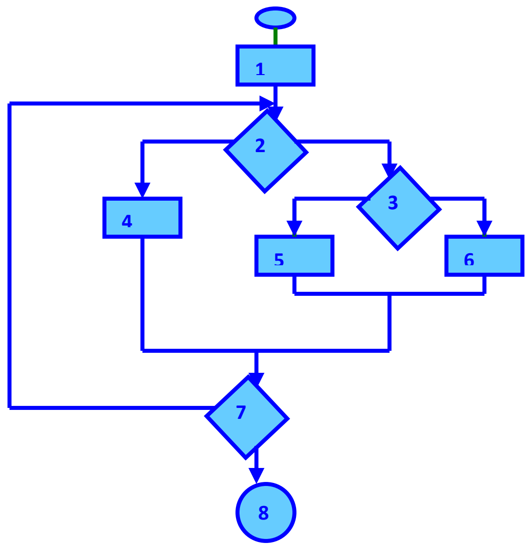

Figure 5: Graph for Example 1.

The cyclomatic complexity for the upper graph (figure 5) is:

- o V(G) = the number of predicate nodes +1 = 3+1 =4, or
- o V(G)= the number of simple decisions +1 = 4.

Well, V(G) = 4, so there are four independent paths:

The path 1: 1, 2, 3, 6, 7, 8

The path 2: 1, 2, 3, 5, 7, 8;

The path 3: 1, 2, 4, 7, 8;

The path 4: 1,2,4,7,2,4,…,7,8.

Now, test cases should be designed to exercise these paths.

Example2. (cyclomatic complexity)

Cyclomatic complexity for graph which is represented in Figure 6 is:

V(G) = E – N + 2 = 17 – 13 + 2 = 6.

So, the basis set of independent paths is:

1-2-10-11-13;

1-2-10-12-13;

1-2-3-10-11-13;

1-2-3-4-5-8-9-2;

1-2-3-4-5-6-8-9-2;

1-2-3-4-5-6-7-8-9-2.

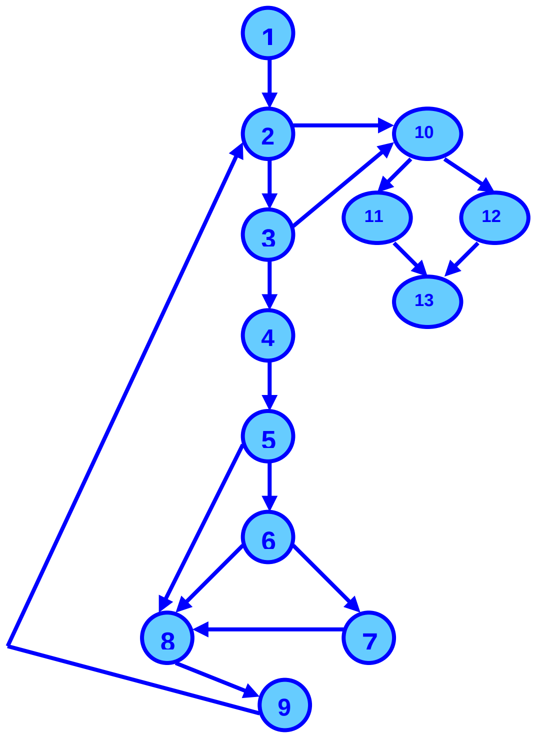

Figure 6: Graph for Example 2.

Example3.

Here are presented corresponding graph matrix and connection matrix for graph which is depicted in Figure 7.

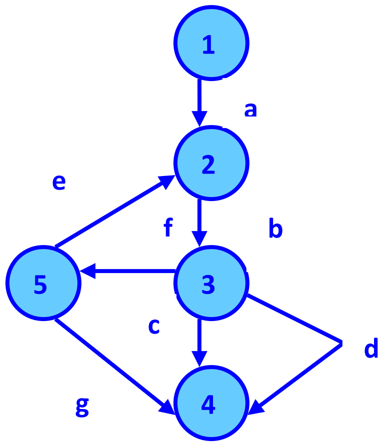

Figure 7: Graph for Example 3.

Table 1: Graph Matrix

node	1	2	3	4	5
1		a			
2			b		
3				d, c	f
4					
5		e		g	

Table 2: Connection Matrix

node	1	2	3	4	5	connections
1		1				1-1=0
2			1			1-1=0
3				1,1	1	3-1=2
4						0
5		1		1		2-1=1

Cyclomatic complexity is 2+1=3

3.8. Loop Testing

There are four types of loops:

1. Simple loops;
2. Concatenated loops;
3. Nested loops;
4. Unstructured loops.

3.8.1. Simple Loops

It is possible to execute the following tests:

- Skip the loop entirely;
- Only one pass through the loop;
- Two passes through the loop;
- m passes through the loop where m<n;
- n-1, n, n+1 passes through the loop, where n is the maximum number of allowable passes through the loop.

A typical simple loop is depicted in Figure 8.

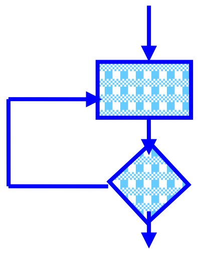

Figure 8: Simple Loop

3.8.2. Nested Loops

If one uses this type of loops, it can be possible that the number of probable tests increases as the level of nesting grows. So, one can have an impractical number of tests. To correct this, it is recommended to use the following approach:

- Start at the innermost loop, and set all other loops to minimum values;
- Conduct simple loop tests for the innermost loop and holding the outer loop at their minimum iteration parameter value;
- Work outward, performing tests for the next loop;
- Continue until all loops have been tested.

A typical nested loop is depicted in Figure 9.

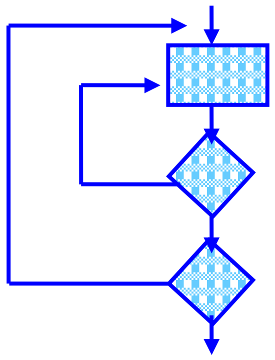

Figure 9: Nested Loop

3.8.3. Concatenated Loops

These loops are tested using simple loop tests if each loop is independent from the other. In contrary, nested loops tests are used.

A typical concatenated loop is presented in Figure 10.

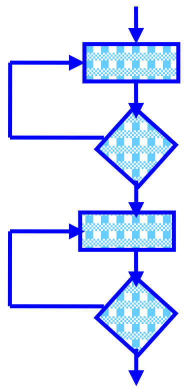

Figure 10: Concatenated Loop

3.8.4. Unstructured Loops

This type of loop should be redesigned.

A typical unstructured loop is depicted in Figure 11.

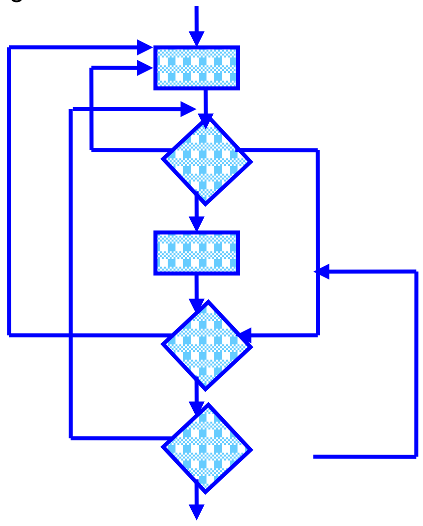

Figure 11: Unstructured Loop

3.9. Control Structure Testing

Two main components of classification of Control Structure Testing (Figure 12) are described below.

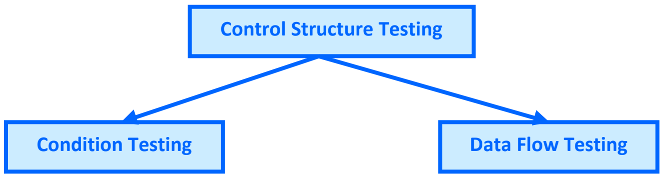

Figure 12: Classification of Control Structure Testing

3.9.1. Condition Testing

By this technique, each logical condition in a program is tested.

A ***relational expression*** takes a form:

$E_1 < \text{relational - operator} > E_2$,

Where E1 and E2 are arithmetic expressions, and relational operator is one of the following: <, =, $\neq$, $\leq$, >, or $\geq$.

A ***simple condition*** is a Boolean variable or a relational expression, possibly with one NOT operator.

A ***compound condition*** is made up of two or more simple conditions, Boolean operators, and parentheses.

This technique determines not only errors in the conditions of a program but also errors in the whole program.

3.9.2. Data Flow Testing

By this technique, one can choose test paths of a program based on the locations of definitions and uses of variables in a program.

Unique ***statement number*** is allocated for each statement in a program. For statement with S as its statement number, one can define:

DEF(S) = {X| statement S contains a definition of X}

USE(S) = {X| statement S contains a use of X}.

The definition of a variable X at statement S is live at statement S' if there exists a path from statement S to S' which does not contain any condition of X.

A definition-use chain (or DU chain) of variable X is of the type [X, S, S'] where S and S' are statement numbers, X is in DEF(S), USE(S'), and the definition of X in statement S is live at statement S'.

One basic strategy of this technique is that each DU chain be covered at least once.

4. Test Techniques According to the Project of the IEEE Computer Society, 2004.

The IEEE Computer Society is established to promote the advancements of theory and practice in the field of software engineering.

This Society completed *IEEE Standard 730* for ***software quality assurance*** *(it is any systematic process of checking to see whether a product or service being developed is meeting specified requirements*, see [18]) in the year 1979. This was the first standard of this Society. The purpose of *IEEE Standard 730* was to provide uniform, minimum acceptable requirements for preparation and content of software quality assurance plans. Another, new standards are developed in later years.

These standards are meaningful not only for promotion software requirements, software design and software construction, but also for software testing, software maintenance, software configuration management and software engineering management.

So, for improving software testing and for decreasing risk on all fields, there is classification of test techniques according to this Society, which is listed below.

▪ *Based on the software engineer's intuition and experience:*

1. **Ad hoc testing** – Test cases are developed basing on the software engineer's skills, intuition, and experience with similar programs;
2. **Exploratory testing** – This testing is defined like simultaneous learning, which means that test are dynamically designed, executed, and modified.

▪ *Specification-based techniques:*

1. **Equivalence partitioning**;
2. **Boundary-value analysis**;
3. **Decision table** – Decision tables represent logical relationships between inputs and outputs (conditions and actions), so test cases represent every possible combination of inputs and outputs;
4. **Finite-state machine-based –** Test cases are developed to cover states and transitions on it;
5. **Testing from formal specifications** – The formal specifications (the specifications in a formal language) provide automatic derivation of functional test cases and a reference output for checking test results;
6. **Random testing** – Random points are picked within the input domain which must be known, so test cases are based on random.

▪ *Code-based techniques*:

1. **Control-flow-based criteria** – Determine how to test logical expressions (decisions) in computer programs (see [19]). Decisions are considered as logical functions of elementary logical predicates (conditions) and combinations of conditions' values are used as data for testing of decisions. The definition of every control-flow criteria includes a statement coverage requirement as a component part: every statement in the program has been executed at least once. Control–flow criteria are considered as program-based and useful for white-box testing. For control-flow criteria, the objects of investigation have been relatively simple: Random Coverage, Decision Coverage (*every decision in the program has taken all possible outcomes at least once*), Condition Coverage (*every condition in each decision has taken all possible outcomes at least once*), Decision/Condition Coverage (*every decision in the program has taken all possible outcomes at least once and every condition in each decision has taken all possible outcomes at least once*) , etc.
2. **Data flow-based criteria**
3. **Reference models for code-based testing** – This means that the control structure of a program is graphically represented using a flow graph.

▪ *Fault-based techniques:*

1. **Error guessing** – Test cases are developed by software engineers trying to find out the most frequently faults in a given program. The history of faults discovered in earlier projects and the software engineer's expertise are helpful in this situations;
2. **Mutation testing** - *A mutant* is a modified version of the program under test. It is differing from the program by a syntactic change. Every test case exercises both the original and all generated mutants. Test cases are generating until enough mutants have been killed or test cases are developed to kill surviving mutants.

▪ *Usage-based techniques*:

1. **Operational profile** – From the observed test results, someone can infer the future reliability of the software;
2. **Software Reliability Engineered Testing**.

▪ *Techniques based on the nature of the application:*

1. **Object-oriented testing** – By this test technique we can find where the element under test does not perform as specified. Besides, the goal of this technique is to select, structure and organize the tests to find the errors as early as possible (see [25]).
2. **Component-based testing** – Is based on the idea of creating test cases from highly reusable test components. *A test component* is a reusable and context-independent test unit, providing test services through its contract-based interfaces. More about this test technique on: http://www.componentbasedtesting.org/site/.
3. **Web-based testing –** Is a computer-based test delivered via the internet and written in the "language" of the internet, HTML and possibly enhanced by scripts. The test is located as a website on the tester's server where it can be accessed by the test-taker's computer, the client. The client's browser software (e.g. Netscape Navigator, MS Internet Explorer) displays the test, the test-taker completes it, and if so desired sends his/her answers back to the server, from which the tester can download and score them (see [20]).
4. **GUI testing –** Is the process of testing a product that uses a graphical user interface, to ensure it meets its written specifications (see [6]).
5. **Testing of concurrent programs**;
6. **Protocol conformance testing** – ***A protocol*** *describes the rules with which computer systems have to comply in their communication with other computer systems in distributed systems* (see [23]). ***Protocol conformance testing*** *is a way to check conformance of protocol implementations with their corresponding protocol standards, and an important technology to assure successful interconnection and interoperability between different manufacturers* (see [24]). Protocol conformance testing is mostly based on the standard ISO 9646: "Conformance Testing Methodology and Framework" [ISO 91]. However, this conventional method of standardization used for protocol conformance test, sometimes gives wrong test result because the test is based on static test sequences.
7. **Testing of real-time systems** – More than one third of typical project resources are spent on testing embedded and real-time systems. Real-time and embedded systems require that a special attention must be given to timing during testing. According to the [21], real-time testing is defined as *evaluation of a system (or its components) at its normal operating frequency, speed or timing.* But, it is actually a conformance testing, which goal is to check whether the behavior of the system under test is correct (conforming) to that of its specification (see [22]). Test cases can be generated offline or online. In the first case, the complete test scenarios and verdict are computed a-priori and before execution. The offline test generation is often based on a coverage criterion of the model, on a test purpose or a fault model. Online testing combines test generation and execution.
8. **Testing of safety-critical systems**.

- *Selecting and combining techniques*:

1. **Functional and structural**;
2. **Deterministic vs. random** - Test cases can be selected in a deterministic way or randomly drawn from some distribution of inputs, such as is in reliability testing.

5. Conclusion

Software testing is a component of **software quality control** (**SQC**). *SQC means control the quality of software engineering products, which is conducting using tests of the software system* (see [6]). These tests can be: *unit tests* (this testing checks each coded module for the presence of bugs), *integration tests* (interconnects sets of previously tested modules to ensure that the sets behave as well as they did as independently tested modules), or *system tests* (checks that the entire software system embedded in its actual hardware environment behaves according to the requirements document). SQC also includes formal check of individual parts of code, and the review of requirements documents.

SQC is different from **software quality assurance** (**SQA**), *which means control the software engineering processes and methods that are used to ensure quality* (see [6]). Control conduct by inspecting quality management system. One or more standards can be used for that. It is usually *ISO 9000*. SQA relates to the whole software development process, which includes the following events: software design, coding, source code control, code reviews, change management, configuration management, and release management.

Finally, *SQC is a control of products, and SQA is a control of processes*.

Eventual bugs and defects reduce application functionality, do not look vocational, and disturb company's reputation. Thence, radically testing is very important to conduct. At that way, the defects can be discovered and repaired. Even if customers are dissatisfied with a product, they will never recommend that product, so product's cost and its popularity at the market will decrease.

Besides, *customer testing* is also very important to conduct. Through this process one can find out if application's functions and characteristics correspond to customers, and what should be changed in application to accommodate it according to customer's requests.

Large losses can be avoided if timely testing and discovering bugs in initial phases of testing are conducting. Deficits are minor if the bugs are discovered by testing within the company, where developers can correct errors rather than if the bugs are discovered in the phase of customer testing, or when the application is started "live" in some other company or system for which the application is created. In that case, the losses can be enormous.

Therefore software testing is greatly important, and test techniques too, because they have the aim to improve and make easier this process.

There is considerable controversy between software testing writers and consultants about what is important in software testing and what constitutes responsible software testing.

So, some of the major controversies include:

- ***What constitutes responsible software testing?*** – Members of the "context-driven" school of testing believe that the "best practices" of software testing don't exist, but that testing is a collection of skills which enable testers to chose or improve test practices proper for each unique situation. Others suppose that this outlook directly contradicts standards such as *IEEE 829* test documentation standard, and organizations such as *Food and Drug Administration* who promote them.
- ***Agile vs. traditional*** – Agile testing is popular in commercial circles and military software providers. Some researchers think that testers should work under conditions of uncertainly and constant change, but others think that they should aim to at process "maturity".
- ***Exploratory vs. scripted*** – Some researchers believe that tests should be created at time when they are executed, but others believe that they should be designed beforehand.
- ***Manual vs. automated*** – Propagators of agile development recommend complete automation of all test cases. Others believe that test automation is pretty expensive.
- ***Software design vs. software implementation*** – The question is: Should testing be carried out only at the end or throughout the whole process?
- ***Who watches the watchmen*** – Any form of observation is an interaction, so the act of testing can affect an object of testing.

REFERENCES

[1] http://www.his.sunderland.ac.uk/~cs0mel/comm83wk5.doc, February 08, 2009.

[2] Stacey, D. A., "Software Testing Techniques"

[3] Guide to the Software Engineering Body of Knowledge, Swebok – A project of the IEEE Computer Society Professional Practices Committee, 2004.

[4] "Software Engineering: A Practitioner's Approach, 6/e; Chapter 14: Software Testing Techniques", R.S. Pressman & Associates, Inc., 2005.

[5] Myers, Glenford J., IBM Systems Research Institute, Lecturer in Computer Science, Polytechnic Institute of New York, "The Art of Software Testing", Copyright 1979. by John Wiley & Sons, Inc.

[6] Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/

[7] http://www2.umassd.edu/CISW3/coursepages/pages/CIS311/outline.html

[8] Parezanovic, Nedeljko, "Racunarstvo i informatika", Zavod za udzbenike i nastavna sredstva – Beograd, 1996.

[9] Wei–Tek, Tsai, "Risk – based testing", Arizona State University, Tempe, AZ 85287

[10] Redmill, Felix, "Theory and Practice of Risk-based Testing", Software Testing, Verification and Reliability, Vol. 15, No. 1, March 2005.

[11] IEEE, "IEEE Standard Glossary of Software Engineering Terminology" (IEEE Std 610.12-1990), Los Alamitos, CA: IEEE Computer Society Press, 1990.

[12] http://www.testingstandards.co.uk/living_glossary.htm#Testing, February 08, 2009.

[13] http://www.pcmag.com/encyclopedia_term/0,2542,t=white+box+testing&i=54432,00.asp, February 08, 2009.

[14] http://searchsoftwarequality.techtarget.com/sDefinition/0,,sid92_gci1242903,00.html, February 08, 2009.

[15] Janardhanudu, Girish, "White Box Testing", https://buildsecurityin.us-cert.gov/daisy/bsi/articles/best-practices/white-box/259-BSI.html, February 08, 2009.

[16] http://www.pcmag.com/encyclopedia_term/0,2542,t=black+box+testing&i=38733,00.asp, February 08, 2009.

[17] http://www.pcmag.com/encyclopedia_term/0,2542,t=gray+box+testing&i=57517,00.asp, February 08, 2009.

[18] http://searchsoftwarequality.techtarget.com/sDefinition/0,,sid92_gci816126,00.html, February 08, 2009.

[19] Vilkomir, A, Kapoor, K & Bowen, JP, "Tolerance of Control-flow testing Criteria", Proceedings 27th Annual International Computer Software and applications Conference, 3-6 November 2003, 182-187, or http://ro.uow.edu.au/infopapers/88

[20] http://www2.hawaii.edu/~roever/wbt.htm, February 08, 2009.

[21] http://www.businessdictionary.com/definition/real-time-testing.html, February, 2009.

[22] Mikucionis, Marius, Larsen, Kim, Nielsen, Brian, "Online On-the-Fly Testing of Real-time systems", http://www.brics.dk/RS/03/49/BRICS-RS-03-49.pdf, February, 2009.

[23] Tretmans, Jan, "An Overview of OSI Conformance Testing", http://www.cs.aau.dk/~kgl/TOV03/iso9646.pdf

[24] http://www.ifp.uiuc.edu/~hning2/protocol.htm, February 2009.

[25] http://it.toolbox.com/blogs/enterprise-solutions/better-object-oriented-testing-21288, February 2009.

[26] http://pages.cs.wisc.edu/~bart/fuzz/fuzz.html, February, 2009.

[27] http://www.goldpractices.com/practices/mbt/index.php, February, 2009.

[28] http://blogs.msdn.com/nihitk/pages/144664.aspx, February, 2009.

Reviewers by Countries

Argentina
Olsina, Luis; National University of La Pama
Ovando, Gabriela P.; Universidad Nacional de Rosario
Rossi, Gustavo; Universidad Nacional de La Plata
Australia
Abramov, Vyacheslav; Monash University
Begg, Rezaul; Victoria University
Bem, Derek; University of Western Sydney
Betts, Christopher; Pegacat Computing Pty. Ltd.
Buyya, Rajkumar; The University of Melbourne
Chapman, Judith; Australian University Limited
Chen, Yi-Ping Phoebe; Deakin University
Hammond, Mark; Flinders University
Henman, Paul; University of Queensland
Palmisano, Stephen; University of Wollongong
Ristic, Branko; Science and Technology Organisation
Sajjanhar, Atul; Deakin University
Sidhu, Amandeep; University of Technology, Sydney
Sudweeks, Fay; Murdoch University
Austria
Derntl, Michael; University of Vienna
Hug, Theo; University of Innsbruck
Loidl, Susanne; Johannes Kepler University Linz
Stockinger, Heinz; University of Vienna
Sutter, Matthias; University of Innsbruck
Walko, Zoltan
Brazil
Parracho, Annibal; Universidade Federal Fluminense
Traina, Agma; University of Sao Paulo
Traina, Caetano; University of Sao Paulo
Vicari, Rosa; Federal University of Rio Grande
Belgium
Huang, Ping; European Commission
Canada
Fung, Benjamin; Simon Fraser University
Grayson, Paul; York University
Gray, Bette; Alberta Education
Memmi, Daniel; UQAM
Neti, Sangeeta; University of Victoria
Nickull, Duane; Adobe Systems, Inc.
Ollivier-Gooch, Carl; The University of British Columbia
Paulin, Michele; Concordia University
Plaisent, Michel; University of Quebec
Reid, Keith; Ontario Ministry og Agriculture
Shewchenko, Nicholas; Biokinetics and Associates
Steffan, Gregory; University of Toronto
Vandenberghe, Christian; HEC Montreal
Croatia
Jagnjic, Zeljko; University of Osijek
Czech Republic
Kala, Zdenek; Brno University of Technology
Korab, Vojtech; Brno University of technology
Lhotska, Lenka; Czech Technical University
Cyprus
Kyriacou, Efthyvoulos; University of Cyprus
Denmark
Bang, Joergen; Aarhus University
Edwards, Kasper; Technical University Denmark
Orngreen, Rikke; Copenhagen Business School
Estonia
Kull, Katrin; Tallinn University of Technology
Reintam, Endla; Estonian Agricultural University
Finland
Lahdelma, Risto; University of Turku
Salminen, Pekka; University of Jyvaskyla
France
Bournez, Olivier
Cardey, Sylviane; University of Franche-Comte
Klinger, Evelyne; LTCI – ENST, Paris
Roche, Christophe; University of Savoie
Valette, Robert; LAAS - CNRS
Germany
Accorsi, Rafael; University of Freiburg
Glatzer, Wolfgang; Goethe-University
Gradmann, Stefan; Universitat Hamburg
Groll, Andre; University of Siegen
Klamma, Ralf; RWTH Aachen University
Wurtz, Rolf P.; Ruhr-Universitat Bochum
Greece
Katzourakis, Nikolaos; Technical University of Athens
Bouras, Christos J.; University of Patras and RACTI
Hungary
Nagy, Zoltan; Miklos Zrinyi National Defense University
India
Pareek, Deepak; Technology4Development
Scaria, Vinod; Institute of Integrative Biology
Shah, Mugdha; Mansukhlal Svayam
Ireland
Eisenberg, Jacob; University College Dublin
Israel
Feintuch, Uri; Hadassah-Hebrew University
Italy
Badia, Leonardo; IMT Institute for Advanced Studies
Berrittella, Maria; University of Palermo
Carpaneto, Enrico; Politecnico di Torino
Japan
Hattori, Yasunao; Shimane University
Livingston, Paisley; Linghan University
Srinivas, Hari; Global Development Research Center
Obayashi, Shigeru; Institute of Fluid Science, Tohoku University
Mexico
Morado, Raymundo; University of Mexico
Netherlands
Mills, Melinda C.; University of Groningen
Pires, Luís Ferreira; University of Twente
New Zealand
Anderson, Tim; Van Der Veer Institute

Welcome to IPSI BgD Conferences and Journals!

http://www.internetconferences.net

http://www.internetjournals.net

VIPSI-2009 AMALFI
Hotel Santa Caterina
Amalfi, Italy
March 5 to March 8, 2009

VIPSI-2009 BELGRADE
Belgrade, Serbia
April 2 to Aprli 5, 2009

VIPSI-2009 CROATIA - OPATIJA
Opatija/Abbazia
Villa Ariston
May 28 to May 31, 2009

VIPSI-2009 TIVAT
Tivat, Montenegro,
August 20 to August 23, 2009

VIPSI-2009 SLOVENIA
Lake Bled in Alps
Slovenia
September 24 to September 27, 2009

VIPSI-2009 VENICE
Venice, Italy
September 27 to September 30, 2009

VIPSI-2010 AMALFI
Hotel Santa Caterina
Amalfi, Italy
March 4 to March 7, 2010

VIPSI-2010 TIVAT
Tivat, Montenegro
June 3 to June 6, 2010

VIPSI-2010 VENICE
Venice, Italy
September 30 to October 3, 2010

VIPSI-2010 MILOCER
Milocer, Montenegro
December 30 to December 31, 2010

CIP – Katalogizacija u publikaciji
Narodna biblioteka Srbije, Beograd
ISSN 1820 – 4511 =
The IPSI BGD Transactions on Advanced Research
COBISS.SR - ID 119128844

ISSN 1820-4511
9 771820 451006