NSF Workshop Report

Electronic Design Automation Past, Present, and Future

July 8–9, 2009
Arlington, Virginia

Robert Brayton (UC Berkeley)
Jason Cong (UC Los Angeles)

NSF Workshop

Electronic Design Automation
Past, Present, and Future

July 8-9, 2009

Robert Brayton (UC Berkeley) and Jason Cong (UCLA)[1]

Executive Summary

Electronic Design Automation (EDA) has been an immensely successful field, helping to manage the exponential increase in our capability to implement integrated circuits that currently incorporate billions of transistors. At the same time, it fostered and used theories in computation and modeling, successfully combining theory and practice. EDA has completely transformed the way that electronic engineers design and manufacture integrated circuits. It was one of the earliest to engage in inter-disciplinary collaboration, where the computer scientists and engineers in EDA successfully collaborated with electrical engineers, physicists, chemists, theoretical computer scientists, applied mathematics and optimization experts, and application domain specialists.

This workshop was organized to 1) reflect on the success of EDA to see if its practice can influence other fields of computer science, and if its methodology can be applied to other application domains, and 2) to review the progress made under the National Design Initiative and evaluate what new directions and topics should be added to the Initiative.

This report contains an overview of the EDA area, its funding history, a discussion of some major challenges for the future, related emerging technologies and how EDA experience may help in developing these technologies, educational aspects and challenges, EDA's relation with CS theorists and how this collaboration can be resurrected, and finally in Section 8, a series of recommendations on what might be done to promote EDA and help with the serious challenges it faces in the future. The recommendations are divided into 1) promoting research, 2) supporting educational programs, and 3) encouraging enhanced collaboration with industry. An appendix provides some details of the workshop's organization, participants, and program of talks.

[1] brayton@eecs.berkeley.edu and cong@cs.ucla.edu

Background, Motivation, and Objectives

Electronic Design Automation (EDA) of very large-scale integrated (VLSI) circuits and systems is an important field in computer science and engineering. It has made a significant impact on the development of information technology—in particular, by supporting the successful scaling of Moore's Law in the past 40 years, which in turn created the high-performance and cost-efficient information technology infrastructure that has transformed our life and society. The success of the EDA field is inspiring for many reasons:

- It has successfully managed the exponential increase in design complexity—from the first microprocessor (Intel 4004) with 2,250 transistors to the latest multi-core processor with over a billion transistors.
- It is one of the first fields in computer science and engineering (CS&E) that has applied the concepts and techniques of computational modeling, computational thinking, and computational discovery to an application domain (electronic circuit design) and achieved remarkable success. It has completely transformed the way that electronic engineers design and manufacture integrated circuits. Every circuit being designed today starts with a computational model (specified in an executable programming language) at a high level of abstraction. It then goes through a sequence of synthesis and optimization transformations, followed by rigorous digital simulation and prototyping, as well as formal and semi-formal verification, before it is finally manufactured via advanced lithographical and chemical processes.
- The EDA field is one of the earliest to engage in inter-disciplinary collaboration, where the computer scientists and engineers in EDA successfully collaborated with the electrical engineers to derive various levels of circuit models; with physicists and chemists to derive manufacturing models; with theoretical computer scientists to conduct various kinds of complexity analysis; with applied mathematics and optimization experts to improvise highly scalable simulation and synthesis algorithms; and with application domain specialists to develop intellectual property (IP) libraries, etc.

For these reasons, the first objective of this workshop was to reflect on the success of EDA to see if its practice can influence other fields of computer science, and if its methodology can be applied to other application domains.

Currently, the EDA field is also facing serious challenges in its own domain. For example, non-recurring engineering (NRE) costs associated with VLSI circuit design are skyrocketing, with estimates of over $30M per ASIC design undertaken. The rapid increase in the number of transistors available on a single chip leads to system-on-chip integration, with complex interaction between software and hardware, digital and analog, etc. Moreover, the field of applications enabled by semiconductor technology is growing at a rapid rate, ranging from very high-performance microprocessors and signal processors, to a broad array of low-power portable devices, to micro sense/communicate/actuate networks of chips that are driven by very low per-unit cost and extremely low operating power. Designers must not only create chips that function properly in conventional digital and mixed-signal operation, but must also comprehend sensors that respond to signals from many physical domains, such as pressure, temperature, chemical, and optical. The design problems are further compounded by the introduction of many physical phenomena determining the performance of severely scaled semiconductor devices.

For example, power and performance characteristics of transistors are becoming statistical in nature. The probability of soft or permanent errors is much higher in the new generation of CMOS devices at 32nm or below, or in emerging non-CMOS devices. These present unprecedented challenges to the EDA field.

In order to address these challenges, the National Science Foundation (NSF) and Semiconductor Research Corporation (SRC) held a joint workshop in October 2006 to study the future directions of design automation. They made the recommendation that “research in design technology and tools be increased through a National Design Initiative which focuses on three research areas:

1. The development of a powerful new, physically aware, system design science and methodologies to increase design productivity by one order of magnitude over current techniques for integrated systems containing billions of elements.
2. The creation of robust optimization methodologies that provide guaranteed performance of integrated systems composed of devices whose characteristics are highly variable, that operate in several different physical domains, and that have uncertain reliability.
3. A revamped, systematic, and greatly improved interface to manufacturing to support the design of high-yield systems that obtain maximum utilization of the technology. Such an effort was deemed to be critical "to maintain U.S. leadership in design for integrated nano- and microsystems."

The second objective of this workshop was to review the progress made under the National Design Initiative and evaluate what new directions and topics should be added to the Initiative.

Discussions and Conclusions

1. EDA Defined and History

1.1 What is EDA?

The current Wikipedia definition of EDA is “the category of tools for designing and producing electronic systems ranging from printed circuit boards (PCBs) to integrated circuits. This is sometimes referred to as ECAD (electronic computer-aided design) or just CAD.” The workshop attendees felt that this definition was somewhat narrow, as it focuses mainly on the use of EDA technologies. We agreed that it was equally important to emphasize the following three aspects of EDA:

1. EDA consists of a collection of methodologies, algorithms and tools, which assist and automate the design, verification, and testing of electronic systems.
2. It embodies a general methodology that seeks to successively refine a high-level description to low-level detailed physical implementation for designs ranging from integrated circuits (including system-on-chips), to printed circuit boards (PCBs) and electronic systems.
3. It involves modeling, synthesis, and verification at every level of abstraction.

The second and third aspects of EDA in this definition can be applied easily to different application fields, other than the designing of electronic systems.

1.2 Evolution of EDA

Although the history of design automation algorithms such as Kernighan and Lin's bi-partitioning heuristics and Kenneth Hall's r-dimensional quadratic placement procedures predates the VLSI era, it was not until the early 1980s that the mystique surrounding VLSI chip design and fabrication was unveiled by Mead and Conway. This led to the cultivation of VLSI education in universities and to the flourishing of research in VLSI system design and electronic design automation (EDA) which, in turn, created automation tools for logic synthesis, layout generation, circuit simulation, design verification, reliability modeling, chip testing, debugging, and yield analysis. It is beyond the scope of this report to exhaustively chronicle the evolution of EDA research during the last three decades or to catalog how core disciplines of sciences such as physics, mathematics, computer science, statistics, and biology have shaped EDA research by establishing the foundation of the underlying theories for multi-criteria design optimization and complexity management.

One way of characterizing the advancement of VLSI automation tools is to demarcate them into different eras depending on their chief optimization criterion—namely, area, timing, power, reliability, and nano-scale issues. In the early 1980s, silicon area was at a premium, and the underlying theoretical foundation of layout tools was rooted in graph theory (min-cut, quadri-sectioning), convex optimization (geometric programming), stochastic methods (simulated annealing), evolutionary biology (genetic algorithms), physical laws (force-directed relaxation methods), circuit theory (energy minimization in resistive networks), and spectral methods

(eigenvalue-based partitioning). In the mid-1980s, stringent timing constraints motivated the development of theoretical underpinning for interconnect delay models by using linear algebra and matrix solvers (Krylov sub-space, Arnoldi, Pade, Pade via Lancoz algorithms), numerical analysis (differential quadrature method, finite difference), frequency domain solvers (method of characteristics, FDFD), and other theoretical techniques.

In the early 1990s, energy optimization requirements led to the adaptation and extension of finite element methods (FEM) and gridded techniques to solve large second-order PDE associated with the heat equation. The need for integration of full-chip thermal analysis with chip layout packages was realized by applying fast Fourier and discrete cosine transformation techniques to accelerate the multi-layer Green's function solvers over the quadruple integral spaces. In the late 1990s, higher reliability and quality assurance demands led to the development of technology-centric EDA tools that tackled lithographic limitations, process variations, and anomalous operating conditions. Formal design verification and validation methods were adopted in EDA tools to ensure correctness of the system design by using model checking, SAT solver, static analysis, and other mainstream CS theories. Multiscale/multilevel optimization techniques also received a lot of attention in the late 1990s, especially in physical design, to cope with the rapid increase in design complexity.

By the new millennium, the VLSI industry made a quantum leap by shattering the red brick barriers of 100nm and rapidly inching towards 32nm nodes. Consequently, new-generation EDA design tools for nanoscale CMOS chips started adopting computational quantum physics (density function theory, non-equilibrium Green's function, Wigner's function) to tackle nano-scale issues such as leakage currents through high-k gate dielectrics, as well as to develop multi-scale modeling for the beyond-Moore's-Law technologies such as carbon nano-tubes, graphene and tunneling FETs, quantum dots, single electron transistors, and molecular devices.

In parallel with the progress in modeling and optimization techniques used for DA problems, another significant development was the rise of design abstraction and the use of accurate estimation methods to cope with the exponential increase in design complexity. Mead and Conway led the way with their book that opened up VLSI to the masses with its simplified design rules. Accurate estimation methods, like model order reduction, delay modeling, high-level models, etc., have allowed the design flow to be divided into separate concerns. Design abstraction has risen from polygon drawing, to schematic entry, then to RTL (register-transfer level) specification using hardware description languages (HDLs), and most recently to behavior specifications—for example, in C, C++, SystemC, and Matlab code.

2. Funding for EDA Research

2.1 NSF's Funding of EDA

NSF's funding of academic EDA research has fluctuated between $8 million and $12 million in recent years. For the past one and a half decades, the bulk of this funding came from the Computer & Information Science & Engineering (CISE) Directorate under its core DA program, while additional awards were provided by the Electrical Communications and Cyber Engineering (ECCS) Division in the Engineering Directorate,[2] and to a lesser extent, by the interdisciplinary

[2] For reference, the funding for the CISE Directorate and the ECCS Division of the Engineering Directorates in FY2009 is $574M and $125M, respectively. The total funding for NSF in FY2009 is $5,183M.

Mathematics of Computer Sciences (MCS) program jointly run by CISE and the Division of Mathematical Sciences (DMS). Because the main mission of NSF is to foster groundbreaking discoveries, EDA awards are multifarious, variegated, and disparate, covering the broad spectrum of EDA research that ranges from high-level synthesis of mixed-signal systems to multi-scale simulation of emerging technologies. Special research initiatives like the Information Technology Research (ITR) and National Nanotechnology Initiative (NNI) have provided additional funding in the past to boost EDA research activities. It is fair to estimate that ITR added approximately $3 million/year to the design automation area during its five-year term from FY1999 to FY2004. The CISE directorate also funded approximately $2 million/year of design automation related research from its share of the National Nanotechnology Initiative up until about FY2005.

The CISE Emerging Models and Technologies (EMT) program (recently disbanded) also stimulated EDA research in disruptive nano, bio, and quantum technologies. Typical EDA projects were aimed at developing photonics clock distribution networks, synthesis of biomolecular computing systems, DNA computing using self-organized Wang tiles, multiscale modeling of nano and molecular systems, and synthesis of quantum logic circuits. It is not clear how the nearly $2 million that EMT spent on EDA research will be distributed in the re-clustered CCF division.

The Division of Mathematical Sciences (DMS) has often funded proposals that had mathematical underpinning for EDA research: e.g., Markovian modeling to study convergence of simulated annealing algorithms, convex programming and polyhedral methods, algorithmic semi-algebraic geometry and topology, nonlinear programming under density inequalities, etc. The ECCS division funded several multiscale modeling and simulation projects for emerging nanotechnologies and quantum engineering systems.

NSF has also promoted young educators through its CAREER awards program that funded about five new EDA projects per year from 1996 through 2001 from the design automation program. Design automation researchers also received some funding from the other programs mentioned above, including the EMT program (for nano related research), the CISE/CNS division (for embedded systems research), or from the Engineering Directorate (for analog and mixed signal research). The total number of CAREER awards in such areas is estimated to be about ten. Recently, this number has shown a somewhat declining trend, e.g., altogether five CAREER awards per year. However, it should be noted that during the period 2003-2008, the design automation CAREER awardees were winners of three prestigious Presidential Early Career Awards for Scientists and Engineers (PECASE). All of these funding opportunities contribute to the total $8 to $12M in funding that we estimate for NSF's DA program, which represents about 1.3% to 2% of the total available funding for the CISE Directorate and the ECCS Division of the Engineering Directorate—the two major funding sources for the DA program in NSF.

2.2 Foundational Aspects of EDA and Its Role in CISE

While EDA has been supported traditionally at NSF by CISE, there is a concern that some feel it should be supported by the Engineering Directorate instead. However, CISE is a directorate that supports computer engineering (CE) as well as computer science (CS), and EDA has long been one of the pillars of CE, together with architecture and software. It needs strong support

because, being both an engineering and theory discipline, it is more expensive due to the need for infrastructure support.

EDA research does rest at the interface between engineering and computer science because it is an application of computer science to solve engineering problems. While EDA research must be governed necessarily by engineering requirements and limitations, the **solutions to EDA research problems have long been achieved through the use of math/CS core theories**. For example, solutions to problems in physical design use graph theory; logic design uses Boolean algebra; simulation uses dynamical system theory; and verification uses numerous theories, including models of computation and programming language theory. These solutions also require the development of efficient algorithms for solving optimization problems, numerical analysis of nonlinear systems, satisfiability problems, stochastic analysis, etc. The development of such algorithms obviously requires researchers with strong backgrounds in computer and computational sciences. EDA is not just practice, but a tightly connected combination of theory and practice. This can be easily seen by an examination of almost any paper in the EDA area; it will show that EDA research must be backed up with theory and proofs. Such theoretical results needed to solve EDA problems may indeed solve open questions in math/CS theory through the application of an EDA perspective to the problem. For example, verification is solidly based on mathematics and CS theory; it is inherently multidisciplinary, involving researchers in verification, as well as domain experts who know about the kinds of systems to be verified (mixed-signal circuits, for example).

Moreover, researchers have recently demonstrated that this EDA perspective is valuable in solving problems in the sciences. For example, **researchers are applying EDA ideas to the development of tools for solving problems in physics, chemistry, systems biology, and synthetic biology**. For all of these reasons, we continue to believe that the correct home for EDA is within the CISE directorate at NSF.

2.3 Limitations of Industrial Funding as a Paradigm for EDA Support

Others may feel that since the EDA problems that are tackled are often of direct interest to industry, industry should then be the primary supporter. Indeed, industry has been a good supporter of EDA, primarily through the Semiconductor Research Corporation (SRC). However, these funds are quite limited now, and even in good economic times they have not been sufficient to cover all of EDA's needs. As described above, EDA research requires theoretical studies that may be too far removed from commercial application, thus making it difficult to obtain industrial sponsorship. Further, new research in applying EDA research to the sciences will lack support by industry. Industry will probably continue to be a good supporter of near-term research problems, but we will continue to need NSF support for riskier, longer-term research with less obvious commercial application.

Model checking, as well as model reduction, are examples where, without initial long-term funding (e.g., by NSF and DARPA), and even when industry was much more tolerant of "far out" research, they most likely would not have reached a level where industry would be interested. For example, model checking was developed from programming language theory over a period extending back to 1981. For the first ten years, it was supported exclusively by NSF. Now major EDA companies like Synopsys and Cadence are marketing model checkers.

2.4 SRC's Support of EDA

When the SRC began in 1982, the area of design sciences was an initial focus area, and EDA research was a major part. Such funding constituted about 25% of SRC's budget; the remainder was devoted to technology and manufacturing. Today's SRC organization has two design-related areas: computer-aided design and test, and integrated circuits and systems. These have absorbed an increasing share of member-directed contributions and now make up about 45% of SRC-funded research (the total EDA-related funding from SRC is estimated to be about $5M for 2009). Interest in design comes not only from CAD member companies, but from integrated device manufacturers, fabless, and fab-lite companies, and even from equipment manufacturers and foundries. There are several reasons for this:

1. It is getting harder to continue on the Moore's Law density-performance curve using technology scaling alone. Improvements in design techniques and design tools have provided a critical boost to this continued pace.
2. The International Technology Roadmap for Semiconductors (ITRS) has had an increasing emphasis on design in recent years, with much more material in chapters on design and design drivers. The current ITRS emphasizes "the importance of software as an integral part of semiconductor products" and "software design productivity as a key driver of overall design productivity."
3. Productivity improvements and improved time-to-market are key drivers for improvements in design automation.
4. In order to be competitive, a diverse set of applications (no longer just general-purpose microprocessors) drives industry to show differentiation in design (not just technology), in both hardware and software.

SRC continues to be an important funder of design automation research, supporting faculty in both computer science and electrical engineering departments and connecting them with their industrial counterparts. As well a formal verification, optimization techniques for "improving" circuits and systems are another area in which work firmly grounded in theory has been put into practice by industry. "Improving" is in quotes because this no longer means to make better according to a single measure—now multi-objective optimization is needed to meet area, speed, power, manufacturability, and a host of additional requirements. While progress in logic synthesis, placement and routing, and test necessarily required "short thin designers" to apply heuristics, and "seat of the pants" techniques to make progress and show comparability with manual efforts, *these approaches are being replaced by algorithmic processes with strong theoretical foundations.*

Design at the logic level increasingly requires knowledge of the physical properties of gates, wires, and even transistors. Pressure to reduce power, maintain performance, and still have working chips has required designers to look down to the manufacturing level as well as up to the system design. For example, power minimization relies on reducing leakage at the bottom, as well as employing system-level techniques such as multi-core architectures. All these areas require strong and increased connections with physics, as well as theory of computation and fundamental algorithms.

2.5 Partnerships Between NSF, SRC, DoD, and Industry

The foundational efforts described, which have enabled optimized logic and physical design, formal verification, design for test, and design for manufacturability, have not relied, and cannot rely, on industry support alone. SRC plays a key role in combining some of the resources of industry to focus on pre-competitive areas of mutual interest. One successful example is the Focus Center Research Program (FCRP), which is jointly funded by SRC member companies and DARPA. In 2008, there were five FCRP centers nationwide, with GSRC and C2S2 centers directly related to the design automation program. The EDA-related funding from these two centers was estimated to be from $4M to $5M in 2008.

A similar joint funding model is also being explored by NSF. Recently, NSF partnered with industry in areas such as analog/mixed signal design and multi-core design and architecture to support work important to industry and also important to part of the CISE mission.

The total available funding to the design automation research is estimated to be around $20M. It combines various programs from NSF, SRC, and FCRP, with about half provided by the NSF CISE program. However, other countries with competitive strength are making much more substantial investments in this area. We discuss this in the next section.

2.6 Comparison with Some International Funding Sources

During the workshop, there were several discussions about the levels of funding support for EDA overseas, which seemed to suggest that the U.S. is under-funding this area. We list some of the data provided.

2.6.1 Taiwan

Funding for EDA in Taiwan comes from direct government funding and from university/industry/government partnerships. One of the significant government funding programs related to EDA is the system-on-chip program, funded at $70M USD per year since 2001. Topics include integrated circuits, embedded software and design automation research. We estimate that a significant portion (say 50%) of this program is used to support EDA. Other government funding programs also include research programs related to EDA, such as the nanotechnology program, funded at $100M USD per year since 2003, and the telecom program, funded at $70M USD per year since 1998 (topics include wireless, broadband and Internet telecommunications). These government programs include support for interaction with non-Taiwan companies for longer-term stays for students and faculty..

In the university/industry/government program, the government is the primary enabler by encouraging industrial matching and participation. It is worth noting that government grants have only a 5% overhead and industry grants only a 20% overhead.

2.6.2 Europe

European funding for R&D happens on various levels, more or less uncoordinated. Although there is no program specifically dedicated to electronic design automation, many European countries have national programs to support R&D in areas like information and communication technology (ICT), nanotechnology, embedded systems, advanced computing, and software technology, which all contain EDA aspects. The European Community has budgeted for and

supported information technology, microelectronics and similar technologies since the early 1980s via so-called "Framework Programs."

Currently, the seventh such framework program (FP7) covers the period from 2007 to 2013. The program assigns $9.050 billion Euros to ICT and $3.475 billion Euros to nanosciences, nanotechnologies, materials and new production technologies, distributed over the entire period (http://cordis.europa.eu/fp7/budget_en.html). The program attempts to foster cooperation between countries. Participants must organize themselves in project groups composed from institutions from at least three different countries.

The EUREKA Consortium is another source, currently covering 38 countries. This supports European international projects with funding provided by national sources. The consortium raised funds on the order of $3.1 billion Euros in 2008. Key members are Austria, Belgium, Finland, France, Germany, Spain, Switzerland, the UK, and Israel (http://www.eureka.be/contacts/fundingList.do). For example, "Electronics, Microelectronics," supported 50 projects with a budget of $195.26 million Euros over ten years (http://www.eureka.be/thematic/showPrjThematic.do?area=1.1).

One of the latest initiatives of EUREKA is "CATRENE," meaning "Cluster for Application and Technology Research in Europe on NanoElectronics," which started in 2008 and is to finish in 2012. CATRENE has a $3 billion Euro budget. As of August 2009, 143 partners from 13 European countries were participating. Partners are both industry (e.g., Infineon, NXP, Philips, ST-Microelectronics, Airbus and Volkswagen, and CADENCE) and universities. Currently 13 projects are pending, and two are explicitly addressing EDA (http://www.medeaplus.org/web/projects/project_list.php).

Complementary to the CATRENE program is ENIAC-JRT, targeted at $3 billion Euros for the 2008 to 2013 time frame (http://www.medeaplus.org/web/downloads/clips_medea/Elektronik%28Jan09%29.pdf).
Its predecessor, MEDEA+, covered 2001 to 2008 with a budget of $4 billion Euros. It supported 70 projects (http://www.eureka.be/inaction/searchStrategic.do) of which 15 were EDA. 75% of the funding was from companies (http://www.medeaplus.org/web/downloads/medeaplus_brochure.pdf).

As another data point, Prof. Giovanni De Micheli (a former professor at Stanford and grantee of the NSF design automation program) provided us with data concerning a large effort in Switzerland that he directs. The *nano-tera.ch* program is a grant secured for the period 2008 to 2011, but it will happen with an 18-month delay. Ten projects were started in June 2009, with ten more coming in January 2010. Funding of $56M (with an additional commitment of $56M from each of six Swiss institutions) was inserted in the Swiss federal budget for 2008 to 2011. Money comes directly from the SER (Swiss Ministry of Research) to support collaborative research. Technically, six institutions formed a consortium to run *nano-tera*: EPFL (lead), ETHZ, CSEM, and Neuchatel, Basel and Lugano Universities. Technical information on this program can be found at http://www.nano-tera.ch/. Two points to be made are 1) this program is independent of EU/FP7; links with FP7 may be developed, but for now this is a Swiss program, and 2) actual research is just starting with the hiring of pre/post-doctoral students.

It is not clear what percentage of these funding opportunities is dedicated to the DA-related research. However, if we simply take 2% of the funding dedicated for nanosciences,

nanotechnologies, materials and new production technologies ($3,475 billion Euros) as part of the FP7 program (recall that 2% is the share of DA programs in the CISE Directorate and the ECCS Division of the Engineering Directorate in NSF), it amounts to $105M USD over five years, which is 2 to 3X larger than DA program funding from NSF.

These comparisons raise a lot of concerns about the level of U.S. investment in design automation, a key area of information technology. For example, it is alarming to see that the government funding for the DA program (estimated to be more than $35M/year) in Taiwan, with its GDP 2.7% of that of the U.S. (based on 2007 numbers), is 1.5 to 2X times higher than the total combined funding in the U.S. from NSF, SRC, and FCRP in design automation. (It is probably not a coincidence that Taiwan is doing extremely well in the IC industry; for example, it has the largest share of the worldwide IC foundry market.) This underscores the need and urgency for the U.S government and industry to partner in order to significantly strengthen the design automation support needed to keep our competitive advantages in this area.

3. Foundational Areas for Future DA Support

3.1 Transformative and Incremental Research

The National Science Board (NSB) and the U.S. Congress recently mandated NSF to look into the possibilities of funding transformative research. However, while accepting the need for and encouraging transformative technologies, NSF program directors and the research community in general recognize that historically much progress has been accomplished by consistently pushing important areas over long periods of time. For example, in EDA, model checking involved 28 years of research with long periods of solid (but sometimes incremental) progress; a similar trend occurred with model reduction. In funding research, it is important to keep in mind that many fundamental breakthroughs are achieved typically after years of steady progress; moreover, the exact nature and impact of a genuine breakthrough is often difficult to predict.

3.2 Verification and Model Checking

Verification is an essential and increasingly important part of EDA, and formal model checking has become a major contributor. Model checking arose from basic research in the programming languages and logics of the programs community. The early papers were published in POPL (Symposium on Principles in Programming Languages) and LICS (Logic in Computer Science). Only later was it realized that model checking could be used for verifying the correctness of sequential circuit designs. Indeed, model checking and other state-exploration techniques are probably easier for computer hardware than for computer software. Hardware companies have used model checking since the mid-1990s, while software companies like Microsoft have only recently begun to take a serious interest in formal verification.

Model checking is also a good example of a case where initial support by NSF paid off. The first papers were written in 1981 and 1982. Research was supported entirely by NSF until the early 1990s. When the power of symbolic model checking with OBDDs was realized, SRC also began to fund research in this area. Computer companies did not directly support research at universities on this topic, and EDA companies like Cadence and Synopsys did not develop commercial model checking tools until 2000, when bounded model checking with fast propositional SAT algorithms was proposed. (Incidentally, the current rash of fast SAT algorithms, now used for many purposes in EDA as well as other fields, were also developed

primarily by EDA researchers in universities, e.g., GRASP at the University of Michigan and Chaff at Princeton, where this recent research was motivated by EDA applications.)

Formal verification is inherently multidisciplinary; researchers must have a strong background in mathematics and theory. They need to know about mathematical logic, automata theory, OBDDs, SAT algorithms, decision procedures (like linear real arithmetic), programming language theory, models for concurrency, static program analysis and symbolic evaluation. To deal with analog and mixed-signal circuits operating in noisy environments, they will need to know differential equations, probability theory, and stochastic processes. A strong background in CS theory is necessary, but not sufficient. Deep knowledge of digital design and computer hardware is needed, often involving joint projects between CS and ECE groups or working closely with someone at a company interested in using the techniques. Since companies are often reluctant to release their designs to researchers in academia, summer internships for graduate students at companies are often very important.

Much important research still needs to be done in verification.

- Scalability will always be a problem because of the state explosion in highly concurrent systems.
- Embedded systems involving discrete and continuous behavior require a breakthrough.
- Little research has been done on establishing the correctness of analog and mixed-signal circuits.
- Assertions about non-functional properties such as power might be checked, but little research has been done here.
- Developing tools that regular engineers can use is a major challenge. Difficulties exist in writing specifications in a notation like temporal logic and in specifying the environment in which a device to be verified will operate. Counter-examples can be quite long, involving many state variables and inputs, and locating its cause can be challenging.

3.3 Synthesis Research

The development of logic and physical and high-level synthesis before the early 1980s opened up the use of higher levels of describing a design, such as the register transfer level (RTL), which is almost universally used today. High-level synthesis has had an almost equally long history, while physical synthesis started at the beginning of integrated circuit developments. Over the years, synthesis capabilities have been extended significantly, both in scale, speed and quality of results. Also, as newer methods of implementing logic, such as FPGAs, were developed, synthesis methods were adapted and devised for these new devices and their multiple forms. Methods for front-end estimation have played a critical role in allowing earlier design convergence, as well as yielding better results.

The acceptance of synthesis methods has been critically dependent on providing results that can compete or be better than manual methods. Also, synthesis techniques have been critical to the acceptance of new implementation methods. For example, several start-up ventures have invented implementation structures which, on paper and with some circuits implemented by hand, looked promising. However, the automatic synthesis of logic into these structures was vastly underestimated and resulted in failure of such ventures.

Although synthesis is frequently thought of as being mature and not needing further support, it is fundamental and still needs to be exploited and extended to larger ranges of scalability. Synthesis is also critically intertwined with verification; e.g., some synthesis methods are not used because they can't be verified formally, while the formal verification task can be simplified if it is given some synthesis history and equally strong methods for circuit reduction, such as induction. Although synthesis can be made scalable by partitioning the problem into smaller parts, this can limit the quality of results that can be obtained. Thus, extending the scalability of the underlying algorithms can improve quality and make resulting designs more competitive. Such development will aid both synthesis and verification since both areas share much of the fundamental algorithms, such as SAT, BDDs, representation and manipulation of logic, abstraction, speculation, interpolation, induction, etc.

Another development is the synthesis for different objectives. Early synthesis was aimed at decreasing area and delay. More recently, other objectives have come into play, such as power, noise, thermal, verifiability, manufacturability, variability and reliability. It is clear that more criteria will be seen as new technologies develop, and new models and optimization techniques will be needed to address such requirements.

Finally, synthesis is facing the challenges of raising the level of abstraction. For example, current SOCs can accommodate thousands of simple processor cores (such as Intel-386 equivalent processors). Instead of synthesizing a design into a network of standard cells (the current practice), one may consider synthesizing it into a network of processors, or a mixture of processors and standard cells. In this case, a C/C++/SystemC based behavior specification is a more natural starting point, as such specifications are more software friendly. New modeling, synthesis, and verification techniques are needed to support such new design methodology and a new level of design abstraction.

3.4 Programming Language Research

A good programming language may be crucial to achieving scalable design methodologies. It is generally agreed that the language design aspect of Verilog/VHDL leaves much to be desired. Unfortunately, the current EDA response seems to be kitchen-sink languages (SystemVerilog) that combine every aspect of every language (Verilog, e, Vera, ...) into the language definition. This complexity slows down the standardization process, and definition release of languages like SystemVerilog 3.1a, which took three major revisions just to get the "core" language right.

In contrast, the programming language/software world may have things to offer: clean core semantics, types and modularity, core languages together with well-designed libraries, static analysis, etc. A secondary benefit of a clean language design is that it may be easier to work with on tools within academia. At the moment, the investment to build the front-end for SystemVerilog is enormous, and hard to justify in an academic context.

Languages can be a great opportunity for industry/academic alliances. Language innovations are more easily started within academia with a clean slate, while a company has so much legacy code that it won't change, or it won't think about new languages easily. However, industry can provide challenging designs to judge the effectiveness of the language design or to motivate language constructs; and ideally, academics can adopt the best features. A good

demonstration of research ideas from programming language/verification communities influencing industrial design is “Bluespec.”

3.5 Analog and Mixed-Signal Design

Automation techniques for the growing portion of systems made up of analog circuits lag behind those in the digital realm. This is particularly critical given two quantitative facts: 1) 66% of today's integrated circuits are mixed-signal designs, i.e., they incorporate analog components; 2) numerous studies show that the analog portions of these designs are most frequently at fault when chips fail at first silicon. Moreover, scaling to the nano-scale level has led to the breakdown of clean digital abstractions to the point that at the level of physical implementation, the tools and techniques used for “digital” design are closely related to those for mixed-signal and RF design. We expect this trend to accelerate.

Given this data, it is essential to continue to invest in fundamental solutions for the growing “non-digital” side of EDA. **Improvements will rely on foundational mathematical research** to bring increased automation. Where human experts rely on intuition, design automation tools rely on a diverse range of aggressive optimization formulations: numerical, combinatorial, geometric. A unique attribute of these analog problems is that concurrent consideration of functional, electrical, and geometric aspects are necessary to compete with the manual designs. Experts often can transform large mixed-signal problems into small, workable abstractions that preserve the essential design features.

3.6 Nonlinear Model Reduction

Progress over the last twenty years has rendered the analogous "model order reduction" problems solvable for linear systems (e.g., via robust Pade' approximations), and for finite state digital systems (e.g., initially via symbolic model checking with BDDs, later with Boolean SAT-based bounded model checking). Unfortunately, no such robust theory exists for the more common nonlinear cases that dominate in the mixed-signal and low-level digital realm. Nor are there complete solutions for designing or assuring correct behavior of these analog systems in the presence of unavoidable statistical manufacturing fluctuations. Digital systems are designed to hide the physics of the fabrication process; analog systems are designed to exploit these behaviors. As fundamental components move further into the nano-scale regime, analog systems are increasingly vulnerable to these small upsets. Thus, great opportunities exist in building the next generation of fundamental numerical, combinatorial and geometric algorithms to handle these essential designs. Moreover, improvements in fundamental verification and in algorithmic simulation are mutually reinforcing. Larger and more complex artifacts can be designed and used only when their component interactions can be guaranteed with assurance; synthesis tools which rely on iteratively exploring a huge number of different solution configurations can be targeted to larger and more challenging designs only when each solution candidate can be evaluated in one second, instead of one hour or even one day. Also, nonlinear model reduction will be a key enabler in verification and model checking of hybrid systems.

4. Key EDA Challenges

We see the following as being the main EDA challenges.

4.1 Scalable Design Methodologies

At its core, design automation is about developing design methodologies that best address the emerging technology challenges. ASIC design methodology based on standard cells and synchronous timing was critical in providing the foundational basis for the design tools that eventually enabled it. Future nano-scale designs face significant complexity challenges which demand design methodologies that will continue to scale in the face of increasing complexity. The following attributes are critical for success:

1. **Disciplined, predictable design**. The design process needs to be highly disciplined; only then does it lend itself to automation. Further, predictability is important, both in the design process as well as in design quality. A predictable design process leads to predictable design schedules, and a predictable design quality leads to dependable results. Current design flows may not converge, or have unpredictable tool runtimes, or provide unexpected results.

2. **Intuitive design environments, simplified user interfaces**. Current design environments are too complex, with too many parameters that cannot be understood by designers. This is very limiting and needs to be replaced with very intuitive and simplified interfaces that can lead to fast design cycles, and thus enable designers to better explore the design space.

3. **Appropriate abstractions.** An important component of the ASIC design methodology was the separation of the electrical concerns from the logical aspects of hardware design. With nanoscale scaling, this separation is under threat. Hardware designers increasingly need to understand the underlying electrical models to better predict power consumption and reliability of circuits. This diminishing separation of concerns makes it harder to have multiple entry points into the design flow based on expertise; this limits the pool of designers and eventually the number of designs. Automatic abstraction techniques can play an increasingly larger role in helping with this. Our current block-based signoff techniques may not be sufficient to allow efficient design (in manpower and area) in the face of the increasing region of influence from physical effects.

4. **Scalable design methodologies along the axis of different circuit functions.** One example is that the lack of automation for embedded arrays poses an increasing problem. Arrays (by virtue of size and number) are fertile ground for automation; the need for efficient design is great, but the highly structured nature of the circuits lends itself to automation. Another example is analog circuit design, which also requires specialized design flows.

5. **Standardized interfaces.** It is essential that we have open interfaces for different parts of the design flow. Proprietary formats from vendors impede progress in our field.

4.2 Scalable Design Synthesis, Validation/Verification

Existing design synthesis and verification techniques are no longer scalable. Their costs and limitations are restricting the design of future computing platforms, and even the inclusion of additional features in existing platforms. If this is not suitably addressed, the impact will be felt in

all of computer science. This directly hits the economic basis of Moore's law—increased computation per unit cost with each generation, which in turn drives the development of novel applications that exploit this cheaper computation. This can no longer be counted on and will severely limit future computer science. There are multiple possible solution directions to be explored here:

- Higher level of design abstraction
- Better design principles and languages
- Formal and semi-formal verification techniques
- Runtime validation
- Extending the scalability of the underlying fundamental algorithms

4.3 Dealing With New Technology

Design in the late- and post-Si era needs to deal with new devices as well as manufacturing technology. Important changes in the future include 3D designs, graphene-based and other emerging devices, as well as new lithography techniques. These will need new techniques and tools for modeling, synthesis and simulation.

4.4 Designing with Uncertainty and Fragility

This adds a critical dimension to existing design methodologies as reliable circuits and systems will have to be built using unreliable fabrics. There are uncertainties due to physics, e.g., increased soft-errors in finer geometries due to energized particle hits and uncertainties in manufacturing in fine geometries due to system and random variations. These contribute a diverse set of failure modes that need to be accounted for at appropriate levels of the design.

4.5 New Classes of Algorithms.

There are several new classes of algorithms that need to be explored when developing scalable design methodologies.

1. **Linear/sub-linear algorithms**. Increasing complexity and problem scale cause quadratic, or in some cases even log-linear algorithms, to have unacceptable run times. A significant push is needed in exploring linear and sub-linear algorithms across design tools.
2. **Incremental algorithms**. Algorithms that can recognize and exploit incremental design changes can significantly help reduce design time.
3. **Parallel algorithms**. As we move to the multi/many-core era, it is critical that EDA algorithms be able to exploit future platforms.
4. **Deterministic algorithms**. With long design cycles, it is important for the algorithms to be deterministic. This is critical in reproducing results and providing predictability in the design process, and especially critical for parallel programs where races are an important contributing factor to non-determinism
5. **Design for security**. To ensure data privacy, it is increasingly important for designs to be resilient in the face of security attacks. Examples of this are various forms of side channel attacks. These can be tackled potentially as part of the design process.

5. Emerging Areas and EDA Technology

Given the new definition of EDA in Section 2.1, we felt that areas involving highly complex systems requiring modeling, analysis, and transformation among different levels of abstraction are good candidates that can benefit from EDA technologies. Examples of such areas, often in emerging fields, include the following:

1. **Biology systems.** Biology will transform into a precise, quantitative, bottom-up, predictive discipline, much as physics and engineering did over the last century. Such systems feature many individual entities that interact extensively and are organized hierarchically, where logical functionality arises from the hybrid interplay of discrete and continuous-time dynamics. Understanding how most biological systems work will require numerical and Boolean tools for analysis and design, similar to how CAD tools became indispensable to VLSI design. In biology, effective leveraging of, e.g., computational, abstraction and verification techniques from EDA, will be critical for catalyzing progress in core science areas such as systems and synthetic biology.
   - **System biology** captures interactions of discrete molecular elements (genes, proteins, etc.) that lead to collective properties at multiple levels (metabolic function, organ function, etc.). The outcome might be a predictive network hypothesis (the biological circuit) that correlates system behavior to lower-level structures.
   - **Synthetic biology** focuses on modification of biological systems at the molecular level to achieve new functions, such as bacteria that can attack cancer cells, or to lead to new bio-fuels. The EDA community has already made key contributions (e.g., automated abstraction techniques to improve simulation efficiency, applying BDD technology to accelerate drug design, oscillator phase macro-models for quantitative understanding of circadian systems, etc.). This trend will accelerate using compelling aspects of EDA (deep foundational underpinnings and tools for large-scale systems) to enable fundamental progress and new discoveries in the core science area of biological systems.
2. **Emerging computing/communications/storage fabrics and manufacturing substrates**. Nano-electronics, nano-photonics, nano-electromechanical systems, [3] and flexible electronics,[4,5] where diversity of functionality, manufacturability, variability, and reliability provide challenges and opportunities for modeling, analysis, synthesis, and integration.
3. **Analysis, characterization, and potential design of hybrid electronic/biological systems**. Examples include biological neural networks integrated digital and analog electronics-based stimuli and readouts.
4. **Cyber physical systems**. Such systems consist of the interaction of vast numbers of embedded systems, often requiring real-time control, such as intelligent transportation systems.
5. **Datacenter design and optimization**. It is increasingly important yet difficult to address performance, energy, reliability, interconnect, and cooling issues involving thousands to

[3] Korkin, Anatoli; Rosei, Federico (Eds.), *Nanoelectronics and Photonics: From Atoms to Materials, Devices, and Architectures,* Springer, 2008 (ISBN: 978-0-387-76498-6).
[4] FlexTech Alliance: http://www.flextech.org/.
[5] T.-C. Huang and K.-T. Cheng, "Design for Low Power and Reliable Flexible Electronics: Self-Tunable Cell-Library Design," *IEEE/OSA Journal of Display Technology (JDT)*, Vol. 5, Issue 6, June 2009. pp. 206-215.

millions of entities (servers, storage, and I/Os) in a modern large-scale datacenter under a dynamic workload.

6. **Software systems.**[6] These are especially relevant in the following two areas: 1) scalable and more precise large-scale software analysis, and 2) tools and methodologies to extract and manage concurrency. Current EDA technology has supported well the high degree of concurrency in IC and electronic system designs.

This list can be viewed as an initial recommendation for applying EDA techniques to adjacent/emerging fields, but it is by no means meant to be exhaustive. There are more candidates. For example, the workshop attendees discussed the possibility of applying EDA technologies to the design and analysis of social networks, and to the design and analysis of quantum information processing systems, especially quantum communication and quantum cryptography. However, the applicability and benefits of EDA technology to these fields are less certain (in comparison to other techniques that address problems in these fields). Therefore, these are not included in the initial list of recommendations from the workshop. Further joint investigations would be needed with other domain experts from physics, information science, etc., to develop a sharper focus and a more convincing justification, and to establish a consensus on feasibility and identify verifiable order-of-magnitude improvements.

The most readily transferable EDA assets to adjacent/related disciplines include:

- Modeling and analysis at multiple levels of abstraction
- Synthesis and transformation with optimization
- Validation of functional and structural properties as well as performance requirements
- Use of formal methods

The vast knowledge accumulated in these areas for solving IC and electronic system design problems can be applied rather directly to solve any of the new applications listed.

6. Educational Perspective

There is an apparent discrepancy between new graduate students from the U.S. and those from overseas in terms of their exposure to EDA. In the U.S. there is little teaching of core EDA topics at the undergraduate level—like logic synthesis, physical design, simulation (discrete and analog), testing, and formal verification. More seems to be done at leading universities overseas at the undergraduate level. Consequently, fewer domestic students apply to graduate schools in this area, and some who might have been attracted by the subject matter are lost to other disciplines. The workshop attendees agreed that a good design background is important in EDA, and for the most part, such courses are being offered. However, these mainly teach the use of canned CAD tools and cannot cover much about EDA algorithm topics in any depth. There was a feeling that a good senior-level introductory CAD class could be developed and offered more broadly in the U.S. universities.

Exactly what subset to teach is a real challenge, because EDA is a very broad and interdisciplinary field, and continues to expand; e.g., embedded systems is a relatively new topic

[6] J. R. Larus, T. Ball, Manuvir Das, R. DeLine, M. Fahndrich, J. Pincus, S. K. Rajamani, R. Venkatapathy, "Righting Software," *Software, IEEE Publication*, Vol. 21, Issue 3, May-June 2004, pp. 92-100.

of EDA. An ideal undergraduate course should bring out this breadth, but should avoid being just an enumeration of disparate topics; rather it should emphasize a set of problem areas which contain common underlying algorithmic themes. This would allow exposure to some algorithms in depth and give a flavor of the algorithmic and theoretic foundations of EDA.

At the graduate level, very few universities have the manpower to cover all of EDA since there are so many areas and topics. To emphasize this, the skill sets needed by an employer in the EDA field were used to analyze the kinds of skills and knowledge that should be taught. The graph shown in Figure 1 was used as a guide. The top layer lists the set of products that are part of an EDA company's current offerings. These include extraction, simulation, static timing analysis, place and route, synthesis, engineering change, and formal verification. The next layers of the graph (oval nodes) show software packages that are part of subroutines used in these tools. For instance, synthesis needs timing analysis, placement, logic synthesis and model checking. Extraction needs function-approximation methods, PDE solvers, model order reduction, and machine learning. The next layer lists the academic disciplines required by the people who implement state-of-the-art tools in the listed areas. For example, discrete optimization is used in machine learning, placement, routing, search, and logic optimization. The layer at the bottom categorizes the underlying mathematics as continuous and discrete.

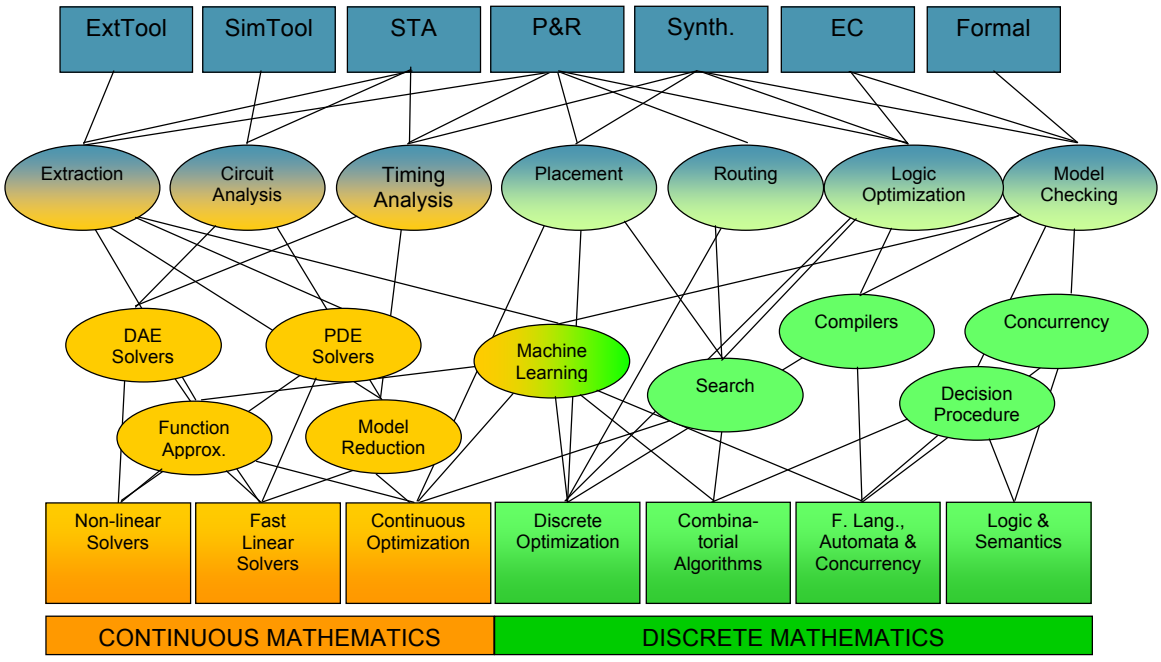

Figure 1. Fundamental Areas and Domain Knowledge in EDA

It was informative to look at a similar graph for some of the adjacent or emerging technologies that might be part of the future. That graph differed only in the first layer and the interdependences. Some of the future technologies listed were multi-domain micro-systems (such as micro-mechanics), new device and process modeling, software verification, systems biology, parallel computation, etc. Many of the disciplines that have been required in EDA are still fundamental and form the basis for future technologies. In addition, the types of complexities met, and the scale of the problems to be addressed, will be similar to those already encountered in EDA and already solved to some extent.

The conclusion is that training in EDA fundamentals and experience with EDA-scale problems are required for continuing technological innovations. Of course, not all of the domain knowledge needs to be taught by EDA personnel and certainly existing courses in other parts of EE and CS should be utilized as part of a graduate program. Even for the main core EDA courses, there are neither the numbers of professors nor students to warrant courses being taught each year. A suggestion was that a graduate curriculum in EDA should offer relevant in-depth courses in alternate years. In addition, it was suggested that there be a definite movement towards "nano-widget" courses (covering nano-wires, nano-switches, nano-memories, etc.) where students are encouraged to take these courses and professors are encouraged to participate, prepare and adapt EDA material which would supplement such courses. The EDA workshop group noted that in Europe, fewer actual graduate courses are taught because the education depends more on self-study, seminars, and learning while doing the research. However, it was not recommended that the U.S. take this approach because we felt that it is an educational advantage to offer a large variety of courses to graduate students.

On another note, in the past NSF and DARPA jointly funded MOSIS. MOSIS is a program providing access to fabrication of prototype and low-volume production quantities of integrated circuits. It lowers the cost of fabrication by combining designs from many customers onto multi-project wafers (MPW). The demise of this funding has been a serious impediment for advancing EDA education in universities, and to a certain extent, in research as well (more so at smaller universities). While the expenses for funding such a program have gone up dramatically, in order to maintain U.S. competitiveness it would be important to resurrect this support. While funding this could be expensive, NSF mechanisms that support infrastructure could be used. These include the CISE Computing Research Infrastructure (CRI) program, the NSF-wide Major Research Instrumentation (MRI) program, and the office of Cyber-Infrastructure (OCI). NSF MOSIS funds for the use of the DARPA secure foundry, or even foreign foundries could be investigated.

Another topic discussed at the workshop was the need to combat negative perceptions and negative but realistic observations about the EDA field. Some of these are:

- Currently, many EDA companies are hurting financially, and job opportunities are down.
- EDA summer internships are very tight.
- Venture capital for start-ups in EDA has decreased significantly. These have been a vital component of EDA and have served as major centers for research and development and employment of PhDs.
- Faculty positions in EDA are tight, aggravated by the difficulty of obtaining funding to support research and students.
- Student interest in EDA as a career has decreased in recent years.
- There are reduced industrial research efforts in EDA with the dissolution of high-quality research groups at Cadence and Synopsys, while the large system design companies have throttled back on the research components of their activities.
- Transition of academic research to industry is much harder than it used to be. Technologies are more complex and it is harder to get new ideas into the sophisticated and mature software offered by EDA vendors.

On the positive side are the following factors:

- EDA will not go away and cannot stagnate. It is absolutely necessary for the support of the design of complex systems such as micro-chips. EDA expertise is needed by EDA software companies and by large system-design houses. Start-ups will continue to be valuable in finding niche applications and researching solutions, and nurturing core EDA technologies as well as emerging ones. These should see a resurgence as the economy improves.
- Cooperation between industry researchers and developers and university faculty and students remains very high, probably among the highest among any discipline within computer science/engineering.
- As technology shrinks, the problems get harder, so not less but more EDA activity is required. This increased complexity puts more emphasis on modeling, model reduction, abstraction, etc.—techniques in which expert EDA personnel are well versed.
- EDA engineers are well paid, apparently better than most other types of engineers. Moreover, if the trend of fewer students entering the field continues, then supply and demand will assert itself; demand and salaries will increase.
- EDA training in its various disciplines, including complex and large problem solving, will be valuable as new growth areas come into play—such as nano-technologies, biology and other life science applications, new energy technologies, new energy conservation methods such as smart buildings, and maybe even the financial sector. Some of these are listed in the preceding section. Indeed, we see that students with EDA backgrounds were hired into these sectors.
- Aside from the new emerging hot areas, EDA continues with its own hot areas, such as system-level design, embedded software, design for manufacturing including lithographic and scaling problems, issues of robustness and unreliable components, parallelism, design and application of many-core (1000+) processors, application of probabilistic methods to enhance scaling of algorithms and problem solutions, and new methods for derivative and incremental design.

The bottom line is that EDA continues to have its hot and exciting areas with lots of new technology on the horizon (nano, bio, optical, etc.). Tremendous but exciting challenges are looming. EDA provides a very flexible foundation for solving future problems in the design of complex and large systems, with many ideas transferable to other fields, and researchers are still well paid. Future students should look down the road four to five years and keep these aspects in mind for when they might expect to graduate. EDA professors need to convey the idea that EDA is more than just supporting semiconductor implementation flows; it is broadly about the optimization, implementation and verification of complex systems using deep knowledge of an underlying technology.

7. Theory and EDA (Re-Engaging the Theory/Algorithm Community with EDA Research)

The theory/algorithm community has had a long history of involvement with EDA efforts, dating back to VLSI work in the 1980s. Floorplanning and layout/routing problems, among others, were studied extensively. Over time, many of the key algorithmic questions in these areas were either

resolved, or reduced to hard (and open) problems, and the strong initial connections between the communities became weaker.

Current evolution of EDA and theoretical methods suggests a number of directions for fruitful collaborations between the algorithms and EDA communities:

1. **The effectiveness of SAT solvers**. In the EDA community, SAT solvers have been engineered to the point where SAT, rather than being viewed as a hard NP-Complete problem, is seen as an "easy" subroutine used to reduce other hard problems. However, we do not yet have a clear understanding of why the SAT instances encountered in EDA problems are easier to solve. Such a study has both theoretical and practical merit. From a theoretical perspective, a better understanding of “easy” instances of SAT will naturally further develop our understanding of algorithms in the exponential time regime. This is an area that has been sorely underdeveloped in the algorithms community. EDA applications provide both a test-bed of instances, as well as concrete applications for this work. In related work, there has been extensive research on determining the threshold for random SAT. Specifically, researchers have looked into determining the precise ratio of variables to clauses to determine the exact point above which a random SAT formula is almost always satisfiable, and below which it is almost always not satisfiable. Although random instances are not likely to be the same as those encountered in practice, the insights gleaned from this research are likely to be very useful.
2. **New algorithm design paradigms for EDA applications.** Randomization and approximations have been among the most exciting developments in the realm of algorithm design over the past 20 years. Randomization allows the design (usually) of very simple and practical algorithms with strong (probabilistic) guarantees on performance. Many problems in the EDA pipeline include elements like state space exploration, or searches in high-dimensional parameter spaces. These problems can benefit greatly from randomized methods. Repeatability is an important concern for such applications, and this issue should be kept in mind when designing randomized algorithms (proper seed/trace management can address this issue quite easily). Approximation techniques are also a powerful way of obtaining guaranteed quality bounds for an optimization in an efficient manner. Approximation techniques are quite mature, and could be brought to bear on a number of the problems discussed. Another issue is that of incremental (or progressive) methods. The “waterfall” model of EDA pictures the design process as a pipeline with outputs from a phase progressively feeding into the next. It is important when operating such a pipeline that small changes made in one stage do not cause drastic changes in the final design. Incremental (and streaming) methods in algorithms are designed to adapt solutions to slight changes in data. Progressive methods are designed to provide better quality solutions as more time is invested in the computation. These two paradigms could benefit EDA pipeline design immensely.
3. **New computational models.** EDA systems are very compute-intensive, especially in the verification stages. Desktop computers now typically have several levels of caches, and are rapidly migrating to a multicore architecture which combines parallelism and a caching hierarchy. Efficiency of algorithms under these new architectures requires attention to cache efficiency and parallelism in addition to the traditional measure of the number of operations executed by the algorithm (i.e., the sequential running time). It is also desirable for the code to be portable across different architectures, while maintaining efficiency. The

cache-oblivious setting offers a simple and attractive model for sequential portable cache-efficient algorithm design, and many provably efficient algorithms have been designed in this model and experimentally run efficiently on modern uniprocessors with a cache hierarchy. In the last couple of years, attention has turned to modeling multicores effectively, and some promising results have been developed for basic algorithms in the theory community. EDA would benefit from incorporating efficient cache-oblivious [7, 8, 9] and multicore[10,11] techniques into EDA algorithms. This holds the promise of much faster parallel multicore implementations for EDA problems than currently exist, and could extend the scale of problems that we can feasibly solve quite significantly.

4. **New classes of algorithms.** New classes of algorithms are needed in all of EDA. These include sub-linear algorithms, incremental algorithms, and the cache-oblivious and multicore algorithms mentioned above. Sub-linear algorithms would be very beneficial in EDA in cases where a large number of candidate solutions need to be examined to determine if they satisfy a given property. Often in such cases, very fast (sub-linear time) algorithms can reject any candidate solution that is significantly "far" from the desired property. The use of such algorithms could help to speed up EDA computations quite significantly. Incremental and fully dynamic algorithms[12,13] are another class that has been well studied in theoretical algorithm design, and they hold much promise for EDA. Here, when components are added or modified in a large design, one seeks algorithms that incorporate the change in time that is much smaller than the size of the design. These algorithms typically run in time that is a function of the size of the change being made, and a very slow-growing function of the size of the design. In fully dynamic algorithms, most incremental and decremental changes are allowed. Many very fast fully dynamic algorithms have been developed for basic problems, such as graph connectivity and minimum spanning tree. This is an area that potentially could speed up EDA algorithms significantly.

5. **Use of EDA benchmarks.** Engagement of theorists through EDA benchmarks. EDA research generates large-scale benchmarks (e.g., SAT instances arising from verification) that are of value to the algorithms community. This engagement can help provide a better formal understanding of effective EDA algorithms.

6. **Robustness in design.** The theory community can provide insights into design techniques with formally provable robustness properties, even in the face of security attacks.

---

[7] M. Frigo, C. Leiserson, H. Prokop, and S. Ramachandran, "Cache-Oblivious Algorithms. In Proc. of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)," pages 285-297, 1999.

[8] L. Arge, M. Bender, E. Demaine, B. Holland-Minkley, and I. Munro, "An Optimal Cache-Oblivious Priority Queue and Its Application to Graph Algorithms," SIAM Journal on Computing, Volume 36, Issue 6, pages 1672-1695, 2007.

[9] R. Chowdhury and V. Ramachandran, "The Cache-Oblivious Gaussian Elimination Paradigm: Theoretical Framework, Parallelization and Experimental Evaluation," In Proc. ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pages 71-80, 2007.

[10] R. Chowdhury and V. Ramachandran, "Cache-Efficient Dynamic Programming Algorithms for Multicores," Proc. ACM SPAA, 2008, pages 207-216.

[11] L. G. Valiant, "A Bridging Model for Multi-Core Computing," In Proc. 16th European Symposium on Algorithms (ESA), volume 5193, pages 13--28, 2008.

[12] Holm, K. de Lichtenberg, M. Thorup, "Poly-Logarithmic Deterministic Fully-Dynamic Algorithms for Connectivity, Minimum Spanning Tree, 2-Edge, and Biconnectivity," Journal of the ACM, volume 48, pages 723-760, 2001.

[13] C. Demetrescu and G.F. Italiano, "A New Approach to Dynamic All Pairs Shortest Paths," Journal of the ACM, volume 51, pages 968-992, 2004.

7. **Statistical design.** With statistical variations playing an increasingly important role in design, there are opportunities in new techniques for modeling, analysis, and design (optimization) with statistical variations.

8. Recommendations to NSF

Based on the discussions presented in the preceding sections, the EDA workshop would like to make the following recommendations for improving DA programs in the future.

8.1 Research Programs

1. **New funding programs to support mid-scale or large-scale research efforts that couple design with EDA.** Many workshop attendees pointed out that current DA researchers no longer have direct interactions with circuit/system designers. Thus they do not have first-hand experience with the new design problems nor direct feedback on the fruitfulness of their research. However, modern system-on-a-chip designs in nano-scale technologies require design teams of substantial size. Therefore, it is very important to create large funding opportunities to support innovative design projects and couple them with leading-edge DA researchers. Such funding may be established jointly by the Experimental System Program and the DA Program at NSF CISE.
2. **New funding programs to support joint research programs between research groups from universities, commercial EDA companies, and large systems houses.** This would alleviate some of the recently diminished research in EDA.
3. **New funding programs to support shared infrastructure for design and design automation.** This would provide additional collaborations between industry and academia. However, significant resources are needed at the university which would produce non-commensurate research benefits. (It was pointed out that infrastructure development can be done by university groups, even with limited industrial support). The NSF CRI/MRI programs exist to help with this, but they need to be better utilized.
4. **New funding to support exploration of DA for emerging areas.** NSF should substantially increase its funding to the DA program so that it can form partnerships with other research programs in NSF to explore the new frontier for DA, in particular, with the following initiatives:
    - Joint program with the CPS program to explore design automation for cyber physical systems.
    - Joint program with architecture and networking programs to explore data center design and optimization.
    - Joint program with the software division to investigate design automation techniques for scalable and more precise large-scale software analysis, and tools and methodologies to extract and manage concurrency.
    - Joint programs with biological sciences to initiate programs in design automation for system biology and synthetic biology.

- Joint programs with the Engineering Directorate to explore design automation for emerging computing/communications/storage fabrics and manufacturing substrates.

5. **New funding to support interaction between DA and theory communities, as well as interaction between DA and mathematical sciences.** Such collaborations do exist, but are done in an ad hoc fashion. A well-formalized program within the DA program will greatly facilitate such collaborations.

8.2 Education Programs

1. **Support for the development of a senior-level EDA course.** This would emphasize the underlying algorithmic and theoretic foundations of EDA, while motivating EDA's breadth and flexibility with specific interesting applications. Materials might be broadly submitted by many faculties to a central PI who would meld the contributions into a viable semester course and make the materials available online.

2. **Support from NSF to develop shared courseware infrastructure in EDA**. Some faculty members have had exposure to connexions (cnx.org), an open platform for course sharing, which might be utilized.

3. **An increased post-doc program to alleviate the lack of research positions for new graduates.** Such a program was perhaps part of the stimulus effort, but was quite limited and not specific to EDA.

8.3 Collaboration with Industry

1. **An enhanced program to support longer-term faculty/industry interactions**. A tight connection with industrial reality and practice has always been critical in EDA. IP houses and the IDMs (integrated device manufacturers) jealously guard their data (design data, test data, etc.) while the costs of fabrication in the latest technologies will be tremendous. Access to technologies is important for academic research. This can be seeded by enhanced faculty stays in industry or conversely, visits by technical leaders from industry to academia. There needs be a commitment by industry to implement designs for the purpose of outside research. This could be enabled by matching NSF and industry contributions. In the Engineering Directorate there is a GOALI program to enable this; perhaps a similar program is needed for CISE.

2. **An enhanced program to support summer students working at EDA companies.** Students would be located physically at the company. Proposals for funding would be a joint effort between a faculty member and a research staff person at the company. This program could include small start-ups as well as EDA vendors and large system design houses.

3. **A program to help faculty members and graduate researchers spin off start-ups to commercialize successful research projects**. This would be similar to a SBIR program, but more focused on EDA. The goal would be to help cross the "death valley"—from a research paper or prototype to the first customer adoption, so that VCs or the large EDA companies could take over from there.

4. **A program to help marry faculty to existing start-ups** (related to the above). This would encourage new ventures in EDA-type activities.

We estimate that a 2.5X increase in the current funding level is needed to support these new initiatives. Part of it can hopefully be shared with other programs in CISE or other directorates. We also hope to see continued partnership and cost-sharing with industry, such as the multi-core program with SRC.

9. Acknowledgements

This workshop was sponsored by the National Science Foundation CISE/CCF Division under the grant CCF-0930477. The authors would like to thank the program director, Dr. Sankar Basu, for his support of this workshop and for providing valuable feedback to the drafts of this report. The lists of speakers and participants are available at http://cadlab.cs.ucla.edu/nsf09/. In particular, the authors would like to thank Prof. Sharad Malik (Princeton University), Dr. Pinaki Mazumder (NSF), and Suresh Venkatasubramanian (University of Utah) for serving as group discussion leaders. Their summary reports of the group discussions form the basis of many parts of Sections 1 to 4 and Section 7 of this report. Finally, the authors are grateful to UCLA staff members Alexendra Luong and Cassandra Franklin for their support in organizing the workshop, and also to Janice Wheeler for editing this report.

Appendix: Workshop Organization and Schedule

This NSF-sponsored workshop on EDA was held on July 8 and 9, 2009 in Arlington VA. There were two invited keynote addresses and 19 invited speakers.

Keynote Talks

1. *The Brave New Old World of Design Automation Research*, Ralph Cavin and Bill Joyner (SRC), and Wally Rhines (Mentor Graphics)
2. *Future IT Infrastructure Research Challenges: An HP Labs View*, Prith Banerjee (HP)

Invited Talks

1. *The Future of Electronic Design Automation: Methodology, Tools and Solutions*, Sharad Malik, Princeton
2. *EDA—Electronic Design Automation or Electronic Design Assistance?*, Andreas Kuehlmann, Cadence Design Systems
3. *Front-End SoC Design: The Neglected Frontier*, Arvind, MIT
4. *EDA Challenges in Systems Integration*, Jochen A. G. Jess, Eindhoven University (emeritus)
5. *Is Today's Design Methodology a Recipe for a "Tacoma Narrows" Incident?*, Carl Seger, Strategic CAD Labs, Intel Corp.
6. *Statistical Model Checking of Simulink Models*, Edmund M. Clarke, CMU
7. *Deconstructing Concurrency Heisenbugs*, Shaz Qadeer, Microsoft
8. *Test and Validation Challenges in the Late-Silicon Era*, Tim Cheng, UC Santa Barbara
9. *A Faulty Research Agenda*, Rupak Majumdar, UC Los Angeles
10. *Numerical Modeling and Simulation for EDA: Past, Present and Future*, Jaijeet Roychowdhury, UC Berkeley
11. *Analog CAD: Not Done Yet*, Rob A. Rutenbar, CMU
12. *A Flat Earth for Design and Manufacturing*, Jason Hibbeler, IBM
13. *Collaborative Innovation of EDA, Design, and Manufacturing*, Jyuo-Min Shyu, National Tsing Hua University
14. *From Computability to Simulation, Optimization, and Back*, Igor Markov, University of Michigan
15. *Working Around the Limits of CMOS*, Mary Jane Irwin, Penn State University
16. *More Moore's Law Through Computational Scaling—and EDA's Role*, David Z. Pan, University of Texas at Austin
17. *Robotics-Based Fabrication and Assay Automation for In Vitro Diagnostics Technologies*, Jim Heath, Caltech
18. *Synthetic Biology: A New Application Area for Design Automation Research*, Chris Myers, University of Utah
19. *EDA and Biology of the Nervous System*, Lou Scheffer, Howard Hughes Medical Institute

These talks took place on July 8. Abstracts and PowerPoint slides can be found on the URL: http://cadlab.cs.ucla.edu/nsf09/. July 9 was devoted to breakout sessions which were divided into five groups. Each group was asked to focus on a specific set of questions and a topic, although participants were encouraged to contribute to any of the topics during their discussions. The separate groups focused on the following topics.

1. **EDA Past, Present and Future Support**. The group consisted of P. Mazumder (NSF—leader), W. Joyner (SRC), J. Roychowdhury (Berkeley), C. Myers (Utah), E. Clarke (CMU), R. Rutenbar (CMU), and S. Venkatasubramanian (Utah). The group focused on providing a historical perspective and also discussed the funding situation for EDA.

2. **Research Opportunities and Interaction with Industry**. The group members consisted of Arvind (MIT), S. Basu (NSF), J. Hibbeler (IBM), R. Majumdar (UCLA), S. Malik (Princeton—leader), L. Scheffer (Howard Hughes Medical Institute), C. Seger (Intel), J-M. Shyu (NTHU Taiwan). The group focused on the general theme of looking at the future and discussing challenges and methods.

3. **EDA for Emerging/Adjacent Technologies.** This group consisted of Jason Cong (UCLA—leader), Prith Banerjee (HP), Tim Cheng (UCSB), Jim Heath (Cal.Tech.), Igor Markov (U. Mich.), Shaz Qadeer (Microsoft), Vijaya Ramachandran (UT Austin), and Lenore Zuck (NSF). The group focused on an understanding of EDA and its applicability to emerging/adjacent fields. Discussions were centered on giving a more precise definition of EDA, identifying emerging areas, and those aspects of EDA readily transferable to adjacent areas.

4. **Educational Aspects.** This group consisted of Robert Brayton (UC Berkeley—leader), Mary Jane Irwin (Penn State), Andreas Kuehlmann (Cadence), Jochen Jess (Eindhoven), and David Pan (UT Austin). The group concentrated on the following educational topics during the discussion: core curriculum for EDA, advice for new students, how to sell EDA and combat misconceptions, good news and bad news for EDA, put situation of EDA industry in perspective.

5. **EDA and Theory.** This group consisted of Richard Lipton (Georgia Tech), Vijaya Ramachandran (UT Austin), and Suresh Venkatasubramanian (U. Utah). They were tasked with listening to the first day's talks with the purpose of identifying areas where computer science theorists and EDA personnel can effectively collaborate.

After the groups met on the second day, the workshop reconvened to hear a summary of the reports of the sessions. Later, these groups wrote up their discussions and conclusions; these form the main parts of this report.