HARDWARE SPECIFICATION, VERIFICATION AND SYNTHESIS: MATHEMATICAL ASPECTS pdf

1: FORMULA - Modeling Foundations - Microsoft Research

Hardware Specification, Verification and Synthesis: Mathematical Aspects Mathematical Sciences Institute Workshop Cornell University, Ithaca, New York, USA July , Proceedings.

CAV Award Nomination deadline: February 20, For details about the CAV award nomination, please see the following page: Paper Submission Paper submissions in CAV fall into one of the following three categories see more information below: Simultaneous submission to other conferences with proceedings or submission of material that has already been published elsewhere is not allowed. The PC chairs may solicit further reviews after the rebuttal period. Regular papers at CAV will follow a full double blind review process, which means that author names and affiliations must be omitted from the submission. Additionally, if a submission refers to prior work done by the authors, the reference should be made in the third person. These are firm submission requirements, and any regular paper that does not conform to these requirements will be rejected without review. Authors can include a clearly marked appendix at the end of their submissions that is exempt from the page limit restrictions. However, the reviewers are not obliged to read the contents of these appendices. These papers should contain original research and sufficient detail to assess the merits and relevance of the contribution. Papers will be evaluated on basis of a combination of correctness, technical depth, significance, novelty, clarity, and elegance. Authors of accepted regular papers will be invited but are not required to submit the relevant artifact for evaluation by the artifact evaluation committee. Tool Papers Tool Papers should not exceed 8 pages, not counting references. Note that tool papers require the submission of an artifact for evaluation by the paper submission deadline. Tool papers should describe system and implementation aspects of a tool with a large potential user base experiments not required, rehash of theory strongly discouraged. Papers describing tools that have already been presented in any conference will be accepted only if significant and clear enhancements to the tool are reported and implemented. Artifacts must be submitted separately in the artifact evaluation track in addition to the manuscript, which needs to be submitted in the main track. In special cases, where an artifact cannot be submitted, the authors should contact the Artifact Evaluation Chair Michael Tautschnig to find alternate modes of artifact evaluation. Artifacts will be evaluated concurrently with the review process and the program committee will have access to the artifact evaluation while making their decision. These papers are expected to describe the use of formal methods techniques in industrial settings or in new application domains. Papers in this category do not necessarily need to present original research results but are expected to contain novel applications of formal methods techniques as well as an evaluation of these techniques in the chosen application domain.

HARDWARE SPECIFICATION, VERIFICATION AND SYNTHESIS: MATHEMATICAL ASPECTS pdf

2: CiteSeerX â€" The Verification of a Bit-Slice ALU

*Hardware Specification, Verification and Synthesis: Mathematical Aspects: Mathematical Sciences Institute Workshop. Cornell University Ithaca, New (Lecture Notes in Computer Science) [Miriam Leeser, Geoffrey Brown] on www.enganchecubano.com *FREE* shipping on qualifying offers.*

Specification[edit] Formal methods may be used to give a description of the system to be developed, at whatever level s of detail desired. This formal description can be used to guide further development activities see following sections ; additionally, it can be used to verify that the requirements for the system being developed have been completely and accurately specified. The need for formal specification systems has been noted for years. Development[edit] Once a formal specification has been produced, the specification may be used as a guide while the concrete system is developed during the design process i. If the formal specification is in an operational semantics, the observed behavior of the concrete system can be compared with the behavior of the specification which itself should be executable or simulateable. Additionally, the operational commands of the specification may be amenable to direct translation into executable code. If the formal specification is in an axiomatic semantics, the preconditions and postconditions of the specification may become assertions in the executable code. Verification[edit] Once a formal specification has been developed, the specification may be used as the basis for proving properties of the specification and hopefully by inference the developed system. Human-directed proof[edit] Sometimes, the motivation for proving the correctness of a system is not the obvious need for reassurance of the correctness of the system, but a desire to understand the system better. Consequently, some proofs of correctness are produced in the style of mathematical proof: A "good" proof is one which is readable and understandable by other human readers. Critics of such approaches point out that the ambiguity inherent in natural language allows errors to be undetected in such proofs; often, subtle errors can be present in the low-level details typically overlooked by such proofs. Additionally, the work involved in producing such a good proof requires a high level of mathematical sophistication and expertise. Automated proof[edit] In contrast, there is increasing interest in producing proofs of correctness of such systems by automated means. Automated techniques fall into three general categories: Automated theorem proving , in which a system attempts to produce a formal proof from scratch, given a description of the system, a set of logical axioms, and a set of inference rules. Model checking , in which a system verifies certain properties by means of an exhaustive search of all possible states that a system could enter during its execution. Abstract interpretation , in which a system verifies an over-approximation of a behavioural property of the program, using a fixpoint computation over a possibly complete lattice representing it. Some automated theorem provers require guidance as to which properties are "interesting" enough to pursue, while others work without human intervention. Model checkers can quickly get bogged down in checking millions of uninteresting states if not given a sufficiently abstract model. Proponents of such systems argue that the results have greater mathematical certainty than human-produced proofs, since all the tedious details have been algorithmically verified. The training required to use such systems is also less than that required to produce good mathematical proofs by hand, making the techniques accessible to a wider variety of practitioners. Critics note that some of those systems are like oracles: There is also the problem of " verifying the verifier "; if the program which aids in the verification is itself unproven, there may be reason to doubt the soundness of the produced results. Some modern model checking tools produce a "proof log" detailing each step in their proof, making it possible to perform, given suitable tools, independent verification. The main feature of the abstract interpretation approach is that it provides a sound analysis, i. Moreover, it is efficiently scalable, by tuning the abstract domain representing the property to be analyzed, and by applying widening operators [11] to get fast convergence. Applications[edit] Formal methods are applied in different areas of hardware and software, including routers, Ethernet switches, routing protocols, and security applications. There are several examples in which they have been used to verify the

functionality of the hardware and software used in DCs[clarification needed]. Intel uses such methods to verify its hardware and firmware permanent software programmed into a read-only memory [citation needed]. Dansk Datamatik Center used formal methods in the s to develop a compiler system for the Ada programming language that went on to become a long-lived commercial product. There are many areas of hardware, where Intel have used FMs to verify the working of the products, such as parameterized verification of cache coherent protocol, [17] Intel Core i7 processor execution engine validation [18] using theorem proving, BDDs , and symbolic evaluation , optimization for Intel IA architecture using HOL light theorem prover, [19] and verification of high performance dual-port gigabit Ethernet controller with a support for PCI express protocol and Intel advance management technology using Cadence. Formal methods are most likely to be applied to safety-critical or security-critical software and systems, such as avionics software. For sequential software, examples of formal methods include the B-Method , the specification languages used in automated theorem proving , RAISE , and the Z notation. In functional programming , property-based testing has allowed the mathematical specification and testing if not exhaustive testing of the expected behaviour of individual functions. The Object Constraint Language and specializations such as Java Modeling Language has allowed object-oriented systems to be formally specified, if not necessarily formally verified. For concurrent software and systems, Petri nets , process algebra , and finite state machines which are based on automata theory - see also virtual finite state machine or event driven finite state machine allow executable software specification and can be used to build up and validate application behavior. Another approach to formal methods in software development is to write a specification in some form of logicâ€”usually a variation of first-order logic FOL â€”and then to directly execute the logic as though it were a program. There is also work on mapping some version of English or another natural language automatically to and from logic, and executing the logic directly. Examples are Attempto Controlled English , and Internet Business Logic, which do not seek to control the vocabulary or syntax. A feature of systems that support bidirectional English-logic mapping and direct execution of the logic is that they can be made to explain their results, in English, at the business or scientific level. You can help by converting this section to prose, if appropriate. Editing help is available.

HARDWARE SPECIFICATION, VERIFICATION AND SYNTHESIS: MATHEMATICAL ASPECTS pdf

3: dblp: Mathematical Science Institute Workshop

Hardware Specification, Verification and Synthesis: Mathematical Aspects Mathematical Sciences Institute Workshop. Cornell University Ithaca, New York, USA.

Approaches[edit] One approach and formation is model checking , which consists of a systematically exhaustive exploration of the mathematical model this is possible for finite models , but also for some infinite models where infinite sets of states can be effectively represented finitely by using abstraction or taking advantage of symmetry. Usually this consists of exploring all states and transitions in the model, by using smart and domain-specific abstraction techniques to consider whole groups of states in a single operation and reduce computing time. Implementation techniques include state space enumeration , symbolic state space enumeration, abstract interpretation , symbolic simulation , abstraction refinement. The great advantage of model checking is that it is often fully automatic; its primary disadvantage is that it does not in general scale to large systems; symbolic models are typically limited to a few hundred bits of state, while explicit state enumeration requires the state space being explored to be relatively small. Another approach is deductive verification. It consists of generating from the system and its specifications and possibly other annotations a collection of mathematical proof obligations, the truth of which imply conformance of the system to its specification, and discharging these obligations using either interactive theorem provers such as HOL , ACL2 , Isabelle , Coq or PVS , automatic theorem provers, or satisfiability modulo theories SMT solvers. This approach has the disadvantage that it typically requires the user to understand in detail why the system works correctly, and to convey this information to the verification system, either in the form of a sequence of theorems to be proved or in the form of specifications of system components e. Software[edit] Formal verification of software programs involves proving that a program satisfies a formal specification of its behavior. Subareas of formal verification include deductive verification see above , abstract interpretation , automated theorem proving , type systems , and lightweight formal methods. Fully featured dependently typed languages support deductive verification as a special case. Another complementary approach is program derivation , in which efficient code is produced from functional specifications by a series of correctness-preserving steps. An example of this approach is the Bird-Meertens Formalism , and this approach can be seen as another form of correctness by construction. These techniques can be sound , meaning that the verified properties can be logically deduced from the semantics, or unsound, meaning that there is no such guarantee. A sound technique yields a result only once it has searched the entire space of possibilities. An example of an unsound technique is one that searches only a subset of the possibilities, for instance only integers up to a certain number, and give a "good-enough" result. Techniques can also be decidable , meaning that their algorithmic implementations are guaranteed to terminate with an answer, or undecidable, meaning that they may never terminate. Because they are bounded, unsound techniques are often more likely to be decidable than sound ones. Verification and validation[edit] Main article: Validation is the complementary aspect. Validation usually can be done only dynamically, i. Automated program repair[edit] Main article: Automatic bug fixing Program repair is performed with respect to an oracle , encompassing the desired functionality of the program which is used for validation of the generated fix. A variety of techniques are employed, most notably using satisfiability modulo theories SMT solvers, [4] and genetic programming , [5] using evolutionary computing to generate and evaluate possible candidates for fixes. The former method is deterministic, while the latter is randomized. Program repair combines techniques from formal verification and program synthesis. Fault-localization techniques in formal verification are used to compute program points which might be possible bug-locations, which can be targeted by the synthesis modules. Repair systems often focus on a small pre-defined class of bugs in order to reduce the search space. Industrial use is limited owing to the computational cost of existing techniques. Industry use[edit] The growth in complexity of designs increases the importance of formal verification techniques in the hardware industry. Important aspects of

hardware design are amenable to automated proof methods, making formal verification easier to introduce and more productive.

4: EECS C Spring Homepage

*Hardware Specification, Verification and Synthesis: Mathematical Aspects - Workshop Proceedings [Miriam Leeser, Geoffrey Brown] on www.enganchecubano.com *FREE* shipping on qualifying offers. Current research into formal methods for hardware design is presented in the papers in this volume.*

5: Formal methods - Wikipedia

Current research into formal methods for hardware design is presented in the papers in this volume. Because of the complexity of VLSI circuits, assuring design validity before circuits are manufactured is imperative.

6: Research Area: DMA | EECS at UC Berkeley

Workshop on Hardware Specification, Verification and Synthesis: Mathematical Aspects July 05 - 07, Verification and Synthesis: Mathematical Aspects table of.

7: Call for Papers | CAV

Hardware verification techniques offer the possibility of significantly enhanced assurance for secure systems at the lowest levels of system design and implementation. Security can provide an important and challenging applications arena in which hardware-oriented formal approaches can be tried and refined.

8: Formal verification - Wikipedia

Add tags for "Hardware Specification, Verification and Synthesis: Mathematical Aspects: Mathematical Sciences Institute Workshop. Cornell University Ithaca, New York, USA. Cornell University Ithaca, New York, USA.

HARDWARE SPECIFICATION, VERIFICATION AND SYNTHESIS: MATHEMATICAL ASPECTS pdf

Te Llevare De La Mano Para Que No Te Caigas C windows forms tutorial Green guide to cars and trucks Global books in print Rights and responsibilities : human rights to human security Health, information, and migration Whole body MR imaging IV. The literary style. 1930. Civil engineering mechanics of materials Writing the basic resume ch. 6. The Legacy of a legend A circle round the sun Jesus for beginners Pleasure and quality of life The intellectual war on science Law of equivalents in its relation to political and social ethics Theorizing critical multicultural analysis of childrens literature Jennifer weiner who do you love 10 Cheshire cycle tours. Regalars Charles Brackett Whats the time, Little Wolf? Mayra Veronica 2007 Calendar For prodigals other sinners An American Emmaus Project management scholarly articles Some of my friends have tails. Immigrant children and education Len Rieser Panurge orator: The Parisian lady Retail management a strategic approach Twin falls family history center newsletter Christine warren the others Organic harmonies, 1855-1862 I purr, therefore I am Functions of management lecture notes Male and female reproductive disorders, male reproductive disorders Jose / Foxit disable ument protection popup In Search of Meaning and Coherence Introduction to Mark Performance evaluation, benchmarks, and attribution analysis Frederigo da Montefelotro by Denis Mack Smith