WWW.BDD-PORTAL.ORG: An Electronic Basis for Cooperative Research in EDA

Christoph Meinel, University of Trier, Germany
Arno Wagner, Institute for Telematics at Trier, Germany

1. Introduction

Today the Internet offers the possibility of standardized and global communication without a need for special hardware or expensive infrastructure. While a lot of resources are available on the World Wide Web (WWW), there is no central place to access these resources in a convenient way. There are search engines, but they are indexing only small segments of the web. According to a recent survey[1] the search engine with the best coverage covers only 13 percent of all pages. And even where search engines reach, the result of their searches are unstructured and often the user is presented with an all-or-nothing situation, where queries either return far to many hits or none at all. One solution is the recent notion of so called "portal"-sites, that organize the WWW contents into categories and in some cases even grade the quality. But what efforts are underway are mostly targeted at the general public and relative small groups, like research communities, are not commercially interesting to existing portals. Another Problem is that the features needed in research portals are different from other portals. While links to members of the community, lists of events like conferences or workshops, and material like software or documentation are fairly standard, research portals might need special content, not commonly found in other portals.

To address the needs of the (Binary) Decision Diagram [2,3] (BDD) research community, we have created a specialized portal site for this area. The specific problem with BDDs is that it is usually impossible to evaluate the performance of a new BDD algorithm without actually trying it out. On the other hand these algorithms are very sensitive to the environment they are executed in, consume a lot of resources when run and are difficult to include in existing software packages. So comparison of new methods with existing ones is generally problematic. To address this problem our portal includes a system for the online "publication" of BDD functionality by allowing the use of tools that contain BDD methods via a WWW interface. The system is mainly aimed at facilitating the process of determining whether a BDD method is suitable for a specific application and at allowing easy comparison between methods. It is open to all researchers who wish to publicise their results in this way. Because applications using BDDs are usually quite memory and processor intensive, the system needs to do computation in a distributed way on a cluster of similar computers. The system represents a major effort and greatly facilitates the evaluation of research results in BDD algorithms. Given the importance of BDD in verification, simulation, synthesis and similar uses in circuit design and other areas, such an effort seems to have been overdue.

The Paper is structured as follows: Section 2 gives an overview about (Binary) Decision Diagrams and their specific characteristics. Section 3 discusses the system that allows online usage of BDD methods. Section 4 reviews technical details of the system and Section 5 gives an overview over the portal site the system is embedded in. The paper ends with a conclusion in Section 6.

2. About Decision Diagrams

(Binary) Decision Diagrams (BDDs), most notably in the form of Ordered Binary Decision Diagrams[2,3] (OBDDs), are presently the most important data structure for the representation of Boolean expressions. They are extensively used for simulation, modelling and verification of digital circuits, often being orders of magnitude more powerful than other techniques. But while BDD-based methods perform well in many cases, the underlying problem of representing subsets of a Boolean vector space is known to be hard. This means that circuit descriptions given as a relatively small Boolean formula can have an extremely large BDD representation. Unfortunately formulas and other compact representations are unsuitable for use in computations and at present it seems BDDs are the best representation for use in such computations. BDD representations are sometimes small, when just using a straight-forward construction. More often they are too large to be represented in computer memory (e.g. larger than 1000 MB), but can be made significantly smaller by changing the parameters of the representation while the BDD is being built or computations are performed. Today, these size optimization techniques are critical for the size of tasks that can be done efficiently with BDDs. On the other hand the problem of BDD minimization is not fully understood yet, and there is a lot of research going on to find better optimization techniques, for special cases as well as for the general case.

When a new optimization method is found, researchers usually publish tables containing a limited set of benchmark results, obtained from doing calculations on publicly known benchmarks. These tables usually list speed and memory needs, as well as cases where the result could not be computed with the available resources. While it is possible to get a first impression by looking at these tables, BDD methods are known to be very sensitive to the actual nature of a computation. A method that performs better on some benchmark circuits might well have catastrophic performance on others. Unfortunately it is not possible to make meaningful predictions about the behaviour from the limited published information. To make things even worse, BDD method performance is quite sensitive to the computational environment used. Practical experience shows that it is often infeasible for someone other than the creator of a new method to create an experimental environment in which a new method performs as in the published results. This is due to variations in the experimental set up and due to details of the actual implementation that where not described in the publications. This is not a failure on the part of the authors of such publications. Rather it is extremely difficult to describe a computation environment with the degree of accuracy needed. A reason is that there are a lot of characteristics to a computational environment. A second reason is that not all of these characteristics are obvious. Furthermore some characteristics are unimportant, while others are not.

For these reasons when a new method is evaluated to decide, whether is should be used in a software package to replace an older method, comparing the two methods is very difficult. In essence the only feasible option is to reimplement the new method within the package and try it out. A process that can take several months and has an unpredictable outcome. On could argue, that obtaining the code of the new method would be sufficient to do test runs. Unfortunately this would mean that the old method would have to be implemented within the tool the new method is contained in. Because a meaningful evaluation of the performance of a new method takes so much effort, it is usually not done, greatly hampering scientific cooperation and use of academic research results into industrial applications.

3. Decision Diagrams in the WWW

Our approach of dealing with these problems is to offer a web based system, where comparison between different heuristics can be done in a standardized computational environment. While this does not eliminate the need for individual evaluation of a method, it allows to do a meaningful comparison of new and older methods with little effort, enabling parties interested in new methods to select those for more intensive study that are most likely to fit their needs.
BDD computations take a lot of resources, but do not need very much input parameters. Usually just one or two data files and some parameters will be enough to describe complex tasks. We therefore use a simple WWW interface that allows researchers to request computations, an example can be found in Figure 1. First the user optionally gives his email address to be mailed the results (results can also viewed via the WWW, but as computations may take long, the email feedback is convenient). Then he chooses some parameters. In the example given, these are the choice between the method in question (here Block Restricted Sifting) and a comparison method, usually ordinary Sifting. Finally the user transfers a data file containing the circuit or chooses one of the predefined circuits. After pressing the submit button a new job is created, queued and executed as soon as the needed resources become available. During the waiting time the user is being kept updated about the status of his job. The status can be waiting, with a certain number of jobs to be executed before this particular job, or it can be active, meaning currently under computation.

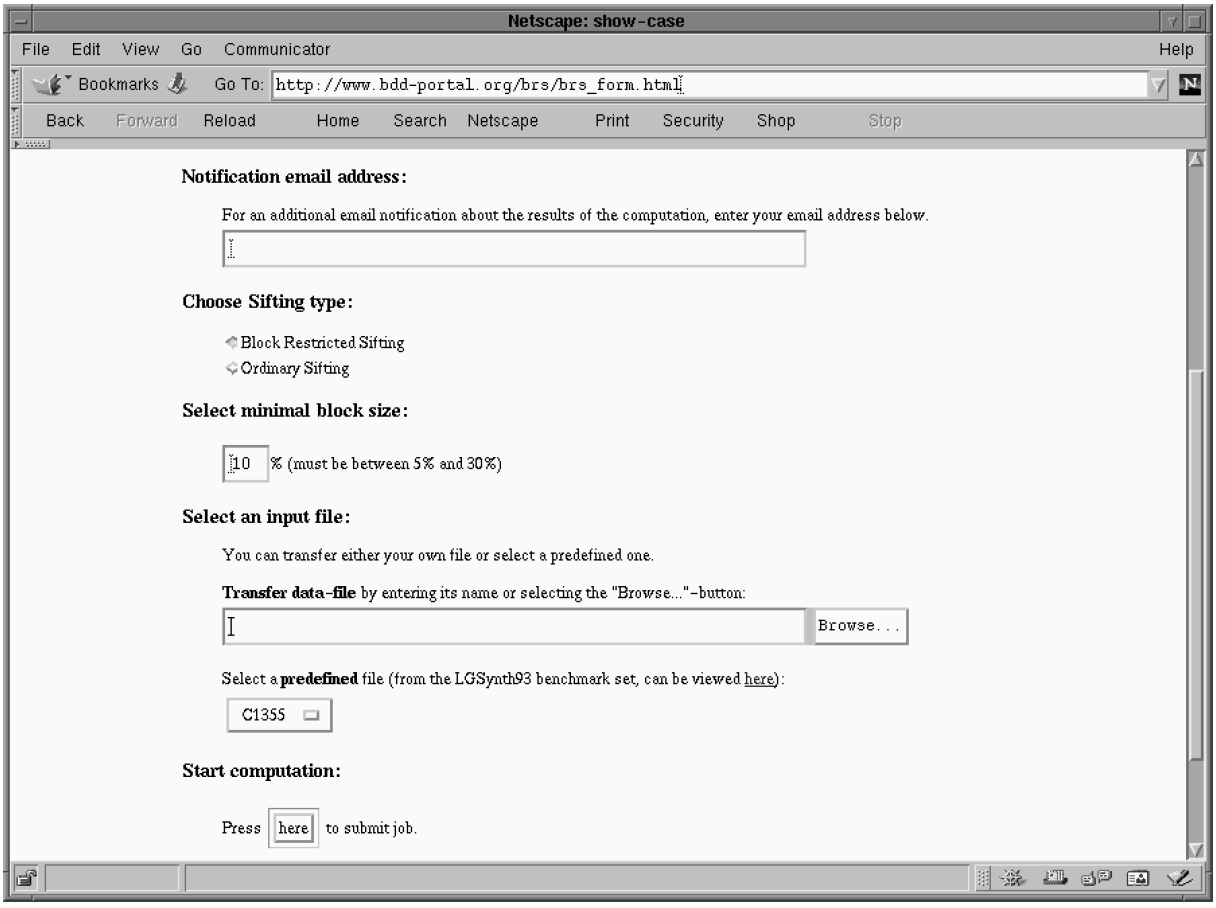

Figure 1

While the interface presented to the user is simple, there are a number of requirements the actual system below this interface layer needs to fulfil in order to be of practical use.

Flexibility and Speed: To achieve an appropriate overall speed it is necessary to distribute the actual computations on a number of computers. The number of computers used in the system should be easily adjustable, and it should even be possible to include computers that are at some other place and are only reachable by an Internet connection. To maintain comparability of results it is necessary that the pool of computers can be divides into groups of machines with the same characteristics.

Reliability: As computations can take in the order of hours and as there might be a number of pending computations, there should be mechanisms that guarantee that jobs submitted will not be lost in case of a crash and will be executed automatically once the system works again. Restart of the system in case of failure and periodical self-examination to determine whether the system still works are also important for a high level of reliability.

Security: Security here means the protection of intellectual property. No designer is willing to give away his carefully optimized circuits, just to test some new heuristic. BDDs are a canonical representation of circuits, that don't contain implementational details. It is therefore often possible to give encodings of BDDs as input instead of the circuits themselves. Where this does not work, circuit details can be obscured[4], If a circuit to be used as input to a computation is not secret, it can be used directly of course.

Ease of Use: The system should be easy to use, not requiring a long learning period. Ideally the user interface would be so intuitive that almost no explanation how it works is necessary. We believe our web interface satisfies this condition.

4. Technical Aspects

To achieve the system characteristics described in the last section, we have chosen the overall system structure shown in Figure 2. To the left side is the user interface. Currently there is only the web interface. The central component in the middle is the scheduler, which manages and coordinates every computation done with the system. The interface(s) communicate with the scheduler via a cleanly defined protocol, at the moment by using the file system. In case remote interfaces should be included into the system communication via the Internet is also possible. The scheduler is also responsible for crash recovery. To allow for automated recovery in most cases, the state of the scheduler is dumped whenever some significant state change occurs. As this state is not very large, dumping does not represent a significant overhead. From these state dumps both automated and manual recovery of the waiting jobs, waiting queues and other information important for continuing the computations is possible. In case of automated recovery, for example after a power fail, the computations resume without any administrator intervention. In case of software related crashes, several attempts at automatic recovery are made, but naturally there remains the possibility that the system state is corrupt in a way that cannot be corrected automatically. In the latter case the administrator is notified immediately via email. On the right side of Figure 2 is the cluster of computers performing the actual computations. Every one of these computers is fitted with a small demon that takes requests for specific computations from the scheduler and returns results and diagnostic information in case of failures. The demons also monitor resource consummation of the tools during the actual computations in order to catch errors within the tools, that lead to too high memory use or too much use of CPU time. This

monitoring is important, since the tools containing the BDD methods are research tools that might still contain errors and get out of hand. The computers used are non-dedicated and other computations are done on them as well. The monitoring ensures that the computers do not become unavailable because of errors in the tools.
At the moment computations are done on a number of Intel Pentium III 500 Computers running Linux 2.2.

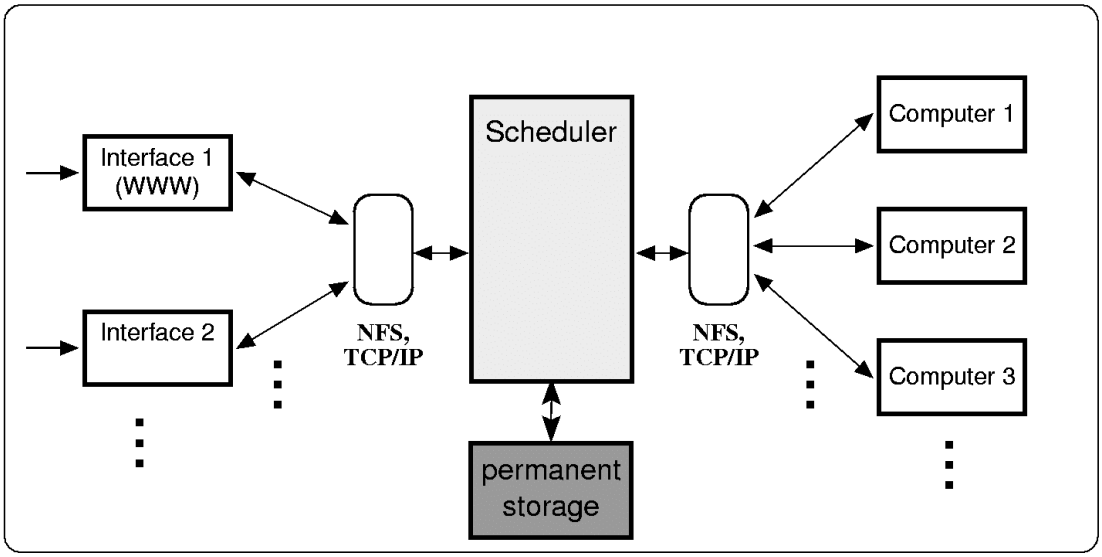

Figure 2

The scheduler itself was developed object-orientated using EIFFEL, which lead to a clean structure and easy incorporation of extensions. The dumping mechanism was added by creating a special class, DUMPAPLE, and having every other class carrying vital data inherit from this class and implement some dumping features that DUMPABLE contains in deferred form. Another benefit is that EIFFEL allows a design method called "design by contract" [5], that is very useful in creating (nearly) error free code. All the interfaces and the demons supervising the actual computations where done in PERL[6], which is very well suited for this kind of small applications.
Currently the following BDD methods are available: Sample Sifting[7], Block Restricted Sifting [8], Linear Sifting[9] Simulated Annealing and the Genetic Method within CUDDs[10] Nanotrav. Sample Sifting and Block Restricted Sifting within SMV[11], and finally Lazy Group Sifting[12] within VIS[13]. The number of methods will grow further as research yields new results.

5. The Portal

The structure and components of our BDD portal can be found in Figure 3. The topmost component is a list of links to homepages of active researchers. "Active" means that they have at least one refereed publication in the area of (Binary) Decision Diagrams. This list aims to be a complete representation of the BDD research community. Where no homepage could be found, an email address is provided instead.
The second component is a collection of all relevant events like conferences, workshops and other events. Events can be browsed by date, name, or, if applicable, submission deadline. The list is frequently updated and kept current.

The third component is a database of literature on BDDs available via the web. This database contains information about technical reports, papers and other texts with the possibility of full-text searches, even where the documents are only available in PostScript or PDF formats. This is a major step upward compared with the possibilities of conventional search engines. Search results will include the first lines of text of the documents found, and a link to their original location. The documents in the database are collected by a specialized robot, that is driven by the links to the homepages of researchers. The robot also does pre-processing of the documents so their text can be entered into the database. The robot creates the searchable index as well.

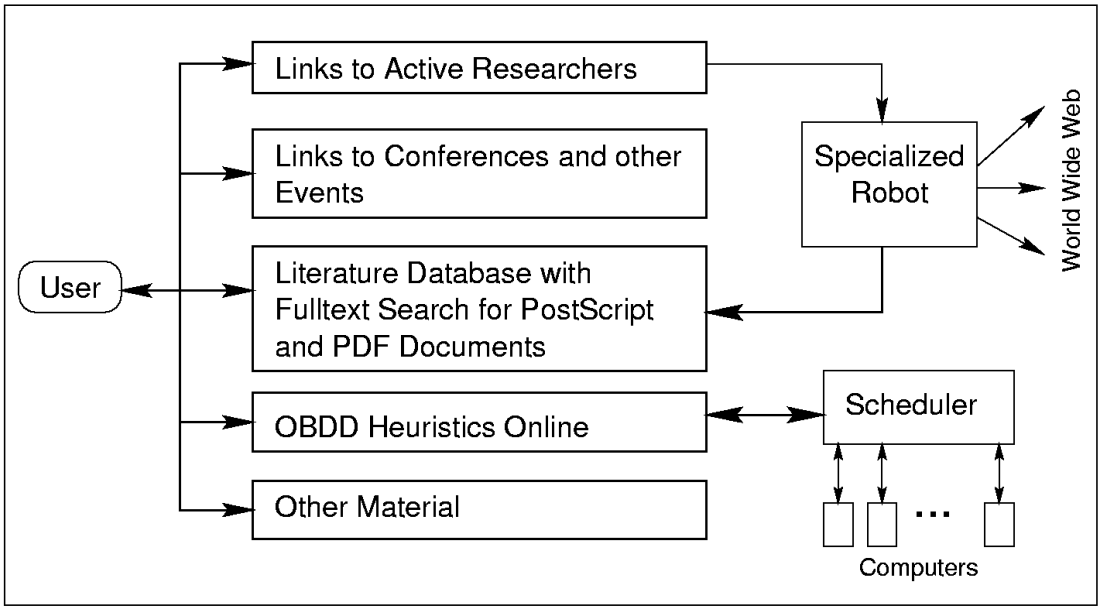

Figure 3

The fourth component is the active component allowing the use and evaluation of BDD heuristics via the web. Other material, like links to relevant benchmark circuits, is available as well.

6. Conclusion

We have explained why conventional text publications are insufficient in the area of BDD research. To solve this problem, we have created a system that allows the evaluation and comparison of new BDD methods with little effort and without the need for specific hard- or software with the user. Instead an Internet-based approach was chosen. This approach consists of a central server, coordinating requests and distributing the actual computations onto a cluster of computers. The system is open to anyone. Inclusion of further tools and methods into the system is easy and contribution of that nature by researchers in the area of BDDs are welcome. As there was previously no satisfactory way to allow researchers and practitioners to evaluated new methods unless they where willing to do al lot of work, the system represents a major step forward for the cooperation within this research community. Furthermore we have built a WWW portal site, for the BDD research community. This portal links researchers, events and other relevant material. It also includes a specialized literature search engine, that features a search mechanism superior to other conventional search engines. The collected information greatly facilitates retrieving information in the area of BDD research from the WWW. We believe that

our efforts are an important step in improving the usage of the WWW for BDD research and has exemplary character for other research areas. The portal can be accessed at http://www.bdd-portal.org.

Abbreviations:

EDA	Electronic Design Automation
WWW	World Wide Web
BDD	(Binary) Decision Diagram
OBDD	Ordered Binary Decision Diagram
PDF	Portable Document Format
NFS	Network File System
TCP	Transmission Control Protocol
IP	Internet Protocol

Literature:

1. Lawrence, S., Gilles, C. L., Accessibility of information on the web. Nature Volume 400, Number 6740 1999.
2. Bryant, R. E., Graph Based Algorithms for Boolean Function Manipulation. IEEE Transactions on Computers, C-35 1986 pp 677-691
3. Meinel, Theobald. Algorithms and Data Structures in VLSI Design. Springer, 1998.
4. Hauck, S., Knoll, S., Data security for web based CAD. Proceedings of the 35th Design Automation Conference, 1998
5. Meyer, Bertrand, Object Oriented Software Construction, Second Edition. Prentice Hall,1997. 1254 p.
6. Wall, L., Christiansen, T., Schwarz, Programming Perl, 2nd Edition. O'Reilly, 1996.
7. Meinel, C., Slobodová, Sample Methods for Minimization of OBDDs. IWLS'98, Granlibakken Resort, Lake Tahoe, USA, 1998
8. Meinel, C., Slobodová, Speeding up Variable Ordering of OBDDs. Proceedings of IEEE ICCD'97, Austin, Texas, USA, 1997.
9. Meinel, C., Theobald, T., Local Encoding Transformations for Optimizing OBDD-Representations of Finite State Machines. In proceedings FMCAD'96, Springer, LNCS 1166, 1996, pp 404-418
10. Somenzi, F. Colorado University decision diagram package. Available at <ftp://vlsi.colorado.edu/pub/>, [referenced 15.3.2000]
11. CMU School of Computer Science. Formal methods – model checking. Available at <http://www.cs.cmu.edu/~modelcheck/>. [referenced 15.3.2000]
12. Somenzi, F., Higuchi, H., Lazy Group Sifting for Efficient State Traversal. IWLS'99, Granlibakken Resort, Lake Tahoe, USA, 1999
13. VIS. Available at <http://www-cad.eecs.berkeley.edu/research/vis/>. [referenced 15.3.2000]
14. Meinel, C., Wagner, A., Evaluation of OBDD-Heuristics via the Internet. IWLS'99, Granlibakken Resort, Lake Tahoe, USA, 1999