Increasing Efficiency of Symbolic Model Checking by Accelerating Dynamic Variable Reordering

Christoph Meinel, Christian Stangier[*]
FB IV - Informatik, Universität Trier
D-54286 Trier, Germany
email: meinel,stangier@uni-trier.de

Extended Abstract

Model checking has been proven to be a powerful tool in verification of sequential circuits, reactive systems, protocols, etc. The model checking of systems with huge state spaces is possible only if there is a very efficient representation of the model. Ordered Binary Decision Diagrams (OBDDs) allow an efficient *symbolic* representation of the model

Due to the huge number of operations applied to the OBDDs during model checking, the computation time is strongly related to the size of the OBDDs. As the order of the input variables has a strong influence on the size of the OBDDs, well suited variable orders have to be found. There exist a lot of heuristics for finding reasonable initial variable orders. In practice, techniques that improve the size of a given OBDD by changing the variable order dynamically during the computation have been proven to be more powerful (e. g. *Sifting* introduced by Rudell). For a survey to OBDDs and their algorithms see [2]. Such dynamic reordering strategies are especially useful for symbolic model checking, since the represented functions (e. g. reachable state sets) are changing during computation. As a consequence, the variable order has to be adapted to fulfil the new requirements.

OBDD based model checking tools like SMV use variable reordering techniques for the reduction of OBDD sizes. Indeed, variable reordering is strongly recommended since the huge amount of computations like iterations or fixpoint computations requires too much time or is impossible due to memory limitations, if the underlying OBDDs are too large. Also, the represented functions, like reachable state sets, are changing during computation and thus, the variable order has to be adapted to avoid explosion of OBDD sizes.

Recent research [3] has shown, that although sifting is the fastest common reordering technique, it is often too time consuming for an application during model checking. It emerges that a large fraction of the computation time is spent on sifting without a gain in OBDD size. Another fact is that sifting is to costly to be invoked as often as required if functions are changing.

Our goal is to accelerate the variable reordering process but retaining good OBDD sizes. To obtain this, we adapted two methods introduced by Meinel and Slobodova called **Block Restricted Sifting (BRS)** and **Sample Sifting** to the needs of model checking.

The idea behind **BRS** is to move the variables during reordering only within fixed blocks instead of moving them through the complete order. From theory it is known, that changing the variable order of a block does not affect the size of the other blocks.

The determination of the block boundaries follows from a communication complexity argument. A small information flow between two parts of an OBDD indicates a good candidate for a block boundary. If there is only little information flow between two blocks the distribution of variables to these blocks is well chosen. Improving the variable order inside the blocks will lead to a significant reduction of the OBDD size. The information flow is best indicated by the number of subfunctions that cross one level. With the aid of a *subfunction profile* we get easily computable structure information of the represented function.

The approach for block restricted sifting is the following:

1. Compute the subfunction profile,
2. detect the local minima of the profile,
3. specify the block boundaries,
4. sift within the blocks.

Block restricted sifting is well suited for symbolic model checking, since it substantially accelerates the reordering. This is possibly due to the observed fact, that the subfunction profiles have distinct “waists” that serve as boundaries.

Sampling is a common approach to optimization problems with huge search spaces. The idea behind the sampling strategy is to choose a relevant sample from the problem instance, to solve the optimization problem for the chosen subset and to apply the solution to the complete instance.

Applied to the problem of reordering BDD variables the

Sampling strategy can be described as follows:

1. Choose some OBDDs or subOBDDs from the common shared OBDD,
2. copy these BDDs to a different location,
3. reorder only the sample,
4. shuffle the variables of the original 0BDD to the new computed order of the sample.

Step 1 is the most critical during this sampling process. As mentioned above one should choose a relevant sample. If there is some knowledge about the represented functions, it can be used for the choice of samples. For example the OBDD representing the system's transition relation may often be a good choice for a sample. The chosen OBDDs should not be too small, so that as many variables of the representation as possible are contained in the sample. If there is no knowledge about the represented functions the sample may be chosen randomly from the single roots of the shared OBDDs.

The reordering (Step 3) can be done with any reordering technique. We used the standard sifting algorithm.

If the OBDD size increases after Step 4 the OBDD is reshuffled to the original order. One may repeat this process to get better results.

The main advantage of sample sifting is that the reordering of only a small fraction of the OBDD is significantly faster than reordering the whole OBDD. Also, the peak OBDD sizes of the sample during the reordering are smaller than the peak sizes during reordering the complete OBDD. Although there is more memory usage due to copying the sample (Step2), the overall memory consumption may be much smaller than compared to normal sifting.

Experiments using Block Restricted Sifting

We integrated the block restricted sifting strategy in the SMV tool (version 2.5). Each time sifting is invoked, block restricted sifting is performed instead. Furthermore, we had to take care about present and next state variables, because some OBDD routines of SMV rely on restricted positioning of the variables. Therefore, each next state variable is always placed directly beside its corresponding present state variable and never in another block.

There are several parameters that may be adjusted:

– The *minimum fraction* of the number of variables, that a block must contain,
– the *gradient* of the levels beside a local minimum,
– an *overlapping* of blocks may be specified, too.

For an experimental basis we used the benchmark suite of Yang and the examples that come with SMV.

It emerged that for symbolic model checking larger block sizes are required than for combinational verification, i. e. up to 25% instead of 10%. In most cases an overlapping of 2 variables was sufficient.

We also performed experiments, where the block restricted sifting is invoked earlier. This resulted in most cases to better OBDD sizes without requiring more computation time for the reordering.

We were able to improve the computation time for every benchmark model and for any parameter setting without increasing the OBDD sizes to much.

Experiments using Sample Sifting

For sample sifting there are different methods to choose a sample:

TR+RS Choose the OBDDs for the transition relation and the reachable state set.

MAXBDD Choose the two largest OBDDs.

FRAC Choose OBDDs with a given minimum size randomly and copy samples until a fraction $frac$ of the total OBDD size is reached.

All these methods can be combined. We also provide different methods for copying fractions of OBDDs using different depth first styles: Copy a node when *backtracking* or *visiting* until $frac$ is reached. The first method results in a subOBDD of the original function, i. e. the lower part of the OBDD. The second method leaves unvisited edges that have to be set to the 1- or 0-sink after termination. Therefore, it results in a modified but closely related function to the original function. In the case of symbolic model checking sample sifting requires larger samples than for combinational verification.

We were able to improve the computation time for every benchmark model. But, in contrast to BRS the correct choice of parameters, i. e. the sampling type is of high importance.

We are now implementing the possibility to change the sampling type during the computation. For the transition relation (TR) as a sample type we did already apply this technique with success. There, if the TR is too small we simply switch the sample type to MAXBDD.

For further reading see [1]. There is also the possibility to test our heuristics online at:
`www.informatik.uni-trier.de/TI/OHO.html`

References

[1] C. Meinel and C. Stangier. Increasing Efficiency of Symbolic Model Checking by Accelerating Dynamic Variable Reordering. Technical Report 98-23, University of Trier, Department of Computer Science, 1998.

[2] C. Meinel and T. Theobald. *Algorithms and Datastructures in VLSI-Design*. Springer, 1998.

[3] B. Yang, R. E. Bryant, D. R. O'Hallaron, et al. A Performance Study of BDD-based Model Checking. In *Proc. of 2nd Intl. Conf. on Formal Methods in CAD*, pages 255–289, 1998.