Chapter 7

Conclusion

This thesis presents two major theoretical contributions:

1. Techniques were developed for the parameterization of Boolean functions, so that, given a vector function, we can generate an alternative more compact representation that spans the same range as the original one, and

2. Novel contributions were made to the theory of disjoint support decomposition of Boolean functions by providing a new canonical form to represent the unique maximal disjoint decomposition of a logic function and a novel and efficient algorithm that can automatically discover this decomposition in polynomial time.

Both of these theoretical contributions were applied to the problem of formal verification of digital systems and two new techniques were developed for the use symbolic simulation in verifying digital systems. These approaches expand the robustness and scalability of symbolic simulation and expand its accessibility for adoption in current industrial design practices.

7.1 Parameterized approaches in symbolic simulation

Parametric representations help to control BDD explosion in symbolic simulation, which is the central problem in formal verification. Based on our parameterization techniques, we have shown simulation results on industrial design blocks that show the significant performance gains that these techniques can achieve both over logic simulation and symbolic simulation. In particular we have shown how CBSS (Cycle-Based Symbolic Simulation) provides many orders of magnitude better performance, measured in test vectors simulated per second, over logic simulation. At the same time, CBSS does not require any change in the verification user model, and thus can replace logic simulation in a transparent fashion. The same checks, or assertions, that are used to verify the outputs of logic simulation – see Section 1.1 – can be used in the context of CBSS to verify the expressions for the output of the digital system. DSD-SS (Disjoint-Support Decomposition based Symbolic Simulation) provides an exact parameterization technique, and we have shown that for a fixed amount of memory resource it can explore a much broader search space compared to a pure symbolic simulation approach. DSD-SS also provides a significant performance gain over logic simulation.

The two new solutions presented can be viewed as trade-off points in a search breadth vs. simulation scalability plane as indicated qualitatively in Figure 7.1: logic and symbolic simulations are at the extremes of scalability and search breadth, while CBSS and DSD-SS provide additional trade-off points between the two parameters. A major advantage of DSD-SS is that the memory resources required for the simulation can be traded off for reduced search breadth.

7.2 Disjoint support decompositions

Disjoint support decompositions (DSD) are a useful property of Boolean functions that can be used in many areas of computer-aided design automation. The ability to partition a function into blocks that depend only on a small portion of the support set of the function is valuable for decreasing the computational complexity of the function. Applications of DSD to the synthesis domain span from the routing arena, where the decomposition suggests the clustering of input signals that only affect a portion of the circuit, to multi-level synthesis, where it provides an automatic way of moving from a flat representation to a hierarchical one based on the decomposition tree. In this thesis we explored

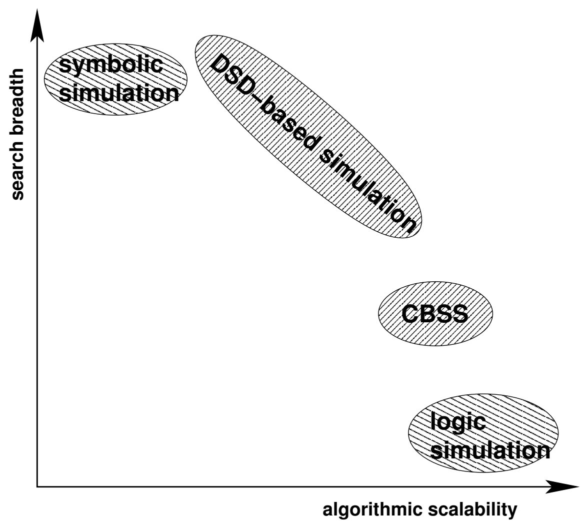

Figure 7.1: Trade-offs of in the breadth vs. scalability plane

some aspects of the benefits that decompositions can provide in verification and, in particular, simulation. Here, we exploited the fact that DSD exposes the inherently parallel components in the computation of a function and we used this fact to generate a simplified function. DSDs can also be used to generate a good initial variable order for the BDD of a function, by clustering inputs togheter that affect the same blocks.

In our experiments with the algorithm we introduced, we found that most functions, at least among the ones related to industrial designs, have meaningful decompositions, which is a promising starting point for the possible applications we have outlined and others that we have not thought about.

7.3 The future of this work

This work, as with most research, suggests future directions of research. From a practical standpoint, the efficiency of the implementations can be considerably improved. Such improvements would not only increase the robustness of the algorithms, but also provide even better results when comparing the performance to other simulators. Currently, the front-end of the simulators is limited to a simple logic intermediate language. Supporting standard hardware description languages would increase the ease of experimenting with current industrial designs and provide better insights on the strengths and weaknesses of our techniques.

The robustness of the parameterization techniques introduced could only be evaluated accurately by building a usage framework around the raw simulations engines we have, so that a user can provide assertions, or specify correct behaviour for the design under simulation, and available test vectors to direct the search when the simulator needs to approximate.

Parameterization is a general technique that can be applied in other phases of the simulation flow. We would like to explore the additional robustness that we could obtain by deploying these or other parameterizations in the simulation of the combinational network in order to reduce the memory resources required there.

The application of the theory of disjoint support decompositions to other areas of design automation such as the ones indicated above is a direction that promises to lead to a broad range of problems and solutions. The few cases where DSD can not break a function into sufficiently small components suggest that there are more complex types of decomposition for which a canonical form has yet to be defined.

In a broader spectrum, the verification of digital designs is becoming the hardest problem in design automation and the immediate bottleneck to remove in order to maintain the growth trends that the IC industry has seen in the past 40 years. Current verification methods are struggling to handle the complexity of a single component module in a digital system on a chip design. We believe that the answer to this problem will come from two directions: improved scalability of verification algorithms on one end, and a modular verification methodology on the other. Parameterization techniques are one way of improving such scalability; however, we can se how, even only within the symbolic simulation approach, other techniques are needed to improve scalability and robustness during the whole verification flow.