Boolean Satisfiability Strategies to Optimize Logic Circuit Design

Sami Rahim
Henry M. Gunn High School

INTRODUCTION

The effective tool of satisfiability is derived from the laws of mathematical logic. When logic variables are combined together using logic operators, the result is a large statement which ultimately ends in one of two outputs (true or false) depending on the assignment of each variable. Satisfiability is the process of determining if the entire statement can be true — if a given statement can be true for any assignment of its variables, it is deemed satisfiable (Boolos & Purgess, 2007). Algorithms have been created that can determine if any logic statement is satisfiable or not — these are called SAT solvers.

The tool of satisfiability can be applied to decision problems and optimization. Nearly fifty years ago, it was shown that any instance of a decision problem P in NP (which denotes the complexity of the problem) can be encoded into a SAT formula that can determine the validity of the instance (Cook 1971) (Karp 1972).

By creating a Boolean equivalent to an instance of the optimization (that is, where a value has been assigned to the objective, Z), multiple instances can be run in a SAT Solver until the largest/smallest value for Z has been identified such that the statement is satisfiable (Cook, 1971). We can describe the optimization model as **Φ** where:

$$\Phi = < Vars, Feasible, Obj >$$
$$Vars = \{p_1, \dots, p_n\}$$
$$Feasible(p_1, \dots, p_n) = \{True, False\}$$
$$Obj(p_1, \dots, p_n) = \mathbb{Q}$$

An instance **q** of the optimization can be determined as the solution to the optimization (using SAT) if:

$$SAT(Feasible(\boldsymbol{q}p_1, \dots, \boldsymbol{q}p_n) \ \wedge \ SAT(Obj(\boldsymbol{q}p_1, \dots, \boldsymbol{q}p_n) \leq all\ Obj(\Phi)$$

DATA AND FINDINGS

The Berkeley Logic Interchange Format (BLIF) model is input in to a SAT Solver and the following commands derive the complexity:

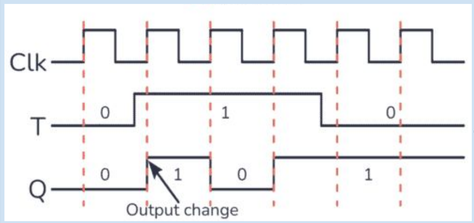
“
read_blif [file name].blif
strash
print_stats
i/o: 384/1 lat = 0 and = 58055 lev = 135
”

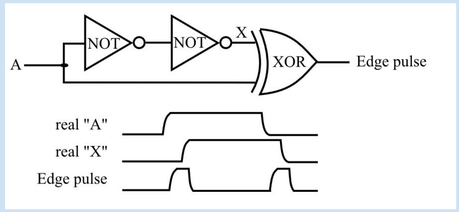
Here, the command “fraig” is run, which detects structurally different logic that carries equivalent value.

“
fraig
print_stats
i/o: 384/1 lat = 0 and = 56963 lev = 131
”

	Raw Model	Optimized Model
Format	BLIF	BLIF
levels	135	131
AND gates	58055	56963

In circuit design, a clock can be used to detect changes in logic output based on the flow of electricity.





To actually implement this type of redundancy, SAT can be applied. We can use SAT to detect equivalences in logic. For example, given we take inputs to 2 clocks, q and p; the SAT solver can test for the satisfiability of this following logic statement:
~((q AND p) OR (~q AND ~p))

Two clocks can be determined redundant if they meet both:

1. The input contains the same logic structure
2. The edge pulse is identical based on the truth value of the input

When these criteria are satisfied, the inputs become virtually indistinguishable, and the resulting behavior is correspondingly consistent.

IMPLICATIONS & NEXT STEPS

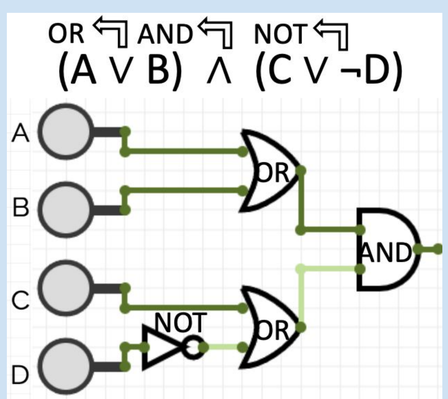
The implications of these findings directly affect the circuit design industry. The ability to optimize circuits by detecting redundancies and reducing the number of logic gates can lead to more efficient and cost-effective chip designs, which is vital for the development of technology. While past research has been conducted to optimize chip design, this research extends those findings by focusing on edge scenarios, particularly with clock circuits, showing an outline for redundancy removal. This work is the first step in a fully implemented circuit design optimization technique.
In the future, removing redundancies of edge circuits should implement SAT solvers to remove the redundancies and improve the efficiency of circuit designs.

RESEARCH METHODOLOGY

Logic circuit design concerns the manipulation of binary values abstracted to bits using logic gates to output signals based on inputs. These binary variables and their manipulation through logic gates concerns mathematical logic. Satisfiability (SAT) is a decision procedure that concerns logic statements; mathematical logic is a form of algebra essential in computing.

Model Name	Boolean Satisfiability
Problem Type	Decision problem (determine if a solution exists)
Variables	Boolean values (true/false, 1/0)
Formation	•One Boolean statement made of variables connected with AND, OR, and NOT logic gates •given a set of assignments to each variable, the entire statement will have a truth value (true/false, 1/0)
Solving approach	SAT Solver, “ABC” program
Solution	Outputs satisfiable/unsatisfiable depending on if there exists an assignment to each variable that renders the entire statement “true”
Example	Figure seen on right

The connection between these two concepts can be used to address the optimization of circuit design.



CONCLUSIONS AND ANALYSIS

This research reveals several important insights into the optimization of logic circuits and the application of SAT solvers for detecting redundancies in circuit design. Specifically, the use of BLIF (Berkeley Logic Interchange Format) models demonstrated that SAT for optimization could significantly reduce complexity, as evidenced by the 2.1% reduction in AND gates by SAT solver optimization commands such as the “fraig” technique.
By testing and optimizing the model, we discovered that using a SAT solver not only helped in identifying redundancies but also improved the model’s performance by reducing the number of redundant gates and levels in the circuit.
Most importantly, a potential for redundancy removal was discovered through edge logic circuits, and the use of SAT solvers for this strategy were found.
These findings can contribute to circuit design and optimization. The results also emphasize the role of Boolean satisfiability in the world of circuit design.

REFERENCES

Bacchus, F. (Presenter). (2021, April 1). SAT for Optimization. Lecture presented at 50 Years of Satisfiability: The Centrality of SAT in the Theory of Computing, . ources/tech-talks/sat-and-smt-solvers/
Cook, S. (n.d.). The complexity of theorem-proving procedures. STOC '71: Proceedings of the third annual ACM symposium on Theory of computing. https://dl.acm.org/doi/10.1145/800157.805047 https://personal.utdallas.edu/~dxd056000/cs6363/LectureNotes.pdf
50 Years of Satisfiability: The Centrality of SAT in the Theory of Computing. (n.d.).
Karloff, H. (n.d.). Linear Programming. https://books.google.com/books?hl=en&lr=&id=ke1FAAAAQBAJ&oi=fnd&pg=PR5&dq=linear+programming&ots=IeLxn47bYy&sig=fyQxz5OBvCJ_1Oc3vcuM56Jh7_A#v=onepage&q=linear%20programming&f=false
Karp, R. (n.d.). Reducibility among combinatorial problems. Complexity of Computer Computations. https://www.cs.purdue.edu/homes/hosking/197/canon/karp.pdf
Mileti, J. (n.d.). A Mathematical Introduction to Mathematical Logic. https://mileti.math.grinnell.edu/MathematicalLogic.pdf
Narodytska, N. (Presenter). (n.d.). SAT-based Techniques for Optimization and Synthesis [Lecture].
Stanford University. (n.d.). Digital Circuits. Introduction to Logic. http://intrologic.stanford.edu/extras/circuits.html