An SMT-based Approach to Optimization on Pseudo-Boolean Functions and Its Application to Power Estimation

Yongjian Li[1], William N. N. Hung[2], and Xiaoyu Song[3]

No Institute Given

Abstract. Solvers for SAT Modulo Theories (SMT) can nowadays handle large industrial (verification problems over theories such as the integers, arrays, or equality. This paper addresses an SMT-based approach to handle Pesudo-Boolean (abbr. PB) optimization problems and its application to the problem of power estimation of combinational circuits. There are two key points in our approach. First we formulate as an SMT problem whether an objective function of a PB function is greater or equal to an integer value. Second, we try to find the greatest value which satisfies the above requirement. Furthermore, we apply our method to the problem of power estimation. The experiments on typical benchmark circuits show that our approach can handle moderate scale of circuits with a precise solution. Our work demonstrates the feasibility of solving some difficult problems in the field of optimization on PB functions with the help of the most advanced SMT-solver such as Z3.

1 Introduction

In mathematics, a pseudo-Boolean function is a function of the form $f : \mathbb{B}^n \to \mathbb{R}$, where $\mathbb{B} = \{0, 1\}$ is Boolean domain and n is a nonnegative integer. $\mathbb{R}$ is Real domain [**?**]. Pseudo-Boolean functions play a major role in optimization models in a variety of areas, including operation research (abbrev. OR) like scheduling , sequencing, and time tabling) [**?**], statistical mechanics (spin glasses [**?**]), computer science (maximum satisfiability [**?**,**?**]), etc.

Due to the existence of factors in Integer (or Real) domain and the operators on these domains such as addition and multiplication, the PB-relating problems are at first-order (or term) level, not Boolean level. Therefore, methods basing on Boolean domain such as SAT or BDD cannot directly be applied to PB-related problems. In contrast to SAT (or BDD), *Satisfiability Modulo Theories* (SMT) is concerened with the satisfiability of first-order class of formulas with respect to some background theory. A natural question arises whether the PB-relating problems can be formulated in the framework of SMT and SMT solver can be used for the solution of a PB-relating problem?

In the past, the major concerns of the VLSI designer were area, performance, cost and reliability; power considerations were mostly of only secondary importance. However, this situation has begun to change increasingly in recent years, and power is being given comparable weight to factors such as area and speed. Perhaps the primary driving factor has been the remarkable success and growth of the class of mobile computing devices (portable notebooks, Tablet PC, smart mobile-phone) which demand high-speed computation and complex functionality with low power consumption [**?**,**?**].

Two sources of power dissipation in CMOS combinational circuits are as follows [**?**,**?**,**?**]: dynamic power dissipation due to switching activity that takes place during charging and discharging of the capacitors, and static power dissipation due to leakage current.

Dynamic power dissipation due to the switching current from charging and discharging capacitances can be a major contributor to the total power dissipated in the circuit. To estimate the maximum switching activity that a circuit might experience, it is necessary to search for a vector pair $< \overline{V_1}, \overline{V_2} >$ that tends to maximize this activity. Therefore the complexity is the number of all possible two input vectors, which is $O(4^n)$, where n is the number of primary inputs. It has been established that finding this vector pair is an NP-complete problem.

At the same time, due to the continuing decrease in feature size and increase in chip density, static power dissipation due to leakage current has become a major component of the overall power dissipated by a circuit. However, unlike the maximum switching current discussed earlier which depends on two-input vectors, the maximum wake-up current depends only on one input vector [16]. Hence, to estimate the maximum instantaneous power-up current one need to search for an input vector $\overline{V}$ that maximizes the wake-up activity of the circuit. So the complexity of the estimation of the leakage power is $O(2^n)$.

The above two power estimation problems can be naturally formulated in the framework of optimization on Pseudo-Boolean functions. Namely, we can use two kinds of Pseudo-Boolean functions to represent the two kinds of power dissipation, thus the peak values of the power dissipation can be boiled down to computing the maximal value of the corresponding formulated Pseudo-Boolean functions. In this work, we deal with the more difficult problem: dynamic power dissipation.

Related work There has been a lot of papers on Pseudo-Boolean functions or power estimation in the research filed of Mathematics, Operational Research, and Computer Science [**?**,**?**,**?**,**?**,**?**,**?**]. Among these, we think, the work in[**?**] is more closely related to ours.

In operational research, the problem on Pseudo-Boolean functions are typically solved with methods based on linear programming . Following this way, we can take advantage of the modern LP solver such as CPLEX [**?**].

Evolutional algorithms and local search-based algorithms are also be used to try to solve the constraint problems on Pseudo-Boolean functions. Methods for the analysis of evolutionary algorithms on Pseudo-Boolean functions is discussed in [**?**,**?**].

With the fast development of modern BDD and SAT tools, more work attempt to use the tools mentioned above to solve these problems. Several of these encodings are based on building Binary Decision Diagrams (BDDs), and it is argued that BDD-based encodings have some advantages, such as sharing the same BDD for representing many constraints. Recently, SAT solvers have been extended to handle Pseudo-Boolean(PB) constraints [?], which are simple inequalities that are equivalent to 0C1 integer linear programming (ILP) constraints. PB constraints are more expressive and can replace an exponential number of CNF constraints.

In [?], a variant of SMT is introduced where the theory T becomes progressively stronger, and its correctness is proved by using the Abstract DPLL Modulo Theories framework. Two different examples of applications of this SMT variant are tried: weighted Max-SAT and weighted Max-SMT.

In[?], Devadas et al. formulated the power dissipation of CMOS circuits as a Boolean function in term of the primary inputs. They attempted to maximize the function by solving a weighted max-satisfiability problem using exact and approximate algorithms. The technique is proved to be only applicable only to small circuits. Leakage power, which also contributes to the total power consumed in a CMOS circuit in addition to the dynamic or switching activity power, was discussed in[?]. In their work, Aloul et al. used a SAT-based approach to determine the minimum or maximum leakage state of a CMOS combinational circuit. The work in[?] demonstrates the validity of using advanced ILP solvers, SAT-based and generic-based, in addressing the above two estimation problems. His experiments show that generic ILP solvers are likely to achieve better results than SAT-based tools. ATPG-based techniques are also adopted to power estimations in [?].

Our motivation and Contribution The main motivation of this work is to adapt the problem of optimization on pseudo-Boolean functions to SMT. The intuition behind this motivation is very clear: SMT should provide a solution if the above optimization problem can be boiled down to the satisfiability of first-order class of formulas, since SMT is concerned with the satisfiability of first-order class of formulas.

The main contributions of this paper are twofold. The first one is to develop a general framework to adapt a SMT-slover to solve the the optimization problem on pseudo-Boolean functions. First we formulate as an SMT problem whether an objective function of a PB function is greater or equal to an integer value. Second, after we compute a low and upper bound of the PB function, we try to find the precise value which satisfies the above requirement. The second is to try an empirical experiment to demonstrate the feasibility of our approach. Our choice of case studies is one of the aforementioned power estimation problems: dynamic power estimation. A PB function is derived from a circuit netlist structure according to the power estimation problem under consideration. Then the above SMT-based approach is applied to compute the maximal value of the PB function, which is corresponding to the maximal power consumption.

The remainder of this paper is organized as follows: Section 2 provides a preliminary introduction for the formulation of problems of of pseudo-Boolean optimization, SMT, and power estimation. Section **??** provides an overview of of our approach, proves its correctness, and discusses its computation complexity. Section **??** illustrates the details to implement our approach for the estimation of dynamic power dissipation. Section **??** shows our experiment results when we apply our method to the typical benchmark circuits. Section **??** concludes the paper.

2 Preliminary

2.1 Problem Formulation

In this work, we focus on a specific pseudo-Boolean (PB) constraint problem as follows: consider a Boolean functions $f_i\ x_1\ x_2\ ...\ x_n$ on binary variables x_1, x_2, ..., x_n, where $0 < i < m$. Our task to solve this PB problem is concerned with finding an assignment to the variables to make a given objective function as follows to be maximal,

$$\sum_{i<m} w_i \cdot f_i\ x_1\ x_2\ ...\ x_n \tag{1}$$

where W_i is a natural number which is greater than 0.

For instance, let x_1, x_2, x_3 be variables, $f_1\ x_1\ x_2\ x_3 = \neg x_1$, $f_2\ x_1\ x_2\ x_3 = \neg(x_1\ \wedge\ x_2)$, $f_3\ x_1\ x_2\ x_3 = \neg x_1 \vee (\neg(x_1\ \wedge\ x_2))$, and $f_4\ x_1\ x_2\ x_3 = \neg(x_3 \wedge (\neg(x_1\ \wedge\ x_2)))$; $w_1 = 1$, $w_2 = 2$, $w_3 = 1$ and $w_4 = 1$. The assignment $\{x_1 \mapsto 0, x_2 \mapsto 1, x_3 \mapsto 0\}$ is an assignment which makes all f_i to be 1, thus the value of the objective function is the maximal one: 5.

Obviously, the problem of the above PB problem is NP-hard, and its complexity is $m * 2^n$. It is difficult to compute the maximal value of the objective function when n becomes large.

2.2 SAT Modulo Theories

In computer science and mathematical logic, an SMT problem is a decision problem for logical formulas with respect to combinations of background theories expressed in classical first-order logic with equality. Examples of theories typically used in computer science are the theory of real numbers, the theory of integers, and the theories of various data structures such as lists, arrays, bit vectors and so on. SMT can be thought of as a form of the constraint satisfaction problem and thus a certain formalized approach to constraint programming.

Formally speaking, an SMT instance is a formula in first-order logic, where some function and predicate symbols have their interpretations, and an SMT-solver such as Z3 is called to determine whether such a formula is satisfiable. If thus, a model is also returned by the SMT-solver to give an interpretation that makes the formula true.

2.3 Estimation of Maximal Dynamic Power Dissipation

Now we firstly introduce the dynamic power dissipation for combinational circuits G under study. Here we simply regard such a circuit as a set of gates. There are three simplifying assumptions.

- The only capacitance in a CMOS logic gate is at the output node of the gate.
- Either current is owing through some path from V_{DD} to the output capacitor or current is owing from the output capacitor to ground.
- Any change in a logic gate output voltage is a change from V_{DD} to ground or vice versa.
- For a circuit under study, the input vectors to the primary inputs of the circuit is applied at the start of each cycle. The circuit is stabilized at the start of each cycle but the first cycle.

Formally, the relation between the logical behavior of a COMS combinational gate g_i whose logic function on primary input variables $\overline{x}$ and the power dispatched under a state transition which is caused by a change of the values of the primary inputs from $\overline{V_1}$ to $\overline{V_2}$ is formalized as the following equation:

$$P_{g_i} = 0.5 \cdot C_i \cdot \frac{V_{dd}^2}{T_{cycle}} \cdot (f_i\ \overline{V_1} \oplus f_i\ \overline{V_2}) \tag{2}$$

where P_{g_i} is the power energy dispatched by the CMOS gate, C_i is the out capacitance for the gate, V_{dd}^2 is the supply voltage of the power source, and $1/T_{cycle}$ is the number of gate output transitions per clock cycle. Here f_i is the Boolean function being performed by the gate, and $\overline{V_1}$ and $\overline{V_2}$ are two input vectors to the primary inputs $\overline{x}$ of the gate g_i.

Usually the capacitance C_i is assumed to be directly propositional to the fan out number W_i of the gate. Therefore, from the equation above and the assumptions made so far, the product $W_i \cdot (f_i\ \overline{V_1} \oplus f_i\ \overline{V_2})$ is also directly propositional to P_{g_i}. In order to maximize dynamic power dissipation, we need to search for an input vector pair $< \overline{V_1}, \overline{V_2} >$ such that trends to maximize the sum of the weighted products:

$$\sum_{g_i \in G} W_i \cdot (f_i\ \overline{V_1} \oplus f_i\ \overline{V_2}) \tag{3}$$

In essence, the estimation of dynamic power dissipation is the estimation of the weighted switching activity in the circuit under study after the stimulus of the value changing of primary inputs. The problem of computing the maximal value of (3) is equivalent to finding the maximal value of the objective function:

$$\sum_{i<m} W_i \cdot f_i'\ \overline{x}\ \overline{y} \tag{4}$$

where $f_i'\ \overline{x}\ \overline{y} = (f_i\ \overline{x} \oplus f_i\ \overline{y})$.

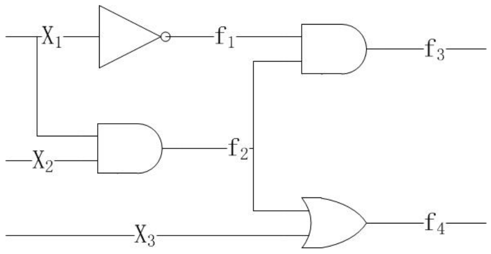

Fig. 1. An example circuit for estimation of the weighted switching activity

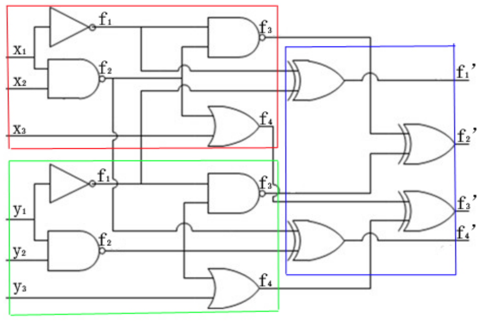

Fig. 2. The derived circuit for estimation of the weighted switching activity of the original circuit in Fig. 1

Thus, we can transform the problem of identifying two-vector pair sequence $(\overline{V_1}, \overline{V_2})$ that produce weighted maximum switching activity in the circuit under study into identifying an input vector $(\overline{V_1 V_2})$ that maximizes weighted gate output activity of another transformed circuit, which is composed of two copies of the former circuit by connecting each non-input nodes by a XOR gate. For instance, if we want to compute the weighted switching activity of a circuit in Fig 1, which is formulated in Equation (3), we only need compute the the maximal value of the objective function which is determined by logical functions of another circuit which is shown in Fig 2, which is formulated by Equation(4).

The correspondence of fig 1 and 2 is as obvious: circuit 2 is composed of two copies of that in 1. We use a XOR gate to connect one output of a gate in one copy and the corresponding one in the another copy; and each of the XOR gate is associated with a weight which is the fanout of the original gate whose output is the input of the XOR gate. The objective function consists of the weighted sum of all the XOR outputs.

The objective function to estimate the weighted switching activity of the circuit in fig 1 is as follows:

$$\begin{array}{ll} f_1\ \overline{x} = \neg x_1 & W_1 = 1 \\ f_2\ \overline{x} = \neg(x_1\ \wedge\ x_2) & W_2 = 2 \\ f_3\ \overline{x} = \neg(\neg x_1 \wedge (\neg(x_1\ \wedge\ x_2))) & W_3 = 1 \\ f_4\ x_1\ x_2\ x_3 = (\neg(x_1\ \wedge\ x_2)) \vee x_3 & W_4 = 1 \end{array}$$

$$\sum_{i<m} W_i \cdot f'_i\ \overline{x}\ \overline{y}$$

where $\overline{x} = x_1\ x_2\ ...\ x_n$, $\overline{y} = y_1\ y_2\ ...\ y_n$, and $f'_i\ \overline{x}\ \overline{y} = (f_i\ \overline{x} \oplus f_i\ \overline{y})$.

3 An Overview of an SMT-based Approach to Optimization on Pseudo-Boolean Functions

As we have mentioned in Section 2, an SMT problem is on the satisfiability of a first-order formula. Therefore, the first step we do is to formulate a first-order formula that the maximal value of the PB function is greater or equal to a bound value *val*. In detail, the formulated SMT problem IsGeN *val* is as follows:

- The definition of variables used in the PB function;
- The definition of the PB function by a list of constraints, which are usually derived from the application domain. In our case, a netlist information is used to derive the constraints;
- An assertion to specify whether the PB function is greater or equal to a bound value *val*.
- A command to ask the SMT-solver to find a model for the formula.

An overview of our SMT-based Approach to Optimization on Pseudo-Boolean Functions is shown in Fig. **??**.

In Fig. **??**, function check n will play a key role. An SMT problem IsGeN n will be created, then an SMT-solver is called to find whether IsGeN n can be satisfied. The SMT problem IsGeN n can either defined in a text (e.g. SMT-LIB2 style) or a programmatic (e.g. C, PYTHON, SCALA) format. In our work, we use a text format.

In order to find a precise value of the maximal value of a PB function, we need an upper and lower bound for the PB function. The upper bound Hi can be the value by assuming all the f_i true, and the lower bound Lo a value that is computed by some heuristic methods such as Hill Climbing or Genetic algorithms. Here we can regard the the lower and upper bound as two parameters and they are used to initialize lo and Hi respectively. We first check whether a satisfying model exists if we set the object function is not less than the value $Mid = (Hi + Lo)/2$ by calling IsGeN Mid; if a model is found, we set the current value of Lo to be Mid; otherwise we tune the current value of Hi value to be Mid. This process is repeated until the condition where $Hi = Lo + 1$. If check Hi, then Hi is returned; otherwise Lo is returned.

Now we argue the correctness of the above approach. First check $Mid = true$ implies that the maximal value of the PB function is greater or equal to Mid;

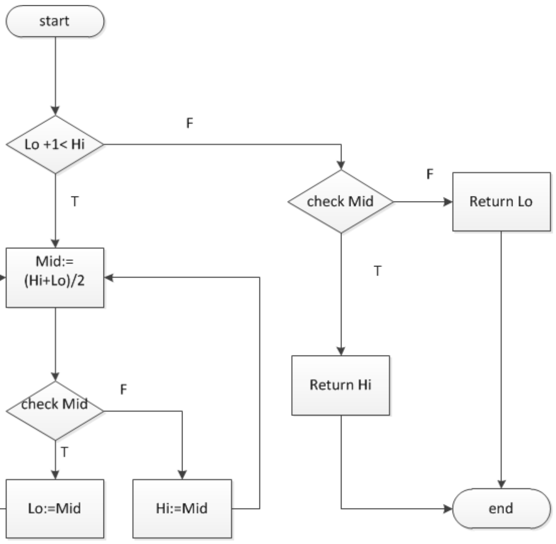

Fig. 3. An overview of our SMT-based Approach

therefore we need improve the lower bound by setting $Lo = Mid$; otherwise this means that the maximal value of the PB function is less than val, therefore we need decrease the upper bound by setting $Hi = Mid$. After the iterating steps is stopped, this means the last value of the maximal value of PB can neither decreased nor decreased. Namely, the last value returned is the precise value of the maximal value of the PB function.

Roughly speaking, the total time of the above procedure is at most $log_2^{upper-low} \times t$, where *upper* is the upper bound, *low* the low bound which we initially set. t is the time used to call IsGeN Mid once.

4 Implementation of an SMT-based Approach to Estimation of Dynamic Power Dissipation

In this section, we will illustrate some implementation details of the SMT-based approach to estimation of dynamic power dissipation.

As we have shown in Fig. **??**, the key is to implement the function check n. There are three tasks in check n.

1. generation of the SMT problem IsGeN n. In our case, we written the problem in a file with a suffix ".smt". This file complies with SMT-LIB2 standard. An example of this file is shown in Fig. **??**.
2. calling SMT-solver to solve the aforementioned SMT problem. In our case, we just use a system call to Z3 to solve this question. The result returned by Z3 is also written into a file.

3. analyzing the returned result by the SMT-solver. In our case, we just only need analyze the contents output by the SMT-solver.

```
0(declare − const x1 (_ BitVec 1))
1(declare − const x2 (_ BitVec 1))
2(declare − const x3 (_ BitVec 1))
3(declare − const y1 (_ BitVec 1))
4(declare − const y2 (_ BitVec 1))
5(declare − const y3 (_ BitVec 1))
6(define − fun fx1 () (_ BitVec 1)  (bvnot x1))
7(define − fun fx2 () (_ BitVec 1)  (bvnot (bvand x1 x2)))
8(define − fun fx3 () (_ BitVec 1)  (bvnot (bvand fx1 fx2)))
9(define − fun fx4 () (_ BitVec 1)  (bvnot (bvand (bvnot x3) (bvnot fx2))))
10(define − fun fy1 () (_ BitVec 1)  (bvnot y1))
11(define − fun fy2 () (_ BitVec 1)  (bvnot (bvand y1 y2)))
12(define − fun fy3 () (_ BitVec 1)  (bvnot (bvand fy1 fy2)))
13(define − fun fy4 () (_ BitVec 1)  (bvnot (bvand (bvnot y3) (bvnot fy2))))
14(define − fun f1' () (_ BitVec 1)
 (bvnot (bvand (bvnot (bvand fy1 (bvnot fx1))) (bvnot (bvand fx1 (bvnot fy1))))))
15(define − fun f2' () (_ BitVec 1)
 (bvnot (bvand (bvnot (bvand fy2 (bvnot fx2))) (bvnot (bvand fx2 (bvnot fy2))))))
16(define − fun f3' () (_ BitVec 1)
 (bvnot (bvand (bvnot (bvand fy3 (bvnot fx3))) (bvnot (bvand fx3 (bvnot fy3))))))
17(define − fun f4' () (_ BitVec 1)
 (bvnot (bvand (bvnot (bvand fy4 (bvnot fx4))) (bvnot (bvand fx4 (bvnot fy4))))))
18(define − fun objFun ()  (_ BitVec 3)
19(bvadd (bvmul (_ bv1 3) (concat (_ bv0 2) f1') )
20 (bvadd (bvmul (_ bv2 3) (concat (_ bv0 2) f2') )
21 (bvadd (bvmul (_ bv1 3) (concat (_ bv0 2) f3') )
22 (bvadd (bvmul (_ bv1 3) (concat (_ bv0 2) f4') )
23 (_ bv0 3)))))
24(assert (bvuge objFun (_ bv5 3)))
25(check − sat)
26(get − model)
```

Fig. 4. Formally Modelling Power Estimation Problem in SMT

The SMT file is divided into five parts:

(1) Lines 0-6 defines each input as a bit, or a bit vector with width 1;

(2) Lines 6-17 specifies logical functions of all the gates. The logical function of the output node of the gate is specified as a bit-vector function of the input nodes of the gate. These definitions can also be seen as the constraints of the PB problem.

(3) Lines 18-23 defines that the PB function is the sum of weighted outputs of some functions. Here we must estimate the width by the largest value of the weighted outputs. In this case, we need 3 bits to encode because the largest value of the PB function is 5.
(4) Lines 24 asserts that the PB function should be greater than or equal to 3.
(5) Lines 25 and 26 calls the SMT solver to check whether the above specification is satisfiable, and find a model (or an assignment to the inputs) if the specification is satisfiable.

Here we emphasize that we use the type of bit vector to define the object function. Before we adopt the bit vector to define the objective function, we also have tried to use type int to directly define it. However, this strategy is less efficient than that adopting the bit vector theory.

Task 1 in the function check n can be automated by analyzing the structure of the derived circuit which is shown in Fig. 2. In our experiments, we implement this task in Forte, which provides strong supports to analyzing structures of netlists in EXLIF. Here we also notice that the difference between IsGeN n and IsGeN n' in SMT-LIB 2.0 format only lies the assertions on the value of PB function, which is shown in Line 24 in Fig. **??**. Therefore, we can analyzes the structure of derived circuit netlist in the preprocessing step, and save the contents from Parts (1)-(3) in a template file. When the SMT file of IsGeN n is created, contents of Parts (1)-(3) can be directly copied from the template file, then the assertions on the value of the PB function and Part (5) can be appended. The redundant process of the analyzing the structure of the netlist can be done only once.

The implementation of the work flow in Fig. **??** by Forte is shown in Fig. **??**.

In our case, our function compMaxPower *ckt n upper lower* is the main body to compute the maximal dynamic power consumption. duplicate *ckt* is used to create the derived circuit which is shown in Fig. 2 from the original circuit which is shown in Fig. 1. creatTemplate generates the aforementioned template of the SMT file, and compute the possibly maximal value $maxPossible$ of PB function. neededWidth *ckt* computes the width of the bit vector needed to encode $maxPossible$. The function binary_chop implements the aforementioned binary-chopping strategy. Here a function fn is a functor parameter, which checks whether the searching procedure is up or down by the number *mid*.

Function decide copies the aforementioned template file, adds an assertion to specify that the PB function is greater or equal to a bound value, then generate the SMT problem Is_Gen n in a file, calls the SMT solver z3 to decide whether a model exists for the assertion, then checks the result returned by the SMT-solver z3. In fact check n is implemented by decide *template wid n*.

5 Experiments

We have done experiments to implement the aforementioned approaches based on the FORTE working tool developed in the INTEL[**?**], which uses symbolic trajectory evaluation (abbrev. STE) for hardware verification. We mainly use

```
letrec binary_chop fn Hi Lo   =
  Lo+1 < Hi =>
     let Mid = Lo + (Hi-Lo)/2 then
     fn Mid => binary_chop fn Hi Mid
     | binary_chop fn  (Mid - 1) Lo
  |fn Hi=>Hi|Lo;

let decide template wid n=
  let fin = fopen (template^".smt.temp") "r" then
  let fout = fopen ("Is_Gen "^(int2str n)^".smt") "w" then
  rec_copy fin fout fseq
    defAssertOnObj wid fout n fseq
   fprintf fout "(check-sat)" fseq
   fprintf fout "(get-model)" fseq
   fclose fout fseq
   let cmd=(sprintf "z3 %s.smt > %s.result" (template, template)) in
   print "call z3 to do ..." fseq
   val (ret,msgs) = exec cmd in
   ret != 0 => eprintf "z3 failed with message(s):%S" msgs |
   let fresult=fopen (template^".result") "r" in
   let sat=str_is_prefix "sat" (fgets fresult) then
   fclose fresult fseq
   sat ;

let compMaxPower ckt lower=
  let dupCkt=duplicate ckt then
  val (template, maxPossible)=creatTemplate dupCkt in
  let wid=neededWidth maxPossible then
  let check n=decide template wid n in
  let maxPow=binary_chop check upper lower then
   maxPow ;
```

Fig. 5. Implementation of the flow in Fig.**??**

the features of circuit manipulation and BDD of FORTE to compute the logical functions and weights of nodes, logical constraints of netlists.

Table **??** lists the scale of the benchmark circuits of ISCAS05 [**?**] in the number of the gates and gates.

Table 1. Scale of Benchmark Circuits

bench	c432	c499	c880	c1355	c1908
Input	72	82	120	82	66
Gates	160	202	383	546	880

Table 2. Comparison on optimal results among HL, SA, and GA algorithms

.bench	c0	c19	c432	c499	c880	c1355	c1908
HL	5	8	198	215	423	415	970
SA	5	8	167	205	413	460	947
GA	5	8	191	212	419	464	968
SMT	5	8	203	221	447	514	1025

In Table **??**, except the trivial cases, we can observe that the difference between the optimal results obtained from heuristics-based methods and the precise values that obtained from SMT ranges from 3% to 10%. Therefore, usually the optimal values computed by the heuristics is acceptable when we consider power estimation in practical design of circuits.

In fact, this work is begun after heuristics-based algorithms such as Hill Climbing or Genetic algorithms have been tried to solve the estimation of dynamic power consumption. Because only a locally optimal solution can be obtained, a natural question arises: what difference between the precise and optimal solutions does exist? We need a tool to tell us the precise solution. SMT solvers are used to try to give an answer.

We must admit that our work is a brute-force use of the SMT-tool Z3. We only use an SMT-solver to solve the PB optimization problem, and need not know the detail of the decision procedure of the SMT-solver. Here we mainly rely on the bit-vector theory to decide whether there exists a solution to the satisfiability for the formula that the PB function is greater or equal to a bound value. In fact, the efficiency of a SMT-solver depends on the choice of the background theory and the corresponding decision procedure. If we have more powerful decision procedure devoted to PB optimization problem, our approach will be more efficient.

References

[Ass07] Assim Sagahyroon, Fadi A.Aloul. Using sat-based techniques in power estimation. *Microelectronics Journal*, pages 706–715, 2007.

[BGJR88] Francisco Barahona, Martin Grötschel, Michael Jünger, and Gerhard Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Oper. Res.*, 36(3):493–513, May 1988.

[BH02] Endre Boros and Peter L. Hammer. Pseudo-boolean optimization. *Discrete Appl. Math.*, 123(1-3):155–225, November 2002.

[BHMJ99] Endre Boros, Peter L. Hammer, Michel Minoux, and David J. Rader Jr. Optimal cell flipping to minimize channel density in vlsi design and pseudo-boolean optimization. *Discrete Applied Mathematics*, 90(1-3):69–88, 1999.

[Fad02] Fadi A. Aloul, Soha Hassoun, Karem A. Sakallah, David Blaauw. Robust sat-based search algorithm for leakage power reduction. *Integrated Circuit Design. Power and Timing Modeling, Optimization and Simulation*, pages 167–177, 2002.

[Fei01] Fei Li, Lei He. Maximum current estimation considering power gating. *ISPD'01 Proceedings*, pages 106–111, 2001.

[HYH99] Mark C. Hansen, Hakan Yalcin, and John P. Hayes. Unveiling the iscas-85 benchmarks: A case study in reverse engineering. *IEEE Des. Test*, 16(3):72–80, July 1999.

[IBM] IBM. *CPLEX Optimizer*. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html.

[Ing03] Ingo Wegener. Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. *Springer US*, pages 349–369, 2003.

[Ing05] Ingo Wegener. On the analysisi of a simple evolutinary algorithm on quadratic pseudo-boolean functions. *Journal of Discrete Algorithms*, pages 61–78, 2005.

[Int] Intel. *Forte from Intel.* http://www.intel.com/software/products/ open-source/tools1/verification.

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

[MF97] S. Manich and J. Figueras. Maximizing the weighted switching activity in combinational cmos circuits under the variable delay model. In *Proceedings of the 1997 European conference on Design and Test*, EDTC '97, pages 597–602, Washington, DC, USA, 1997. IEEE Computer Society.

[Mic99] Michael S.Hsiao. Peak power estimation using genetic spot optimization for large vlsi circuits. *Proceedings of Design, Automation and Test in Europe Conference and Exibition*, 1999.

[NO06] Robert Nieuwenhuis and Albert Oliveras. On sat modulo theories and optimization problems. In *Proceedings of the 9th international conference on Theory and Applications of Satisfiability Testing*, SAT'06, pages 156–169, Berlin, Heidelberg, 2006. Springer-Verlag.

[Ped96] Massoud Pedram. Power minimization in ic design: principles and applications. *ACM Trans. Des. Autom. Electron. Syst.*, 1(1):3–56, January 1996.

[SA07] Assim Sagahyroon and Fadi A. Aloul. Using sat-based techniques in power estimation. *Microelectronics Journal*, 38(67):706 – 715, 2007.

[SDJ$^+$95] Caterina De Simone, Martin Diehl, Michael Jnger, Petra Mutzel, Gerhard Reinelt, and Giovanni Rinaldi. Exact ground states of ising spin glasses: New experimental results with a branch and cut algorithm, 1995.

[Sri92] Srinivas Devadas, Kurt Keutzer, Jacob White. Estimation of power dissipation in cmos combinational circuits using boolean function manipulation. *Computer Aided Design*, pages 373–383, 1992.

[S.T98] S.Thompson, P.Packan, M.Bohr.MOS scaling. Transistor challenges for the 21st century. *Intel Tchnology Journal*, 1998.

[WW05] Ingo Wegener and Carsten Witt. On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions. *J. Discrete Algorithms*, 3(1):61–78, 2005.