GRCircuit - Functional Macro Blocks Circuit Recovering Tool

Leandro Maia Maia Silva (✉ elmaia@dcc.ufmg.br)

Universidade Federal de Minas Gerais Instituto de Ciencias Exatas <https://orcid.org/0000-0003-3783-9398>

Fabricio Vivas Andrade

CEFET-MG: Centro Federal de Educacao Tecnologica de Minas Gerais

Luiz Filipe Menezes Vieira

UFMG: Universidade Federal de Minas Gerais

Research Article

Keywords: circuit recovering, circuit' structural information, subgraph isomorphism, SAT

Posted Date: April 30th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-437409/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Abstract

Considering that the more information you can gather about a particular circuit, you can address problems more accurately in the Electronic Design Automation (EDA) field, therefore, many tools focus on obtaining the maximum amount of information about the input to which it is provided in order to determine which are the best algorithms to each instance. Some of these tools are the Boolean Satisfiability (SAT) problem solvers; which, for the most part, receive formulas described in Conjunctive Normal Form (CNF) as input. The circuits encoding process to the CNF format, unfortunately, destroy much of the information that could have been used to optimize SAT solvers, as part of this information must be recovered to avoid applying generic algorithms in the solution of SAT problems. One of the difficult aspects of retrieving this information corresponds to the matching of clauses to its respective logic gates, as well as which sets of logic gates correlate to a functional block. The present work makes use of subgraph isomorphism algorithms to recover circuits encoded in CNF-DIMACS maximizing the number of clauses handled, both at the level of logic gates as well as more complex structural blocks, which allow their identification at higher levels of abstraction. Our tool was able to successfully recover all circuits

Full-text

Due to technical limitations, full-text HTML conversion of this manuscript could not be completed. However, the manuscript can be downloaded and accessed as a PDF.

Figures

$$\begin{aligned}
z = not(x) &\equiv \left(\neg x \vee \neg z \right) \wedge \left(x \vee z \right) \\
z = and(x_1, \dots, x_n) &\equiv \left[\bigwedge_{i=1}^n (x_i \vee \neg z) \right] \wedge \left(\bigvee_{i=1}^n \neg x_i \vee z \right) \\
z = or(x_1, \dots, x_n) &\equiv \left[\bigwedge_{i=1}^n (\neg x_i \vee z) \right] \wedge \left(\bigvee_{i=1}^n x_i \vee \neg z \right) \\
z = nand(x_1, \dots, x_n) &\equiv \left[\bigwedge_{i=1}^n (x_i \vee z) \right] \wedge \left(\bigvee_{i=1}^n \neg x_i \vee \neg z \right) \\
z = nor(x_1, \dots, x_n) &\equiv \left[\bigwedge_{i=1}^n (\neg x_i \vee \neg z) \right] \wedge \left(\bigvee_{i=1}^n x_i \vee z \right) \\
z = xor(x_1, x_2) &\equiv \left(x_1 \vee x_2 \vee z \right) \wedge \left(\neg x_1 \vee x_2 \vee \neg z \right) \wedge \\
&\quad \left(x_1 \vee \neg x_2 \vee \neg z \right) \wedge \left(\neg x_1 \vee \neg x_2 \vee z \right) \\
z = xnor(x_1, x_2) &\equiv \left(x \vee z \vee z \right) \wedge \left(x \vee z \vee z \right) \wedge \\
&\quad \left(x \vee z \vee z \right) \wedge \left(x \vee z \vee z \right)
\end{aligned}$$

Figure 1

Tseitin transformation for some logic gates.

Gate	Operation	CNF- Signature	Graph
$c = a.b$	$(\neg a \vee \neg b \vee c) \wedge$ $(a \vee \neg c) \wedge$ $(b \vee \neg c)$

$c = \overline{a.b}$	$(\neg a \vee \neg b \vee \neg c) \wedge$ $(a \vee c) \wedge$ $(b \vee c)$

$c = a + b$	$(a \vee b \vee \neg c) \wedge$ $(\neg a \vee c) \wedge$ $(\neg b \vee c)$

$c = \overline{a + b}$	$(a \vee b \vee c) \wedge$ $(\neg a \vee \neg c) \wedge$ $(\neg b \vee \neg c)$

$c = \overline{a}$	$(\neg a \vee \neg b) \wedge$ $(a \vee b)$

$c = a \oplus b$	$(\neg a \vee \neg b \vee \neg c) \wedge$ $(a \vee b \vee \neg c) \wedge$ $(a \vee \neg b \vee c) \wedge$ $(\neg a \vee b \vee c)$

$c = \overline{a \oplus b}$	$(\neg a \vee \neg b \vee c) \wedge$ $(\neg a \vee b \vee \neg c) \wedge$ $(a \vee \neg b \vee \neg c) \wedge$ $(a \vee b \vee c)$

Figure 2

CNF-Signatures and graphs of some logic gates

Figure 3

Graphs of three different ANDs.

Figure 4

Full-Adder's circuit

Figure 5

Full-Adder Graph / Inputs and Outputs Hilighted

Figure 6

Part of the circuit C1355 from ISCAS 85.

Figure 7

Conflict graph between Full-Adder and simple logic gates

(a) Original

1 : 1 118 0
 2 : -1 -118 0
 37: 118 154 0
 38: 4 154 0
 39: -118 -4 -154 0

(b) Clauses

(c) Possible recovering

Figure 8

Part of the circuit C432 from ISCAS 85.

Figure 9

Synthesis of the recovery process using GRCircuit tool