Re-implement “Constraint Solving for Synthesis and Verification of Threshold Logic Circuits”

LSV Final Project Report

r10943109 丁宣匀　　r10943089 陳妍喻

I. Introduction

In our final project, we tried to re-implement the paper “*Constraint Solving for Synthesis and Verification of Threshold Logic Circuits*” written by Nian-Ze Lee and Jie-Hong R. Jiang from National Taiwan University, which was published on *2020 IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

Since threshold logic (TL) circuits are gaining increasing attention due to their strong bind to neural network applications, the automatic synthesis and verification of TL circuits are important.

The paper formulated the collapse operation for TL functions and a necessary and sufficient condition for collapsibility, which can achieve an average of 18% gate count reduction on top of synthesized TL circuits. It also proposed 2 ways to verify the collapsed TL circuits: TL-to-MUX tree, TL-to-PB constraints. Both ways can perform equivalence checking of TL circuits, and therefore verify the correctness of collapsed TL circuits.

In our final project, we tried to re-implement the collapse operation for TL functions and the TL-to-MUX tree conversion for verification.

II. Problem Formulation

From a general perspective, given a TL circuit, minimize its TLG counts as much as possible by performing collapse operation.

From a local perspective, given two TLGs *u* and *v*, decide whether two TLGs can be merged into one TLG or not.

To simplify the problem, the paper considered the special form of linear combination of *u* and *v* as the following formulation states and Figure 1.

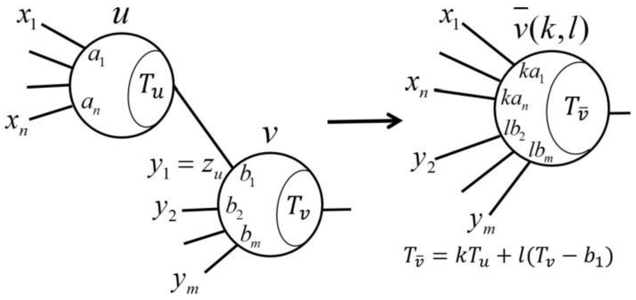


Fig. 1. Collapse of two TLGs via their linear combination.

Given two TLGs *u* = [a_1, …, a_n; T_u] with inputs (x_1, …,x_n) and *v* = [b_1, …, b_m; T_v] with inputs (y_1, …,y_m), let *u* be a fanin of *v*, and assume y_1 = z_u. The *TLG collapsing problem of u to v via linear combination* asks whether there exists two positive parameters *k* and *l* such that the TLG ¯v (k , l) = [ka_1, …, ka_n, lb_2, …, lb_m; kT_u + l (T_v − b_1)] with inputs (x_1, …, x_n, y_2, …, y_m) satisfies f¯v (x_1, …, x_n, y_2, …, y_m) = f_v (f_u (x_1, …, x_n), y_2, …, y_m) for all truth assignments to variables (x_1, …, x_n, y_2, …, y_m).

III. Methods

A. Synthesis

In the paper, it defined three functions

$$f_u^+(x_1,\ldots,x_n) \overset{\text{def}}{=} \sum_{i=1}^{n} a_i x_i$$

$$f_v^-(y_2,\ldots,y_m) \overset{\text{def}}{=} \sum_{j=2}^{m} b_j y_j$$

$$f_{v\circ u} \overset{\text{def}}{=} f_v(f_u(x_1,\ldots,x_n), y_2,\ldots,y_m)$$

and four sets of truth assignments

$$\phi_1 \overset{\text{def}}{=} \{(\beta_2, \ldots, \beta_m) \mid f_v^-(\beta_2, \ldots, \beta_m) \leq T_v - b_1 - 1\}$$
$$\phi_2 \overset{\text{def}}{=} \{(\beta_2, \ldots, \beta_m) \mid f_v^-(\beta_2, \ldots, \beta_m) \geq T_v\}$$
$$\phi_3 \overset{\text{def}}{=} \{(\beta_2, \ldots, \beta_m) \mid T_v - b_1 \leq f_v^-(\beta_2, \ldots, \beta_m) \leq T_v - 1\}$$
$$\phi_4 \overset{\text{def}}{=} \{(\alpha_1, \ldots, \alpha_n) \mid f_u^+(\alpha_1, \ldots, \alpha_n) \leq T_u - 1\}$$

for $(\alpha_1, \ldots, \alpha_n) \in \mathbb{B}^n$ and $(\beta_2, \ldots, \beta_m) \in \mathbb{B}^{m-1}$.

Based on the above truth assignments, the paper derived 2 theorems which can decide the collapsing feasibility and find the parameter *k* and *l* if the collapsing is feasible.

Theorem 1 forms the necessary and sufficient conditions for collapsing. However, it is difficult to compute due to the need of solving Subset Sum Problem, which is a NP-hard problem.

$$l(T_v - b_1 - \max_{\beta \in \phi_1}\{f_v^-\}) \geq k(\max\{f_u^+\} - T_u) + 1, \ \ \phi_1 \neq \emptyset$$
$$l(\min_{\beta \in \phi_2}\{f_v^-\} + b_1 - T_v) \geq k(T_u - \min\{f_u^+\}), \ \ \phi_2 \neq \emptyset$$
$$k(T_u - \max_{\alpha \in \phi_4}\{f_u^+\}) \geq l(\max_{\beta \in \phi_3}\{f_v^-\} + b_1 - T_v) + 1.$$

Theorem 2 implies the sufficient conditions for collapsing and it is efficient to compute. The time complexity is linear to the fanin size of the TLG.

$$l \geq k(\max\{f_u^+\} - T_u) + 1, \ \ \phi_1 \neq \emptyset$$
$$lb_1 \geq k(T_u - \min\{f_u^+\}), \ \ \phi_2 \neq \emptyset$$
$$k \geq l(b_1 - 1) + 1.$$

We and the paper both apply Theorem 2 for collapsing TLGs.

B. Verification

The paper proposed two methods for verifying whether two given TL circuits have the same functionality. One is MUX-based verification method, the other is PB-based verification method.

In our implementation, we performed MUX-based verification method to verify our collapsed circuits: Given two TL circuits, we first convert TL circuits into equivalent Boolean logic circuits which are based on MUX gates, and apply the command \`*cec*\` in ABC environment for equivalence checking.

VI. Re-implementation

Our goal is that given a TL circuit, minimize its TLG counts as much as possible by performing collapse operation, and also verify the correctness of the results.

Figure 2 is our flow. Figure 3 is the pseudocode for \`*Collapse circuit*\`. Figure 4 is the pseudocode for \`*Verify collapsed circuit*\`.

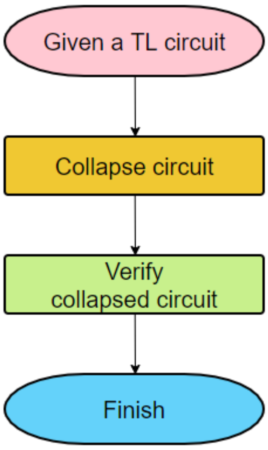
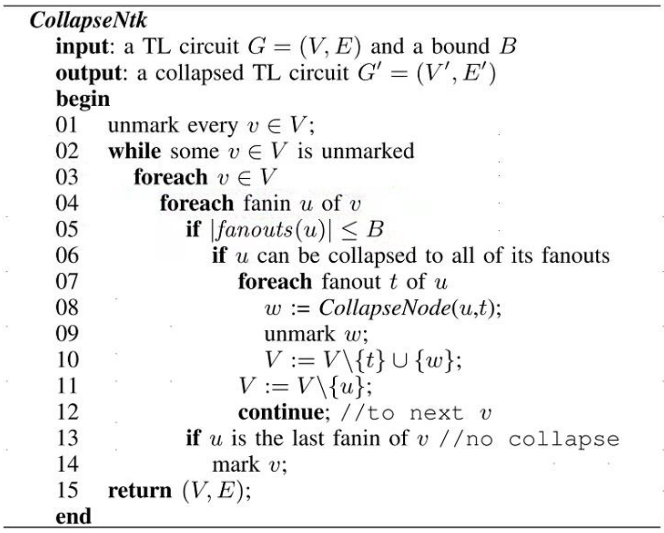
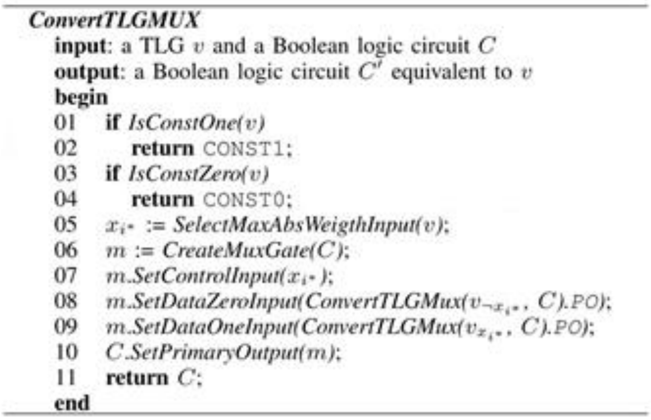


Fig. 2



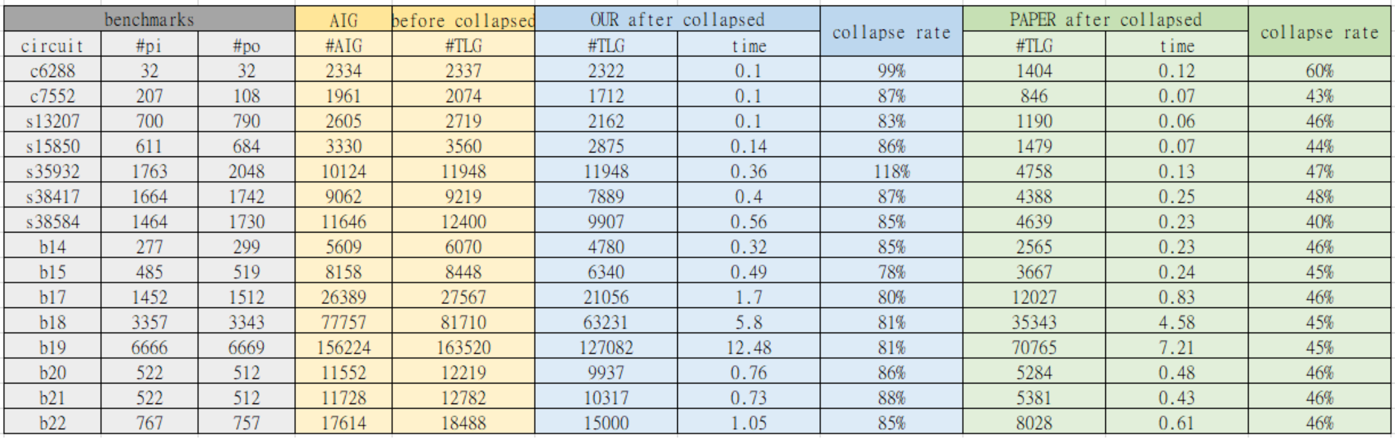
```
CollapseNtk
  input: a TL circuit $G = (V, E)$ and a bound $B$
  output: a collapsed TL circuit $G' = (V', E')$
  begin
      unmark every $v \in V$;
      while some $v \in V$ is unmarked
          foreach $v \in V$
              foreach fanin $u$ of $v$
                  if $|fanouts(u)| \leq B$
                      if $u$ can be collapsed to all of its fanouts
                          foreach fanout $t$ of $u$
                              $w$ := CollapseNode($u$,$t$);
                              unmark $w$;
                              $V := V \backslash \{t\} \cup \{w\}$;
                          $V := V \backslash \{u\}$;
                          continue; //to next v
                  if $u$ is the last fanin of $v$ //no collapse
                      mark $v$;
      return $(V, E)$;
  end
```

Fig. 3. Algorithm: Collapse a TL circuit.



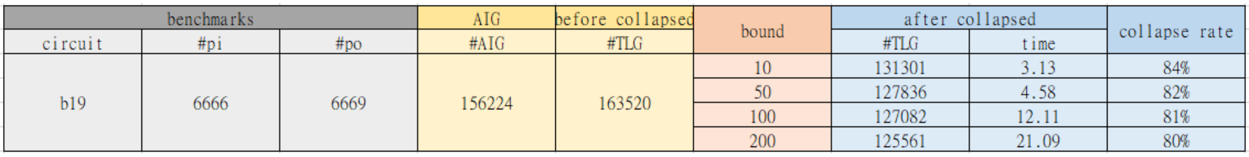
```
ConvertTLGMUX
  input: a TLG $v$ and a Boolean logic circuit $C$
  output: a Boolean logic circuit $C'$ equivalent to $v$
  begin
      if IsConstOne($v$)
          return CONST1;
      if IsConstZero($v$)
          return CONST0;
      $x_{i^*}$ := SelectMaxAbsWeigthInput($v$);
      $m$ := CreateMuxGate($C$);
      $m$.SetControlInput($x_{i^*}$);
      $m$.SetDataZeroInput(ConvertTLGMux($v_{\neg x_{i^*}}$, $C$).PO);
      $m$.SetDataOneInput(ConvertTLGMux($v_{x_{i^*}}$, $C$).PO);
      $C$.SetPrimaryOutput($m$);
      return $C$;
  end
```

Fig. 4 Algorithm: Convert a TLG into a MUX tree.



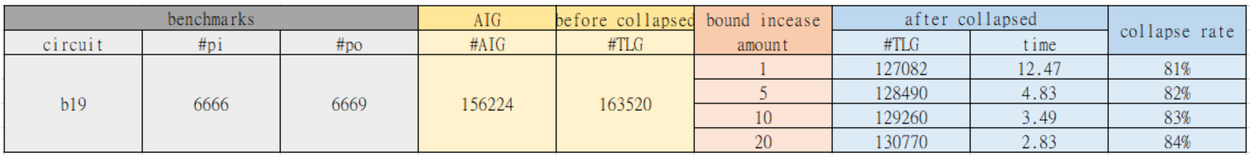
benchmarks			AIG	before collapsed	OUR after collapsed		collapse rate	PAPER after collapsed		collapse rate
circuit	#pi	#po	#AIG	#TLG	#TLG	time		#TLG	time	
c6288	32	32	2334	2337	2322	0.1	99%	1404	0.12	60%
c7552	207	108	1961	2074	1712	0.1	87%	846	0.07	43%
s13207	700	790	2605	2719	2162	0.1	83%	1190	0.06	46%
s15850	611	684	3330	3560	2875	0.14	86%	1479	0.07	44%
s35932	1763	2048	10124	11948	11948	0.36	118%	4758	0.13	47%
s38417	1664	1742	9062	9219	7889	0.4	87%	4388	0.25	48%
s38584	1464	1730	11646	12400	9907	0.56	85%	4639	0.23	40%
b14	277	299	5609	6070	4780	0.32	85%	2565	0.23	46%
b15	485	519	8158	8448	6340	0.49	78%	3667	0.24	45%
b17	1452	1512	26389	27567	21056	1.7	80%	12027	0.83	46%
b18	3357	3343	77757	81710	63231	5.8	81%	35343	4.58	45%
b19	6666	6669	156224	163520	127082	12.48	81%	70765	7.21	45%
b20	522	512	11552	12219	9937	0.76	86%	5284	0.48	46%
b21	522	512	11728	12782	10317	0.73	88%	5381	0.43	46%
b22	767	757	17614	18488	15000	1.05	85%	8028	0.61	46%

Table I. Statistics of collapsing AIG circuits and the comparison to the statistics in the paper.



benchmarks			AIG	before collapsed	bound	after collapsed		collapse rate
circuit	#pi	#po	#AIG	#TLG		#TLG	time	
b19	6666	6669	156224	163520	10	131301	3.13	84%
					50	127836	4.58	82%
					100	127082	12.11	81%
					200	125561	21.09	80%

Table II. Comparison to different maximum bound setting.



benchmarks			AIG	before collapsed	bound increase	after collapsed		collapse rate
circuit	#pi	#po	#AIG	#TLG	amount	#TLG	time	
b19	6666	6669	156224	163520	1	127082	12.47	81%
					5	128490	4.83	82%
					10	129260	3.49	83%
					20	130770	2.83	84%

Table III. Comparison to different bound increment per iteration.

V. Experimental Results

Same as the paper, we developed in ABC environment, and we used the combinational logic of circuits in ISCAS benchmark suites for evaluation. The fanout bound parameter B was set to be increased from 1 up to 100 iteratively. Different from the paper, we did not impose any fanin size limitation nor any weight/threshold value limitation.

The experimental results are in Table I, II III.

In Table I, our collapse rates were worse than the collapse rates in the paper. This may result from the fact that we did not impose any limitation on fanin size or weight/threshold value The collapsed TLGs may have a large number of fanins or have higher weight/threshold value, which may make it more difficult to merge these collapsed TLGs at the later stage.

In Table II, we imposed different maximum fanout bound parameter B and increased the bound from 1 up to B iteratively. The statistics showed that the bigger parameter B was, the better the collapse rate was. This is very reasonable because a bigger fanout bound parameter B means a more flexible constraint on collapsing.

In Table III, we imposed different bound increments. As the statistics showed, the collapse rate became worse if the increment was bigger. This phenomenon confirmed the statement in the paper, which stated that if no limitation is imposed from the beginning, a TLG with large fanout size would be collapsed to all its fanouts earlier if collapsible, resulting in a large number of collapsed TLGs, which are more difficult to be collapsed with other TLGs.

VI. Summary

We re-implemented the paper "*Constraint Solving for Synthesis and Verification of Threshold Logic Circuits*". However, our results differed from the statistics in the paper. This phenomenon may be result from the fact that we imposed less limitation on collapse operation.