MajorSat: A SAT Solver to Majority Logic

Speaker : Ching-Yi Huang

Authors: Yu-Min Chou, Yung-Chih Chen[*], Chun-Yao Wang, Ching-Yi Huang

National Tsing Hua University, Taiwan

[*]Yuan Ze University, Taiwan

Jan. 27, 2016

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Decision Order Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Introduction (1/2)

- The majority function is a concise way to represent a Boolean expression
- The majority function, denoted as $M(x_1, x_2, \ldots, x_n)$, is an odd-input function which is evaluated as 1 iff more than half of inputs are 1
  - E.g. $M(a, b, c, d, e) = 1$ if a, b, c are 1
- The majority function can express any logic represented by OR or AND operations
  - E.g. $(a \vee b \vee c) \equiv M(a, b, c, 1, 1)$

    $(a \wedge b \wedge c) \equiv M(a, b, c, 0, 0)$
- Recently, majority logic attracts more attentions and some synthesis algorithms and axiomatic system for majority logic have been proposed

Introduction (2/2)

- The Boolean satisfiability (SAT) problem can be expressed in various forms
  - E.g. conjunctive-normal-form (CNF), disjunctive-normal-form (DNF), the conjunction of majority functions, …etc
- The conjunctive-normal-form (CNF) solvers for the SAT problem have had a remarkable achievement and have been widely used in the domains of synthesis and verification of logic circuit
- To express specific logic functions such as majority decision problems, majority functions can be more compact and expressive compared to traditional CNF

Motivation (1/2)

- It's impractical to convert the large-size majority function into the CNF for been solved by CNF SAT solvers
  - The time required to convert the majority function to the CNF grows exponentially with the size of the majority function
  - Modern CNF SAT solvers may be not able to store so many clauses
  - E.g. M(*a, b, c, d, e*) = (*a* ∨ *b* ∨ *c*) ∧ (*a* ∨ *b* ∨ *d*) ∧ (*a* ∨ *b* ∨ *e*) ∧ (*a* ∨ *c* ∨ *d*) ∧ (*a* ∨ *c* ∨ *e*) ∧ (*a* ∨ *d* ∨ *e*) ∧ (*b* ∨ *c* ∨ *d*) ∧ (*b* ∨ *c* ∨ *e*) ∧ (*b* ∨ *d* ∨ *e*) ∧ (*c* ∨ *d* ∨ *e*), $\binom{n}{\lceil n/2 \rceil}$ clauses, where *n* is the size of the majority function

Motivation (2/2)

- Therefore, we propose a new SAT solver – MajorSat, which can directly solve the instances with majority functions and CNF clauses
- Definition 1: A majority expression, denoted as ME, is a conjunction of majority functions
  - $M(a, b, \overline{c}) \wedge M(\overline{d}, \overline{e}, 0, f, g)$ is an ME

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Decision Order Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Preprocessing

- Property 1:
  - A majority function with size n can be simplified to a majority function with size $(n - 2)$ by removing two inputs that are either the same variable but with different phases, or 0 and 1, while preserving the same result
- E.g. M(~~a,~~ a, b, c, ~~$\bar{a}$~~, 0, 1) can be reduced to M(a, b, c, 0, 1), and M(a, b, c, ~~0~~, ~~1~~) can be further reduced to M(a, b, c)

Conflicts Analysis (1/6)

- Property 2:
  - The ME is **UNSAT** if it satisfies the following conditions simultaneously
    - (1) There exists a majority function of size n with an input $x \in \{a, \bar{a}, 0, 1\}$ appearing more than $\lfloor n/2 \rfloor$ times
    - (2) There exists another majority function of size m with an input $y \in \{a, \bar{a}, 0, 1\}$ appearing more than $\lfloor m/2 \rfloor$ times
    - (3) $x = \bar{y}$
- E.g. An ME: M(*a*, *a*, *a*, *b*, *c*) ∧ M(*ā*, *ā*, *ā*, *ā*, *d*, *e*, *f*)

M(*a*, *a*, *a*, *b*, *c*) ⟷ conflicts with ⟷ M(*ā*, *ā*, *ā*, *ā*, *d*, *e*, *f*)

3 $> \lfloor 5/2 \rfloor = 2$

4 $> \lfloor 7/2 \rfloor = 3$

Conflicts Analysis (2/6)

- Definition 2: Given a majority function *s*, the minimum number of variables required to be assigned such that *s* is 1 is denoted as $MinV_s$.
  - E.g. For a majority function *s*: M(*a, a, b, c, d*), $MinV_s$ is 2, i.e., (*a, b*), (*a, c*), or (*a, d*) = (1, 1)
- Property 3:
  - Given two majority functions *s* and *t*, the ME = $s \wedge t$ is **UNSAT** if
    - *s* and *t* have the same variable set with size w, and the phase of each variable in *s* is opposite to the phase of that in *t*
    - Both $MinV_s$ and $MinV_t$ > $\lfloor w/2 \rfloor$

Conflicts Analysis (3/6)

- Example:
  - ✓ $M(a, b, c) \wedge M(\bar{a}, \bar{b}, \bar{c})$ is UNSAT

    $w = 3, MinV_{M(a, b, c)} = MinV_{M(\bar{a}, \bar{b}, \bar{c})} = 2 > \lfloor w/2 \rfloor = 1$
  - ✓ $M(a, b, c) \wedge M(\bar{a}, \underline{\bar{b}}, \underline{\bar{b}}, \underline{\bar{b}}, \bar{c})$ doesn't work with property 3

    $w = 3, MinV_{M(\bar{a}, \bar{b}, \bar{b}, \bar{b}, \bar{c})} = 1 \not> \lfloor w/2 \rfloor = 1$
  - ✓ $M(a, b, c) \wedge M(\bar{a}, \bar{b}, \bar{b}, \bar{b}, \bar{c}, \bar{c}, \bar{c})$ is UNSAT

    $w = 3, MinV_{M(a, b, c)} = MinV_{M(\bar{a}, \bar{b}, \bar{b}, \bar{b}, \bar{c}, \bar{c}, \bar{c})} = 2 > \lfloor w/2 \rfloor = 1$

Conflicts Analysis (4/6)

- Property 4:
  - Given a majority function of size *n*, resolving a literal *a* (assigning 0 to the literal *a*) that occurs k times in the function implies all the other literals that occur $\geq \lceil n/2 \rceil - k$ times for satisfying the majority function
  - E.g. In M(*a, a, b, b,* $\bar{c}$, $\bar{c}$, *d*)(*n*=7), resolving the literal *a* (*a* = 0, *k* = 2) implies both literals *b* and $\bar{c}$ to 1
- Construct the implication graph based on Property 4
  - The ME is **UNSAT** if there exists a strongly-connected component in the implication graph containing nodes of a variable with opposite phases

Conflicts Analysis (5/6)

Consider the ME containing only 3-input majority functions:

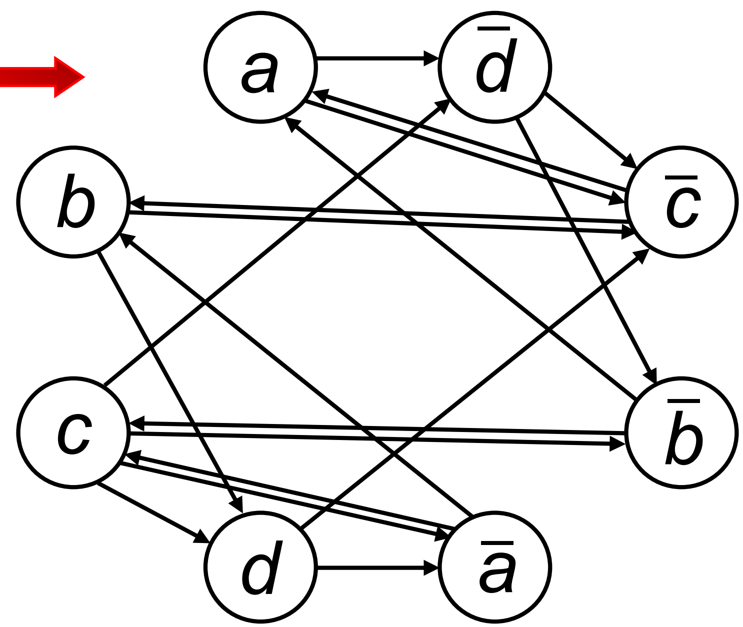
$$F = M(a, b, c) \wedge M(\bar{b}, \bar{c}, d) \wedge M(\bar{a}, \bar{c}, \bar{d})$$

The potential conflicts hidden in the ME can be extracted by forming the following implications

$\bar{a} \rightarrow (b \wedge c)$, $\bar{b} \rightarrow (a \wedge c)$, $\bar{c} \rightarrow (a \wedge b)$

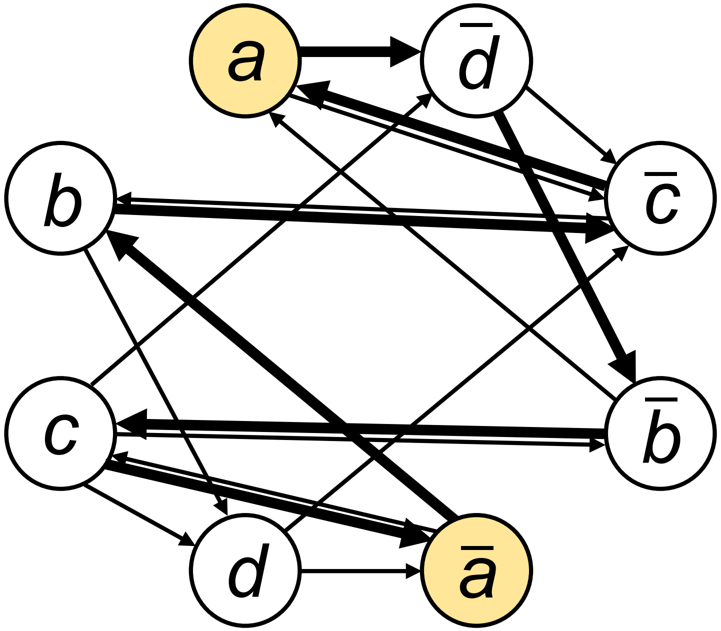
$b \rightarrow (\bar{c} \wedge d)$, $c \rightarrow (\bar{b} \wedge d)$, $\bar{d} \rightarrow (\bar{b} \wedge \bar{c})$

$a \rightarrow (\bar{c} \wedge \bar{d})$, $c \rightarrow (\bar{a} \wedge \bar{d})$, $d \rightarrow (\bar{a} \wedge \bar{c})$



Conflicts Analysis (6/6)

Assigning variable a = 1 leads to $\bar{a}$ = 1, and assigning variable $\bar{a}$ = 1 leads to a = 1



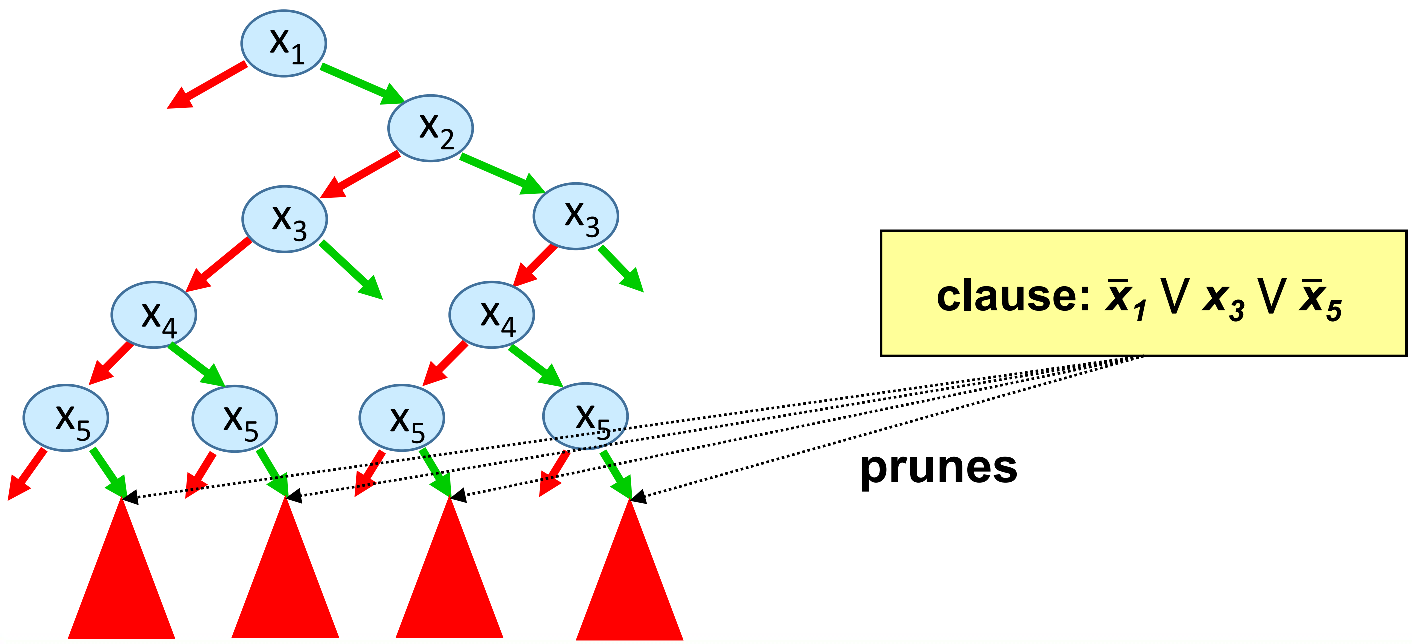
Node a and node ā belong to the same **strongly-connected component**, which means the original formula $M(a, b, c) \wedge M(\bar{b}, \bar{c}, d) \wedge M(\bar{a}, \bar{c}, \bar{d})$ is **UNSAT**

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Decision Order Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Searching with Conflict-driven Learning Technique

- Searching procedure
  - Determine the values of variables one by one
  - Record the reasons of conflicts
    - Can help in pruning the search space



Majority Propagation

- Property 5 (Majority Propagation):
  - During the searching procedure, when a majority function *s* of size *n* with *k* inputs have been assigned to 0

  ⇨ Any unassigned literal in *s* that occurs $\geq \lceil n/2 \rceil - k$ times will be implied to 1 for satisfying *s*

- E.g. In *M(a, a, a, b, c, e, e, f, g)*, if literals *e* and *g* have been assigned 0 (*k* = 3), the literal *a* is implied to 1 since *a* occurs three times, which $\geq \lceil 9/2 \rceil - 3 = 2$

Learning Example (1/3)

The following example shows the procedure of the searching with conflict-driven learning technique:

$$M(a, a, a, b, c, c, e, 1, 1) \wedge M(\bar{a}, \bar{b}, c, e, 1) \wedge M(d, \bar{e}, f, g, h)$$

Decide f = 0

$$M(a, a, a, b, c, c, e, 1, 1) \wedge M(\bar{a}, \bar{b}, c, e, 1) \wedge M(d, \bar{e}, f, g, h)$$

Decide g = 0

$$M(a, a, a, b, c, c, e, 1, 1) \wedge M(\bar{a}, \bar{b}, c, e, 1) \wedge M(d, \bar{e}, f, g, h)$$

Majority Propagation

Imply $d = 1$, $\bar{e} = 1$, and $h = 1$

Learning Example (2/3)

M($a, a, a, b, c, c, e, 1, 1$) ∧ M($\bar{a}, \bar{b}, c, e, 1$) ∧ M($d, \bar{e}, f, g, h$)

⇩ *Decide c = 0*

M($a, a, a, b, c, c, e, 1, 1$) ∧ M($\bar{a}, \bar{b}, c, e, 1$) ∧ M($d, \bar{e}, f, g, h$)

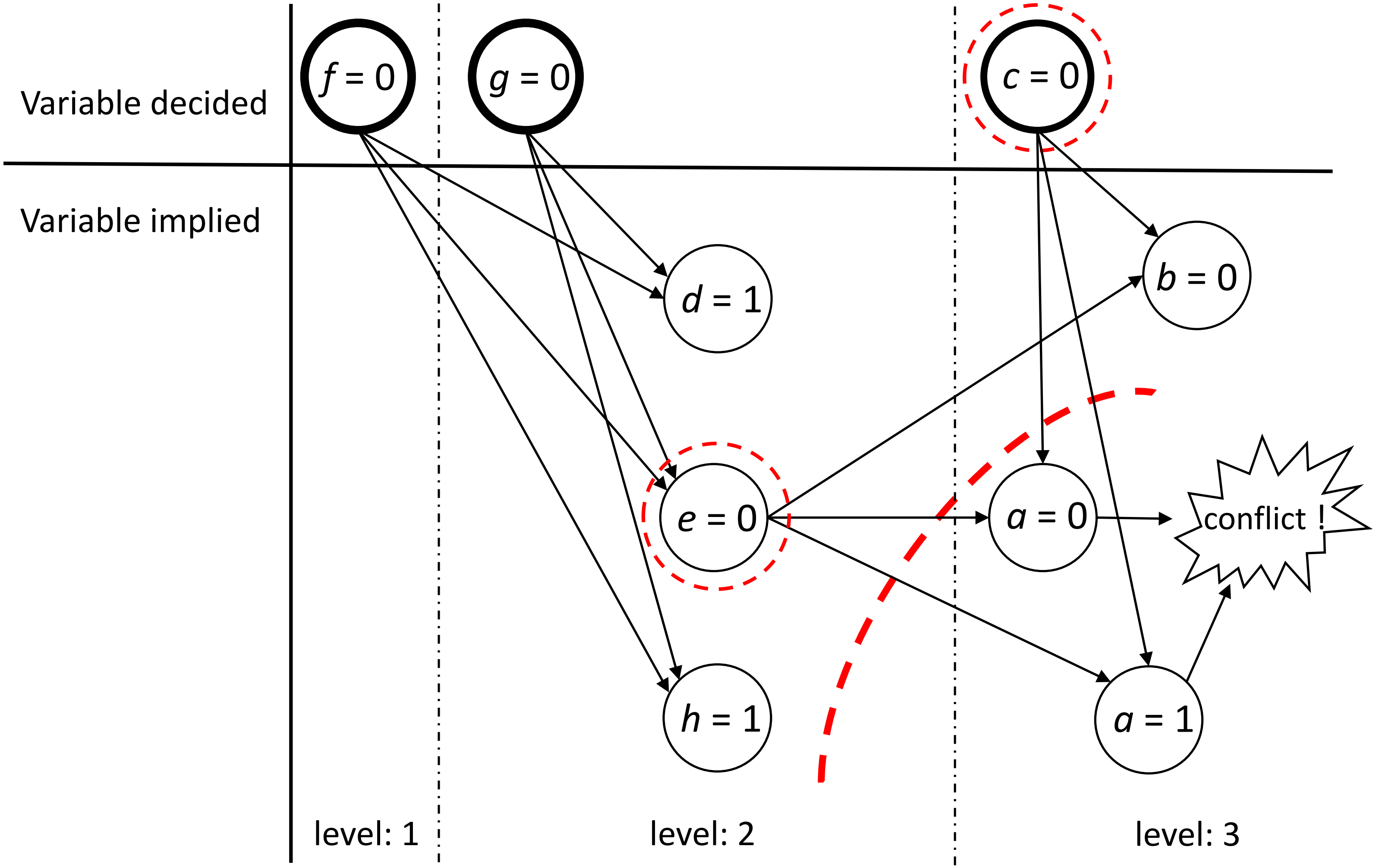
⇩ *Majority Propagation*

Imply $a = 1, \bar{a} = 1$, and $\bar{b} = 1$

⇩

Conflict !

Learning Example (3/3)



The clause **(*e*** ∨ ***c*)** is learned, and is added to the original ME

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Decision Order Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Variable Decision Order Heuristic (1/4)

Definition 3: $threshold_m$ and $weight_m(x)$ are defined as $\lceil n/2 \rceil$ and the appearance time of the variable *x* in a majority function *m* of size *n*

Definition 4: The score function of variable *x* in a majority function *m* and a clause *c* is denoted as $scoreM_m(x)$ and $scoreC_c(x)$, which are

$$scoreM_m(x) = \begin{cases} 0 & \text{if } x \text{ is absent in } m \\ 1 - (threshold_m - weight_m(x))/\text{size of } m & \text{if } x \text{ is in } m \end{cases}$$

$$scoreC_c(x) = \begin{cases} 0 & \text{if } x \text{ is absent in } c \\ 1 & \text{If } x \text{ is in } c \end{cases}$$

Definition 5: The score function $score(x)$ is to decide the variable decision, which is

$$score(x) = \sum_{m \in M} scoreM_m(x) + \sum_{c \in C} scoreC_c(x)$$

Variable Decision Order Heuristic (2/4)

- According to **Definition 4**, the scores of variables are related to their appearance times in majority functions
  - Choosing the variable of a higher score can increase the probability of reaching the satisfiable branch
- E.g. Given an expression F: $M(a, a, a, b, b, \bar{c}, d) \wedge M(\bar{b}, \bar{c}, \bar{d}) \wedge (\bar{a} \vee b \vee c)$,

$$score(a) = (1 - (4 - 3)/7) + 0 + 1 = 13/7 = 39/21$$

$$score(b) = (1 - (4 - 2)/7) + (1 - (2 - 1)/3) + 1 = 50/21$$

$$score(c) = (1 - (4 - 1)/7) + (1 - (2 - 1)/3) + 1 = 47/21$$

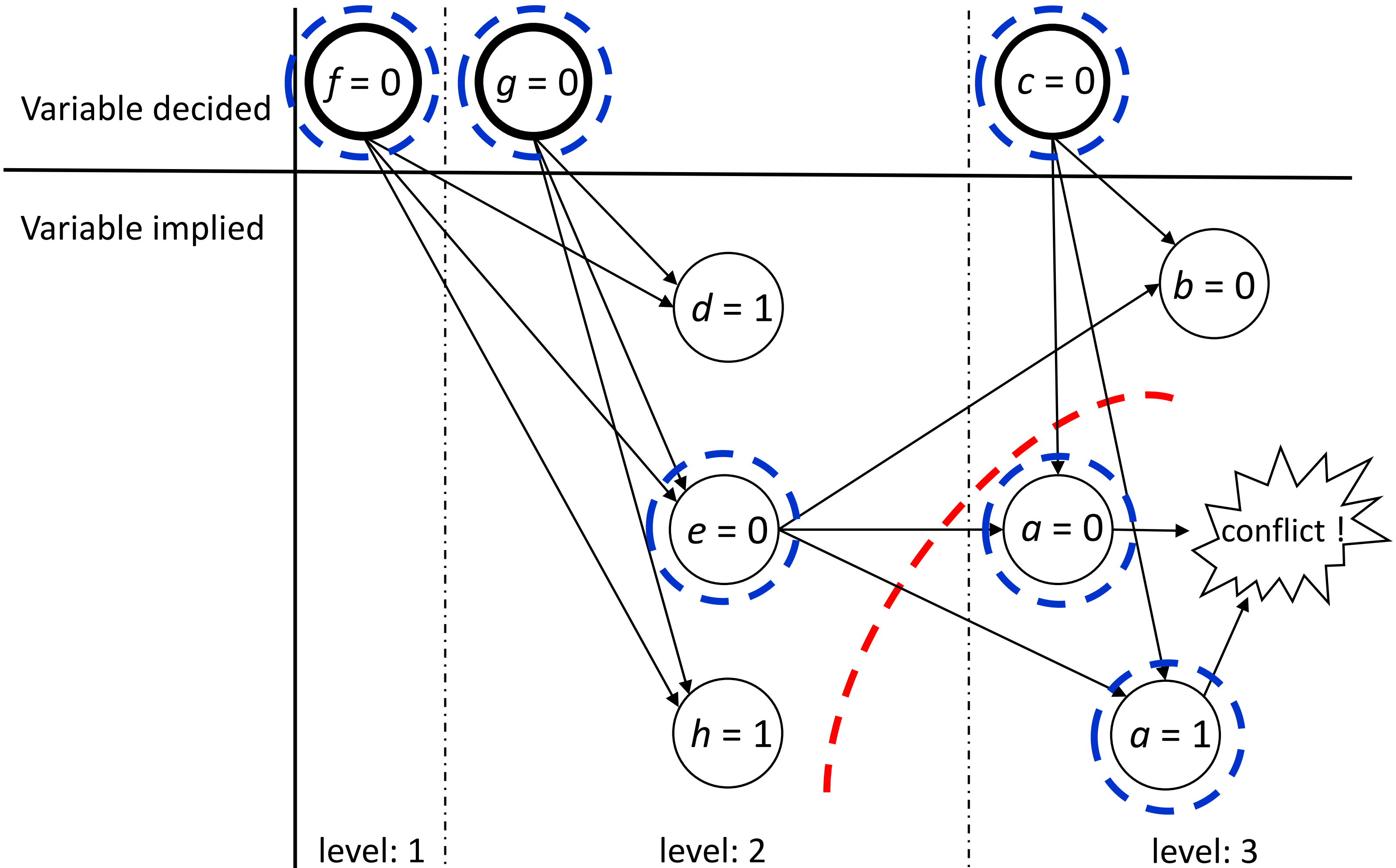
$$score(d) = (1 - (4 - 1)/7) + (1 - (2 - 1)/3) + 0 = 26/21$$

$score(b) > score(c) > score(a) > score(d)$, which indicates that the variable decision order is ***b > c > a > d***

Variable Decision Order Heuristic (3/4)

- Update scores of variables when conflicts happen
  - Add 1 to the scores of variables on the paths from conflict nodes to decision nodes
  - Recompute the variable decision order
  - Lead the search to unsatisfiable branches
    - Help in learning more conflict clauses

Variable Decision Order Heuristic (4/4)



***score*(*a*), *score*(*c*), *score*(*e*), *score*(*f*), and *score*(*g*) are added by 1 after the conflict happens**

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Decision Order Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Majority Gate Transformation (1/2)

- The characteristic function of a majority gate can be expressed by an **ME**

$$\Downarrow$$

$$(d \rightarrow M(a, b, c)) \wedge (\bar{d} \rightarrow \overline{M}(a, b, c))$$

$$\Downarrow$$

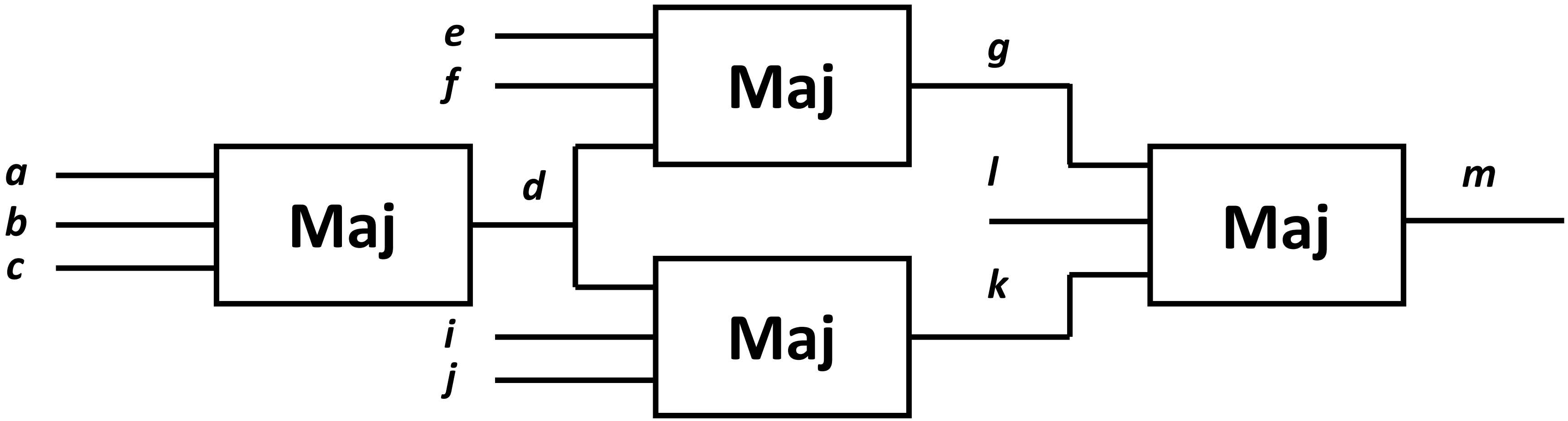
$$(\bar{d} \vee M(a, b, c)) \wedge (d \vee M(\bar{a}, \bar{b}, \bar{c}))$$

$$\Downarrow$$

$$M(a, b, c, \bar{d}, \bar{d}, \bar{d}, 1, 1, 1) \wedge M(\bar{a}, \bar{b}, \bar{c}, d, d, d, 1, 1, 1)$$

Majority Gate Transformation (2/2)

- Therefore, the characteristic function of a majority network can be also expressed as an **ME**



The satisfiablility of the above network can be evaluated through an ME:

$$\mathrm{M}(a, b, c, \bar{d}, \bar{d}, \bar{d}, 1, 1, 1) \wedge \mathrm{M}(\bar{a}, \bar{b}, \bar{c}, d, d, d, 1, 1, 1)$$
$$\wedge\, \mathrm{M}(d, e, f, \bar{g}, \bar{g}, \bar{g}, 1, 1, 1) \wedge \mathrm{M}(\bar{d}, \bar{e}, \bar{f}, g, g, g, 1, 1, 1)$$
$$\wedge\, \mathrm{M}(d, i, j, \bar{k}, \bar{k}, \bar{k}, 1, 1, 1) \wedge \mathrm{M}(\bar{d}, \bar{i}, \bar{j}, k, k, k, 1, 1, 1)$$
$$\wedge\, \mathrm{M}(g, i, k, \bar{m}, \bar{m}, \bar{m}, 1, 1, 1) \wedge \mathrm{M}(\bar{g}, \bar{i}, \bar{k}, m, m, m, 1, 1, 1)$$
$$\wedge\, m$$

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Decision Order Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Experimental Environment

- Platform
  - Intel Xeon® E5530 2.40GHz CentOS 4.6 platform with 64GB memory
  - C++
- Benchmarks
  - CNF benchmarks from **SATLIB** for verifying the correctness
  - Randomly-generated benchmarks of ME with different scales for testing the efficiency

Experimental Results (1/2)

- CNF benchmarks with different numbers of variables and clauses

Benchmarks	|variable|	|clause|	Golden Result	Solving Result	
				CNF	ME
uf20-91	20	91	SAT	SAT	SAT
uf50-218	50	218	SAT	SAT	SAT
uf75-325	75	325	SAT	SAT	SAT
uf100-430	100	430	SAT	SAT	SAT
uf125-538	125	538	SAT	SAT	SAT
uf150-645	150	645	SAT	SAT	SAT
uuf50-218	50	218	UNSAT	UNSAT	UNSAT
uuf75-325	75	325	UNSAT	UNSAT	UNSAT
uuf100-430	100	430	UNSAT	UNSAT	UNSAT
uuf125-538	125	538	UNSAT	UNSAT	UNSAT
uuf150-645	150	645	UNSAT	UNSAT	UNSAT

Experimental Results (2/2)

- The experiments on randomly-generated ME benchmarks of different scales
  - (number of variables)_(number of majority functions)_(size of majority function)
- The solving time of MajorSat is less than the time of converting ME into CNF coupled with the solving time of CNF solvers

Benchmarks	MajorSat		MiniSat		Lingeling	
	$t_{sol}(s)$	$t_{conv}(s)$	$t_{sol}(s)$	$total(s)$	$t_{sol}(s)$	$total(s)$
75_75_17	2.37	1.37	> 1000	> 1000	> 1000	> 1000
75_75_19	9.51	5.53	> 1000	> 1000	> 1000	> 1000
75_75_21	12.38	22.80	> 1000	> 1000	> 1000	> 1000
75_75_23	20.16	97.58	> 1000	> 1000	> 1000	> 1000
75_75_25	42.37	410.07	> 1000	> 1000	> 1000	> 1000
75_75_27	118.14	> 1000	—	> 1000	—	> 1000
75_75_29	158.01	> 1000	—	> 1000	—	> 1000
100_100_11	0.15	0.04	3.55	3.59	10.70	10.74
100_100_13	2.81	0.14	475.10	475.24	404.40	404.54
100_100_15	12.94	0.43	> 1000	> 1000	> 1000	> 1000
100_100_17	59.59	2.10	> 1000	> 1000	> 1000	> 1000
100_100_19	140.05	8.10	> 1000	> 1000	> 1000	> 1000
100_100_21	894.18	30.36	> 1000	> 1000	> 1000	> 1000
125_125_11	2.88	0.04	895.80	895.84	237.60	237.64
125_125_13	11.08	0.17	> 1000	> 1000	> 1000	> 1000
125_125_15	152.87	0.59	> 1000	> 1000	> 1000	> 1000

Outline

- Introduction
- Solving Methods
  - Conflict Analysis
  - Conflict-driven Learning
  - Variable Order Decision Heuristic
- Majority Gate Transformation
- Experimental Results
- Conclusions

Conclusions

- We propose a new SAT solver – MajorSat – for solving majority logic
- Several properties about majority functions are also investigated to increase the efficiency of MajorSat
- The experimental results show that MajorSat is more efficient in solving majority expressions than CNF solvers

Thanks for Attention

- Q&A

The Overall Flow of MajorSat