Logic Synthesis & Verification, Fall 2014

National Taiwan University

Problem Set 2

Due on 2014/11/05 before lecture

1 [Cofactor and QBF]

(a) (6%) Given two arbitrary Boolean functions f and g and a Boolean variable v, prove that $(\neg f)_v = \neg(f_v)$ and $(f\ \langle op\rangle\ g)_v = (f_v)\ \langle op\rangle\ (g_v)$ for $\langle op\rangle = \{\oplus\}$.

(b) (16%) Prove the following implications or disprove by showing counterexamples. Consider "$\rightarrow$" and "$\leftarrow$" of "$\leftrightarrow$" separately when it is needed.

$$\forall x, \exists y. f(x,y,z) \leftrightarrow \exists y, \forall x. f(x,y,z)$$
$$\neg\forall x, \exists y. f(x,y,z) \leftrightarrow \exists x. (\neg\exists y. f(x,y,z))$$
$$\exists x. (f(x,y) \wedge g(x,y)) \leftrightarrow (\exists x. f(x,y)) \wedge (\exists x. g(x,y))$$
$$\exists x. (f(x,y) \vee g(x,y)) \leftrightarrow (\exists x. f(x,y)) \vee (\exists x. g(x,y))$$

(c) (8%) For an arbitrary QBF $\exists z. f(x,y,z)$, find a function $g(x,y)$ such that $\exists z. f(x,y,z) = f(x,y,g(x,y))$. Express the onset, offset, and don't-care set of g in terms of function f.

2 [BDD Procedures]

(a) (8%) Give a procedure `COFACTOR(`F`,`l`)` that takes an ROBDD F and a literal l (e.g., $l = x$ or $l = \neg x$) as input and produces the cofactored ROBDD $F|_l$ as output.

(b) (4%) Express BDD `COMPOSE(`F`,`v`,`G`)`, which substitutes variable v in function F with function G, in terms of BDD `ITE` and `COFACTOR`.

3 [BDD Operations]

Let $f = \neg ab\neg c \vee a\neg cd \vee ac\neg d$ and $g = c \oplus d \oplus e$.

(a) (8%) Draw the (shared) ROBDDs of f and g under variable ordering $a < b < c < d < e$ (with a on top).

(b) (8%) Reduce the above ROBDDs with complemented edges.

(c) (8%) Compute the ROBDD (with no complemented edges) of `ITE(`f`,0,`g`)`.

4 [SAT Solving]

(20%) Consider SAT solving the CNF formula consisting of the following 10 clauses

$$C_1 = (a + b + c), C_2 = (a + b + c' + d'), C_3 = (a + b' + c),$$
$$C_4 = (a + b' + c'), C_5 = (a + c' + d), C_6 = (a' + b + c),$$
$$C_7 = (a' + b' + d), C_8 = (a' + b' + c' + d'), C_9 = (b + d), C_{10} = (b' + c + d').$$

(a) (10%) Apply implication and conflict-based learning in solving the above CNF formula. Assume the decision order follows a, b, c, and then d; assume each variable is assigned 0 first and then 1. Whenever a conflict occurs, draw the implication graph and enumerate all possible learned clauses under the Unique Implication Point (UIP) principle. (In your implication graphs, annotate each vertex with "`variable = value@decision_level`", e.g., "$b = 0@2$", and annotate each edge with the clause that implication happens.) If there are multiple UIP learned clauses for a conflict, use the one with the UIP closest to the conflict in the implication graph.

(b) (10%) The **resolution** between two clauses $C_i = (C_i^* + x)$ and $C_j = (C_j^* + x')$ (where C_i^* and C_j^* are sub-clauses of C_i and C_j, respectively) is the process of generating their **resolvent** $(C_1^* + C_j^*)$. The resolution is often denoted as

$$\frac{(C_i^* + x) \qquad (C_j^* + x')}{(C_1^* + C_j^*)}$$

A fact is that a learned clause in SAT solving can be derived by a few resolution steps. Show how that the learned clauses of (a) can be obtained by resolution with respect to their implication graphs.

5 [SAT Solving]

(14%) Show that a CNF formula ϕ is unsatisfiable if and only if an empty clause can be obtained through resolution.