EECS 144 & 244, Fall 2011 **Lee, Roychowdhury, Seshia**

HW 4: Test Generation and SAT Solving

Assigned: October 5, 2011 *Due: October 14, 2011*

1. **CNF and DNF** (25 points)
   Consider the following DNF formula on $2n$ variables:

   $$(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee \ldots \vee (x_{2n-1} \wedge x_{2n})$$

   (a) (15 points) Suppose you rewrite the above formula into CNF without introducing any new variables, by repeated application of the following distributive law: $y_i \vee (y_j \wedge y_k) = (y_i \vee y_j) \wedge (y_i \vee y_k)$.
   Prove that the resulting CNF formula is exponential-size in n.
   [Hint: use mathematical induction]

   (b) (10 points) Can you rewrite the above formula in CNF using a total of n additional (new) variables so that the CNF comprises a total of $2n + 1$ clauses? Justify your answer.

2. (20 points)

   This problem involves the use of MINISAT, a publicly-available, open-source SAT solver. See the installation instructions below for information on how to obtain and install MINISAT.

   There are several files distributed with this homework assignment. Stub code is provided in `hw4.pl`. In the `benchmarks` directory, there are several benchmarks in BENCH format (this is the same format as HW1). `problem1a.bench`, `problem2a.bench`, and `problem2b.bench` are the files needed for this homework, all the rest of the `.bench` files are to help you debug your code.

   Background information: A *miter* is a circuit construction often used in equivalence checking. Let $D_1(x_1, ..., x_n)$ and $D_2(x_1, ..., x_n)$ be two different implementations of the same circuit. A miter between D_1 and D_2 is the circuit that evaluates to false when D_1 and D_2 are equivalent, and true when D_1 and D_2 differ in at least one output bit. The miter between D_1 and D_2 can be stated more succinctly as: $\neg(D_1(x_1, ..., x_n) = D_2(x_1, ..., x_n))$

   (a) Modify the circuit in `problem1a.bench` so that the `miter` output will be satisfiable if and only if one of the equivalences is violated (i.e., create the miter). See `problem1a.bench` for more information. Submit `problem1a.bench`.

   (b) Modify `hw4.pl` to generate clauses that represent each circuit node. You only need to modify the `bench_to_cnf` function. All the places in which modifications are required are labeled with `FILL THIS IN`. Use the code in the `NOT1` case to see an example of what should be done. Submit `hw4.pl`.

(c) Run `hw4.pl` on the CNF file generated from `problem1a.bench`. See below for usage instructions for `hw4.pl`. Submit CNF result file, i.e., the file specified with the `--res` option.

3. (20 points)

   Use `hw4.pl` to determine whether the circuits in `problem2a.bench` and `problem2b.bench` are satisfiable. State the results and submit the CNF result files.

4. (20 points)

   Create a BENCH file consisting of an equivalence checking problem between the circuit in Figure 1 with a stuck-at-1 fault on signal h and a version of the same circuit without the stuck-at-1 fault. To model a stuck-at-fault in ISCAS BENCH format, use the following property: $a \vee \overline{a}$. Run `hw4.pl` on the BENCH file and report the outcome. Submit BENCH file and CNF result file.

INSTALLATION INSTRUCTIONS:

1. Download `http://minisat.se/downloads/minisat-2.2.0.tar.gz`
2. Unpack MINISAT: `tar -xzf minisat-2.2.0.tar.gz`
3. Install MINISAT:
   (i) `cd minisat`
   (ii) `export MROOT=`\`pwd\` for Bash; `setenv MROOT` \`pwd\` for CSH
   (iii) `cd core`
   (iv) `make`
   (v) After following steps (i) - (iv), you should find `minisat` in your current directory.
4. Finally, you will need to modify the stub code so that it calls MINISAT properly. It is currently setup to run on system. For example, if you installed MINISAT in `/foo/bar/minisat-2.2.0`
   you will need to change line 209 to read:
   `my $minisat = "/foo/bar/minisat-2.2.0/minisat";`.
5. Mac OS X only. MINISAT won't compile on Mac unless you comment out the following lines in `Main.cc`:

```
double mem_used = memUsedPeak();
if (mem_used != 0) printf("Memory used : %.2f MB\n", mem_used);
```

STUB CODE:
Stub code is provided for this assignment. The usage instructions for `hw4.pl` are:

```
Usage   : hw4.pl [options]
Options :
          --file=<bench file>   : BENCH file
          --cnf=<cnf file>      : CNF file
          --res=<res file>      : CNF result file
NOTE: --file, --cnf, and --res must be specified.

Example:
hw4.pl --file=demorgans.bench --cnf=demorgans.cnf --res=demorgans.res
```

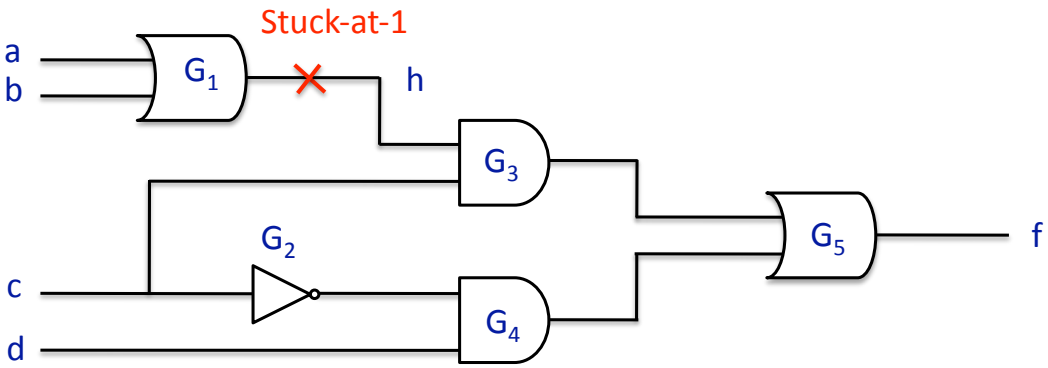

Figure 1: Circuit for problem 4