ENGR6963 FORMAL VERIFICATION
Homework Sets Assigned in the Summer of 2014

M.V. Iordache

Electrical Engineering, LeTourneau University

Posted on http://mviordache.name/ENGR6963

ENGR6963 Formal Verification

Homework Set 1

1. Express the following statements in propositional logic.
   (a) Let x denote "The system is live" and y "The system is bounded". Write a formula for "The system is neither live nor bounded".
   (b) Let x denote "The system is deadlock free" and y "The system is live". Write a formula for "The system is deadlock free but not live".
   (c) Let x denote "The system is repetitive", y "The system is live", and z "The system is responsive". Write a formula for "If the system is responsive, it is also live and repetitive".
   (d) Define appropriate variables and write a formula for "A triangle is equilateral if and only if it is isosceles and has one 60° angle".
2. Rewrite the following formulas so that they involve only $\wedge$, $\vee$, and $\neg$ connectives.
   (a) $(z \wedge \neg x) \rightarrow y$.
   (b) $z \leftrightarrow \neg y$.
3. Using a proof by contradiction show that $\neg((z \vee \neg x) \rightarrow (y \wedge z)) \models \neg(x \wedge y)$.
4. Show that $\neg(x \wedge y) \not\models \neg((z \vee \neg x) \rightarrow (y \wedge z))$.
5. Show that $((x \wedge y) \rightarrow z) \leftrightarrow (x \rightarrow (y \rightarrow z))$ is a tautology.
6. Express the following statements in first order logic.
   (a) "For all reachable markings μ there is at least one enabled transition t." Assume two atomic formulas $R(\mu)$ and $E(\mu, t)$ defined as follows: $R(\mu)$ is true when μ is reachable, and $E(\mu, t)$ is true when t is enabled by μ.
   (b) "There is a reachable marking μ for which $L\mu \leq b$." Assume an atomic formula $R(\mu)$ defined as above.
   (c) "For all reachable markings μ, $L_a\mu \leq b_a$ implies $L\mu \leq b$. Furthermore, $L_a D(\cdot, t_u) \leq 0$ for all uncontrollable transitions t_u." Assume two atomic formulas $R(\mu)$ and $U(t)$ defined as follows: $R(\mu)$ is true when μ is reachable, and $U(t)$ is true when t is uncontrollable.
   (d) Define appropriate atomic formulas and express the following statement in logic. "For all reachable markings μ that do not belong to the set F, and for all transitions t, there is a firing sequence σ enabled by μ that contains t so that for all markings reached by firing σ no marking is in the set F."
7. A formula in the prenex normal form has all quantifiers at the beginning of the formula. For example, $\forall\mu\forall t\exists k(I(\mu, t) \geq 0 \wedge O(\mu, t) \geq k)$ is in the prenex normal form. However, $\forall\mu\forall t(I(\mu, t) \geq 0 \wedge \exists k O(\mu, t) \geq k)$ is not in the prenex normal form. Write the following expressions in the prenex normal form.

(a) $\neg(\forall x P(x)) \vee \neg(\exists y Q(y))$.

(b) $(\exists x P(x)) \rightarrow (\exists y Q(y))$.

He revealeth the deep and secret things: he knoweth what is in the darkness, and the light dwelleth with him. Dan 2:22

ENGR6963 Formal Verification

Homework Set 2

1. Assume a Kripke structure M and the atomic propositions P, Q, and R. Determine in each case the logic (LTL, CTL, CTL*) of the following formulas.

   (a) $M, s_1 \models \mathbf{EF}\, R$.

   (b) $M, s_1 \models \mathbf{F}\, R$.

   (c) $M, s_1 \models \mathbf{AG}(R \rightarrow \mathbf{EF}\, Q)$.

   (d) $M, s_1 \models \mathbf{EFG}\, P$.

   (e) $M, s_1 \models \mathbf{A}((\mathbf{G}\, P) \vee (\mathbf{EFG}\, P))$.

2. The Kripke structure M shown in the figure is labeled by the atomic propositions P, Q, and R. Indicate whether the following formulas are satisfied.

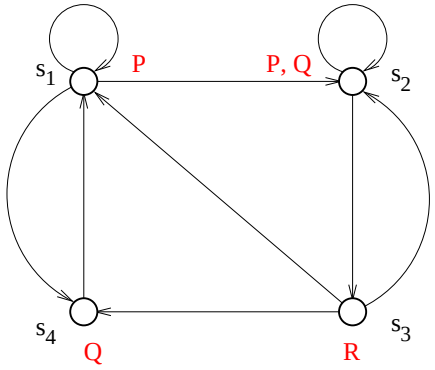
   (a) $M, s_1 \models \mathbf{EF}\, R$.

   (b) $M, s_1 \models \mathbf{F}\, R$.

   (c) $M, s_1 \models \mathbf{AG}(R \rightarrow \mathbf{EF}\, Q)$.

   (d) $M, s_1 \models \mathbf{EFG}\, P$.

   (e) $M, s_1 \models \mathbf{A}((\mathbf{G}\, P) \vee (\mathbf{EFG}\, P))$.



3. Assume a Kripke structure M and a state s of M. Write formulas expressing the following requirements. If possible, provide both LTL and CTL formulas.

   (a) From every state reachable from the state s it is possible to reach a state satisfying either P or Q.

   (b) From every state reachable from the state s it is impossible to reach a state in which all of P, Q, and R are true.

   (c) For every state reachable from the state s, if P is satisfied, either Q or R will be eventually satisfied.

4. Provide an example showing that $\mathbf{GF}\, P \rightarrow \mathbf{GF}\, Q$ is not equivalent to $\mathbf{GF}\, (P \rightarrow \mathbf{GF}\, Q)$.

Who being the brightness of *his* glory, and the express image of his person, and upholding all things by the word of his power, when he had by himself purged our sins, sat down on the right hand of the Majesty on high. Heb 1:3

ENGR6963 Formal Verification

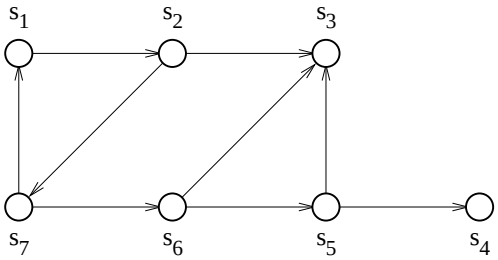
Homework Set 3

1. Consider two atomic propsitions p and q of a Kripke structure.
   (a) Write an LTL formula requiring that no state should satisfy both p and q.
   (b) Write an LTL formula requiring that whenever p is satisfied, the next state should satify q.
   (c) Write a CTL formula requiring the existence of a path on which whenever p is satisfied, the next state will satisfy q.
   (d) Write a CTL formula requiring the existence of a path $\pi = s_0 s_1 s_2 \ldots$ in which s_0, s_2, s_4, $\ldots$ satisfy p and s_1, s_3, s_5, $\ldots$ satisfy q.
   (e) Write a CTL formula requiring that any sequence of three states $s_{k-1}s_k s_{k+1}$ on any path π has the property that if s_k satisfies q, then at least one of s_{k-1} and s_{k+1} satisfies q.

2. Assume that proposition p stands for process P waiting to enter the critical section and proposition q stands for process Q being in the critical section. It is required that for any execution path $\pi = s_0 s_1 s_2 \ldots$, for any sequence of states $s_k s_{k+1} \ldots s_n$ satisfying p at all times, at most one of the states s_k, s_{k+1}, $\ldots s_n$ should satisfy q. The following formula is proposed to express this requirement: $\neg\mathbf{E}(p\mathbf{U}(q \wedge p \wedge \mathbf{E}(p\mathbf{U}(p \wedge q))))$.
   (a) Find a counterexample showing that the formula does not express correctly the requirement.
   (b) Correct the formula.

3. Assume that proposition p stands for process P waiting to enter the critical section and proposition q stands for process Q being in the critical section. Consider the following formula: $\varphi = \neg\mathbf{E}((p \wedge \neg q)\mathbf{U}((p \wedge q \wedge \mathbf{E}((p \wedge q)\mathbf{U}(p \wedge \neg q \wedge \mathbf{E}(p\mathbf{U}(p \wedge q)))))))$.
   (a) Assume the following sets of proporsitions are satisfied along a path $\pi = s_0 s_1 s_2 \ldots$: $\{p\}$, $\{p,q\}$, $\{p,q\}$, $\{p\}$, $\{p,q\}$, $\{p,q\}$, $\{p\}$, $\{p,q\}$, $\{p,q\}$, $\{p\}$, ... Could it be that s_0 satisfies φ?
   (b) Assume the following sets of proporsitions are satisfied along a path $\pi = s'_0 s'_1 s'_2 \ldots$: $\{p\}$, $\{p\}$, $\{p,q\}$, $\{p,q\}$, $\{p\}$, $\{p\}$, $\{p\}$, $\{p\}$, ... Could it be that s'_0 satisfies φ?
   (c) Describe in words the fairness constraint imposed by the formula φ.

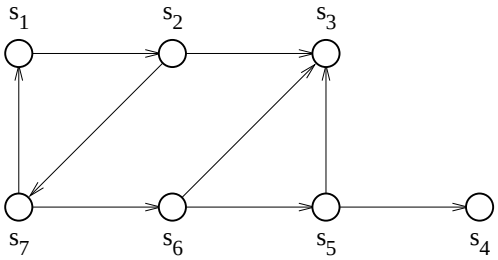
4. Consider the Kripke structure shown in the figure.
   (a) A deadlock state has no outgoing transition. Indicate the set of deadlock states Y_1.
   (b) Let $pre_\forall(Y) = \{s \in S : \forall s' \in S,\ s \to s' \Rightarrow s' \in Y\}$ denote the set of states s for which <u>all</u> outgoing transitions lead to states in Y. Find recursively the fixed point $\mu Y.Y \cup pre_\forall(Y)$. Use for $pre_\forall(\emptyset)$ the value of Y_1 determined at part (a).
   (c) Let δ be an atomic proposition that is true in all states of Y_1, that is, $[\delta] = Y_1$. Write a CTL formula φ in terms of δ so that $[\varphi] = Y$.



5. Consider the Kripke structure shown in the figure. Let δ be an atomic proposition satisfied by s_6 (that is, $[\delta] = \{s_6\}$). Let $post_{\exists}(Y) = \{s \in S : \exists s' \in Y,\ s' \to s\}$.

   (a) Find recursively the fixed point $\mu Y.Y \cup post_{\exists}(Y)$ using $Y_1 = [\delta]$.

   (b) Is it possible to write a CTL formula φ in terms of δ so that $[\varphi] = Y$?

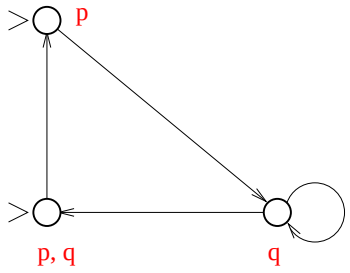


Every good gift and every perfect gift is from above, and cometh down from the Father of lights, with whom is no variableness, neither shadow of turning. Jas 1:17

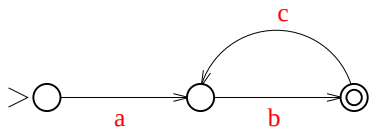
ENGR6963 Formal Verification

Homework Set 4

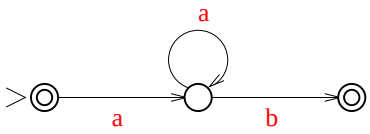
1. Let p and q be atomic propositions. Convert the Kripke structure to a Büchi automaton. Note that the Kripke structure has two initial states.



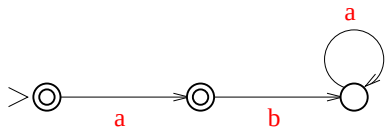
2. For all Büchi automata below that accept a nonempty language, express their accepted lamguage by means of an ω-regular expression.



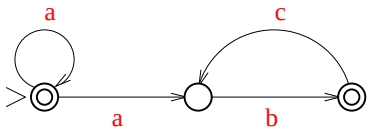
(a)



(b)



(c)



(d)

3. Let b_0, b_1, and b_2 be the bits of a register. Let b_0', b_1', and b_2' denote their value after the next active clock edge. Assume that at every active clock edge the bits are updated according to the equations $b_0' = b_0 \oplus b_1$ (where $\oplus$ is exclusive or), $b_1' = b_0 + b_1 + b_2$, and $b_2' = b_0 b_1 b_2$. Verify with nuXmv that for any initial value of the bits, there is no reachable state, other than possibly the initial state, in which all bits are 1. Attach your program and verification results.

4. Write a NuXMV program modeling the traffic lights of an intersection involving two roads. Assume no left turns and only red and green lights (no yellow light). Verify that that reasonable constraints are satisfied (such as not all lights are green at the same time). Attach the program and the verification results.

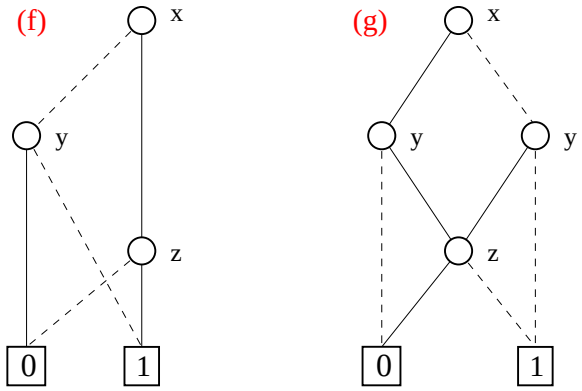
... in all things it behoved him to be made like unto *his* brethren, that he might be a merciful and faithful high priest in things *pertaining* to God, to make reconciliation for the sins of the people. (Heb 2:17)

ENGR6963 Formal Verification

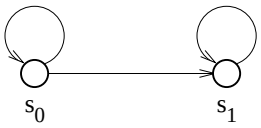
Homework Set 5

1. Consider the function $f = \overline{y}x + y\overline{z}$. Assume the variable order x, y, z.
   (a) Draw the binary decision tree of f.
   (b) Draw the ROBDD of f.

2. Let f and g be the functions implemented by the BDDs shown in the figure.
   (a) Write the expressions of f and g in terms of x, y, and z.
   (b) Obtain the ROBDD of $h = \overline{f} + g$.
   (c) Obtain the ROBDD of $\forall x.f$.
   (d) Obtain the ROBDD of $\exists y.f$.



3. Let S be the set of states of an automaton and R the transition relation, that is, $(s, s') \in R$ when there is a transition from s to s'. Note that $D = \{s \in S : \forall s' \in S, (s, s') \notin R\}$ is the set of deadlock states.
   (a) Given the BDD of R, indicate how BDD functions such as `apply` and `restrict` should be used in order to determine the BDD representing the states in D.
   (b) Apply your procedure on the automaton shown below.



   i. Assume the binary variable x encodes the two states: $x = 0$ denotes s_0, and $x = 1$ denotes s_1.
   ii. Determine the BDD B_R representing R.
   iii. Obtain B_D, the BDD of the set D, by applying to B_R functions such as `restrict` and `apply`.
   iv. Find the set D by applying the definition $D = \{s \in S : \forall s' \in S, (s, s') \notin R\}$, and verify that it is consistent with the BDD B_D that you determined.

4. Given an automaton of set of states S, a set $Y \subset S$ is a livelock if there is no transition from a state $s \in Y$ to a state $s' \notin Y$.

   (a) Write a first-order logic formula describing whether Y is a livelock.

   (b) Assuming a BDD representation of Y and of the transition relation R, indicate how you would apply BDD operations in order to determine whether Y is a livelock.

ENGR6963 Formal Verification

Homework Set 6

1. Write a NuXMV program that models the dining philosopher problem and verifies whether the philosophers have fair access to the "eat" state. Use at least three philosophers. Note that resources can be modeled by monitor places as in the file monitor2.smv. Attach the program and the verification results.

2. Write a NuXMV program modeling the traffic lights of an intersection involving two roads. Draw also the most relevant finite state machines implemented in the program.

   - Assume red, yellow, and green lights.
   - Assume no left turns.
   - The green light should be on for at least 10 clock cycles. The yellow light should be on for one clock cycle.
   - Use nondeterministically assigned variables to model incoming traffic. If there is incoming traffic on one of the two roads, and the light is red, the light should become green eventually. The light should eventually become green even if there is incoming traffic also on the other road.

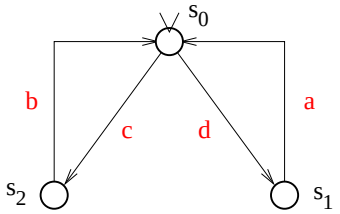
   Debug the program and verify that reasonable constraints are satisfied (such as not all lights are green at the same time). Attach the program and the verification results.

God is our refuge and strength, a very present help in trouble. (Psa 46:1)

ENGR6963 Formal Verification

Homework Set 7

1. (a) The file `fsm.pml` shows a few ways of implementing automata in Promela. Implement the following automaton using nondeterministic transitions. Assume all events are enabled.



   (b) Examine the automaton and verify that it has no event traps.

   (c) Verify in SPIN that the LTL formula $\mathbf{G}(s_0 \rightarrow \mathbf{F}s_2)$ is not satisfied.

   (d) Explain the results.

2. Assume a road crossing two train tracks. The train gate should be closed when trains are present or when they approach the crossing point on either of the two tracks. Write a Promela model of this problem. Use a separate process for each track. Each process should represent an automaton with several states, indicating whether a train is close, far, or present. Include an LTL formula requiring that always eventually the train gate is open. Verify the LTL formula in SPIN and interpret the results.

3. Convert to Promela the NuXMV model of sync1.smv. Use communication channels to pass variables between processes and the supervisor. Verify the Promela file in SPIN.

And, behold, there are last which shall be first, and there are first which shall be last. (Luk 13:30)

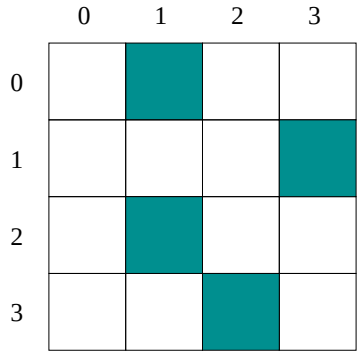
ENGR6963 Formal Verification

Homework Set 8

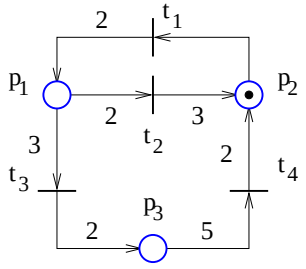
1. (Adapted from the examples of *Computer Lab 1: Model Checking and Logic Synthesis using Spin*, R. Murray et al, HYCON-EECI, Spring 2012.) A robot should navigate the maze shown in the figure. It should start from the position $(0, 0)$ and eventually reach the position $(3, 3)$. The robot cannot move on the diagonal, that is, the row number and the column number may not change at the same time. Use SPIN to find a trajectory that avoids the obstacles and reaches the desired end state.

   Hint: A possible solution would involve the follwong steps:

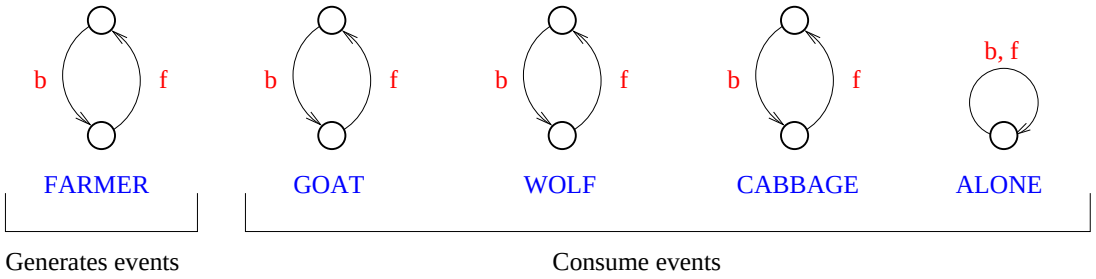
   - A process that randomly adds or subtracts 1 to the row number or the column number that describe the position of the robot.
   - An LTL formula that requires the desired state to be reached without reaching the forbidden states.
   - SPIN can verify the complement of the formula and provide a counterexample.



2. Verify in SPIN whether the following Petri net can reach a state at which $\mu_1 = \mu_2 = 1$ and $\mu_3 = 2$. If yes, use the simulation feature of SPIN to obtain the firing sequence leading to this state. Note the integer weights of the transition arcs. Hint: The error trail is easier to read if the Promela program uses printf to display messages indicating the transitions that are fired.



3. (Adapted from the examples of *Computer Lab 1: Model Checking and Logic Synthesis using Spin*, R. Murray et al, HYCON-EECI, Spring 2012.) Consider the following puzzle problem. A farmer has to get on the other side of a river a goat, a wolf, and a cabbage. The farmer can only use a small boat. The farmer can take with him into the boat only the cabbage or only one of the two animals. The wolf and the goat must not be together unless the farmer is present. The goat and the cabbage must not be together unless the farmer is present. The problem is to find a strategy to cross the river. Use SPIN to find a solution as follows.



(a) Let b and f denote the events associated with the farmer going back and forth across the river.

(b) The automaton modeling the farmer generates the events b and f. The other automata can use these events to change their state.

(c) To enforce the constraint that only one item or animal may be in the boat, only one automaton may respond to an event generated by the farmer automaton. (Atomic statements could be used to test and change an event variable without interruption.)

(d) The `alone` automaton models the situation when the farmer does not take anything with him in the boat.

(e) Write a Promela process for each automaton shown in the figure.

(f) Write an LTL formula so that a counterexample to the formula is a solution to the problem.

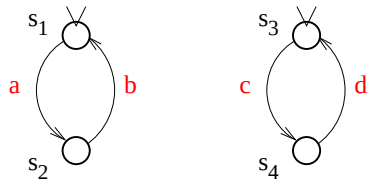
He hath made his wonderful works to be remembered: the LORD is gracious and full of compassion. (Ps 111:4)

ENGR6963 Formal Verification

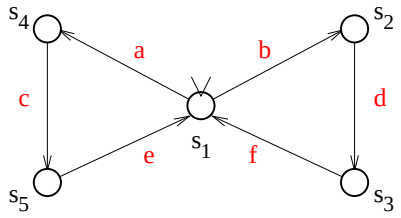
Homework Set 9

1. Assume p and q are atomic propositions. Mark all formulas that are closed under stuttering.
   (a) $\mathbf{GF}p$.
   (b) $\mathbf{G}(p \rightarrow \mathbf{F}q)$.
   (c) $\mathbf{G}(p \rightarrow \mathbf{XF}q)$.
   (d) $\mathbf{F}(p \rightarrow \mathbf{X}\,q)$.
   (e) $\mathbf{G}(p \rightarrow \mathbf{X}\,q)$.

2. (a) Implement in SPIN the following automata as two different processes.
   (b) Apply a progress label to the state s_2 and verify that SPIN reports a no-progress cycle.
   (c) Verify that the specification $\mathbf{GF}\,s_4$ is true only under the weak-fariness assumption.



3. (a) Implement in SPIN the following automaton.
   (b) Verify that the weak-fairness assumption does not guarantee $(\mathbf{GF}\,s_3) \wedge (\mathbf{GF}\,s_5)$.
   (c) Consider the assumption that if s_1 is infinitely often visited, then the events a and b take place infinitely often. Include this assumption into the LTL formula $(\mathbf{GF}\,s_3) \wedge (\mathbf{GF}\,s_5)$ and verify that under this assumption the formula is satisfied.



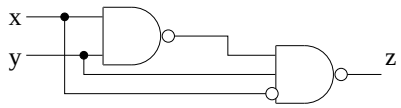
4. Implement in SPIN the dining philosopher problem. Use at least four philosophers. Verify that deadlock is possible. Hint: The resource places can be seen as the places of a supervisor. The file supervisor.pml demonstrates a fair supervision method.

I know that, whatsoever God doeth, it shall be for ever: nothing can be put to it, nor any thing taken from it: and God doeth *it*, that *men* should fear before him. (Ecc 3:14)

ENGR6963 Formal Verification

Homework Set 10

1. Assume that a SAT solver should verify whether there are values of x and y so that $z = 1$. Write a CNF formula involving x, y, and z in the format of the SAT problem. Auxiliary variables may be used.



2. Consider the pseudo-boolean formula $(2x_1 + 3x_2 + 4x_3 \leq 6) \wedge (x_1 + x_2 \geq x_3)$. Convert the formula to the CNF format of SAT problems. Follow this approach.
   (a) Find a ROBDD representing the formula. Assume the order x_1, x_2, x_3.
   (b) Using the ROBDD obtain the CNF encoding.

3. (a) Apply the bitwise encoding to $x_1 + x_2 + x_3 + x_4 \leq 1$ in order to obtain a CNF encoding (the form expected by SAT solvers).
   (b) How would you obtain the CNF encoding of $x_1 + x_2 + \ldots + x_n \leq k$? Illustrate your approach on $x_1 + x_2 + x_3 \leq 2$.

4. Consider the formula $c \wedge (a \vee \overline{c}) \wedge (\overline{a} \vee \overline{b}) \wedge (\overline{a} \vee b) \wedge (\overline{a} \vee c \vee d)$.
   (a) Sketch a resolution graph showing that the formula is UNSAT.
   (b) Find the MUS (the minimal unsatisfiable subformula).

5. Write the following formulas in the CNF format of SAT problems.
   (a) $\exists x_1, x_2 \, \forall y. \, (x_1 \vee y) \wedge (\overline{x_2} \vee \overline{y}) \wedge (x_1 \vee x_2)$.
   (b) $\exists x_1, x_2 \in \{0, 1\} \, \forall y \in \{1, 2, 3\}. \, (x_1 \vee (3y \leq 5)) \wedge (x_2 \vee (y \geq 3))$.

And he said, The things which are impossible with men are possible with God. Lk 18:27

ENGR6963 Formal Verification

Homework Set 11

1. Use the `cbmc` tool to verify the file `lock_example.c`. Use the `--no-winding-assertions` option to ensure that unwinding unbounded loops will not result in an error.

   (a) Verify that there is no error when the unwind limit is set to 1.

   (b) Verify that there is an error when the unwing limit is set to 2.

   (c) Use the error trace to identify the programming error. What is the programming error?

2. Use the `cbmc` tool to verify the following program. Use a high enough unwinding limit in order to expose the error. Use the `--no-winding-assertions` option to ensure that unwinding the recursion will not result in an error. Indicate what the error is and the command-line options for which `cbmc` reported it.

```
int f(int *ar, char x, int n) {
  if(x < n) {
    ar[x] = f(ar, x + 1, n);
    return (ar[x] + ar[x]/2);
  }
  return 2;
}

void main() {
  int array[200];
  f(array, 0, 200);
}
```

Jesus answered and said unto them, Verily I say unto you, If ye have faith, and doubt not, ye shall not only do this *which is done* to the fig tree, but also if ye shall say unto this mountain, Be thou removed, and be thou cast into the sea; it shall be done. (Mat 21:21)

ENGR6963 Formal Verification

Homework Set 12

1. In each case, indicate whether there are p, q, and r satisfying the given logical sentences for all possible values of the free variables.

   (a) $(p(x) \vee q(x,y)) \wedge r(y)$
   $\neg q(a.a)$
   $\neg p(a) \wedge (q(x,b) \rightarrow r(x))$

   (b) $(p(x) \vee q(x,y)) \wedge r(y)$
   $\neg r(x) \vee \neg q(y.x)$

   (c) $(p(x) \vee q(x,y)) \wedge r(y)$
   $\exists x.\neg q(x.x)$
   $\exists x.\exists y.\neg p(x) \wedge (q(x,y) \rightarrow r(x))$

2. Assume the formula $(x^2 = y \vee e^x < y) \wedge (y < -3 \vee 3x^2 + y = -1)$, where x and y are real variables. Assume a decision procedure is available for checking the existence of solutions to equations in real numbers subject to conjunctions of inequality constraints. Illustrate *lazy translation into SAT* on the given formula, showing how it proves satisfiability or unsatisfiability.

He will fulfil the desire of them that fear him: he also will hear their cry, and will save them. Psa 145:19