CSC410

Exam 2

Date: December 1, 2023
weight: 20% of course mark, length: 110 minutes

First Name: _____

Last Name: _____

Student Number: _____

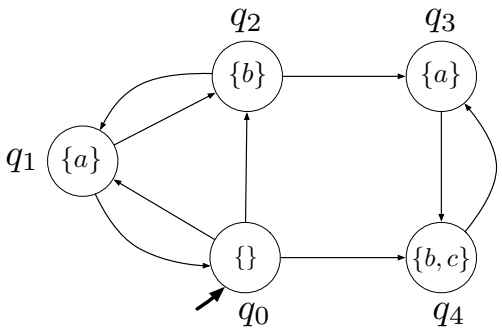
Read Before You Start

- Write your name and student number, and please do it so that it matches your record on Quercus.
- This exam contains **5 problems** (some with multiple subproblems). The idea is to distribute the credit more, so that no one part is worth too much of the total mark.
- Problem 5(c) and 5(d) are for graduate students only. Undergraduate students do not need their credit for a full mark.
- The exam totals 130 points for undergraduate students and 150 points for graduate students.
- The exam is designed so that the answers require **no English** words. You are welcome to write English sentences, but be warned that they will not be graded.
- The exam may look long, but most questions have very short answers, and the rest also do not involve a lot of writing. The generous 110-minute time is there to remove any stress from this process. It is not an indication that this exam requires 2 hours of work!
- You **may not** use any known (from the text book) LTL or CTL equalities in your answers for problems 2 and 3. If you want to use one, you will have to prove it.

Temporal Logic and Model Checking

Problem 1: Model Checking (20 points)

Consider the labeled transition system below where each state is labeled with the set of atomic propositions that it satisfies.



Answer the two questions below, but keep in mind that a wrong answer does carry a negative weight. For example, "correct answer 1, correct answer 2" will get more marks than "correct answer 1, correct answer 2, wrong answer 1". Otherwise, you can just list all states for every item, and you are guaranteed to include all the correct answers! No justification is necessary. We will only mark the list of states.

(a) List the states that satisfy the LTL formula $\neg\Box(a \mathsf{U} b)$. Be careful with the negation in the LTL formula!

states satisfying $(a \mathsf{U} b)$: q_2, q_3, q_4
states satisfying $\Box(a \mathsf{U} b)$: q_3, q_4
states satisfying neither $\Box(a \mathsf{U} b)$ nor $\neg\Box(a \mathsf{U} b)$: q_1, q_2
states satisfying $\neg\Box(a \mathsf{U} b)$: q_0

(b) List the states that satisfy the CTL formula $\exists\big(b \mathsf{U} \forall(a \mathsf{U} c)\big)$.

states satisfying $\forall(a \mathsf{U} c)$: q_3, q_4
states satisfying $\exists\big(b \mathsf{U} \forall(a \mathsf{U} c)\big)$: q_2, q_3, q_4

Problem 2: LTL (30 points)

(a) Consider a system with two processes, P_1 and P_2, one processor, and a *round robin* style scheduler that keeps switching between P_1 and P_2 at every 2 time step. Let atomic proposition p_1 be true when process P_1 is running and atomic p_2 be true when process P_2 is running. We start with process P_1 and we want only one process to be running at a time.

Write an LTL formula that specifies the valid runs of the system under the above *round robin* scheduler. In other words, you want to permit only executions of the form:

$$(p_1 \wedge \neg p_2) \quad (p_1 \wedge \neg p_2) \quad (p_2 \wedge \neg p_1) \quad (p_2 \wedge \neg p_1) \quad (p_1 \wedge \neg p_2) \quad (p_1 \wedge \neg p_2 \quad (p_2 \wedge \neg p_1) \quad (p_2 \wedge \neg p_1) \ \ldots$$

$p_1 \wedge \bigcirc p_1$

$\square\big(\neg(p_1 \wedge p_2)\big)$

$\square\big((p_1 \wedge \bigcirc p_1) \implies \bigcirc\bigcirc p_2)\big)$

$\square\big((p_2 \wedge \bigcirc p_2) \implies \bigcirc\bigcirc p_1)\big)$

$\square\big((p_1 \wedge \bigcirc p_2) \implies \bigcirc\bigcirc p_2)\big)$

$\square\big((p_2 \wedge \bigcirc p_1) \implies \bigcirc\bigcirc p_1)\big)$

(b) Prove or disprove: if a path π satisfies

$$\square\big((a \implies \Diamond b) \wedge \square(a \implies \Diamond b)\big)$$

then a is guaranteed to happen infinitely often on π. In other words, π also satisfies $\square\Diamond a$.

False. Counterexample:

$$\neg a \wedge \neg b \quad \neg a \wedge \neg b \quad \neg a \wedge \neg b \ \ldots$$

which satisfies the above formula, and a never occurs on it.

Problem 3: CTL (30 points)

(a) Prove or disprove:

$$\neg\forall\Box\varphi \equiv \exists\Diamond\neg\phi$$

Here is a clean logical proof:

$$\begin{aligned}
TS \models \neg\forall\Box\varphi &\iff \text{(by semantics of } \neg\text{)} \\
\neg(TS \models \forall\Box\varphi) &\iff \text{(by semantics of } \forall\text{)} \\
\neg(\forall\pi \in Paths(TS) :\ \pi \models \Box\varphi) &\iff \text{(by semantics of } \Box\text{)} \\
\neg(\forall\pi \in Paths(TS) :\ \forall m \geq 0 :\ \pi[m] \models \varphi) &\iff \text{by basic logic} \\
\exists\pi \in Paths(TS) :\ \exists m \geq 0 :\ \neg(\pi[m] \models \varphi) &\iff \text{by semantics of } \neg \\
\exists\pi \in Paths(TS) :\ \exists m \geq 0 :\ \pi[m] \models \neg\varphi &\iff \text{by semantics of } \Diamond \\
\exists\pi \in Paths(TS) :\ \pi \models \Diamond\neg\varphi &\iff \text{by semantics of } \exists \\
TS \models \exists\Diamond\neg\varphi &\iff \text{by semantics of } \exists
\end{aligned}$$

(b) Formalize the following English specification as a CTL formula using the atomic propositions $\{req, acc, den\}$: "Every time the system receives a request, the following three choices are available to it: it can immediately deny it in the next step, immediately accept it in the next step, or ignore it. But, if ignored, no future acceptances or denials should be issued.". Note that ignoring a request means that neither an acceptance nor a denial is issued in the next step. Hence, you **may not** assume that $\neg acc \rightarrow den$ and vice versa.

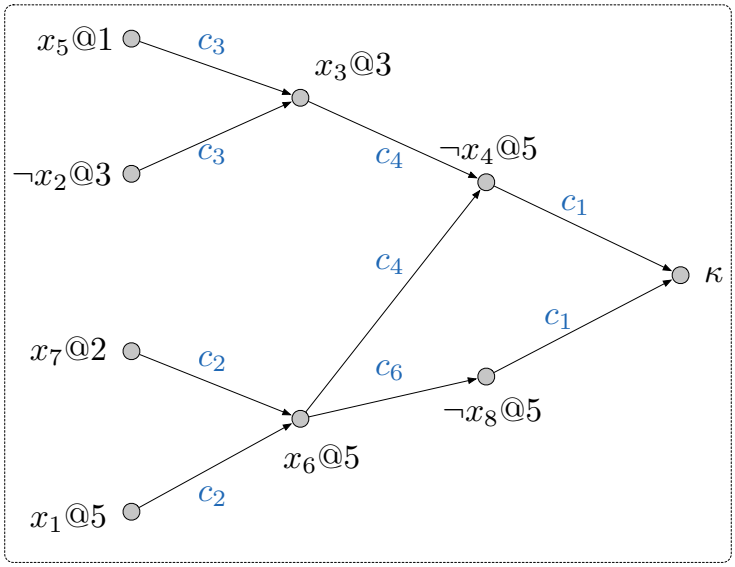
$$\forall\Box(req \implies (\exists\bigcirc acc \wedge \exists\bigcirc den \wedge \exists\bigcirc\forall\Box(\neg acc \wedge \neg den))$$

Problem 4: DPLL Conflict Clause Learning (30 points)

Consider a formula F that is being checked for satisfiability in CNF using the following set of clauses:

$$c_1 = x_4 \vee x_8 \qquad c_2 = \neg x_1 \vee x_6 \vee \neg x_7 \qquad c_3 = x_2 \vee x_3 \vee \neg x_5$$
$$c_4 = \neg x_3 \vee \neg x_4 \vee \neg x_6 \qquad c_5 = \neg x_2 \vee \neg x_3 \vee \neg x_5 \qquad c_6 = \neg x_6 \vee \neg x_8$$

Consider a hypothetical snapshot of the DPLL algorithm where we inherit the following implication graph from decisions previously made at levels 1 and 2 of the DPLL algorithm, and we add a current decision node at level 5.



(a) Complete the above implication graph (in place) for level 5 to get a conflict. You will notice that part of this graph would already exist at level 3. That is intentional.

(b) Produce **all** *asserting* clauses that could be learned by the DPLL algorithm before it backtracks. In the case of each clause, indicate the level to which the algorithm would backtrack after learning the clause. To justify your answer, write down the sequence of binary resolution steps (and all the intermediate clauses) that gets you to your asserting clause. For clarity, label each clause that you use with its identifier from the list, e.g. $c_5 : \neg x_2 \vee \neg x_3 \vee \neg x_5$ instead of just $\neg x_2 \vee \neg x_3 \vee \neg x_5$. What is the smallest asserting clause the algorithm can learn?

Hint: Now you need to use the information about what part of the graph existed before level 5 and what part is new. You do not need to process the existing parts to find an asserting clause at level 5.

$$\frac{c_6 : \neg x_6 \vee \neg x_8 \qquad c_1 : x_4 \vee x_8}{x_4 \vee \neg x_6}$$

$$\frac{x_4 \vee \neg x_6 \qquad c_4 = \neg x_3 \vee \neg x_4 \vee \neg x_6}{\neg x_3 \vee \neg x_6}$$

$\neg x_3 \vee \neg x_6$ is the first asserting clause. It corresponds to the UIP $x_6@5$, which is on every path from $x_1@5$ to the conflict node. If we backtrack to level 3 with this clause, it will become a unit clause resulting to a decision $\neg x_6@3$ at that level.

If we continue:

$$\frac{\neg x_3 \vee \neg x_6 \qquad c_2 : \neg x_1 \vee x_6 \vee \neg x_7}{\neg x_1 \vee \neg x_3 \vee \neg x_7}$$

$\neg x_1 \vee \neg x_3 \vee \neg x_7$ is another asserting clause that the algorithm can learn, and it will backtrack to level 3 after learning it. This one corresponds to the trivial UIP $x_1@5$. This clause will become unit at level 3, and result in a decision $\neg x_1@3$.

We can continue and learn the trivial $\neg x_1 \vee x_2 \vee \neg x_5 \vee \neg x_7$, but this is unnecessary. The clause above is the better stand in for this clause.

Decision Procedures

Problem 5: SAT Encoding

(20 + 20 points) The goal of this problem is to select a number of teachers to cover a number of subjects. This is one problem, broken into parts so that you may earn partial credits and have some guidance.

- Let $T = \{T_1, \ldots, T_n\}$ a set of teachers.
- Let $S = \{S_1, \ldots, S_m\}$ be a set of subjects.
- Each subject $s \in S$ is taught by a set $T(s)$ of the teachers ($T(s) \subset T$). Additionally, you may want to make use of the function $subsets(T, n) = \{T' \subseteq T \mid |T'| = n\}$ where $|T|$ denotes the cardinality of the set T.

Given a natural number $k \leq n$, we want to check if it is possible to recruit *at most* k teachers and cover all the subjects. We use the following boolean variables:

- Teachers t_i, for $i = 1, \ldots, n$, where t_i is true if and only if the corresponding teacher is recruited.
- Subjects s_j for $j = 1, \ldots, m$, where s_j is true if and only if the corresponding subject is covered (taught).

You are only allowed to use these variables in your answers to the questions below. Follow the steps (a)-(d) to produce the encoding for this problems.

(a) (10 points) Write a constraint (formula in propositional logic) that ensures that a subject is considered covered only if at least one teacher who covers it is recruited. Think of this as relating the variables s_i and t_i together. Your formula should be in CNF (conjunctive normal form).

$$\bigwedge_{j=1}^{m} \left(s_j \implies \bigvee_{t \in T(s_j)} t \right)$$

(b) (10 points) Write a constraint (formula in propositional logic) that ensures that **all subjects** in S are covered.

$$\bigwedge_{j=1}^{m} s_j$$

(c) **(graduate problem: 10 points)** Write a constraint (formula) that ensures that **at most** k teachers are recruited. Your formula should be in CNF (conjunctive normal form).

$$\bigwedge_{c \in subsets(T, k+1)} \bigvee_{t \in c} \neg t$$

(d) **(graduate problem: 10 points)** We have so far solved the problem of recruiting *at most* k teachers. Now assume that we want *exactly* k teachers to be recruited, such that all the subjects are taught. What other constraints are required (in addition to the ones you listed in (a),(b), and (c) parts)? Write a CNF formula for the additional constraints.

$$\bigwedge_{c \in subsets(T, n-k+1)} \bigvee_{t \in c} t$$

CTL Semantics

$s \models a$ iff $a \in L(s)$

$s \models \neg \Phi$ iff not $s \models \Phi$

$s \models \Phi \wedge \Psi$ iff $(s \models \Phi)$ and $(s \models \Psi)$

$s \models \exists \varphi$ iff $\pi \models \varphi$ for some $\pi \in Paths(s)$

$s \models \forall \varphi$ iff $\pi \models \varphi$ for all $\pi \in Paths(s)$

$\pi \models \bigcirc \Phi$ iff $\pi[1] \models \Phi$

$\pi \models \Phi \mathsf{U} \Psi$ iff $\exists j.\ \pi[j] \models \Psi \wedge \forall 0 \leq k < j.\ \pi[k] \models \Phi$

LTL Semantics

$\sigma \models \text{true}$

$\sigma \models a$ iff $a \in A_0$ (i.e., $A_0 \models a$)

$\sigma \models \varphi_1 \wedge \varphi_2$ iff $\sigma \models \varphi_1$ and $\sigma \models \varphi_2$

$\sigma \models \neg \varphi$ iff $\sigma \not\models \varphi$

$\sigma \models \bigcirc \varphi$ iff $\sigma[1 \ldots] = A_1 A_2 A_3 \ldots \models \varphi$

$\sigma \models \varphi_1 \mathsf{U} \varphi_2$ iff $\exists j \geqslant 0.\ \sigma[j \ldots] \models \varphi_2$ and $\sigma[i \ldots] \models \varphi_1$, for all $0 \leqslant i < j$

$\sigma \models \Box \varphi$ iff $\forall i.\ \sigma[i...] \models \varphi$

$\sigma \models \Diamond \varphi$ iff $\exists i \geq 0.\ \sigma[i...] \models \varphi$