CSC410

CTL Class Examples

(with solutions)

Writing CTL Specifications

(a) **Warm up: Mutual Exclusion**: processes 1 and 2 are never simultaneously in their critical sections. Like before, use c_1 and c_2 as your atomic propositions.

Solution.

$$\forall\Box(\neg c_1 \vee \neg c_2)$$

(b) **Mutual Exclusion and Nonstarvation**: The processes 1 and 2 access the critical section in a way that they permit another process to have the possibility of accessing it in between their accesses.

Solution. The solution can be either:

$$\forall\Box(c_i \implies \forall\Diamond\neg c_i) \qquad \text{or} \qquad \forall\Box(c_i \implies \exists\Diamond\neg c_i)$$

depending on the interpretation of the English sentence.

The first says that P_i *must* exist on all futures, the second that it *exists* in some future (and opens the possibility of another process entering). The first is the stronger requirement: every state has at least one path, so "all paths leave" implies "some path leaves". The converse fails: in a system where P_i can stay in its critical section forever on one path, but can also leave on another, the second formula holds and the first does not. The ambiguitiy of English here is that it does not address the branching nature of the computation at all.

The second formula cannot be expressed in LTL

(c) **Robot Mission Planning**: a warehouse robot executes a fixed plan in an environment that can cause nondeterminism (hence branching) in the behaviour of the robot. Let atomic proposition $goal$ indicate that the robot is docked at its charging station and atomic proposition $hazard$ indicate that the robot is in a restricted zone. A *plan* is

- *weak*, if some outcome reaches the goal.
- *strong* if all outcomes reach the goal.
- *strong cyclic* if the robot is never stuck; i.e. each partial execution can always be extended to reach the goal.

Specify, for the initial state:

(i) the plan is weak;
(ii) the plan is weak, and avoids the restricted zone until the goal is reached;

(iii) the plan is strong, and avoids the restricted zone until the goal is reached;
(iv) the plan is strong cyclic.

Which of these can be expressed in LTL, and how are they related by implication?

Solution.

(i) $\exists\Diamond\, goal$
(ii) $\exists(\neg hazard \,\mathsf{U}\, goal)$
(iii) $\forall(\neg hazard \,\mathsf{U}\, goal)$
(iv) $\forall\Box\exists\Diamond\, goal$

Only (iii) is expressible in LTL, as $\neg hazard \,\mathsf{U}\, goal$. Implications: (iii) $\Rightarrow$ (ii) $\Rightarrow$ (i) (every state has a path), and (iv) $\Rightarrow$ (i) ($\forall\Box$ includes the initial state). (iii) and (iv) are incomparable: if every path reaches the goal and the robot can then drift into a dead end, (iii) holds but (iv) fails; if the robot can wander forever while the goal stays reachable, (iv) holds but (iii) fails.

Parts (i), (ii) and (iv) cannot be expressed in LTL

(d) **Self-Healing Service**: a server manages a service instance and restarts it when it fails. Use $down$ for when the server has failed and its negation for when it is running. We may be interested in some vague property like:

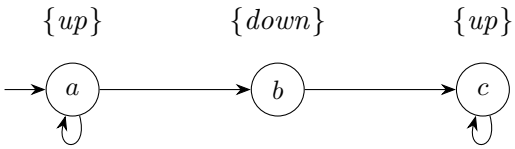
The server suffers transient faults, but they are bounded in number.

We can streamline it better, by specifying:

(i) the service may fail, but every failure is followed by a recovery;
(ii) on every execution, the service eventually stays up for good;
(iii) the system always reaches a state from which it is guaranteed to stay up;
(iv) What is the difference between the last two items?

Solution.

(i) $\exists\Diamond\, down \;\wedge\; \forall\Box(down \implies \forall\Diamond\, up)$ (the first conjunct guards against vacuity: a service that never fails satisfies the second trivially)
(ii) $\Diamond\Box\, up$ (an LTL formula: it is required on every path)
(iii) $\forall\Diamond\forall\Box\, up$
(iv) $\forall\Diamond\forall\Box\, up \Rightarrow \Diamond\Box\, up$: informally, every path reaches a state s with $s \models \forall\Box\, up$, and it stays up from there on. Conversely, consider the system below, where a is the initial state. Every path is a^ω or $a^k b\, c^\omega$, so it is eventually up for good, and hence every path satisfies $\Diamond\Box\, up$. But, $a \not\models \forall\Box\, up$ (since $a \to b$), and the path a^ω never visits c, the only state satisfying $\forall\Box\, up$.



Parts (i) and (iii) cannot be expressed in LTL (in (i), because of $\exists\Diamond\, down$; its second conjunct is LTL: $\Box(down \to \Diamond\, up)$). Part (ii) cannot be expressed in CTL.

(e) **Task-Processing Workflow**: A server is processing tasks. For each task, use $done$ for the task is finished and closed, $pending$ for some work for the task is still pending, and $ready_t$ for the task is enabled (i.e. ready to run). Imagine we have two tasks, so all these atomic propositions have subscripts 1 or 2. Specify the following three requirements typical *soundness* requirements:

(i) *option to complete*: from every reachable state the task can still be closed;
(ii) *proper completion*: when the task is closed, nothing is pending;
(iii) *no dead tasks*: every task can be executed in some run.

Solution.

(i) $\forall\Box\exists\Diamond\, done$
(ii) $\forall\Box(done \implies \neg pending)$
(iii) $\exists\Diamond\, ready_{pay} \wedge \exists\Diamond\, ready_{ship} \wedge \exists\Diamond\, ready_{refund}$

Parts (i) and (iii) cannot be expressed in LTL; only (ii) can: $\Box(done \to \neg pending)$

(f) **Traffic Light**: If the light is red and at some point in the future switches to yellow, then there is a possibility that it will eventually turn to green.

Solution.

$$\forall\Box\big((r \wedge \exists \bigcirc y) \implies \exists \bigcirc (y \wedge \exists\Diamond\, green)\big)$$

and, if we wanted to enforce that the green must come right after the end of the yellow period?

$$\forall\Box\big((r \wedge \exists \bigcirc y) \implies \exists \bigcirc (y \wedge \exists(y \,\mathsf{U}\, green))\big)$$

Proving/Disproving CTL Equalities

Notation (as on the slides). s is a state and $\pi = \pi[0]\pi[1]\ldots$ a path; $\pi[i\ldots]$ is its suffix from i; $\Phi \equiv \Psi$ iff $\forall s.\ (s \models \Phi \iff s \models \Psi)$. For a state formula Φ, $\pi \models \Phi$ iff $\pi[0] \models \Phi$.

(At) $s \models a \iff a \in L(s)$
($\wedge$) $s \models \Phi \wedge \Psi \iff s \models \Phi$ and $s \models \Psi$
($\neg$) $s \models \neg\Phi \iff s \not\models \Phi$
($\vee$) $s \models \Phi \vee \Psi \iff s \models \Phi$ or $s \models \Psi$ (derived: $\Phi \vee \Psi := \neg(\neg\Phi \wedge \neg\Psi)$)
($\exists$) $s \models \exists\varphi \iff \pi \models \varphi$ for some $\pi \in Paths(s)$
($\forall$) $s \models \forall\varphi \iff \pi \models \varphi$ for all $\pi \in Paths(s)$
($\bigcirc$) $\pi \models \bigcirc\Phi \iff \pi[1\ldots] \models \Phi$
(U) $\pi \models \Phi \,\mathsf{U}\, \Psi \iff \exists j.\ \pi[j\ldots] \models \Psi$ and $\pi[k] \models \Phi$ for all $0 \le k < j$
($\Diamond$) $\pi \models \Diamond\Phi \iff \exists i \ge 0.\ \pi[i] \models \Phi$ (derived: $\Diamond\Phi := true \,\mathsf{U}\, \Phi$)

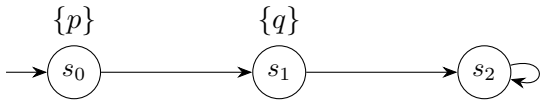
(a) $\forall\Diamond\Phi \wedge \forall\Diamond\Psi \equiv \forall\Diamond(\Phi \wedge \Psi)$

Solution. Not equivalent.

($\Leftarrow$) holds:

$$
\begin{aligned}
& s \models \forall\Diamond(\Phi \wedge \Psi) \\
\iff & \forall \pi \in Paths(s).\ \pi \models \Diamond(\Phi \wedge \Psi) && [\forall] \\
\iff & \forall \pi \in Paths(s).\ \exists i \geq 0.\ \pi[i] \models \Phi \wedge \Psi && [\Diamond] \\
\iff & \forall \pi \in Paths(s).\ \exists i \geq 0.\ (\pi[i] \models \Phi \ \wedge\ \pi[i] \models \Psi) && [\wedge] \\
\implies & \forall \pi \in Paths(s).\ \big((\exists i \geq 0.\ \pi[i] \models \Phi) \wedge (\exists i \geq 0.\ \pi[i] \models \Psi)\big) && [\exists \text{ forward-distributes over } \wedge] \\
\iff & \big(\forall \pi \in Paths(s).\ \exists i \geq 0.\ \pi[i] \models \Phi\big) \wedge \big(\forall \pi \in Paths(s).\ \exists i \geq 0.\ \pi[i] \models \Psi\big) && [\forall \text{ distributes over } \wedge] \\
\iff & \big(\forall \pi \in Paths(s).\ \pi \models \Diamond\Phi\big) \wedge \big(\forall \pi \in Paths(s).\ \pi \models \Diamond\Psi\big) && [\Diamond] \\
\iff & s \models \forall\Diamond\Phi \ \wedge\ s \models \forall\Diamond\Psi && [\forall] \\
\iff & s \models \forall\Diamond\Phi \wedge \forall\Diamond\Psi && [\wedge]
\end{aligned}
$$

($\Rightarrow$) fails. Let s_0 be the initial state of the system below, with $\Phi = p$ and $\Psi = q$.



$$
\begin{aligned}
& s_0 \models \forall\Diamond p \wedge \forall\Diamond q && \Leftarrow\ Paths(s_0) = \{s_0 s_1 s_2^\omega\} \ \wedge\ p \in L(s_0) \ \wedge\ q \in L(s_1) && [\forall, \Diamond, \wedge, \text{At}] \\
& s_0 \not\models \forall\Diamond(p \wedge q) && \Leftarrow\ \neg\exists i \geq 0.\ \{p, q\} \subseteq L\big((s_0 s_1 s_2^\omega)[i]\big) && [\forall, \Diamond, \wedge, \text{At}]
\end{aligned}
$$

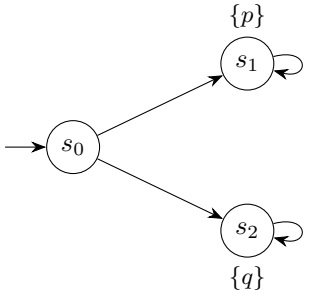
(b) $\forall\Diamond\Phi \vee \forall\Diamond\Psi \equiv \forall\Diamond(\Phi \vee \Psi)$

Solution. Not equivalent.

($\Rightarrow$) holds:

$$
\begin{aligned}
& s \models \forall\Diamond\Phi \vee \forall\Diamond\Psi \\
\iff & s \models \forall\Diamond\Phi \ \vee\ s \models \forall\Diamond\Psi && [\vee] \\
\iff & \big(\forall \pi \in Paths(s).\ \pi \models \Diamond\Phi\big) \vee \big(\forall \pi \in Paths(s).\ \pi \models \Diamond\Psi\big) && [\forall] \\
\implies & \forall \pi \in Paths(s).\ (\pi \models \Diamond\Phi \ \vee\ \pi \models \Diamond\Psi) && [\forall \text{ forward-contracts over } \vee] \\
\iff & \forall \pi \in Paths(s).\ \big((\exists i \geq 0.\ \pi[i] \models \Phi) \vee (\exists j \geq 0.\ \pi[j] \models \Psi)\big) && [\Diamond] \\
\iff & \forall \pi \in Paths(s).\ \exists i \geq 0.\ (\pi[i] \models \Phi \ \vee\ \pi[i] \models \Psi) && [\exists \text{ distributes over } \vee] \\
\iff & \forall \pi \in Paths(s).\ \exists i \geq 0.\ \pi[i] \models \Phi \vee \Psi && [\vee] \\
\iff & \forall \pi \in Paths(s).\ \pi \models \Diamond(\Phi \vee \Psi) && [\Diamond] \\
\iff & s \models \forall\Diamond(\Phi \vee \Psi) && [\forall]
\end{aligned}
$$

($\Leftarrow$) fails. Let s_0 be the initial state of the system below, with $\Phi = p$ and $\Psi = q$.



$$s_0 \models \forall\Diamond(p \vee q) \quad \Longleftarrow \quad Paths(s_0) = \{s_0 s_1^\omega,\ s_0 s_2^\omega\} \wedge p \in L(s_1) \wedge q \in L(s_2) \qquad [\forall, \Diamond, \vee, \text{At}]$$

$$s_0 \not\models \forall\Diamond p \vee \forall\Diamond q \quad \Longleftarrow \quad \neg\exists i.\ p \in L\big((s_0 s_2^\omega)[i]\big) \wedge \neg\exists i.\ q \in L\big((s_0 s_1^\omega)[i]\big) \qquad [\vee, \forall, \Diamond, \text{At}]$$

(c) $\exists\Diamond\Phi \vee \exists\Diamond\Psi \equiv \exists\Diamond(\Phi \vee \Psi)$

Solution. Equivalent. First a lemma, which can also be inlined (it is the same chain as for $\Diamond$ and $\vee$ in LTL).

Lemma 1. For every path π: $\pi \models \Diamond\Phi \vee \pi \models \Diamond\Psi \iff \pi \models \Diamond(\Phi \vee \Psi)$.

$$\begin{aligned}
& \pi \models \Diamond\Phi \vee \pi \models \Diamond\Psi \\
\iff & (\exists i \geq 0.\ \pi[i] \models \Phi) \vee (\exists j \geq 0.\ \pi[j] \models \Psi) && [\Diamond] \\
\iff & \exists i \geq 0.\ (\pi[i] \models \Phi \vee \pi[i] \models \Psi) && [\exists \text{ distributes over } \vee] \\
\iff & \exists i \geq 0.\ \pi[i] \models \Phi \vee \Psi && [\vee] \\
\iff & \pi \models \Diamond(\Phi \vee \Psi) && [\Diamond]
\end{aligned}$$

Now the equality:

$$\begin{aligned}
& s \models \exists\Diamond\Phi \vee \exists\Diamond\Psi \\
\iff & s \models \exists\Diamond\Phi \vee s \models \exists\Diamond\Psi && [\vee] \\
\iff & \big(\exists\pi \in Paths(s).\ \pi \models \Diamond\Phi\big) \vee \big(\exists\pi' \in Paths(s).\ \pi' \models \Diamond\Psi\big) && [\exists] \\
\iff & \exists\pi \in Paths(s).\ (\pi \models \Diamond\Phi \vee \pi \models \Diamond\Psi) && [\exists \text{ distributes over } \vee] \\
\iff & \exists\pi \in Paths(s).\ \pi \models \Diamond(\Phi \vee \Psi) && [\text{Lemma 1}] \\
\iff & s \models \exists\Diamond(\Phi \vee \Psi) && [\exists]
\end{aligned}$$