# COMP2620/6262 (Logic) Tutorial

Week 8

Semester 1, 2025

## Tutorial Quiz

In each tutorial, apart from week 2, there is a short quiz on skills practised in the previous tutorial. Your top 7 quiz attempts, out of the 9 available, will collectively count for 50% of your final mark.

This week's quiz is on **tableaux** for propositional logic. Your tutor will hand out blank paper, on which you should clearly write your university ID and name. Your tutor will also hand out paper with all tableaux rules for propositional logic. They will then write a set of signed propositions on the whiteboard. You should construct a *complete* tableaux for this set of signed propositions, continuing every branch until it closes or terminates open. You should remember to number your lines; to label on the right each new signed proposition by which lines justify it; and to cross each branch that can close, with line justifications beside any crosses. You do not need to explicitly extract a satisfying model. You will have **fourteen minutes** to construct this tableau.

You are not permitted to have any other resource on the table during this quiz, including any electronic device. If you finish your quiz before time elapses you may put your hand up and your tutor will collect your sheet. Once you have done this, you may get a device out and start work silently on this week's questions. If you are still working when time elapses you must stop writing immediately and let your tutor collect your paper.

## This Week's Exercises

This tutorial involves the tableaux rules, with branching rules for quantifiers, for first order logic:

$$\frac{\mathbf{T}:\bot}{\times} \qquad \frac{\mathbf{T}:\neg\varphi}{\mathbf{F}:\varphi} \qquad \frac{\mathbf{F}:\neg\varphi}{\mathbf{T}:\varphi} \qquad \frac{\mathbf{T}:\varphi\vee\psi}{\mathbf{T}:\varphi \quad\quad \mathbf{T}:\psi} \qquad \frac{\mathbf{F}:\varphi\vee\psi}{\begin{array}{c}\mathbf{F}:\varphi\\ \mathbf{F}:\psi\end{array}}$$

$$\frac{\mathbf{T}:\varphi\wedge\psi}{\begin{array}{c}\mathbf{T}:\varphi\\ \mathbf{T}:\psi\end{array}} \qquad \frac{\mathbf{F}:\varphi\wedge\psi}{\mathbf{F}:\varphi \quad\quad \mathbf{F}:\psi} \qquad \frac{\mathbf{T}:\varphi\to\psi}{\mathbf{F}:\varphi \quad\quad \mathbf{T}:\psi} \qquad \frac{\mathbf{F}:\varphi\to\psi}{\begin{array}{c}\mathbf{T}:\varphi\\ \mathbf{F}:\psi\end{array}}$$

$$\frac{\mathbf{T}:\forall x\varphi}{\begin{array}{c}\mathbf{T}:\varphi[a_1/x]\\ \mathbf{T}:\varphi[a_2/x]\\ \vdots\\ \mathbf{T}:\varphi[a_n/x]\end{array}} \qquad \frac{\mathbf{F}:\exists x\varphi}{\begin{array}{c}\mathbf{F}:\varphi[a_1/x]\\ \mathbf{F}:\varphi[a_2/x]\\ \vdots\\ \mathbf{F}:\varphi[a_n/x]\end{array}}$$

where $a_1, \ldots, a_n$ are all terms (= variables) in the tableau appearing free before or after this line. If no variables appear free before this line, the conclusion is $\varphi[a/x]$ only.

$$\frac{\mathbf{F}:\forall x\varphi}{\mathbf{F}:\varphi[a_1/x] \quad \cdots \quad \mathbf{F}:\varphi[a_n/x] \quad \mathbf{F}:\varphi[a/x]} \qquad \frac{\mathbf{T}:\exists x\varphi}{\mathbf{T}:\varphi[a_1/x] \quad \cdots \quad \mathbf{T}:\varphi[a_n/x] \quad \mathbf{T}:\varphi[a/x]}$$

where $a_1, \ldots, a_n$ are all terms (= variables) in the tableau appearing free before or after this line, and $a$ does not appear free earlier in the tableau.

1. **The test at the start of the next tutorial will resemble this question.**

   In the last tutorial we only briefly practiced the branching quantifier rules, so here we will do some more.

For each of these signed propositions, use the tableaux method to extract a finite satisfying model. You should not attempt to construct the whole tableaux, because you cannot, as some branches would be infinite. It suffices to find one open terminated branch. You should likewise avoid pursuing any branches that you think will close, or multiple open branches; this question is all about being thoughtful and pursuing one path through the tableau to get to a model.

Because the **T** rule for $\forall$ and **F** rule for $\exists$ can fire more than once, your tableaux, if worked out in detail, might become quite repetitive. You may skip some repetitive steps if you justify your rules by explaining which line you got started with, which line this part of your tableau resembles, and which substitutions for bound variables you used to get there, e.g. 'from (3), as for (6), with $[b/x]$ and $[c/y]$'. You will be able to do this in the tutorial test next week also. Do not skip any steps the first time you apply one of these rules.

- $\mathbf{T} : \forall x(\exists y\, Rxy \wedge \exists y \neg Rxy)$ (where $R$ is a binary predicate)
  **Solution.**

$$(1)\ \mathbf{T} : \forall x(\exists y\, Rxy \wedge \exists y \neg Rxy)\ \checkmark\checkmark$$
$$(2)\ \mathbf{T} : \exists y\, Ray \wedge \exists y \neg Ray \text{ from } (1)\ \checkmark$$
$$(3)\ \mathbf{T} : \exists y\, Ray \text{ from } (2)\ \checkmark$$
$$(4)\ \mathbf{T} : \exists y \neg Ray \text{ from } (2)\ \checkmark$$
$$(5)\ \mathbf{T} : Raa \text{ from } (3)$$
$$(6)\ \mathbf{T} : \neg Rab \text{ from } (4)\ \checkmark$$
$$(7)\ \mathbf{F} : Rab \text{ from } (6)$$
$$(8)\ \mathbf{T} : \exists y\, Rby \wedge \exists y \neg Rby \text{ from } (1)\ \checkmark$$
$$(9)\ \mathbf{T} : \exists y\, Rby \text{ from } (8)\ \checkmark$$
$$(10)\ \mathbf{T} : \exists y \neg Rby \text{ from } (8)\ \checkmark$$
$$(11)\ \mathbf{T} : Rba \text{ from } (9)$$
$$(12)\ \mathbf{T} : \neg Rbb \text{ from } (10)\ \checkmark$$
$$(13)\ \mathbf{F} : Rbb \text{ from } (12)$$

  The model extracted is $\{a, b\}$ with $R$ interpreted as $\{(a, a), (b, a)\}$.
  Some notes on this solution:
  - There are plenty of other interpretations of $R$ that would work just as well; all we need is that $a$ and $b$ are each related to exactly one thing. There are also many models with more than two elements, but finding a two element model is the most efficient solution.
  - Because we only need to find one terminated open branch, we simply ignore branches that we don't think are worth exploring. We in fact did this four times in this proof, because the **T** rules for $\exists$, applied to lines (3), (4), (9), and (10), are all branching rules.
  - This proof is a bit repetitive because of the second rule application to line (1), so we could abbreviate it from line (8) on as:

$$(8)\ \mathbf{T} : Rba \text{ from } (1)\text{, as for } (5)\text{, with } [b/x] \text{ and } [a/y]$$
$$(9)\ \mathbf{F} : Rbb \text{ from } (1)\text{, as for } (7)\text{, with } [b/x] \text{ and } [b/y]$$

- $\mathbf{T} : \forall x(\exists y(Rxx \to \neg Rxy) \wedge \forall y\, Ryy)$
  **Solution.**

$$(1)\ \mathbf{T} : \forall x(\exists y(Rxx \to \neg Rxy) \wedge \forall y\, Ryy)\ \checkmark\checkmark$$
$$(2)\ \mathbf{T} : \exists y(Raa \to \neg Ray) \wedge \forall y\, Ryy \text{ from } (1)\ \checkmark$$
$$(3)\ \mathbf{T} : \exists y(Raa \to \neg Ray) \text{ from } (2)\ \checkmark$$
$$(4)\ \mathbf{T} : \forall y\, Ryy \text{ from } (2)\ \checkmark$$
$$(5)\ \mathbf{T} : Raa \to \neg Rab \text{ from } (3)\ \checkmark$$
$$(6)\ \mathbf{T} : Raa \text{ from } (4)$$

(7) $\mathbf{T} : Rbb$ from (4)

(8) $\mathbf{T} : \neg Rab$ from (5) ✓

(9) $\mathbf{F} : Rab$ from (8)

(10) $\mathbf{F} : Rba$ from (1), as for (9), with $[b/x]$ and $[a/y]$

The model is $\{a, b\}$ with $R$ interpreted as $\{(a, a), (b, b)\}$. A common mistake here would be to think we are done after line (9). Why do we need to continue? A plausible but wrong answer is that we do not yet know whether $(b, a)$ is in the interpretation of $R$. In fact, tableaux do not always give a **T** or **F** sign to every possible part of the relation; if there is something missing, that means that we get a satisfiying model either way. The correct answer is that the branch has not terminated because we have not yet reapplied line (1) for the new variable $b$.

If you do not understand the justification of line (10) you should reconstruct it yourself: first apply the **T** rule for the $\forall$ on line (1), but this time with the new variable $b$; then the **T** rule for $\wedge$; then the **T** rule for $\exists$, choosing the branch with $a$; then the **T** rule for $\neg$.

- $\mathbf{F} : \exists x \forall y(\neg \forall z\, Rzz \vee (Rxy \to Ryx))$

  **Solution.**

  (1) $\mathbf{F} : \exists x \forall y(\neg \forall z\, Rzz \vee (Rxy \to Ryx))$ ✓✓✓

  (2) $\mathbf{F} : \forall y(\neg \forall z\, Rzz \vee (Ray \to Rya))$ from (1) ✓

  (3) $\mathbf{F} : \neg \forall z\, Rzz \vee (Rab \to Rba)$ from (2) ✓

  (4) $\mathbf{F} : \neg \forall z\, Rzz$ from (3) ✓

  (5) $\mathbf{F} : Rab \to Rba$ from (3) ✓

  (6) $\mathbf{T} : \forall z\, Rzz$ from (4) ✓✓

  (7) $\mathbf{T} : Raa$ from (6)

  (8) $\mathbf{T} : Rbb$ from (6)

  (9) $\mathbf{T} : Rab$ from (5)

  (10) $\mathbf{F} : Rba$ from (5)

  (11) $\mathbf{T} : Rbc$ from (1), as for (9), with $[b/x]$ and $[c/y]$

  (12) $\mathbf{F} : Rcb$ from (1), as for (10), with $[b/x]$ and $[c/y]$

  (13) $\mathbf{T} : Rcc$ from (6), as for (7), with $[c/z]$

  (14) $\mathbf{T} : Rca$ from (1), as for (9), with $[c/x]$ and $[a/y]$

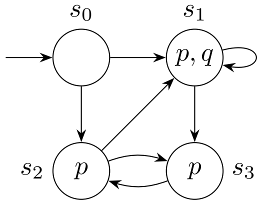
  (15) $\mathbf{F} : Rac$ from (1), as for (10), with $[c/x]$ and $[a/y]$

  The model is $\{a, b, c\}$ with $R$ interpreted as $\{(a, a), (b, b), (c, c), (a, b), (b, c), (c, a)\}$.

2. Turn the following descriptions of transition systems into diagrams. The states $s_0$ are start states.

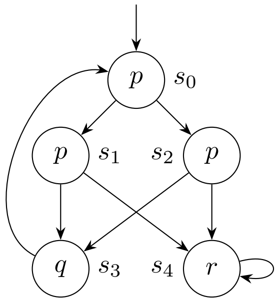
- $S = \{s_0, s_1, s_2, s_3\}$; $\to = \{(s_0, s_1), (s_0, s_2), (s_1, s_1), (s_1, s_3), (s_2, s_1), (s_2, s_3), (s_3, s_2)\}$; $L(s_0) = \emptyset$; $L(s_1) = \{p, q\}$; $L(s_2) = L(s_3) = \{p\}$.

  **Solution.**




- $S = \{s_0, s_1, s_2, s_3, s_4\}$; $\to = \{(s_0, s_1), (s_0, s_2), (s_1, s_3), (s_1, s_4), (s_2, s_3), (s_2, s_4), (s_3, s_0), (s_4, s_4)\}$; $L(s_0) = L(s_1) = L(s_2) = \{p\}$; $L(s_3) = \{q\}$; $L(s_4) = \{r\}$.

  **Solution.**




3. For each of the transition systems of the previous question, suggest some paths, starting at the state states. Then for each path you suggest, suggest some LTL propositions that would be satisfied by that path. Try to use all the new connectives of LTL.

   **Solution.**

   There are of course many possible answers here; what matters is that each path is an infinite list of states (not propositional variables) that follows the arrows of the diagrams.

   For example, for the first transition system we could have $s_0$ followed by $s_2s_1s_3$ repeated forever; or we could have $s_0$ followed by $s_1$ forever. For the second system we could have an infinite loop of $s_0s_1s_3$, or we could alternate between $s_0s_1s_3$ and $s_0s_2s_3$ ten times each, then go to $s_0s_1$, then get stuck in the sink state $s_4$ forever.

   There are also many propositions we could choose. Some will be satisfied no matter which path we consider: say $XGp$ for the first, or $G(p\,U(q \vee r))$ for the second. Some will depend on the path chosen. For example the path in the first system that loops around $s_2s_1s_3$ forever will satisfy $G(q \vee Xq \vee XXq)$, while the path in the second system that gets stuck in the sink state will satisfy $FGr$.

4. Recall the semantics of LTL:

   - $\vDash_{\mathcal{M},\sigma} p$ if $p \in L(\sigma_0)$
   - $\vDash_{\mathcal{M},\sigma} \bot$ never
   - $\vDash_{\mathcal{M},\sigma} \neg\varphi$ if it is not the case that $\vDash_{\mathcal{M},\sigma} \varphi$
   - $\vDash_{\mathcal{M},\sigma} \varphi \wedge \psi$ if $\vDash_{\mathcal{M},\sigma} \varphi$ and $\vDash_{\mathcal{M},\sigma} \psi$
   - $\vDash_{\mathcal{M},\sigma} \varphi \vee \psi$ if $\vDash_{\mathcal{M},\sigma} \varphi$ or $\vDash_{\mathcal{M},\sigma} \psi$ (or both)
   - $\vDash_{\mathcal{M},\sigma} \varphi \to \psi$ if $\vDash_{\mathcal{M},\sigma} \varphi$ implies $\vDash_{\mathcal{M},\sigma} \psi$
   - $\vDash_{\mathcal{M},\sigma} X\varphi$ if $\vDash_{\mathcal{M},\sigma_{\geq 1}} \varphi$
   - $\vDash_{\mathcal{M},\sigma} G\varphi$ if for all natural numbers $i$, $\vDash_{\mathcal{M},\sigma_{\geq i}} \varphi$
   - $\vDash_{\mathcal{M},\sigma} F\varphi$ if there exists a natural number $i$ such that $\vDash_{\mathcal{M},\sigma_{\geq i}} \varphi$
   - $\vDash_{\mathcal{M},\sigma} \varphi\,U\,\psi$ if there exists a natural number $i$ such that $\vDash_{\mathcal{M},\sigma_{\geq i}} \psi$, and for all $h < i$ we have $\vDash_{\mathcal{M},\sigma_{\geq h}} \varphi$

   Argue using the semantics that the following propositions are equivalent (regardless of the system $\mathcal{M}$ or path $\sigma$).

   - $X(\varphi \vee \psi)$ and $X\varphi \vee X\psi$.

     **Solution.**

     $\vDash_{\mathcal{M},\sigma} X(\varphi \vee \psi)$
     iff $\vDash_{\mathcal{M},\sigma_{\geq 1}} (\varphi \vee \psi)$
     iff $\vDash_{\mathcal{M},\sigma_{\geq 1}} \varphi$ or $\vDash_{\mathcal{M},\sigma_{\geq 1}} \psi$
     iff $\vDash_{\mathcal{M},\sigma} X\varphi$ or $\vDash_{\mathcal{M},\sigma} X\psi$
     iff $\vDash_{\mathcal{M},\sigma} X\varphi \vee X\psi$.

   - $XG\varphi$ and $GX\varphi$.

     **Solution.**

     $\vDash_{\mathcal{M},\sigma} XG\varphi$
     iff $\vDash_{\mathcal{M},\sigma_{\geq 1}} G\varphi$

iff for all natural numbers $i$, $\vDash_{\mathcal{M},(\sigma_{\geq 1})_{\geq i}} \varphi$

But $(\sigma_{\geq 1})_{\geq i}$ is just a path in $\sigma$ starting at position $i$, where $i$ must be greater than or equal to 1, so this is equivalent to
for all positive integers $i \geq 1$, $\vDash_{\mathcal{M},\sigma_{\geq i}} \varphi$
iff for all natural numbers $i$, $\vDash_{\mathcal{M},\sigma_{\geq i+1}} \varphi$
iff for all natural numbers $i$, $\vDash_{\mathcal{M},\sigma_{\geq i}} X\varphi$
iff $\vDash_{\mathcal{M},\sigma} GX\varphi$

- $\neg\varphi\, U\, \varphi$ and $F\varphi$.
  **Solution.**
  In all cases $\vDash_{\mathcal{M},\sigma} \varphi\, U\, \psi$ implies $\vDash_{\mathcal{M},\sigma} F\psi$, because part of the semantics of $U$ is that its right proposition must come true eventually, which is exactly the semantics of $F$.

  So we will focus on the other direction: say $\vDash_{\mathcal{M},\sigma} F\varphi$, so there exists a natural number $i$ such that $\vDash_{\mathcal{M},\sigma_{\geq i}} \varphi$. In particular let $i$ be the smallest such number. By this choice, it is not the case that $\vDash_{\mathcal{M},\sigma_{\geq h}} \varphi$ for any $h < i$. But this is exactly the requirement for $\vDash_{\mathcal{M},\sigma_{\geq h}} \neg\varphi$ for all such $h$. These two facts together define $\vDash_{\mathcal{M},\sigma} \neg\varphi\, U\, \varphi$.
- $\bot\, U\, \varphi$ and $\varphi$
  **Solution.**
  $\vDash_{\mathcal{M},\sigma} \bot\, U\, \varphi$ iff there exists a natural number $i$ such that $\vDash_{\mathcal{M},\sigma_{\geq i}} \varphi$, and for all $h < i$ we have $\vDash_{\mathcal{M},\sigma_{\geq h}} \bot$. But $\bot$ can never be satisfied, so there cannot exist any natural numbers $h$ less than $i$, so the Until statement holds exactly if $i$ is 0, i.e. $\vDash_{\mathcal{M},\sigma_{\geq 0}} \varphi$. But $\sigma_{\geq 0}$ is $\sigma$, so this is the same as saying $\vDash_{\mathcal{M},\sigma} \varphi$.

5. Give LTL propositions to specify the desirable properties of an elevator (lift) in a two story building. In each state the elevator should be on the ground or first floor, and buttons might have been pressed, or not, by people wanting the elevator. Because there are only two floors, only one button is required on each floor and no buttons are required inside the elevator. To simplify the system you may assume that different buttons are never pressed at the exact same moment, although they might be pressed close enough together that the elevator has not been able to fulfil the first request.

   Then draw a transition system whose paths satisfy your propositions.

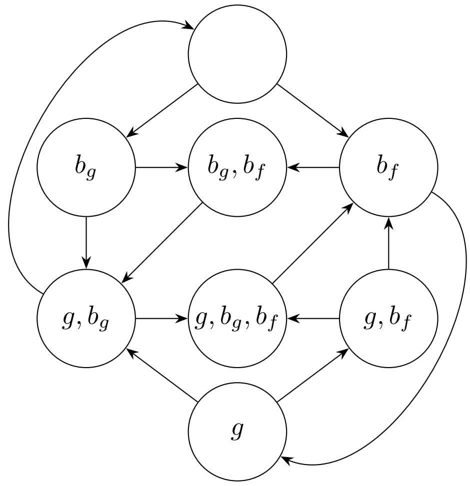
   **Solution.** Your propositions and transition system might not exactly match mine.

   We only need three propositions: $g$ for 'the elevator is on the ground floor' (if negated, it is on the first floor), $b_g$ for 'the ground floor button is pushed', and $b_f$ for 'the first floor button is pushed'.

   We will put all our propositions inside $G$, because they should hold forever, not just at the start of a path through our system:

   - If the elevator is on the floor on which a button is pressed, it should immediately move floors and clear that button: $G(g \wedge b_g \to X(\neg g \wedge \neg b_g))$ and $G(\neg g \wedge b_f \to X(g \wedge \neg b_f))$;
   - If the elevator is on a different floor than on which a button is pressed, it should immediately move floors, but not clear the button (which will happen at the next step): $G(g \wedge b_f \to X(\neg g \wedge b_f))$ and $G(\neg g \wedge b_g \to X(g \wedge b_g))$.

We would like it to be possible for the buttons to be pressed, and for both to be pressed simultaneously, but this cannot be directly expressed in LTL, because it requires an existential quantification over possible paths through the system. For this reason a one state system, unlabelled by any proposition, that loops to itself forever technically satisfies our specification! But here is a more reasonable system:




6. (Time consuming and open ended; no complete solution wil be presented). Suggest some LTL propositions that would help to specify a three floor elevator. In particular, how do you ensure that the elevator does not keep itself busy moving up and down between two of the floors while ignoring request for the third?

   You will find a transition system for such an elevator too large to construct in reasonable time, but you are welcome to think about how it might be done.

   **Solution.** Some remarks:

   - We need rather a lot of propositions. We will want one for each elevator position (say, $g$ for ground, $f$ for first, and $s$ for second); propositions for the direction of travel of the elevator (say $u$ for upwards and $d$ for downwards) because that effects e.g. whether an elevator moving from the first to third floor picks up a passenger from the second; propositions for each button on floors ($b_g$ and $b_s$ for the ground and second floor, but two buttons $b_{fu}$ and $b_{fd}$ for the first floor); and three propositions for each button inside the elevator (using '$i$' to mean inside, say $i_g$, $i_d$, and $i_s$). The combination of all these propositions makes for a rather large number of possible states!
   - We would need some safety propositions ruling out impossible actions, e.g. $G(g \rightarrow \neg f \wedge \neg s)$ - if the elevator is on the ground floor, it is not on the other floors - and $G(u \rightarrow \neg d)$ - if it is moving up, it is not moving down.
   - Liveness (on request) can be specified by propositions like $G(b_g \rightarrow Fg)$ - if the ground floor button is pressed, the elevator will eventually reach the ground floor. But this raises the possibility that the system might take a thousand steps to fulfil the request! If you think you can guarantee service in, say, six steps, you could specify this with $G(b_g \rightarrow g \vee Xg \vee XXg \vee \ldots \vee XXXXXXg)$.
   - The $U$ connective is quite useful for expressing some properties. For example if the button on the ground floor is pressed, it should stay pressed until the elevator reaches the ground floor: $G(b_g \rightarrow b_g \, U(g \wedge \neg b_g)$.
   - We need to think about unusual user behaviour. What should the system do if a user gets into the elevator then presses the button for the floor they are already on? What if the down button is pressed on the first floor, but if the user on that floor gets inside the elevator and requests the second floor?