Concurrency: 2^{nd} partial examination

You may consult the slides of the lectures. No other document or electronic device is allowed. Answers should be formulated in French or English, and preferably in a rigorous and sharp style.

Please write the solutions to the two parts in separate sheets.

First part

Exercise 1 (traces, 4.5 points) *Recall that in the context of CCS, the* traces *of a process P are defined as*

$$tr(P) = \{s \in \mathcal{L}^* \mid P \stackrel{s}{\Rightarrow} \cdot\}$$

and we write $P =_{tr} Q$ if $tr(P) = tr(Q)$. Show that if P, Q, R are CCS processes and $P =_{tr} Q$ then $(P \mid R) =_{tr} (Q \mid R)$.

Exercise 2 (confluence, 3 points) *Suppose P is a CCS process that is fully terminating and such that for every derivative Q of P we have:*

$$\frac{Q \stackrel{\tau}{\rightarrow} Q_1 \qquad Q \stackrel{\tau}{\rightarrow} Q_2}{Q_1 \approx Q_2}$$

Show that this implies that for every derivative Q of P we have:

$$\frac{Q \stackrel{\tau}{\Rightarrow} Q_1 \quad \textit{and} \quad Q \stackrel{\tau}{\Rightarrow} Q_2}{\exists\, Q_1', Q_2' \;\; (\; Q_1 \stackrel{\tau}{\Rightarrow} Q_1', \quad Q_2 \stackrel{\tau}{\Rightarrow} Q_2', \quad \textit{and} \quad Q_1' \approx Q_2' \;)}$$

Exercise 3 (definability, 2.5 points) *In the context of SCCS/Meije, we specify a ternary operator rr (round robin) by the rule:*

$$\frac{P_0 \stackrel{\alpha}{\rightarrow} P_0'}{rr(P_0, P_1, P_2) \stackrel{\alpha}{\rightarrow} rr(P_1, P_2, P_0')}$$

Show that the operator rr is definable as a SCCS/Meije process. This amounts to define a SCCS/Meije process $RR(P_0, P_1, P_2)$, parametric in P_0, P_1, P_2, which is strongly bisimilar to $rr(P_0, P_1, P_2)$.

Second part

Exercise 4 (testing semantics, 2 points) *Given a process P and a test T, i.e. a process containing the special action ω, remember that we say that*

P must T *if for every sequence of* τ*-transitions from* $P \,|\, T$*, we eventually reach a state in which the action* ω *is enabled. Furthermore, we say that* P*,* Q *are* must-equivalent *if for every test* T*,*

$$P \text{ must } T \quad \textit{if and only if} \quad Q \text{ must } T$$

- *Give an example of two processes that are weak bisimilar, but not must equivalent. (Hint: remember that weak bisimilarity is not sensitive to divergency, while must equivalence is.)*

- *Give an example of two processes that are must equivalent, but not weak bisimilar. (Hint: set one of the processes to be* $a.b.c.0 + a.b.d.0$ *.)*

Exercise 5 (output prefix encoding, 3 points) *Consider Boudol's encoding of the output-prefix construct into the asynchronous* π*-calculus, which is:*

$$\begin{array}{rcl} [\![x(y).P]\!] & = & x(z).(\nu w)(\overline{z}w | w(y).[\![P]\!]) \\ [\![\overline{x}y.Q]\!] & = & (\nu z)(\overline{x}z | z(w).\overline{w}y | [\![Q]\!]) \end{array}$$

and $[\![\cdot]\!]$ *homomorphic on all the other operators.*

Say whether the encoding is fully abstract with respect to must equivalence, or not. Justify formally yours answer.

Exercise 6 (dining philosophers, 5 points) *Is it possible to code in the* π*-calculus with full (or mixed) choice a solution to the dining philosophers problem, without introducing randomization? By "solution" we mean that if there are hungry philosophers, then some of them (not necessarily all) will eventually eat, and we want the philophers to be be symmetric, i.e. to run the same code modulo renaming, and to start in the same state. Furthermore, we want the solution to be distributed, i.e. no central coordinator or memory. You can assume that each philosophers can communicate directly with his neighbors via two channels.*

Justify formally your answer. Namely, if your answer is "yes", then give the code. If your answer is "no", then prove that for any possible code the scheduler can induce a computation in which nobody eats.