Prof. Dr. Ir. Joost-Pieter Katoen
apl. Prof. Dr. Thomas Noll
B. Kaminski, C. Matheja

Concurrency Theory WS 2015/2016

— 1st Exam —

First Name: ______________________________

Second Name: ______________________________

Matriculation Number: ______________________________

Degree Programme (please mark):

- ◦ CS Bachelor
- ◦ CS Master
- ◦ CS Lehramt
- ◦ SSE Master
- ◦ Other: ______________________________

General Information:

- Mark every sheet with your **matriculation number**.
- Check that your copy of the exam consists of **10 sheets (20 pages)**.
- Duration of exam: **120 minutes**.
- No helping materials (e.g. books, notes, slides) are permitted.
- Give your solution on the respective sheet. Also use the backside if necessary. If you need more paper, ask the assistants.
- Write with blue or black ink; do **not** use a pencil or red ink.
- Make sure all electronic devices are switched off and are nowhere near you.
- Any attempt at deception leads to failure for this exam, even if detected only later.

	Σ Points	Points obtained
Task 1	**28**	
Task 2	**28**	
Task 3	**15**	
Task 4	**14**	
Task 5	**24**	
Task 6	**11**	
Σ	**120**	

Task 1 (Labeled Transition Systems) (20+5+3 Points)

(a) Consider the following CCS process definition:

$$\begin{aligned} A &= \big(B + (D \parallel E)\big) \setminus \{\text{sync}\} \\ B &= (\text{a}.D \parallel \overline{\text{a}}.E) + \text{b}.nil \\ D &= \text{sync}.D \\ E &= \overline{\text{sync}}.E \end{aligned}$$

Derive all legal outgoing transitions $A \stackrel{\alpha}{\longrightarrow} A'$ (for some $A' \in \mathsf{Prc}$) by giving a corresponding derivation tree.

(b) Reconsider the CCS process definition from Task 2 (a):

$$\begin{aligned} A &= \big(B + (D \parallel E)\big) \setminus \{\text{sync}\} \\ B &= (\text{a}.D \parallel \overline{\text{a}}.E) + \text{b}.\mathit{nil} \\ D &= \text{sync}.D \\ E &= \overline{\text{sync}}.E \end{aligned}$$

Draw LTS(A) and label the nodes with the corresponding CCS processes.

(c) Give the trace language $\mathrm{Tr}(A)$ of A.

Task 2 (HML and Bisimulation) (18+8+2 Points)

Consider the following CCS processes:

$A = a.B + a.C$	$B = b.A + a.C + b.D$	$C = b.A + a.B + b.E$
$D = c.E + b.C$	$E = b.B + c.D$	$F = c.F + b.G$
$G = b.F + a.G + b.H$	$H = a.G$	$I = a.b.H + a.G$

(a) Draw $LTS(A)$, $LTS(H)$ and $LTS(I)$, respectively. Prove or disprove: $A \sim H$, $A \sim I$ and $H \sim I$, where $\sim$ denotes strong bisimilarity.

For proving or disproving that two processes are strongly bisimilar, you *may* use the game characterization of bisimilarity. For disproving you may alternatively provide an HML formula which is satisfied by only one of two processes.

(b) Provide a (possibly recursive) HML specification expressing that pattern aba is enabled in each state until action c is enabled (although c might never be enabled at all).

(c) Check whether H satisfies your HML specification provided in (b).

Task 3 (Modeling with CCS) (4+1+6+4 Points)

Let $A = \{a, b, c, d\}$. For a word $w \in A^*$ and $x \in A$, we write $\#_x(w)$ to denote the number of occurrences of symbol x in w. Moreover, $\mathit{pref}(L)$ denotes the prefix-closure of $L \subseteq \mathit{Act}^*$.

(a) Provide a CCS process P whose trace language is $Tr(P) = \mathit{pref}(L_1)$, where

$$L_1 = \{w \in A^* \mid \#_a(w) > \#_b(w)\}.$$

(b) Prove or disprove: There exists a CCS process P with $Tr(P) = \mathit{pref}(L_1)$ such that $LTS(P)$ is finite.

(c) Provide a CCS process Q whose trace language is

$$Tr(Q) = L_2 = \{w \in (\tau^* a \tau^*)^m (\tau^* b \tau^*)^n \mid m \geq n \geq 0\}.$$

(d) Prove or disprove: There exists a CCS process Q with $Tr(Q) = L_2$ such that $LTS(Q)$ is finite.

Task 4 (Preservation of Bisimilarity) (14 Points)

For $\beta \in \mathsf{Act}$, let $\odot\,\beta$ be a unary CCS operator with the following semantics:

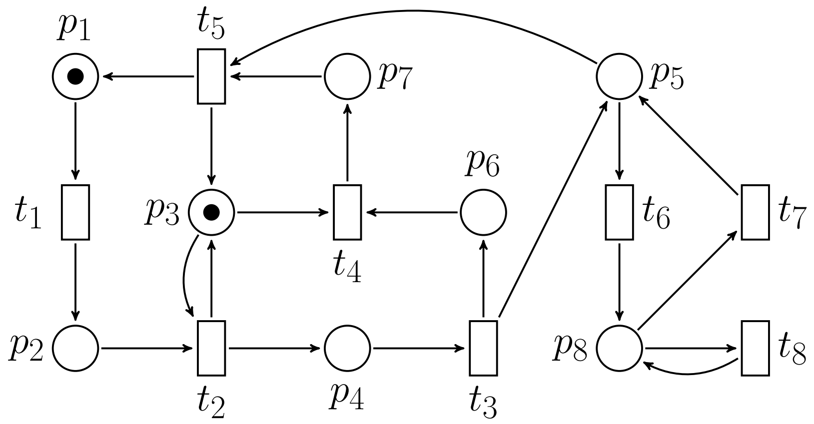
$$\text{(suff1)}\ \frac{P \xrightarrow{\alpha} P'}{P \odot \beta \xrightarrow{\alpha} P' \odot \beta}$$

$$\text{(suff2)}\ \frac{P \not\rightarrow}{P \odot \beta \xrightarrow{\beta} \mathit{nil}}$$

Prove or disprove: $\odot\,\beta$ preserves strong bisimilarity, i.e. for any processes S and T with $S \sim T$ it holds that $S \odot \beta \sim T \odot \beta$.

Task 5 (True Concurrency Semantics) (14+10 Points)

Consider the following elementary net N:



(a) Give the marking graph of N.

(b) Provide three non-isomorphic branching processes B_1, B_2, B_3 of N such that $B_1 \sqsubseteq B_2$ and $B_2 \not\sqsubseteq B_3 \not\sqsubseteq B_1$.

Task 6 (Petri–Net–Acceptable Languages) (3+8 Points)

Recall the following definition of Petri–net–acceptable languages: Let $N = (P, T, F, M_0)$ be an elementary net and let $\mathsf{Lab}\colon T \to \Sigma$, where Σ is a finite alphabet, be a labelling of the transitions. The language of N is defined as

$$\mathcal{L}(N, \mathsf{Lab}) \;=\; \{w \in \Sigma^* \mid w = \mathsf{Lab}(t_1)\cdots\mathsf{Lab}(t_k),\, \sigma = t_1 \cdots t_k,\, M_0 \xrightarrow{\sigma} M\}$$

A language L is called Petri–net–acceptable iff there exist an elementary net N with labelling Lab such that $L = \mathcal{L}(N, \mathsf{Lab})$. Prove or disprove:

a) If L is context–free, then L is Petri–net–acceptable.

b) If L is Petri–net–acceptable, then L is context–free.

Hint: The prefix–closure of the language $\{a^n bc^m bd^k \mid n \geq m \geq k \in \mathbb{N}\} \subseteq \{a, b, c, d\}^*$ *is not context–free.*