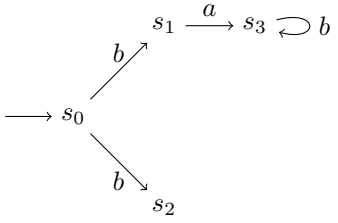
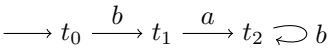
Examen. Documents autorisés. Durée 1h30

1 (Bi)simulation forte

Exercice 1 Soient les systèmes $\mathcal{S}$ et $\mathcal{T}$ suivants :

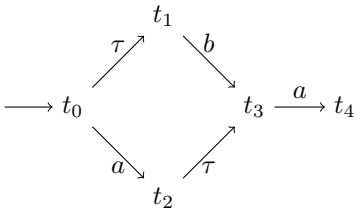


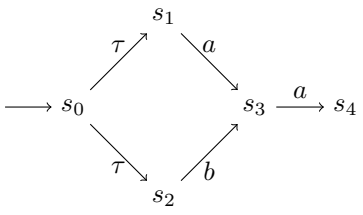


1. $\mathcal{S}$ est il simulé par $\mathcal{T}$?
2. Le prouver.
3. $\mathcal{T}$ est il simulé par $\mathcal{S}$?
4. Le prouver.
5. $\mathcal{S}$ et $\mathcal{T}$ sont ils bisimilaires ?
6. Le prouver.

2 (Bi)simulation faible

Exercice 2 Soient les systèmes $\mathcal{S}$ et $\mathcal{T}$ suivants :





1. $\mathcal{S}$ est il simulé par $\mathcal{T}$?
2. Le prouver.
3. $\mathcal{T}$ est il simulé par $\mathcal{S}$?
4. Le prouver.
5. $\mathcal{S}$ et $\mathcal{T}$ sont ils bisimilaires ?
6. Le prouver.

3 Calcul de processus CCS

Exercice 3 (Systèmes de transitions) Soient les définitions de processus CCS :

$$P \triangleq \mathsf{a.b} \| \mathsf{b.a} \qquad\qquad Q \triangleq \nu \mathsf{c}.(\mathsf{a.c.b} \| \mathsf{b.}\overline{\mathsf{c}}\mathsf{.a})$$

Dessinez les systèmes de transitions associés aux processus P et Q (chaque état sera représenté par un processus CCS).

3.1 Modélisation

On souhaite modéliser en CCS un allocateur mémoire non bloquant qui permet d'allouer deux types de blocs différents. On ne cherchera pas à représenter l'utilisation qui est faite de cette mémoire. On considère les actions (et co-actions) $\texttt{req}_i$, $\texttt{all}_i$, $\texttt{nil}_i$ et $\texttt{lib}_i$ correspondant respectivement à la requête d'un bloc de type i, à l'allocation de ce bloc, à la non-allocation si aucun bloc de ce type n'est disponible et enfin à sa récupération par le système, pour $i = 1, 2$. Pour gérer le nombre de blocs, on utilisera également les processus compteurs suivants $C_i^{k,n}$, avec leurs actions et co-actions paramétrées par $i = 1, 2$:

$$\begin{array}{ll} C_i^{0,n} & \triangleq \texttt{plus}_i.C_i^{1,n} + \texttt{test}_i.\overline{\texttt{zero}}_i.C_i^{0,n} \\ C_i^{k,n} & \triangleq \texttt{plus}_i.C_i^{k+1,n} + \texttt{moins}_i.C_i^{k-1,n} + \texttt{test}_i.\overline{\texttt{nonzero}}_i.C_i^{k,n},\ k \in [1, n-1] \\ C_i^{n,n} & \triangleq \texttt{moins}_i.C_i^{n-1,n} + \texttt{test}_i.\overline{\texttt{nonzero}}_i.C_i^{n,n} \end{array}$$

Exercice 4 (Utilisateurs)

1. Écrire en CCS un processus $Utilisateur_i$ typique, représentant le fonctionnement d'un unique utilisateur demandant un bloc de type i et le libérant ensuite, s'il en a obtenu un. Toutes les interactions possibles avec l'allocateur devront être envisagées.
2. Écrire un processus *Utilisateurs* qui permet de modéliser l'arrivée de nouveaux utilisateurs quelconques.

Exercice 5 (Allocateur, exercice bonus)

1. On s'intéresse d'abord à l'allocation de blocs de type 1 uniquement. Le protocole global permettant d'allouer n blocs sera représenté par le processus $Protocole \triangleq Alloc \| C_1^{n,n}$. Décrire le processus *Alloc* tel que le système réponde aux requêtes d'allocation et de libération faites par les utilisateurs.
2. Modifier *Protocole* pour que les utilisateurs ne puissent plus interférer avec les événements liés au comptage des blocs.
3. Modifier *Protocole* pour pouvoir maintenant allouer les deux types de blocs. Veiller à obtenir un parallélisme maximal.
4. On suppose que tout bloc de type 2 peut se transformer en 2 blocs de type 1 et réciproquement. Modifier *Alloc* pour exploiter au mieux les blocs disponibles selon les demandes d'allocation.