PC4 – Modélisation de systèmes répartis

Échange de messages et aspects temporels

INF447

La modélisation FSP permet de décrire le fonctionnement de processus, ainsi que l'entrelacement de leurs exécutions. Mais, parfois, les processus peuvent avoir des vitesses d'exécution très différentes, par exemple lorsqu'ils sont situés sur des machines différentes. Pour se synchroniser, les processus doivent également utiliser des canaux de communications qui sont plus ou moins performants. Nous allons nous intéresser maintenant aux problèmes que peuvent soulever ces contraintes.

La première des choses pour construire un modèle d'un système réparti est la donnée d'un ensemble de processus $p_1, \ldots, p_n$. Ces processus modélisent des exécutions sur n machines indépendantes. Chaque processus peut lui même être complexe et se décomposer en un ensemble de sous-processus concurrents (au sens vu dans la première partie du module).

Ensuite, dans le cadre d'un système réparti, la synchronisation d'actions de FSP qui impose leur exécution simultanée n'est plus réaliste car on ne peut assurer une synchronisation parfaite entre deux machines. La communication doit donc devenir *asynchrone*. Pour cela, nous introduisons deux primitives atomiques (c'est-à-dire indivisibles) :

- $send(m)$ *to* i pour décrire l'envoi d'un message m au processus p_i
- $rec(m)$ pour décrire la reception d'un message m d'un autre processus

Remarquez qu'ici le message n'est pas une donnée qui est transportée d'un processus à un autre mais un identifiant permettant de relier un *rec* effectué par un processus au *send* qui lui correspond dans le processus envoyeur.

Ces deux actions primitives s'ajoutent au modèle FSP existant. Nous parlerons d'*événements*[1] en faisant une distinction entre les événements *internes*, qui correspondent à une séquence d'opérations exécutées par le processus (*cf.* une action dans la modélisation FSP), et les événements *de communication*, qui sont relatifs aux sus-dites primitives (*send* et *rec*).

Afin de formaliser le concept de communication, nous introduisons maintenant quelques notations. Nous considérons n processus indépendants $p_1, \ldots, p_n$. Lors d'une exécution d'un processus, nous notons e_i^x le x-ième événement produit par le processus p_i. Une trace d'exécution du processus p_i

1. En FSP, on employait plutôt le terme d'action mais dans le cadre de cette séance, nous emploierons le vocabulaire standard des systèmes répartis : événement.

est donc :

$$e_i^1 \; e_i^2 \; \ldots \; e_i^x \; e_i^{x+1} \ldots$$

Exercice 1 (*Événements et relations causales*)

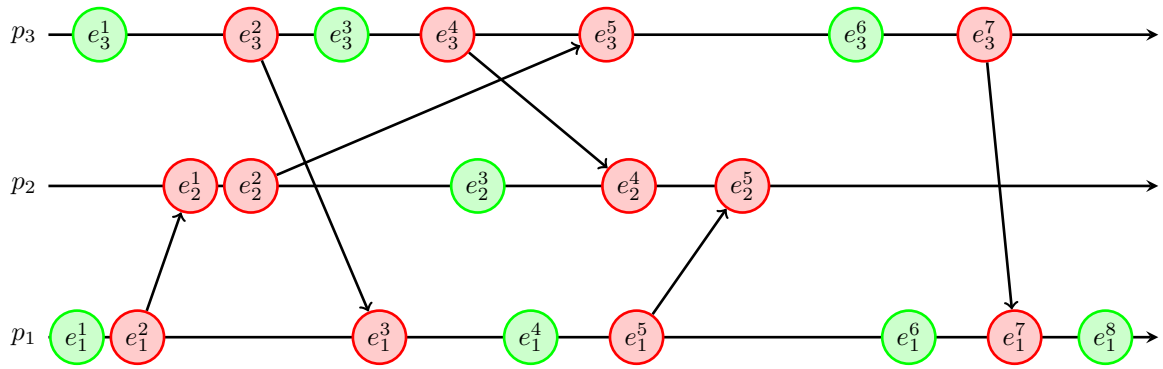
On définit alors la relation suivante dite de *causalité* : «*un événement* e_i^x *précède un événement* e_j^y». Elle s'écrit $e_i^x \xrightarrow{ev} e_j^y$.

Un événement e_i^x précède un événement e_j^y si et seulement si une parmi les trois conditions suivantes est vraie :

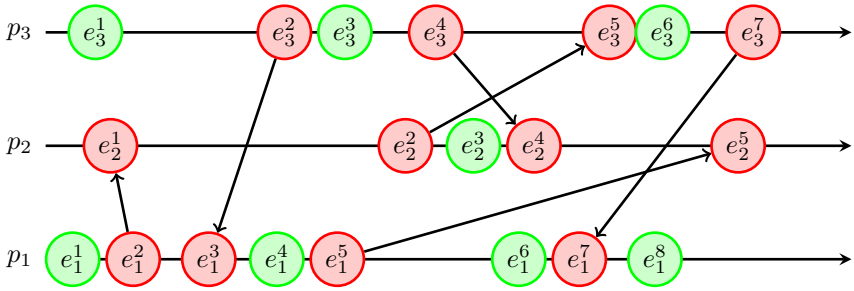
1. les événements se déroulent au sein d'un même processus
2. les deux événements concernent le même message m
3. une transition par un événement indépendant existe

▷ **Question 1.1 :**
Dans le but de vérifier que vous n'avez pas survolé, hagards, les quelques lignes précédentes, veuillez définir formellement ces trois conditions (attention, c'est trivial).

Voici un petit exemple d'une exécution d'un système distribué :



▷ **Question 1.2 :**
Quelle différence faites-vous entre ce diagramme et la réprésentation ci-dessous ?



Un *chemin causal* est défini comme étant une suite d'événements consécutifs au sens de la relation $\xrightarrow{ev}$. Par extension, le *passé causal* (respectivement *futur causal*) d'un événement e est l'ensemble des événements situés avant (respectivement après) e dans l'ensemble des chemins causaux.

▷ **Question 1.3 :**
Dans le diagramme précédent, quel est le passé causal de e_2^2 ? Quel est son futur causal ?

Soit F un ensemble de n événements donnés, un événement par processus. Une *coupe* est l'ensemble des événements antérieurs aux événements de F dans chaque processus. On appelle *frontière* cet ensemble F. Par exemple dans le diagramme précédent, une coupe C_1 est définie par sa frontière $\{e_1^3, e_2^1, e_3^4\}$ et la coupe C_2 est définie par $\{e_1^3, e_2^1, e_3^5\}$ [2].

▷ **Question 1.4 :**
Quelle différence fondamentale voyez-vous entre les coupes C_1 et C_2 ? Définissez formellement une coupe *cohérente*.

Avez-vous une formulation qui permettrait à un enfant de cinq ans de comprendre ce qu'est une coupe cohérente sur un diagramme de temps comme celui représenté ci-dessus ?

Exercice 2 (*Cohérence d'états et exécutions*)

Dans ce contexte, à quoi ressemblent nos bons vieux états de FSP ? Nous utilisons ici une notation simplifiée où σ_i^x représente l'état dans lequel se trouve le processus i immédiatement après l'événement e_i^x. Naturellement, le processus est dans cet état jusqu'à l'événement e_i^{x+1}.

On peut définir une relation causale entre les états par la transition $\xrightarrow{state}$. Ainsi, nous avons :

$$e_i^{x+1} \xrightarrow{ev} e_j^y \iff \sigma_i^x \xrightarrow{state} \sigma_j^y$$

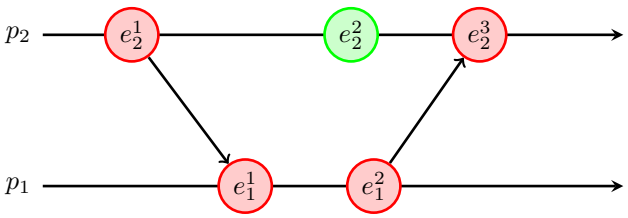
Un *état global* du système est un ensemble d'états locaux, par exemple $\Sigma = \{\sigma_1^{x_1}, \ldots, \sigma_n^{x_n}\}$.

▷ **Question 2.1 :**
Qu'est-ce qu'un état global cohérent ? Qu'est-ce que cela signifie concrètement dans un système distribué ?

On peut maintenant revenir au concept de *trace* qui a été vu dans le modèle FSP. On parle également d'*exécution*.

▷ **Question 2.2 :**
Qu'est-ce qu'une exécution cohérente ?

Partons d'un diagramme relativement simple.



▷ **Question 2.3 :**
Déterminez l'ensemble des exécutions cohérentes du système et tentez de le représenter.

▷ **Question 2.4 :**
Comment pourrait-on utiliser la composition usuelle de FSP (||) de processus pour modéliser un système réparti ?

2. Un événement à la frontière d'une coupe est considéré comme étant inclus dans la coupe.

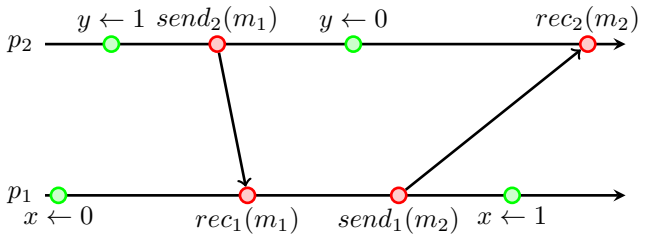
Exercice 3 (*Propriétés*)

On dit qu'un état global Σ_2 est *atteignable* à partir d'un état global Σ_1 s'il existe une exécution cohérente qui permet de passer de Σ_1 à Σ_2. Un état global peut vérifier une propriété donnée, ou pas (on parle également de prédicat).

▷ **Question 3.1 :**
Qu'est-ce qu'une propriété globale stable ? Donnez quelques exemples de propriétés globales stables.

Définissez la *vivacité* et la *sûreté* d'un système distribué.

Un ultime petit exercice. Voici un diagramme.



▷ **Question 3.2 :**
Dans ce système, quelle est la caractéristique de la propriété : $x = 0 \ \wedge \ y = 0$.