TD – Systèmes Distribués

Mémoire partagée distribuée : le modèle Linda

Master TI $1^{ère}$ année – UPPA – Eric Cariou

1 Les mémoires partagées distribuées

En parallélisme, une mémoire partagée est une mémoire qui est accessible et utilisée par plusieurs processus simultanément, une mémoire commune. Dans un cadre distribué, l'abstraction est identique à la différence que les processus s'exécutent sur des machines différentes et réparties sur le réseau.

Le principal problème est alors de créer une mémoire virtuelle utilisable par tous les processus distribués à partir des mémoires locales de chaque machine ou processus. Il faut également que l'accès aux données de la mémoire puisse se faire aussi facilement et équitablement à partir de n'importe quel processus distant.

2 Le modèle Linda

2.1 Description (simplifiée)

Linda est un modèle de mémoire partagée basé sur la notion d'environnement global partagé appelé l'espace des tuples. Un "tuple" (ou un n-uplet en français) est une séquence de champs, qui peuvent soit contenir une valeur, soit contenir une variable.

Le mécanisme d'interaction de base est le "pattern-matching" (appariement) avec un tuple de l'espace des tuples. Un tuple T1 est apparié avec un tuple T2 si T1 possède les mêmes valeurs que T2 et aux mêmes emplacements (ce qui implique que T1 et T2 ont forcément le même nombre de champs). Les autres champs de T1 seront des variables.

Il y a 3 opérations exécutables par les processus sur l'espace des tuples :

- `in(t)` : recherche d'un tuple `t'` dans l'espace des tuples pour lequel `t` est apparié evec `t'`. Si il est trouvé, il est enlevé de l'espace avec affectation aux variables de `t` des valeurs de `t'` se trouvant aux mêmes emplacements. Sinon le processus reste bloqué tant qu'il n'y a pas de tuple disponible que `t` apparie.
- `read(t)` : possède le même comportement que `in` sauf que ne supprime pas le tuple de l'espace.
- `out(t)` : ajoute `t` à l'espace. `t` est un tuple ne contenant que des valeurs et pas de variables.

Ces primitives sont non déterministes. Par exemple, lorsqu'un processus effectue un `in` et qu'il y a plusieurs tuples pouvant être retirés de l'espace, le choix entre ces tuples est fait de façon non déterministe.

L'adressage de Linda est dit associatif par le contenu : on ne demande pas à accéder à un emplacement mémoire à partir de son adresse ou d'un identificateur, on demande à récupérer des informations par rapport à un contenu que l'on précise.

2.2 Exemples

Dans la description précédente, la notion de variable correspond en fait à l'écriture d'une valeur dans une variable. On peut lire le contenu d'une variable pour préciser une valeur. L'écriture dans une variable `var` est notée `?var` et la lecture de sa valeur `var` tout court. Les types des variables et constantes seront ici des chaînes de caractères ou bien des nombres.

Quelques exemples d'instructions manipulant des tuples et les 3 primitives, avec nb = 10 et chaine = "bonjour" :

- `out(20, ''hello'')` : rajoute un tuple à 2 champs dans l'espace contenant les valeurs 20 et "hello".
- `out(20, chaine)` : rajoute un tuple à 2 champs dans l'espace contenant les valeurs 20 et "bonjour" (la valeur contenue dans la variable `chaine`).
- `out(nb, ''hello'')` : rajoute un tuple à 2 champs dans l'espace contenant les valeurs 10 (la valeur contenue dans la variable `nb`)et "hello".
- `in(?nb, ''hello'')` : retire de l'espace un tuple à 2 champs commençant par un entier quelconque et la chaine "hello". `nb` contient alors la valeur du premier champ de ce tuple.
- `read(20,?chaine)` : lit de l'espace un tuple à 2 champs commençant par l'entier 20 et avec ensuite une chaine quelconque. `chaine` contient alors la valeur du second champ de ce tuple. Le tuple n'est pas retiré de l'espace.

3 Questions

1. Détailler succintement comment les 4 opérations suivantes peuvent être réalisées via un modèle Linda :
   - Ajout dans la mémoire des informations sur une personne (nom + age) et création de son identificateur associé.
   - Récupération des informations sur une personne (nom + age) à partir de son identificateur.
   - Récupérer l'identificateur d'une personne à partir de son couple nom/age.
   - Suppression de la mémoire des informations sur la personne dont on fournit l'identificateur.
2. Quelles contraintes sont imposées pour certaines opérations ? Comment les limiter ?
3. On suppose que l'espace des tuples est physiquement partagé entre les mémoires locales de tous les processus et que les tuples ne sont pas dupliqués. Écrire un algorithme permettant de mettre en œuvre les opérations `in`, `read` et `out`.
4. Quels sont les limites et inconvénients de cet algorithme ? Proposer des améliorations.