Oraux blancs MPI
type CCMT

Florian Bourse

Exercice 1

cf annexe pour un rappel des règles de la déduction naturelle

1. Pour chacun des séquents suivants, prouver qu'ils sont déductibles ou prouver qu'ils ne le sont pas en exhibant un contre-modèle.
   (a) $\neg p \vdash \neg(p \wedge q)$
   (b) $\neg p \vdash \neg(p \vee q)$
   (c) $\neg p, p \to q \vdash \neg q$
   (d) $\neg q, p \to q \vdash q \to p$
2. On s'intéresse maintenant au typage du code OCaml suivant :

```
let f = fun x -> x 0 + 1;;

let g = fun x -> 1 + x;;

let h = fun x -> f g + g x;;
```

   Les règles de typage considérées sont les suivantes :

$$\frac{}{\Gamma \vdash \texttt{0} : \texttt{int}}(\texttt{0}_i) \qquad \frac{}{\Gamma \vdash \texttt{1} : \texttt{int}}(\texttt{1}_i) \qquad \frac{\Gamma \vdash \texttt{x} : \texttt{int} \quad \Gamma \vdash \texttt{y} : \texttt{int}}{\Gamma \vdash \texttt{x + y} : \texttt{int}}(\texttt{+}_i)$$

$$\frac{\Gamma, \texttt{x} : \sigma \vdash \texttt{e} : \tau}{\Gamma \vdash \texttt{fun x -> e} : \sigma\texttt{->}\tau}(\texttt{->}_i) \qquad \frac{\Gamma \vdash \texttt{f} : \sigma\texttt{->}\tau \quad \Gamma \vdash \texttt{x} : \sigma}{\Gamma \vdash \texttt{f x} : \tau}(\texttt{->}_e)$$

   Montrer que le séquent suivant est déductible :

$$\vdash \texttt{f} : \texttt{(int -> int) -> int}$$

3. Montrer que le séquent suivant est déductible :

$$\vdash \texttt{g} : \texttt{int -> int}$$

4. Quel est le type de la fonction `h` ? Démontrer le en prouvant la déductibilité d'un séquent dont les hypothèse sont $\Gamma = \{\texttt{f} : \texttt{(int -> int) -> int}, \texttt{g} : \texttt{int -> int}\}$

Exercice 2

Deux fils d'exécution A et B doivent communiquer au travers d'un seul tableau partagé `char T[256];` (pour tampon) que l'on suppose de taille suffisante, à l'aide de 2 fonctions

`void depot(char buf[]);`
`void recuperer(char buf[]);`

qui permettent respectivement d'écrire un message dans le tampon et de lire le message du tampon.
A et B doivent communiquer à tour de rôle, en commençant par $A \rightarrow B$ (A envoie un message à B).

Nous supposons l'existence d'un type `semaphore_t`, ainsi que 3 fonctions dont la description suit :

`semaphore_t *init(int n);` permet de créer un sémaphore dont la valeur initiale du compteur est n ;

`void P(semaphore_t *s);` décroît la valeur du compteur du sémaphore. Si cette valeur est nulle, alors cette instruction bloque jusqu'à ce qu'elle devienne non nulle ;

`void V(semaphore_t *s);` incrémente la valeur du compteur du sémaphore.

On suppose que 2 sémaphores `SA` et `SB` ont été initialisé avant le lancement des deux fils d'exécution. Voici le code du fil d'exécution A :

```
{ // Code du fil d'exécution A
  char message[256]; char reponse[256];
  while (true) {
  /* Génère le message à envoyer */
    P(SA);
    depot(message);
    V(SB);
    P(SA);
    recuperer(reponse);
    V(SB);
  }
}
```

1. Écrire le code du fil d'exécution B.
2. Quelles doivent être les valeurs initiales des compteurs de `SA` et `SB` ?
3. On suppose maintenant qu'un troisième fil C veuille communiquer avec B en utilisant le même tampon `T`. Les fils A et C se comportent de la même manière. B peut recevoir un message de A ou de C, la réponse doit être récupérée par le processus expéditeur du message.

   Expliquez les problèmes éventuels si on ne change pas les codes de A ni de B et que C utilise le même code que A.
4. On suppose maintenant l'existence d'un type `mutex_t` représentant un mutex, ainsi que 2 fonctions dont la description suit :

   `void lock(mutex_t *m);` permet de verrouiller un mutex ;
   `void unlock(mutex_t *m);` permet de déverrouiller un mutex.

   On suppose qu'un mutex `m` a été initialisé avant le lancement des fils d'exécution. Proposer une modification du code de A qui permet de résoudre le problème.

A Annexe : Rappel des règles de la déduction naturelle

Les arbres de preuves doivent être effectués à partir de l'ensemble de règles fourni ci-dessous.

$$\frac{}{\Gamma, p \vdash p}(Ax)$$

$$\frac{\Gamma \vdash p \quad \Gamma \vdash q}{\Gamma \vdash p \wedge q}(\wedge_i) \qquad \frac{\Gamma \vdash p \wedge q}{\Gamma \vdash p}(\wedge_e) \qquad \frac{\Gamma \vdash p \wedge q}{\Gamma \vdash q}(\wedge_e)$$

$$\frac{\Gamma \vdash p}{\Gamma \vdash p \vee q}(\vee_i) \qquad \frac{\Gamma \vdash q}{\Gamma \vdash p \vee q}(\vee_i) \qquad \frac{\Gamma \vdash p \vee q \quad \Gamma, p \vdash \varphi \quad \Gamma, q \vdash \varphi}{\Gamma \vdash \varphi}(\vee_e)$$

$$\frac{\Gamma, p \vdash q}{\Gamma \vdash p \to q}(\to_i) \qquad \frac{\Gamma \vdash p \to q \quad \Gamma \vdash p}{\Gamma \vdash q}(\to_e)$$

$$\frac{\Gamma, p \vdash \bot}{\Gamma \vdash \neg p}(\neg_i) \qquad \frac{\Gamma \vdash p \quad \Gamma \vdash \neg p}{\Gamma \vdash \bot}(\neg_e)$$

$$\frac{\Gamma, p \vdash q \quad \Gamma, \neg p \vdash q}{\Gamma \vdash q}(t.e.) \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash p}(\bot_e)$$