

1 Logique du premier ordre

1.1 exercise

On donne la formule logique φ suivante.

$$\forall x. \forall y. \exists z. (\neg(x < y) \vee ((x < z) \wedge (z < y)))$$

1. Quel est son arbre de syntaxe abstraite ?
2. Quelles sont ses formules atomiques ?
3. Comporte-t-elle des variables liées ? des variables libres ? lesquelles ?
4. Quels sont ses termes ?

1.2 exercise

Traduire en formules logiques du premier ordre les phrases suivantes. Introduire tous les prédicats nécessaires.

1. Dans une école, il existe des ordinateurs non connectés au réseau local.
2. Dans les écoles, tous les ordinateurs sont connectés à un réseau local.
3. Dans chaque école, au moins un ordinateur est connecté à la fois au réseau local et à internet.

1.3 exercise

On cherche à déterminer si un tableau a contient un doublon, c'est-à-dire un élément apparaissant deux fois. On se donne la fonction C suivante.

```
bool duplicate(int a[], int n) {
    for (int i = 0; i < n; i++) {
        for (int j = i + 1; j < n; j++) {
            if (a[i] == a[j]) { return true; }
        }
    }
    return false;
}
```

1. Voici quatre formules à propos d'un tableau a de taille n .
 - (a) $\forall i \in [0, n[. \forall j \in [0, n[. i \neq j \rightarrow t[i] = t[j]$
 - (b) $\forall i \in [0, n[. \exists j \in [0, n[. i \neq j \wedge t[i] = t[j]$
 - (c) $\exists i \in [0, n[. \forall j \in [0, n[. i \neq j \rightarrow t[i] = t[j]$
 - (d) $\exists i \in [0, n[. \exists j \in [0, n[. i \neq j \wedge t[i] = t[j]$

Pour chacune de ces formules, donner une traduction en langue naturelle et un exemple de tableau validant la formule. Quelle formule exprime effectivement le plus fidèlement la présence d'un doublon ?

- Donner les invariants pour les deux boucles de la fonction `duplicate`, écrits comme des formules de logique du premier ordre.

1.4 exercise

On considère le problème de la recherche d'un motif m dans un texte t , et on se donne la fonction C suivante. La fonction renvoie l'indice de début d'une occurrence de m dans t s'il en existe une, et -1 sinon.

```
int search(char *m, char *t) {
    int lm = strlen(m), lt = strlen(t);
    for (int i = 0; i <= lt - lm; i++) {
        int j = 0;
        while (j < lm) {
            if (m[j] != t[i+j]) break;
            j++;
        }
        if (j == lm) return i;
    }
    return -1;
}
```

- Donner des formules de logique du premier ordre exprimant la spécification du problème. Dans ces formules, on notera le résultat renvoyé par r .
- Donner des formules de logique du premier ordre exprimant les invariants des deux boucles.

1.5 exercise

On considère la fonction `longest_repetition` de l'exercice 59 page 315. Donner des invariants pour ses deux boucles, sous la forme de formules de logique du premier ordre.

2 Dédution naturelle

2.1 exercise

Construire des arbres de dérivation pour les jugements suivants (sans raisonnement par l'absurde).

- $(\varphi_1 \wedge \varphi_2) \rightarrow \psi \vdash \varphi_1 \rightarrow (\varphi_2 \rightarrow \psi)$
- $\varphi \rightarrow (\psi \rightarrow \theta), \psi \rightarrow \varphi \vdash \psi \rightarrow \theta$
- $\neg\varphi \vee \psi \vdash \varphi \rightarrow \psi$
- $\varphi \rightarrow \neg\varphi \vdash \neg\varphi$

2.2 exercise

Donner une dérivation de $\neg\neg\varphi, \varphi \vee \neg\varphi \vdash \varphi$, sans utiliser la règle de raisonnement par l'absurde.

2.3 exercise

Donner une dérivation de $\neg\neg\varphi \vdash \varphi$.

2.4 exercise

Démontrer que $\neg(\varphi_1 \wedge \varphi_2)$ et $\neg\varphi_1 \vee \neg\varphi_2$ sont prouvablement équivalents, c'est-à-dire que chacun on peut dériver l'autre. Indication : l'une des directions est plus facile que l'autre. On pourra admettre le tiers exclu, c'est-à-dire la provabilité du séquent $\vdash \varphi \vee \neg\varphi$ pour toute formule φ .

2.5 exercise

Donner des dérivations pour les séquents suivants.

1. $\forall x.\varphi \vdash \neg\exists x.\neg\varphi$
2. $\vdash \forall x.\forall y. x = y \rightarrow y = x$
3. $\vdash \forall x.\forall y. (\exists z. x = z \wedge y = z) \rightarrow x = y$