# IGL752 – Techniques de vérification et de validation

Université de Sherbrooke

# Devoir 4

| | |
|---|---|
| Enseignant: | Michael Blondin |
| Date de remise: | mercredi 27 mars 2019 à 15:30 |
| À réaliser: | individuellement ou en équipe de deux |
| Modalités: | remettre par courriel ou en classe, au début du cours, en copie imprimée ou manuscrite lisible |
| Bonus: | chaque question marquée par ★ vaut 0.5 point bonus sur les 60 points de la session attribués aux devoirs |

Dans ce devoir, l'abréviation ≪ BDD ≫ réfère à ≪ diagramme de décision binaire ordonné et réduit ≫.

**Question 1.** 4 P.

Montrez que l'ordre assignée aux variables d'une expression booléenne peut avoir un impact sur le nombre de sommets du BDD obtenu pour cette expression.

**Question 2.** 4 P.

Donnez un algorithme qui résoud le problème suivant:

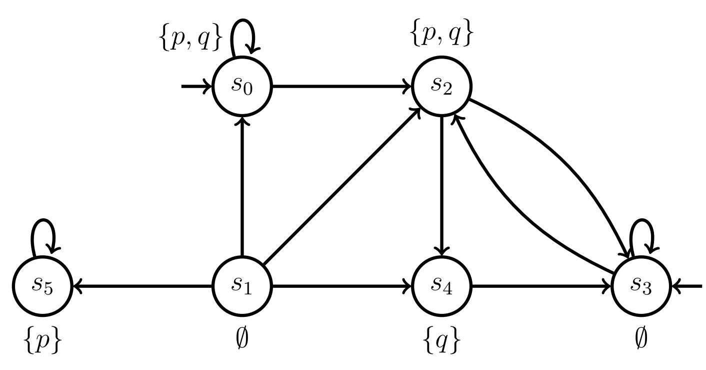
Entrée: un BDD $B$ sur les variables $x_1 < x_2 < \cdots < x_n$, et un sommet $u$ de $B$ représentant une expression booléenne $\varphi$

Sortie: un sommet $v$ qui représente $\neg\varphi$

Votre algorithme peut (et doit probablement) modifier $B$ lors du calcul de $v$.

**Question 3.** 12 P.

Considérons la structure de Kripke $\mathcal{T} = (S, \rightarrow, I, AP, L)$ suivante:




Supposons que chaque état de $\mathcal{T}$ est encodé par la représentation binaire de son indice; autrement dit: $s_0 = 000$, $s_1 = 001$, $s_2 = 010$, $s_3 = 011$, $s_4 = 100$ et $s_5 = 101$.

(a) Donnez une expression booléenne $\varphi$ qui représente l'ensemble $[\![p]\!]$;

(b) Donnez une expression booléenne $\psi$ qui représente l'ensemble $[\![q]\!]$;

(c) Construisez un BDD pour $\varphi$ à l'aide de la procédure Build;

(d) Construisez un BDD pour $\psi$ à l'aide de la procédure Build;

(e) Construisez un BDD pour $\varphi \lor \psi$ à l'aide de la procédure Apply;

(f) Expliquez comment procéder afin de déterminer si $\mathcal{T} \models \exists \mathsf{X}(p \lor q)$ à partir du BDD obtenu en (e). Vous n'avez pas à calculer le résultat des opérations mentionnées dans vos explications; il suffit de décrire ce qu'il faudrait calculer. Vous pouvez introduire de nouvelles variables au besoin.

Vous devez faire évoluer le même BDD en (c), (d) et (e). Donnez également l'arbre de récursion en (c), (d) et (e). Si vous utilisez des optimisations de Build ou Apply, dites lesquelles.

**Question 4.** 10 P.

Considérons le programme suivant constitué de deux fonctions et d'une variable booléenne globale $x$:

```
      bool x ∈ {faux,vrai}

      foo(bool y):                 bar():
f0:     si x ∨ y:            b0:     x = ¬x
          bar()              b1:     foo(x)
f1:       bar()
        sinon:
          x = ¬y
f2:     assert(x)
```

(a) Modélisez le programme avec un système à pile $\mathcal{P}$;

(b) Donnez un $\mathcal{P}$-automate $A$ tel que $Conf(A)$ est l'ensemble des configurations où l'assertion est enfreinte;

(c) Construisez partiellement un $\mathcal{P}$-automate qui accepte $\text{Pre}^*(Conf(A))$; plus précisément donnez 4 nouvelles transitions obtenues à partir de $A$ en exécutant l'algorithme de saturation vu en classe;

(d) Si vous aviez entièrement calculé le $\mathcal{P}$-automate qui accepte $\text{Pre}^*(Conf(A))$ en (c), comment auriez-vous pu déterminer si un appel à `foo` peut enfreindre l'assertion?

**★ Question 5.** (+0.5 P.)

Soit $u$ un sommet d'un BDD sur les variables $x_1 < x_2 < \ldots < x_n$, et soit $f_u(x_1, x_2, \ldots, x_n)$ la fonction booléenne calculée par $u$. Nous dénotons par $[\![u]\!] = \{b \in \{0,1\}^n : f_u(b) = 1\}$ l'ensemble des affectations qui satisfont $f_u$. Pour toute affectation $b \in \{0,1\}^n$, nous dénotons par $\text{val}(b)$ la valeur décimale de $b$. Par exemple, $\text{val}(1101) = 13$, $\text{val}(0110) = 6$ et $\text{val}(0000) = 0$.

Donnez un algorithme qui résoud le problème suivant en *temps polynomial*:

| | |
|---|---|
| ENTRÉE: | un BDD $B$ sur les variables $x_1 < x_2 < \cdots < x_n$, et un sommet $u \neq 0$ de $B$ |
| SORTIE: | $\min\{\text{val}(b) : b \in [\![u]\!]\}$ |

Autrement dit, votre algorithme doit retourner la plus petite affectation qui satisfait la fonction booléenne associée à $u$.