E.S.I 2009/2010 – EMD1 – MCP 4SI(Q) – Durée 1h30 – Doc. interdit

Barème : (5 + 5 + 5 + 5)

1) Donner la complexité de l'algorithme suivant:

```
x := a;  y := b;
TQ  (x <= y)
        x := x+1;  y := y-1
FTQ
```

2) Donner une réduction fonctionnelle entre les BJ_n schémas et les D-schémas ($BJ_n \leq_{FN} D$)

3) En utilisant le système de preuve de HOARE, trouver la plus faible précondition E nécessaire pour que l'énoncé ' E {P} (a=fib(n)) ' soit un théorème. P étant le programme suivant:

```
i := 1;  a := 1;  b := 1;
TQ  ( i < n )
        i :=  i+1;  u := a;  a := a+b;   b := u
FTQ
```

4) Prouver l'énoncé

```
(n >= 0)
{
        x:=a;  y:=n;  r:=1;
        TQ  (y > 0)
           SI (y mod 2 = 0)
                x := x*x;  y := y/2
           SINON
                r := r*x;   y := y-1
           FSI
        FTQ
}
```

($r = a^n$)

Rappel (système formel de Hoare)

Aff: t(exp) { x := exp } t(x)

Seq: E{P}F , F{Q}S |-- E{P;Q}S

Imp1: E ⇒ F , F{P}S |-- E{P}S

Imp2: E{P}F , F ⇒ S |-- E{P}S

Cnd1: E&B{P}S , E & ¬B ⇒ S |-- E{ SI (B) P FSI } S

Ite: E&B{P}E |-- E{ TQ (B) P FTQ} E & ¬B

Cnd2: E&B{P}S , E & ¬B{Q}S |-- E{ SI (B) P SINON Q FSI } S

Rappel (Suite de Fibonacci)

fib(0 = fib(1) = 1

fib(n) = fib(n-1) + fib(n-2) pour n ≥2

Rappel (BJn et D schémas)

- Equation à point fixe pour les langages Bjn :

BJ = A / BJ;BJ / si P alors BJ sinon BJ fsi / faire $\underbrace{P:BJ; P:BJ; ...P:BJ}_{k \text{ fois, avec } 1<= k <= n}$ fait

- Equation à point fixe pour les langages D :

D = A / D;D / si P alors D sinon D fsi / tantque P faire D fin

avec A l'ensemble des actions simples, P l'ensemble des symboles de prédicats.

Corrigé EMD1 2009/2010 – MCP 4SI(Q)

1) Donner la complexité de l'algorithme suivant:

```
x := a;  y := b;
TQ  (x <= y)
        x := x+1;  y := y-1
FTQ
```

La complexité des affectations et du test est o(1).
Le nombre d'itérations de la boucle TQ est aux environs de (b-a)/2 car elle positionne x et y au milieu de l'intervalle [a,b].
La complexité du programme est donc o(b-a)

2) Donner une réduction fonctionnelle entre les BJ_n schémas et les D-schémas ($BJ_n \leq_{FN} D$)

Pour trouver une réduction fonctionnelle entre 2 schémas L1 et L2, il suffit de donner une transformation de L1 vers L2.
La seule différence entre les BJ_n schémas et les D-schémas est l'instruction de boucle. Il suffit donc de montrer qu'une boucle de type

```
Faire
        P1: B1;
        P2: B2;
        …
        Pk: Bk
Fait    (où les Pi sont des prédicats et les Bi des bloc d'instructions Bjn)
```

se traduit en une boucle TQ ayant la forme:

```
continu := vrai;
TQ  continu
        SI P1
                Trans(B1);
                SI P2
                        Trans(B2);
                        SI P3
                                Trans(B3);
                                …
                                                        SI Pk
                                                                Trans(Bk)
                                                        SINON
                                                                continu := faux;
                                                        FSI
                                ...
                        SINON
                                continu := faux
                        FSI
                SINON
                        continu := faux
                FSI
        SINON
                continu := faux
        FSI
FTQ
```

où Trans(Bi) applique la même transformation par récurrence sur le bloc Bi pour obtenir un bloc d'instructions de type D équivalent.
'continu' est une nouvelle variable booléenne à rajouter à chaque application de la transformation.

3) *En utilisant le système de preuve de HOARE, trouver la plus faible précondition E nécessaire pour que l'énoncé ' E {P} (a=fib(n)) ' soit un théorème. P étant le programme suivant:*
i := 1; a := 1; b := 1;
TQ (i < n)
i := i+1; u := a; a := a+b; b := u
FTQ

Soient :
P1 = [i := 1; a := 1; b := 1]
P2 = [TQ (i < n) P3 FTQ]
P3 = [P4; b := u]
P4 = [P5; a := a+b]
P5 = [i := i+1; u := a]
S = (a=fib(n))

Pour que E {P} S soit vrai, il suffit, d'après la règle (SEQ) que :
a) E{P1}F et b) F{P2}S soient vrais
Pour que F{P2}S soit vrai, il suffit d'après la règle (IMP2) que:
c) F{P2}(F&i>=n) et d) (F&i>=n) => S. F étant un invariant de la boucle TQ
Pour que l'implication d) soit vraie, on peut choisir pour le prédicat F, la condition suivante:
F : (a=fib(i) & i<=n)
L'énoncé c) (F{P2}(F&i>=n)) est vrai d'après la régle (ITE) si l'énoncé suivant est vrai:
e) (F & i<n) {P3} F
Ce dernier est vrai par (SEQ) si les 2 énoncés suivants sont vrais:
f) (F & i<n) {P4}G et g) G{b:=u}F
De même f) est vrai par (SEQ) si : h) (F & i<n) {P5}H et i) H{a:=a+b}G sont vrais
L'énoncé h) est vrai par (SEQ) si : j) (F & i<n) {i:=i+1}I et k) I{u:=a}H sont vrais.
Les énoncés g,i et k sont vrais par (AFF) en prenant comme préconditions respectivement:
G = [u/b]F c-a-d (a=fib(i) & i<=n)
H = [a+b/a]G c-a-d (a+b=fib(i) & i<=n)
I = [a/u]H c-a-d (a+b=fib(i) & i<=n)
L'énoncé j) est vrai par (IMP1) si les 2 énoncés suivant sont vrais:
l) (F & i<n) => J et m) J{i:=i+1}I
m est vrai par AFF en prenant comme précondition J = [i+1/i]I c-a-d (a+b=fib(i+1) & i+1<=n)
Cependant l'implication l) : (a=fib(i) & i<=n & i<n) => (a+b=fib(i+1) & i+1<=n) n'est pas vérifiée puisqu'on n'a aucune relation sur b dans son antécédent (a=fib(i) & i<=n & i<n).
Cela suggère de rajouter une telle relation dans l'invariant F
On prend alors comme invariant F : (a=fib(i) & b=fib(i-1) & i<=n)
L'implication d) reste toujours vrai
Les prédicats intermédiaires trouvés par AFF deviennent :
G = [u/b]F c-a-d (a=fib(i) & u=fib(i-1) & i<=n)
H = [a+b/a]G c-a-d (a+b=fib(i) & u=fib(i-1) & i<=n)
I = [a/u]H c-a-d (a+b=fib(i) & a=fib(i-1) & i<=n)
J = [i+1/i]I c-a-d (a+b=fib(i+1) & a=fib(i) & i+1<=n)
Maintenant la nouvelle implication l) est vraie:
l) (a=fib(i) & b=fib(i-1) & i<=n & i<n) => (a+b=fib(i+1) & a=fib(i) & i+1<=n)

Pour que l'énoncé a) E{P1}F soit vrai par (SEQ), il faudrait que :
n) E{ i := 1; a := 1}K et o) K{b := 1}F soient vrais
L'énoncé o) est vrai par AFF en prenant K = [1/b]F c-a-d (a=fib(i) & 1=fib(i-1) & i<=n)
L'énoncé n) est vrai par (SEQ) si les 2 énoncés suivants sont vrais:
p) E{i:=1}L et q) L{a:=1}K
L'énoncé q) est vrai par AFF en prenant L = [1/a]K c-a-d (1=fib(i) & 1=fib(i-1) & i<=n)
L'énoncé p) est vrai par AFF en prenant E = [1/i]L c-a-d (1=fib(1) & 1=fib(0) & 1<=n)
Comme 1=fib(1) & 1=fib(0) sont toujours vrais, la plus faible précondition nécessaire pour que S soit vrai après la fin du programme est **E : n >=1**

4) Prouver l'énoncé
(n >= 0)
{
```
        x:=a;  y:=n;  r:=1;
        TQ  (y > 0)
           SI (y mod 2 = 0)
                x := x*x;  y := y/2
           SINON
                r := r*x;   y := y-1
           FSI
        FTQ
```
}
($r = a^n$)

Posons
E : (n>=0) S : ($r = a^n$)
P = [P1;P2]
P1 = [x:=a; y:=n; r:=1]
P2 = [TQ (y > 0) P3 FTQ]
P3 = [SI (y mod 2 = 0) P4 SINON P5 FSI]
P4 = [x := x*x; y := y/2]
P5 = [r := r*x;   y := y-1]

Pour que E{P}S soit vrai par SEQ, il faudrait que :
a) E{P1}F et b) F{P2}S soient vrais.
Pour que b) soit vrai par IMP2, il faudrait que:
c) F{P2}(F&y<=0) et d) **(F&y<=0) => S** (F étant l'invariant de la boucle TQ)
Pour que c) soit vrai par ITE, il faudrait que :
e) (F & y>0) {P3}F soit vrai
Pour que e) soit vrai par CND2, il faudrait que:
f) (F & y>0 & y mod 2 = 0) {P4} F et g) (F & y>0 & y mod 2 <> 0) {P5} F soient vrais
Pour que g) soit vrai par SEQ, il faudrait que :
h) (F & y>0 & y mod 2 <> 0) {r := r*x} G et i) G {y := y-1} F soient vrais.
L'énoncé i) est vrai AFF en prenant G = [y-1/y]F
Pour que h) soit vrai par IMP1, il faudrait que :
j) (F & y>0 & y mod 2 <> 0) => H et k) H { r := r*x} G soient vrais.
L'énoncé k) est vrai par AFF en prenant H = [r*x/r]G = [r*x/r]([y-1/y]F).
L'implication j) devient alors : **(F & y>0 & y mod 2 <> 0) => [r*x/r]([y-1/y]F)**

Pour que f) (F & y>0 & y mod 2 = 0) {x := x*x; y := y/2} F soit vrai par SEQ, il faudrait que:

l) (F & y>0 & y mod 2 = 0) {x := x*x} I et m) I {y := y/2} F soient vrais.
L'énoncé m) est vrai par AFF à condition de prendre I = [(y/2) / y]F
Pour que l) soit vrai par IMP1, il faudrait que :
n) (F & y>0 & y mod 2 = 0) => J et o) J {x := x*x} I soient vrais.
L'énoncé o) est vrai par AFF en prenant J = [x*x/x]I = [x*x/x]([(y/2) / y]F)
L'implication n) devient alors : **(F & y>0 & y mod 2 = 0) => [x*x/x]([(y/2) / y]F)**

Pour que l'énoncé a) E { x:=a; y:=n; r:=1 } F soit vrai par SEQ, il faudrait que:
p) E { x:=a; y:=n; } K et q) K { r:=1 } F soient vrais.
L'énoncé q) est vrai par AFF pour K = [1/r]F
Pour que p) soit vrai par SEQ, il faudrait que :
r) E { x:=a } L et s) L { y:=n } K soient vrais.
L'énoncé s) est vrai par AFF en prenant L = [n/y]K = [n/y]([1/r]F)
Pour que r) soit vrai par IMP1, il faudrait que :
t) E => M et u) M { x:=a } L soient vrais.
L'énoncé u) est vrai par AFF en prenant M = [a/x]L = [a/x]([n/y]([1/r]F))
L'implication t) devient alors E => **[a/x]([n/y]([1/r]F))**

Il ne reste qu'à trouver le bon invariant F qui vérifie les 4 implications :
d) (F&y<=0) => ($r = a^n$)
j) (F & y>0 & y mod 2 <> 0) => [r*x/r]([y-1/y]F)
n) (F & y>0 & y mod 2 = 0) => [x*x/x]([(y/2) / y]F)
t) (n >= 0) => [a/x]([n/y]([1/r]F))
En déroulant la boucle, on peut observer que la relation suivante reste toujours correcte:
$r(x^y) = a^n$
On propose alors **l'invariant F : $r(x^y) = a^n$ & y>=0**
L'implication d) est vérifiée car en rajoutant la condition y >= 0 à F, l'antécédent de l'implication devient équivalent à ($r(x^0) = a^n$) donc au conséquent aussi.
L'implication j) devient :
($r(x^y) = a^n$ & y >= 0 & y>0 & y mod 2 <> 0) => $rx(x^{y-1}) = a^n$ & y-1 >= 0
c-a-d :
($r(x^y) = a^n$ & y >= 0 & y>0 & y mod 2 <> 0) => $r(x^y) = a^n$ & y >= 1
qui est vraie car y>0 implique bien que y >= 1
L'implication n) devient :
($r(x^y) = a^n$ & y >= 0 & y>0 & y mod 2 = 0) => $r(xx)^{y/2} = a^n$ & y/2 >= 0 qui est vraie aussi car d'une part y>=0 => y/2>=0 et d'autre part car $(xx)^{y/2} = x^y$
L'implication t) devient :
(n >= 0) => ($1(a^n) = a^n$ & n >= 0) qui est vraie car
($a^n = a^n$ & n >= 0) est équivalent à (Vrai & n >= 0) qui est équivalent à (n >= 0)
et donc l'implication est triviale (de la forme A => A)