Verificação de Software e Sistemas
Teorias do Módulo da Satisfatibilidade
Terceira Lista de Exercícios

1) **(Solução de Equações)** Resolva as seguintes equações com o SMT solver Z3:

a) 2x - y + 3z = 0
4x + 2y - z = 0
x - y + 2z = 0

b) 4x1 + 5x2 + 2x3 = 42
3x1 + 3x2 + 2x3 = 27
2x1 + 2x2 + 2x3 = 33

2) **(Satisfatibilidade de fórmulas lógicas)** Utilizando o Z3 determine se as seguintes fórmulas são satisfeitas:

a) $(\neg a \vee \neg b) \wedge (\neg b \vee c) \wedge b$
b) $(((d \wedge c) \vee (p \wedge \neg ((c \wedge \neg (d))))) \equiv ((c \wedge d) \vee (p \wedge c) \vee (p \wedge \neg (d))))$
c) $(((a) \wedge ((b) \rightarrow \neg (a)) \equiv \neg (b))$

3) **(C API do Z3)** Desenvolver um algoritmo em C, utilizando a C API do Z3, para verificar os códigos abaixo:

a)

```
float c; int n=4;
while (n>0){
  c = 10/n;
  n--;
}
```

b)

```
#include <assert.h>
static float sqrt(float val) {
   val = nondet_float(); //assign a non-deterministic value to the variable val
   __ESBMC_assume(val>0 && val<1000);
   float x = val/10;
   float dx;
   double diff;
   double min_tol = 0.00001;
   int i, flag;
   flag = 0;
   if (val == 0 ) x = 0;
```

```
   else {
      for (i=1;i<20;i++) {
         if (!flag) {
            dx = (val - (x*x)) / (2.0 * x);
            x = x + dx;
            diff = val - (x*x);
            if (fabs(diff) <= min_tol) flag = 1;
         }
         else
            x =x;
      }
   }
   return (x);
}
void main(void) {
  assert(sqrt(4)==2);
  assert(sqrt(16)>=4);
  assert(sqrt(10)>=3,16);
  assert(sqrt(50)>=7,07);
}
```

4) **(Tradução de Código)** Desenvolver um tradutor de código C para script SMT-LIB 2 para os seguintes programas:

a)

```
float c; int n=4;
while (n>0){
   c = 10/n;
   n--;
}
```

b)

```
int main() {
  int a[2], i, x;
  if (x==0)
    a[i]=0;
  else
    a[i+2]=1;
  assert(a[i+1]==1);
}
```

c)

```
#include <assert.h>
static float sqrt(float val) {
   val = nondet_float(); //assign a non-deterministic value to the variable val
```

```
   __ESBMC_assume(val>0 && val<1000);
   float x = val/10;
   float dx;
   double diff;
   double min_tol = 0.00001;
   int i, flag;
   flag = 0;
   if (val == 0 ) x = 0;
   else {
      for (i=1;i<20;i++) {
         if (!flag) {
            dx = (val - (x*x)) / (2.0 * x);
            x = x + dx;
            diff = val - (x*x);
            if (fabs(diff) <= min_tol) flag = 1;
         }
         else
            x =x;
      }
   }
   return (x);
}
void main(void) {
  assert(sqrt(4)==2);
  assert(sqrt(16)>=4);
  assert(sqrt(10)>=3,16);
  assert(sqrt(50)>=7,07);
}
```

Data da entrega: 31 de maio de 2016 (terça-feira).
Após esta data será descontado 2 pontos por dia de atraso.
A lista de exercícios deve ser resolvida e entregue individualmente.

24/05/2016