

PGENE549 – Verificação de Software e Sistemas
Terceira Lista de Exercícios

1) Descreva com a suas próprias palavras o que você entende sobre verificação de modelos (*model checking*) e como isto difere de outras técnicas usadas para verificação de hardware e software. A sua resposta deve indicar quais são as entradas para verificar o software e quais são as suas saídas, e quais benefícios e implicações isto tem. Você também deve incluir uma explicação dos termos *modelo* e *propriedade de corretude*.

2) Mostre que as duas expressões *if-then-else* abaixo são equivalentes:

$$!(a \mid b) ? h : ! (a == b) ? f : g \qquad ! (!a \mid !b) ? g : (!a \& \& !b) ? h : f$$

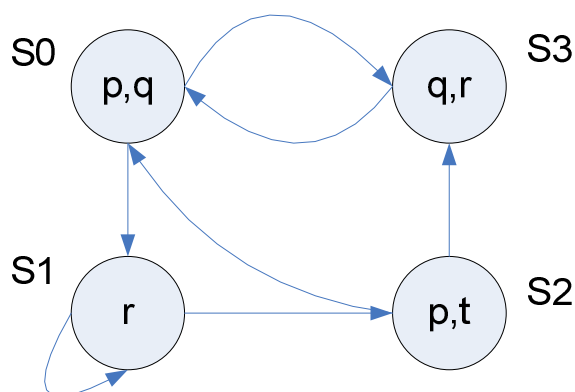
Você pode assumir que as variáveis têm somente um bit.

3) Considere três pessoas A, B e C que querem sentar em uma fileira, mas:

- A não quer sentar próximo do C.
- A não quer sentar na cadeira da esquerda.
- B não quer sentar no lado direito do C.

Escreva uma fórmula proposicional que é satisfeita se e somente se existe uma atribuição de assentos para as três pessoas que satisfaz todas as restrições. A fórmula é satisfeita? Caso positivo, forneça uma atribuição.

4) Considere a seguinte estrutura *Kripke*:



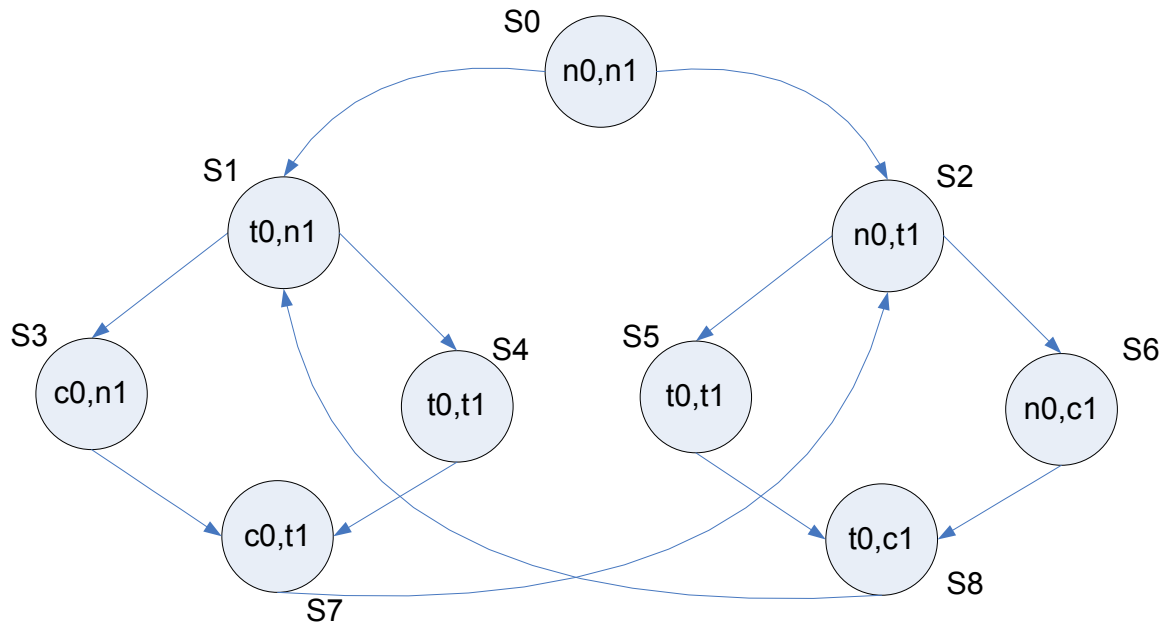
Verifique se S_0 satisfaz ϕ e S_2 satisfaz ϕ para as seguintes fórmulas CTL ϕ :

- a) **AF** q
- b) **AG EF** (p \vee r)

- c) **EX EX r**
- d) **AG AF q**

Explique a sua resposta.

5) Considere o seguinte modelo de Kripke da exclusão mútua:



onde as proposições atômicas n_i , t_i e c_i com $i = 0,1$ significa “processo i em uma seção não-crítica”, “processo i tentando entrar na região crítica” e processo i na região crítica”, respectivamente. Aplique o algoritmo de verificação de modelo do CTL para checar se as seguintes fórmulas se mantêm no estado S_0 deste modelo:

- a) **AG** $\neg(C_0 \wedge C_1)$
- b) **AG**(($t_0 \Rightarrow \mathbf{AF} C_0$) \wedge ($t_1 \Rightarrow \mathbf{AF} C_1$))

Note que você precisará transformar estas fórmulas para as aquelas que são semanticamente equivalentes para que você possa aplicar o algoritmo.

6) Expresse as seguintes propriedades em CTL sempre que possível:

- a) Depois de p , q nunca é verdadeiro. (esta restrição é requerida em todos os possíveis caminhos de computação.)
- b) O evento p precede ambos s e t em todos os caminhos de execução. (você pode achar mais fácil expressar primeiro a negação desta propriedade.)
- c) Entre os eventos q e r , o evento p nunca é verdadeiro.

7) Construa uma estrutura Kripke para a implementação de uma versão simplificada da técnica de inserção de bytes (*byte stuffing*) usada na camada de enlace em redes de computadores.

```
#define DLE 16
#define STX 2
#define ETX 3
int main (void) {
    uchar in[6] = {DLE, STX, NUL, DLE, ETX, '\0'};
    uchar out[6];
    int i = 0;
    int j = 0;
    while (in[i] != '\0') {
        switch (in[i]) {
            case (DLE):
                if (in[i+1]==STX || in[i+1]==ETX) {
                    out[j] = in[i];
                } else {
                    out[j] = in[i];
                    out[++j] = DLE;
                };
                break;
            default:
                out[j] = in[i];
                break;
        }
        i++;
        j++;
    }
    out[j] = '\0';
    return 0;
}
```

Considere que a proposição atômica p verifica se $out[4]$ é igual a ETX e a proposição atômica q verifica se $out[5]$ é igual a ETX. Deste modo, verifique se a estrutura *Kripke* do código acima satisfaz a propriedade **AG** ($p \wedge q$).

Data de entrega: 16 de dezembro de 2014 (terça-feira).

Após esta data será descontado 2 pontos por dia de atraso.

A lista de exercícios deve ser resolvida e entregue individualmente.

02/12/2014