
Model Checking no SMV

Érique Moser
DAS410099 - Verificação Formal

Sumário

1. Características;
2. Estrutura;
3. Instalação e Execução;
4. Exemplo: Elevador;
5. Problema do Escoamento Multifásico.
6. Aplicação prática
7. Referências

Características

Origem

- SMV:
 - Desenvolvido por K. L. McMillan;
 - Orientado por E. M. Clarke.
 - Carnegie-Mellon University (Pittsburgh, PA, USA).
- Executa a verificação de modelo simbólico (BDD);
- Utiliza lógica CTL.

Características

- Utiliza composições hierárquicas e permite composição de autômatos, inclusive a utilização de variáveis compartilhadas;
- Os autômatos são descritos textualmente;
- Suposições de justiça podem ser incluídas;
- O SMV fornece um contra-exemplo quando a propriedade solicitada não é satisfeita;
- O SMV tem uma semântica muito declarativa e entende qualquer fórmula booleana ligando os valores atuais e futuros das variáveis.

Pós e Contras

Pontos positivos:

- É uma das ferramentas com maior probabilidade de verificar completamente um sistema complexo.

Pontos negativos:

- Falta de facilidade para simulação;
- Linguagem escolhida para descrever os autômatos.

Estrutura

Estrutura

Módulos

Variável local

Atribuições

DEFINE

FAIRNESS

SPEC

Módulos

- Um programa SMV pode consistir em vários módulos:
 - Cada módulo é uma descrição encapsulada que pode ser instanciada várias vezes dentro do modelo.
- Um módulo pode conter instâncias de outros módulos, permitindo que uma hierarquia estrutural seja construída;
- Cada descrição SMV deve conter um módulo chamado main, sem parâmetros funcionais:
 - Esse módulo forma a raiz da hierarquia do modelo e o ponto de partida.

Variável Local

- Abreviada por VAR;
- Lista todos os estados (na forma de conjuntos);
- Uma variável pode ser:
 - booleana;
 - tipo de enumeração;
 - subintervalo inteiro.

Atribuições

- Abreviado por ASSIGN;
- O valor das variáveis em cada estado é definido usando init e next;
- São as atribuições dos estados:
 - inicial;
 - próximo.
- Sua concepção está orientada para descrever uma relação de "possível próximo estado"
 - a notação "next" se aplica para variáveis e não ao estado completo, o que permite decompor alterações em várias partes

DEFINE

- Atribui o valor de uma expressão à um símbolo;
- Poderia ser feito utilizando VAR e ASSING, mas símbolos dessa forma não aumentam o espaço de estados;
- Por outro lado, símbolos não aceitam valor não determinístico;
- Para DEFINE, o tipo de variável não precisa ser declarada.

FAIRNESS

- Suposições de justiça podem ser incluídas;
- Implica algum comportamento que o sistema deve ter.

SPEC

- Utiliza sintaxe CTL;
- Exemplos:
 - AG EX TRUE
 - verifica se é livre de deadock
 - AG((request = Tr) -> AF state = busy)
 - se houver uma requisição, o sistema ficará ocupado

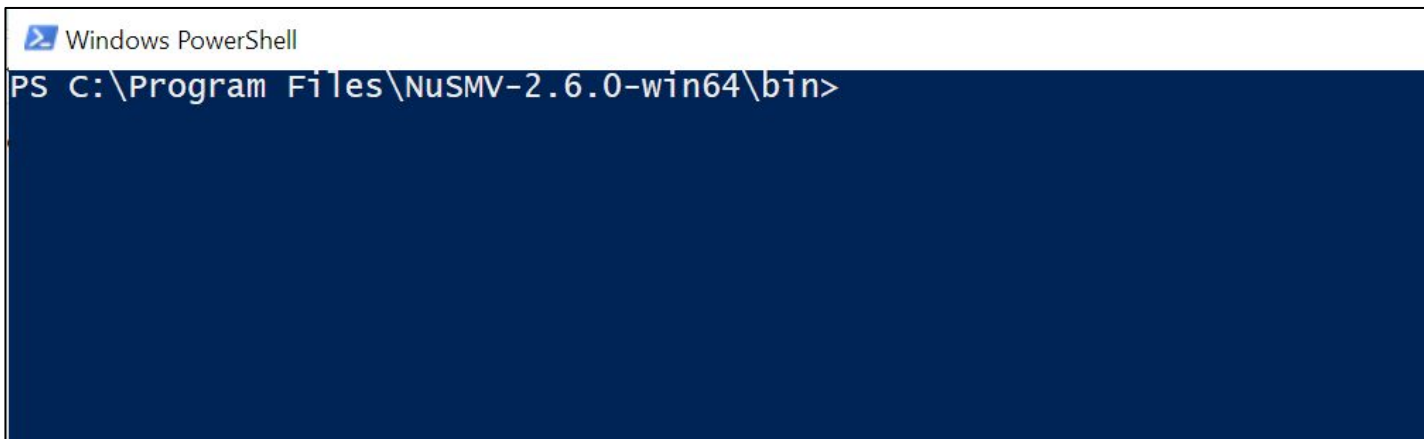
Instalação

Passos

1. Download em:
<https://nusmv.fbk.eu/NuSMV/download/getting-v2.html>
2. Extrair o arquivo .tar.gz e colar a pasta em *C:\Program Files*
3. Colocar a pasta *C:\Program Files\NuSMV-2.6.0-win64\bin* em **Path** nas variáveis de ambiente

Execução

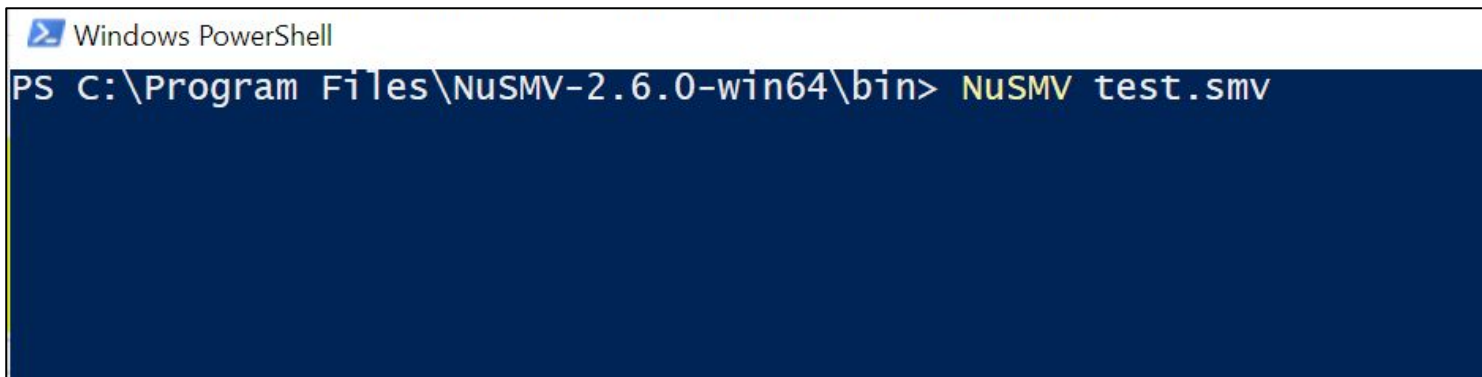
- Os códigos podem ser escritos no NotePad com a extensão .smv e devem ser colocados na pasta /bin do NuSMV
- O *prompt* deve ser aberto na pasta /bin onde foi colocado o SMV:



```
Windows PowerShell
PS C:\Program Files\NuSMV-2.6.0-win64\bin>
```

Execução

- O arquivo test.smv pode ser aberto utilizando o comando:
NuSMV test.smv

A screenshot of a Windows PowerShell terminal window. The title bar at the top says "Windows PowerShell". The command prompt shows the current directory as "C:\Program Files\NuSMV-2.6.0-win64\bin" and the command "NuSMV test.smv" has been entered. The terminal background is dark blue.

```
Windows PowerShell
PS C:\Program Files\NuSMV-2.6.0-win64\bin> NuSMV test.smv
```

- Uma vez feito isso, o model checker retornará com o resultado.

Exemplo: Elevador

Problema do elevador (Systems and Software Verification)

- Elevador de 4 andares;
- Pode ser requisitado em cada andar;
- Para simplificar a cabine sobe e desce de acordo com a sequência 0, 1, 2, 3, 2, 1, 0, 1, 2, 3, 2, etc.



Declaração das variáveis

- Variáveis:
 - cabine
 - de 0 a 3
 - direção
 - sobe e desce
 - pedidos
 - vetor de booleanos de posição 0 a 3
 - indica se há pedido pendente no piso

```
MODULE main
VAR
    cabin : 0 .. 3;
    dir : {up,down};
    request: array 0 .. 3 of boolean;
```

Relações de transição

- Da cabine:
 - indica em quais momentos a cabine pode subir, descer ou ficar parada;
- Da direção:
 - indica quando a cabine deve alterar o sentido da direção
 - vale ressaltar que nessa parte o SMV permite que a definição ocorra antes da definição do próximo valor da cabine
- A construção "case ... esac" é uma seleção de casos em que o autômato evolui conforme explícito na direita, sob determinadas condições da esquerda.

Relações de transição

ASSIGN

```
next(cabin) := case
    dir=up & cabin < 3 : cabin + 1;    --moves up
    dir=down & cabin > 0 : cabin - 1;    --moves down
    TRUE : cabin;                      --stuck
esac;

--sempre que chega em uma das extremidades, muda direção

next(dir) := case
    dir=up & next(cabin) = 3 : down;    --switch dir
    dir=down & next(cabin) = 0 : up;    -- switch dir
    TRUE: dir;                          --keep on
esac;
```

Evolução dos pedidos

- Indica qual momento um pedido pode aparecer e desaparecer;
- O modelo permite que a solicitação ocorra a qualquer momento, exceto se a cabine estiver no andar.

Evolução dos pedidos

```
next(request[0]) := case
    next(cabin) = 0 : FALSE;      --disappears
    request[0] : TRUE;            --remains
    TRUE : {FALSE,TRUE};
esac;
next(request[1]) := case
    next(cabin) = 1 : FALSE;      --disappears
    request[1] : TRUE;            --remains
    TRUE : {FALSE,TRUE};
esac;
next(request[2]) := case
    next(cabin) = 2 : FALSE;      --disappears
    request[2] : TRUE;            --remains
    TRUE : {FALSE,TRUE};
esac;
next(request[3]) := case
    next(cabin) = 3 : FALSE;      --disappears
    request[3] : TRUE;            --remains
    TRUE : {FALSE,TRUE};
esac;
```

Estados iniciais

- Indicam a posição inicial da cabine, da direção e dos pedidos.

```
init(cabin)      := 0;  
init(dir)        := up;  
init(request[0]) := FALSE;  
init(request[1]) := FALSE;  
init(request[2]) := FALSE;  
init(request[3]) := FALSE;
```

Verificação...

- Exemplos:
 - verificar se "todos os pedidos são eventualmente satisfeitos"
 - $AG (AF!request[0] \ \& \ AF!request[1] \ \& \ AF!request[2] \ \& \ AF!request[3])$
 - verificar se "os pedidos são eventualmente todos satisfeitos (simultaneamente)"
 - $AG \ AF \ (!request[0] \ \& \ !request[1] \ \& \ !request[2] \ \& \ !request[3])$

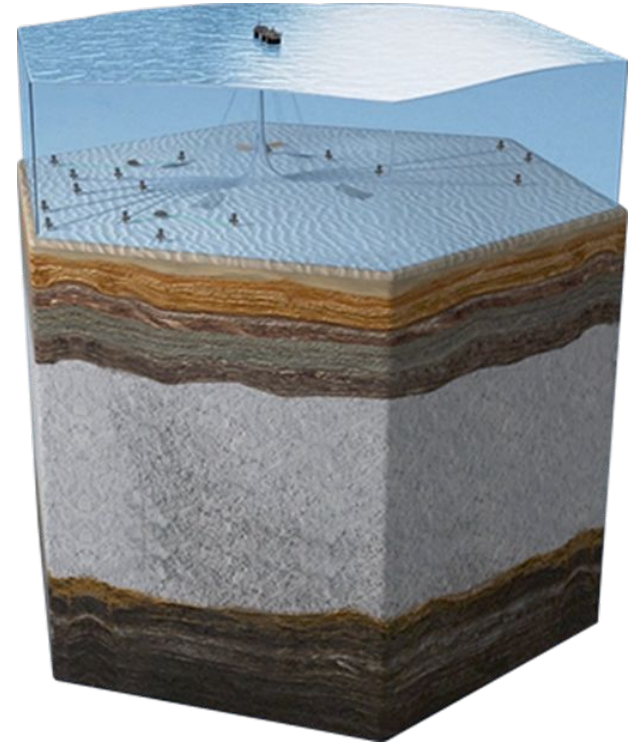
Verificação na prática

->Aplicação no NuSMV

Problema do Escoamento Multifásico

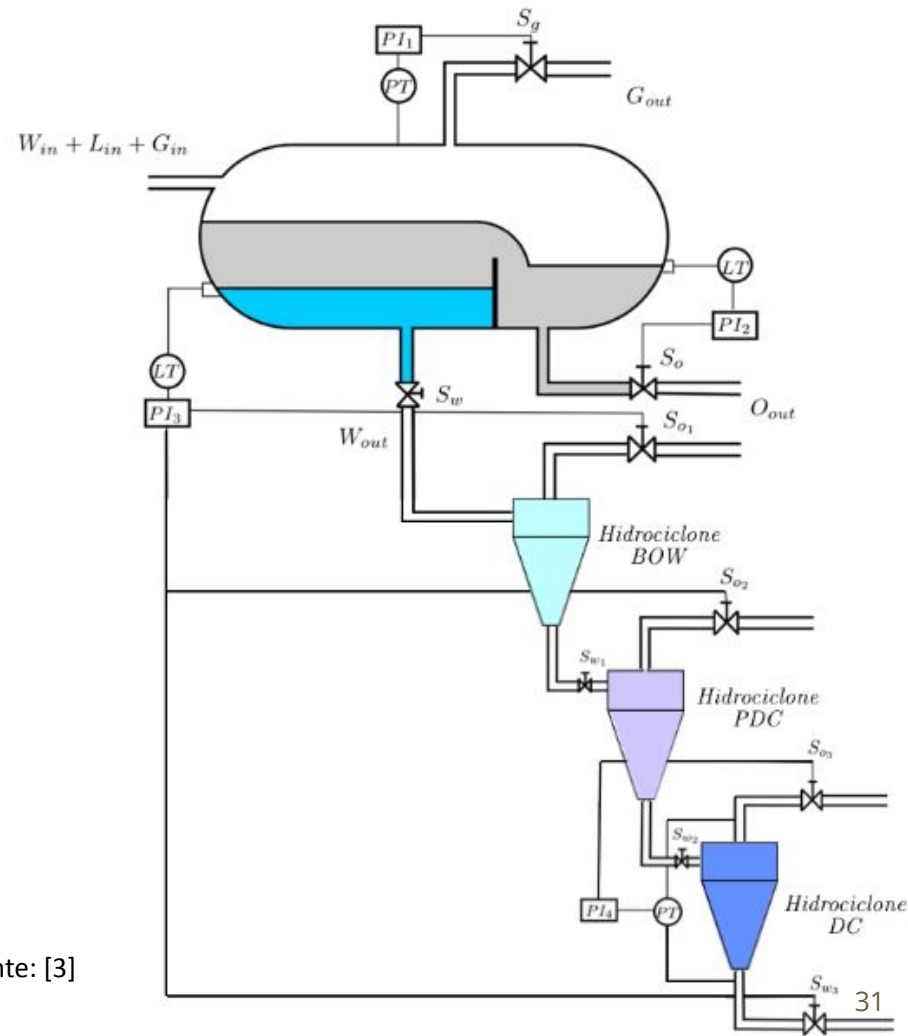
Tratamento Primário de Petróleo

- Constitui um projeto na área de Petróleo e Gás.
- O processamento primário é o primeiro tratamento do petróleo na Plataforma.
- O objetivo é efetuar a separação gás/óleo/água.



Fonte: [2]

- Envolve criticidade devido ao fato de que nenhum reservatório pode exceder seus níveis de capacidade, evitando assim acidentes.
- Esse sistema é composto por
 - i) separador: que efetua a separação trifásica do gás, líquido e água;
 - ii) três Hidrociclones: que livram a água do óleo remanescente.



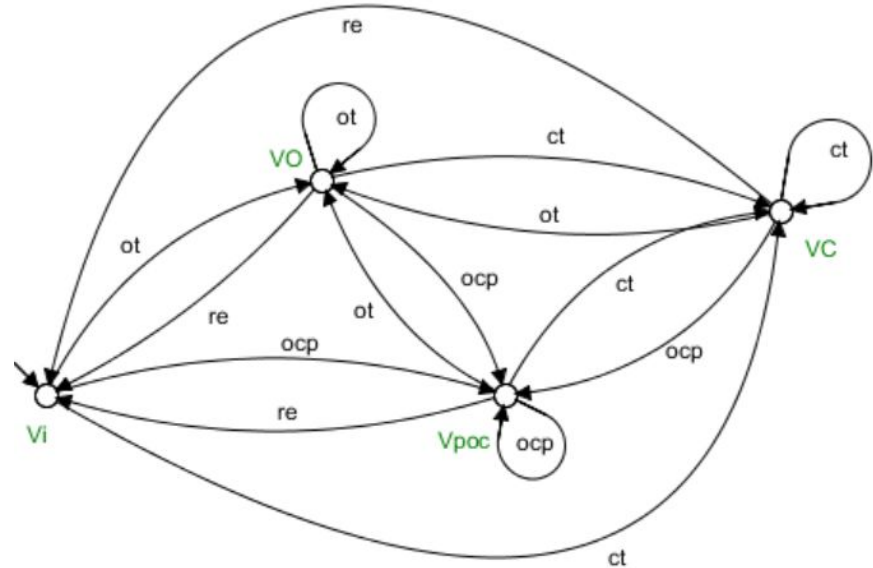
Fonte: [3]

Modelagem da planta

- Corresponde à válvula de controle e o controlador de nível;
- Comportamento adequado às válvulas S0, S01, S02, S03 e Sw3 com ação similar à válvula de pressão Sg: o comando de abrir a válvula ocorre quando o valor da variável está acima do setpoint, e vice versa.

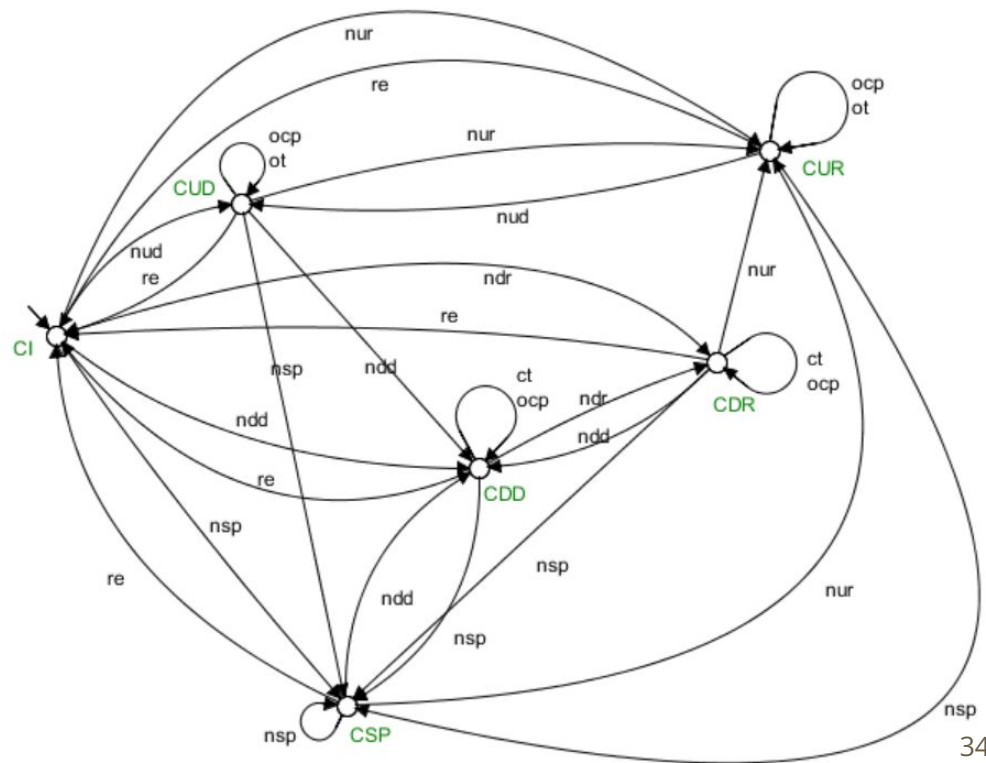
Modelagem válvula

Estado	Descrição
VC	totalmente fechada
VO	totalmente aberta
VPOC	parcialmente aberta
Vi	estado inicial
Evento	Descrição
ocp	abrir ou fechar em posição intermediária
ot	abrir a válvula totalmente
ct	fechar totalmente
re	reset



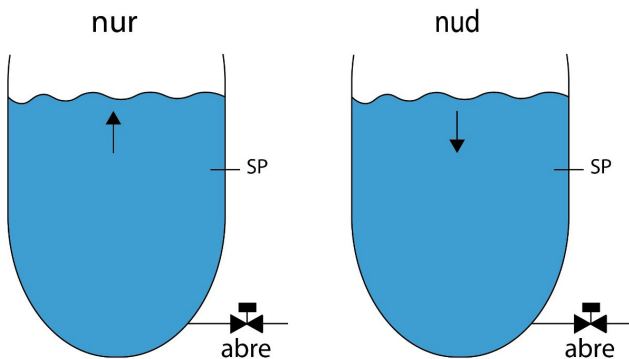
Modelagem controlador

Estado	Descrição
CSP	nível no setpoint
CDD	abaixo do setpoint, descendo
CDR	abaixo do setpoint, subindo
CUD	acima do setpoint, descendo
CUR	acima do setpoint, subindo
CI	estado inicial
Evento	Descrição
nsp	sensor no setpoint
ndd	nível abaixo do setpoint, descendo
ndr	nível abaixo do setpoint, subindo
nud	nível acima do setpoint, descendo
nur	nível acima do setpoint, subindo
re	evento de reset

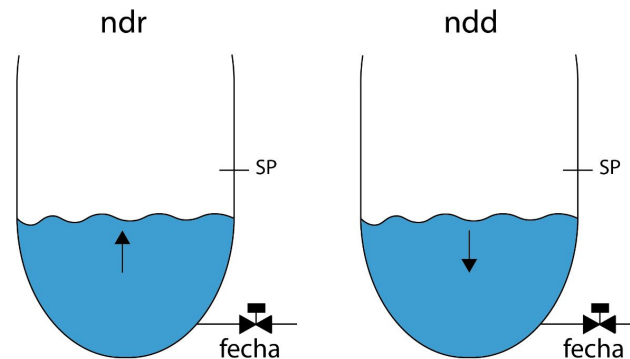


Configuração dos níveis

Nível Alto



Nível Baixo



Verificação

- Dado o sistema completo com a composição feita, é possível percorrer o autômato para saber se há algum deadlock, se existe alguma situação perigosa: ex: nível do reservatório alto e o controlador mandando encher o reservatório.

Aplicação prática

Descrição das variáveis

- Variáveis:
 - válvula
 - VI, VO, VC, VPOC
 - movimento
 - ot, ct, ocp, re
 - controlador
 - CI, CSP, CDD, CDR, CUD, CUR
 - nível
 - nsp, ndd, ndr, nud, nur, re

```
MODULE main
VAR
    valvula : {VI,VO,VC,VPOC};
    movimento : {ocp,ot,ct,re};
    controlador : {CI,CSP,CDD,CDR,CUD,CUR};
    nivel : {nsp,ndd,ndr,nud,nur,re};
```

Descrição da válvula

Relação de transição da válvula:

- Indica qual posição a válvula pode assumir a depender do movimento que ela pode executar.
 - EX: se a válvula estiver aberta e ocorrer o movimento de fechar a válvula, o próximo estado será válvula fechada.

```
ASSIGN
  next(valvula):= case
    valvula=VI & movimento=ot : VO;
    valvula=VI & movimento=ocp : VPOC;
    valvula=VI & movimento=ct : VC;
    valvula=VO & movimento=re : VI;
    valvula=VO & movimento=ot : VO;
    valvula=VO & movimento=ct : VC;
    valvula=VO & movimento=ocp : VPOC;
    valvula=VPOC & movimento=ocp : VPOC;
    valvula=VPOC & movimento=re : VI;
    valvula=VPOC & movimento=ot : VO;
    valvula=VPOC & movimento=ct : VC;
    valvula=VC & movimento=ct : VC;
    valvula=VC & movimento=re : VI;
    valvula=VC & movimento=ot : VO;
    valvula=VC & movimento=ocp : VPOC;
    TRUE : {VI,VO,VC,VPOC};
  esac;
```

Descrição do autômato

Relação de transição do controlador:

- Indica qual estado o controlador pode assumir a depender do nível que o tanque se encontra.
 - Ex: se o controlador estiver em estado de setpoint e o nível aumentar, o controlador assume o estado controlador em nível alto.
 - Caso o controlador estiver em nível alto, a válvula pode realizar o movimento de abrir.

```
next(controlador):= case
  controlador=CI & nivel=nur : CUR;
  controlador=CI & nivel=nud : CUD;
  controlador=CI & nivel=ndr : CDR;
  controlador=CI & nivel=ndd : CDD;
  controlador=CI & nivel=nsp : CSP;
  controlador=CSP & nivel=nsp : CSP;
  controlador=CSP & nivel=re : CI;
  controlador=CSP & nivel=ndd : CDD;
  controlador=CSP & nivel=nur : CUR;
  controlador=CUR & (movimento=ocp | movimento=ot) : CUR;
  controlador=CUR & nivel=re : CI;
  controlador=CUR & nivel=nud : CUD;
  controlador=CUR & nivel=nsp : CSP;
  controlador=CUD & (movimento=ocp | movimento=ot) : CUD;
  controlador=CUD & nivel=re : CI;
  controlador=CUD & nivel=nur : CUR;
  controlador=CUD & nivel=ndd : CDD;
  controlador=CUD & nivel=nsp : CSP;
  controlador=CDD & (movimento=ocp | movimento=ct) : CDD;
  controlador=CDD & nivel=re : CI;
  controlador=CDD & nivel=nsp : CSP;
  controlador=CDD & nivel=ndr : CDR;
  controlador=CDR & (movimento=ocp | movimento=ct) : CDR;
  controlador=CDR & nivel=re : CI;
  controlador=CDR & nivel=ndd : CDD;
  controlador=CDR & nivel=nsp : CSP;
  controlador=CDR & nivel=nur : CUR;
  TRUE : {CI,CSP,CDD,CDR,CUD,CUR};
esac;
```


Condição inicial

- Indica a posição inicial da válvula e do controlador.

```
init(valvula)      := VI;  
init(controlador) := CI;
```

Verificação de propriedades

Acessibilidade;

Segurança;

Vivacidade;

Liberdade de deadlock;

Justiça.

Acessibilidade

Alguma situação pode ser alcançada

(A1) "O controlador sempre pode retornar ao SetPoint"

AG (EF controlador=CSP)

(A2) "Não podemos ter um estado de nível subindo e uma válvula fechada"

**!EF (nivel=nur &
valvula=VC)**

(A3) "Sempre podemos retornar para uma condição inicial"

**AG EF (controlador=CI &
valvula=VI)**

(A4) "Verifica se no futuro cada um dos estados serão alcançados um dia"

**AG EF ((controlador=CI |
controlador=CSP | controlador=CDD |
controlador=CDR |
controlador=CUD | controlador=CUR))**

Segurança

Um evento nunca pode ocorrer

(S1) "Desde que a válvula não esteja fechada, o nível do tanque não sobre"

**A[!(nivel=nur) U
(valvula=VC)]**

(S2) "A válvula e o controlador serem resetados nunca ocorrerá"

**AG !(movimento=re &
nivel=re)**

Vivacidade

Sob certas condições, algum evento acabará por ocorrer

(V1) "Se a válvula abrir, o nível do tanque vai abaixar eventualmente"

**AG(movimento=ot -> EF
nivel=ndd)**

(V2) "Verifica se um determinado estado do controlador não vai ocorrer sempre"

**AG ((controlador=CI -> EF !(controlador=CI)) &
(controlador=CSP -> EF !(controlador=CSP)) &
(controlador=CDD -> EF !(controlador=CDD)) &
(controlador=CDR -> EF !(controlador=CDR)) &
(controlador=CUD -> EF !(controlador=CUD)) &
(controlador=CUR -> EF !(controlador=CUR)))**

Liberdade de deadlock

Algum evento indesejável nunca ocorrerá

(D1) "Qualquer que seja o estado alcançado, existirá um sucessor imediato"

AG EX TRUE

Justiça

Sob certas condições, um evento ocorrerá (ou deixará de ocorrer) indefinidamente

(J1) "O nível do tanque subirá infinitamente frequentemente"

AG AF (nivel=nur)

(J2) "O nível do tanque descera infinitamente frequentemente"

AG AF (nivel=ndd)

$$A \overset{\infty}{F} P \equiv AG AF P.$$

Referências

Referências

- [1] DREAMSTIME. Homens e mulheres no elevador, vetor desenhado à mão. Disponível em: <https://pt.dreamstime.com/homens-e-mulheres-no-elevador-vetor-desenhado-%C3%A0-m%C3%A3o-desenho-de-image166262561>. Acesso em: 09 mar. 2022.
- [2] PETROBRAS. Pré-Sal. Disponível em: <https://petrobras.com.br/pt/nossas-atividades/areas-de-atuacao/exploracao-e-producao-de-petroleo-e-gas/pre-sal/>. Acesso em: 18 set. 2021.
- [3] Martins, A. E. A. (2018). *(TCC) Diagnose de falhas de uma unidade de separação trifásica usando modelos a eventos discretos*. Escola Politécnica, UFRJ.
- [4] Bérard, B., Bidoit, M., Finkel, A., Laroussinie, F., Petit, A., Petrucci, L., Schnoebelen, P., & McKenzie, P. (2001). **Systems and Software Verification**. In *Systems and Software Verification*. Springer Berlin Heidelberg. <https://doi.org/10.1007/978-3-662-04558-9>
- [5] Cavada, R., Cimatti, A., Keighren, G., Olivetti, E., Pistore, M., & Roveri, M. (n.d.). **NuSMV 2.6 Tutorial**.
- [6] FBK. **NuSMV: a new symbolic model checker**. Disponível em: <https://nusmv.fbk.eu/>. Acesso em: 09 mar. 2022.
- [7] UNIVERSITY, Carnegie Mellon. **Model Checking Guided tour**. Disponível em: <http://www.cs.cmu.edu/~modelcheck/tour.htm>. Acesso em: 09 mar. 2022.

Obrigado

E-mail: erique.moser@posgrad.ufsc.br

Telefone: (48) 99164-0985

Site: [linkedin.com/in/eriquemoser](https://www.linkedin.com/in/eriquemoser)