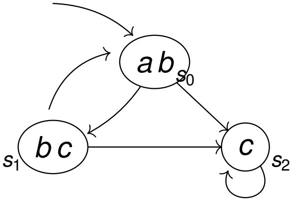
Verificação de Modelos
(Model Checking)

Estes slides são baseados nas notas de aula da Profa.
Corina Cîrstea

Lógica Temporal Linear (LTL)

- Estrutura Kripke:



- suposição adicional: cada estado tem no mínimo um sucessor $\Rightarrow$ processos infinitos !

- alguns caminhos de computação:

$$s_0 \rightarrow s_1 \rightarrow s_0 \rightarrow s_1 \rightarrow \ldots$$
$$s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow \ldots$$
$$s_0 \rightarrow s_2 \rightarrow s_2 \rightarrow \ldots$$

- fórmulas LTL são interpretadas através de *caminhos de computação*

Sintaxe da LTL

Fixar um conjunto *Prop* de proposições atômicas.

fórmulas LTL são de dois tipos:

- **fórmulas de caminho**:
  - $\mathtt{tt}$, p onde p é uma proposição atômica
  - se f e g são fórmulas de caminho, então também são:

$\neg f$	not f
$f \wedge g$	f and g
$\mathbf{X}\, f$	at the ne**X**t point in time, f
$\mathbf{F}\, f$	at some point in the **F**uture, f
$\mathbf{G}\, f$	**G**lobally (at all future points) f
$f\, \mathbf{U}\, g$	f **U**ntil g

- **fórmulas de estado:**

$\mathbf{A}\, f$	along **A**ll computation paths, f holds

- **prioridades de ligação:** operadores unários ; $\mathbf{U}$; $\wedge$ e $\vee$; $\rightarrow$

Semântica da LTL

Fixar uma estrutura Kripke $M = (S, R, V)$.

A semântica da LTL define:

- quando um caminho de computação π através de M satisfaz uma *fórmula de caminho f*,

  Notação: $\pi \models f$ se π satisfaz f

- quando um estado s de M satisfaz uma *fórmula de estado* ϕ.

  Notação: $s \models \phi$ se s satisfaz ϕ

Significado dos Operadores Temporais (Pictoricamente)

Considere que $s_0 \longrightarrow s_1 \longrightarrow s_2 \longrightarrow \ldots$ seja um caminho de computação.

$s_0 \to \ldots$	$s_0 \to \ldots$	$s_1 \to \ldots$	$\ldots$	$s_i \to \ldots$	$\ldots$
X f	$\ldots$	f	$\ldots$	$\ldots$	$\ldots$
F f	$\ldots$	$\ldots$	$\ldots$	f	$\ldots$
G f	f	f	f	f	f
f **U** g	f	f	f	g	$\ldots$

Note:

- f aqui é uma fórmula de caminho, por isso é interpretado através de *caminhos* !!

Semântica do LTL (Continuação)

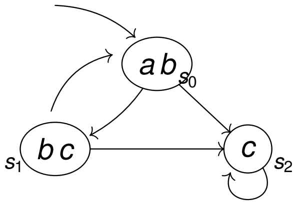
- Dado $\pi = s_0 \to s_1 \to s_2 \to \ldots$, considere $\pi^i = s_i \to s_{i+1} \to \ldots$.
- Definir quando uma fórmula de caminho f se mantém em π:

$\pi \models \mathtt{tt}$		
$\pi \models p$	sse	$p \in V(s_0)$
$\pi \models \neg f$	sse	$\pi \models f$ não é verdadeira
$\pi \models f \wedge g$	sse	$\pi \models f$ e $\pi \models g$
$\pi \models \mathbf{X}\, f$	sse	$\pi^1 \models f$
$\pi \models \mathbf{F}\, f$	sse	existe i tal que $\pi^i \models f$
$\pi \models \mathbf{G}\, f$	sse	$\pi^i \models f$ para todo $i \geq 0$
$\pi \models f\, \mathbf{U}\, g$	sse	existe i tal que $\pi^0 \models f, \ldots, \pi^{i-1} \models f, \pi^i \models g$

- Finalmente, definir quando uma fórmula de estado $\mathbf{A}\, f$ se mantém *em um estado $s \in S$*:

  $s \models \mathbf{A}\, f$ sse $\pi \models f$ para todos os caminhos π iniciando em s

Semântica do LTL – Exemplo



$s_0 \to s_1 \to s_0 \to s_1 \to \ldots \models \mathbf{X}\, c$

$s_0 \to s_1 \to s_0 \to s_1 \to \ldots \models \mathbf{F}\, c$

$s_0 \to s_1 \to s_0 \to s_1 \to \ldots \not\models \mathbf{G}\, c$

$s_2 \to s_2 \to s_2 \to s_2 \to \ldots \models \mathbf{G}\, c$

$s_0 \to s_1 \to s_2 \to s_2 \to \ldots \models \mathbf{F}\,\mathbf{G}\, c$

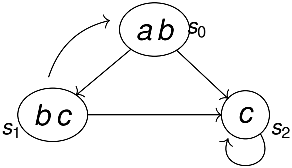
$s_0 \to s_1 \to s_2 \to s_2 \to \ldots \models b\,\mathbf{U}\, c$

$\pi \models \mathbf{G}\,\mathbf{F}\, c$ para qualquer π

Note:

- $\pi \models \mathbf{G}\,\mathbf{F}\, f$ sse f ocorre infinitas vezes ao longo de π.

Semântica do LTL – Exemplo



$s_0 \models \mathbf{A}\,(a \wedge b)$
$s_0 \models \mathbf{A}\ \mathbf{X}\,c$
$s_0 \not\models \mathbf{A}\ \mathbf{X}\,(b \wedge c)$
$s_0 \models \mathbf{A}\ \mathbf{F}\,c$

$s_0 \models \mathbf{A}\ \mathbf{F}\,a$
$s_0 \models \mathbf{A}\ \mathbf{G}\,\neg(a \wedge c)$
$s_1 \not\models \mathbf{A}\ \mathbf{G}\,c$
$s_2 \models \mathbf{A}\ \mathbf{G}\,c$
$s_0 \not\models \mathbf{A}\ \mathbf{G}\ \mathbf{F}\,a$
$s_0 \models \mathbf{A}\,(\,\mathbf{G}\ \mathbf{F}\,a \rightarrow\ \mathbf{G}\ \mathbf{F}\,c)$
$s_1 \models \mathbf{A}\ (b\,\mathbf{U}\,c)$
$s_2 \models \mathbf{A}\ (b\,\mathbf{U}\,c)$
$s_0 \models \mathbf{A}\ \mathbf{X}\ (b\,\mathbf{U}\,c)$

Note:

- $s \models \mathbf{A}\ \mathbf{G}\,f$ sse f se mantém em todos os estados alcançáveis a partir de s (incluindo s).
- $s \models \mathbf{A}\ \mathbf{G}\ \mathbf{F}\,f$ sse f ocorre infinitas vezes ao longo de cada caminho de s.

Alguns Padrões do LTL

- invariância (sempre): **A G** *p*

  “*p* permanece invariantemente verdadeiro ao longo de todo caminho”

- garantia (eventualmente): **A F** *p*

  “*p* eventualmente se tornará verdadeiro em todos os caminhos”

- estabilidade (nenhum progresso): **A F G** *p*

  “existe um ponto em cada caminho onde *p* irá se tornar invariantemente verdadeiro”

- recorrência (progresso): **A G F** *p*

  “se acontece de *p* ser falso em qualquer ponto em um caminho, *p* é sempre garantido se tornar verdadeiro novamente mais tarde”

  mesmo que: “*p* se mantenha infinitas vezes”

Alguns Padrões do LTL

- resposta: $\mathbf{A}\ \mathbf{G}\,(p \to \ \mathbf{F}\,q)$

  "qualquer estado satisfazendo *p* é eventualmente seguido por um estado satisfazendo *q*"

- precedência: $\mathbf{A}\ \mathbf{G}\,(p \to q\,\mathbf{U}\,r)$

  "a partir de qualquer estado satisfazendo *p*, o sistema irá continuamente satisfazer a propriedade *q* até que a propriedade *r* torne-se verdadeira"

- correlação: $\mathbf{A}\,(\,\mathbf{F}\,p \to \ \mathbf{F}\,q)$

  "se *p* detém em algum momento no futuro, o mesmo acontece com *q*"

De Volta a Exclusão Mútua

Proposições atômicas:

c_0 , c_1	(estado crítico)
n_0 , n_1	(estado não-crítico)
t_0 , t_1	(tentando entrar no estado crítico)

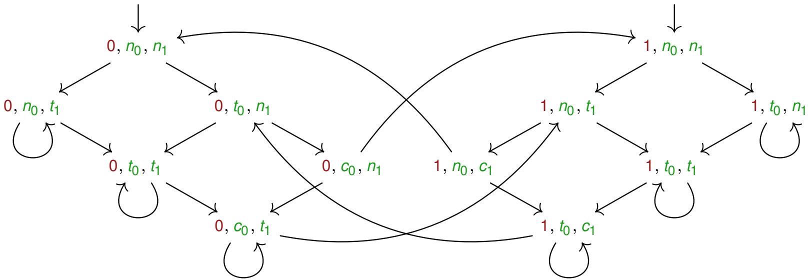
- exclusão mútua: no máximo um processo na região crítica *a qualquer momento*

  $$\mathbf{A}\ \mathbf{G}\, \neg(c_0 \wedge c_1)$$

- ausência de starvation: *sempre que* um processo tenta entrar na sua seção crítica, o mesmo *eventualmente* conseguirá

  $$\mathbf{A}\ \mathbf{G}\ ((t_0 \rightarrow \mathbf{F}\, c_0) \wedge (t_1 \rightarrow \mathbf{F}\, c_1))$$

Exclusão Mútua: Verificação de Corretude



- $\mathbf{A}\ \mathbf{G}\ \neg(c_0 \wedge c_1)$ ✓

  Precisa verificar que $\neg(c_0 \wedge c_1)$ é verdadeiro em todos os estados alcançáveis a partir dos estados iniciais.

- $\mathbf{A}\ \mathbf{G}\ ((t_0 \rightarrow \mathbf{F}\, c_0) \wedge (t_1 \rightarrow \mathbf{F}\, c_1))$ × ✓

  Precisa de suposições de justiça para que a propriedade seja válida.

LTL Fairness Constraints

Considere que ϕ e ψ sejam fórmulas em lógica proposicional sobre *Prop*. Pense em ϕ como uma afirmação de que "algo está habilitado" e em ψ como uma afirmação de que "algo é tomado".

Tipos **Suposições de justiça do LTL:**

- suposições de justiça incondicional: fórmula de caminho da forma

$$\mathbf{G}\,\mathbf{F}\,\psi$$

- strong fairness constraint: fórmula de caminho da forma

$$\mathbf{G}\,\mathbf{F}\,\phi \rightarrow \mathbf{G}\,\mathbf{F}\,\psi$$

- weak fairness constraint: fórmula de caminho da forma

$$\mathbf{F}\,\mathbf{G}\,\phi \rightarrow \mathbf{G}\,\mathbf{F}\,\psi$$

Exercício

Assuma as seguintes proposições atômicas:
start, *ready*, *requested*, *acknowledged*, *enabled*, *running*, *deadlock*.

Especifique as seguintes propriedades de corretude em LTL:

1. É impossível/possível alcançar um estado onde *start* seja verdadeiro mas *ready* não seja.
2. Sempre que ocorre um pedido, o mesmo eventualmente será reconhecido.
3. Aconteça o que acontecer, *deadlock* eventualmente ocorrerá.
4. A partir de qualquer estado, é possível alcançar um estado *ready*.

Solução do Exercício

Assuma as seguintes proposições atômicas:
start, *ready*, *requested*, *acknowledged*, *enabled*, *running*, *deadlock*.

Especifique as seguintes propriedades de corretude em LTL:

1. É impossível/possível alcançar um estado onde *start* seja verdadeiro mas *ready* não seja.
   $$\mathbf{A}\ \mathbf{G}\, \neg(start \wedge \neg ready)$$
2. Sempre que ocorre um pedido, o mesmo eventualmente será reconhecido.
   $$\mathbf{A}\ \mathbf{G}\,(requested \rightarrow \mathbf{F}\, acknowledged)$$
3. Aconteça o que acontecer, *deadlock* eventualmente ocorrerá.
   $$\mathbf{A}\ \mathbf{F}\, deadlock$$
4. A partir de qualquer estado, é possível alcançar um estado *ready*.
   Não é possível expressar esta propriedade - não se pode afirmar a existência de caminhos!

Verificando Fórmulas LTL com o ESBMC

Especifique a seguinte propriedade em LTL para um computador de bicicleta: G (cycle_distance_m $>=$0)

1. Converta a fórmula LTL para C da seguinte forma: $./ltl2c\ \ -f\ '!G(\{cycle\_distance\_m >= 0\})'\ -O\ c\ >$ *bicyle_ltl_formula.c*
   Note: a variável *cycle_distance_m* está declarada como uma variável global em *bicycle.c*
2. Integre a propriedade LTL em C (*bicycle_ltl_formula.c*) no código fonte origial (*bicycle.c*)
3. No arquivo integrado (*bicycle_merged.c*), chame a função *ltl2ba_start_monitor()* antes de chamar *pthread_create*. Chame também *ltl2ba_finish_monitor(pthread_id)* depois das chamadas de função do *pthread_join*

Verificando Fórmulas LTL com o ESBMC (Cont.)

Como último passo, execute o ESBMC da seguinte forma:
esbmc bicycle_merged.c –no-unwinding-assertions
–no-bounds-check –no-pointer-check –no-div-by-zero-check
–unwind 2 –context-switch 2