Universidade do Minho

Time-Critical Reactive systems

Mestrado Integrado em Engenharia Informática

Software Architecture and Calculi

2ºSemestre 2020/21

Aluno:
Etienne Costa

Docente:
José Proença

7 de abril de 2021

Conteúdo

1 Timed automata

Definition : $< \mathrm{L}\ ,\ \mathrm{L}_0, Act, C, Tr, Inv >$

Where:

- L is a set of **locations**,and $L_0 \subseteq$ L the set of initial locations .
- Act is a set of **action** and C a set of **clocks** .
- Tr $\subseteq L \times \zeta(C) \times Act \times$ P(C) $\times L$ is the **transition relation** .

$\zeta(C)$: Denotes the set of clock constraints over a set C of clock variables.

1.1 The Lamp Interrupt

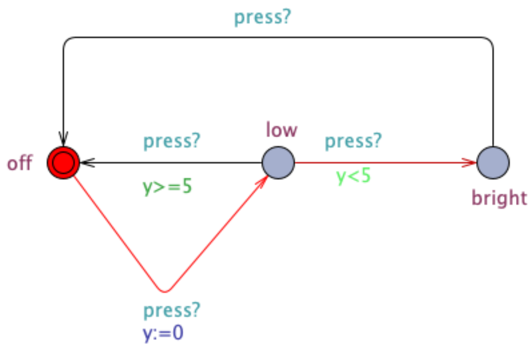

Figura 1: The Lamp Interrupt.

Ex. 1 : Define $<$ L,L$_0, Act, C, Tr, Inv >$.

```

L = { off,low, bright}

L_{0} = {off}

Act = {press}

C = {y}

Tr = { (off,true,press,{y},low),(low,y≥5,press,{},off),(low,y<5,press
    ,{},bright),(bright,true,press,{},off) }

Inv = { off → true, low → true, bright → true }
```

2 Parallel compositions of timed automata

Definition : $\text{ta}_1 \parallel_H ta_2$

Let H $\subseteq Act_1 \cap Act_2$. The parallel composition of ta_1 and ta_2 synchronizing on H is the timed automata

$\text{ta}_1 \parallel_H ta_2 := < L_1 \times L_2, L_{0,1} \times L_{0,2}, Act_{\parallel_H}, C_1 \cup C_2, Tr_{\parallel_H}, Inv_{\parallel_H} >$

Ex. 2: Define the TA of the composition.

```
===== ta_{LAMP}||_{H}ta_{USER} =====

L1 ={ off,low, bright}  L2 ={idle}

L1 × L2 ={(off,idle),(low,idle),(bright,idle)}

L_{0,1} ={off} L_{0,2} ={idle}

L_{0,1} X L_{0,2} = {(off,idle)}

H ⊆ Act_1 ∩ Act_2 = { press }

Act ||_H = {τ_press}

C1 ={y} C2 ={}

C1 ∪ C2 ={y}

Tr||_H= {
((off,idle),true,τ_press,{y},(low,idle)),
((low,idle),y≥5,τ_press,{},(off,idle)),
((low,idle),y<5,τ_press,{},(bright,idle)),
((bright,idle),true,τ_press,{},(off,idle))
}

Inv||_H = {
Inv||_H<off,idle>,
Inv||_H<low,idle>,
Inv||_H<bright,idle>
}

Inv||_H<off,idle> = off → true ∧ idle → true
Inv||_H<low,idle> = low → true ∧ idle → true
Inv||_H<bright,idle> = bright → true ∧ idle → true
```

Ex. 3: Define the TA of the composition.

```
===== ta_Worker ||_H ta_Hammer =====

L1 ={rest,work}  L2 ={free,busy}

L1 × L2 ={
(rest,free),
(rest,busy),
(work,free),
(work,busy)
}

L_0,1 ={rest} L_0,2 ={free}

L_0,1 X L_0,2 = {(rest,free)}

H ⊆ Act_1 ∩ Act_2 = { done,go }

Act ||_H = {hit,τ_done,τ_go}

C1 ={z} C2 ={x,y}

C1 ∪ C2 ={x,y,z}

Inv||_H = {
Inv||_H<rest,free>,
Inv||_H<rest,busy>,
Inv||_H<work,free>,
Inv||_H<work,busy>
}

Inv||_H<rest,free> = rest → true ∧ free → true
Inv||_H<rest,busy> = rest → true ∧ busy → true
Inv||_H<work,free> = work → z≤60 ∧ free → true
Inv||_H<work,busy> = work → z≤60 ∧ busy → true

Tr||_H= {
((rest,free),true,τ_go,{x,y,z},(work,busy)),
((work,busy),y≥5∧z≥10,τ_done,{},(rest,free)),
((work,busy),x≥1,hit,{x},(work,busy)),
((rest,busy),x≥1,hit,{x},(rest,busy))

}
```

```
===== $ta_{Worker-Hammer}\|_H ta_{Nail}$ =====

L1 ={(rest,free),(rest,busy),(work,free),(work,busy)}  L2 ={up,half,done}

L1 × L2 ={
((rest,free),up),
((rest,free),half),
((rest,free),done),
((rest,busy),up),
((rest,busy),half),
((rest,busy),done),
((work,free),up),
((work,free),half),
((work,free),done),
((work,busy),up),
((work,busy),half),
((work,busy),done)
}

$L_{0,1}$ ={(rest,free)} $L_{0,2}$ ={up}

$L_{0,1}$ X $L_{0,2}$ = {((rest,free),up)}

Act_{1} = {hit,$\tau_{go}$,$\tau_{done}$} Act_{2} = {hit,noname}

H ⊆ $Act_1$ ∩ $Act_2$ = { hit }

$Act \|_H$ = {$\tau_{hit}$,$\tau_{done}$,$\tau_{go}$,noname}

C1 ={x,y,z} C2 ={}

C1 ∪ C2 ={x,y,z}

Inv$\|_H$ = {
Inv$\|_H$<(rest,free),up>,
Inv$\|_H$<(rest,free),half>,
Inv$\|_H$<(rest,free),done>,
Inv$\|_H$<(rest,busy),up>,
Inv$\|_H$<(rest,busy),half>,
Inv$\|_H$<(rest,busy),done>,
Inv$\|_H$<(work,free),up>,
Inv$\|_H$<(work,free),half>,
Inv$\|_H$<(work,free),done>,
Inv$\|_H$<(work,busy),up>,
Inv$\|_H$<(work,busy),half>,
Inv$\|_H$<(work,busy),done>
}

Inv$\|_H$<(rest,free),up> = rest → true ∧ free → true ∧ up → true
Inv$\|_H$<(rest,free),half>= rest → true ∧ free → true ∧ half → true
```

```
Inv‖_H <(rest,free),done>= rest → true ∧ free → true ∧
    done → true
Inv‖_H <(rest,busy),up>= rest → true ∧ busy → true ∧ up →
     true
Inv‖_H <(rest,busy),half>= rest → true ∧ busy → true ∧
    half → true
Inv‖_H <(rest,busy),done>= rest → true ∧ busy → true ∧
    done → true
Inv‖_H <(work,free),up>= work → z≤60 ∧ free → true ∧ up →
     true
Inv‖_H <(work,free),half>= work → z≤60 ∧ free → true ∧
    half → true
Inv‖_H <(work,free),done>= work → z≤60 ∧ free → true ∧
    done → true
Inv‖_H <(work,busy),up>= work → z≤60 ∧ busy → true ∧ up →
     true
Inv‖_H <(work,busy),half>= work → z≤60 ∧ busy → true ∧
    half → true
Inv‖_H <(work,busy),done>= work → z≤60 ∧ busy → true ∧
    done → true

Tr‖_H= {
(((res,free),up),true,τ_go,{x,y,z},((work,busy),up)),
(((res,free),half),true,τ_go,{x,y,z},((work,busy),half)),
((res,free),done),true,τ_go,{x,y,z},((work,busy),done)),
(((res,free),done),true,noname,{},((rest,free),up)),
(((res,busy),up),x≥1,τ_hit,{x},((rest,busy),half)),
(((res,busy),half),x≥1,τ_hit,{x},((rest,busy),done)),
(((res,busy),done),true,noname,{},((rest,busy),up)),
(((work,busy),up),y≥5 ∧ z≥10,τ_done,{},((rest,free),up)),
(((work,busy),half),y≥5 ∧ z≥10,τ_done,{},((rest,free),half)
    ),
(((work,busy),done),y≥5 ∧ z≥10,τ_done,{},((rest,free),done)
    ),
(((work,busy),done),true,noname,{},((work,busy),up)),
(((work,busy),up),x≥1,τ_hit,{x},((work,busy),half)),
(((work,busy),half),x≥1,τ_hit,{x},((work,busy),done)),
(((work,busy),done),true,noname,{},((work,busy),up))

}
```

3 Timed Labelled Transition Systems

Definition :

Let ta = $<$ L , $L_0, Act, C, Tr, Inv >$

T(ta) = <S,$S_0 \subseteq$ S , N, T >

Ex. 4 : Define T(SwitchA) .

```
L = {off,on}

L_0 = {off}

Act = {out,in}

C = {x}

Inv ={
    off → true,
    on → x≤ 2
   }

Tr = {
    (off,true,in,{x},on),
    (on,x≥1,out,{},off)
}

S = {< off,t > |t ∈ R_0^+} ∪ {< on,t > |0 ≤ t ≤ 2}

S_0= {<off,0}

L_0 = {off}

N = {in,out} ∪ {R_0^+}

T = {
<off,t> --d--> <off,t+d> for all t,d ≥ 0,
<off,t> --in--> <on,0> for all t ≥ 0 ,
<on,t> --d--> <on,t+d> for all t,d ≥ 0 and t + d ≤ 2,
<on,t> --out--> <off,t> for all 1 ≤ t ≤ 2

}
```

4 Traces

Ex. 5: Write 3 possible trace with different nr. of actions.

```
                 === Different nr. of actions ===

Traces = {T1,T2,T3}
T1 = {}
T2 = <0,press>,<3,press>
T3 = <0,press>,<4,press>,<1,press>
```

Ex. 6: Are the timed-language equivalent? Explain.

```
                 === Timed-Language Equivalent ===

TracesW = { {}, {<1,t>}, {<1,t>,<d,t>} } for all 1 ≤ d ≤ 2

TracesZ = { {}, {<1,t>}, {<1,t>,<d,t>} } for all 1 ≤ d ≤ 2

The set of finite timed traces of W and Z coincide ⟹ They are timed-
    language equivalent .
```

5 Timed-Bisimulation

```
R = {(s,v),(t,v)} , é uma simulação .

Rconverso = {(v,s),(v,t)}, é uma simulação .

Podemos concluir que R é uma bisimulação .

S = { <<s,d>,<v,d>> |  d }

```

$$S = \{<< v, d >, < s, d >> |d \in \mathbb{R}_0^+\} \wedge d \leq 10\} \cup$$
$$\{<< v, d >, < t, d >> |d \in \mathbb{R}_0^+\} \wedge d \leq 10\}$$

```
Visto que <<v,0>,<s,0>> ∈ S, podemos concluir que são bisimilares .
```