Redes de Petri Coloridas (Coloured Petri Nets - CPN)

Prof. Eduardo Tavares
Prof. Paulo Maciel
Centro de Informática (UFPE)

Disciplina: Modelos para Sistemas Comunicantes

Abstração

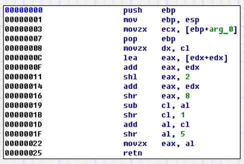
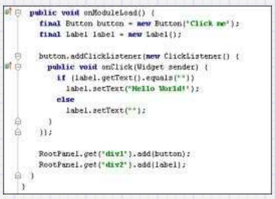
Foco naquilo que é essencial

Detalhes não proeminentes no contexto: descartados

Níveis de Abstração

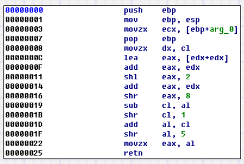
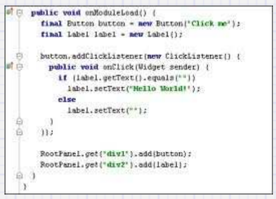
- Nível 1
- Nível 2
- ...
- Nível 3

Linguagens de Programação

Qual é a mais poderosa?

Linguagens de Programação

Qual é a mais poderosa?
R: Resolvem a mesma classe de problemas

Redes Petri

Apropriado para modelagem de sistemas com :
- Concorrência
- Comunicação
- Compartilhamento de recursos
- ...

Place/Transition nets (PT-nets)
- Componentes: Lugares, Transições, Arcos, Tokens (marcação)
- Ausência de tipos de dados e mecanismos de modularidade

Redes de Petri Coloridas (CPN)

Redes de Petri + Linguagem de Programação

Redes de Petri com:
- Tipos de Dados
- Modularidade

High-level Petri Net e Hierarchical Petri Net

Modelos sucintos e estruturados para determinadas situações

Token “carrega” um dado de um determinado tipo

Redes de Petri Coloridas (CPN)

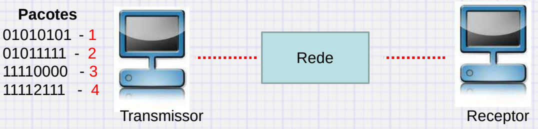
Coloured por razões históricas:
- *Colour set* = Tipo
- *Token Colour* = *Token Value*

Possuem mesmo poder de expressividade de PT-nets
- Uma pode ser traduzida na outra
- Sem ganho teórico

Adoção da linguagem funcional ML (para a CPN considerada)

Técnicas de análise: grafo de alcançabilidade, invariantes, etc.

Exemplo



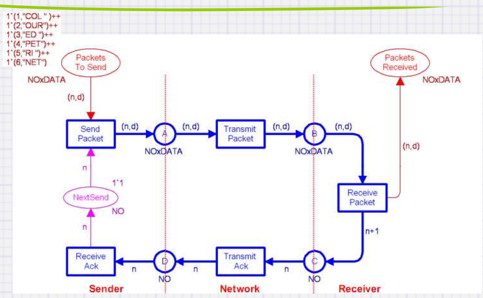
Receptor precisa juntar os dados recebidos

Protocolo *stop-and-wait* (i) transmite um pacote por vez; (ii) Aguarda reconhecimento de recebimento

Assumir uma rede segura

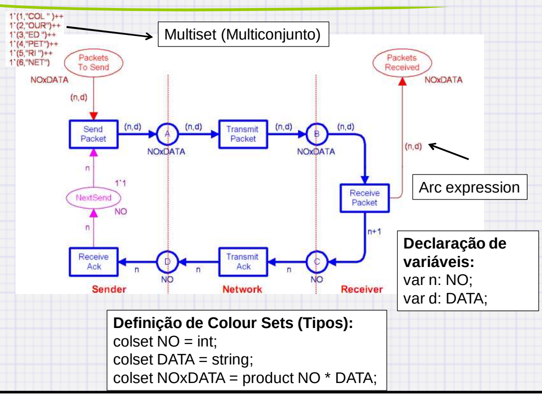
Baseado em ACPN-2010 por Lars Kristensen

Exemplo

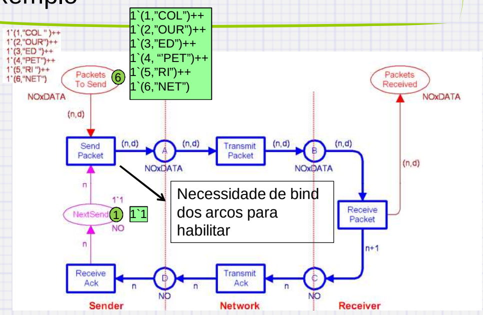


Baseado em ACPN-2010 por Lars Kristensen

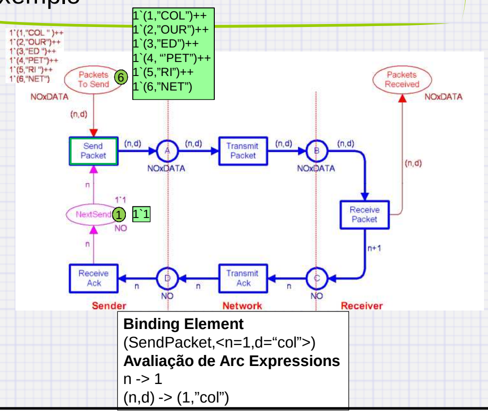
Exemplo

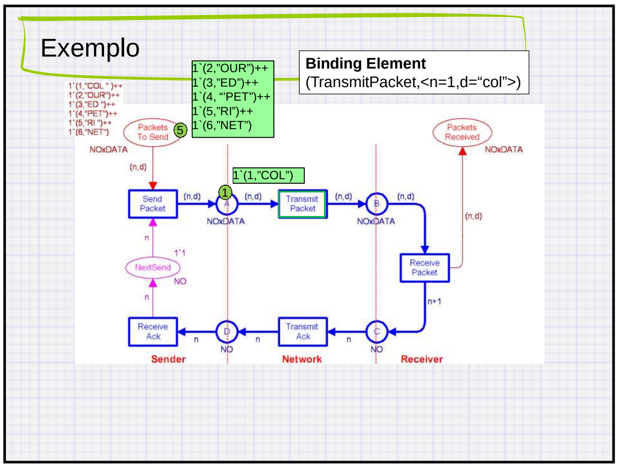


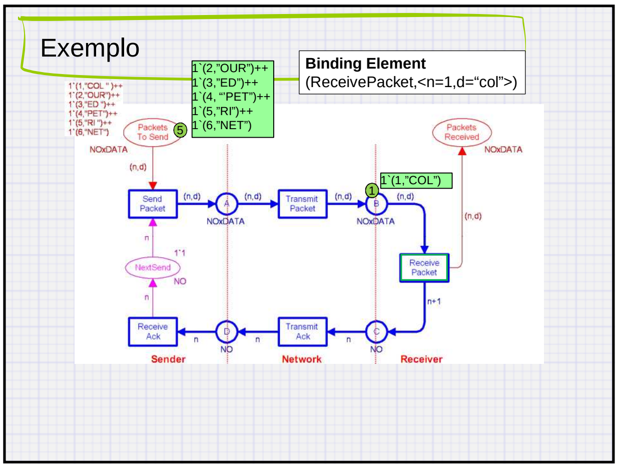
Exemplo

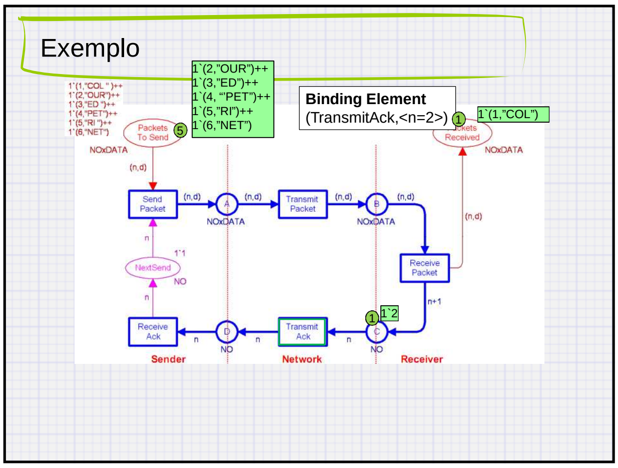


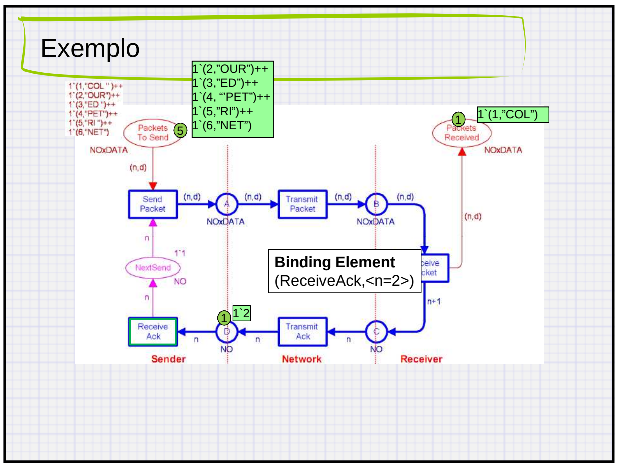
Exemplo



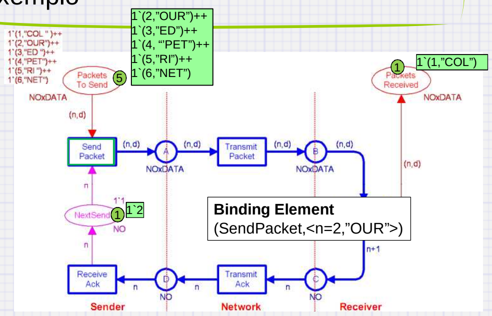

Exemplo
Binding Element
(TransmitPacket,<n=1,d="col">)
1`(2,"OUR")++
1`(3,"ED")++
1`(4, "PET")++
1`(5,"RI")++
1`(6,"NET")
1`(1,"COL")
Packets To Send
Packets Received
NOxDATA
Send Packet
Transmit Packet
Receive Packet
NextSend
Receive Ack
Transmit Ack
Sender
Network
Receiver


Exemplo
Binding Element
(ReceivePacket,<n=1,d="col">)
1`(2,"OUR")++
1`(3,"ED")++
1`(4, "PET")++
1`(5,"RI")++
1`(6,"NET")
1`(1,"COL")
Packets To Send
Packets Received
NOxDATA
Send Packet
Transmit Packet
Receive Packet
NextSend
Receive Ack
Transmit Ack
Sender
Network
Receiver

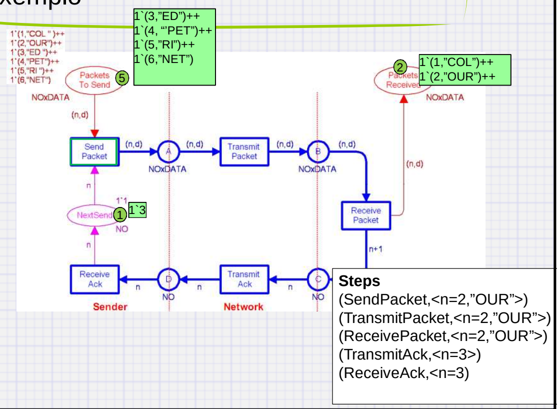

Exemplo
Binding Element
(TransmitAck,<n=2>)
1`(2,"OUR")++
1`(3,"ED")++
1`(4, "PET")++
1`(5,"RI")++
1`(6,"NET")
1`(1,"COL")
1`2
Packets To Send
NOxDATA
Send Packet
Transmit Packet
Receive Packet
NextSend
Receive Ack
Transmit Ack
Sender
Network
Receiver


Exemplo
Binding Element
(ReceiveAck,<n=2>)
1`(2,"OUR")++
1`(3,"ED")++
1`(4, "PET")++
1`(5,"RI")++
1`(6,"NET")
1`(1,"COL")
1`2
Packets To Send
NOxDATA
Send Packet
Transmit Packet
NextSend
Receive Ack
Transmit Ack
Sender
Network
Receiver

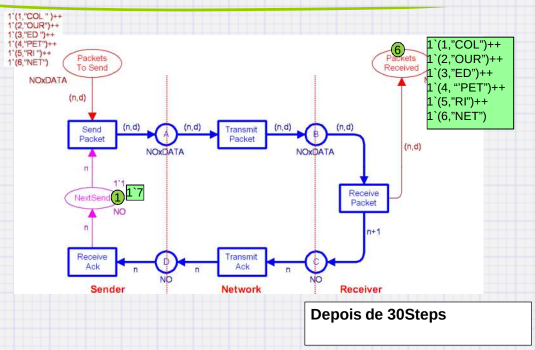
Exemplo



Exemplo



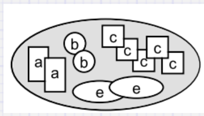
Exemplo



Multiset (Multiconjunto)

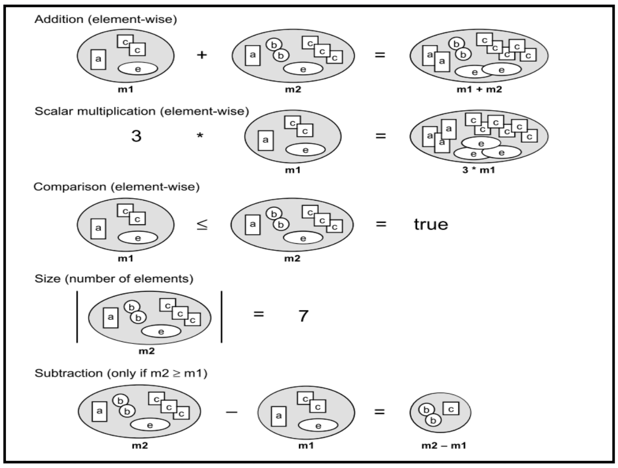
Assumindo S = {s1,s2,s3,...}, um multiset *m* é uma função m:S→ℵ . Pode ser representado como:

$^{++}\sum_{s\in S} m(s)`s = m(s1)`s1 \ ++\ m(s2)`s2 ++ m(s3)`s3 ++ ...$



Ex: S={a,b,c,e}

$^{++}\sum_{s\in S} m(s)`s = 2`a \ ++\ 2`b ++ 5`c ++ 2`e$ m(s) =
- 2 if s = a
- 2 if s = b
- 5 if s = c
- 2 if s = e
- 0 caso contrário



CPN – Não hierárquica

Definition 4.2. A **non-hierarchical Coloured Petri Net** is a nine-tuple $CPN = (P, T, A, \Sigma, V, C, G, E, I)$, where:

1. P is a finite set of **places**.
2. T is a finite set of **transitions** T such that $P \cap T = \emptyset$.
3. $A \subseteq P \times T \cup T \times P$ is a set of directed **arcs**.
4. Σ is a finite set of non-empty **colour sets**.
5. V is a finite set of **typed variables** such that $Type[v] \in \Sigma$ for all variables $v \in V$.
6. $C : P \rightarrow \Sigma$ is a **colour set function** that assigns a colour set to each place.
7. $G : T \rightarrow EXPR_V$ is a **guard function** that assigns a guard to each transition t such that $Type[G(t)] = Bool$.
8. $E : A \rightarrow EXPR_V$ is an **arc expression function** that assigns an arc expression to each arc a such that $Type[E(a)] = C(p)_{MS}$, where p is the place connected to the arc a.
9. $I : P \rightarrow EXPR_\emptyset$ is an **initialisation function** that assigns an initialisation expression to each place p such that $Type[I(p)] = C(p)_{MS}$.

CPN – Não hierárquica

Definition 4.3. For a Coloured Petri Net $CPN = (P, T, A, \Sigma, V, C, G, E, I)$, we define the following concepts:

1. A **marking** is a function M that maps each place $p \in P$ into a multiset of tokens $M(p) \in C(p)_{MS}$.
2. The **initial marking** M_0 is defined by $M_0(p) = I(p)\langle\rangle$ for all $p \in P$.
3. The **variables of a transition** t are denoted $Var(t) \subseteq V$ and consist of the free variables appearing in the guard of t and in the arc expressions of arcs connected to t.
4. A **binding** of a transition t is a function b that maps each variable $v \in Var(t)$ into a value $b(v) \in Type[v]$. The set of all bindings for a transition t is denoted $B(t)$.
5. A **binding element** is a pair (t, b) such that $t \in T$ and $b \in B(t)$. The set of all binding elements $BE(t)$ for a transition t is defined by $BE(t) = \{(t, b) \mid b \in B(t)\}$. The set of all binding elements in a CPN model is denoted BE.
6. A **step** $Y \in BE_{MS}$ is a non-empty, finite multiset of binding elements.

CPN – Não hierárquica

Definition 4.4. A binding element $(t, b) \in BE$ is **enabled** in a marking M if and only if the following two properties are satisfied:

1. $G(t)\langle b\rangle$.
2. $\forall p \in P : E(p,t)\langle b\rangle \ll\!= M(p)$.

When (t, b) is enabled in M, it may **occur**, leading to the marking M' defined by:

3. $\forall p \in P : M'(p) = (M(p) -\!\!- E(p,t)\langle b\rangle) +\!\!+ E(t,p)\langle b\rangle$.

Definition 4.5. A step $Y \in BE_{MS}$ is **enabled** in a marking M if and only if the following two properties are satisfied:

1. $\forall (t,b) \in Y : G(t)\langle b\rangle$.
2. $\forall p \in P : \ \sum^{++}_{MS\,(t,b)\in Y} E(p,t)\langle b\rangle \ll\!= M(p)$.

When Y is enabled in M, it may **occur**, leading to the marking M' defined by:

3. $\forall p \in P : M'(p) = (M(p) -\!\!- \sum^{++}_{MS\,(t,b)\in Y} E(p,t)\langle b\rangle) +\!\!+ \sum^{++}_{MS\,(t,b)\in Y} E(t,p)\langle b\rangle$.

CPN – Não hierárquica

Definition 4.4. A binding element $(t,b) \in BE$ is **enabled** in a marking M if and only if the following two properties are satisfied:

1. $G(t)\langle b \rangle$.
2. $\forall p \in P : E(p,t)\langle b \rangle \ll\!= M(p)$.

When (t,b) is enabled in M, it may **occur**, leading to the marking M' defined by:

3. $\forall p \in P : M'(p) = (M(p) -\!- E(p,t)\langle b \rangle) +\!+ E(t,p)\langle b \rangle$.

Definition 4.5. A step $Y \in BE_{MS}$ is **enabled** in a marking M if and only if the following two properties are satisfied:

1. $\forall (t,b) \in Y : G(t)\langle b \rangle$.
2. $\forall p \in P : \sum_{MS\,(t,b)\in Y}^{++} E(p,t)\langle b \rangle \ll\!= M(p)$.

Interleaving será assumido!

When Y is enabled in M, it may **occur**, leading to the marking M' defined by:

3. $\forall p \in P : M'(p) = (M(p) -\!- \sum_{MS\,(t,b)\in Y}^{++} E(p,t)\langle b \rangle) +\!+ \sum_{MS\,(t,b)\in Y}^{++} E(t,p)\langle b \rangle$.

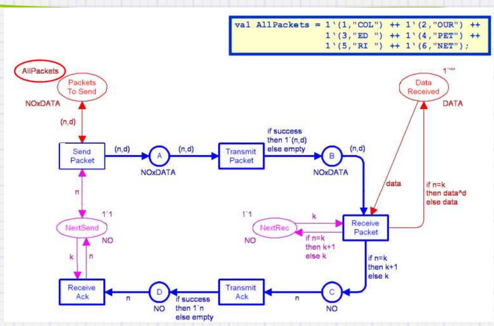
Exemplo 2

Pacotes
01010101 - 1
01011111 - 2
11110000 - 3
11112111 - 4

- Rede não confiável (ex: perdas)
- Transmissor pode retransmitir pacotes e guarda o pacote enviado correntemente
- Receptor aguarda o próximo pacote esperado

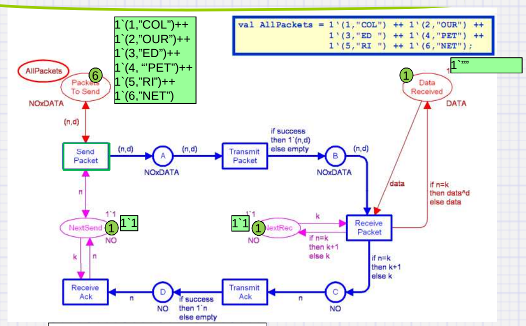
Baseado em ACPN-2010 por Lars Kristensen

Exemplo 2



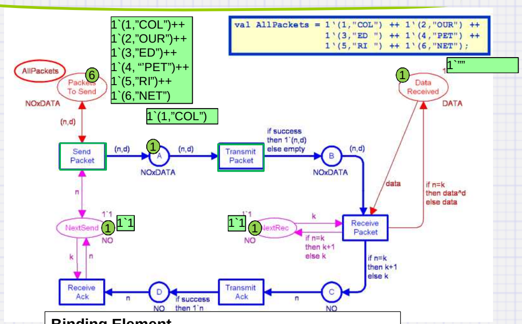
Baseado em ACPN-2010 por Lars Kristensen

Exemplo 2



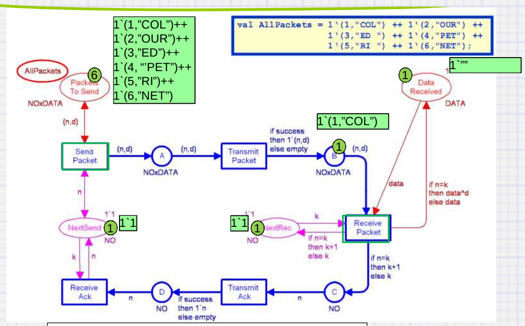
Binding Element
(SendPacket,<n=1,d="COL">)

Exemplo 2



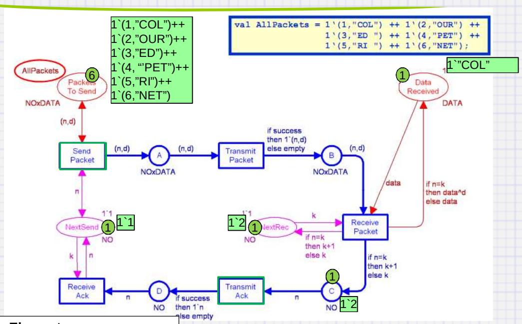
Binding Element
(SendPacket,<n=1,d="COL">)
(TransmitPackage,<n=1,d="COL">,success=true)
(TransmitPackage,<n=1,d="COL",success=false>)

Exemplo 2



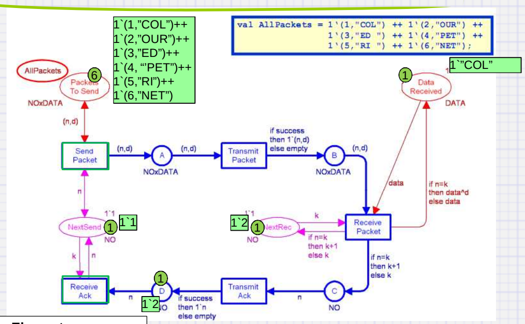
Binding Element
(SendPacket,<n=1,d="COL">)
(ReceivePacket,<n=1,d="COL",k=1,data="">)

Exemplo 2



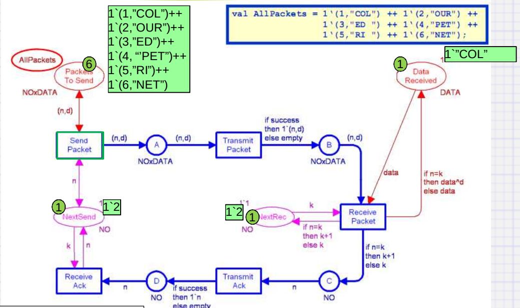
Binding Elements
(SendPacket,<n=1,d="COL">)
(TransmitAck,<n=2, success=true>
TransmitAck,<n=2, success=false>)

Exemplo 2



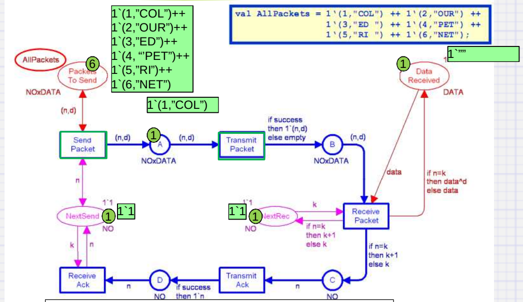
Binding Elements
(SendPacket,<n=1,d="COL">)
(ReceivePack,<n=2, k=1>)

Exemplo 2



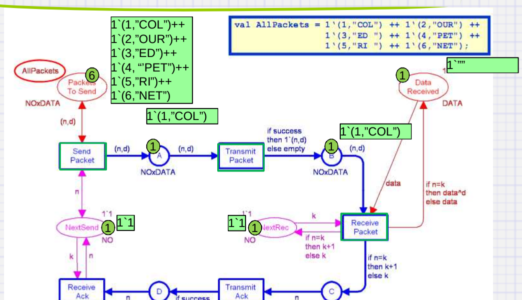
Binding Elements
(SendPacket,<n=2,d="OUR">)

Exemplo 2 – Conflito e Concorrência



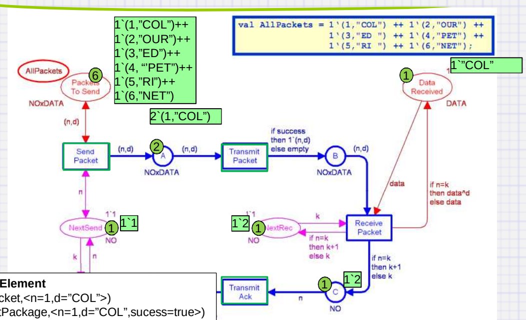
Binding Element
(SendPacket,<n=1,d="COL">)
(TransmitPackage,<n=1,d="COL">,sucess=true)
(TransmitPackage,<n=1,d="COL",sucess=false>)

Exemplo 2 - Conflito e Concorrência



Binding Element
(SendPacket,<n=1,d="COL">)
(TransmitPackage,<n=1,d="COL",sucess=true>)
(TransmitPackage,<n=1,d="COL",sucess=false>
(ReceivePacket, ,<n=1,d="COL",data="",k=1)

Exemplo 2 - Conflito e Concorrência



Binding Element
(SendPacket,<n=1,d="COL">)
(TransmitPackage,<n=1,d="COL",sucess=true>)
(TransmitPackage,<n=1,d="COL",sucess=false>
)
(TransmitAck, ,<n=2, sucess=true>)
(TransmitAck, ,<n=2, sucess=false>)

Espaço de Estados

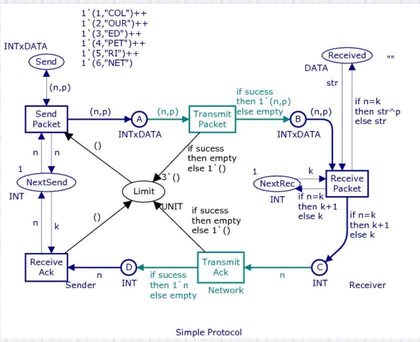
Os espaço de estados de uma CPN é um grafo dirigido SS = (N_{SS}, A_{SS}) com os arcos rotulados a partir de BE, no qual

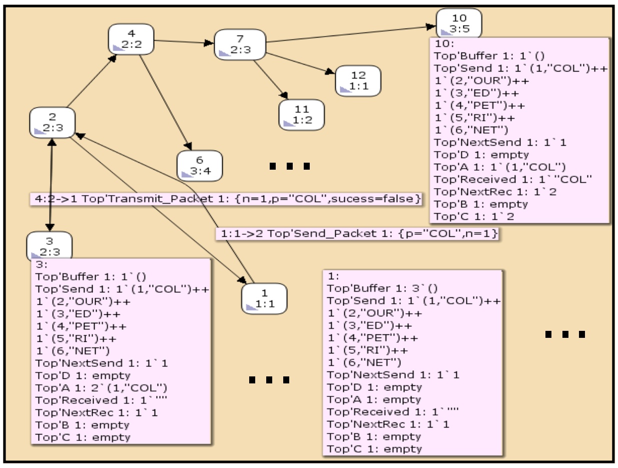
1. $N_{SS} = R(M_0)$ é o conjunto de nós
2. ASS = {(M,(t,b),M') ∈ N_{SS} X BE X N_{SS} | M →M'} é o conjunto de arcos
3. SS é finito e somente se N_{SS} e A_{SS} são finitos

Espaço de Estados

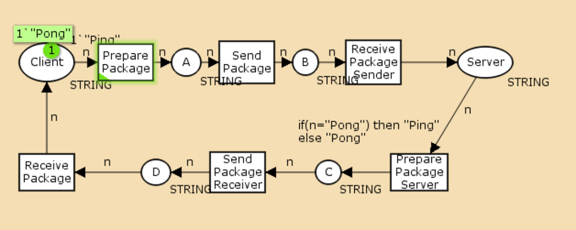
NODES $\leftarrow \{M_0\}$
UNPROCESSED $\leftarrow \{M_0\}$
ARCS $\leftarrow \emptyset$
while UNPROCESSED $\neq \emptyset$ **do**
  Select a marking M in UNPROCESSED
  UNPROCESSED $\leftarrow$ UNPROCESSED $- \{M\}$
  for all binding elements (t,b) such that $M \xrightarrow{(t,b)}$ **do**
    Calculate M' such that $M \xrightarrow{(t,b)} M'$
    ARCS $\leftarrow$ ARCS $\cup \{(M,(t,b),M')\}$
    if $M' \notin$ NODES **then**
      NODES $\leftarrow$ NODES $\cup \{M'\}$
      UNPROCESSED $\leftarrow$ UNPROCESSED $\cup \{M'\}$
    end if
  end for
end while

Espaço de Estados

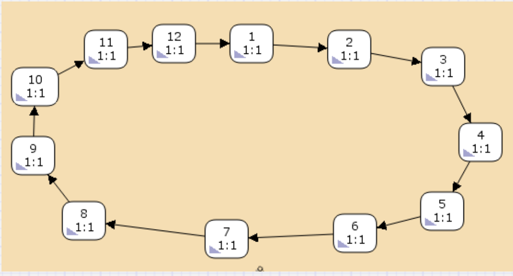




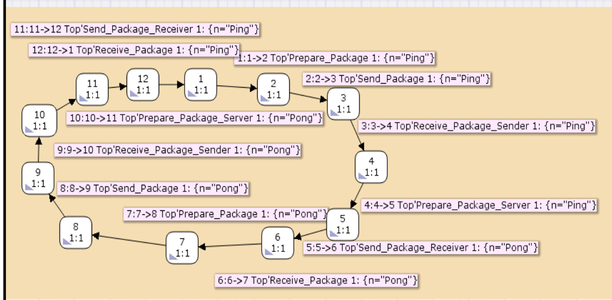
Espaço de Estados



Espaço de Estados



Espaço de Estados



Invariantes

Conceito similar em PTN, aplica-se para CPN

Definition 7.1: Let $A \in \Sigma$ be a type and let $W = \{W_p\}_{p \in P}$ be a set of linear functions such that $W_p \in [C(p)_{MS} \rightarrow A_{MS}]$ for all $p \in P$.

(i) W is a **place flow** iff:
$$\forall (t,b) \in BE: \sum_{p \in P} W_p(E(p,t)\langle b\rangle) = \sum_{p \in P} W_p(E(t,p)\langle b\rangle).$$

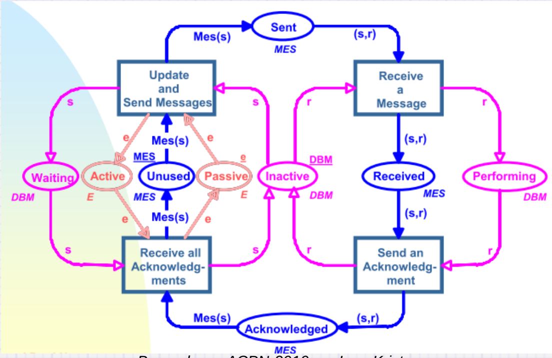
(ii) W determines a **place invariant** iff:
$$\forall M \in [M_0\rangle: \sum_{p \in P} W_p(M(p)) = \sum_{p \in P} W_p(M_0(p)).$$

Theorem 7.2: W is a place flow $\Leftrightarrow$ W determines a place invariant.

$\Rightarrow$ is satisfied for all CP-nets.
$\Leftarrow$ is only satisfied when the CP-net does not have dead binding elements.

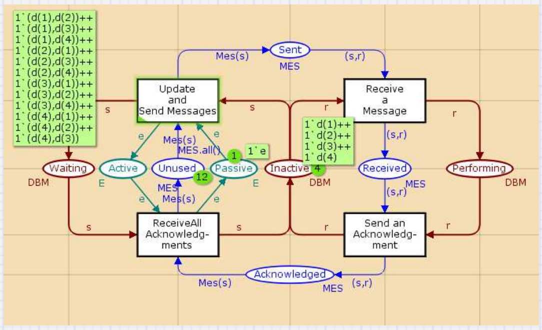
$$W * I = 0$$

Invariantes



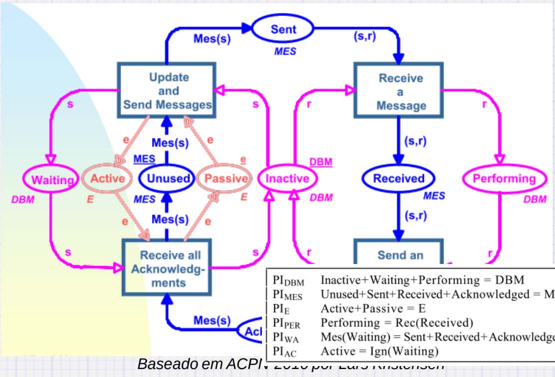
Baseado em ACPN-2010 por Lars Kristensen

Invariantes



Baseado em ACPN-2010 por Lars Kristensen

Invariantes



PI_{DBM}	Inactive+Waiting+Performing = DBM
PI_{MES}	Unused+Sent+Received+Acknowledged = MES
PI_{E}	Active+Passive = E
PI_{PER}	Performing = Rec(Received)
PI_{WA}	Mes(Waiting) = Sent+Received+Acknowledged
PI_{AC}	Active = Ign(Waiting)

Baseado em ACPN-2010 por Lars Kristensen

Invariantes

- Invariantes de transições obtidos de forma similar
- Pesos associados a transições
- Transition flow: conjunto de ocorrências que não tem efeito total (marcação inicial e final são as mesmas)
- A ferramenta mais representativa não tem suporte a invariantes

Redes de Petri Coloridas Hierárquicas (CPN)

Boas práticas na construção de programas: Módulos
- Reusabilidade
- Mais abstração
- Legibilidade
- Diminiuição de erros
- ...

Construção de uma CPN através de submodelos

Submodelos podem ser parametrizados e podem referenciar outros submodelos

Adoção de uma transição que representa a chamada ao submodelo

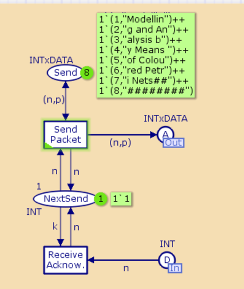
Redes de Petri Coloridas Hierárquicas (CPN)

Submodelo possui interface via lugares
- Lugares de entrada (IN)
- Lugares de saída (OUT)
- Lugares de entrada/saída (I/O)

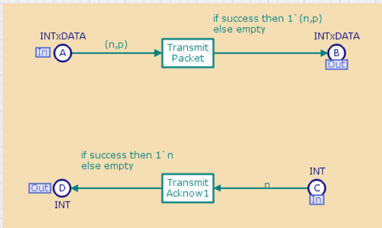
Ex: CPN com a rede não confiável

Considerar 3 submodelos:
- Transmissor
- Rede
- Receptor

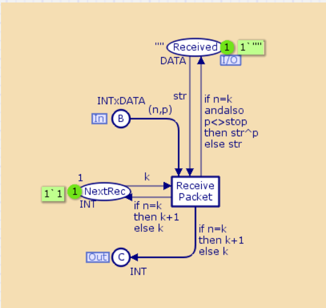
Redes de Petri Coloridas Hierárquicas (CPN)



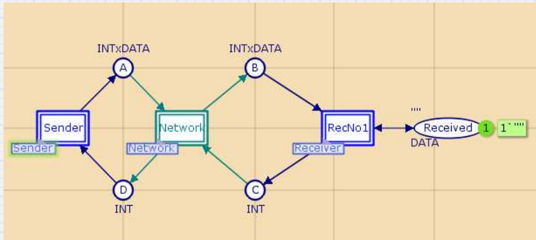
Redes de Petri Coloridas Hierárquicas (CPN)



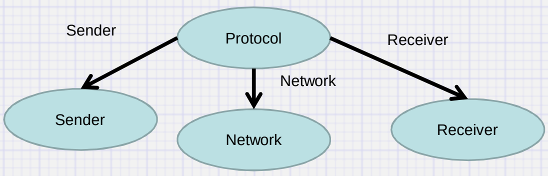
Redes de Petri Coloridas Hierárquicas (CPN)



Redes de Petri Coloridas Hierárquicas (CPN)



Redes de Petri Coloridas Hierárquicas (CPN)



Redes de Petri Coloridas Hierárquicas (CPN)

Fusion sets

- Submodelos compartilhando um mesmo lugar
- O lugar é o mesmo para as instâncias
- Similar ao conceito de variáveis globais
- Adoção precisa ser cuidadosa

Redes de Petri Coloridas Hierárquicas (CPN)

Definition 6.1. A **Coloured Petri Net Module** is a four-tuple $CPN_M = (CPN, T_{\mathsf{sub}}, P_{\mathsf{port}}, PT)$, where:

1. $CPN = (P, T, A, \Sigma, V, C, G, E, I)$ is a **non-hierarchical Coloured Petri Net**.
2. $T_{\mathsf{sub}} \subseteq T$ is a set of **substitution transitions**.
3. $P_{\mathsf{port}} \subseteq P$ is a set of **port places**.
4. $PT : P_{\mathsf{port}} \rightarrow \{\mathsf{IN}, \mathsf{OUT}, \mathsf{I/O}\}$ is a **port type function** that assigns a port type to each port place.

□

Redes de Petri Coloridas Hierárquicas (CPN)

Definition 6.2. A **hierarchical Coloured Petri Net** is a four-tuple $CPN_H = (S, SM, PS, FS)$ where:

1. S is a finite set of **modules**. Each module is a **Coloured Petri Net Module** $s = ((P^s, T^s, A^s, \Sigma^s, V^s, C^s, G^s, E^s, I^s), T^s_{\mathsf{sub}}, P^s_{\mathsf{port}}, PT^s)$. It is required that $(P^{s_1} \cup T^{s_1}) \cap (P^{s_2} \cup T^{s_2}) = \emptyset$ for all $s_1, s_2 \in S$ such that $s_1 \neq s_2$.
2. $SM : T_{\mathsf{sub}} \rightarrow S$ is a **submodule function** that assigns a **submodule** to each substitution transition. It is required that the module hierarchy (see Definition 6.3) is acyclic.
3. PS is a **port–socket relation function** that assigns a **port–socket relation** $PS(t) \subseteq P_{\mathsf{sock}}(t) \times P^{SM(t)}_{\mathsf{port}}$ to each substitution transition t. It is required that $ST(p) = PT(p')$, $C(p) = C(p')$, and $I(p)\langle\rangle = I(p')\langle\rangle$ for all $(p, p') \in PS(t)$ and all $t \in T_{\mathsf{sub}}$.
4. $FS \subseteq 2^P$ is a set of non-empty **fusion sets** such that $C(p) = C(p')$ and $I(p)\langle\rangle = I(p')\langle\rangle$ for all $p, p' \in fs$ and all $fs \in FS$.

□

socket places $P_{\mathsf{sock}}(t)$ of a substitution transition $PT : P_{\mathsf{port}} \rightarrow \{\mathsf{IN}, \mathsf{OUT}, \mathsf{I/O}\}$ is a **port type function**
socket type function $ST(t)$ that maps each socket place of t into its type

Redes de Petri Coloridas Hierárquicas (CPN)

Uma CPN hierárquica pode ser sempre transformada em um CPN não hierárquica

Na teoria, não adiciona maior poder de expressividade ao modelo

CPN hierárquica e não hierárquica são equivalentes

Da mesma forma CPN ↔ PTN

- Lugares em CPN substituído por Lugares em PTN com a quantidade de cores no colour set
- Transições em PTN com as possibilidades de bindings satisfazendo as guardas das transições em CPN

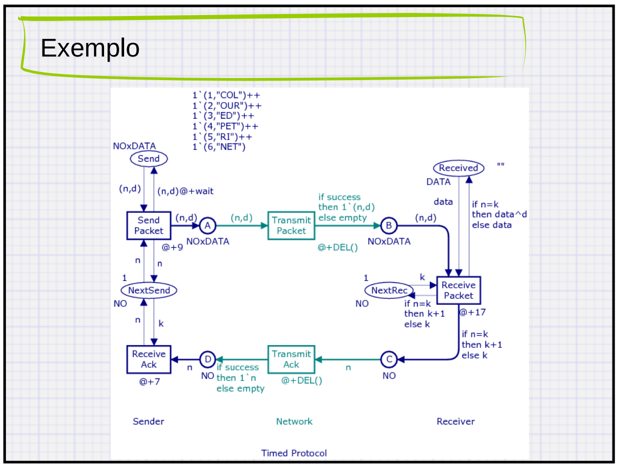
Redes de Petri Coloridas Temporizadas (TCPN)

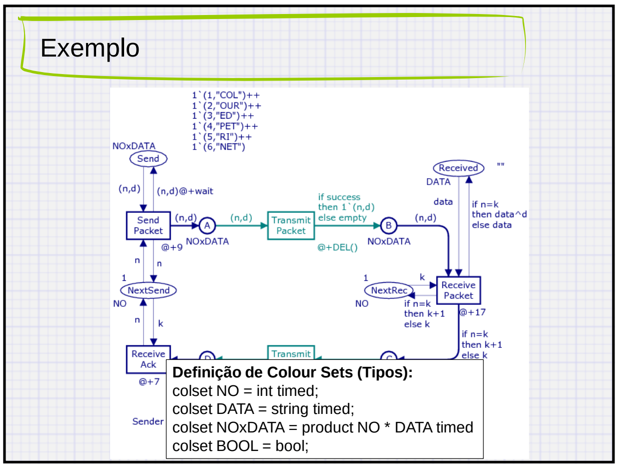
Importância do tempo para avaliação de desempenho/dependabilidade de sistemas

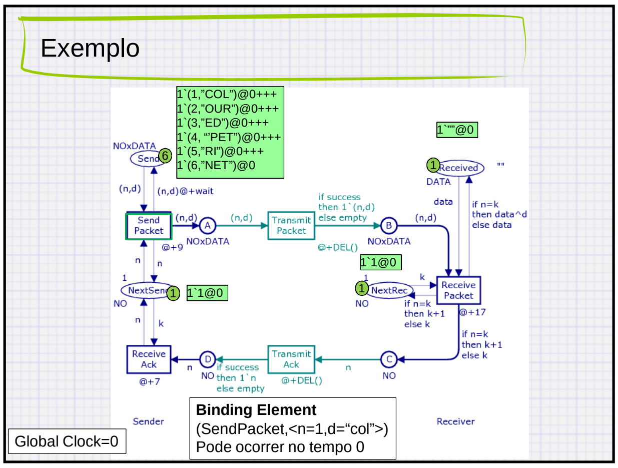
Modelagem de sistemas de tempo real

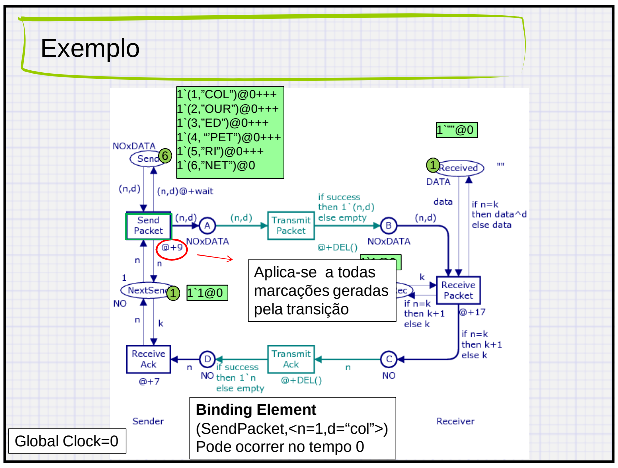
Conceitos

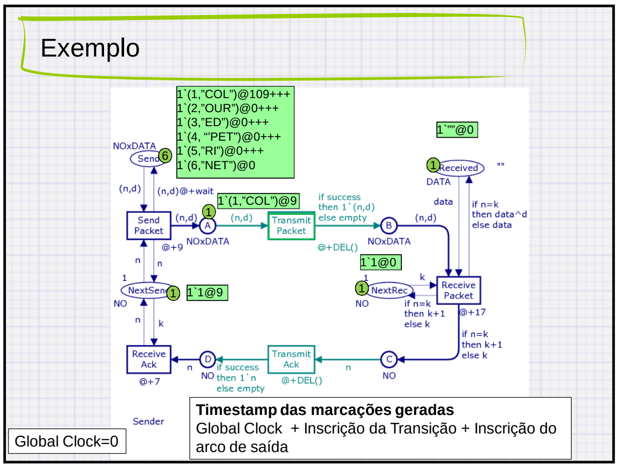
- Tempos associados às marcações(tokens)
- Adoção de um relógio global (global clock)
- O *timestamp* associado à marcação indica quando a mesma está pronta para uso
- O disparo de uma transição é imediato
- *Timed multiset* e *timed colour sets*
- *Timestamp* ∈ TIME (conjunto dos inteiros não negativos)

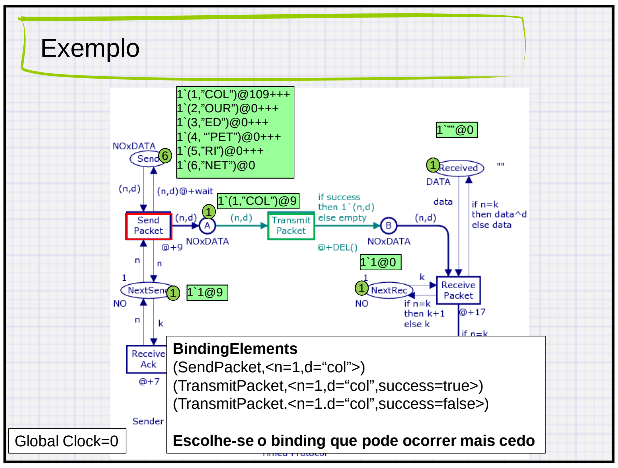

Exemplo
1`(1,"COL")++
1`(2,"OUR")++
1`(3,"ED")++
1`(4,"PET")++
1`(5,"RI")++
1`(6,"NET")
NOxDATA
Send
Received
DATA
Send Packet
Transmit Packet
Receive Packet
NextSend
NextRec
Receive Ack
Transmit Ack
if success then 1`(n,d) else empty
if n=k then data^d else data
if n=k then k+1 else k
if success then 1`n else empty
@+9
@+DEL()
@+17
@+7
Sender
Network
Receiver
Timed Protocol

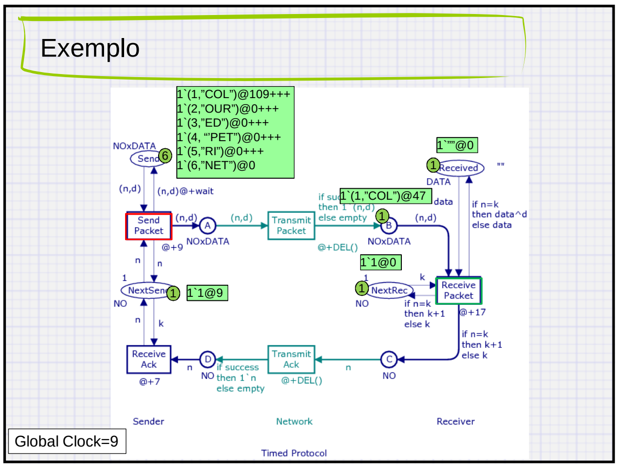

Exemplo
Definição de Colour Sets (Tipos):
colset NO = int timed;
colset DATA = string timed;
colset NOxDATA = product NO * DATA timed
colset BOOL = bool;

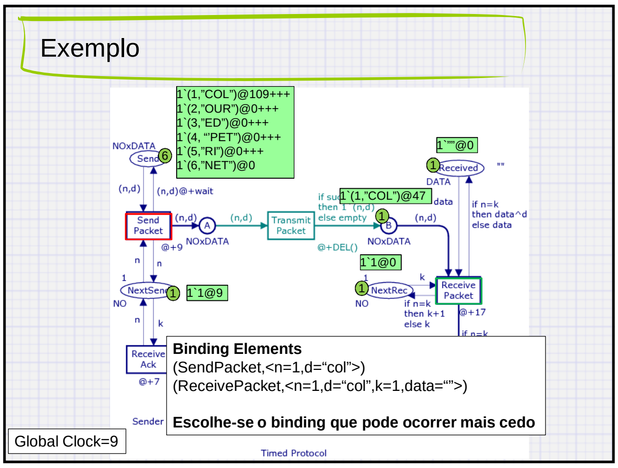

Exemplo
1`(1,"COL")@0+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4, "PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`""@0
1`1@0
Global Clock=0
Binding Element
(SendPacket,<n=1,d="col">)
Pode ocorrer no tempo 0

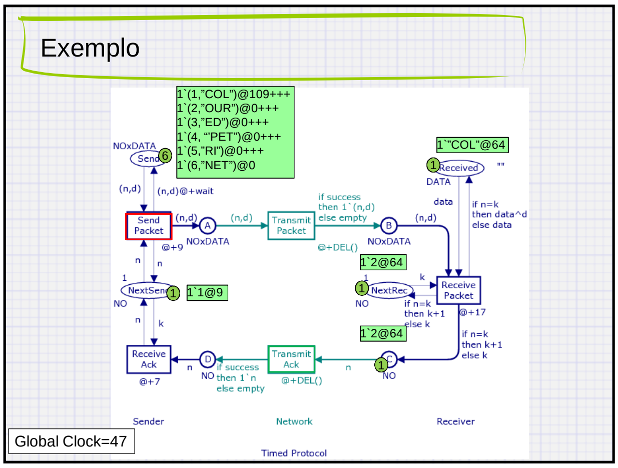

Exemplo
1`(1,"COL")@0+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4, "PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`""@0
1`1@0
Aplica-se a todas marcações geradas pela transição
Global Clock=0
Binding Element
(SendPacket,<n=1,d="col">)
Pode ocorrer no tempo 0

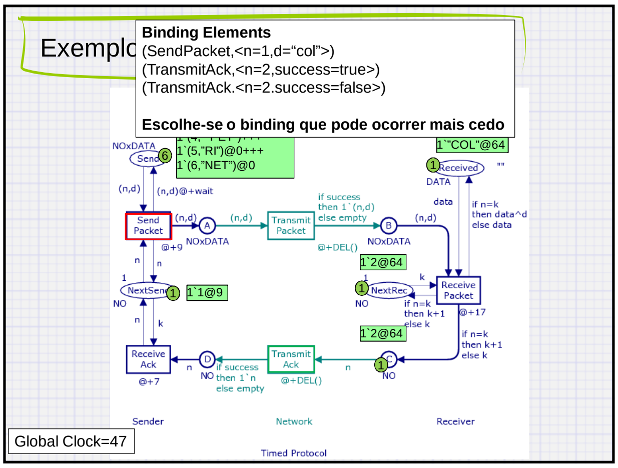

Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4,"PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`(1,"COL")@9
1`""@0
1`1@0
1`1@9
Timestamp das marcações geradas
Global Clock + Inscrição da Transição + Inscrição do arco de saída
Global Clock=0

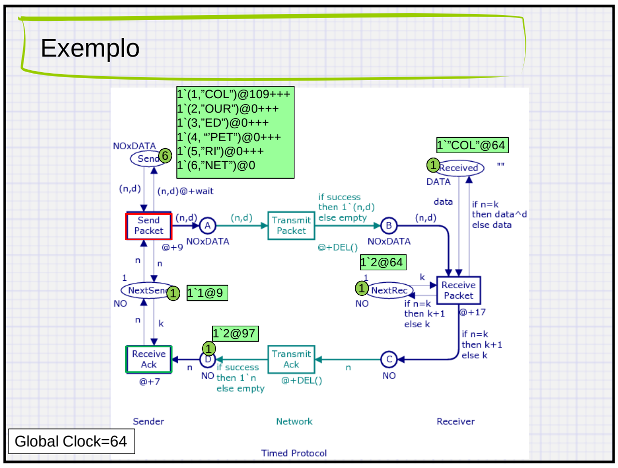

Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4,"PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`(1,"COL")@9
1`""@0
1`1@0
1`1@9
BindingElements
(SendPacket,<n=1,d="col">)
(TransmitPacket,<n=1,d="col",success=true>)
(TransmitPacket.<n=1.d="col",success=false>)
Escolhe-se o binding que pode ocorrer mais cedo
Global Clock=0

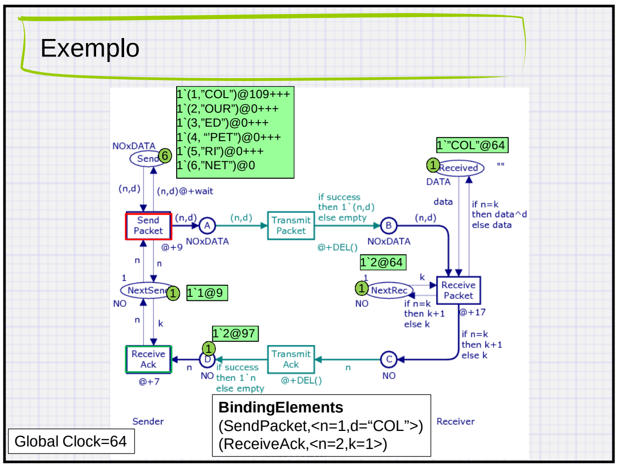

Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4,"PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`(1,"COL")@47
1`""@0
1`1@0
1`1@9
Sender
Network
Receiver
Timed Protocol
Global Clock=9


Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4,"PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`(1,"COL")@47
1`""@0
1`1@0
1`1@9
Binding Elements
(SendPacket,<n=1,d="col">)
(ReceivePacket,<n=1,d="col",k=1,data="">)
Escolhe-se o binding que pode ocorrer mais cedo
Timed Protocol
Global Clock=9

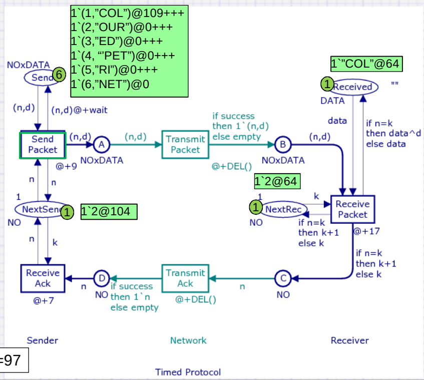

Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4, "PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`"COL"@64
1`2@64
1`1@9
1`2@64
Sender
Network
Receiver
Global Clock=47
Timed Protocol


Exemplo
Binding Elements
(SendPacket,<n=1,d="col">)
(TransmitAck,<n=2,success=true>)
(TransmitAck.<n=2.success=false>)
Escolhe-se o binding que pode ocorrer mais cedo
1`(5,"RI")@0+++
1`(6,"NET")@0
1`"COL"@64
1`2@64
1`1@9
1`2@64
Sender
Network
Receiver
Global Clock=47
Timed Protocol


Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4, "PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`"COL"@64
1`2@64
1`1@9
1`2@97
Sender
Network
Receiver
Global Clock=64
Timed Protocol

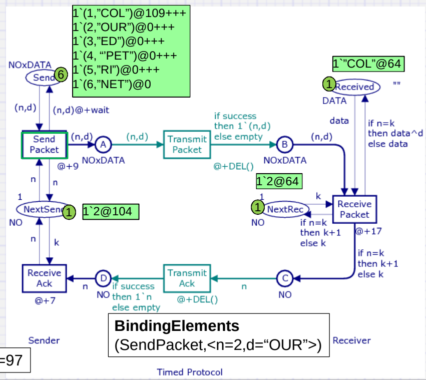

Exemplo
1`(1,"COL")@109+++
1`(2,"OUR")@0+++
1`(3,"ED")@0+++
1`(4, "PET")@0+++
1`(5,"RI")@0+++
1`(6,"NET")@0
1`"COL"@64
1`2@64
1`1@9
1`2@97
Sender
Receiver
Global Clock=64
BindingElements
(SendPacket,<n=1,d="COL">)
(ReceiveAck,<n=2,k=1>)

Exemplo



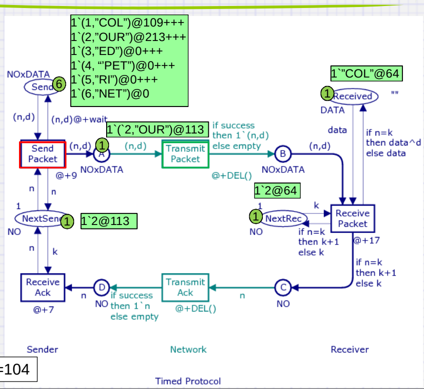
Global Clock=97

Exemplo



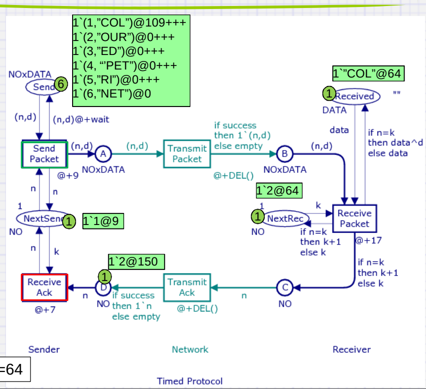
Global Clock=97

Exemplo



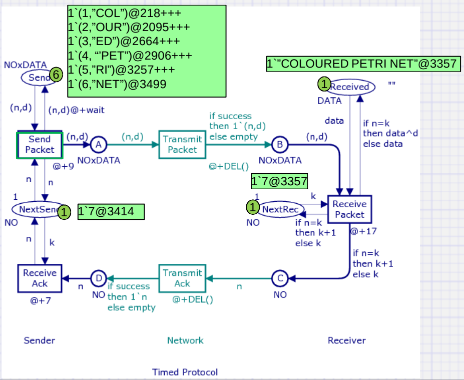
Global Clock=104

Exemplo (Retransmissão)



Global Clock=64

Exemplo (Dead Marking)



Redes de Petri Coloridas Temporizadas (TCPN)

m_d=1'2@405 ++ 2'2@409 +++ 1'2@411+++4'3@410

Assumir um binding element e global clock = 409. Irá remover um token com a cor 2

Remove-se um dos 2'2@409 (removação do maior possível)

Permite que a marcação alcançada usando semântica de passos possa ser alcançada com interleaving em um order arbitrária

Redes de Petri Coloridas Temporizadas (TCPN)

TCPN podem ter marcações com ou sem *timestamps*

Marcações sem *timestamps* sempre estão prontas para participarem de *binding elements*

Binding elements precisam ser *colour enabled* e *ready*

- *Colour enabled* = Marcação
- *Ready* = tempo (*timestamps* da marcação precisa ser menor ou igual ao valor atual do global clock)

Redes de Petri Coloridas Temporizadas (TCPN)

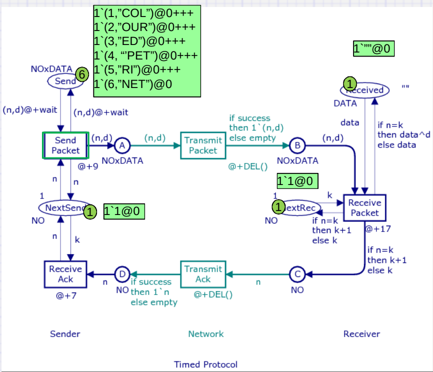
Caso não existam *binding elements* a serem executados, o simulador avança o *global clock* para o o próximo tempo mais cedo no qual *binding elements* possam executar

Cada marcação existe em um intervalo de tempo fechado que pode ser um ponto

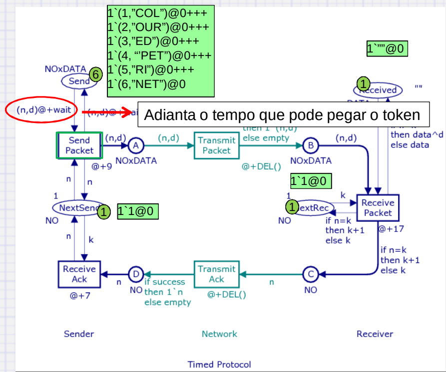
Similarmente, conflitos e concorrência podem acontecer com os *binding elements* no contexto temporizado

TCPN -> CPN (basta remover as inscrições envolvendo tempo). A sequência de ocorrências de uma TCPN é um subconjunto da CPN sem tempo. Tempo adiciona novas restrições

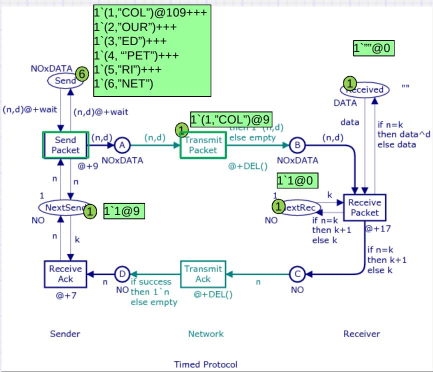
Exemplo



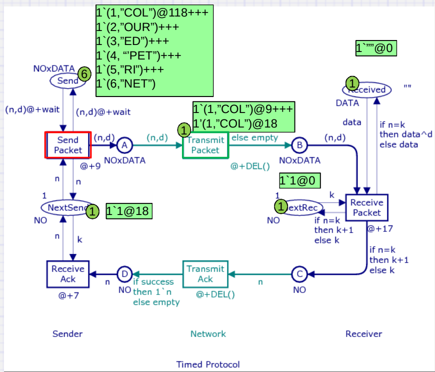
Exemplo



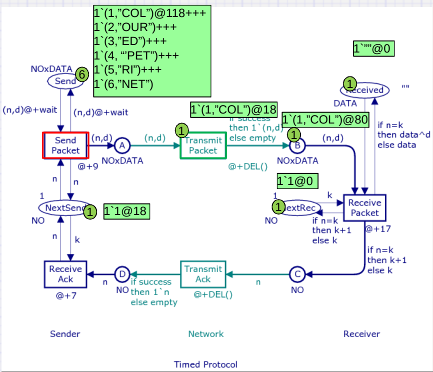
Exemplo



Exemplo



Exemplo



Espaço de estados de TCPN

Espaço de estado similar, acrescentando:

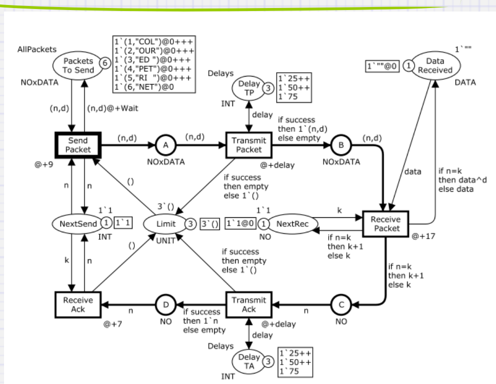
- *timestamps* dos tokens
- O valor do global clock

Duas marcações temporizadas podem ser diferentes mesmo que a correspondente marcação sem tempo sejam iguais

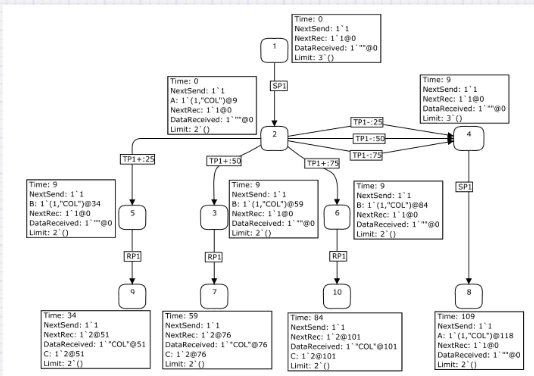
O espaço de estados temporizado pode ser infinito, mesmo que o correspondente espaço de estados sem tempo seja finito

O espaço de estados pode ser maior que o espaço sem tempo

Espaço de estados de TCPN



Espaço de estados de TCPN



Espaço de estados de TCPN

Adoção de *time equivalence method* na CPN tools

Um estado sem o valor absoluto de tempo pode ser adotado pra representar outros estados equivalentes

Construção de um *condensed state space.*

Condensed state space é finito, caso o espaço de estados da CPN sem tempo também seja finito.

Todas a propriedades verificada no espaço de estado completo também são verificadas no *condensed state space*

Timed Multiset (Multiconjunto Temporizado)

Definition 11.1. Let S be a non-empty set and let $\mathbb{T}$ be the set of time values. A **timed multiset** over S is a function $tm : S \times \mathbb{T} \rightarrow \mathbb{N}$ that maps each element $(s,t) \in S \times \mathbb{T}$ into a non-negative integer $tm(s,t) \in \mathbb{N}$. It is required that the sum

$$tm(s) = \sum_{t \in \mathbb{T}} tm(s,t)$$

is finite for all $s \in S$. The non-negative integer $tm(s)$ is the **number of appearances** (or coefficient) of s in tm. The **timestamp list** for an element s is a list

$$tm[s] = [t_1, t_2, \ldots, t_{tm(s)}]$$

satisfying $t_i \leq t_{i+1}$ for all $1 \leq i < tm(s)$. It contains the time values t for which $tm(s,t) > 0$ and a time value t appears $tm(s,t)$ times in the list. **Membership** and **size** are defined as follows, where tm is a timed multiset:

1. $\forall s \in S : s \in tm \Leftrightarrow tm(s) > 0$.
2. $|tm| = \sum_{s \in S} tm(s)$.

A timed multiset tm is **infinite** if $|tm| = \infty$. Otherwise tm is **finite**. The set of all timed multisets over S is denoted S_{TMS}. The empty multiset over S is denoted $\emptyset_{TMS}$ and is defined by $\emptyset_{TMS}(s,t) = 0$ for all $(s,t) \in S \times \mathbb{T}$.

□

Timed Multiset (Multiconjunto Temporizado)

tm_B=1`(1,"COUL")@2030 +++ 1`(2,"OUR")@2015 +++ 2`(2,"OUR")@`2005 +++ 1`(2,"OUR")@1994

tm_B(s,t)=
- 1 if(s,t) = ((1,"COL"),2030)
- 1 if(s,t) = ((2,"OUR"),2015)
- 2 if(s,t) = ((2,"OUR"),2005)
- 1 if(s,t) = ((2,"OUR"),1994)
- 0 caso contrário

tm_B(2,"OUR")=4

tm_B[(2,"OUR")] = [1994,2005,2005,2015]

Formalização TCPN

Definition 11.2. For timed multisets over a set S and time values $\mathbb{T}$, **comparison**, **subtraction**, and **addition of time** on timestamp lists are defined as follows, where $tm[s] = [t_1, t_2, \ldots, t_{tm(s)}]$, $tm_1[s] = [t^1_1, t^1_2, \ldots, t^1_{tm_1(s)}]$, and $tm_2[s] = [t^2_1, t^2_2, \ldots, t^2_{tm_2(s)}]$ are timestamp lists of an element $s \in S$:

1. $tm_1[s] \leq_{[\mathbb{T}]} tm_2[s] \Leftrightarrow tm_1(s) \leq tm_2(s)$ and $t^1_i \geq t^2_i$ for all $1 \leq i \leq tm_1(s)$
2. For $t \in \mathbb{T}$ such that $t \geq t_1$, $tm[s] -_{\mathbb{T}} t$ is the timestamp list
   - $tm[s] -_{\mathbb{T}} t = [t_1, t_2, t_3, \ldots, t_{i-1}, t_{i+1}, \ldots, t_{tm(s)}]$ where i is the largest index for which $t_i \leq t$.
3. When $tm_1[s] \leq_{[\mathbb{T}]} tm_2[s]$, $tm_2[s] -_{[\mathbb{T}]} tm_1[s]$ is the timestamp list defined by
   - $tm_2[s] -_{[\mathbb{T}]} tm_1[s] = ((([t^2_1, t^2_2, \ldots, t^2_{tm_2(s)}] -_{\mathbb{T}} t^1_1) -_{\mathbb{T}} t^1_2) \cdots -_{\mathbb{T}} t^1_{tm_1(s)})$.
4. For $t \in \mathbb{T}$, $tm[s]_{+t}$ is the timestamp list defined by
   - $tm[s]_{+t} = [t_1 + t, t_2 + t, \ldots, t_{tm(s)} + t]$

□

Formalização TCPN

Definition 11.3. For timed multisets over a set S and time values $\mathbb{T}$, **addition**, **comparison**, **subtraction**, and **addition of time** are defined as follows, where tm, tm_1, and tm_2 are timed multisets:

1. $\forall (s,t) \in S \times \mathbb{T} : (tm_1 +\!\!+\!\!+ \, tm_2)(s,t) = tm_1(s,t) + tm_2(s,t)$.
2. $tm_1 \lll= tm_2 \Leftrightarrow \forall s \in S : tm_1[s] \leq_{[\mathbb{T}]} tm_2[s]$.
3. When $tm_1 \lll= tm_2$, $tm_2 --- tm_1$ is the timed multiset defined by
   - $\forall s \in S : (tm_2 --- tm_1)(s) = tm_2(s) - tm_1(s)$;
   - $\forall s \in S : (tm_2 --- tm_1)[s] = tm_2[s] -_{[\mathbb{T}]} tm_1[s]$.
4. For $t \in \mathbb{T}$, tm_{+t} is the timed multiset defined by
   - $\forall s \in S : tm_{+t}(s) = tm(s)$ and $tm_{+t}[s] = tm[s]_{+t}$.

□

Timed Multiset (Multiconjunto Temporizado)

tm_{RP}=1\`(2,"OUR")@2010

tm_B=1\`(1,"COUL")@2030 +++ 1\`(2,"OUR")@2015 +++
2\`(2,"OUR")@\`2005 +++ 1\`(2,"OUR")@1994

tm_{RP}((2,"OUR"))=1

tm_B((2,"OUR"))=4

tm_{RP}[(2,"OUR")] = [2010]

tm_B[(2,"OUR")] = [1994,2005,2005,2015]

(tm_B--- tm_{RP})(s)=
- 1 if(s,t) = (1,"COL")
- 3 if(s,t) = (2,"OUR")
- 0 caso contrário

(tm_B--- tm_{RP})[(2,"OUR")]=[1994,2005,2015]

tm_B--- tm_{RP}=1\`(1,"OUR")@2030 +++ 1\`(2,"OUR")@2015 +++
1\`(2,"OUR")@2005 +++ 1\`(2,"OUR")@1994

Formalização TCPN

Definition 11.4. A **timed non-hierarchical Coloured Petri Net** is a nine-tuple $CPN_T = (P,T,A,\Sigma,V,C,G,E,I)$ where:

1. P is a finite set of **places**.
2. T is a finite set of **transitions** such that $P \cap T = \emptyset$.
3. $A \subseteq P \times T \cup T \times P$ is a set of directed **arcs**.
4. Σ is a finite set of non-empty **colour sets**. Each colour set is either untimed or timed.
5. V is a finite set of **typed variables** such that $Type[v] \in \Sigma$ for all variables $v \in V$.
6. $C : P \to \Sigma$ is a **colour set function** that assigns a colour set to each place. A place p is timed if $C(p)$ is timed, otherwise p is untimed.
7. $G : T \to EXPR_V$ is a **guard function** that assigns a guard to each transition t such that $Type[G(t)] = Bool$.
8. $E : A \to EXPR_V$ is an **arc expression function** that assigns an arc expression to each arc a such that
   - $Type[E(a)] = C(p)_{MS}$ if p is untimed;
   - $Type[E(a)] = C(p)_{TMS}$ if p is timed.

   Here, p is the place connected to the arc a.
9. $I : P \to EXPR_\emptyset$ is an **initialisation function** that assigns an initialisation expression to each place p such that
   - $Type[I(p)] = C(p)_{MS}$ if p is untimed;
   - $Type[I(p)] = C(p)_{TMS}$ if p is timed.

□

Formalização TCPN

Definition 11.5. For a timed Coloured Petri Net $CPN_T = (P,T,A,\Sigma,V,C,G,E,I)$, we define the following concepts:

1. A **marking** is a function M that maps each place $p \in P$ into a multiset $M(p)$ of tokens such that
   - $M(p) \in C(p)_{MS}$ if p is untimed;
   - $M(p) \in C(p)_{TMS}$ if p is timed.
2. A **timed marking** is a pair (M,t^*), where M is a marking and $t^* \in \mathbb{T}$ is the value of the global clock.
3. The **initial timed marking** is the pair $(M_0,0)$, where M_0 is defined by $M_0(p) = I(p)\langle\rangle$ for all $p \in P$.

Formalização TCPN

Definition 11.6. A step $Y \in BE_{MS}$ is **enabled** at time t' in a timed marking (M, t^*) if and only if the following properties are satisfied:

1. $\forall (t,b) \in Y : G(t)\langle b \rangle$.
2. $\sum\limits^{++}_{MS\,(t,b)\in Y} E(p,t)\langle b \rangle \ll= M(p)$ for all untimed places $p \in P$.
3. $\sum\limits^{+++}_{MS\,(t,b)\in Y} (E(p,t)\langle b \rangle)_{+t'} \lll= M(p)$ for all timed places $p \in P$.
4. $t^* \leq t'$.
5. t' is the smallest time value for which there exists a step satisfying conditions 1–4.

When Y is enabled in (M, t^*) at time t', it may **occur** at time t', leading to the timed marking (M', t') defined by:

6. $M'(p) = (M(p) -\!- \sum\limits^{++}_{MS\,(t,b)\in Y} E(p,t)\langle b \rangle) +\!+ \sum\limits^{++}_{MS\,(t,b)\in Y} E(t,p)\langle b \rangle$ for all untimed places $p \in P$.
7. $M'(p) = (M(p) -\!-\!- \sum\limits^{+++}_{MS\,(t,b)\in Y} E(p,t)\langle b \rangle_{+t'}) +\!+\!+ \sum\limits^{+++}_{MS\,(t,b)\in Y} E(t,p)\langle b \rangle_{+t'}$ for all timed places $p \in P$.

□