

CTL for Petri nets

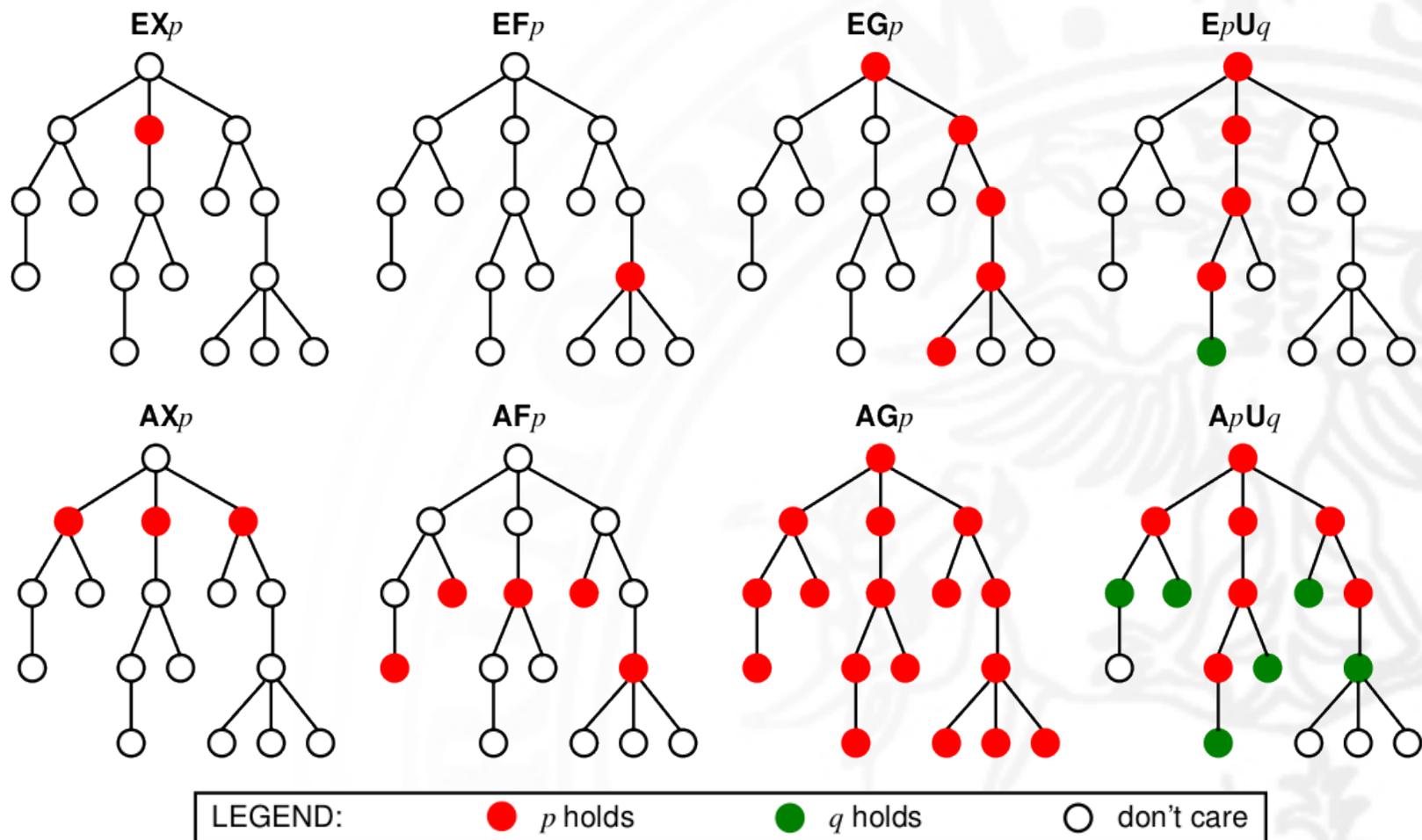
using GreatSPN

CTL su Kripke vs Petri net

- Abbiamo visto come CTL è definito sulle strutture di Kripke (stati, transizioni, etichette sugli stati).
- Il reachability graph di una Petri net è simile, ma più ricco:
 - Ogni stato è una marcatura (variabili intere sui posti) e non da etichette booleane
 - Le transizioni hanno un nome.

CTL per Petri net

- Formule CTL:
 - **state formula**: assumono valore booleane per ciascuna marcatura dell'RG
 - $p, q, p \ \&\& \ q, \ !q$
 - Le formule atomiche (p, q) sono esprimibili come espressioni sui token dei posti, o come condizioni di abilitazione sulle transizioni.
 - **path formula**: assumono valore booleano per ciascuna marcatura presa come radice di un cammino.
 - $EX \ p, \ E \ p \ U \ q, \ AG \ q$



EX, *EU*, and *EG* form a complete set of CTL operators, since:

- $AXp = \neg EX\neg p$ • $AFp = \neg EG\neg p$ • $EFp = EtrueUp$
- $AGp = \neg EF\neg p$ • $ApUq = \neg(E\neg qU(\neg p \wedge \neg q)) \wedge \neg EG\neg q$

Sat-set di CTL

- La valutazione di CTL assegna un valore di verità ad ogni stato del RS, che indica se la formula è soddisfatta in quello stato.
- L'insieme di stati che soddisfano una formula Φ si chiama $\text{Sat}(\Phi)$, o anche il Sat-set di Φ .
- Una formula CTL Φ è soddisfatta nella marcatura iniziale, sse: $m_0 \in \text{Sat}(\Phi)$.

CTL formula grammar

$\langle CTLformula \rangle$	$::=$	$\langle atomicProposition \rangle \mid \langle CTLformula \rangle \mid$ $\langle CTLformula \rangle \text{ " \&\& " } \langle CTLformula \rangle \mid$ $\langle CTLformula \rangle \text{ " \ \ " } \langle CTLformula \rangle \mid$ $\text{ " ! " } \langle CTLformula \rangle \mid \langle CTLformula \rangle \text{ " -> " } \langle CTLformula \rangle \mid$ $\text{ " E " " X " } \langle CTLformula \rangle \mid \text{ " E " " G " } \langle CTLformula \rangle \mid$ $\text{ " E " " [" } \langle CTLformula \rangle \text{ " U " } \langle CTLformula \rangle \text{ "] " } \mid$ $\text{ " A " " X " } \langle CTLformula \rangle \mid \text{ " A " " F " } \langle CTLformula \rangle \mid$ $\text{ " E " " F " } \langle CTLformula \rangle \mid \text{ " A " " G " } \langle CTLformula \rangle \mid$ $\text{ " A " " [" } \langle CTLformula \rangle \text{ " U " } \langle CTLformula \rangle \text{ "] " }$
$\langle atomicProposition \rangle$	$::=$	$\langle inequality \rangle \mid \langle boolvalue \rangle \mid \text{ " ndeadlock " } \mid \text{ " deadlock " } \mid \text{ " en " (trans) }$
$\langle boolvalue \rangle$	$::=$	$\text{ " true " } \mid \text{ " false " }$
$\langle inequality \rangle$	$::=$	$\text{ " (" } \langle expression \rangle \langle comp_oper \rangle \langle number_expr \rangle \text{ ") " }$
$\langle comp_oper \rangle$	$::=$	$\text{ " < " } \mid \text{ " > " } \mid \text{ " <= " } \mid \text{ " >= " } \mid \text{ " = " } \mid \text{ " ! = " }$
$\langle expression \rangle$	$::=$	$\text{ " (" } \langle expression \rangle \langle arit_oper \rangle \langle number_expr \rangle \text{ ") " } \mid \langle term \rangle \mid$ $\text{ " (" } \langle number_expr \rangle \text{ ") " }$
$\langle arit_oper \rangle$	$::=$	$\text{ " + " } \mid \text{ " - " } \mid \text{ " * " } \mid \text{ " / " }$
$\langle term \rangle$	$::=$	$\langle number_expr \rangle \text{ " * " } \langle var \rangle \mid \langle number_expr \rangle \text{ " / " } \langle var \rangle \mid \langle var \rangle$
$\langle number_expr \rangle$	$::=$	$\text{ " (" } \langle number_expr \rangle \langle arit_oper \rangle \langle number_expr \rangle \text{ ") " } \mid \langle number \rangle$
$\langle var \rangle$	$::=$	$\text{ " \# " } place_name$
$\langle numbr \rangle$	$::=$	\mathbb{R}^+

$\langle var \rangle$ è il numero di token in un posto, e si scrive come: #P0 (con il # davanti)

Avviare analisi CTL di una Petri net

Usare il comando apposito.

La schermata che si apre permette di inserire una o più query CTL.

The image shows the GreatSPN Editor interface. The top toolbar contains several icons, with the CTL model checking icon (a gear with a plus sign) highlighted by a red box. A tooltip for this icon reads: "Start CTL model checking. Build the full Reachability Graph. Reachability Set using Decision Diagrams." Below the toolbar, a blue arrow points from the tooltip to the "Add measure" dialog box. The dialog box is titled "Add measure" and contains the following fields and options:

- Target model: ReadWrite
- Solver: RGMEDD2
- Solver parameters:
 - Variable order heuristic: Meta-heuristic
 - Generate counter-examples/witnesses when possible.
- Measures:

Pos:	Measure:
1° <input type="checkbox"/> STAT	Tool statistics. Compute
2° <input type="checkbox"/> CTL	E(#P0==0 U #P1!=0) <input checked="" type="checkbox"/> = Compute

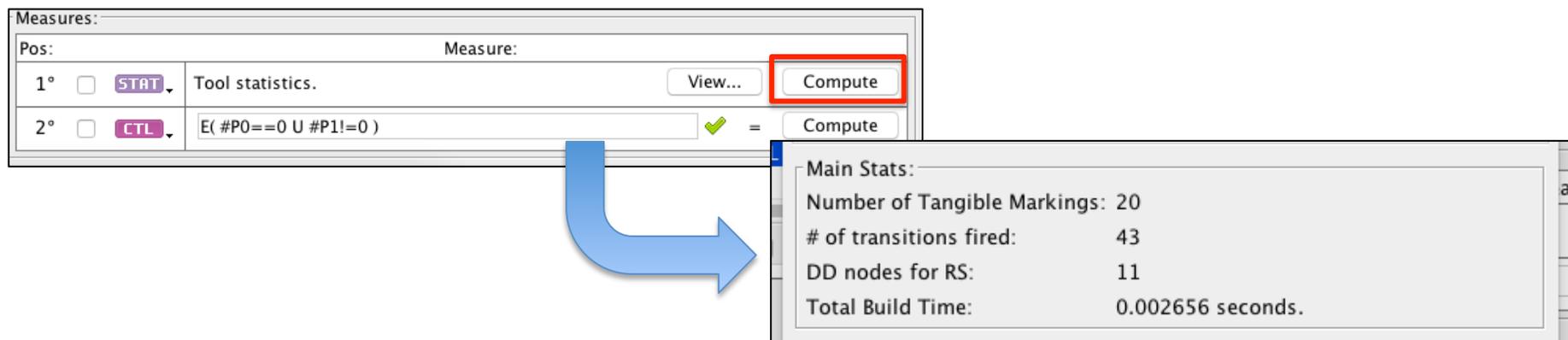
At the bottom of the dialog box, there are buttons for "View log..." and "Compute All".

Analisi CTL di Petri nets

- RGMEDD: un model checker CTL di GreatSPN.
- Funziona con reti P/T, senza colori. Per fare CTL su una rete colorata, si deve prima farne l'unfolding.

RGMEDD: un model checker CTL

- RGMEDD è il model checker CTL di GreatSPN.
- Ha 2 parametri:
 - l'algoritmo per il variable order.
 - la generazione dei contro-esempi CTL (vediamo dopo)
- Calcolare le STAT permette di vedere il numero di marcature nel RS, ed il numero di nodi dell'MDD.



The screenshot shows the 'Measures' dialog box in GreatSPN. It has two rows. The first row is for 'STAT' and the second for 'CTL'. The 'CTL' row contains the formula $E(\#P0==0 \cup \#P1!=0)$ and a green checkmark. The 'Compute' button for the second row is highlighted with a red box. A blue arrow points from this button to a 'Main Stats' window. The 'Main Stats' window displays the following data:

Main Stats:	
Number of Tangible Markings:	20
# of transitions fired:	43
DD nodes for RS:	11
Total Build Time:	0.002656 seconds.

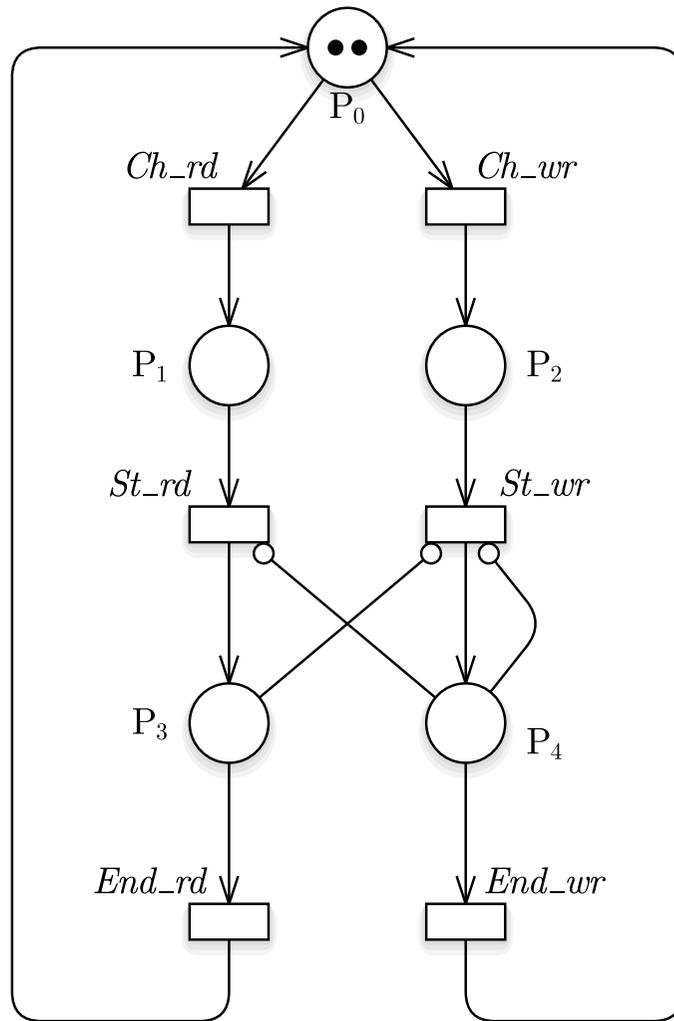
Valutazione di query CTL

- Per ogni query CTL valutata, il programma riporta il valore finale true/false, che indica se la query è soddisfatta nella marcatura iniziale.

ATTENZIONE: una query CTL potrebbe risultare falsa proprio perchè non è vera in m_0 , pur essendo vera in altre marcature.

- Il log riporta un'analisi più dettagliata della query:
 - Viene riportata l'analisi di ogni sotto-formula.
 - per ogni sotto-formula, viene riportata la cardinalità del Sat-set generato.
 - Se la formula è falsa, il sistema prova a mostrare un contro-esempio. Se la formula è vera, il sistema prova a mostrare un esempio (witness) che soddisfa la formula.

Esercizi sul modello Reader/Writer



- Formulare una query CTL per chiedere se può esistere una marcatura raggiungibile dove ci sono due lettori nella sezione critica (cioè P3 ha 2 token).

EF #P3 == 2

Interpretare l'output di RGMEDD

- EF #P3==2
- RGMEDD genera il seguente output:

```
Processing: E F (P3 = 2) -> bool
Eval: P3
    0.000103 sec.
Eval: (P3 = 2)
    0.000070 sec. card = 1
Eval: E F (P3 = 2)
    8 steps:    0.000486 sec. card = 15
--- EF #P3==2 ---
    Formula name: MEASURE0
    Evaluation: true
    Sat-set generation time: 0.000779 sec
    Evaluation time: 0.000793 sec
```

- RGMEDD genera anche un esempio di traccia che verifica la query.

Generated witness:

Initial state is: $P0(2)$

Initial state satisfies: $E \ F \ (P3 = 2)$.

1: $P0(2)$

State 1. does not satisfy: $(P3 = 2)$.

2: $P1(1), P0(1)$

State 2. does not satisfy: $(P3 = 2)$.

3: $P1(2)$

State 3. does not satisfy: $(P3 = 2)$.

4: $P3(1), P1(1)$

State 4. does not satisfy: $(P3 = 2)$.

5: $P3(2)$

State 5. satisfies: $(P3 = 2)$.

Attenzione: solo le query esistenziali possono generare esempi/controesempi.

L'esempio è una traccia che, partendo da m_0 , raggiunge uno stato che soddisfa $\#P3==2$.

Esercizio

- In tutte le esecuzioni del modello, esiste sempre un cammino dove un lettore in attesa entra nella sezione critica (cioè passa da $\#P1>0$ a $\#P3>0$).

AF E[$\#P1>0$ U $\#P3>0$]

(considerare anche il modello con fair scheduling)

Esercizio

- In ogni esecuzione del modello si può sempre trovare, ad un certo punto, uno scrittore nella sezione critica ($\#P4 > 0$).

AF $\#P4 > 0$

(considerare anche il modello con fair scheduling)

Esempi e contro-esempi

- Non è possibile generare esempi/controesempi per tutte le formule:
 - Solo per le **formule esistenziali vere** si può trovare un esempio.
 - Solo per le **formule universali false** si può trovare un controesempio.
- Esercizi:
 - AF ($\#P4 == 1 \rightarrow \exists X \#P4 == 0$): vera
 - vera in quanto tutte le tracce soddisfano la formula, una singola traccia non è significativa.
 - EF ($\#P4 > 0 \ \&\& \ \#P3 > 0$): falsa
 - falsa in quanto non esiste una traccia che soddisfa $\#P4 > 0 \ \&\& \ \#P3 > 0$ in qualche marcatura. Dunque non si può mostrare un contro-esempio.

Deadlock

- Il tool supporta due etichette speciali per gli stati:
 - deadlock: lo stato non attiva nessuna transizione
 - ndeadlock: gli stati non deadlock.
- Come si chiede se dalla marcatura iniziale si può raggiungere un deadlock?
- Come si chiede che l'RS non abbia nessun deadlock?

Qual è la differenza tra EF deadlock e AF deadlock?