« Model checking » de chaînes de Markov

Serge Haddad

LSV CNRS & ENS Cachan

9ème atelier en évaluation de performances
le 1er juin 2008 à Aussois

Plan

Introduction

Qu'est-ce que le model checking ?

- Un modèle $\mathcal{M}$ spécifié dans un formalisme
- Une propriété φ spécifiée dans un langage de formules
- Un algorithme qui vérifie si $\mathcal{M} \models \varphi$
  (et éventuellement exhibe un contre-exemple dans la négative)

Prérequis

- Syntaxe et sémantique d'un formalisme
- Syntaxe et sémantique d'un langage de formules

Un très bref rappel historique

Vérification de programmes

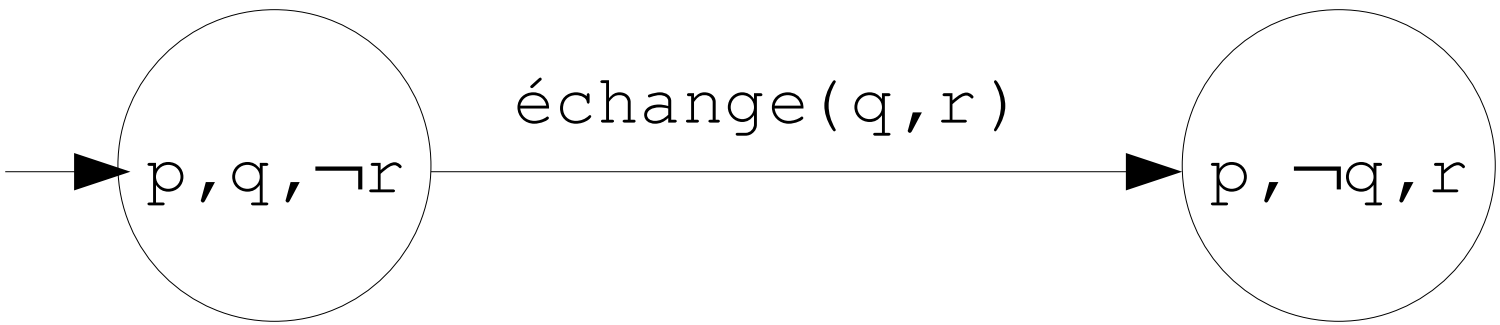
- Le formalisme est le langage de programmation ou une abstraction appropriée.
- Les propriétés à vérifier sont la correction partielle et la terminaison.
- La correction partielle est exprimée via une logique du premier ordre.

Vérification de systèmes réactifs

- De nouvelles caractéristiques :
  la concurrence, le non déterminisme, le temps, le hasard, etc.
- De nouvelles propriétés : sûreté, vivacité, équité, etc.

Les systèmes de transitions

- Un formalisme de bas niveau qui est généralement la sémantique d'un formalisme de haut niveau *(i.e. description implicite)*.
- Un graphe (éventuellement infini) dont les sommets sont les états et les arcs sont les transitions *(si fini beaucoup plus grand que la spécification de haut niveau)*.
- Dans un état, certaines propriétés atomiques sont vérifiées.
- Une transition est (éventuellement) étiquetée par l'événement qui déclenche la transition.
- Un système de transitions a un ensemble d'états initiaux.

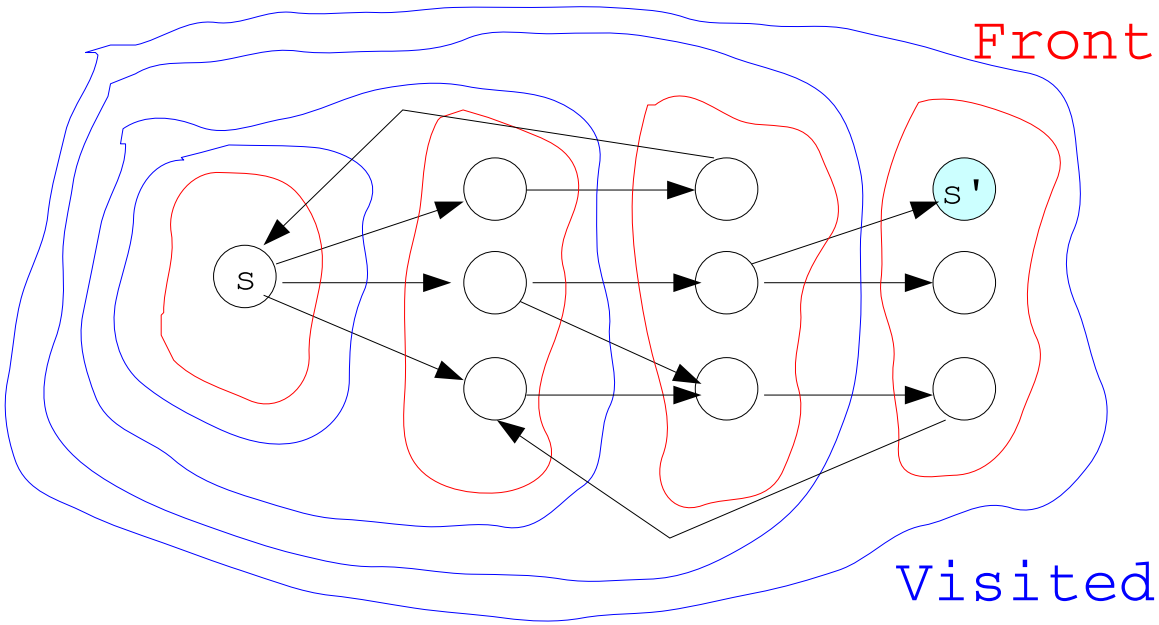


Propriétés génériques des systèmes de transitions

- L'accessibilité : L'état s' est-il accessible de l'état s ?
- Le blocage : Un état sans successeur est-il accessible de l'état s ?
- La quasi-vivacité : L'événement e peut-il apparaître à partir de l'état s ?
- La vivacité : L'événement e peut-il apparaître à partir de tout état accessible depuis l'état s ?
- L'équité (ou sa négation) : Le sous-ensemble d'événements E peut-il être indéfiniment évité à partir d'un état accessible depuis l'état s ?

Comment tester l'accessibilité (1) ?

Par saturation




```
Search()
  Front = Visited = {s} ;
  while s' ∉ Visited and Front≠ ∅ do
    Front = Successor(Front)\Visited ;
    Visited = Visited ∪ Front ;
  return(s' ∈ Visited) ;
```

Comment tester l'accessibilité (2) ?

Par une exploration limitée sans mémoire

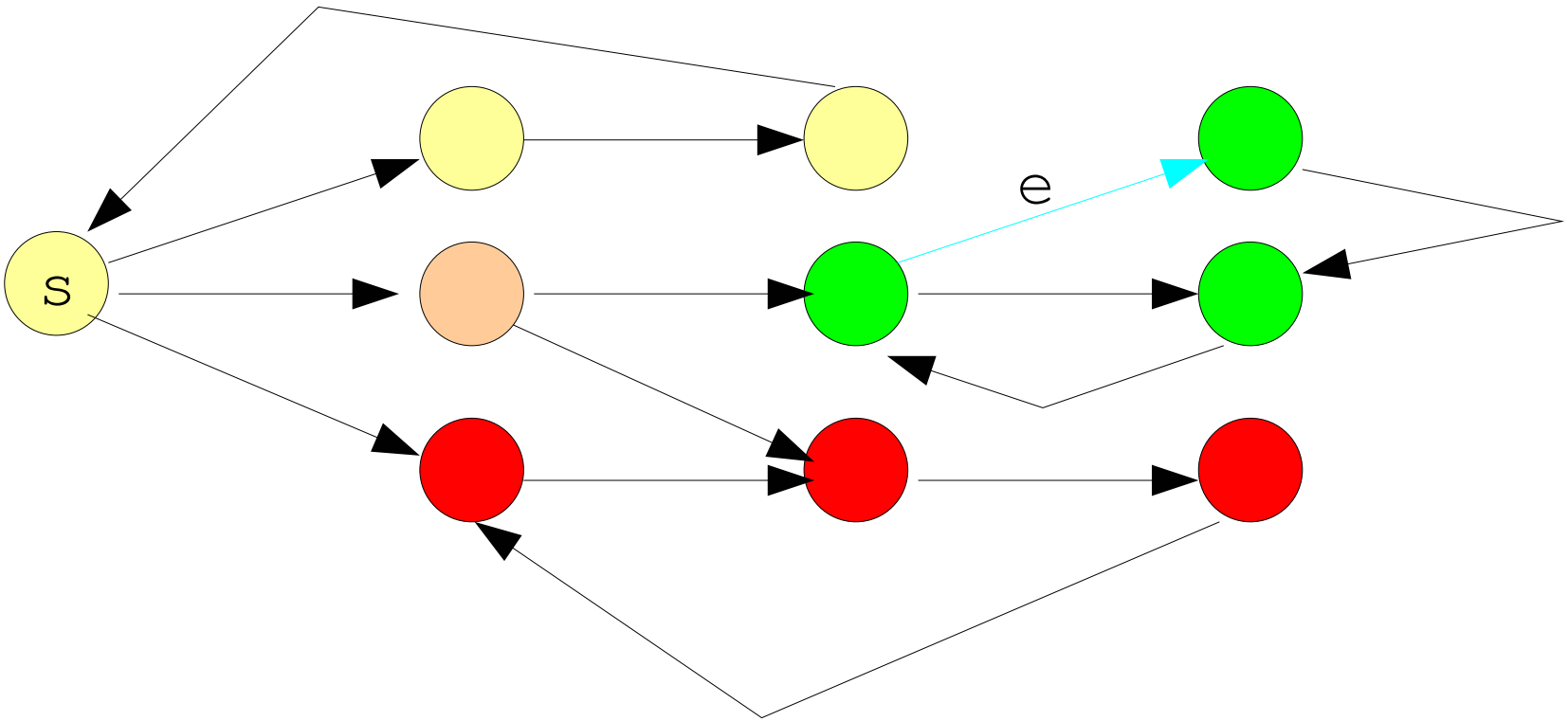
```
Search() return(rsearch(s,s',|S|-1)) ;

rsearch(scur,snext,l)
  if scur=snext or snext ∈ Successor(scur) then return(true) ;
  if l ≤ 1 then return(false) ;
  for sint ∈ S do
    if rsearch(scur,sint,⌊l⌋ /2) and rsearch(sint,snext,⌈l⌉/2) then
      return(true) ;
  return(false) ;
```

Ces deux méthodes illustrent l'alternative « temps versus espace »

Comment tester la vivacité ?

Par l'algorithme de Tarjan

- Calcul des composantes fortement connexes.
- Recherche de l'événement sur chacune des composantes fortement connexes terminales.



Avantages et inconvénients des propriétés génériques

Vérification

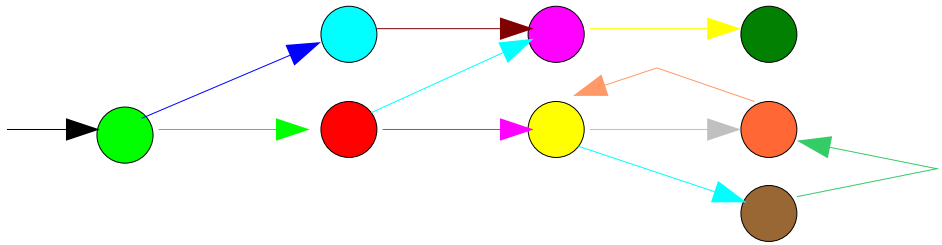
- La vérification des propriétés génériques se fait de manière efficace *(généralement en temps linéaire par rapport à la taille du système de transitions)*,
- mais les algorithmes sont des algorithmes ad hoc.

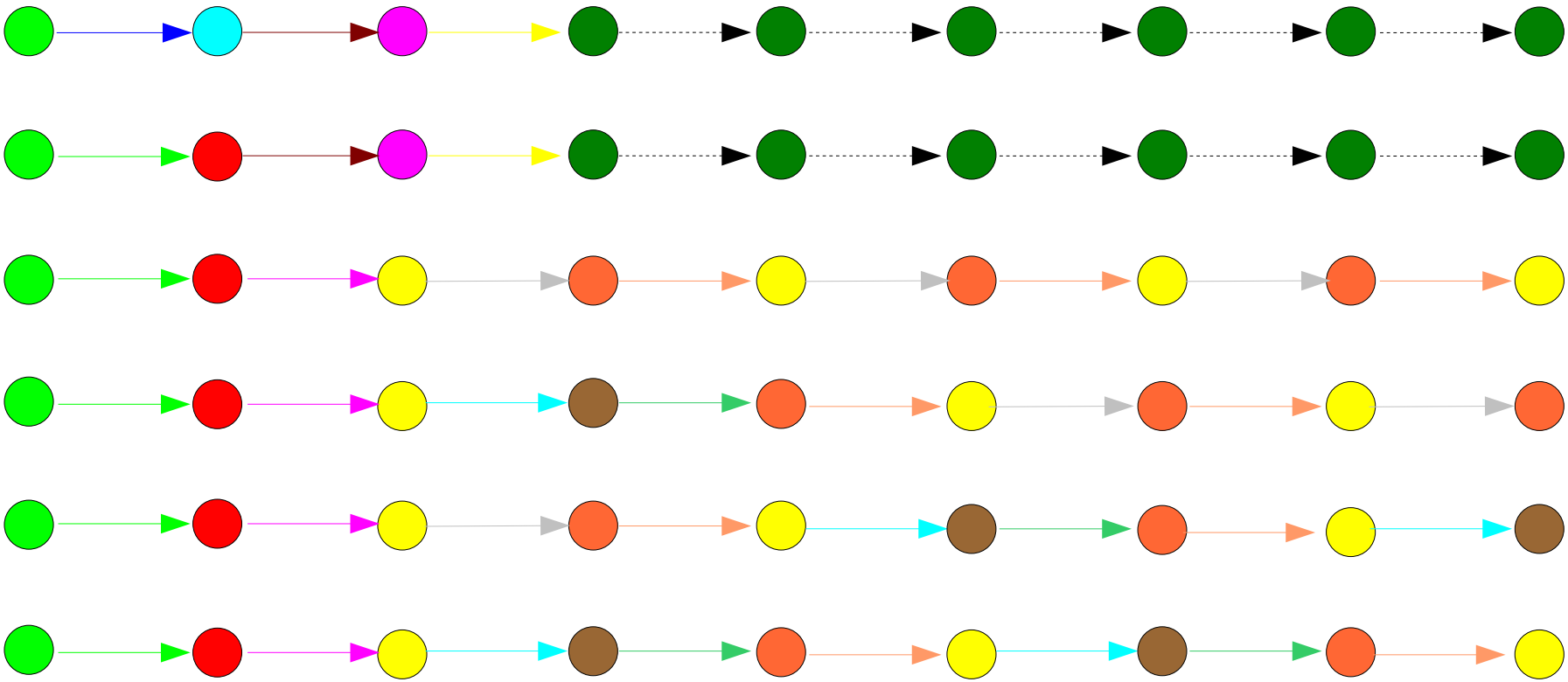
Expressivité

- La satisfaction des propriétés est une garantie de systèmes bien conçus,
- mais reste insuffisante vis à vis du comportement recherché du système *(e.g. toute requête reçue par un serveur sera traitée ou rejetée avec un message d'erreur)*

Les logiques temporelles linéaires

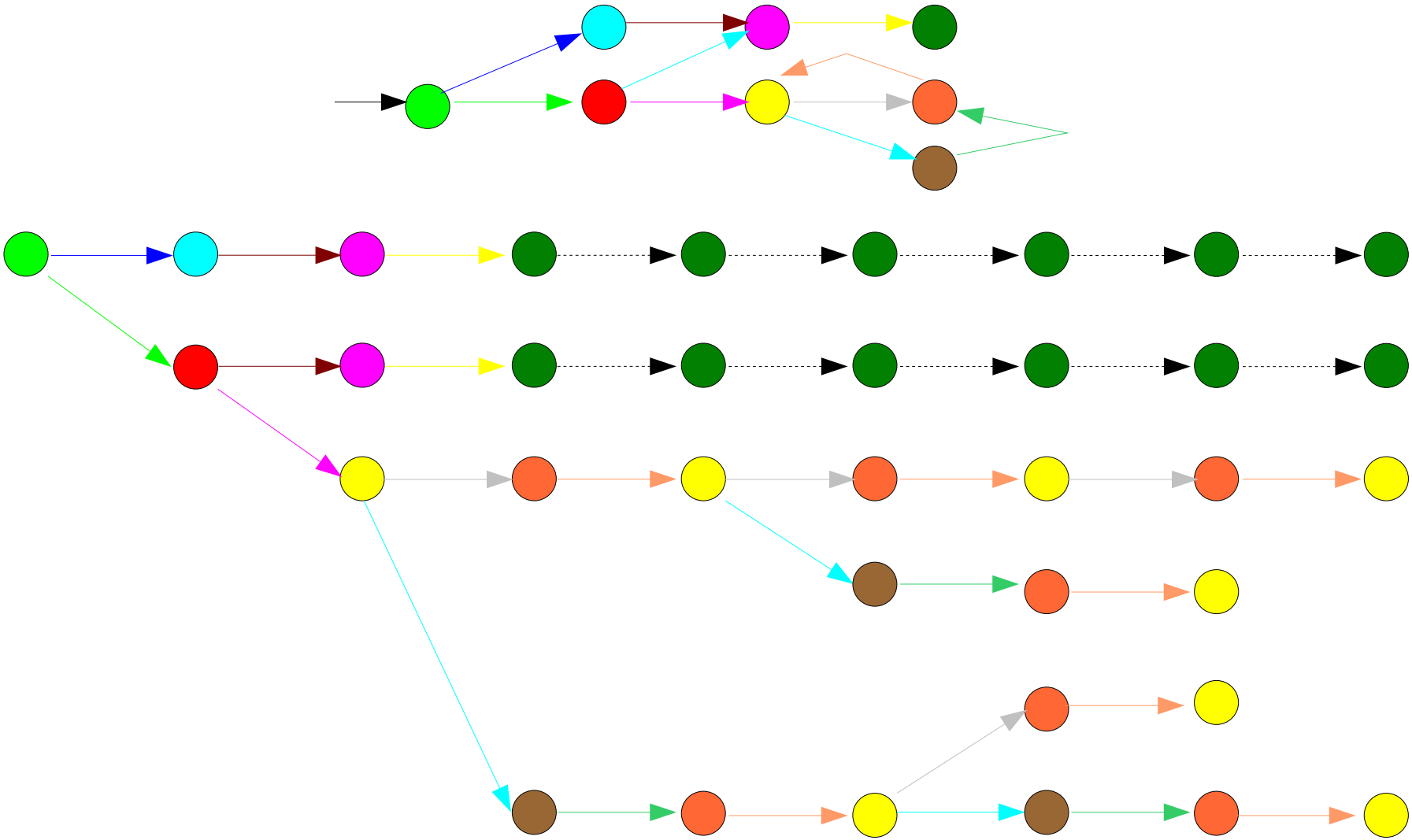
raisonnent sur les séquences infinies du système





Les logiques temporelles arborescentes

raisonnent sur l'arbre (infini) d'exécution



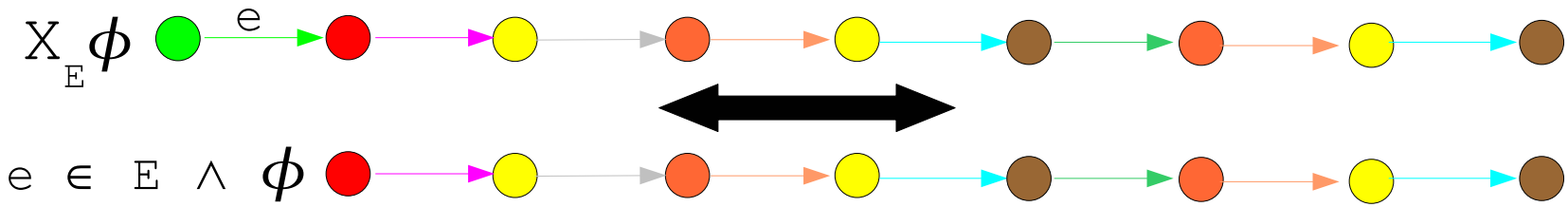
LTL

LTL est une logique temporelle linéaire qui se construit à partir :

- de propositions atomiques $p, q, \ldots$
- des opérateurs booléens $\neg, \vee, \wedge$
- d'opérateurs temporels X_E, F, G, U, W *(peut se réduire à X_E et U)*

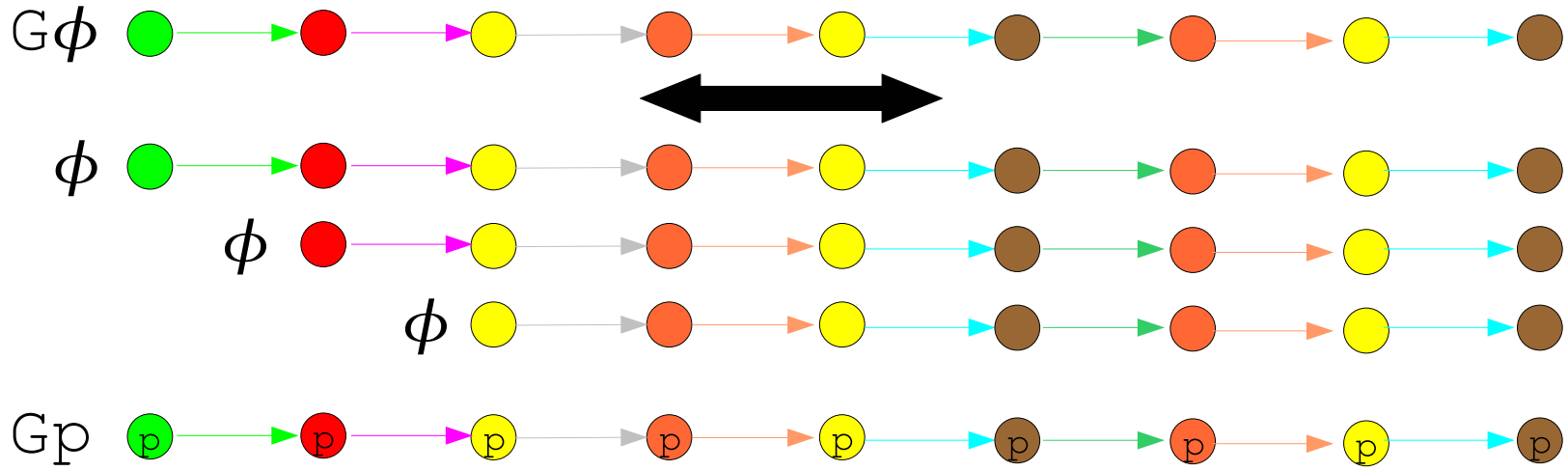
La propriété atomique p est satisfaite si elle est vraie dans le premier état de la séquence.

La formule $X_E\phi$ est satisfaite si le premier événement appartient à E et le suffixe obtenu par suppression du premier état satisfait ϕ.

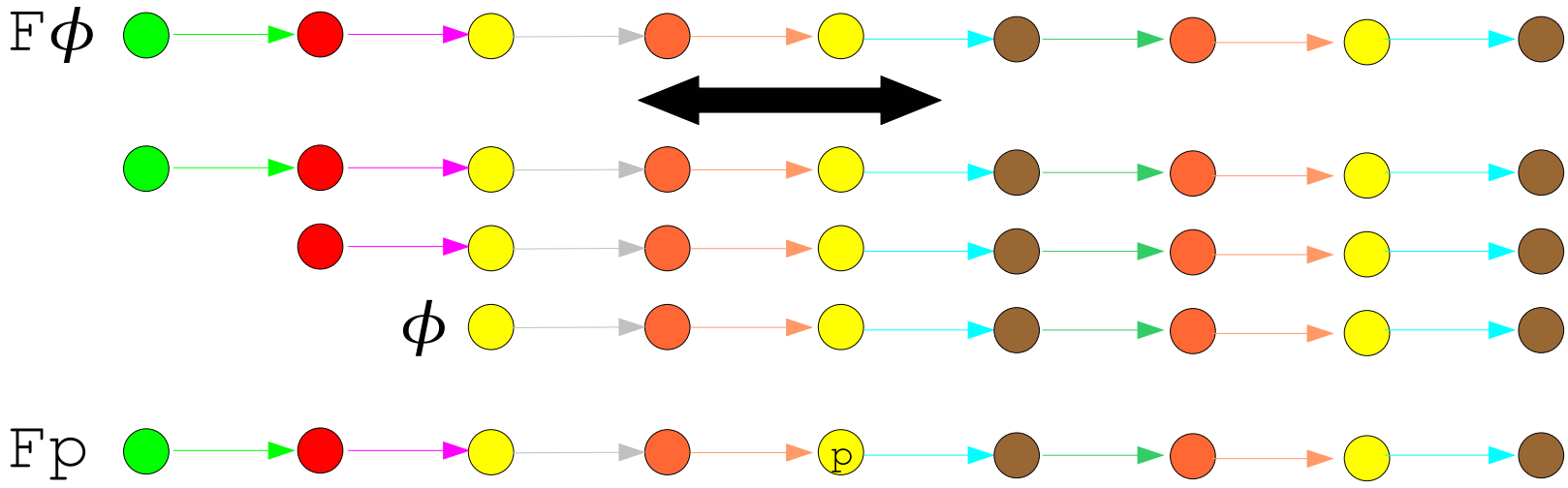


$LTL : G, F$

La formule $G\phi$ est satisfaite si ϕ est satisfaite par tous les suffixes.

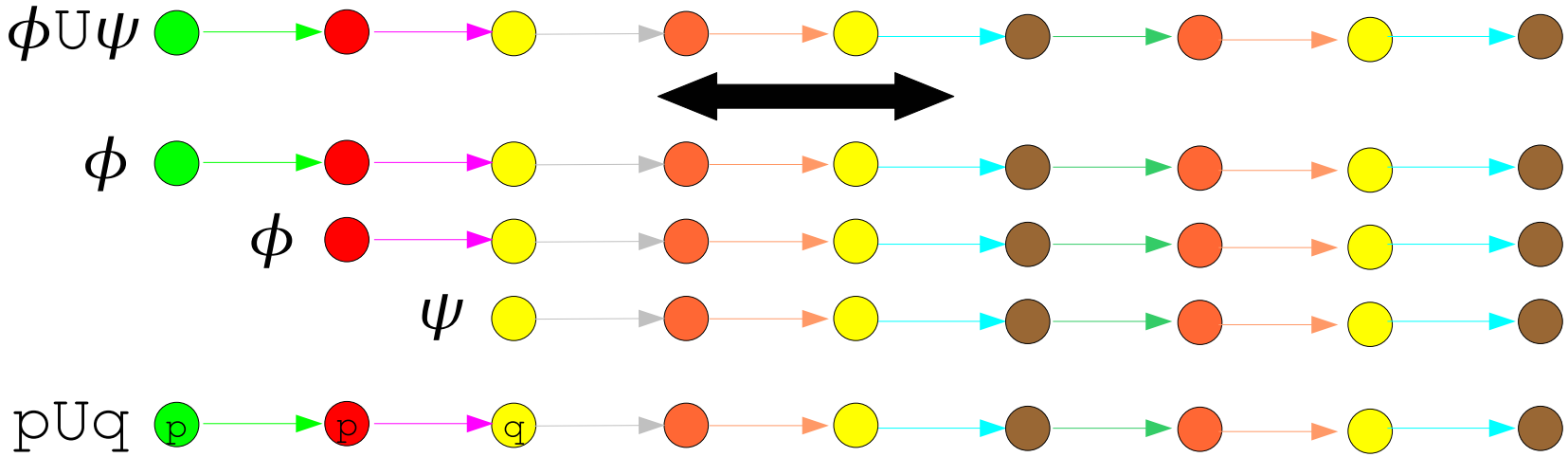


La formule $F\phi$ est satisfaite si ϕ est satisfaite par au moins un suffixe.

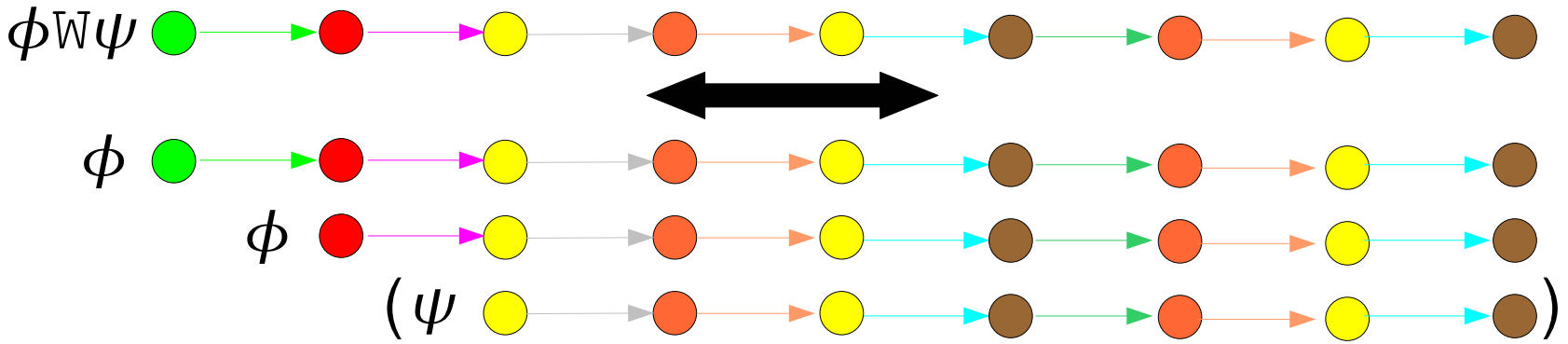


$LTL : U, W$

La formule $\phi U \psi$ est satisfaite s'il existe un suffixe t.q. ψ soit satisfaite et tous les suffixes « précédents » satisfont ϕ.



La formule $\phi W \psi$ est équivalente à $\phi U \psi \vee G\phi$.



CTL^*

CTL^* est une logique temporelle arborescente qui consiste en formules d'états et de chemins (i.e. séquences) définies inductivement.

Une formule de chemins est :

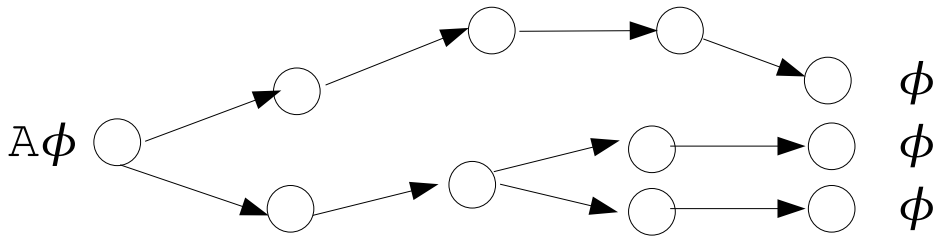
- une formule d'états évaluée sur le premier état du chemin.
- une formule $\neg\varphi, \varphi \vee \psi, \varphi \wedge \psi, X_E\varphi, F\varphi, G\varphi, \varphi U \psi, \varphi W \psi$ avec φ, ψ des formules de chemins.

Une formule d'états est :

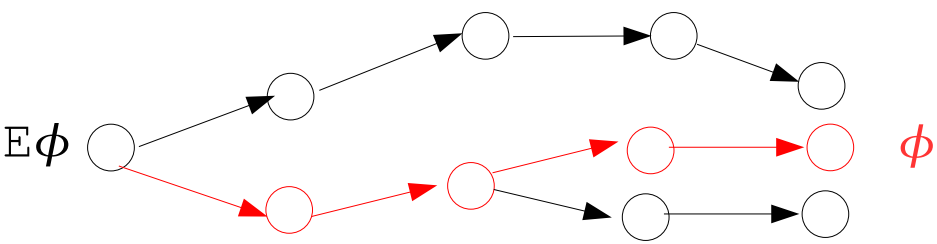
- une propriété atomique.
- une formule $A\varphi, E\varphi$ avec φ une formule de chemins.

CTL^* : A, E

La formule $A\phi$ est satisfaite si ϕ est satisfaite par tous les chemins infinis de l'arbre.



La formule $E\phi$ est satisfaite si ϕ est satisfaite par au moins un chemin infini de l'arbre.



CTL

CTL est un fragment de CTL^* qui restreint les combinaisons entre opérateurs de branchement et opérateurs temporels. Elle peut être définie inductivement uniquement à partir de formules d'états.

Une formule d'états est :

- une propriété atomique.
- une formule $\neg\varphi, \varphi \vee \psi, \varphi \wedge \psi$,
- une formule $AX_E\varphi, AF\varphi, AG\varphi, A\varphi U\psi, A\varphi W\psi$
- une formule $EX_E\varphi, EF\varphi, EG\varphi, E\varphi U\psi, E\varphi W\psi$
- avec φ, ψ des formules d'états.

LTL peut aussi être vue comme un fragment de CTL^* en préfixant chaque formule par l'opérateur A.

LTL et CTL sont incomparables.

Model Checking de CTL

Model-checking de la formule φ sur le modèle $\mathcal{M}$

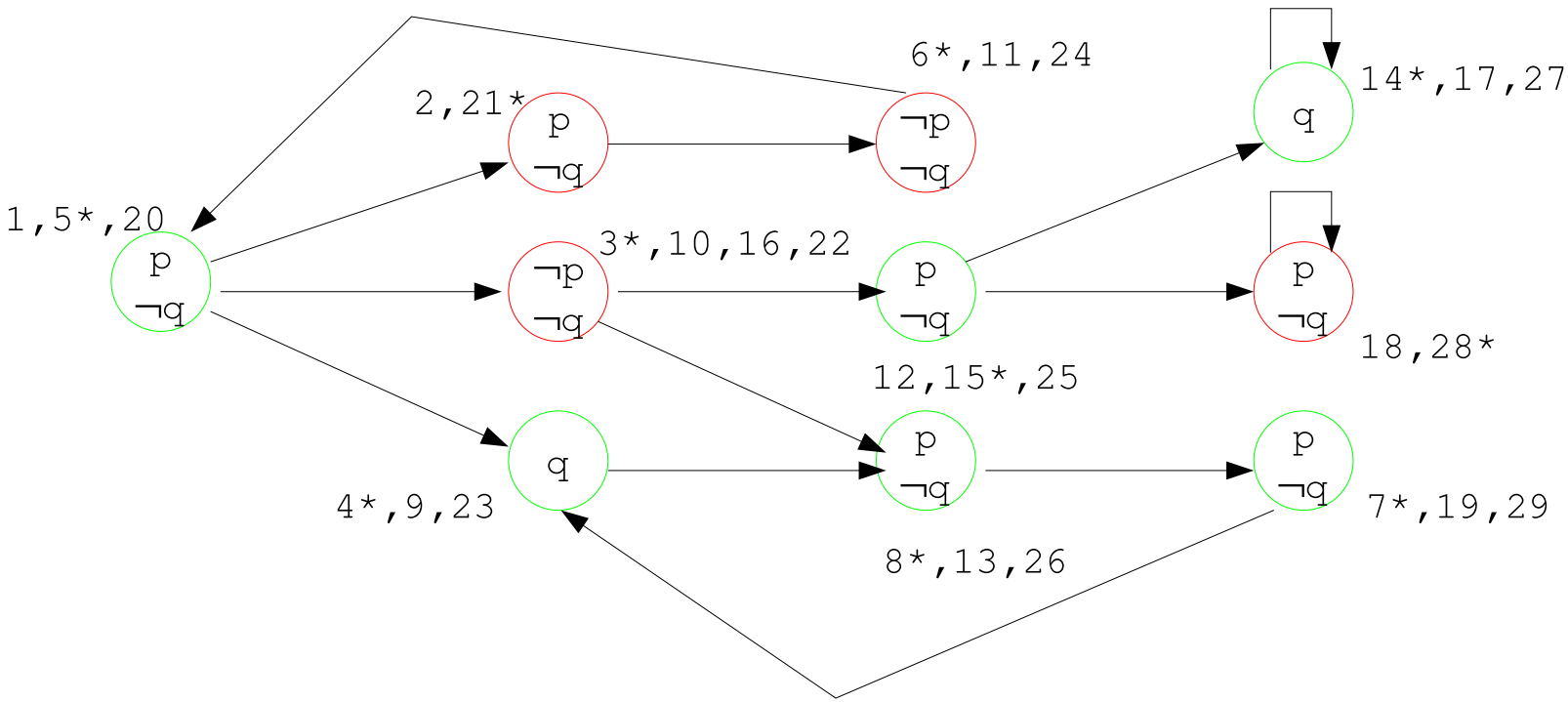
- Construire l'arbre syntaxique de φ
- Model-checking des sous-formules of φ des feuilles vers la racine
- Création des propriétés atomiques intermédiaires $[\varphi']$ correspondant aux sous-formules φ' déjà évaluées
- de telle sorte que le model-checking s'applique toujours à une sous-formule réduite à un opérateur.

Complexité du model-checking

- $PTIME$-complet
- s'effectue en temps $O(|\mathcal{M}| \cdot |\varphi|)$

Model Checking de $EpUq$

Le model-checking de $EpUq$ s'effectue par une exploration de $\mathcal{M}$ en arrière.
(en temps linéaire par rapport à $|\mathcal{M}|$)



Model Checking de LTL

Model-checking de la formule φ sur le modèle $\mathcal{M}$

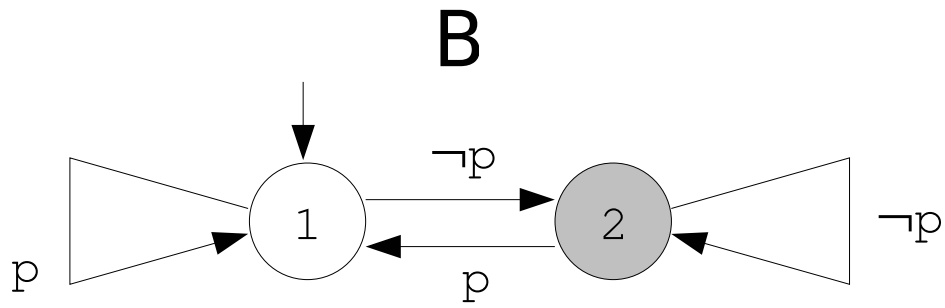
- Construction de l'automate de Büchi $\mathcal{B}$ correspondant à φ
- Construction du produit synchronisé $\mathcal{B} \otimes \mathcal{M}$ de l'automate et du modèle.
- Recherche d'un chemin acceptant dans cet automate.

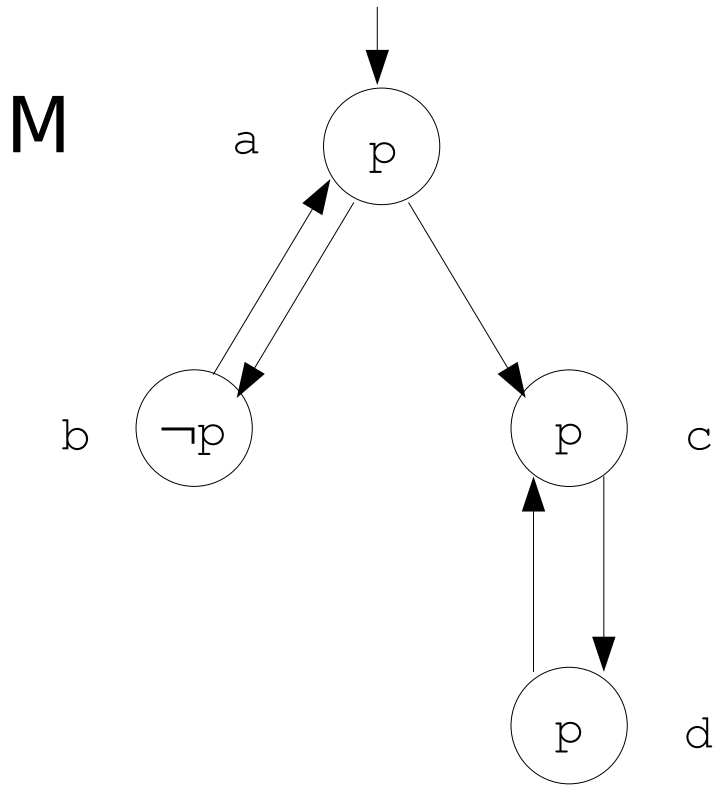
Complexité du model-checking

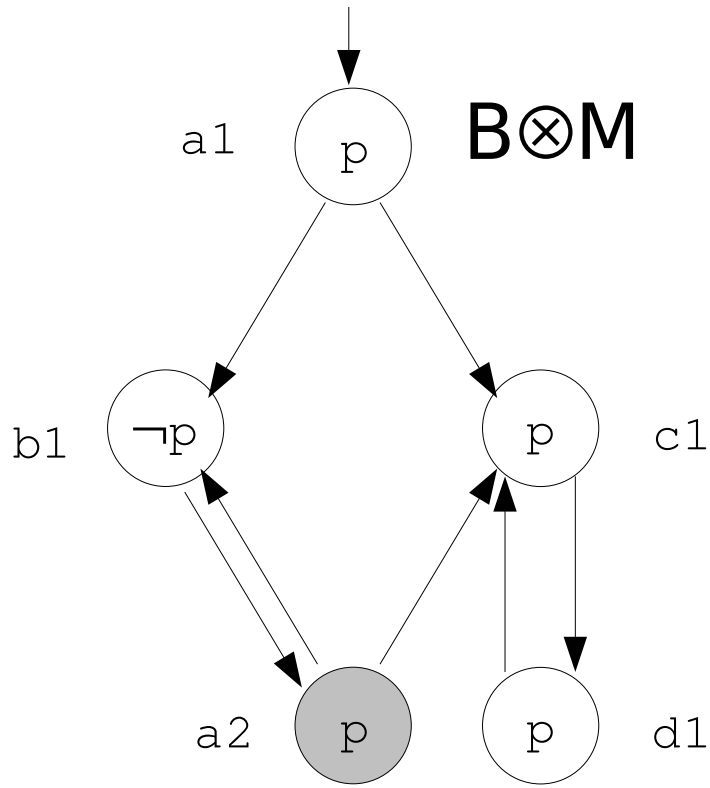
- $PSPACE$-complet (avec des techniques de construction à la volée)
- s'effectue en temps $O(|\mathcal{M}| \cdot 2^{|\varphi|})$

Le produit synchronisé

$\varphi = FGp$, $\neg\varphi = GF\neg p$







Model Checking de CTL^*

Model-checking de la formule φ sur le modèle $\mathcal{M}$

- Elimination de l'opérateur E par l'équivalence $E\psi \equiv \neg A \neg \psi$
- Construction de l'arbre syntaxique de φ
- Model-checking des sous-formules LTL $A\psi$ de φ des feuilles à la racine
- Création des propriétés atomiques intermédiaires $[A\psi]$ correspondant aux sous-formules $A\psi$ déjà évaluées

Complexité du model-checking

- $PSPACE$-complet (avec des techniques de construction à la volée)
- s'effectue en temps $O(|\mathcal{M}| \cdot 2^{|\varphi|})$

Plan

Model Checking de CTL sur une DTMC

Adaptation de CTL pour les DTMC

- Les opérateurs $A\varphi$ et $E\varphi$ sont remplacés par un unique opérateur $P_{\bowtie v}\varphi$ où v est une probabilité et $\bowtie \in \leq, <, \geq, >$.
- $s \models P_{\bowtie v}\varphi$ si la probabilité w qu'un chemin aléatoire issu de s satisfait φ vérifie $w \bowtie v$.
- $s \models P_{\geq 1}\varphi$ signifie que φ est presque sûrement satisfaite à partir de s.
- Attention ! $P_{\geq 1}\varphi$ et $A\varphi$ ne sont pas équivalentes.

Model-checking de la formule φ sur la DTMC $\mathcal{M}$

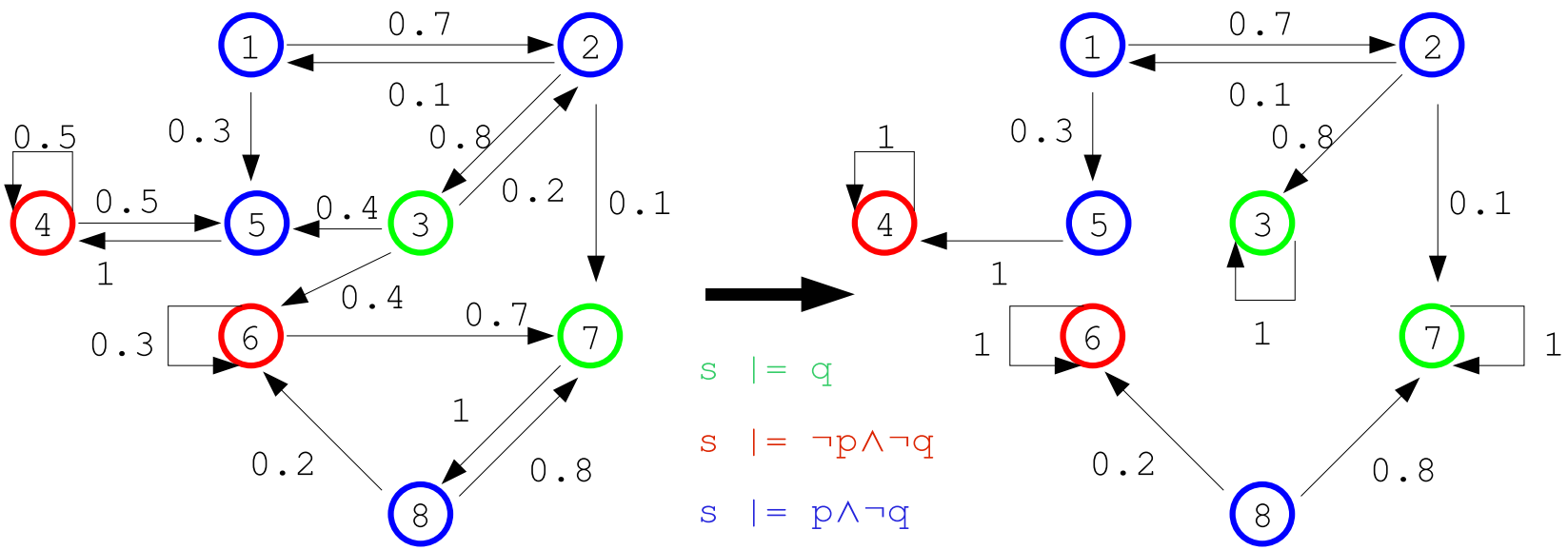
- Construction de l'arbre syntaxique de φ.
- Model-checking des sous-formules de φ des feuilles vers la racine.
- Création des propriétés atomiques intermédiaires $[\psi]$ correspondant aux sous-formules ψ déjà évaluées
- de telle sorte que le model-checking s'applique toujours à une sous-formule réduite à un opérateur.

Model Checking de $P_{\bowtie v} pUq$ sur une DTMC (1)

Transformation de la DTMC $\mathcal{M}$

Construction d'une DTMC $\mathcal{M}'$ obtenue à partir de $\mathcal{M}$ en rendant absorbant les états s tels que :

- soit $s \models q$ i.e. $Pr(Path(s) \models pUq) = 1$,
- soit $s \models \neg p \wedge \neg q$ i.e. $Pr(Path(s) \models pUq) = 0$,
- les états s (non rendus absorbants) appartenant à une composante fortement connexe terminale de $\mathcal{M}'$ vérifient $Pr(Path(s) \models pUq) = 0$.

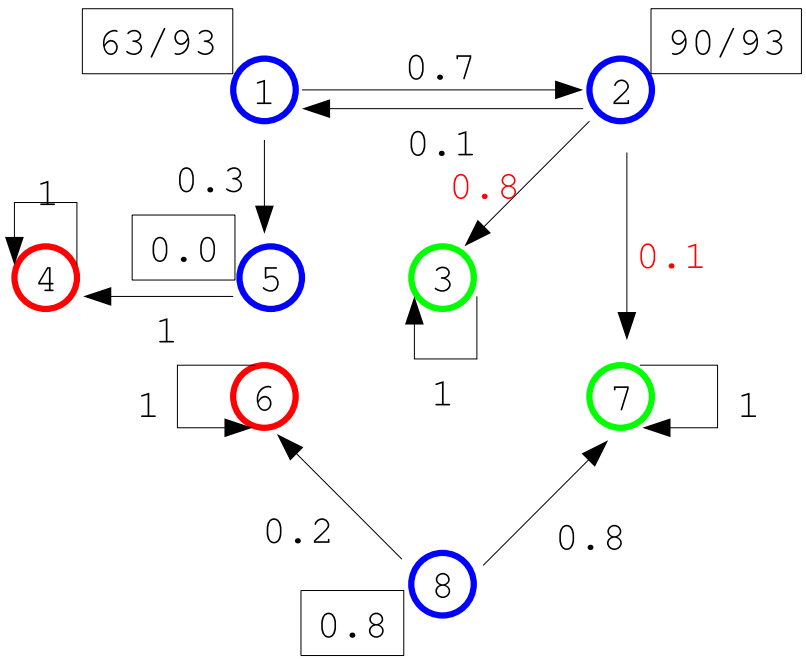


Model Checking de $P_{\bowtie v} pUq$ sur une DTMC (2)

Accessibilité dans $\mathcal{M}'$

Pour un état s transitoire, $Pr(Path(s) \models pUq)$ est la probabilité d'accéder dans $\mathcal{M}'$ aux états s' t.q. $s' \models q$.

Remarque : Si $v \in \{0, 1\}$ l'évaluation de cette formule ne dépend que de la structure du graphe.



$$T=\{1,2\}$$

$$P_{T,T} \begin{pmatrix} 0 & 0.7 \\ 0.1 & 0 \end{pmatrix}$$

$$(Id - P_{T,T})^{-1} \begin{pmatrix} 100/93 & 70/93 \\ 10/93 & 100/93 \end{pmatrix}$$

Model Checking de LTL sur une DTMC

Model-checking de la formule $P_{\bowtie v}\varphi$ sur la DTMC $\mathcal{M}$

- Construction de l'arbre syntaxique de φ.
- Calcul des probabilités des sous-formules of φ des feuilles vers la racine.
- Création des propriétés atomiques intermédiaires $[\psi]$ correspondant aux sous-formules ψ déjà évaluées.
- Transformation de la chaîne après chaque évaluation.

Transformation de la DTMC $\mathcal{M}$

Construction d'une DTMC $\mathcal{M}'$ obtenue à partir de $\mathcal{M}$ après le calcul des probabilités de $Pr(Path(s) \models \psi)$:

- Duplication de chaque état s t.q. $0 < Pr(Path(s) \models \psi) < 1$ en deux états s^y et s^n étiquetés respectivement par $[\varphi]$ et $\neg[\varphi]$
- Les autres états s sont étiquetés selon la valeur (0 ou 1) de $Pr(Path(s) \models \psi)$.

Spécification de $\mathcal{M}'$

On note $py(s) \equiv Pr(Path(s) \models \psi)$ et $pn(s) \equiv 1 - py(s)$

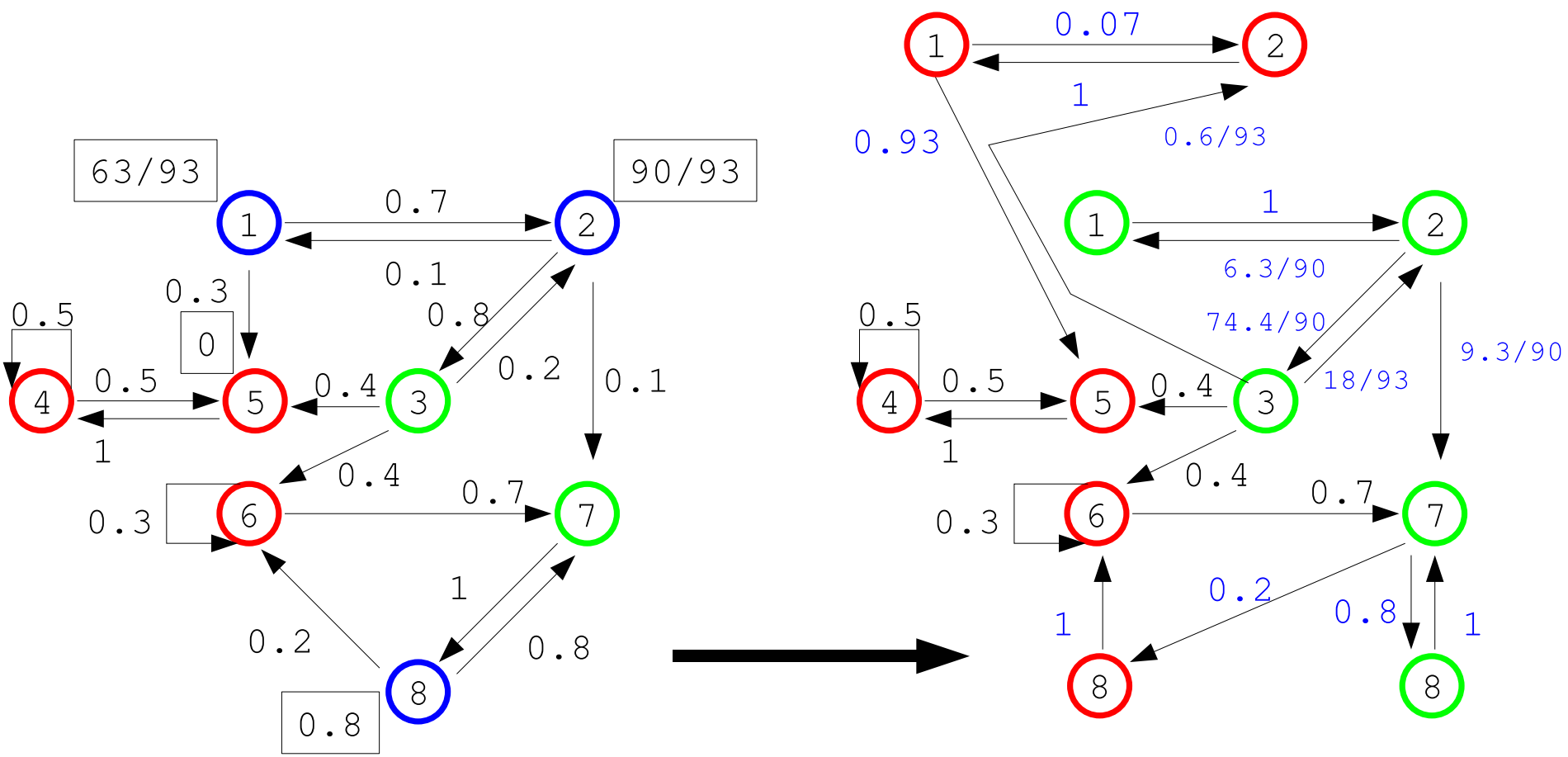
Transformation de la distribution initiale π_0

- Pour les états s non dupliqués, $\pi'_0(s) \equiv \pi_0(s)$.
- Pour les états s dupliqués, $\pi'_0(s^y) \equiv \pi_0(s)py(s)$ et $\pi'_0(s^n) \equiv \pi_0(s)pn(s)$.

Transformation de la matrice de transition P

- Les transitions entre états non dupliqués sont inchangées.
- Soit s' un état non dupliqué et s un état dupliqué,
  $P'[s', s^y] \equiv P[s', s]py(s)$ et $P'[s', s^y] \equiv P[s', s]py(s)$.
- Soit s' un état non dupliqué t.q. $py(s) = 1$ et s un état dupliqué,
  $P'[s^y, s'] \equiv P[s, s']/py(s)$ et $P'[s^n, s'] \equiv 0$.
- Soit s' un état non dupliqué t.q. $py(s) = 0$ et s un état dupliqué,
  $P'[s^n, s'] \equiv P[s, s']/pn(s)$ et $P'[s^y, s'] \equiv 0$.
- Soient s, s' deux états dupliqués,
  $P'[s^y, s'^y] \equiv P[s, s']py(s')/py(s)$, $P'[s^n, s'^n] \equiv P[s, s']pn(s')/pn(s)$
  et $P'[s^y, s'^y] = P'[s^n, s'^n] \equiv 0$.

Illustration de la transformation de $\mathcal{M}$



$$\pi_0'(1^y)=(63/93)\pi_0(1),\quad \pi_0'(1^n)=(30/93)\pi_0(1)$$

$$\pi_0'(2^y)=(90/93)\pi_0(2),\quad \pi_0'(2^n)=(3/93)\pi_0(2)$$

$$\pi_0'(8^y)=0.8\pi_0(8),\quad \pi_0'(8^n)=0.2\pi_0(8)$$

Correction de la transformation de $\mathcal{M}$

Notations

- $Path$ désigne un chemin aléatoire quelconque.
- S_o dénote les états non dupliqués.
- $abs(\mathcal{M}')$ désigne le processus stochastique obtenu à partir de $\mathcal{M}'$ en oubliant les qualificatifs ^{y} et ^{n}.

Observations

- Relativement à la distribution initiale, $abs(\mathcal{M}')$ est une agrégation faible de $\mathcal{M}'$ qui coïncide avec $\mathcal{M}$ *(choix des valeurs de transition)*.
- La probabilité qu'un chemin $Path$ de $\mathcal{M}'$ ne rencontre pas S_o est nulle.
- Par conséquent, la probabilité que, sur un état d'un chemin $Path$ de $\mathcal{M}'$, ψ et $[\psi]$ ne coïncident pas est nulle.

Correction de la transformation de $\mathcal{M}$

Sketch de la preuve

- *En vertu de l'agrégation faible,*

$$Pr_{\mathcal{M}}(Path \models \varphi) = Pr_{abs(\mathcal{M}')}(Path \models \varphi)$$

- *Puisque la satisfaction de la formule ne dépend que de l'abstraction,*

$$= Pr_{\mathcal{M}'}(Path \models \varphi)$$

- *Puisque les chemins sur lesquels $[\psi]$ ne coincide pas avec ψ sont de mesure nulle,*

$$= Pr_{\mathcal{M}'}(Path \models \varphi(\psi \leftarrow [\psi]))$$

LTL versus Automates de Büchi

Comparaison

- Les automates de Büchi sont (légèrement) plus expressifs que les formules LTL.
- Pour certains utilisateurs, les automates de Büchi sont plus compréhensibles en raison de leur représentation graphique.
- La vérification repose sur une opération simple le produit synchronisé.

Difficultés de mise en oeuvre

- Si l'automate n'est pas déterministe, le produit synchronisé n'est pas un processus stochastique...
- mais les automates déterministes sont (nettement) moins expressifs.

Les automates déterministes de Müller

Syntaxe et sémantique

- La condition d'acceptation s'exprime par un ensemble de sous-ensembles d'états $\{R_i\}_{i \in I}$.
- Un chemin est accepté si l'ensemble des états rencontrés infiniment est l'un des R_i.
- Ces automates sont aussi expressifs que les automates de Büchi.

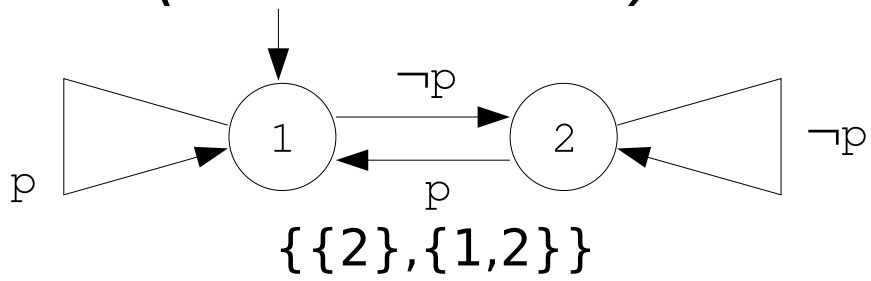
Model checking d'un DMA sur une DTMC

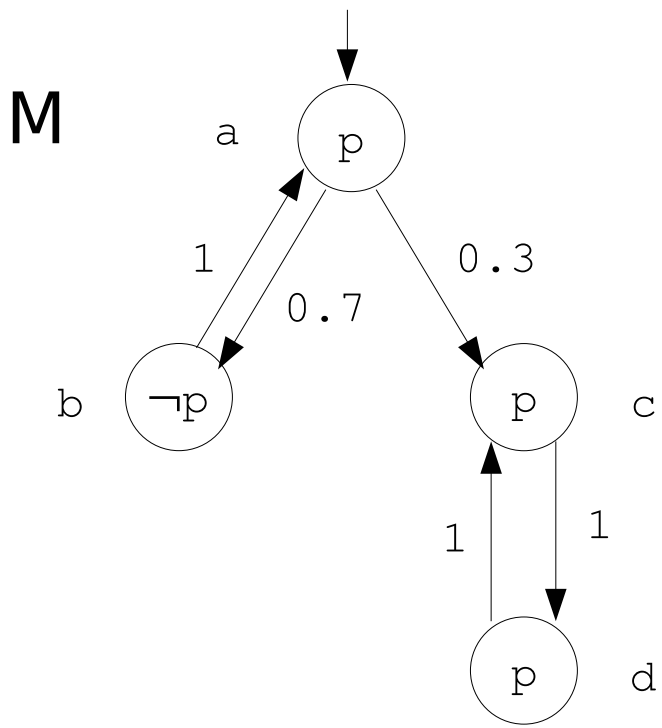
- Construction du produit synchronisé qui est une DTMC.
- Une composante fortement connexe terminale est dite *acceptante* si l'ensemble des états de l'automate qui y apparaissent est l'un des R_i.
- La probabilité de satisfaction du DMA est la probabilité d'atteindre une c.f.c. acceptante.

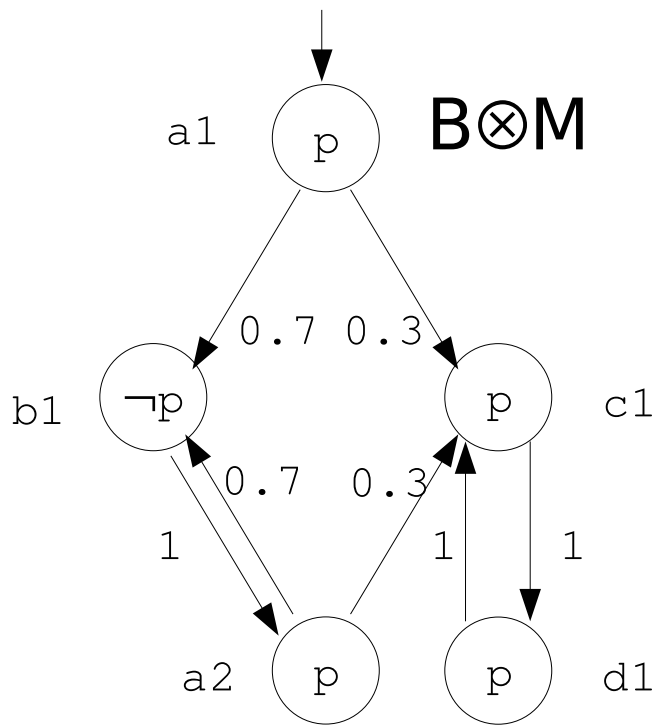
Illustration du produit synchronisé

GF¬p

(déterministe)







Plan

Quelques indices utiles

- Garantie de disponibilité instantanée en régime transitoire
  (probabilité à un instant τ de la disponibilité du service)
- Garantie de disponibilité instantanée en régime stationnaire
  (probabilité à un instant donné de la disponibilité du service en régime stationnaire)
- Garantie de disponibilité dans la durée en régime transitoire
  (probabilité que le service soit constamment disponible entre deux instants τ et τ')
- Garantie de disponibilité dans la durée en régime stationnaire
  (probabilité que le service soit constamment disponible entre deux instants en régime stationnaire. Cette mesure ne dépend que la durée de l'intervalle constitué des deux instants)
- Garantie de disponibilité et de temps de réponse en régime stationnaire
  (probabilité qu'après une requête, le service soit fonctionnel jusqu'à la réponse et que le temps de réponse n'excède pas une borne donnée)

CSL : une logique pour les CTMC

Syntaxe de CSL

Une formule d'états est :

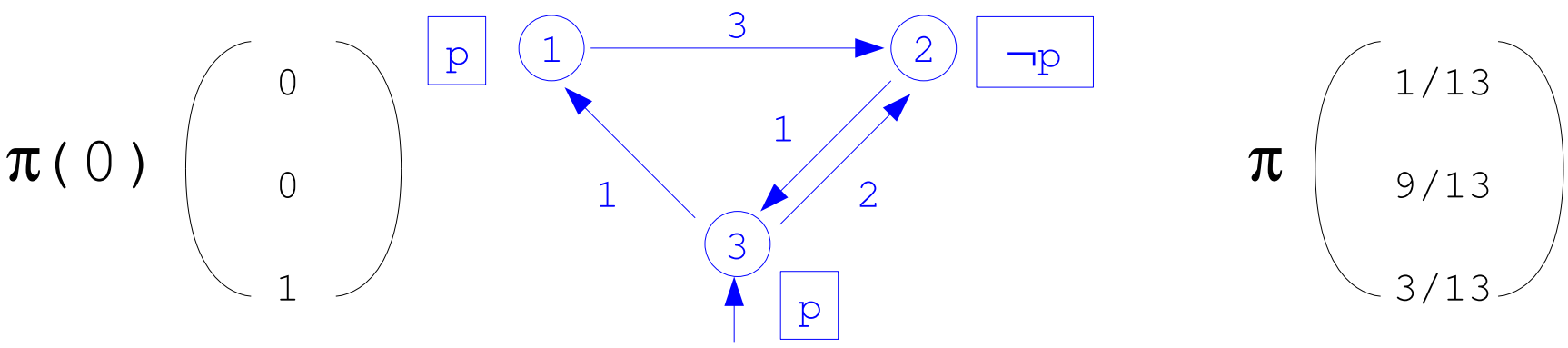
- Une propriété atomique,
- Une formule $\neg\varphi, \varphi \wedge \psi, \varphi \vee \psi$
- Une formule $S_{\bowtie v}\varphi, P_{\bowtie v}X^I\varphi, P_{\bowtie v}\psi U^I\varphi$

avec $\bowtie \in \{<, \leq, \geq, >\}$, v une probabilité, I un intervalle positif et φ, ψ des formules d'états.

Sémantique de CSL

- $s \models S_{\bowtie v}\varphi$ si étant donnée π_s la distribution stationnaire obtenue à partir de s, $\sum_{s' \models \varphi} \pi_s(s') \bowtie v$.
- Un chemin satisfait $X^I\varphi$ si l'instant du premier événement appartient à I et l'état atteint satisfait φ. $s \models P_{\bowtie v}X^I\varphi$ si w la probabilité qu'un chemin issu de s satisfait $X^I\varphi$, vérifie $w \bowtie v$
- Un chemin satisfait $\psi U^I\varphi$ s'il existe un instant $\tau \in I$ t.q. l'état atteint à l'instant τ satisfait φ et les états précédents satisfont φ. $s \models P_{\bowtie v}\psi U^I\varphi$ si w la probabilité qu'un chemin issu de s satisfait $\psi U^I\varphi$, vérifie $w \bowtie v$

CSL : illustration



- $S_{>0.5}p$ n'est pas satisfaite puisque p est vraie avec une probabilité $4/13$ à l'état stationnaire.
- $P_{<0.5}X^{[0,1]}p$ est vraie indépendamment de l'instant du premier changement car la probabilité d'aller de l'état 3 à l'état 1 est $1/3$.
- $P_{>v}pU^{[0,1]}\neg p$ est vraie si $\frac{1}{3}(1-4e^{-3})+\frac{2}{3}(1-e^{-3})>v$.

Formalisation des indices

- Garantie de disponibilité instantanée en régime transitoire

$$P_{\geq 0.99}\ \texttt{true}\ U^{[\tau,\tau]}\ \texttt{available}$$

- Garantie de disponibilité instantanée en régime stationnaire

$$S_{\geq 0.99}\ \texttt{available}$$

- Garantie de disponibilité dans la durée en régime transitoire

$$P_{<0.01}\ \texttt{true}\ U^{[\tau,\tau']}\ \neg\texttt{available}$$

- Garantie de disponibilité dans la durée en régime stationnaire

$$S_{\geq 0.99}P_{<0.01}\ \texttt{true}\ U^{[\tau,\tau']}\ \neg\texttt{available}$$

- Garantie de disponibilité et de temps de réponse en régime stationnaire

$$S_{\geq 0.99}(\ \texttt{req}\ \Rightarrow P_{\geq 0.99}\ \texttt{available}\ U^{[0,3]}\ \texttt{ack}\ )$$

Model checking de $P_{\bowtie v} p U^I q$

Cas $I = [0, \infty[$

- Observation : cette formule ne dépend que de la DTMC incluse.
- Transformation de la DTMC incluse en rendant absorbants les états s t.q. $s \models \neg p \vee q$.
- Calcul dans cette DTMC de la probabilité d'atteindre les états s t.q. $s \models q$.

Cas $I = [0, \tau]$

- Transformation de la CTMC en rendant absorbants les états s t.q. $s \models \neg p \vee q$.
- Calcul dans la CTMC de la probabilité d'être à l'instant τ dans un état s t.q. $s \models q$ *(par la technique d'uniformisation)*.

Cas $I = [\tau, \tau]$

- Transformation de la CTMC en rendant absorbants les états s t.q. $s \models \neg p$.
- Calcul dans cette DTMC de la probabilité d'être à l'instant τ dans un état s t.q. $s \models \neg q$.

Model checking de $P_{\bowtie v} p U^I q$

Cas $I = [\tau, \tau']$

- Observation : la formule $pU^{[\tau,\tau']}q$ est satisfaite par un chemin si le chemin ne rencontre que des états s t.q. $s \models p$ durant $[0, \tau]$ et le suffixe du chemin à partir de τ satisfait $pU^{[0,\tau'-\tau]}q$.
- Transformation de la CTMC en rendant absorbants les états s t.q. $s \models \neg p$ et calcul des probabilités transitoires à l'instant τ en partant de s (disons π^1_s).
- Calcul des probabilités de $pU^{[0,\tau'-\tau]}q$ (disons π^2).
- La probabilité π recherchée est définie par $\pi(s) = \sum_{s' \models p} \pi^1_s(s')\pi^2(s')$.

Cas $I = [\tau, \infty[$

- Observation : la formule $pU^{[\tau,\infty[}q$ est satisfaite par un chemin si le chemin ne rencontre que des états s t.q. $s \models p$ durant $[0, \tau]$ et le suffixe du chemin à partir de τ satisfait $pU^{[0,\infty[}q$.
- Calcul des probabilités de $pU^{[0,\infty[}q$ (disons π^3).
- La probabilité π recherchée est définie par $\pi(s) = \sum_{s' \models p} \pi^1_s(s')\pi^3(s')$.

Limitations de CSL

Principaux inconvénients

- CSL est une logique « à la CTL » : elle ne permet pas d'exprimer des contraintes multiples sur un chemin.
- CSL est une logique basée sur les propriétés des états : elle ne permet pas de raisonner sur les événements.
- Le model checking de CSL est fondé sur des techniques *ad hoc* de modification de chaînes de Markov.

Une première solution : asCSL

- $asCSL$ substitue aux opérateurs X^I et U^I une expression rationnelle sur un alphabet basé sur des propositions atomiques et des événements et un intervalle.
- Une chemin satisfait la formule s'il « génère » un mot de l'expression en un temps compris dans l'intervalle.

Mais ...

- Il n'y a toujours qu'une unique contrainte temporelle sur le chemin.
- Le model checking de $asCSL$ reste fondé sur des techniques *ad hoc*.

CSL^{TA}

Une formule d'états de CSL^{TA} est :

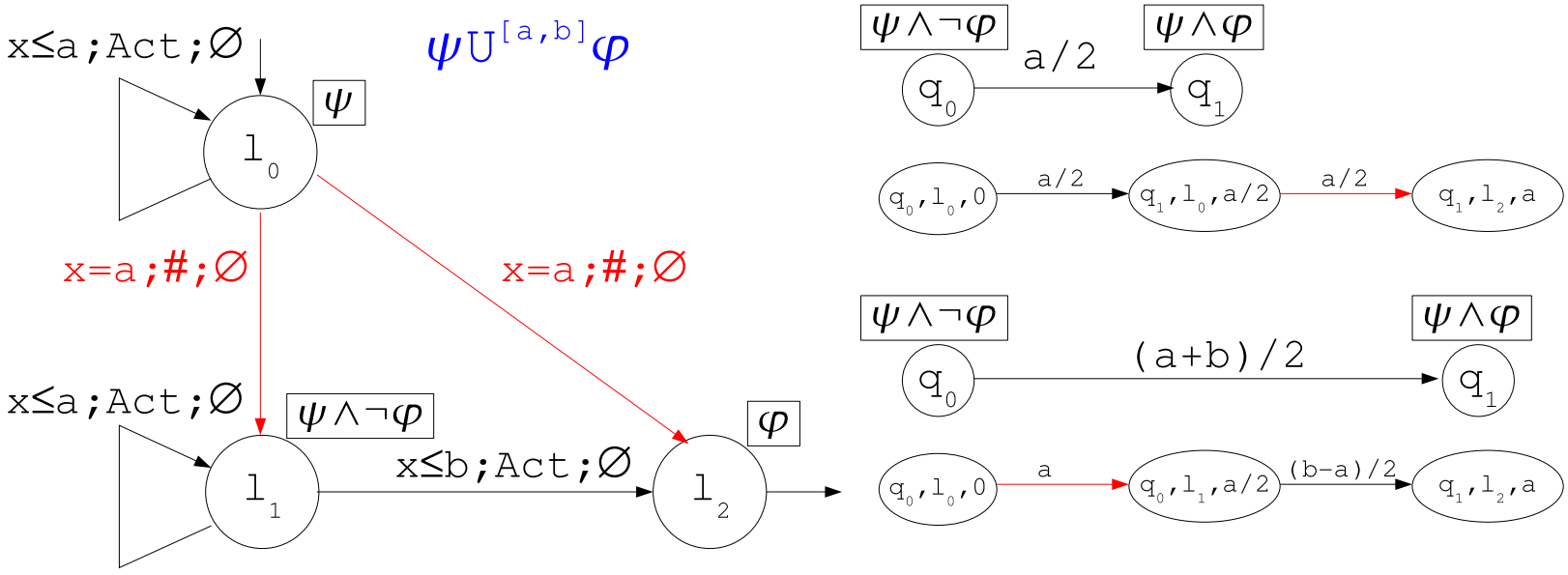
- Une propriété atomique,
- Une formule $\neg\varphi, \varphi \wedge \psi, \varphi \vee \psi$
- Une formule $S_{\bowtie v}\varphi, P_{\bowtie v}\mathcal{A}$

avec $\bowtie \in \{<, \leq, \geq, >\}$, v une probabilité, I un intervalle positif, φ, ψ des formules d'états et $\mathcal{A}$ une formule de chemins.

Une formule de chemins de CSL^{TA} est :

- Un automate temporisé déterministe, avec une unique horloge
- dont chaque état est étiqueté par une formule d'état,
- et chaque transition par un sous-ensemble d'événements ou par l'action urgente $\sharp$.
- La contrainte temporelle d'une transition événementielle est un intervalle non ponctuel tandis que celle d'une transition urgente est un point.

CSL^{TA}



Acceptation d'un chemin par un DTA :

- On couple l'état courant du chemin avec la configuration du DTA.
- Si le prochain événement du chemin a lieu avant la franchissabilité d'une transition urgente, la simulation se poursuit si cet événement et l'état atteint peuvent être couplés avec une transition du DTA (sinon échec).
- Sinon la transition urgente est franchie à condition que la localité atteinte puisse être couplée avec l'état courant du chemin (sinon échec).

Model checking de CSL^{TA} sur une CTMC

Caractérisation du produit synchronisé

- Le produit synchronisé de l'automate et de la chaine (étendu avec un état d'acceptation $\top$ et de rejet $\bot$) est un processus stochastique semi-régénératif
- dont les points de régénération sont (q, l, c) avec c étant soit 0 soit l'une des constantes testées.
- La probabilité d'acceptation par l'automate est la probabilité d'atteindre $\top$.

Calcul de la probabilité d'accessibilité

- Le processus semi-régénératif admet une DTMC incluse dont les états sont les points de régénération.
- La matrice de transition se calcule par analyse transitoire de CTMC « subordonnées ».
- La probabilité d'atteindre $\top$ s'évalue dans la DTMC incluse.

Expressivité de CSL^{TA}

$$CSL \subsetneq CSL^{TA}$$

$$asCSL \subseteq CSL^{TA}$$

Sans emboîtement d'automates, $asCSL \subsetneq CSL^{TA}$

Quelques axes de recherche

Model Checking de DTMC dénombrable

- *Stochastic Context-Free Grammars*
- *Multi-Type Branching Processes*
- *Recursive Markov Chains*

Logiques étendues pour CTMC

- avec analyse d'expressivité
- et méthode d'évaluation

Model checking efficace

- par combinaison avec les techniques d'ordre stochatique
- model checking des systèmes à forme produit
- etc.