CENTRE DE RECHERCHE EN
AUTOMATIQUE DE
NANCY

EVALUATION DE LA ROBUSTESSE D'UN ORDONNANCEMENT SOUS PERTURBATION : APPROCHE PAR DES SED

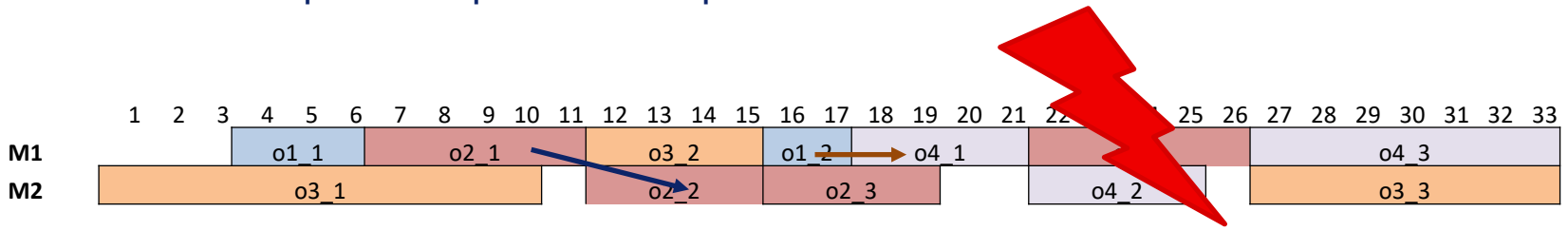
A. AUBRY, S. HIMMICHE, D. LEMOINE, P. MARANGE, S. NORRE

UMR 7039

UNIVERSITÉ DE LORRAINE

CONTEXTE

- **Un atelier de production :**
  - Atelier (Job Shop, Flow Shop, Open Shop) est un ensemble de machines et de jobs (tâches).
  - Chaque job a besoin d'un ensemble d'opérations pour être réalisé.
- **Un ordonnancement de production S :**
  - Allocation des opérations aux machines adaptées .
  - Séquencement des opérations sur les machines (dates de début et de fin).
  - Durée totale de l'ordonnancement C_{max}.
  - Contraintes de précédence:
    - Précédence contrainte par la gamme du job,
    - Précédence contrainte par la séquence des opérations sur la machine.



- **Quel est l'impact de la perturbation sur un ordonnancement prédictif ?**

CONTEXTE

- **Génération d'ordonnancements en tenant plus ou moins compte des perturbations**
  - Problème 1 : Ne tient pas compte des exigences du décideur pour connaitre la robustesse de son ordonnance. Par exemple, Comment lui assurer 90% de disponibilité de son ordonnancement ?
  - Problème 2 : Ne tient pas compte de tout l'intervalle de perturbation

- **L'objectif est de proposer une approche complémentaire avec des outils SED qui permet de modéliser tout l'intervalle de perturbation et d'évaluer la robustesse de celui-ci**

PLAN

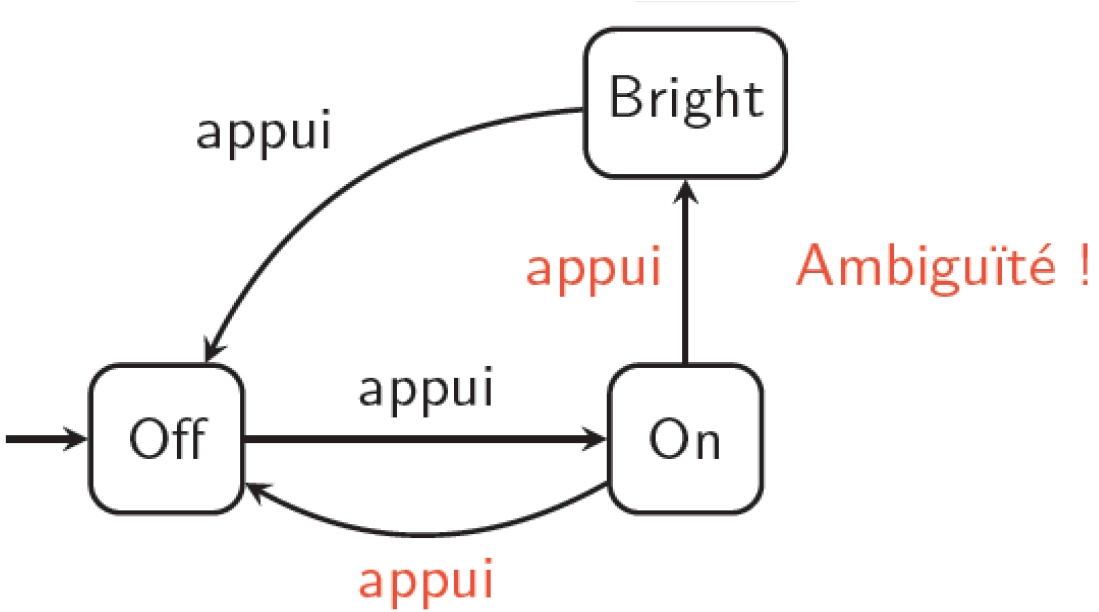
- **Outil de modélisation**
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- **Outil d'évaluation - Vérification**
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- **Approche d'évaluation**
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- **Application sur UppAal SMC**
- **Couplage Approche robustification et Evaluation Niveau de service**
- **Application Cplex-UppAal SMC**

PLAN

- **Outil de modélisation**
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- Outil d'évaluation - Vérification
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- Approche d'évaluation
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- Application sur UppAal SMC
- Couplage Approche robustification et Evaluation Niveau de service
- Application Cplex-UppAal SMC

BESOIN DE MODÉLISER LE TEMPS

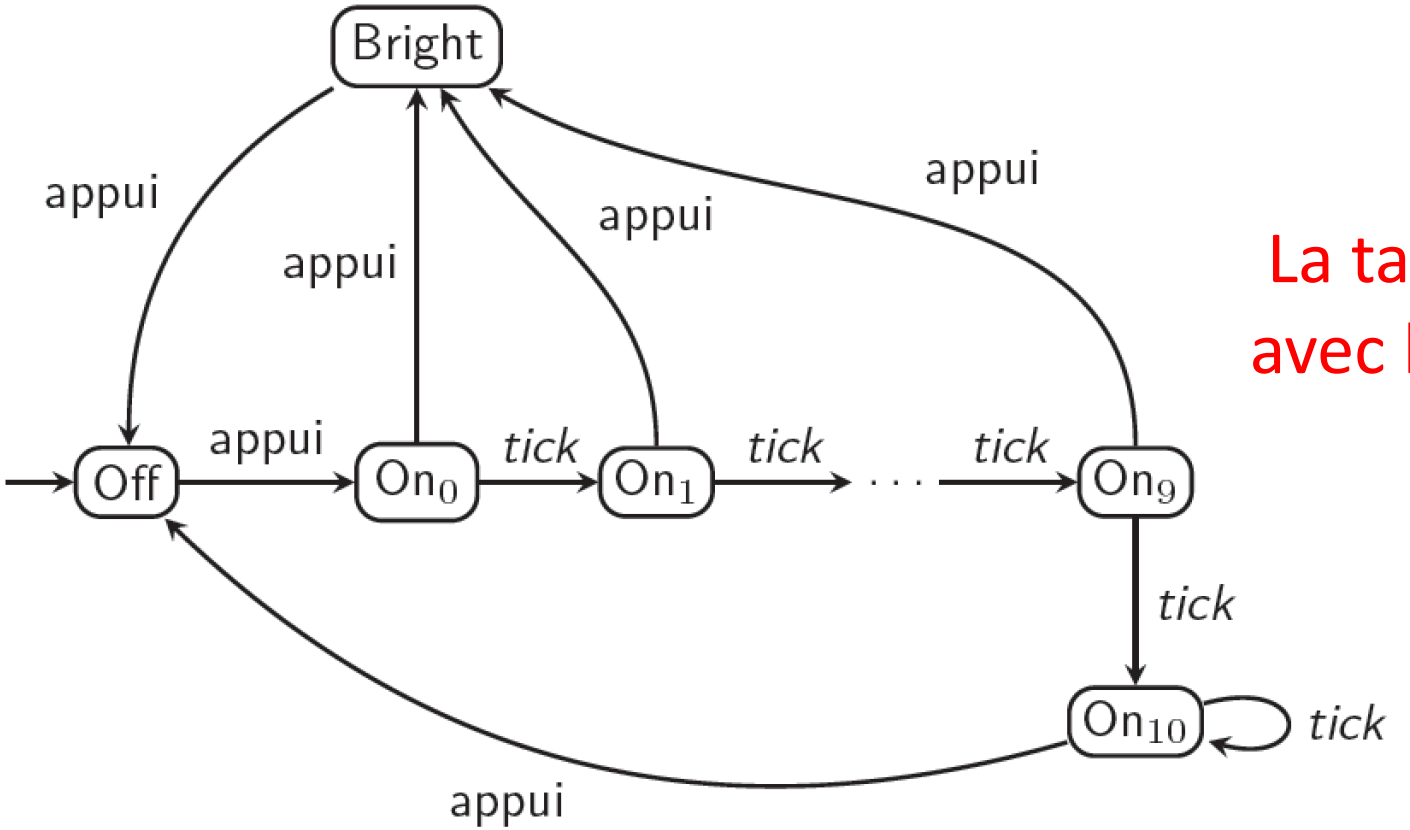
- **Si j'appuie sur le bouton la lumière s'allume. Si j'appuie deux fois (rapidement) sur le bouton, la lumière s'allume plus fort. Si j'appuie à nouveau sur le bouton, la lumière s'éteint.**



Besoin de modéliser le temps

Ajouter une information discrète

- On utilise une action spéciale tick . Ici tick marque l'écoulement de1/10 secondes. On modélise l'appui « rapide » par deux appuis en moins d'une seconde.

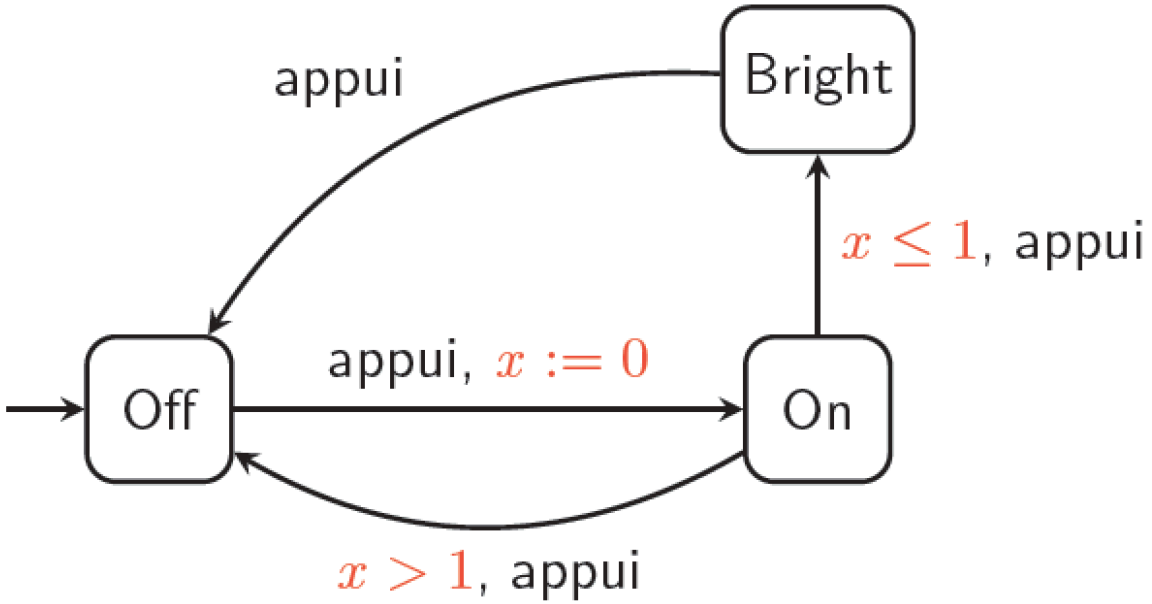


La taille du modèle croît avec la finesse du temps. . .

Besoin de modéliser le temps

Ajouter une information continue

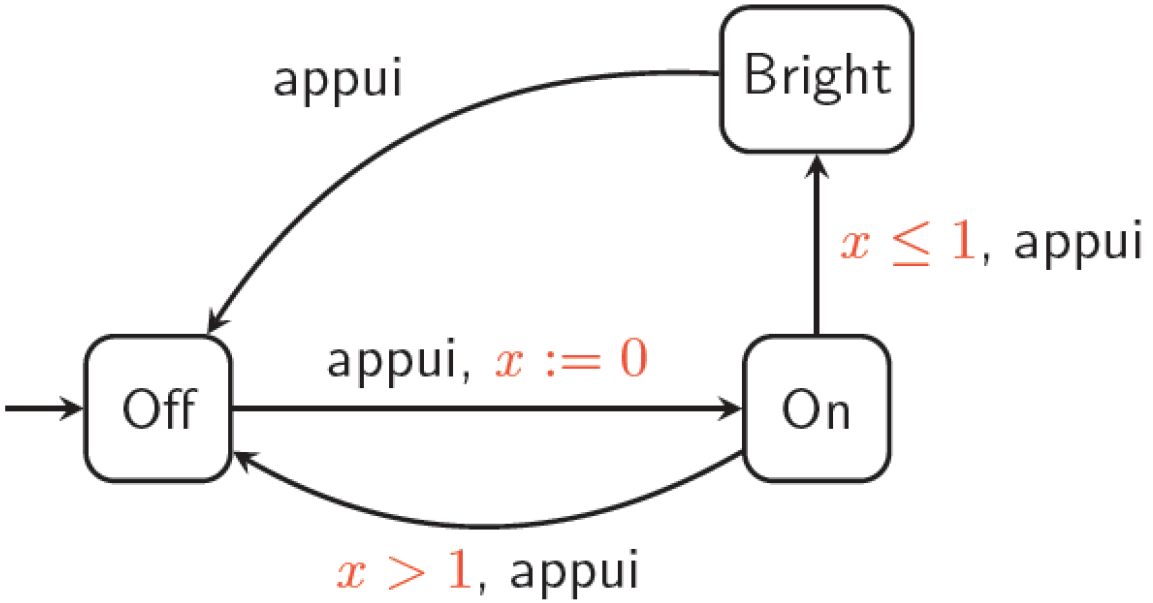
- Le temps est modélisé sur R → Il est donc arbitrairement précis.
- L'exécution est contrainte par des constantes (dans N).
- Au sein du modèle, des horloges mesurent ce temps.



BESOIN DE MODÉLISER LE TEMPS - EXÉCUTION

- **Exécution possible du système**

$$\text{Off} \xrightarrow{0{,}42} \text{Off} \xrightarrow{\text{appui}} \text{On} \xrightarrow{0{,}197} \text{On} \xrightarrow{\text{appui}} \text{Bright} \xrightarrow{2{,}3} \text{Bright} \xrightarrow{\text{appui}} \text{Off} \ldots$$



MODÉLISATION

Un automate temporisé (TA) est formé :

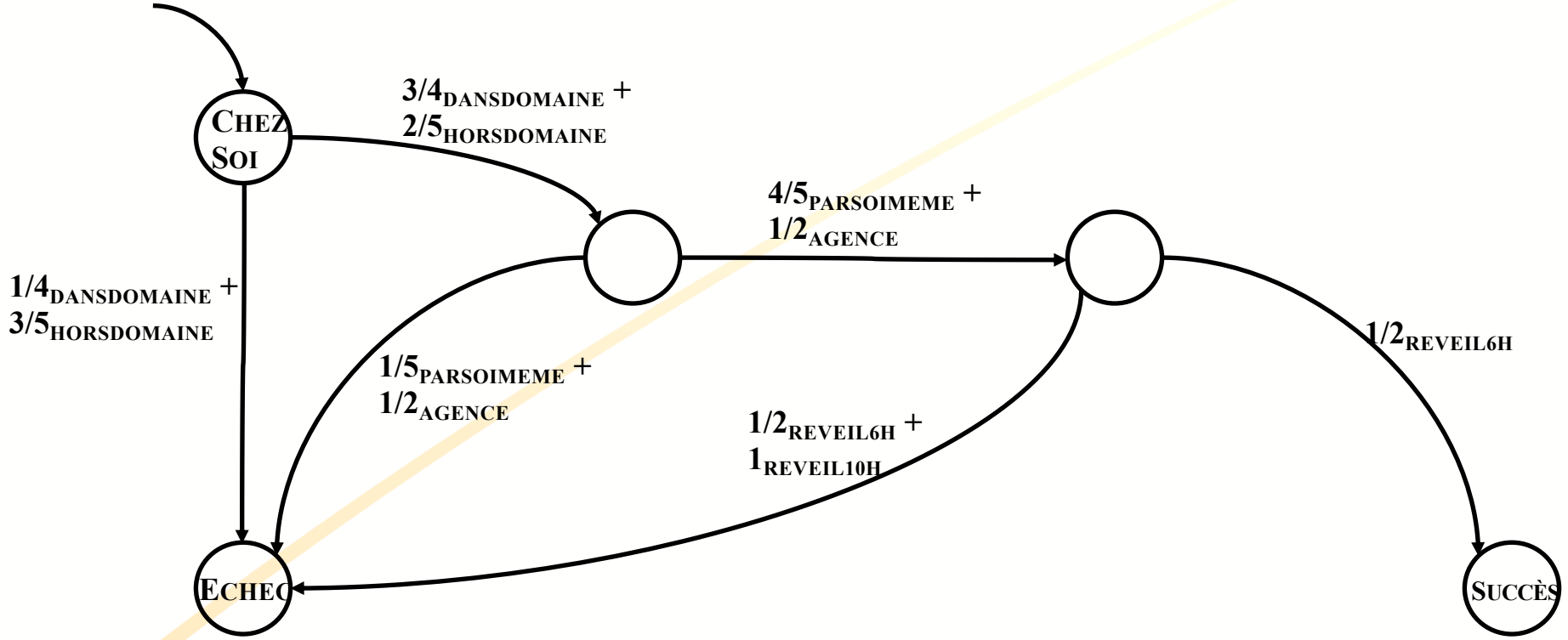
- d'un ensemble fini d'états Q ;
- d'un ensemble fini d'horloges $X = \{x, y, z, \ldots\}$
- d'un alphabet fini $\Sigma = \{a, b, c \ldots\}$;
- d'un ensemble Δ de transitions dont chacune est formée :
  - d'une lettre de Σ ;
  - d'une garde g : une condition sur les horloges : $\bigwedge_{x \in X} x \bowtie c$ $(c \in \mathbb{N}, \bowtie \in \{>, \geq, =, \leq, <\})$;
  - de remises à zéro r : $x := 0, y := 0, \ldots$, parfois noté comme le sous ensemble des horloges remises à zéro : $\{x, y \ldots\}$;
- d'invariants $Inv(q)$ dans les états (gardes) ;
- d'un état initial ;
- d'états finaux ;

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUES

- **Organiser son déplacement à l'école**
  - Choisir une école *dans son domaine* ou *hors de son domaine* ;
  - Réserver un train et un hôtel par *soi-même* ou par *l'agence de voyage* ;
  - Mettre son réveil *à 6h00* ou *à 10h00*.

- **Chacune de ces actions a une probabilité de vous faire apprendre des choses intéressantes.**

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUES



Besoin de modéliser des informations stochastique

- **Un automate probabiliste $A = (Q; E; \{\boldsymbol{P_e}\}_{e \in E}; \boldsymbol{\pi_0}; F)$ est défini par :**
  - Q : un ensemble fini d'états ;
  - E : un alphabet fini d'événement ;
  - Pour tout $e \in E$, $\{\boldsymbol{P_e}\}_{e \in E}$ est une matrice stochastique indicée par Q, i.e. pout tout $q.q' \in Q, \boldsymbol{P_e}[q, q'] \geq 0$ et $\sum_{q' \in Q} \boldsymbol{P_e}[q, q'] = 1$
  - $\boldsymbol{\pi_0}$ la distribution initiale des états ;
  - $F \subseteq Q$, un sous-ensemble d'états finals.

CALCUL DE LA PROBABILITÉ D'UN CHEMIN

- **Définition : Probabilité d'un arc**
  - Un arc est étiqueté par un vecteur de probabilités indicé par E.
  - Ce vecteur est noté comme une somme formelle.

  - Par exemple, l'arc entre *chezsoi* et *echec* est étiquetée par $1/4_{\text{DANSDOMAINE}} + 3/5_{\text{HORSDOMAINE}}$ signifiant que :
    - lorsque **DANSDOMAINE** est choisi en q0, la probabilité que le prochain état soit *echec*, $P_{\text{DansDomaine}}[chezsoi ; echec]$, est égale à 1/4.
    - lorsque **HORSDOMAINE** est choisi en q0, la probabilité que le prochain état soit *echec*, $P_{\text{HorsDomaine}}[chezsoi ; echec]$, est égale à 3/5.

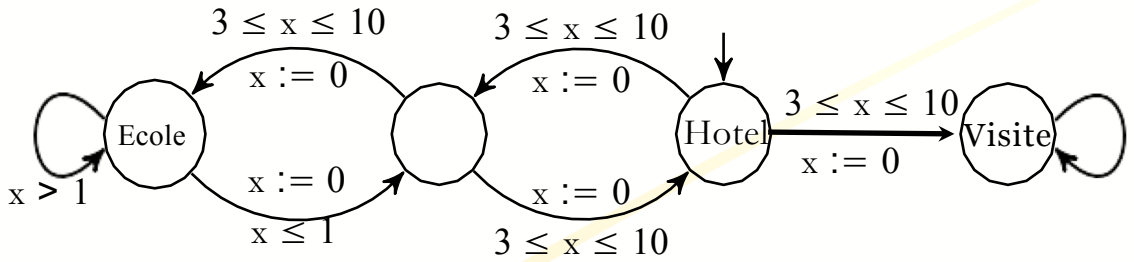
- **Définition : Probabilité d'une stratégie (ou d'un mot)**
  - La probabilité d'acceptation d'un mot w dans un automate A est :
    - $Pr_A(w) = \sum_{q \in Q}(\pi 0[q] \sum_{q' \in F}(\prod_{i=1}^{n} Pa_i))$
  - Par exemple, Quelle est la probabilité de la séquence **DANSDOMAINE, PARSOIMEME, REVEIL6H** ?
    - $Pr_A(w) = 1 * (3/4 * 4/5 * 1/2) = 12/20$

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUE ET TEMPORISÉ

- **Organiser son déplacement vers l'école**
  - Au départ de l'hôtel, je peux aller vers l'école ou partir en visite de Bordeaux;
  - Sur le trajet, je me pose la question « école ou visite ? » entre 3 et 10 min;
  - Si je commence la visite de Bordeaux, je ne fais que ça de mon séjour
  - Si j'arrive à l'école, je me pose la question de partir visiter toutes les heures.

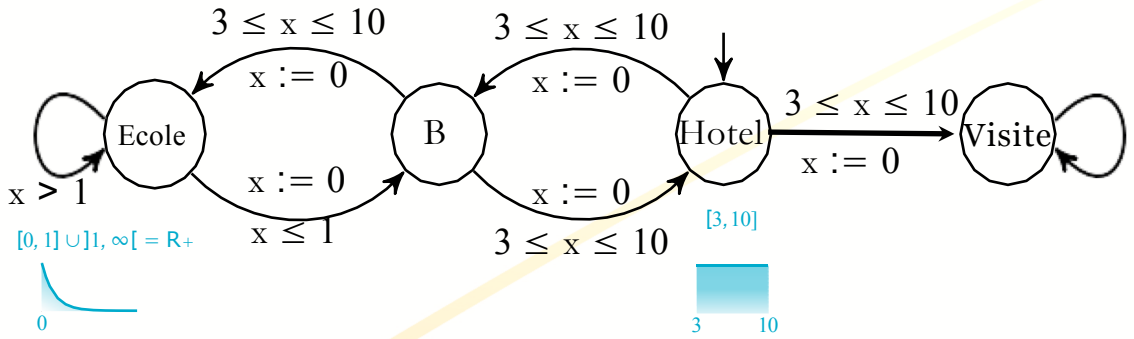
- **Chacune de ces actions a une probabilité de vous faire apprendre des choses intéressantes et le temps pour m'y rendre .**

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUES ET TEMPORISÉES



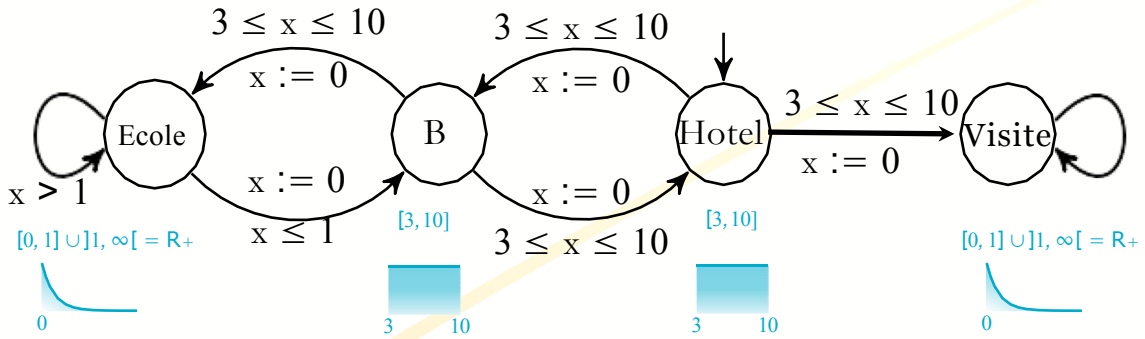
- **Un automate probabiliste $A = (Q; E; X; F, ; (\mu, \rho); q_0)$ est défini par :**
  - $(Q; E; X; F; q_0)$: automate temporisé;
  - μ distribution sur les délais
  - ρ distribution sur les événements

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUES ET TEMPORISÉES



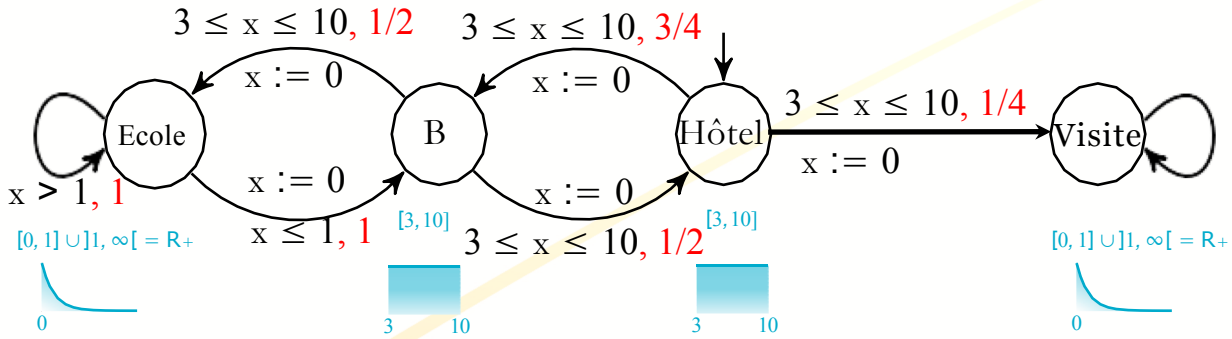
- **Un automate probabiliste $A = (Q; E; X; F, ; (\mu, \rho); q_0)$ est défini par :**
  - $(Q; E; X; F; q_0)$: automate temporisé;
  - μ distribution sur les délais
  - ρ distribution sur les événements

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUES ET TEMPORISÉES



- **Un automate probabiliste $A = (Q; E; X; F, ; (\mu, \rho); q_0)$ est défini par :**
  - $(Q; E; X; F; q_0)$: automate temporisé;
  - μ distribution sur les délais
  - ρ distribution sur les événements

BESOIN DE MODÉLISER DES INFORMATIONS STOCHASTIQUES ET TEMPORISÉES



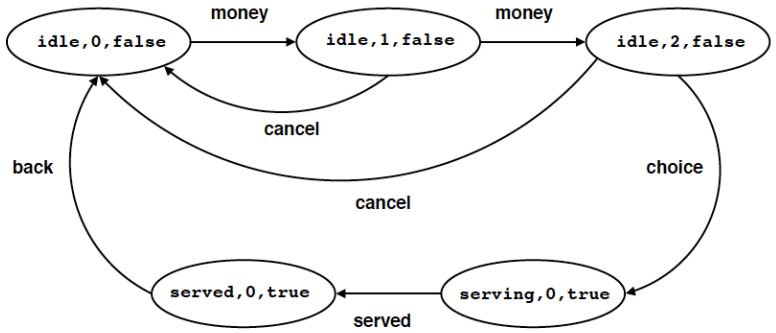
- **Un automate probabiliste $A = (Q; E; X; F, ; (\mu, \rho); q_0)$ est défini par :**
  - $(Q; E; X; F; q_0)$: automate temporisé;
  - μ distribution sur les délais
  - ρ distribution sur les événements

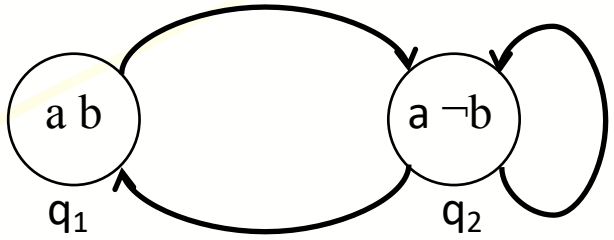
PLAN

- Outil de modélisation
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- **Outil d'évaluation - Vérification**
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- Approche d'évaluation
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- Application sur UppAal SMC
- Couplage Approche robustification et Evaluation Niveau de service
- Application Cplex-UppAal SMC

MODEL-CHECKING : OBJECTIF

- On dispose d'un modèle comportemental d'un système





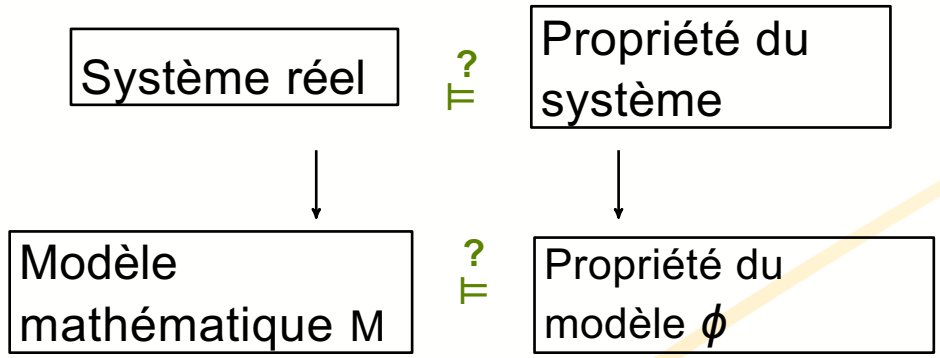
- On souhaite évaluer le respect par ce système d'un certain nombre de propriétés :

  - Accessibilité *Une certaine situation peut être atteinte*
    x peut valoir 0, toute instruction peut être exécutée
  - Invariance *Chaque état local respecte une bonne propriété*
    x ne vaut jamais 0, le tableau ne déborde jamais
  - Sûreté *Quelquechose de mauvais n'arrive jamais*
    j'accède au fichier uniquement si j'ai entré le bon PIN
  - Vivacité *Quelquechose de bon finit par arriver*
    le programme termine, le message finit toujours par être transmis
    le programme revient toujours à l'état initial
  - Équité *Quelquechose de bon se répète infiniment souvent*
    si un processus demande toujours la main, il l'aura infiniment
  - Équivalence comportementale *Est-ce que 2 systèmes sont équivalents ?* système simple de référence VS système optimisé

- **Deux approches :**
  - Simulation
  - Vérification formelle de propriétés

MODEL-CHECKING : PRINCIPES

Principes

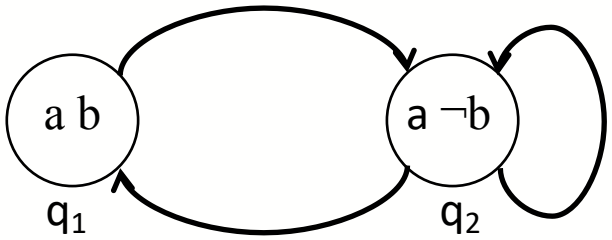


Question : Est-ce que le modèle M vérifie la propriété ϕ ?
Réponse : oui ou non

MODEL-CHECKING : STRUCTURE DE KRIPKE

Objectif :

- Se ramener à une structure de Kripke où toute l'information est portée par les états.
  - $M = < Q, \rightarrow, AP, L, Q_0 >$
    - Q : ensemble fini des états
    - $\rightarrow \subseteq Q \times Q$: relation de transition (relation totale)
    - AP: ensemble de propositions logiques atomiques
    - $L : Q \rightarrow 2^{AP}$ fonction qui étiquette un état avec un ensemble de propositions atomiques vraies dans cet état
    - Q_0 : état initial

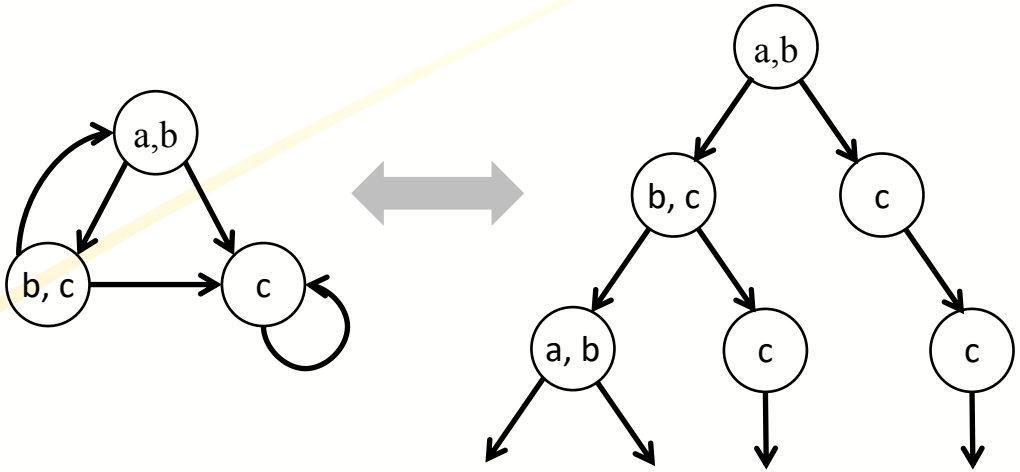


- La technique de transformation dépend du formalisme de départ
- Le passage par la structure de Kripke n'est pas la seule technique (produits d'automates de Büchi par exemple)

MODEL-CHECKING : STRUCTURE DE KRIPKE

A partir de la structure de Kripke …

Dépliage : on prend un état de la structure de Kripke que l'on définit comme la racine d'un arbre qui représente toutes les traces possibles d'exécution

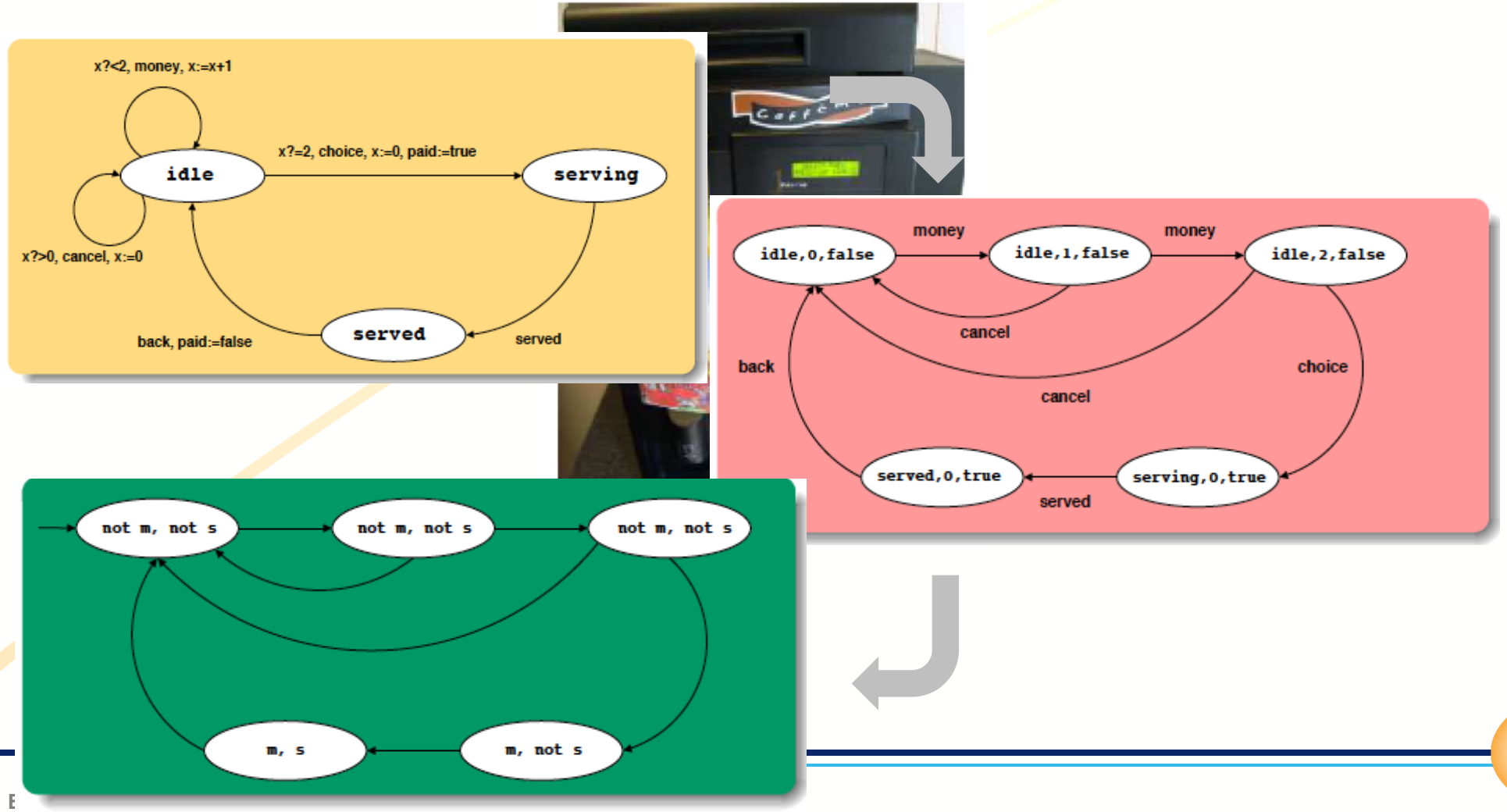


- Exploration exhaustive de l'ensemble des états atteignables par le modèle afin de vérifier qu'une propriété est satisfaite
- Problème d'explosion combinatoire

MODEL-CHECKING : STRUCTURE DE KRIPKE

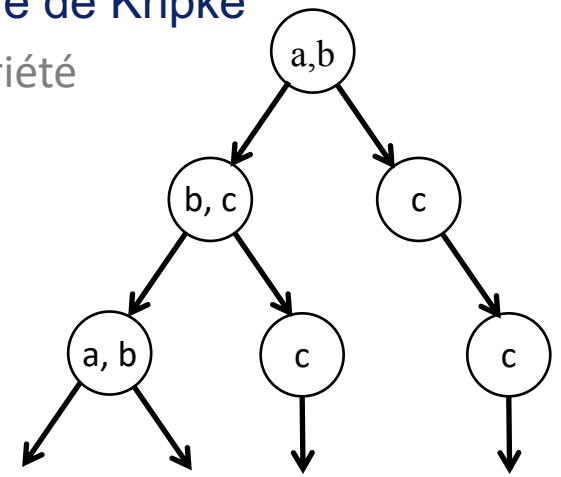
Exemple de transformation vers une structure de Kripke

Machine d'état → système de transition → structure de Kripke



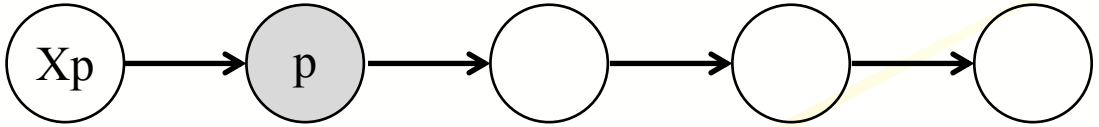
Model-checking : Formalisation des propriétés

- **Les propriétés sont modélisées en logique temporelle**
- **Logique temporelle**
  - Logique classique + Opérateurs dédiés au temps
    - Connecteurs temporels :
      - Suite d'événements attendus le long d'un seul chemin partant de l'état initial
      - X (suivant), F (un jour), G (toujours), U (jusqu'à)
    - Quantification des chemins : concernent un dépliage de la structure de Kripke
      - Quantifie les chemins partant d'un état qui doivent vérifier la propriété
      - A (pour tous les chemins), E (il existe un chemin)

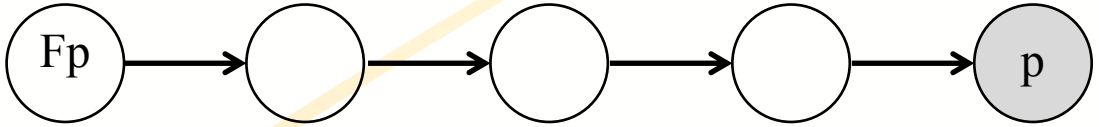


MODEL-CHECKING : CONNECTEURS TEMPORELS

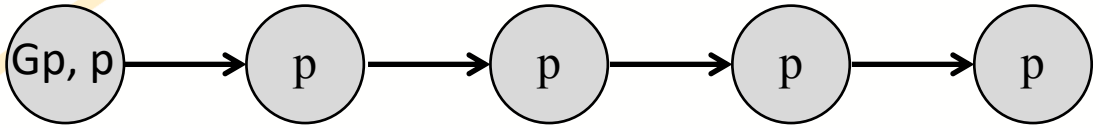
Opérateur NEXT (X)



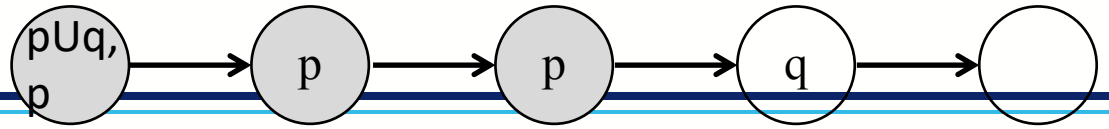
Opérateur SOMETIMES IN THE FUTURE (F)



Opérateur ALWAYS (G)



Opérateur UNTIL (U)



Exemples

Atteignabilité
F(x = 0)

Invariance
G¬(x = 0)

Vivacité
G(p → **F**q)

MODEL-CHECKING : QUANTIFICATEURS DE CHEMINS

- **Les connecteurs temporels ne considèrent qu'un seul chemin à la fois**
- **On veut pouvoir parler de tous les futurs possibles selon les choix d'actions (branchement)**
- **Organisation en arbre**

- **Quantificateurs de chemins**
  - **A** : tous les chemins futurs vérifient la propriété
  - **E** : il existe un chemin futur qui vérifie la propriété

- **On combine les connecteurs temporels et les quantificateurs de chemins**

Exemples

Atteignabilité
EF(x = 0)

Invariance
AG¬(x = 0)

Vivacité
AG(p → **F**q)

MODEL-CHECKING : FORMALISATION DES PROPRIÉTÉS

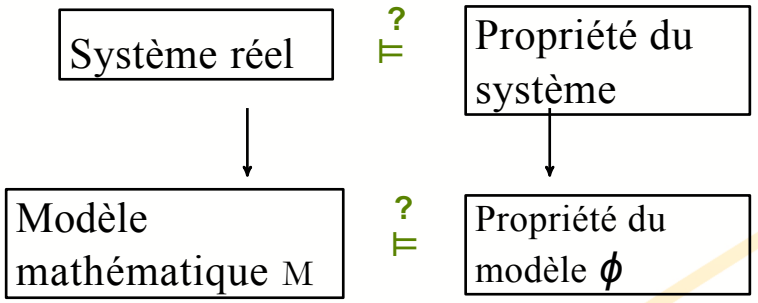
- **Plusieurs logiques selon les branchements autorisés**
  - LTL : Linear Temporal Logic
    - Formule de la forme : **A** *f* (f formule sur un chemin)
    - Pas d'utilisation de **E**
    - Exemples : **A**(**FG***p*) signifie que pour tous les chemins, il existe un état après lequel p sera toujours vrai, **EF**(**AG***p*) pas LTL
  - CTL : Computational Tree Logic
    - Pas de restriction sur les opérateurs **A**, **E**
    - **X**, **F**, **G**, **U** doivent toujours être précédés par un quantificateur de chemin **A** ou **E**.
    - Exemples : **EF**(**AG***p*) signifie qu'il existe un chemin pour lequel il existe un état à partir duquel pour tous les chemins sortant et pour tous les états successeurs, p est vrai. Cette formule n'appartient pas à LTL
    - Restriction de CTL*
  - CTL* : Computational Tree Logic *
    - Aucune restriction sur tous les opérateurs vus (connecteurs et quantificateurs)

Model-checking : Processus de model-checking

1. Modéliser le système par $\mathcal{M}$ et la propriété par φ
2. $\mathcal{M} \models \varphi$? Si non, contre-exemple σ.
3. Analyser les résultats
   - Si oui, **ok**. Attention au lien avec système réel
   - Si non, rejouer σ sur système réel.
     - Si vrai bug, alors **erreur**.
     - Sinon goto (1) en raffinant $\mathcal{M}$ grâce à σ.

PRINCIPES DU MODEL CHECKING PROBABILISTE

Principes



Question : Quelle est la probabilité que le modèle M vérifie la propriété ϕ ?

Réponse : probabilité d'avoir oui à la propriété

VÉRIFICATION PROBABILISTE

- **Comme pour la vérification classique :**
  - Basée sur les traces
  - Propriétés sur les états ou sur les chemins
  - Basée sur l'exploration

- **Mais**
  - Mesure sur l'ensemble des chemins
  - Qualifications
    - Qualitatives : $\forall \rightarrow P_A = 1$ ou $\exists \rightarrow P_A > 0$
    - Quantitatives : valeur P_A

- **2 manières de calculer : probabiliste ou statistique**

VÉRIFICATION PROBABILISTE

- **Probabilité par rapport à un état :**
  - Quelle est la probabilité que la propriété soit vraie ?
  - Quelle est la probabilité que la propriété soit égale à une valeur *a* ?
  - Quelle est la probabilité d'être dans l'état 1 et 2?
  - Quelle est la probabilité de ne pas être dans l'état 1?
  - …
- **Probabilité d'un chemin**
  - Quelle est la probabilité d'avoir un chemin ?
  - Quelle est la probabilité que l'état suivant soit l'état 1?
  - Quelle est la probabilité que tant que la propriété q1 n'est pas vraie, la propriété q2 est vérifiée?
  - Quelle est la probabilité que tant que la propriété q1 n'est pas vraie, la propriété q2 est vérifiée, en un nombre de transition donné?

DÉTERMINATION DE LA PROBABILITÉ

- Calcul de $Pr_A(w) = \sum_{q \in Q}(\pi 0[q] \sum_{q' \in F}(\prod_{i=1}^{n} Pa_i))$
- Probabilité d'être dans un état au pas suivant : Next
  - $Pr_A(N(q) \vDash p) = \sum_{q \in Q}(\pi 0[q] \sum_{q' \in F}(P_e))$
- **Probabilité que tant que la propriété q1 n'est pas vraie, la propriété q2 est vérifiée**
  - Identifier les états où la probabilité d'évolution est 0 ou 1, identifier ces états Q0 et Q1 et les états d'intèret Q'
  - Détermination de la matrice réduire $A = (\Pr(q, q'))_{q,q' \in Q'}$
  - Déterminer le vecteur b d'être dans l'état vérifiant la propriété en un seul pas : $b = (\Pr(q, q'))_{q,q' \in Q1'}$
  - Déduire $\Pr(q \vDash \varphi_1 U \varphi_2) =$ x(s) où x est le point fixe de $\Gamma(y) = A.y + b$

OUTILS

- PRISM [Kwiatkowska et al., 2011]
- UppAal [Larsen et al., 1997]
- MRMC [Katoen et al., 2005]
- SPIN [Holzmann, 1997]

VÉRIFICATION PROBABILISTE : LIMITES

- **Le model-checking probabiliste a les mêmes limites que le model-checking déterministe :**
  - Indécidabilité pour les modèles complexes (hybrides, paramétrés…)
  - Explosion combinatoire
  - Ressources de calcul importante en mémoire et en temps

- **Solution : un méthode approchée : model-checking statistique**
  - Efficace
  - Précision contrôlée
  - Non limitées par la logiques ou les modèles

VÉRIFICATION STATISTIQUE : PRINCIPE

- **Estimer la probabilité qu'un modèle M satisfasse une propriété p à partir d'échantillons**

- **Echantillons : Traces d'une exécution du modèle**
- **Probabilité sur l'espace d'échantillon (borné, observables)**
- **Applicable sur des modèles purement probabiliste**

- **Fonctionnement :**
  - Utiliser le modèle pour générer un échantillon de trace (estimateur de la mesure réelle)
  - Mesurer la probabilité de satisfaire la propriété sur l'échantillon
  - Généraliser la probabilité au modèle initial : Précision, taux d'erreur en fonction du nombre d'échantillons

- **2 types principaux : Propriétés qualitatives ou quantitatives**

VÉRIFICATION STATISTIQUE : PROPRIÉTÉ QUANTITATIVE

- **Données d'entrée : Modèle M, propriété p, précision et taux d'erreur**
- **Objectif : estimer la probabilité avec laquelle M satisfait p**

- **Monte Carlo [Robert, 2004]**
  - Générer une simulation de M et vérifier si elle satisfait p
  - Variable aléatoire Z, réalisation z_i = 1 si la $i^{ème}$ expérience est un succès
    - Variable de bernouilli : $\lim_{N\to\infty} \frac{\sum_{i=1}^{N} z_i}{N} = \gamma$
    - Précision et taux d'erreur pour connaitre le nombre de simulation
      - Bornes de Cherno-Hoeding [Hoeding, 1963] : $N \geq \frac{\ln(2)-\ln\gamma}{(2*\varepsilon)^2}$
      - $\Pr(|\gamma N - \gamma| \geq \varepsilon) \leq \delta$

- **Exemple Précision $(\varepsilon = 0{,}01)$ et erreur $\delta = 0,01$, il faut 13246 simulations**

VÉRIFICATION STATISTIQUE : PROPRIÉTÉ QUALITATIF

- **Objectif : Soit γ la probabilité avec laquelle M satisfait la propriété (non connue), comparer avec une borne θ sans calculer γ**
- **Confronter 2 hypothèses : H : $\gamma \geq \theta$ et k : $\gamma < \theta$**
- **Déterminer les bornes d'erreur (α, β):**
  - α : borne sur l'erreur de type I (K acceptée alors que H est vraie)
  - β : borne sur l'erreur de type II (H acceptée alors que K est vraie)

- **Impossible d'optimiser les 2 bornes en même temps**

- **Afin de garantir des valeurs faibles pour α et β, on utilise une région d'indifférence $[\gamma_1, \gamma_0]$ qui contient θ.**
  - $H_0 : \gamma \geq \gamma_0$ VS $H_1 : \gamma < \gamma_1$,
  - Si $\gamma \in [\gamma_1, \gamma_0]$ alors le résultat n'a pas d'importance.
  - En général, la précision est fixée ε et on prend $\gamma_1 = \theta - \varepsilon$ et $\gamma_0 = \theta + \varepsilon$.

VÉRIFICATION STATISTIQUE : PROPRIÉTÉ QUALITATIF - RÉSOLUTION

- **Méthode "statiques" : le nombre d'échantillons est pré-calculé en fonction des paramètres α, β, ε**
  - Single Sampling Plan → (n, c) tel que $H0$ est accepté si au moins c échantillons sur n satisfont ϕ
- **Méthodes "dynamiques" : le nombre d'échantillons s'adapte au fur et à mesure des simulations. Définition d'un critère d'arrêt en fonction des paramètres**
  - Sequential Probability Ratio Test

OUTILS

- UppAal SMC (Networks of PTA)
- PRISM (DTMC, CTMC, PTA)
- Ymer (Generalized Semi-Markov Processes)
- COSMOS (Linear Hybrid Automata)
- VESTA (DTMC, CTMC)
- PLASMA-LAB (Générique)

Model-checking probabiliste : Comparaison

- **Approche numérique :**
  - Résultats précis
  - Implémentation efficace pour une grande variété de modèles et de logiques
  - Consommation de mémoire proportionnelle au nombre de transition du système

- **Approche statistique**
  - Espace mémoire nécessaire très faible
  - Hypothèses plus faibles sur les modèles et les lois de probabilités utilisées
  - Ne calcule qu'un encadrement probabiliste du résultat
  - Inefficace pour de très faibles probabilités

PLAN

- Outil de modélisation
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- Outil d'évaluation - Vérification
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- **Approche d'évaluation**
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- Application sur UppAal SMC
- Couplage Approche robustification et Evaluation Niveau de service
- Application Cplex-UppAal SMC

PERTURBATIONS: DÉFINITION

- **Identification:**
  - Plusieurs types de perturbations:

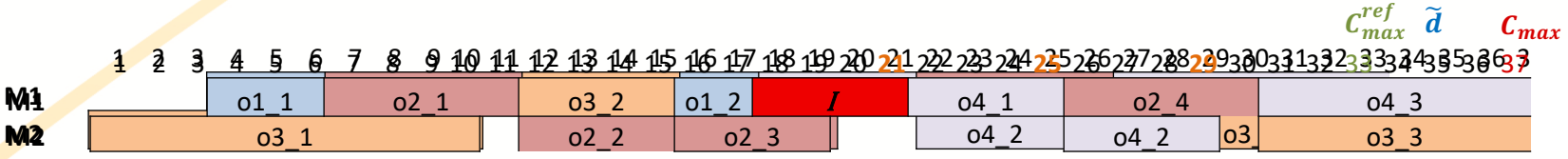
	Incertitude	Aléa
Système	Durée d'exécution d'opérations, Durée de réparation, …	Panne machine, Produit défectueux, …
Environnementale	Demande client, Durée de livraison, d'approvisionnement, …	Rupture d'approvisionnement, Arrivée de commande urgente, …

- **Modélisation:**
  - Stochastique: les perturbations sont des variables aléatoires qui suivent des distributions de probabilité définies.
  - ➔ Dans l'objectif d'utiliser les données d'ateliers et la connaissance existantes sur les perturbations

ROBUSTESSE EN ORDONNANCEMENT: COMMENT L'ÉVALUER?

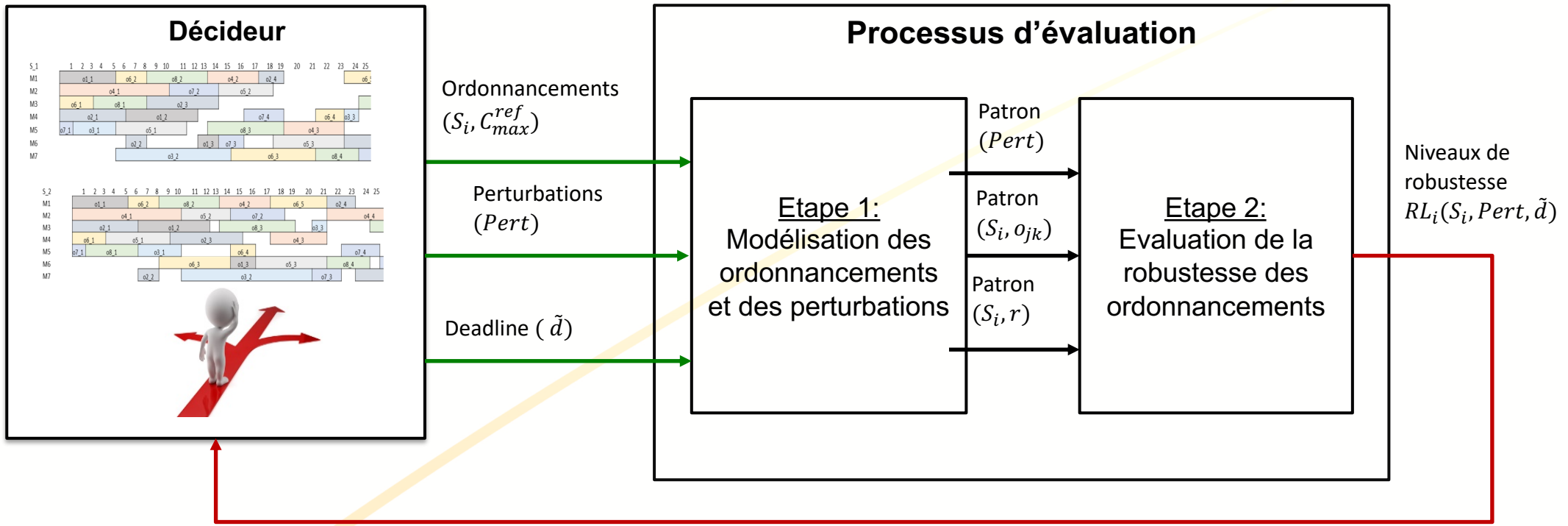
- **Définition:**
  - Une solution est dite robuste si sa performance est peu sensible aux perturbations (Billaut et al., 2008).
- **Mesure de robustesse:**
  - Niveau de service ou niveau de robustesse (Dauzère-Pérès et al.,2008): « La probabilité qu'un critère est inférieur (ou supérieur) à une valeur donnée »
    - Le critère ici est la durée totale de l'ordonnancement (Makespan) C_{max}
    - La valeur donnée est ici une deadline qui doit être respectée $\tilde{d}$.
  - Formellement:

$$RL_i = \Pr\left(C_{max}(S_i, Pert) \leq \tilde{d}\right)$$



  - Cette métrique mesure la probabilité Pr que le Makespan $C_{max}(S_i, Pert)$ d'un ordonnancement S exposée à des perturbations I soit inférieur ou égal à une deadline définie $\tilde{d}$.

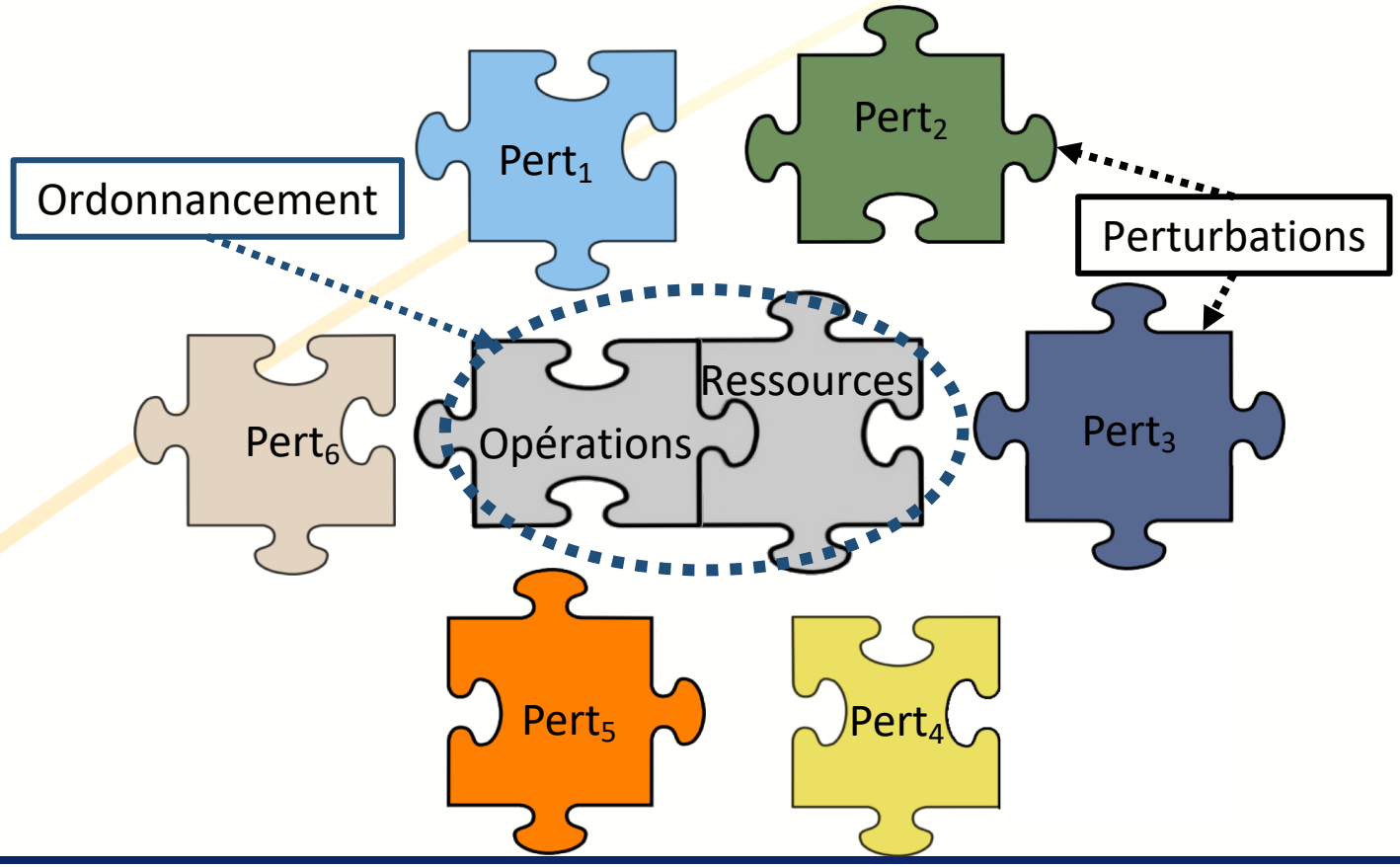
APPROCHE PROPOSÉE



- **Hypothèses :**
  - Pert = incertitudes;
  - Données probabilistes fournies par le décideur ;

MODÉLISATION DU PROBLÈME

- **Approche de modélisation:**
  - Modulable,
  - Modèles communicants,
  - Modèles basés sur des patrons instanciés à la taille et le type du problème traité.



MODÈLE ORDONNANCEMENT

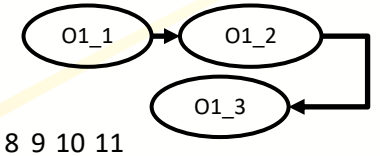
Patron d'opération (o_{jk}):

Comportement de l'opération durant l'exécution de l'ordonnancement.

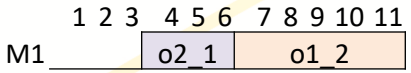
- Attente de l'initialisation des incertitudes
- *Mise à jour de la durée d'exécution de l'opération.*

- Contraintes de précédence

Sur la gamme produit

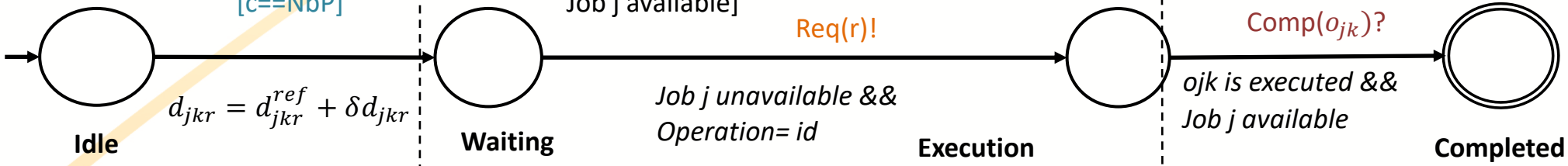


Sur la séquence machine



- Envoi d'une requête
- *Mise à jour de la disponibilité du job et mémorisation de l'opération*

Attente de l'événement de fin d'exécution



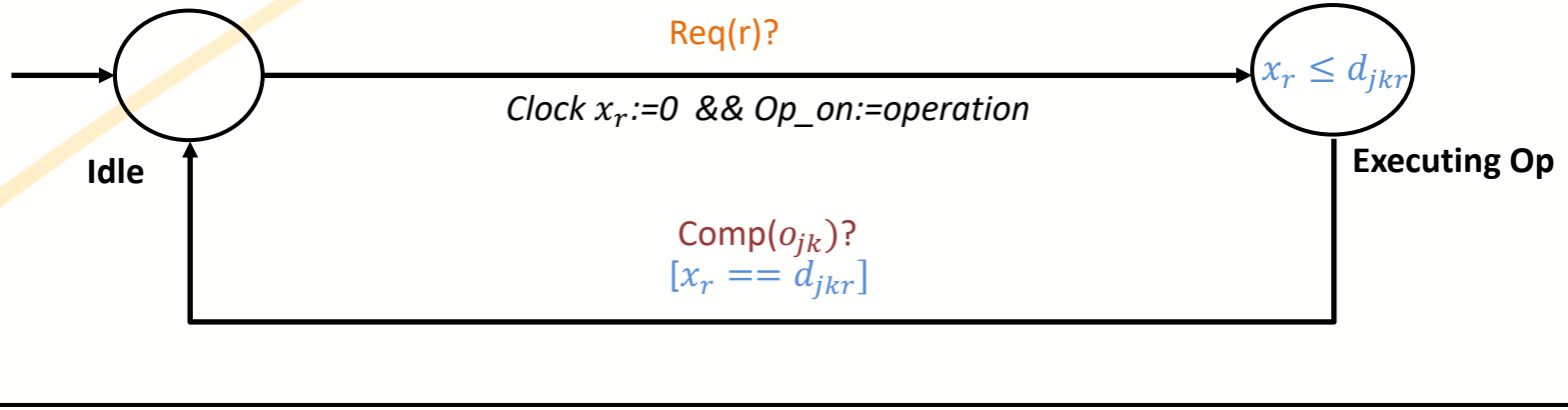
MODÈLE ORDONNANCEMENT

Patron de la ressource (r):

Comportement de la ressource en exécutant une opération

Comportement :

- Attente de la requête *et mise à jour de l'horloge et de l'opération en cours*
- Exécution de la durée d_{jkr},
- Envoi d'un message de fin d'exécution



MODÈLES DE PERTURBATIONS

- **Prise en compte d'une incertitude:**
  - Hypothèse: Toute incertitude génère un retard sur la durée d'exécution d'une opération
  - Chaque durée d'exécution d_{jkr} est composée par :
    - Une durée de référence: d_{jkr}^{ref}
    - Une fluctuation généré par l'incertitude δd_{jkr}
  - Expression de la durée :

$$d_{jkr} = d_{jkr}^{ref} + \delta d_{jkr}$$

  - Nature de la fluctuation :
    - Variable aléatoire suivant une distribution de probabilité connue et évoluant dans un intervalle $[\delta d_{jkr}^{-}, \delta d_{jkr}^{+}]$.

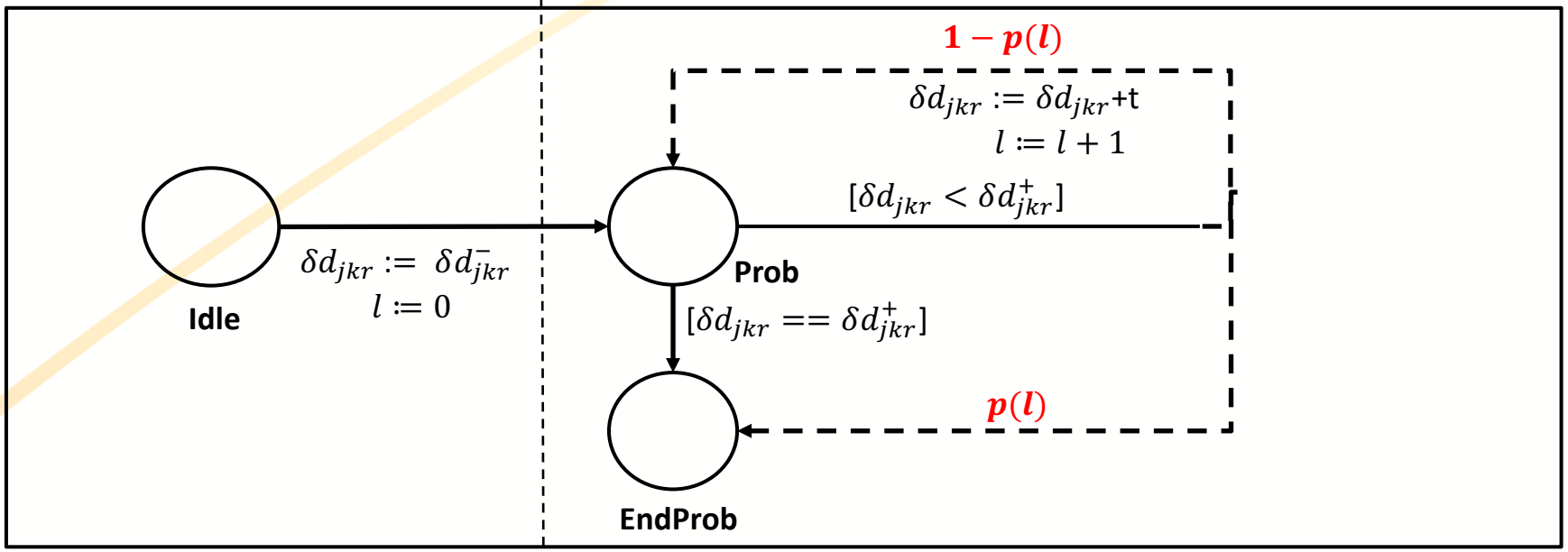
MODÈLE DE PERTURBATIONS

Patron stochastique d'incertitude (Pert):

- Mesurer la fluctuation de la durée d'exécution

- *Mise à jour de la fluctuation à sa borne minimale*
- *Mise à jour du paramètre d'itération*

- Choix probabilistes:
  - p(l) : δd_{jkr} garde sa dernière valeur
  - 1-p(l): δd_{jkr} est incrémentée d'un pas de discrétisation t et le paramètre est mis à jour.



MODÈLE DE PERTURBATIONS

- **Calcul probabiliste de p(l): dans le cas d'une incertitude sur la durée d'exécution**
  - Proposition:

$$\begin{cases} p(0) = \dfrac{F_X(1) - F_X(0)}{F_X\left(d_{jkr}^{max}\right) - F_X\left(d_{jkr}^{min}\right)} & pour\ l = 0 \\ p(l) = \dfrac{F_X(l+1) - F_X(l)}{F_X(l) - F_X(l-1)} . \dfrac{p(l-1)}{1 - p(l-1)} & pour\ l \geq 1 \end{cases}$$

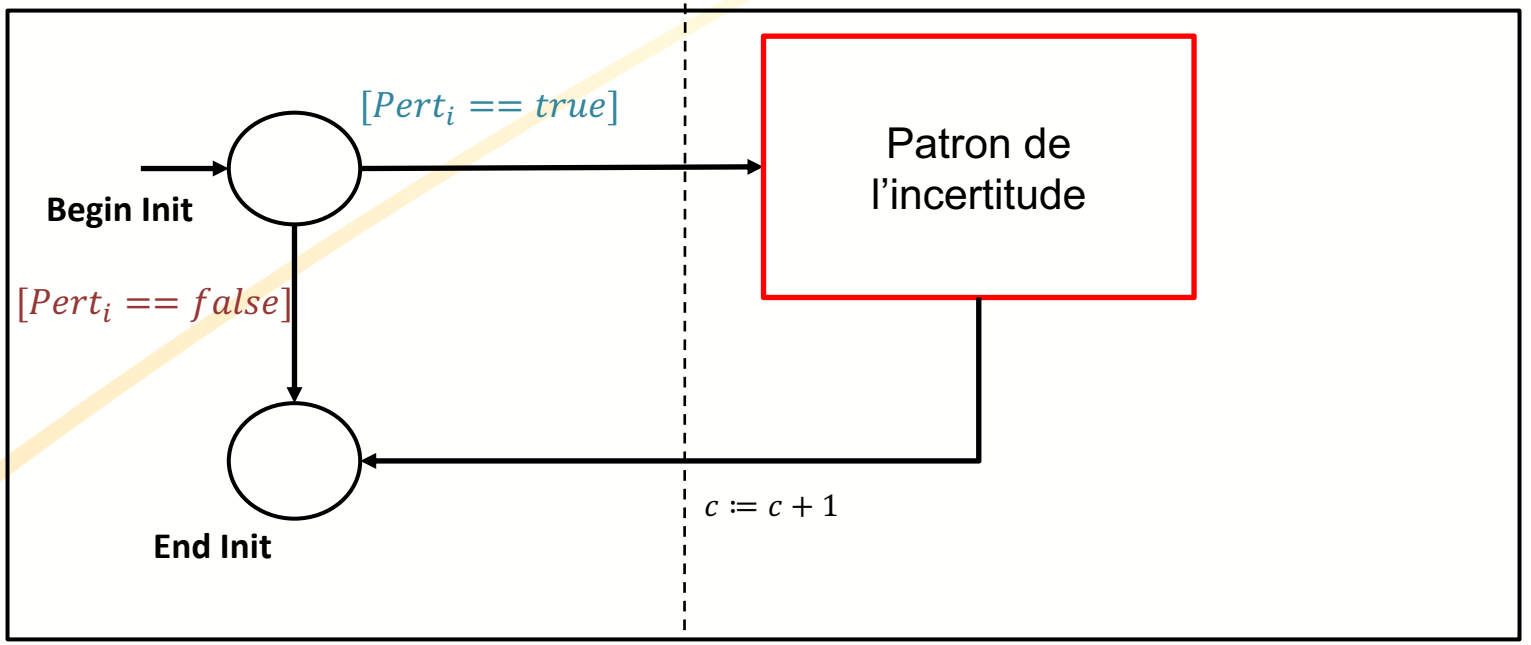
  - Exemple: distribution exponentielle tronquée dans $[d_{jkr}^{min} = d_{jkr}^{ref} + \delta d_{jkr}^{-},\ d_{jkr}^{max} = d_{jkr}^{ref} + \delta d_{jkr}^{+}]$ avec : $\lambda = {}^{1}/_{d_{jkr}^{ref}}$
    - Pour $l = 0$: $\mathrm{p}(0) = \dfrac{e^{-\lambda d_{jkr}^{min}}\left(1 - e^{-\lambda}\right)}{e^{-\lambda d_{jkr}^{min}} - e^{-\lambda\left(d_{jkr}^{max}+1\right)}}$
    - Pour $l = 1$: $\mathrm{p}(1) = e^{-\lambda} . \dfrac{p(0)}{1-p(0)}$ après simplification avec $F_X(x) = 1 - e^{-\lambda x}$
  - ***Avantage :*** *Quelque soit la distribution considérée, tant que F_X est connue, le calcul de $p(l)$ est possible*

MODÈLES DE PERTURBATIONS

- **Modèle d'initialisation des incertitudes:**
  - Initialiser la prise en compte de l'incertitude.

Deux possibilités:
- Incertitude prise en compte
- Incertitude pas prise en compte

Si l'incertitude est considérée:
- Patron de la perturbation exécuté;
- *Mise à jour du nombre de patrons d'incertitudes exécutés*



EVALUATION

- **L'objectif d'évaluation:**
  - Probabilité RL_i est calculé pour chaque S_i face à une panne machine aléatoire.
  - ➔ Mesure de la robustesse des ordonnancements Pr_i

- **Evaluation par vérification statistique:**
  - Propriété à vérifier en langage PCTL [Baier et al, 1998]:

$$\Pr\left(C_{max}(S_i, Pert) \leq \tilde{d}\right) \rightarrow P =? [F \leq \tilde{d} \text{ "All operations } o_{jk} \text{ are Completed"}]$$

  - Quelle est la probabilité que toutes les opérations atteignent leur localité marquée **Completed** avec une durée inférieure ou égale à une deadline donnée $\tilde{d}$

PLAN

- Outil de modélisation
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- Outil d'évaluation - Vérification
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- Approche d'évaluation
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- **Application sur UppAal SMC**
- Couplage Approche robustification et Evaluation Niveau de service
- Application Cplex-UppAal SMC

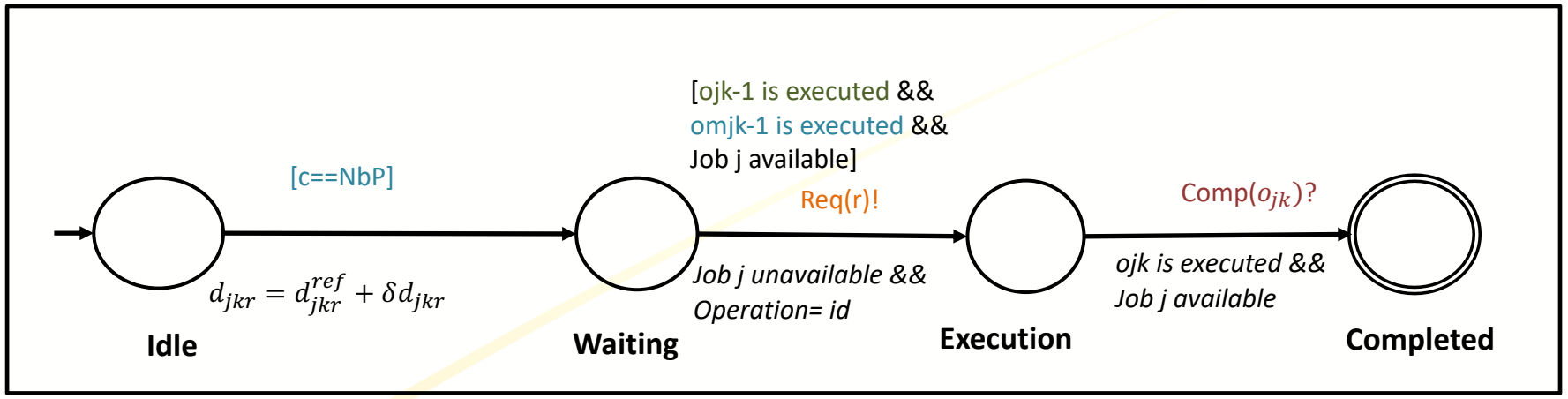
APPLICATION

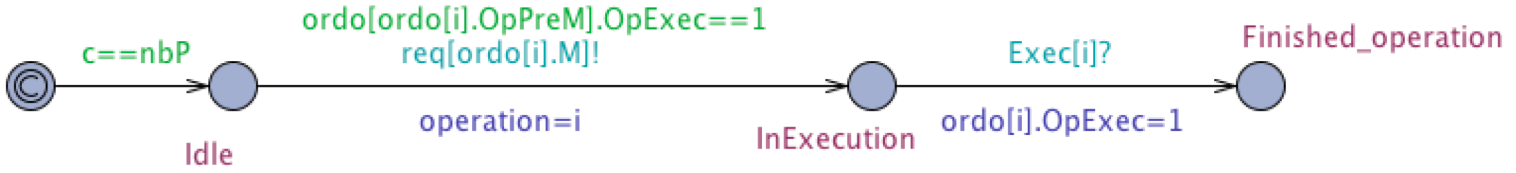
- **Atelier de type : machines parallèles M1, M2**
- **10 opérations :**
  - Durées sur chaque machine : p=[[1,2][2,1][5,2][3,4][8,6][2,5][4,3][2,4][4,2][1,1]];
  - Intervalle de perturbation: $\hat{p}$=[[0,1][1,0][3,1][2,2][5,3][1,1][2,2][1,3][1,1][0,0]];

- **Ordonnancement à évaluer :**
  - Cmax = 12;
  - x = [[1 0][0 1][0 1][1 0][0 1] [1 0][1 0][1 0][0 1][0 1]];

- **Niveau de service attendu par le décideur : 90%**

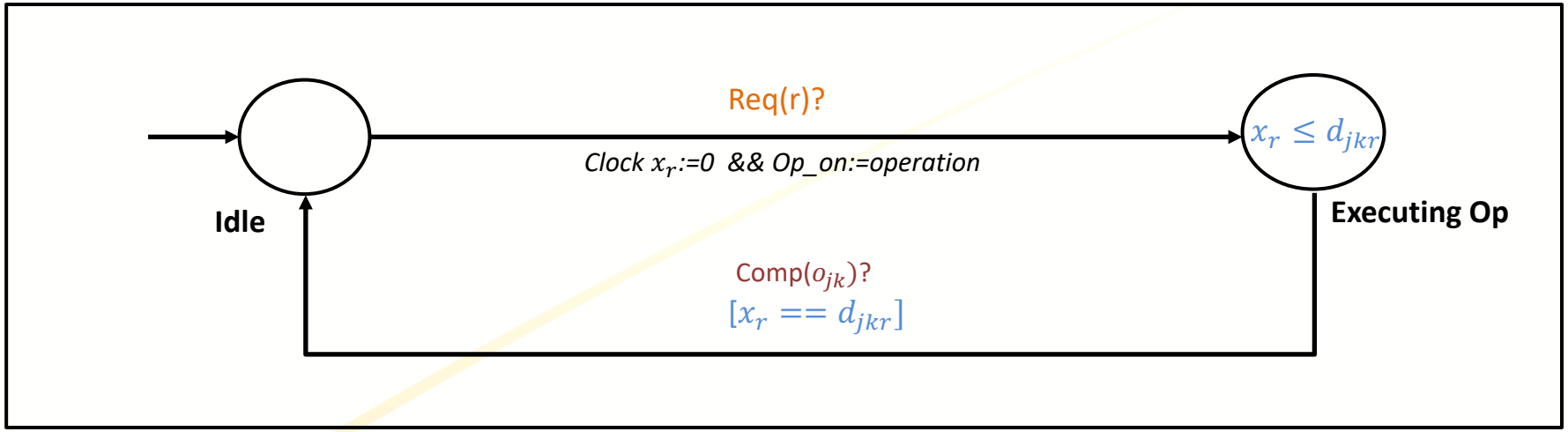
MODÈLE ORDONNANCEMENT





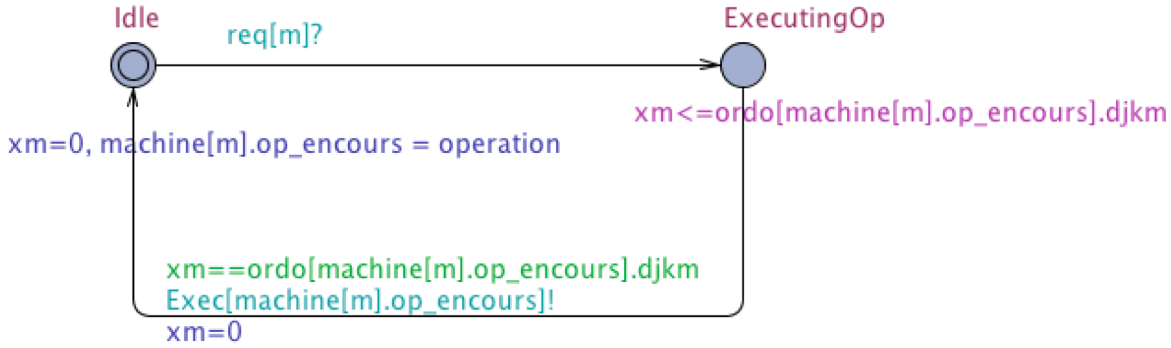

```
const int NbOp = 11;
typedef meta int[0,NbOp-1] id_OpOrdo;
struct{int Pr, Op, djkm; bool OpExec; int OpPre,OpPreM, M;} ordo[11]= {
{ 0, /*0*/ 0,  0,        true,    0, 0, 0 },
{ 1, /*1*/ 1, 0,        false,    0, 0, 1 },
{ 2, /*2*/ 2, 0,        false,    0, 0, 2 },
{ 3, /*3*/ 3, 0,        false,    0, 2, 2 },
{ 4, /*4*/ 4, 0,        false,    0, 1, 1 },
{ 5, /*5*/ 5, 0,        false,    0, 3, 2 },
{ 6, /*6*/ 6, 0,        false,    0, 4, 1 },
{ 7, /*7*/ 7, 0,        false,    0, 6, 1 },
{ 8, /*8*/ 8, 0,        false,    0, 7, 1 },
{ 9, /*9*/ 9, 0,        false,    0, 5, 2 },
{ 10, /*10*/ 10, 0,     false,    0, 8, 1}
};
```

MODÈLE ORDONNANCEMENT

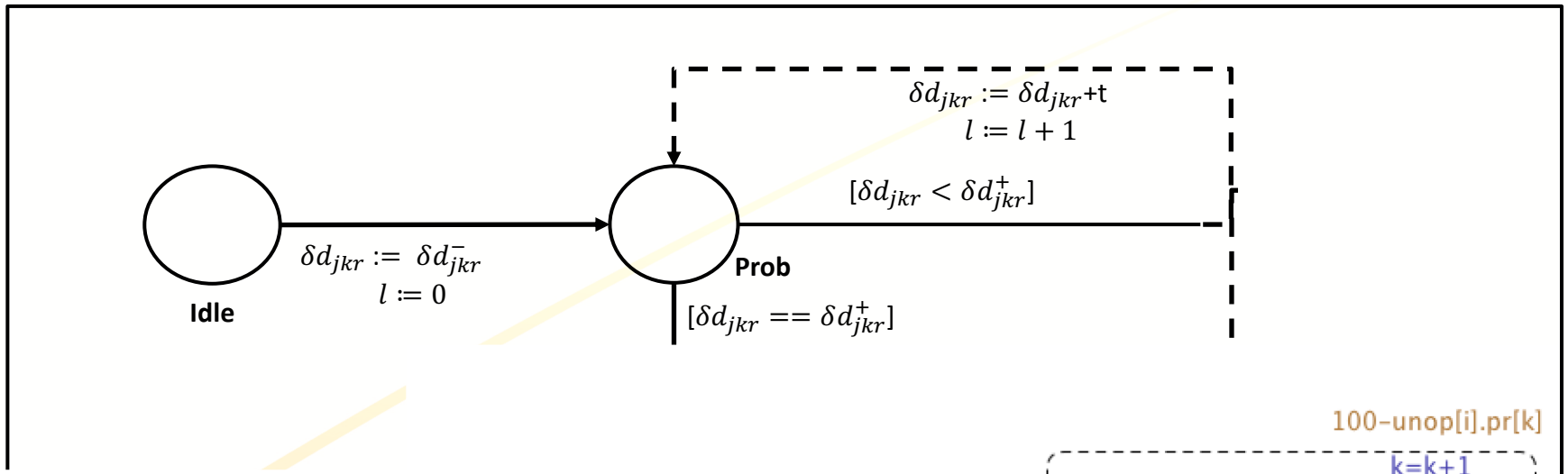



```
const int Nm = 4;
typedef meta int[0,Nm-1] id_M;
struct { int id; int op_encours;} machine[4] = {
{ 0,  -1},
{ 1,  -1},
{ 2,  -1},
{3,   -1}

};
```

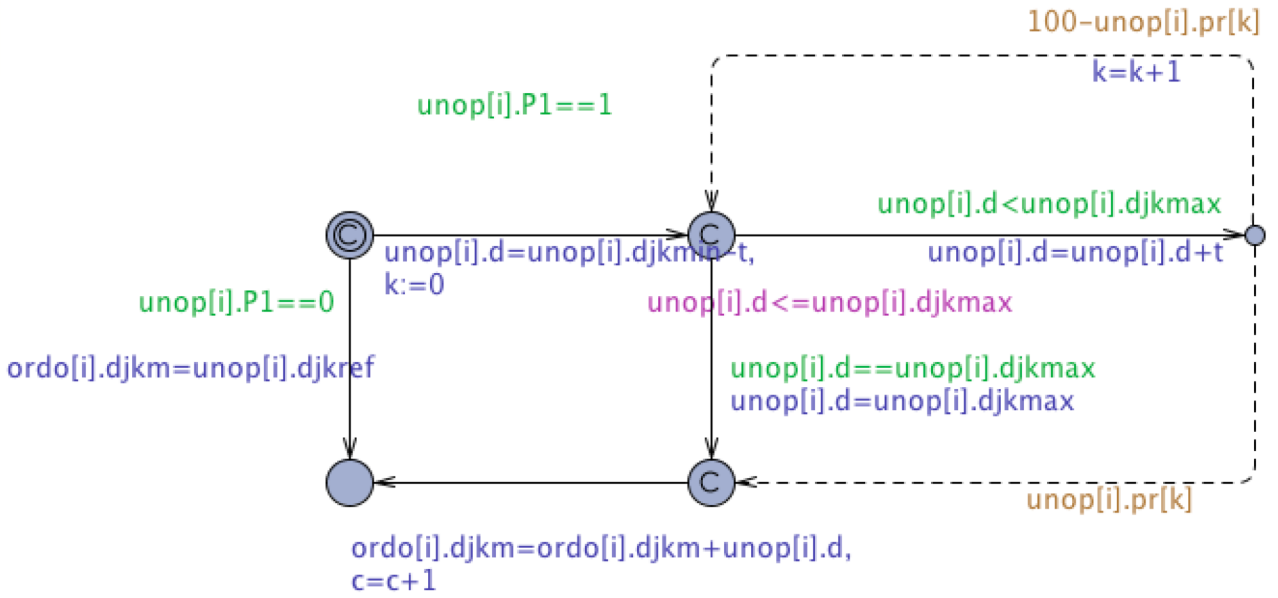


MODÈLE DE PERTURBATIONS




```
// P1= uncertitude sur les durées d'exécutions
//const int NbOp = 7;
typedef meta int[0,NbOp-1] id_Op;
struct{int Op,djkref, djkmin,djkmax,d; bool  P1; int pr[11];} unop[11]= {
{0, 0,   0,0,    0,     false,   {100,0,0,0,0,0,0,0,0,0,0}  },

//Opération 1
//Operation 1 sur Machine1 avec perturbation
//{1, 10, 10,10,  0,     true,   {100,0,0,0,0,0,0,0,0,0,0}  },
//Operation 1 sur Machine2 avec perturbation
//{1, 20, 10,30,  0,     true,   {51,62,100,0,0,0,0,0,0,0,0}  },
//Operation 1 sur Machine 1 sans perturbation
{1, 10, 10,10,  0,     true,  {100,0,0,0,0,0,0,0,0,0,0}  },
//Operation 1 sur Machine 2 sans perturbation
//{1, 20, 20,20,  0,     true,   {100,0,0,0,0,0,0,0,0,0,0}  },
```



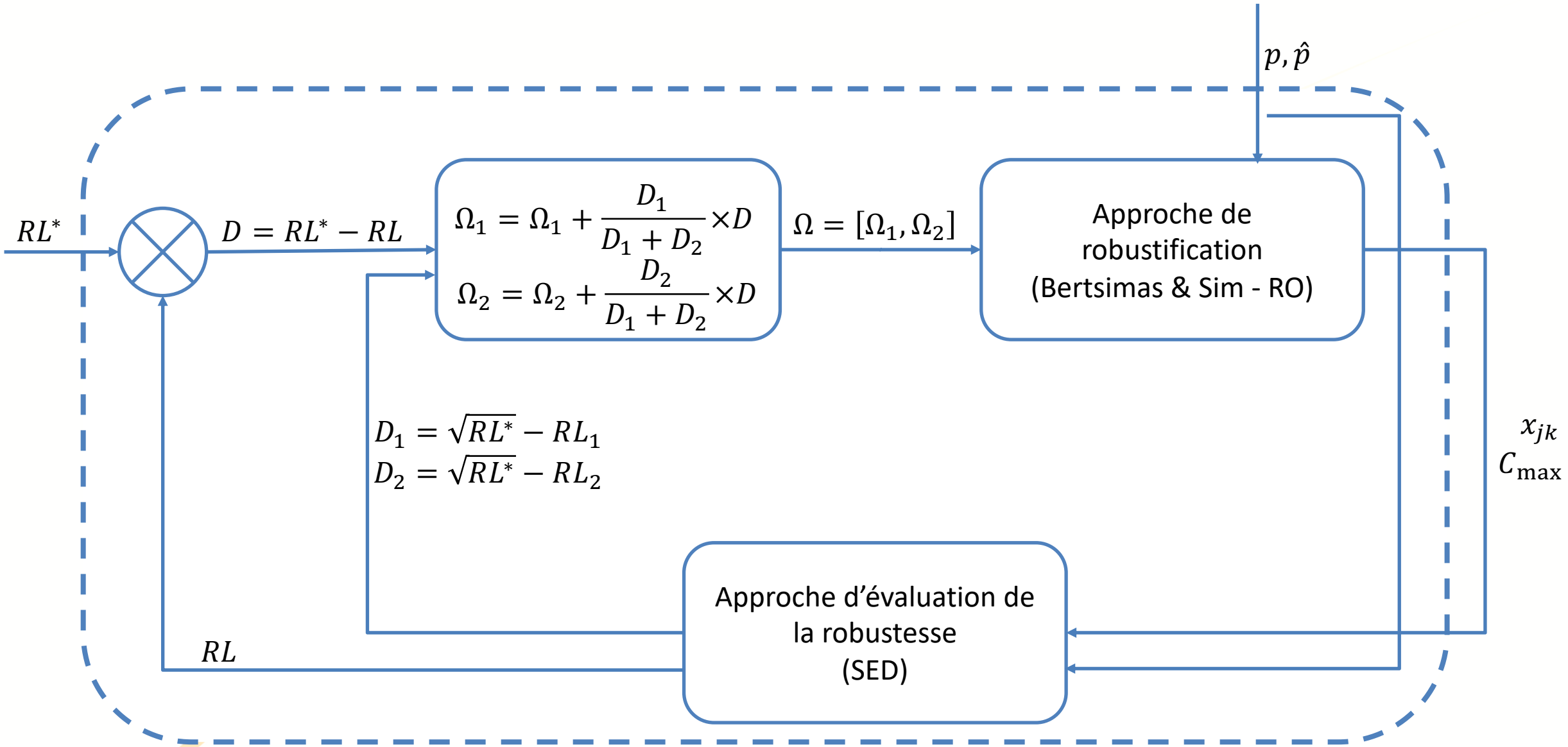
PLAN

- Outil de modélisation
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- Outil d'évaluation - Vérification
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- Approche d'évaluation
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- Application sur UppAal SMC
- **Couplage Approche robustification et Evaluation Niveau de service**
- Application Cplex-UppAal SMC

MISE À JOUR DES COEFFICIENTS Ω

- x_{jk} ordonnancement et Cmax
- RL^* niveau de robustesse souhaité par le décideur (ici 90%)
- RL niveau de robustesse lorsque les perturbations impactent toutes les machines
- RL_1 niveau de robustesse lorsque les perturbations impactent que la machine 1
- RL_2 niveau de robustesse lorsque les perturbations impactent que la machine 2

- Ω_i coefficient de robustification de la machine i à l'itération t (ici 2machines : i = 1, 2)


$p, \hat{p}$
RL^*
$D = RL^* - RL$
$\Omega_1 = \Omega_1 + \frac{D_1}{D_1 + D_2} \times D$
$\Omega_2 = \Omega_2 + \frac{D_2}{D_1 + D_2} \times D$
$\Omega = [\Omega_1, \Omega_2]$
Approche de robustification (Bertsimas & Sim - RO)
$D_1 = \sqrt{RL^*} - RL_1$
$D_2 = \sqrt{RL^*} - RL_2$
x_{jk}
$C_{\max}$
Approche d'évaluation de la robustesse (SED)
RL

PLAN

- Outil de modélisation
  - Automate temporisé
  - Automate stochastique
  - Automate temporisé stochastique
- Outil d'évaluation - Vérification
  - Rappel Model-checking
  - Model-checking numérique
  - Model-checking statistique
- Approche d'évaluation
  - Perturbations
  - Robustesse
  - Processus d'évaluation
- Application sur UppAal SMC
- Couplage Approche robustification et Evaluation Niveau de service
- **Application Cplex-UppAal SMC**

APPLICATION

- **Atelier de type : machines parallèles M1, M2**
- **10 opérations :**
  - Durées sur chaque machine : p=[[1,2][2,1][5,2][3,4][8,6][2,5][4,3][2,4][4,2][1,1]];
  - Intervalle de perturbation: $\hat{p}$=[[0,1][1,0][3,1][2,2][5,3][1,1][2,2][1,3][1,1][0,0]];

- **Ordonnancement à évaluer :**
  - Cmax = 12;
  - x = [[1 0][0 1][0 1][1 0][0 1] [1 0][1 0][1 0][0 1][0 1]];

- **Niveau de service attendu par le décideur : 90%**

REFERENCES

- Billaut, J., Moukrim, A., and Sanlaville, E. (2010). Flexibility and Robustness in Scheduling. ISTE.
- Cassandras, C.G. and Lafortune, S. (2008). Introduction to discrete event systems, second edition. Springer.
- Dauzere-Peres, S., Castagliola, P., and Lahlou, C. (2013). Service level in scheduling. Flexibility and Robustness in Scheduling, Chapter 5, 99{121.
- David, A., Larsen, K.G., Legay, A., Mikucionis, M., and Poulsen, D.B. (2015). Uppaal smc tutorial. International Journal on Software Tools for Technology Transfer, 17(4), 397{415.
- Feng, W., Zheng, L., and Li, J. (2012). The robustness of scheduling policies in multi-product manufacturing systems with sequence-dependent setup times and finite buffers. Computers & Industrial Engineering, 63(4),1145{1153.
- Ghezail, F., Pierreval, H., and Hajri-Gabouj, S. (2010). Analysis of robustness in proactive scheduling: A graphical approach. Computers & Industrial Engineering, 58(2), 193{198.
- Giard, V. (2003). Gestion de la production et des flux: avec CD livre electronique+ Logiciels+ Animations. Economica.
- Himmiche, S., Aubry, A., Marange, P., Duflot, M., and Petin, J.F. (2017). Using statistical-model-checking based simulation for evaluating the robustness of a production schedule. 7th Workshop on Service Orientation in Holonic and Multi-agent Manufacturing: SOHOMA 2017.
- Ivanov, D., Dolgui, A., Sokolov, B., and Werner, F. (2016). Schedule robustness analysis with the help of attainable sets in continuous ow problem under capacity disruptions. International Journal of Production Research,
- Kobetski, A. and Fabian, M. (2009). Time-optimal coordination of flexible manufacturing systems using deterministic finite automata and mixed integer linear programming. Journal of Discrete-Event Dynamic Systems: Theory and Applications, 19(3), 287{315.
- Kouvelis, P., Daniels, R.L., and Vairaktarakis, G. (2000). Robust scheduling of a two-machine ow shop with uncertain processing times. Iie Transactions.
- Larsen, K.G., Pettersson, P., and Yi, W. (1997). Uppaal in a nutshell. International journal on software tools for technology transfer, 1(1-2), 134{152.
- Lefebvre, D. (2017). Evaluating the robustness of scheduling in uncertain environment with Petri nets. In VALUETOOLS 2017 - 11th EAI International Conference on Performance Evaluation Methodologies and Tools. Venice, Italy.
- Ouelhadj, D. and Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. Journal of scheduling,.
- Panek, S., Engell, S., and Stursberg, O. (2006). Scheduling and planning with timed automata. In 16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering, 1973?{1978. Elsevier.
- Plateau, B. and Atif, K. (1991). Stochastic automata networks for modelling parallel systems. Reo Project.
- Xiong, J., Xing, L., and Chen, Y. (2013). Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. International Journal of Production Economics, 141(1), 112{126