Raisonnement diagnostic sur les systèmes à événements discrets

Yannick Pencolé

31 mars 2008

Plan du cours

Plan du cours

Concept de diagnostic

VOIR **COMPRENDRE** **DÉCIDER**

Acquérir les observations

Détecter un problème / **Identifier** la panne

Modifier le système

Infos nécessaires à un bon diagnostic

Infos nécessaires à une bonne décision

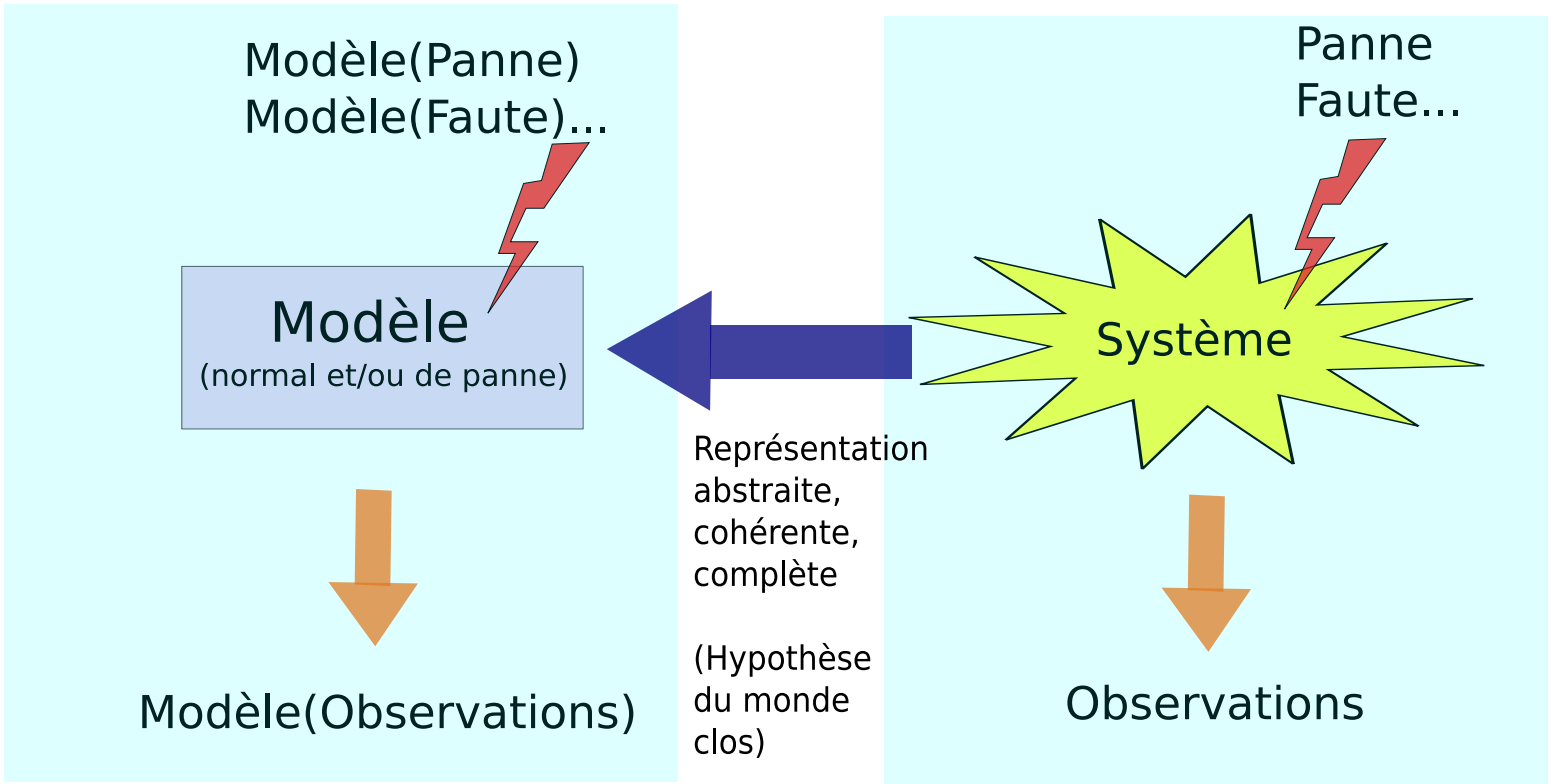
Concept de système et de modèle

- Qu'est-ce qu'un système ?
  - C'est une réalité
  - Assemblage de composants

    An aggregation or assemblage of things so combined by nature or man as to form an integral or complex whole » [Encyclopedia Americana]

- Et un modèle ?
  - Une représentation d'un système
  - Modèle adapté à la tâche à effectuer :
    - compréhension, simulation, planification, commande, diagnostic, suivi
- Distinction entre système et modèle : essentielle
  (en général mais plus particulièrement en diagnostic)

Concept du diagnostic à base de modèle



Diagnostic à base de modèle =

1) Confronter les observations au modèle (cohérence)
2) "Remonter" à la cause (abduction, modèle de panne)

Classification des systèmes et modèles DYNAMIQUES étudiés en diagnostic

- **Continu**
  - Modèle en terme de mode
  - Variables continues, valeurs réelles, temps continu
  - Équations différentielles
  - Modèles numériques : Automatique
  - Modèles qualitatifs (ou semi-quantitatifs)
- **Discret**
  - Variables discrètes et temps discret
  - Notion d' événement
- **Hybride**
  - Mode (continus), Changement de modes (discret)

Plan du cours

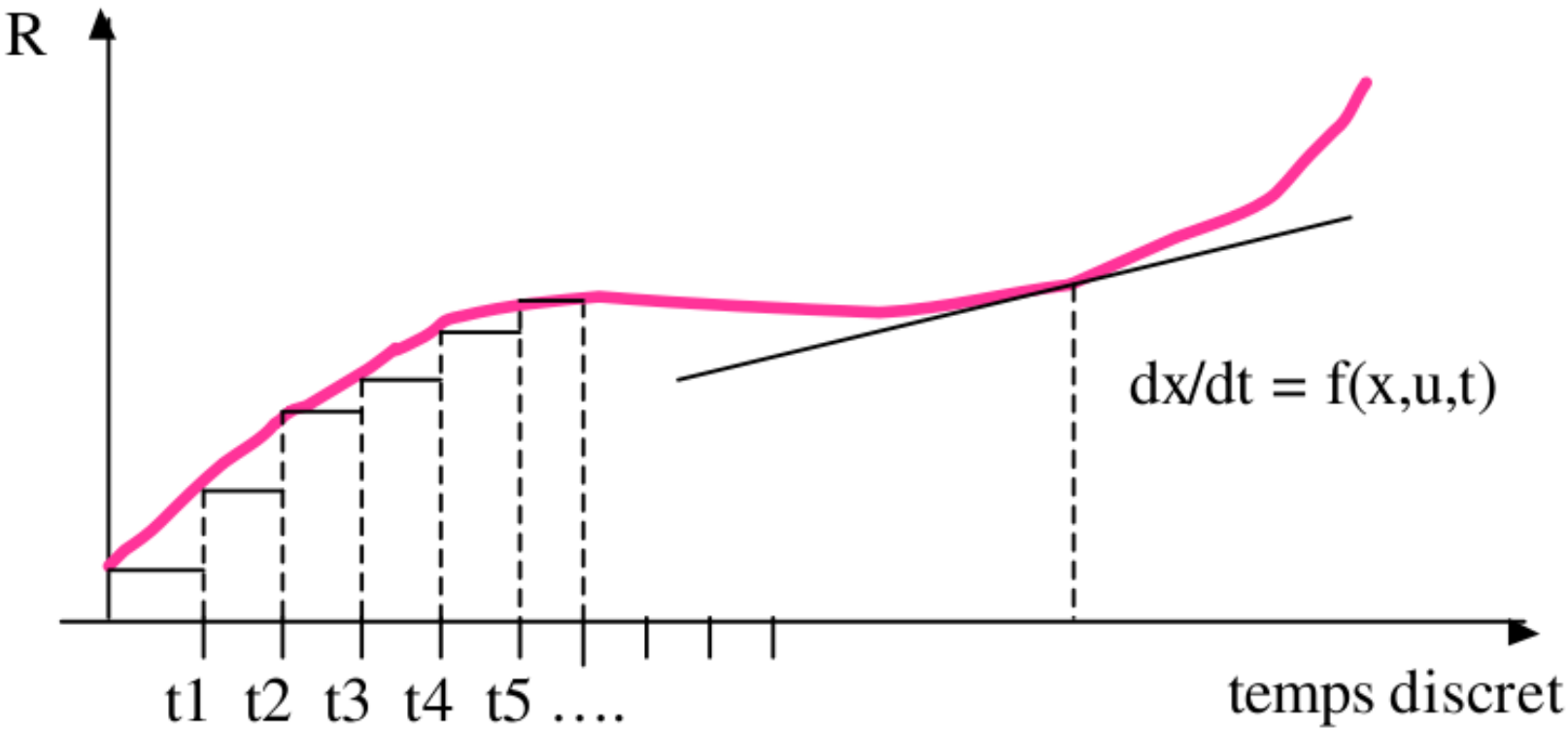
Pourquoi étudier les systèmes à événements discrets ?

- Engouement né de la complexification des systèmes à aborder
- Apparition des ordinateurs et du traitement informatique (numérique)
- Les SED sont en générale des conceptions humaines
  - réseaux de transport,
  - reseaux informatiques,
  - ordinateurs, calculateurs,
  - protocoles manufacturiers,
  - gestion d'entreprises, procédures...
- Utile aussi lorsqu'on peut discretiser un système quelconque (modélisation qualitative)

Temps discret vs Etat discret

- 2 types de discrétisations
- Discretisation du temps
  - échantillonage d'un signal continu par exemple
- Discrétisation de l'état
  - Abstraction d'états en classe discrète d'états

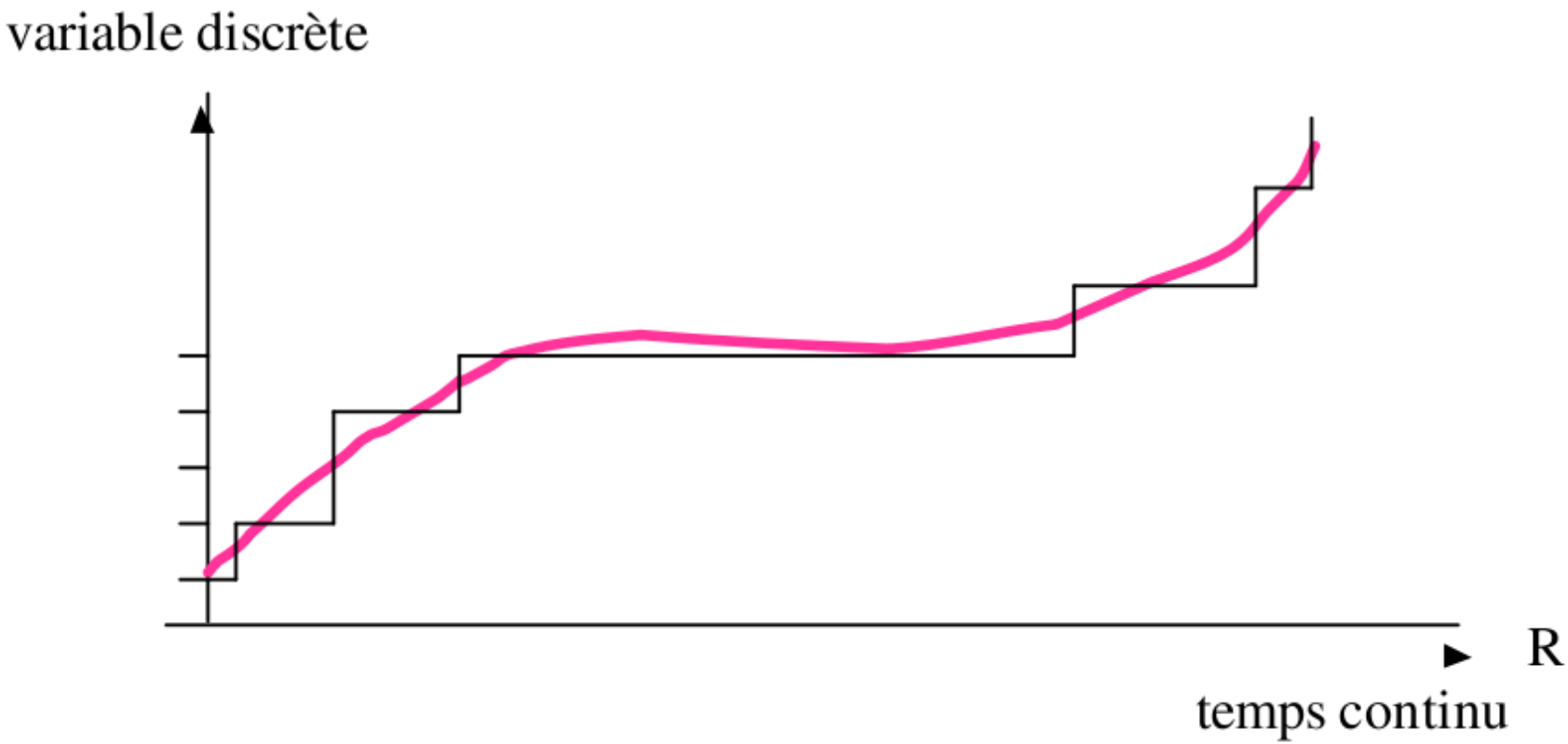
Temps discret



Temps = ensemble d'instant (horloge)

Etat = continu

État discret



Temps = à valeur continue

Etat = discret

Dynamiques temporelle vs événementielle

- Dynamique temporelle
  - à chaque tic d'horloge, on regarde ce qui se passe
  - modèle *time-driven*
  - Synchronisation des composants par l'horloge
  - Dynamique synchrone
- Dynamique événementielle
  - C'est l'occurrence d'un événement sur le système qui sert d'horloge
  - Modèle *event-driven*
  - Dynamique asynchrone (échange de messages, rendez-vous)

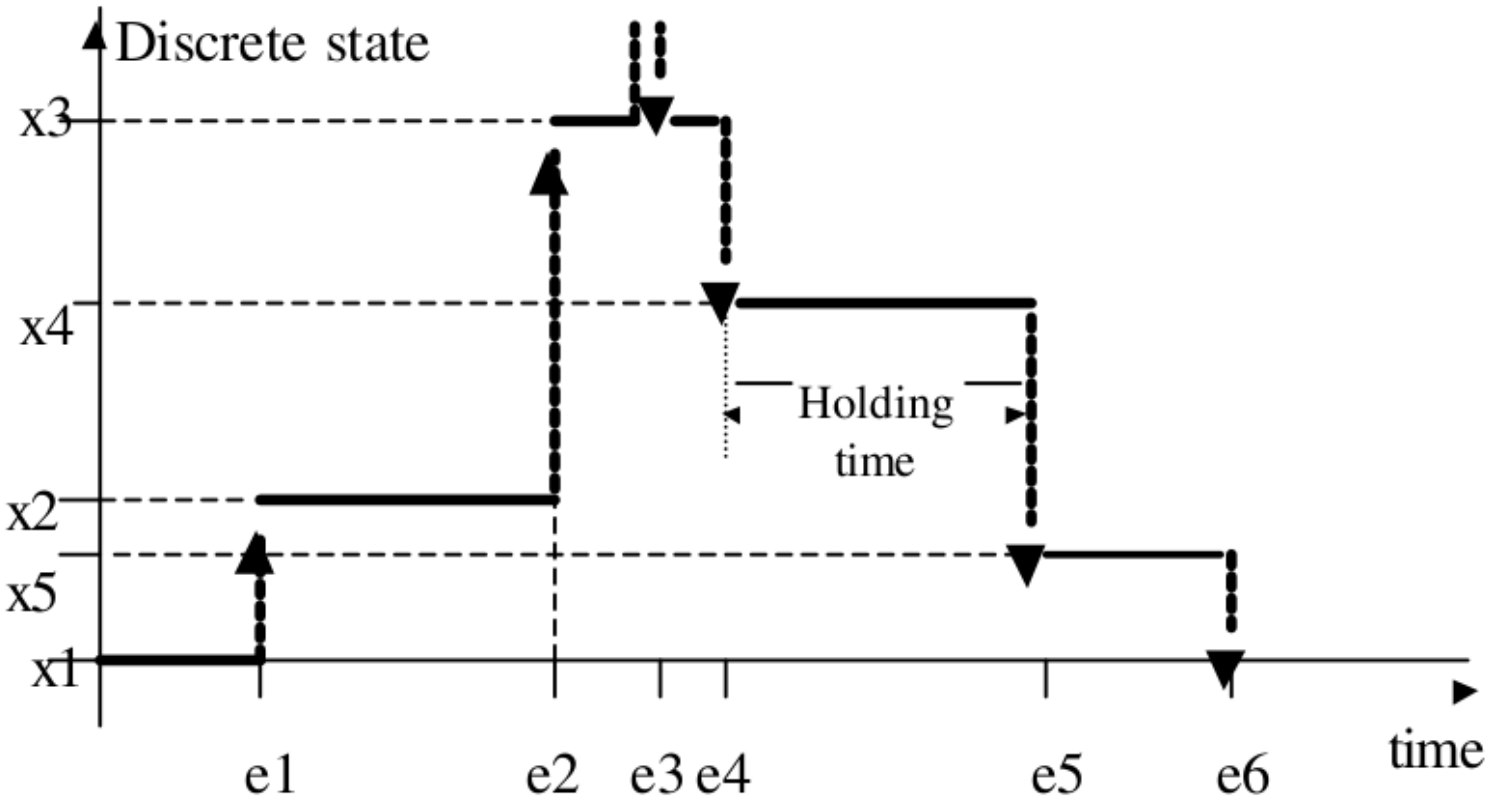
Concept d'événements

- Caractéristiques
  - instantané
  - provoque (éventuellement) un changement d'état
  - contrôlable (action) ou spontané (panne)
  - exogène ou endogène

Exemple

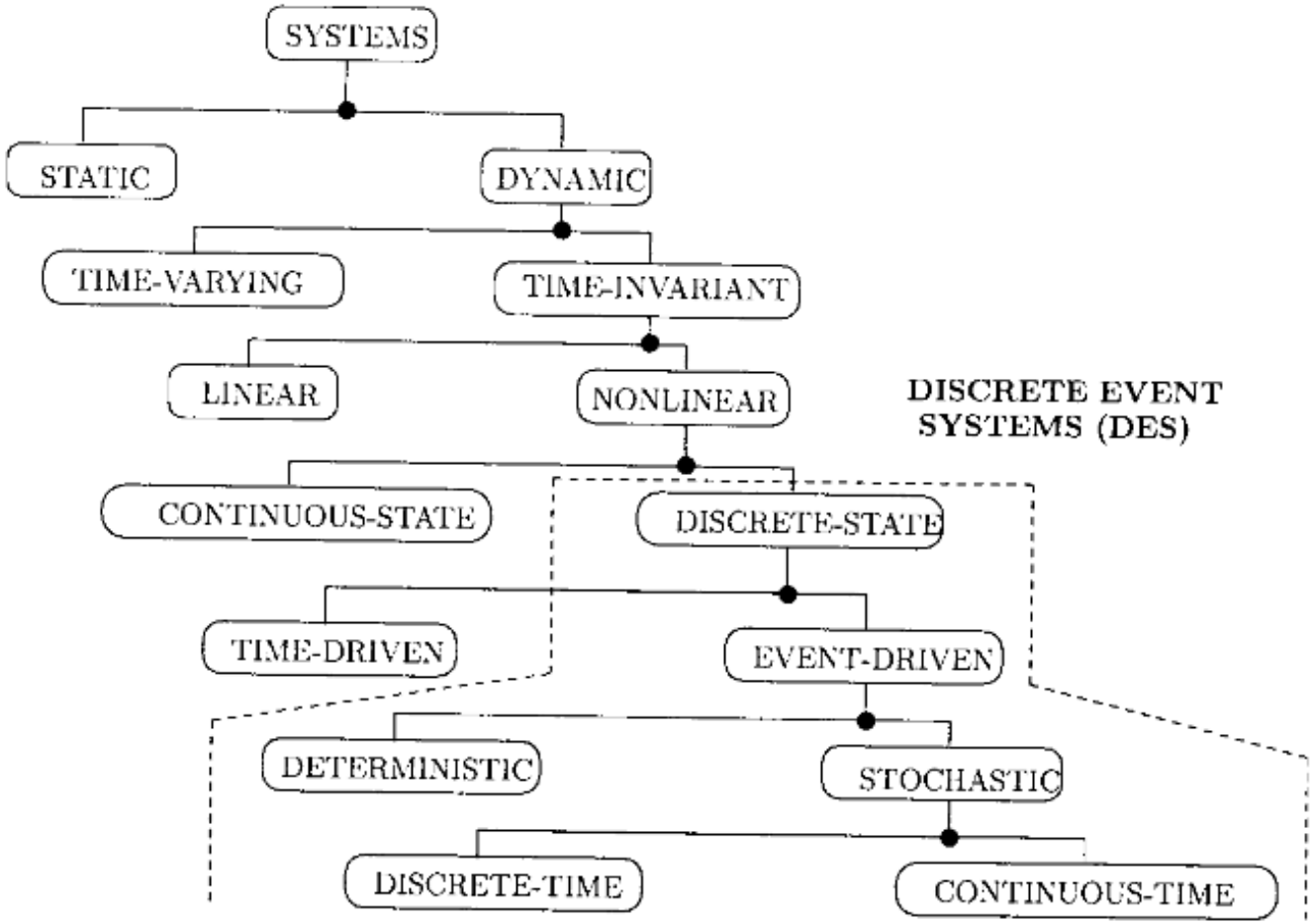
- Une personne presse un bouton. Elle ouvre une vanne.
- La machine se réinitialise. Court-circuit.
- Le niveau maximal de liquide dans le récipient est atteint.

Caractéristiques des SED



Etat discret, dynamique événementielle

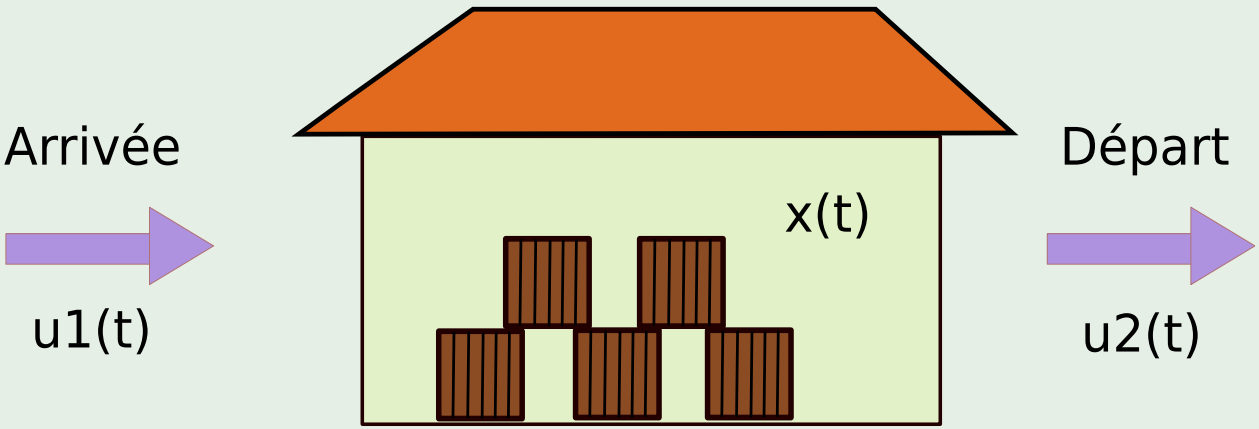
Classification des systèmes



Système discret par nature (1)

Exemple

Entrepôt

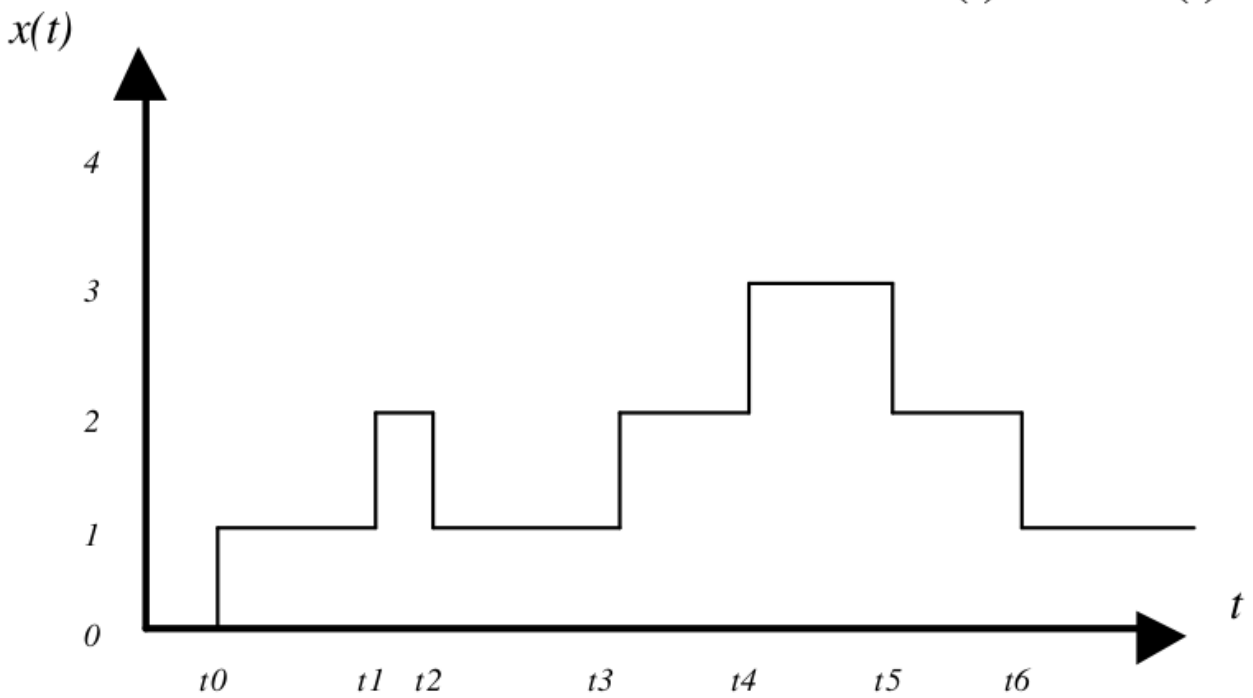


- $x(t)$ nombre de caisses dans l'entrepôt à l'instant t
- $u1(t) = 1$ si une caisse arrive au temps t, 0 sinon
- $u2(t) = 1$ si un caisse part au temps t, 0 sinon

Système discret par nature (2)

Exemple

Si u1(t) alors x(t) = x(t-1) +1
Si u2(t) alors x(t) = x(t-1) -1



Système discrétisable (1)

- Système continu mais modèle discret
  - Selon la tâche à effectuer sur le système
  - Modèle qualitatif

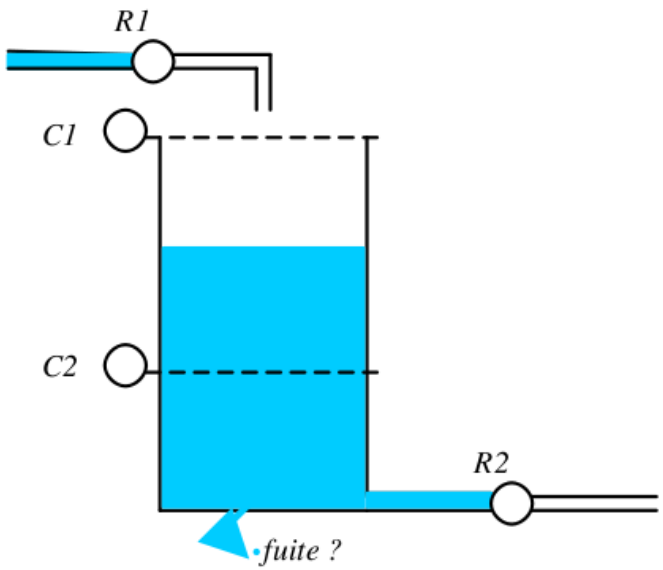
Exemple

Le réservoir

Système discrétisable (2)

Exemple

- Récipient



R1 : {ouvert, fermé}
R2 : {ouvert,fermé}
C1 : {niv_atteint,niv_pas_atteint}
C2 : {niv_atteint,niv_pas_atteint}

Espace d 'états discrets :
R1 x R2 x C1 x C2

Exemples d'événements :
Ouverture de R1
Fermeture de R2
Le niveau passe au dessous de C2
Ca déborde !!

- Détection d 'une fuite :
  - *R1=fermé, R2=fermé, C2=niv_atteint*
  - *événement : « le niveau devient inférieur à C2 »*

Plan du cours

Choix du formalisme de modélisation

- Avant de modéliser :
  1. quelle est la nature de mon système ?
     - monolithique, distribué, continu, discret
  2. quel est l'objectif du diagnostic ?
     - détection, localisation, identification,...
  3. quel est le degré de connaissance de mon système ?
     - connaissance de surface, comportement nominal, comportement de panne
- Choix du formalisme de modélisation
  - Dépend de la réponse aux questions précédentes
- De nombreux formalismes sont possibles
  - Algèbre de processus, Système de transitions (automate), Langage, Règles, Réseau de Petri

Système de transition

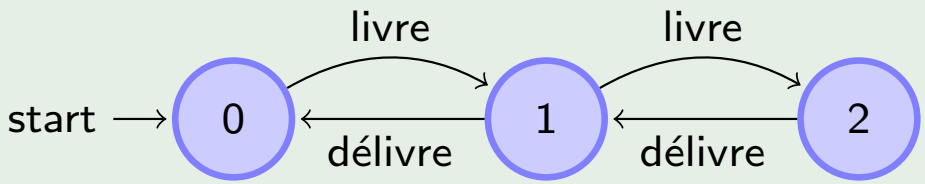
Définition

Système de transition : $G = (X, E, f, x_0)$

- X : ensemble d'états
- E : ensemble des événements
- f : fonction de transition $X \times E \longrightarrow 2^X$
- x_0 : état initial

Exemple

Un entrepôt contenant deux caisses au maximum.



Système de transition : modélisation compositionnelle

- Système : ensemble de composants
- Modélisation monolithique d'un système
  1. compliqué (recensement de tous les comportements possibles)
  2. souvent irréaliste (trop nombreux, même pas codable)
  3. ne met pas à profit le fait qu'un système est un ensemble de composants
- Modélisation compositionnelle
  - Modèle du comportement de chaque composant (plus simple, moins gros)
  - Spécification d'une loi de composition des modèles
  - Modèle du système $\equiv$ Modèles des composants $+$ Loi de composition
- Loi de composition
  - Fondé en général sur un produit de système de transitions
  - Fonction de règles de synchronisation entre événements

Loi de composition (1)

Soit $G_1, \ldots, G_n$ avec $G_i = (X_i, E_i, f_i, x_{0i})$ les modèles locaux, soit ε l'événement vide tel que $\forall i, f_i(x, \varepsilon) = \{x\}$, soit $Sync \subseteq (E_1 \cup \{\varepsilon\}) \times \cdots \times (E_n \cup \{\varepsilon\})$ la relation de synchronisation.

Définition

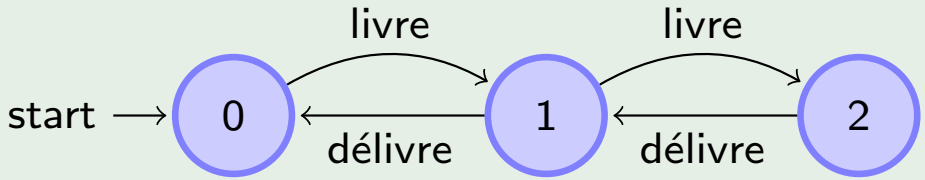
Le produit synchronisé $G = G_1 ||_{Sync} \cdots ||_{Sync} G_n$ est le système de transition : $G = (X, E, f, x_0)$

- $X \subseteq X_1 \times \cdots \times X_n$ (ensemble des états accessibles)
- $E = Sync \subseteq (E_1 \cup \{\varepsilon\}) \times \cdots \times (E_n \cup \{\varepsilon\})$
- $x_0 = (x_{01}, \ldots, x_{0n})$
- f fonction de transition :
  $f((x_1, \ldots, x_n), (e_1, \ldots, e_n)) = \{f_1(x_1, e_1) \times \cdots \times f_n(x_n, e_n)\}$

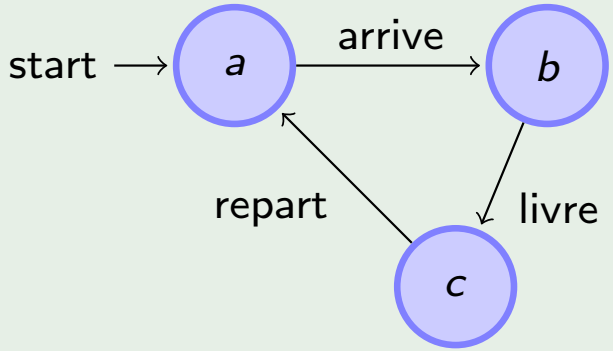
Loi de composition (2)

Exemple

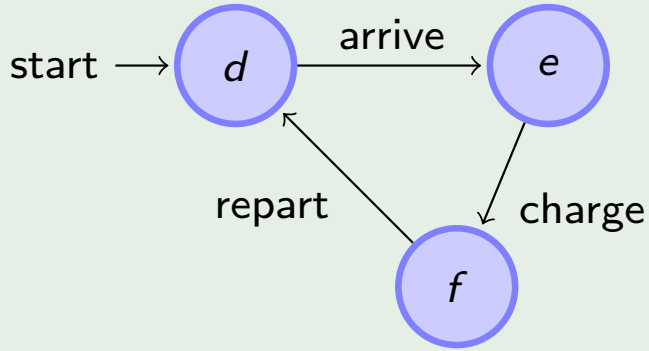
Entrepôt :



Livreur :



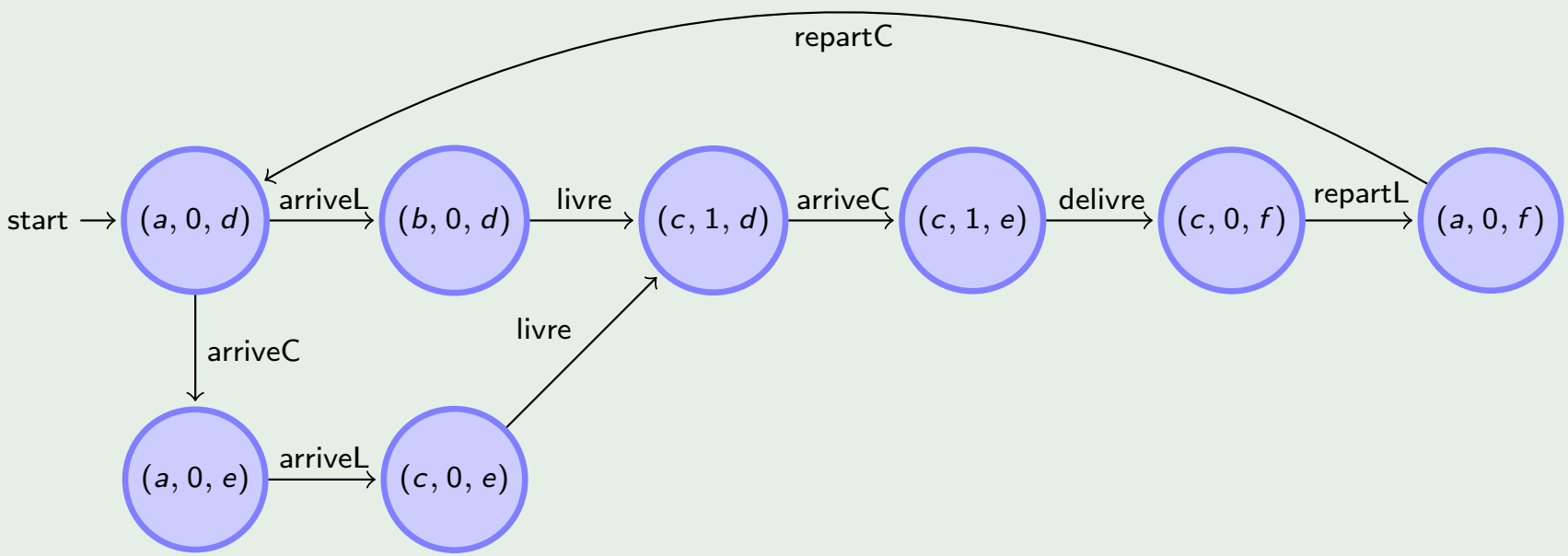
Client :



Relation de synchronisation *Sync* : $arriveL = (arrive, \varepsilon, \varepsilon)$, $repartL = (repart, \varepsilon, \varepsilon)$, $arriveC = (\varepsilon, \varepsilon, arrive)$, $repartC = (\varepsilon, \varepsilon, repart)$, $livre = (livre, livre, \varepsilon)$, $delivre = (\varepsilon, delivre, charge)$.

Loi de composition (3)

Exemple

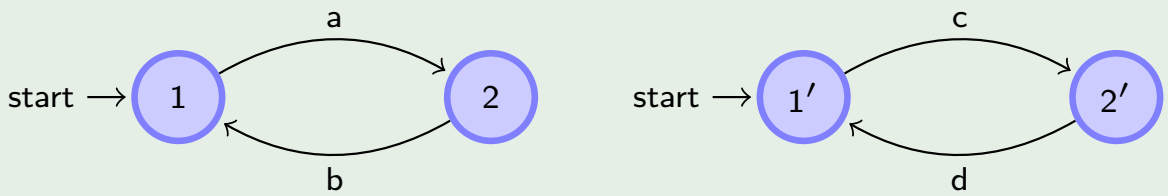


Différentes lois de composition

- Fonction de *Sync*
  1. $\forall(e_1, \ldots, e_n) \in Sync, e_i \neq \varepsilon \Rightarrow$ produit synchrone (tous les composants évoluent en même temps)
  2. $\forall(e_1, \ldots, e_n) \in Sync, \exists! e_i \neq \varepsilon \Rightarrow$ produit libre (tous les composants évoluent indépendamment les uns des autres)
  3. Tout ensemble *Sync* définit un produit synchronisé particulier
- Modèle global $G = G_1 ||_{Sync} \ldots ||_{Sync} G_n$
  - Au pire cas (et en pratique), taille exponentielle en le nombre de composants
  - Explosion combinatoire
  - Problème de calcul

A vous de jouer

Exemple



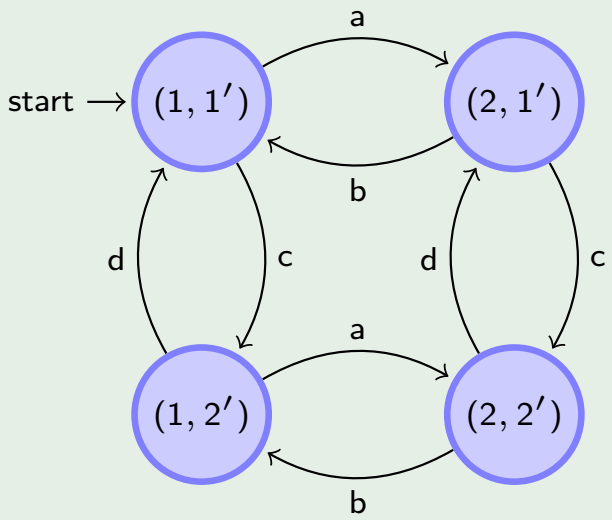
Ecrire le produit libre : $Sync = \{(a, \varepsilon), (b, \varepsilon), (\varepsilon, c), (\varepsilon, d)\}$

Ecrire le produit synchrone : $Sync = \{(a, c), (b, d)\}$

Ecrire le produit synchronisé : $Sync = \{(a, \varepsilon), (b, c), (\varepsilon, d)\}$

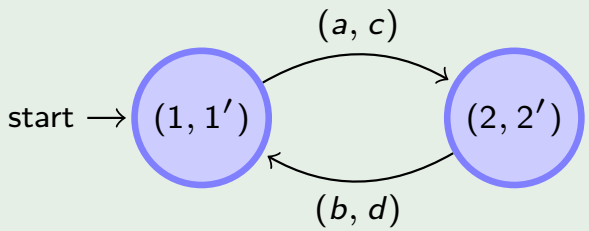
Résultat du jeu : produit libre

Exemple



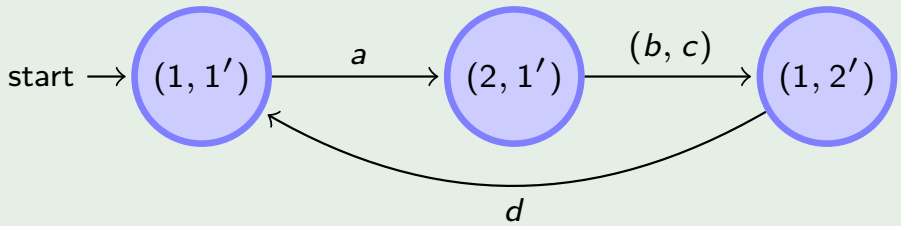
Résultat du jeu : produit synchrone

Exemple



Résultat du jeu : produit synchronisé

Exemple



Réseau de Petri

Définition

Réseau de Petri : $R = (P, T, Pré, Post)$

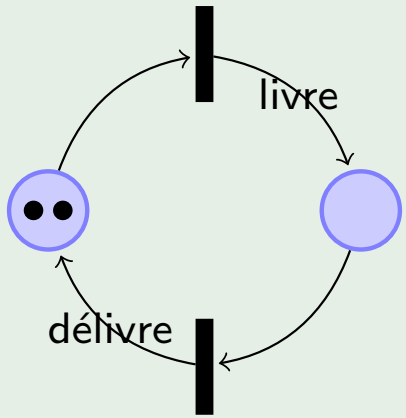
- P : ensemble de places
- T : ensemble de transitions
- $Pré$: fonction incidence arrière $Pré : P \times T \rightarrow \mathbb{N}$
- $Post$: fonction incidence avant $Post : P \times T \rightarrow \mathbb{N}$

Réseau de Petri marqué : (R, M_0) où M_o est le marquage initial.
Fonction de marquage $M : P \rightarrow \mathbb{N}$

Réseau de Petri (2)

Exemple

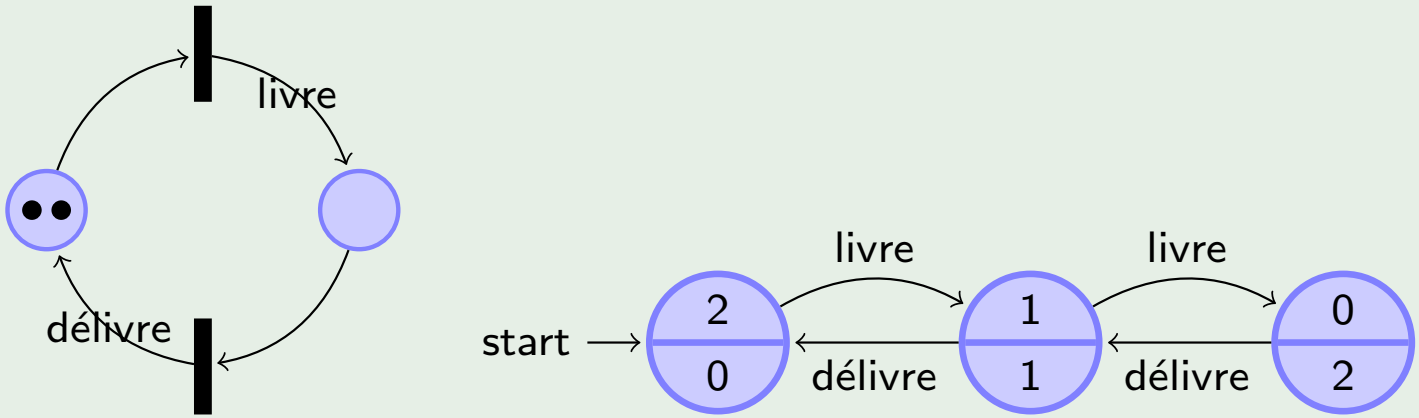
Un entrepôt contenant deux caisses au maximum.



Graphe des marquages accessibles

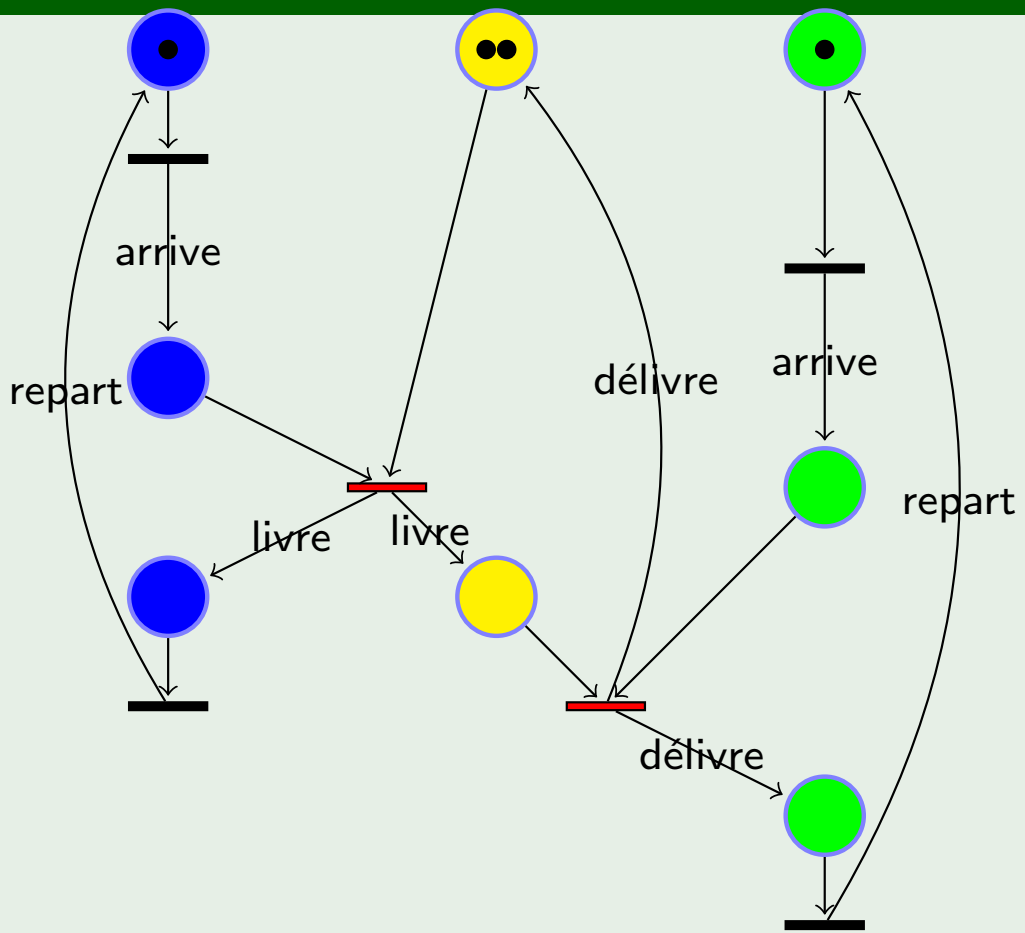
- Etant donné un réseau de Petri marqué
- Par simulation exhaustive, on peut obtenir le graphe des marquages accessibles
- Graphe des marquages $\equiv$ Système de transition

Exemple



Livreur+Entrepôt+Client en RDP

Exemple



RDP vs Modélisation compositionnelle

- Tout d'abord et avant tout, c'est une affaire de goût (éducation, culture)
- Modélisation compositionnelle
  1. Plus intuitif (notion explicite de composants)
  2. Plus simple à définir pour les non-spécialistes (notion de message et de ports)
  3. Pas terrible pour exprimer la concurrence entre composants
  4. Sémantique compositionelle peu maitrisée
- Réseau de Petri
  1. Expression aisée de la concurrence
  2. Sémantique de synchronisation connue et maitrisée (elle est exprimée dans le réseau même)
  3. Pas très intuitif pour un néophite
  4. L'aspect composant n'est pas si clair

Algèbre de processus

Définition

Les algèbres de processus sont un famille de langages formels permettant de modéliser les systèmes (informatiques) concurrents ou distribués.

Exemple

Livreur	**Entrepôt**	**Client**
L1 := arriveL.L2	E0 := livre.E1	C1 := arriveC.C2
L2 := livre.L3	E1 := livre.E2 + delivre.E0	C2 := delivre.C3
L3 := repartL.L1	E2 := delivre.E1	C3 := repartC.C1

Règles, pièces (*Tiles*)

Définition

Soit $V = \{v_1, \ldots, v_n\}$ variables à domaine finie. Un état du système est une affection possible des n variables. Une règle R est un 4-uplet :

$$R = (Pre, I, O, Eff)$$

- Pre : formule logique de précondition. Si l'état courant satisfait la précondition alors la règle peut être tirée (sensibilisée)
- I : événement déclenchant la règle
- O : ensemble d'événements produits par la règle (communication/synchronisation)
- Eff : formule logique définissant l'effet sur l'état (postcondition)

Règles, pièces (*Tiles*)(2)

- Proche des réseaux de Petri, des STRIPS (planification)
- Modélisation plus intuitive, avec plus de sens
- Aspect composant plus clair

Exemple

```
Entrepot(in entree, in sortie) {
variables :
    stock in { 0,1,2 }
rules :
    require(stock <= 1)
        when entree.livre
        effect(stock = old(stock) + 1)
    require(stock >= 1)
        when sortie.delivre
        effect(stock = old(stock) - 1)
}
```

Règles, pièces (*Tiles*)(3)

Exemple

```
Livreur(out magasin) {
events :
    arrive,repart,livre
variables :
   etape in { enRoute, enLivraison,finLivraison },
rules :
    require(etape = enRoute)
        when arrive
        effect(etape = enLivraison)
    require(etape= enLivraison)
        when livre
        output magasin.livre
        effect(etape = finLivraison)
    require(etape = finLivraison)
        when repart
        effect(etape = enRoute)
}
```

```
Entrepot(in entree, in sortie) {


variables :
    stock in { 0,1,2 }
rules :
    require(stock <= 1)
        when entree.livre
        effect(stock = old(stock) + 1)
    require(stock >= 1)
        when sortie.delivre
        effect(stock = old(stock) - 1)
}
```

structure

connect(Livreur.magasin, Entrepot.entree)

Caractérisation d'un SED : langage

- Un SED = un ensemble de séquences d'événements
  - séquence d'événements représenté par une séquence de symboles
- symbole ≡ lettre
- séquence de symbole ≡ mot
- SED ≡ langage
  - ensemble (fini ou non) de mots

Exemple

Entrepôt :

"livre" et "délivre" sont des lettres
"livre délivre livre livre" est un mot du langage entrepôt
"livre délivre délivre" n'en est pas un

Langage

Définition

- Soit Σ un alphabet, ensemble fini de symboles
- Un mot sur Σ est une séquence finie de lettres de Σ
- Un langage $L(\Sigma)$ est un ensemble de mots sur Σ

Définition

Soit L_1, L_2 deux langages sur l'alphabet Σ :

- Union $L_1 + L_2 = L_1 \cup L_2 = \{m, m \in L_1 \vee m \in L_2\}$
- Concaténation $L_1.L_2 = \{m_1.m_2, m_1 \in L_1 \wedge m_2 \in L_2\}$
- Clotûre de Kleene
  $L_1^\star = \{m_1.m_2.\ldots.m_k, k >= 0 \wedge \forall i \in \{1, \ldots, k\}, m_i \in L_1\}$

Opérations régulières

Exemple

Soit $\Sigma = \{a, b\}$ et $L_1 = \{a, bb\}$ $L_2 = \{ab\}$ deux langages :

$L_1 + L_2 = \{a, bb, ab\}$

$L_1.L_2 = \{aab, bbab\}, L_2.L_1 = \{aba, abbb\}$

$L_2^\star = \{\varepsilon, ab, abab, ababab, \ldots\}$

$L_1^\star = \{\varepsilon, a, bb, abb, bba, aa, bbbb, aabb, \ldots\}$

Langage régulier

Définition

Un langage est régulier si et seulement s'il est un langage appartenant à l'ensemble récusivement défini par :

1. Le langage vide $L = \{\}$
2. Le langage contenant le mot vide $L = \{\varepsilon\}$
3. Tout langage singleton $L = \{a\}, a \in \Sigma$
4. $L_1 + L_2$, $L_1.L_2$ et $L_1^\star$ pour tous langages L_1, L_2 réguliers

Théorème

A tout système de transition, on peut associer un langage régulier. Donc, à tout SED, on peut associer un langage régulier (quoique ? ? ?)

Clotûre par préfixe

Définition

La clotûre par préfixe d'un langage L : $\overline{L} = \{m_1, \exists m = m_1.m_2 \in L\}$
(m_1 est un préfixe de m)

Théorème

A tout SED, on fait correspondre un langage clos par préfixe.

Exemple

$L_1 = \{aba, bb\}$

$\overline{L_1} = \{\varepsilon, a, b, ab, bb, aba\}$

Projection de langage

Définition

Soit Σ, Σ' deux alphabets, soit L un langage Σ, la projection $P_{\Sigma'}(L)$ de L sur Σ' est définie récursivement par :

- $P_{\Sigma'}(\varepsilon) = \varepsilon$
- $P_{\Sigma'}(l.m) = l.P(m)$ si $l \in \Sigma'$
- $P_{\Sigma'}(l.m) = P(m)$ sinon

Exemple

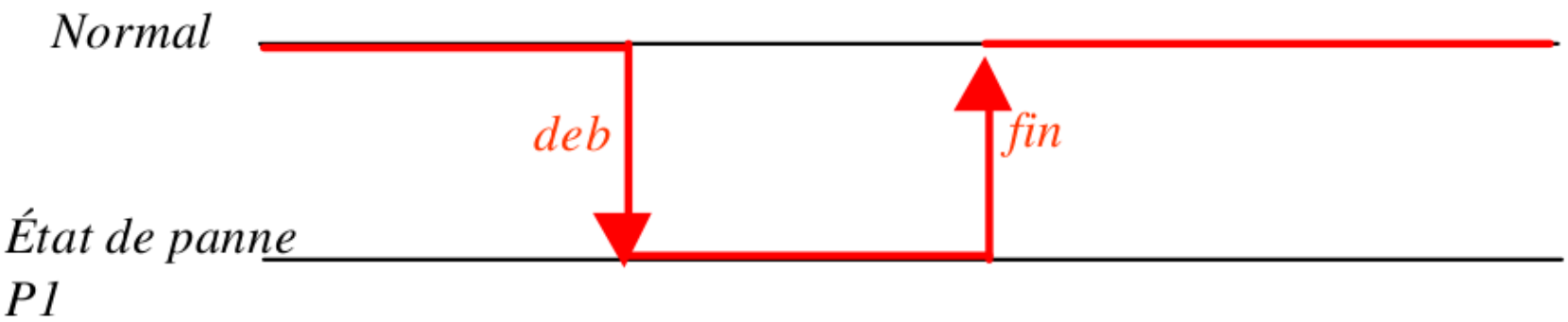
$\Sigma = \{a, b, c\}$, $\Sigma' = \{b, c, d\}$
$L = \{abbbbaac, bbaac\}$

$P_{\Sigma'}(L) = \{bbbbc, bbc\}$

Plan du cours

Notion de panne, de faute

- Pannes, Fautes :
  - Événements non-observables
    1. *début de panne*
    2. éventuellement *fin de panne*

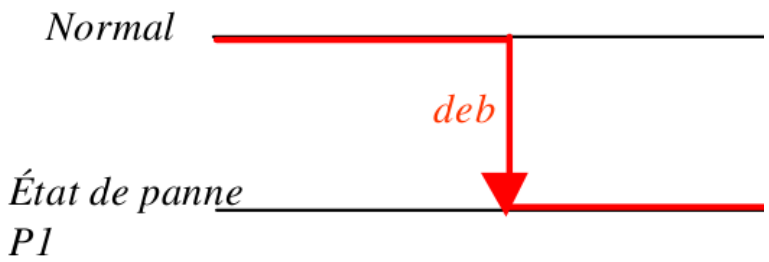


- Terminologie : panne ? faute ? Grand débat franco-français. (et même toulouso-toulousains)
- Dépend de l'application et de son propre champ terminologique

Panne permanente

Définition

Une panne est permanente s'il n'existe aucun moyen de la réparer ou aucune raison qu'elle disparaisse pendant la période de diagnostic.



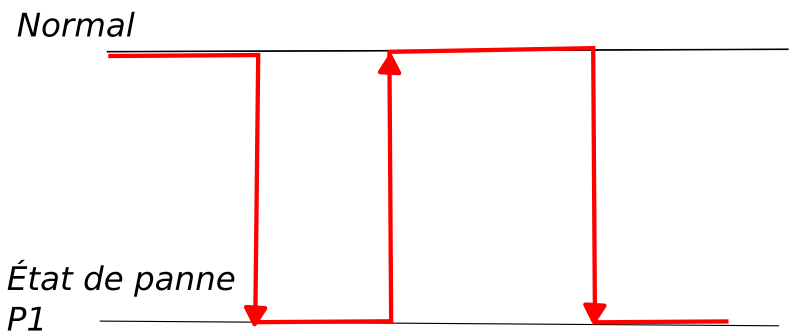
Exemple

Diagnostic en-ligne d'une automobile : une panne moteur est permanente. Cette panne ne peut pas disparaître sans une intervention extérieure (garage)

Panne intermittente

Définition

Une panne est intermittente si elle peut apparaître/disparaître pendant la période de diagnostic à plusieurs reprises.



Exemple

Diagnostic en-ligne d'une automobile : un mauvais contact filaire entre calculateurs peut apparaître et disparaître (vibration, humidité) au cours d'un même trajet.

Modèle de diagnostic

Soit M le modèle SED de diagnostic (ou son langage), il contient toujours :

- Des événements observables O

Il contient parfois :

- Des événements de pannes, de fautes (modèle de panne, de faute, de dysfonctionnement) F
  - Un tel événement est en général non observable (sinon problème trivial)
- Des évenements de communication (modèles de composants interagissants) C

Problème général du diagnostic à base de modèle de SED

- Étant donné un modèle M
- Étant donné des événements observables
  - Flux d'observations arrivant en-ligne (ordre total ou non)
- Quels sont les événements non-observables qui expliquent les observations selon le modèle ?

Objectif de diagnostic

Fonction des connaissances, le diagnostic a plusieurs sous-objectifs :

1. Détection : détecter la présence d'une panne
2. Localisation : identifier le composant où la panne s'est produite
3. Identification : identifier la nature, le type de la panne
4. Propagation : déterminer toutes les conséquences de la panne (relation cause-effet)

Problème de diagnostic : détection

Soit σ une séquence d'observations, soit M un modèle nominal de diagnostic

Définition

Diagnostic par cohérence $\equiv$ détection :
si $P_O(M) \cap \{\sigma\}$ est vide alors le système ne se comporte pas comme prévu $\Rightarrow$ détection d'un problème.

Exemple

Système Livreur+Entrepôt+Client, $O = \{arriveC, arriveL\}$
Si $\sigma = arriveC\ arriveC$ alors problème (le client aurait dû attendre l'arrivé d'un livreur)
Si $\sigma = arriveC\ arriveL\ arriveC$ on reste cohérent :
$P_O(M) \cap \{\sigma\} = \sigma$

Problème de diagnostic : localisation

Soit M un modèle nominal à base de composant $C_1, \ldots, C_n$

Définition

Localisation :
Si $P_{O,C}(M) \cap \{P_C(\sigma)\}$ est vide alors le composant C est un diagnostic candidat.

Exemple

Système Livreur+Entrepôt+Client, $O = \{arriveC, charge\}$
Si $\sigma = arriveC\ arriveC$ alors problème possible chez le client.

Problème de diagnostic : identification et propagation

Soit M un modèle de panne identifiant les pannes $f_1, \ldots, f_n$ et leurs conséquences

Définition

Propagation : ensemble de trajectoires expliquant σ

$$H = M||_O\{\sigma\}$$

Identification :

$$I = P_{Panne}(H)$$

Diagnostic en-ligne vs diagnostic hors-ligne

- Diagnostic hors-ligne
  - On connaît toutes les observations *a priori*
  - Pas de contraintes sur le temps de réponse
  - En général on ne traite que les pannes permanentes

- Diagnostic en-ligne
  - Les observations sont acquises en ligne
    - Suivi, supervision, surveillance
  - Évolution du diagnostic en fonction de nouvelles observations
  - Temps de réponse contraint

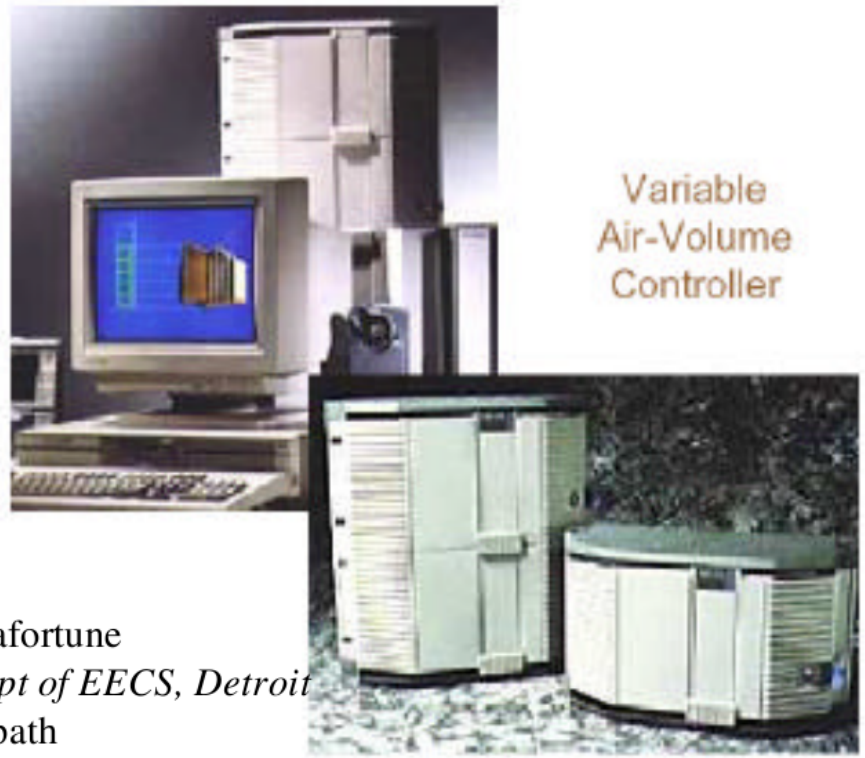
Plan du cours

Diagnostiqueur de Sampath

- Travaux publiés en 94-95-96
  - Meera Sampath et Stéphane Lafortune (Université du Michigan)
  - Parmi les initiateurs pour la caractérisation du problème de diagnostic dans les SED
- Objectifs : identification de l'occurrence d'événements de faute pouvant expliquer une séquence d'observations
- Modèle : proche de la théorie de la contrôlabilité [Ramadge et Wonham 89]
  - Ensemble d'automates synchronisés

Système d'air conditionné

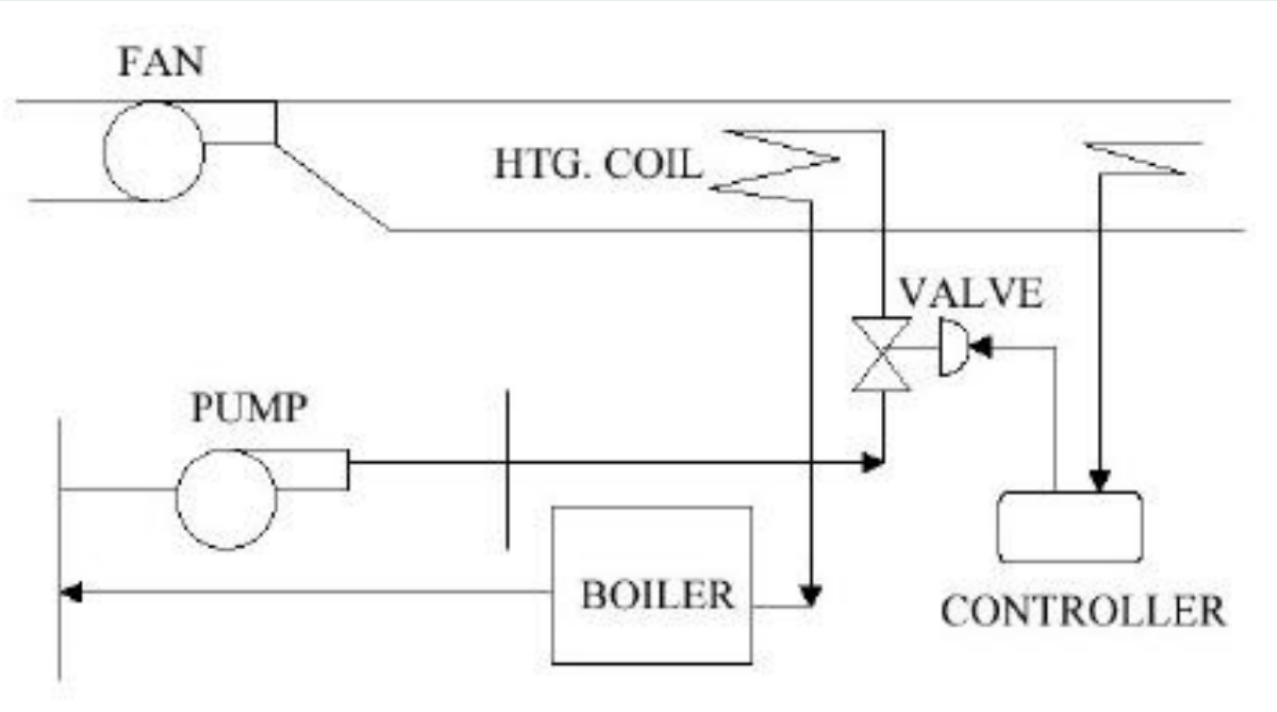
Exemple



Stéphane Lafortune
Dept of EECS, Detroit
Meera Sampath
Wilson Center for Research & Technology

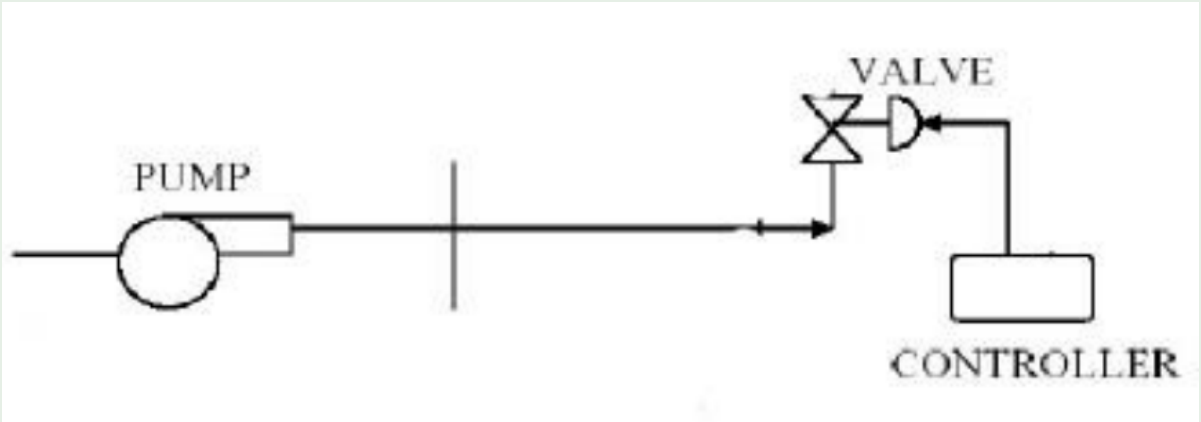
Système d'air conditionné (2)

Exemple



Système d'air conditionné (2)

Exemple



Système d'air conditionné (3)

Exemple

Évenements de pannes permanentes :

- Valve bloquée ouverte : événement spontané **bo**
- Valve bloquée fermée : événement spontané **bf**

Événements observables :

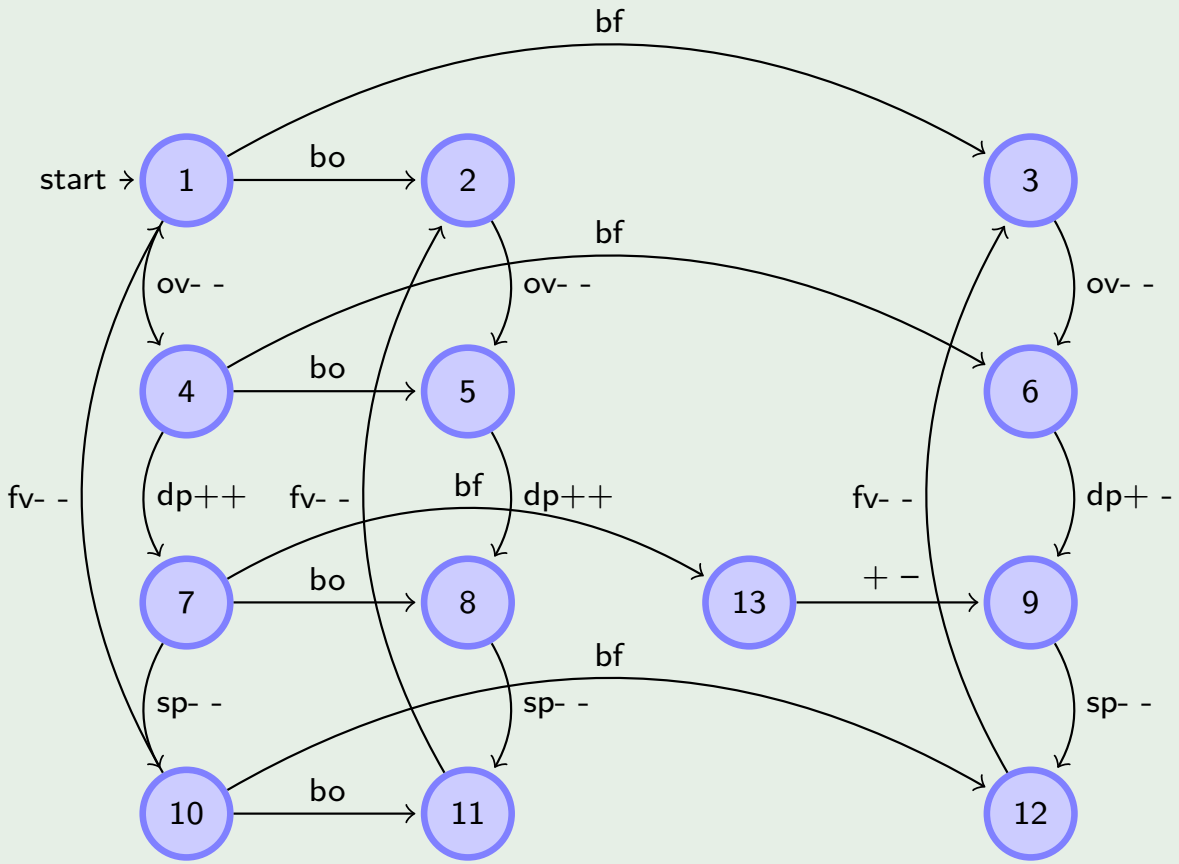
- Actions
  - **ov** : ouvre la valve, **fv** : ferme la valve
  - **dp** : démarre la pompe, **sp** : stoppe la pompe
- Observations d'un capteur de débit
  - ++ : présence d'un débit, - - : absence d'un débit
  - + - : coupure spontanée du débit

Dans la suite, événements composés : **ov**- - signifie "ouverture de la valve et absence de débit"

Système d'air conditionné : Modèle global

Exemple

bo : bloqué ouverte
bf : bloqué fermé
ov : ouvre valve
fv : ferme valve
dp : démarrre pompe
fp : ferme pompe
++ : débit
- - : pas débit
+- : plus aucun débit



Principe du diagnostic

Exemple

1. On suppose connu l'état initial :

$$\text{État initial} = 1$$

2. On observe une séquence d'observations :

$$\textbf{ov-}\ \textbf{-},\ \textbf{dp}\ \textbf{++}$$

3. On cherche à établir l'ensemble des états courants du système

$$\text{États courants (observateur)} = \{7, 8\}$$

4. On cherche à établir si des événements de pannes se sont produits

$$\text{États courants (diagnostiqueur)} = \{7\{\}, 8\{bo\}\}$$

Principe du diagnostiqueur

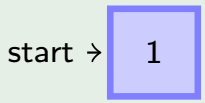
- Algorithme de diagnostic $\equiv$ Recherche de chemins de transitions dans le modèle
- Diagnostiqueur : machine à états finis
  - Précompilation hors-ligne de la recherche de chemins
  - Abstraction des événements non-observables
- Diagnostiqueur $\equiv$ Observateur enrichi
- Machine efficace
  - A chaque nouvelle observation, on tire une transition du diagnostiqueur
  - Le diagnostiqueur reconnaît le langage observable du système :

$$L_{OBS} = P_{OBS}(L(M))$$

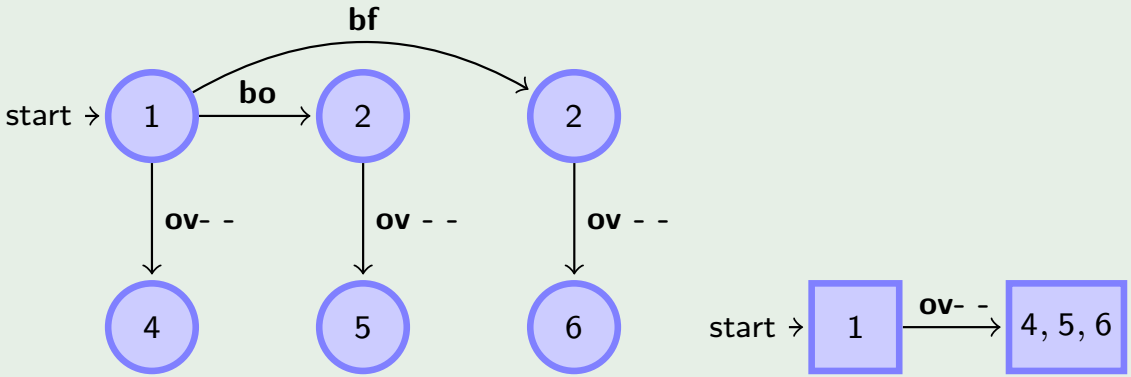
Construction de l'observateur

Exemple

1. Considérer l'état initial du modèle



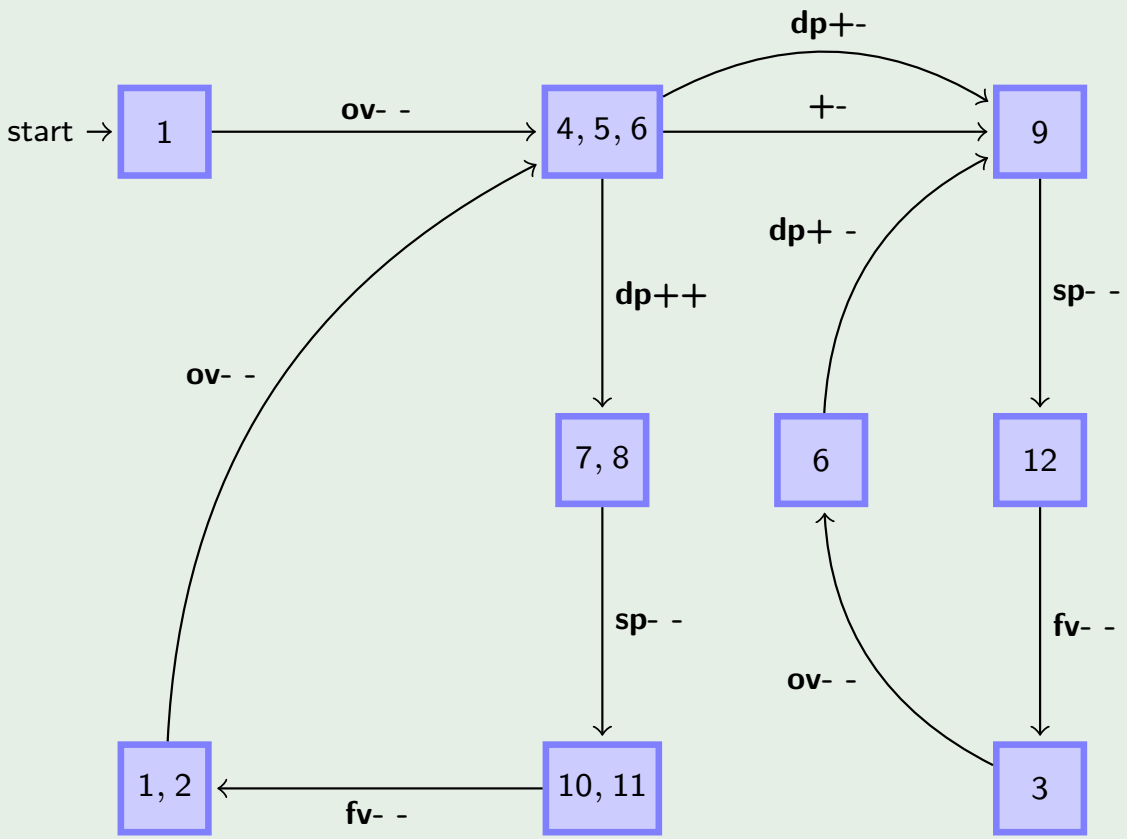
2. Rechercher à partir de ces états les chemins aboutissants à des états *Xo* cible d'une transition observable *o*



3. Considérer les états *Xo* nouvellement construits et réiterer 2 jusqu'à convergence

Observateur complet

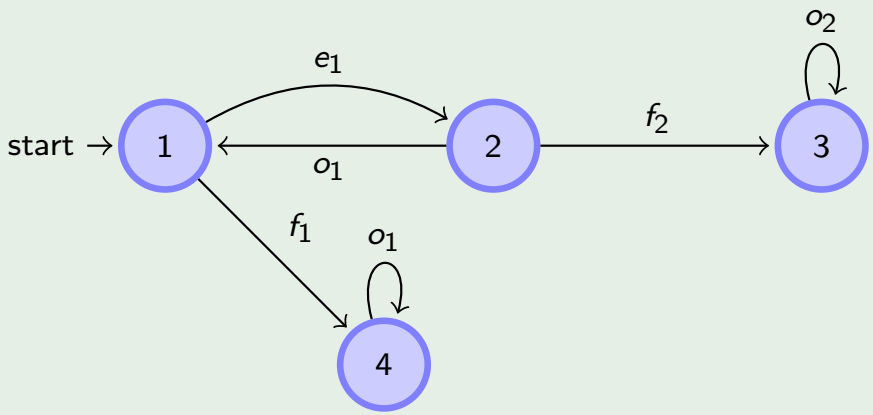
Exemple



À vous de jouer

Exemple

Modèle global :

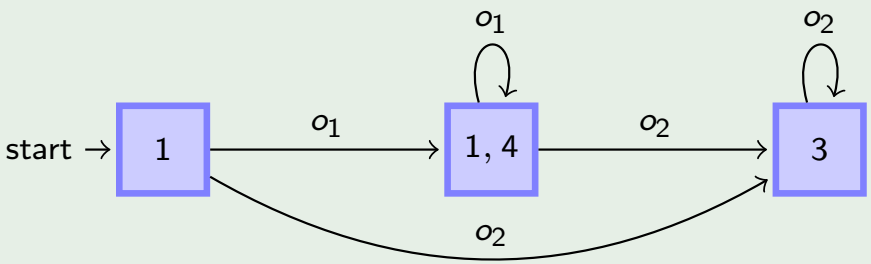


Calculer l'observateur de ce modèle : seuls o_1 ete o_2 sont observables

Résultat

Exemple

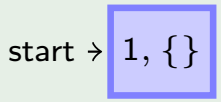
Observateur :



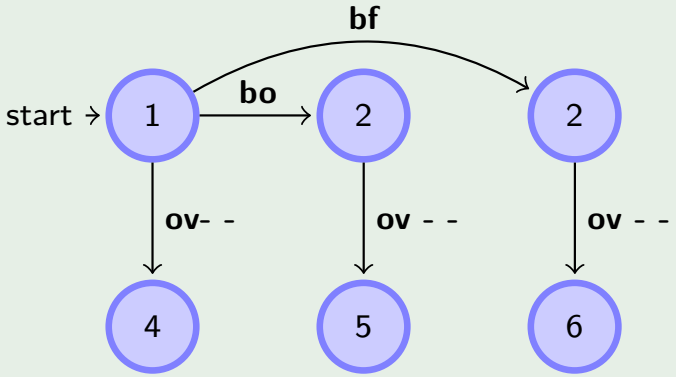
Construction du diagnostiqueur

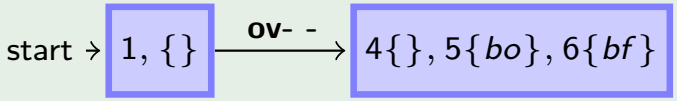
Exemple

1. Considérer l'état initial du modèle, pas de panne



2. Rechercher à partir de ces états les chemins aboutissants à des états *Xo* cible d'une transition observable *o* et propager les pannes

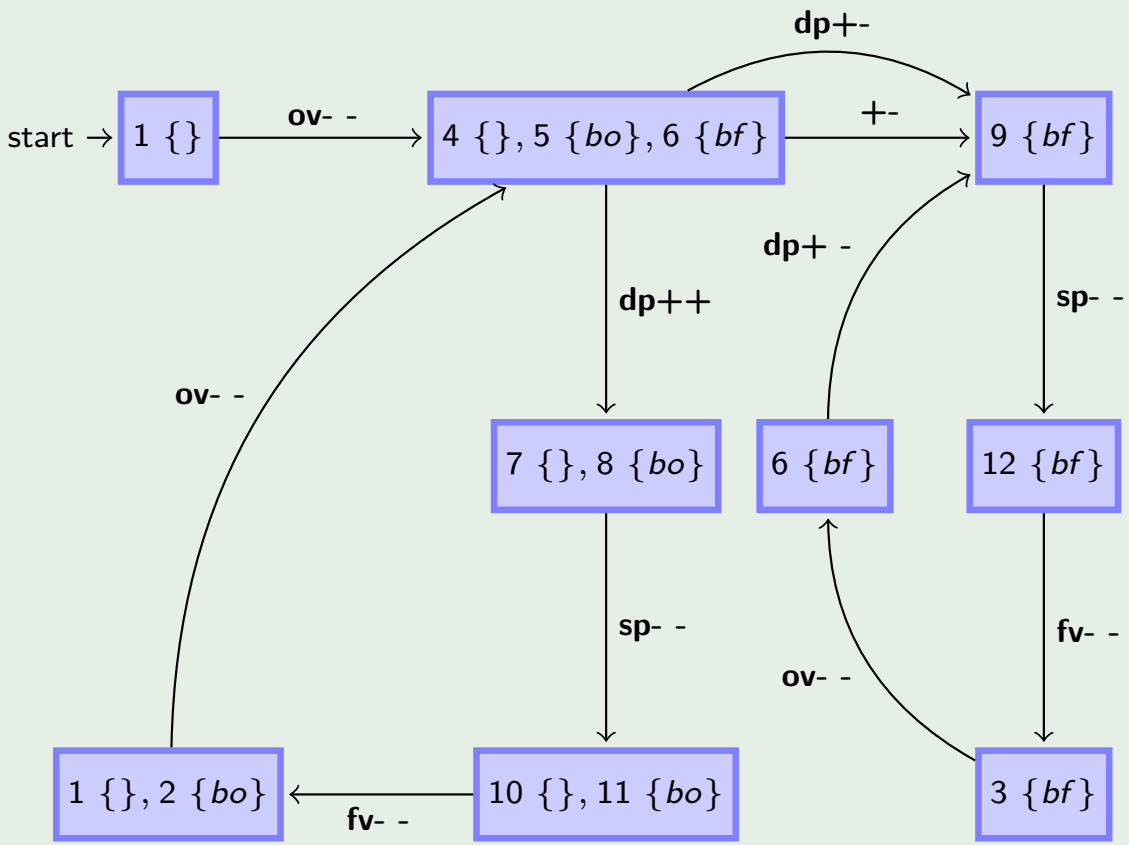




3. Considérer les états *Xo* nouvellement construits et réiterer 2 jusqu'à convergence

Diagnostiqueur complet

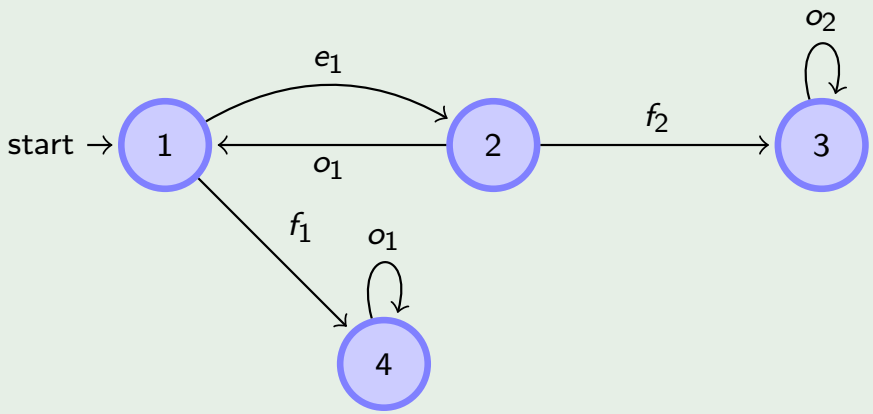
Exemple



À vous de jouer

Exemple

Modèle global :

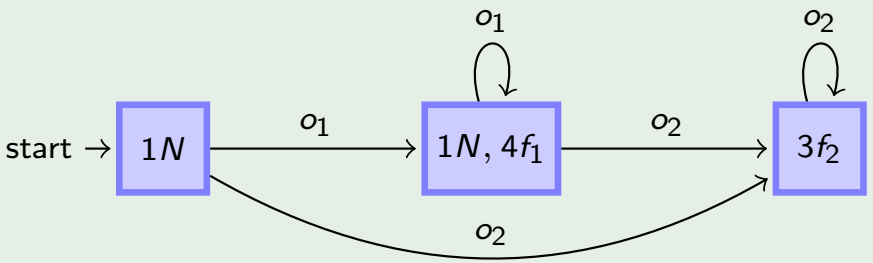


Calculer le diagnostiqueur de ce modèle : seuls o_1 ete o_2 sont observables. f_1 et f_2 sont les événements de pannes.

Résultat

Exemple

Diagnostiqueur :



Diagnostiqueur : la machine idéale et utopique

- Idéal : disposer d'un diagnostiqueur c'est :
  1. avoir une caractérisation complète d'un problème de diagnostic dans le SED
     - tout état du diagnostiqueur est un diagnostic possible
     - chaque diagnostic possible est un état du diagnostiqueur
  2. avoir un algorithme de diagnostic efficace (temps constant)
     - Adapter un diagnostic en fonction d'une nouvelle observation, c'est franchir une transition

- Utopique : Construire un diagnostiqueur, c'est :
  1. Disposer d'un modèle de panne (à base de composant) complet et correct
  2. Être capable de calculer le modèle global (produit synchronisé des composants)
  3. Être capable de calculer l'observateur associé

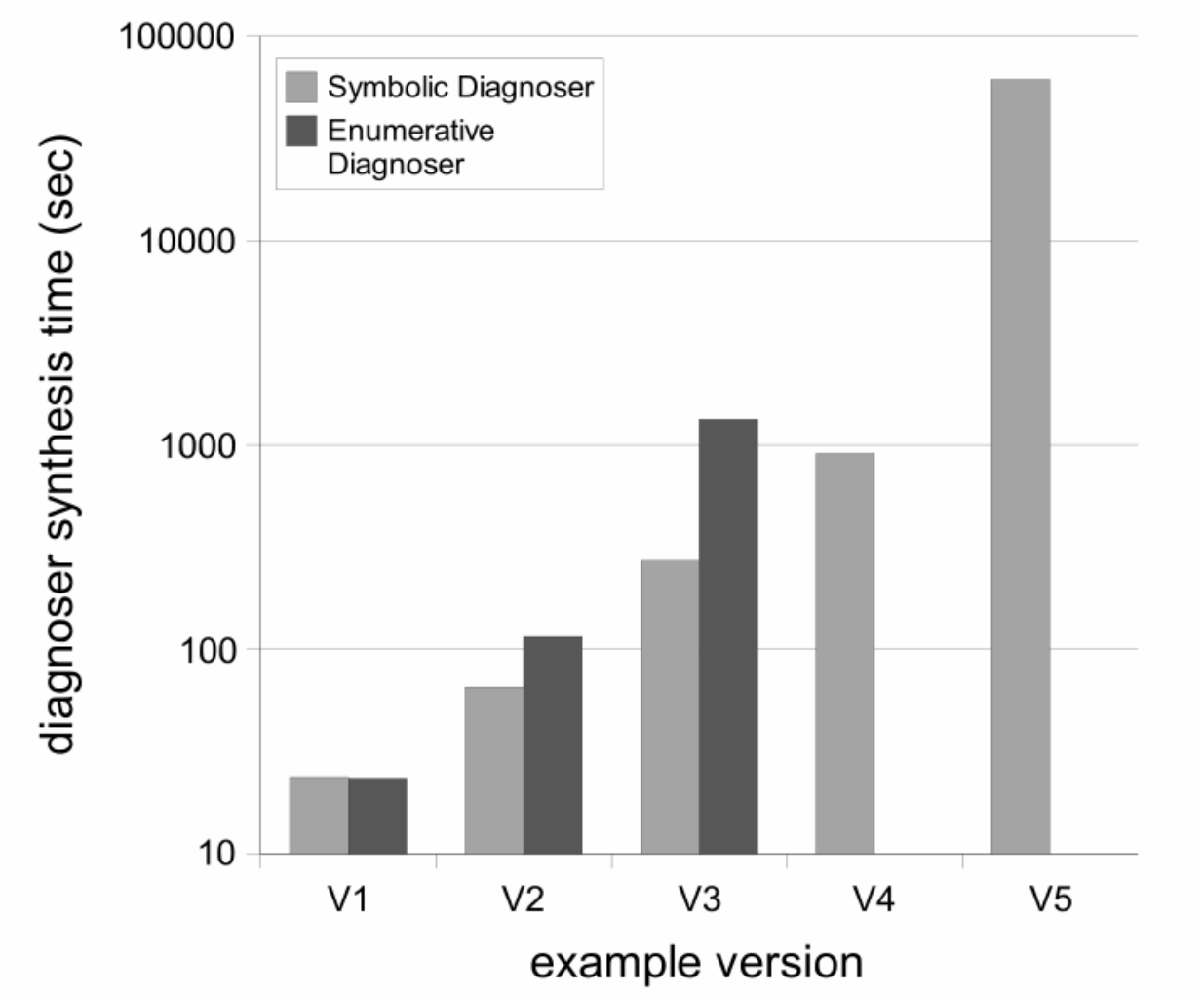
Un peu de complexité algorithmique

- Soit n le nombre de composants, on suppose que chaque composant contient m états, soit P le nombre de pannes dans le système
- Calculer le modèle global, c'est :
  - Déterminer tous ses états : au pire $N = m^n$ états !
  - Algorithme de calcul exponentiel en le nombre de composants : $O(2^n)$
- Calculer le diagnostiqueur, c'est :
  - Déterminer tous ses états : au pire $N_D = 2^N \times 2^P$ états !
  - Algorithme de calcul exponentiel en le nombre de pannes : $O(2^P)$
  - Doublement exponentiel en le nombre de composants : $O(2^{2^n})$

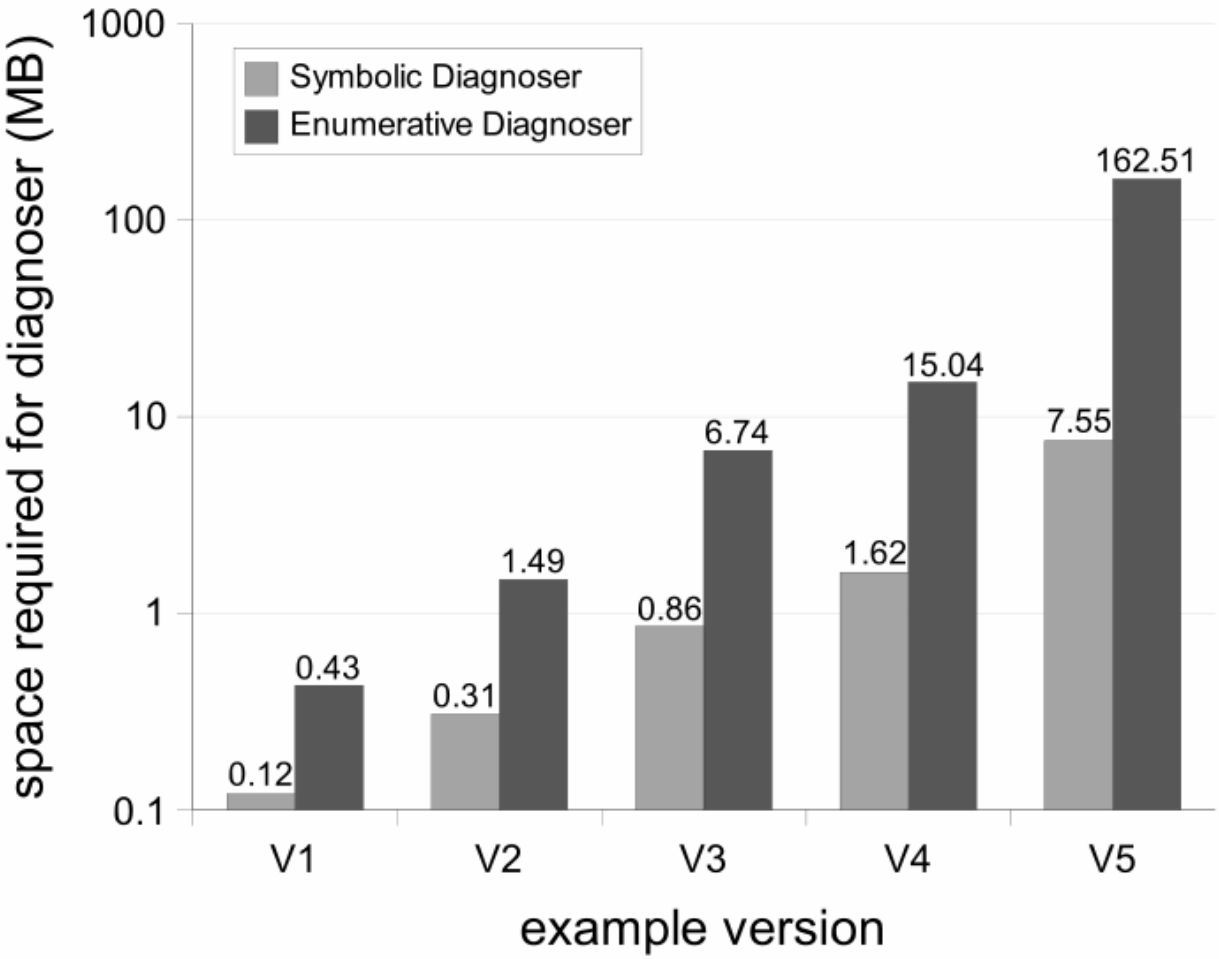
Exemple

Soit $n = 4$, $m = 10$, $f = 5$, au pire $N = 10000$ et $N_D = 10^{10000}$. A titre de comparaison, le nombre d'atomes dans l'univers n'est que de 10^{80}.

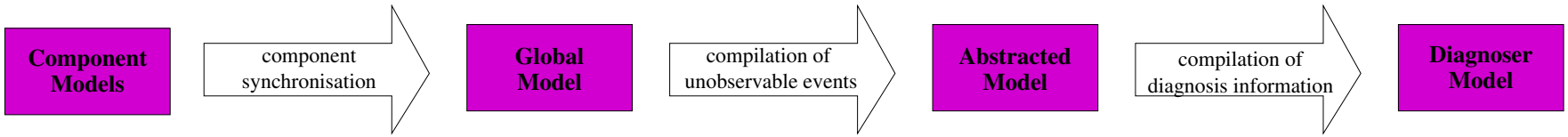
Complexité temporelle en pratique

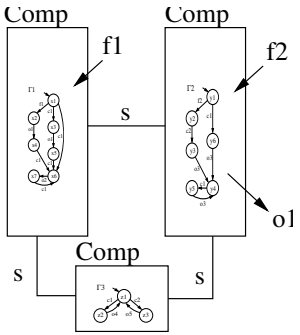


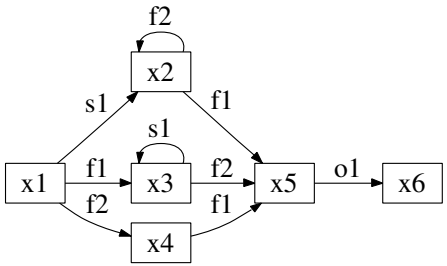
Complexité spatiale en pratique

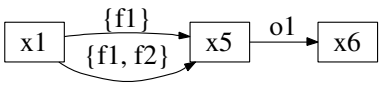


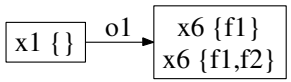
Spectre de modèles/Précompilation







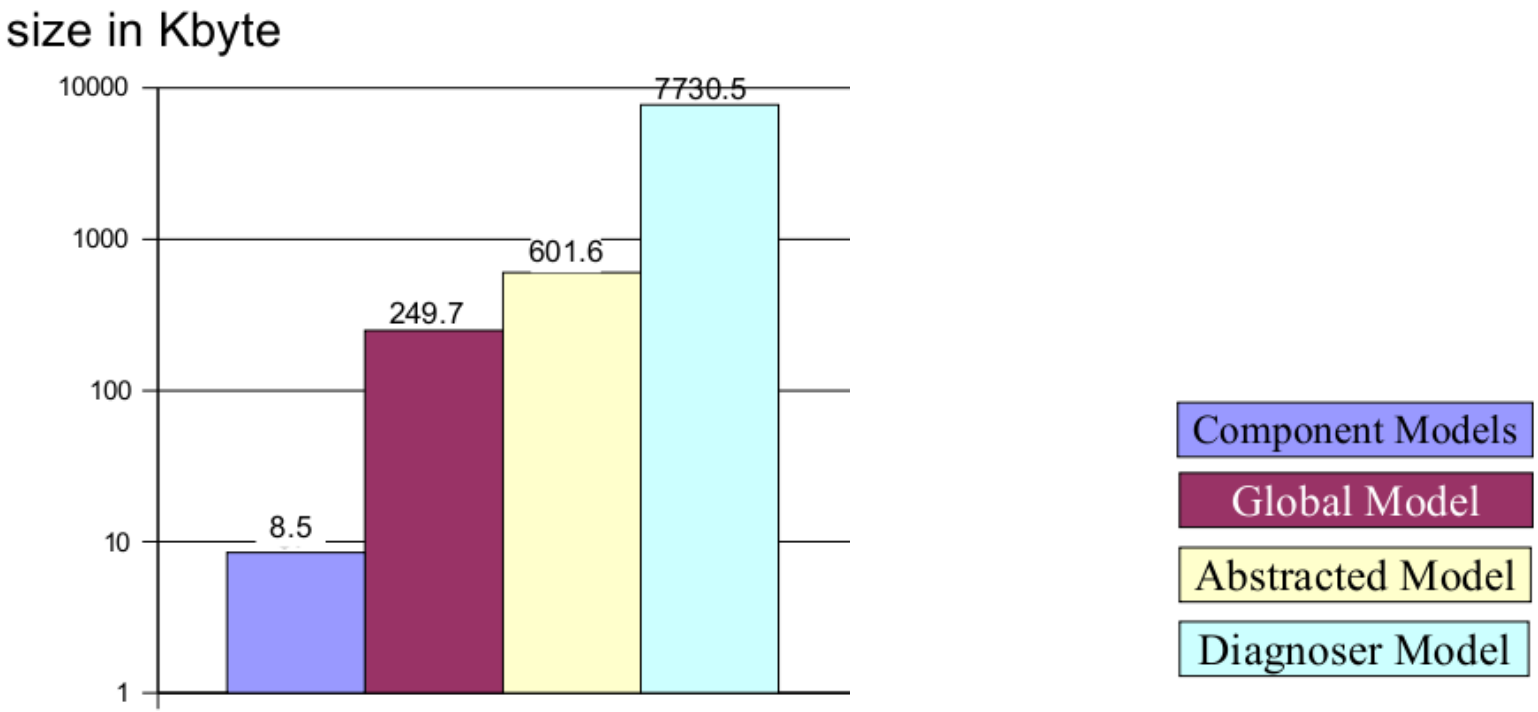




Spectre d'algorithmes et complexité

- Composant : algorithme dit de simulation (spatiale $O(n) = ++$, temporelle $O(2^n) = --$)
  - retrouver tous les chemins expliquant le flux d'observations à partir des modèles de composants
  - produit synchronisé en ligne
- Modèle global : (spatiale $O(2^n) = -$, temporelle $O(2^n) = -$)
  - retrouver tous les chemins expliquant le flux d'observations à partir du modèle global
  - produit synchronisé hors-ligne, déjà précalculé
- Modèle abstrait : modèle global abstrait (spatiale $O(2^n) = -$, temporelle $O(n) = +$)
  - parcours "à un pas" indéterministe d'une machine à état fini
- Diagnostiqueur : (spatiale $O(2^{2^n}) = --$, temporelle $O(1) = ++$)

Spectre d'algorithmes et complexité spatiale (2)



Plan du cours

Extension du diagnostiqueur : diagnostic de motif

- Diagnostiqueur classique : identification de l'occurrence de pannes permanentes
- Idée : généraliser le concept de diagnostic de panne en terme de motif de pannes
- Motif de pannes
  - Propriété temporelle sur l'occurrence d'un ensemble d'événements

Définition de motifs

Définition

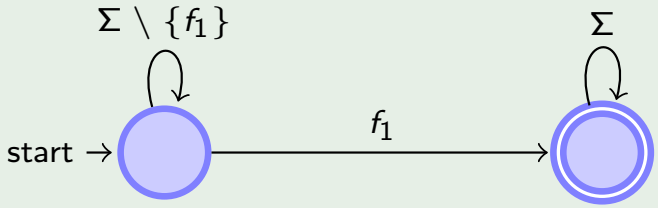
Un motif Mf est un langage sur l'alphabet Σ tel que :

- $\forall w \in Mf, \forall v \in \Sigma^\star, w.v \in Mf$: tout mot dont un préfixe est dans Mf, est dans Mf
- $\forall w \in \Sigma^\star, \exists v \in \Sigma^\star, w.v \in Mf$: pour tout mot w de $\Sigma^\star$, il existe un mot dans Mf dont w est un préfixe.

Exemple

Occurrence d'au moins un événement de faute f_1

$$Mf = (\Sigma \setminus \{f_1\})^\star.f_1.(\Sigma)^\star$$

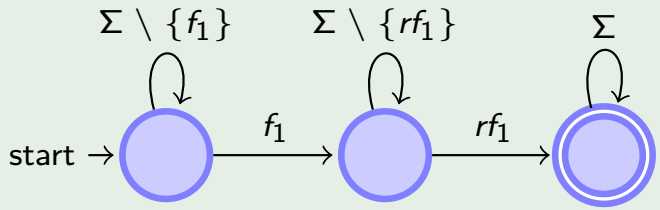


Définition de motifs (2)

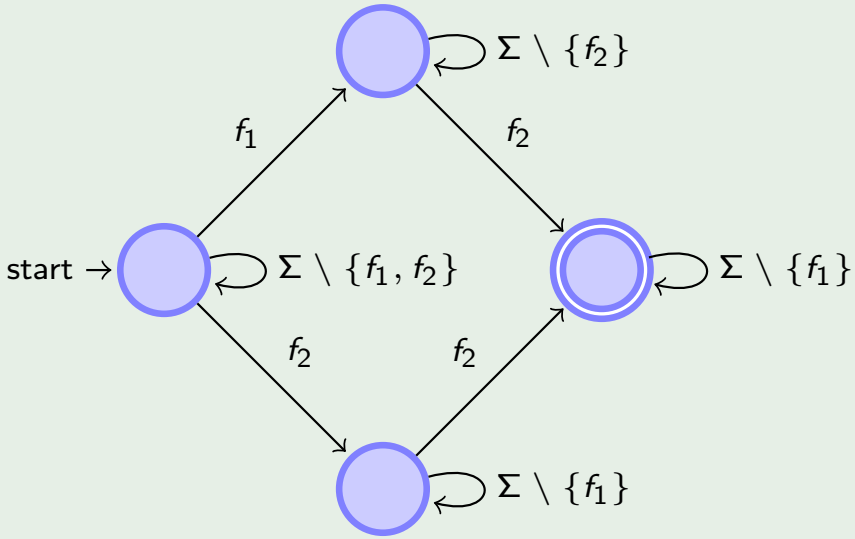
Exemple

Occurrence d'au moins une faute intermittente f_1

$$Mf = (\Sigma \setminus \{f_1\})^\star.f_1.(\Sigma \setminus \{rf_1\})^\star.rf_1.(\Sigma)^\star$$



Occurrence d'au moins deux fautes f_1 et f_2



Diagnostiqueur de motif

Définition

Soit $L(M)$ le langage associé au modèle M et Mf un motif, le langage $L(M)||_{\Sigma}Mf = L(M) \cap Mf$ est le langage de reconnaissance de Mf.

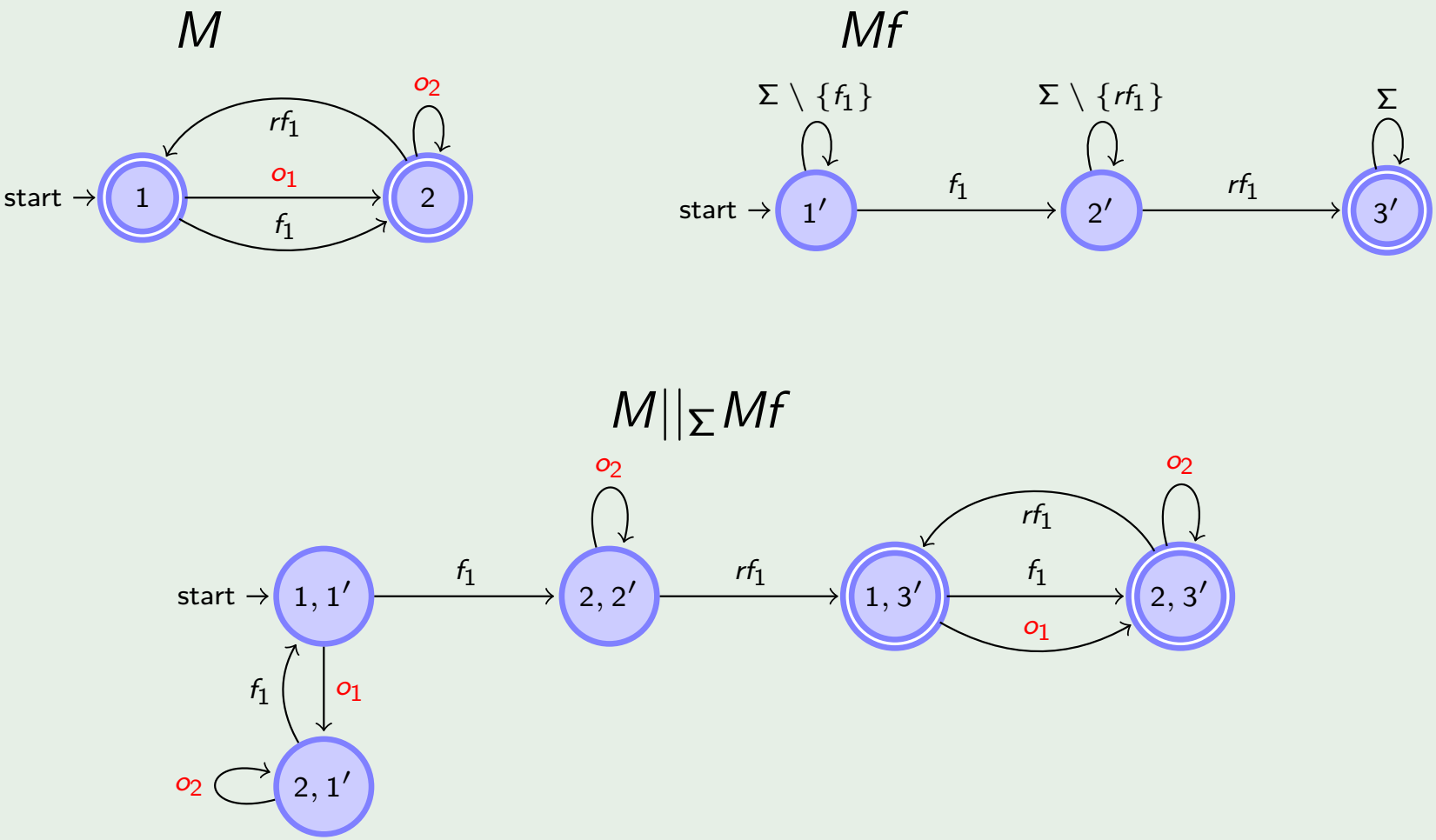
- Intersection des langages $\equiv$ produit synchrone du modèle M et d'un motif Mf (automate)
- Chaque état de $M||_{\Sigma}Mf$ est du type : $(x, reconnu)$
  - x est un état de M
  - $reconnu$ est un booléen (vrai si le motif est reconnu, faux sinon)

Définition

Le diagnostiqueur du motif Mf est l'observateur de $M||_{\Sigma}Mf$.

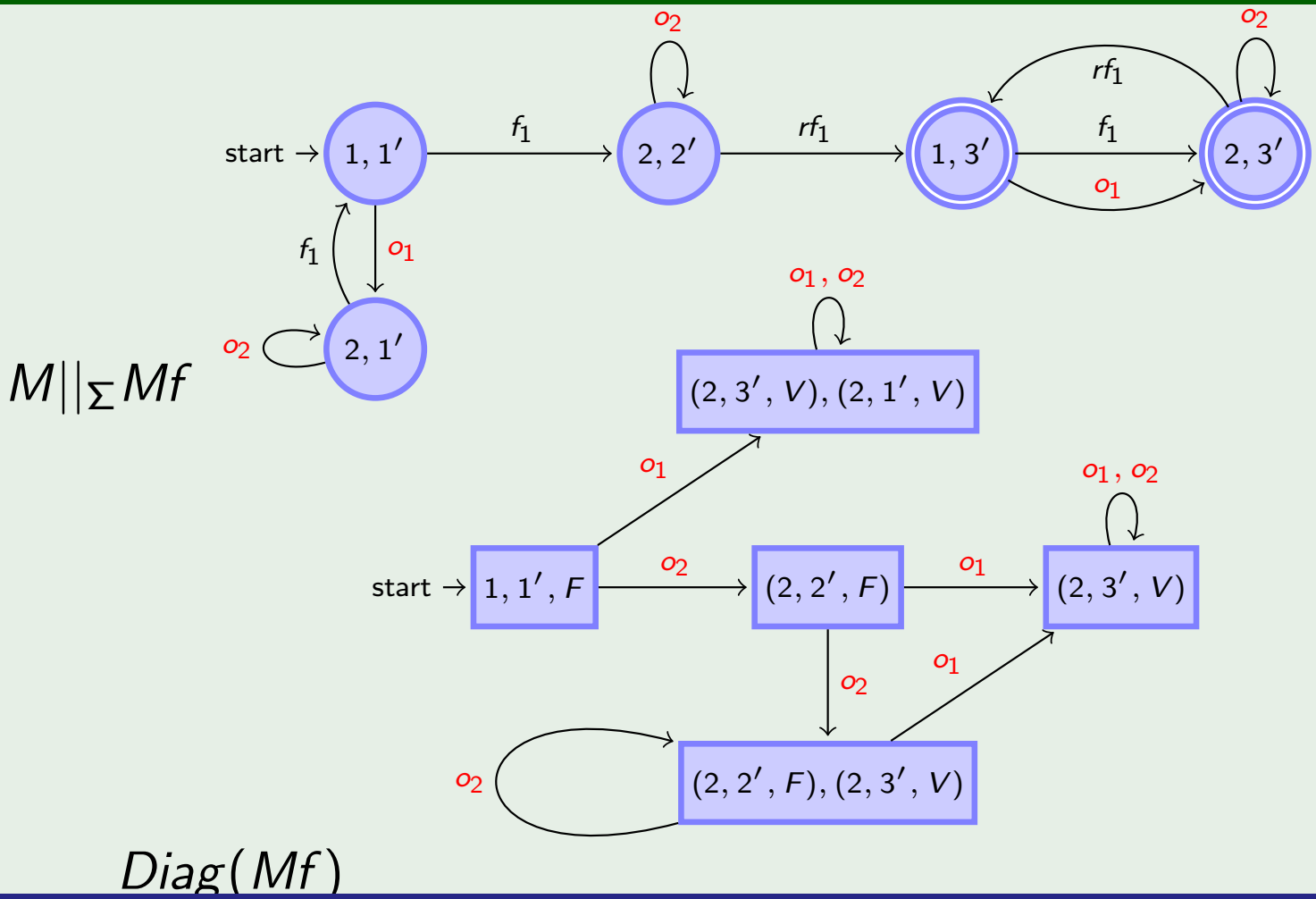
Diagnostiqueur de motif (2)

Exemple



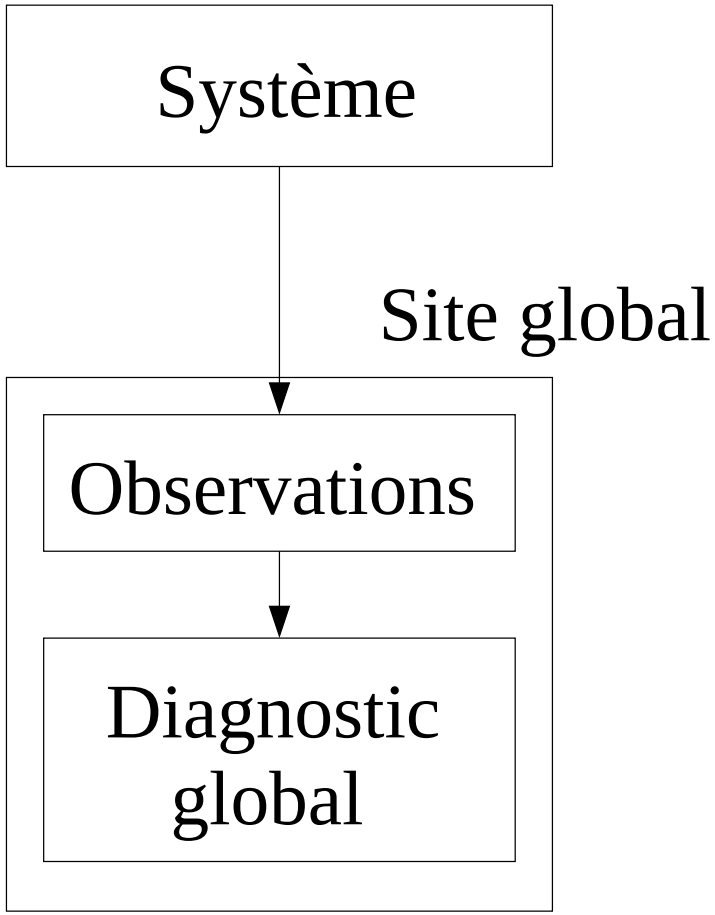
Diagnostiqueur de motif (3)

Exemple

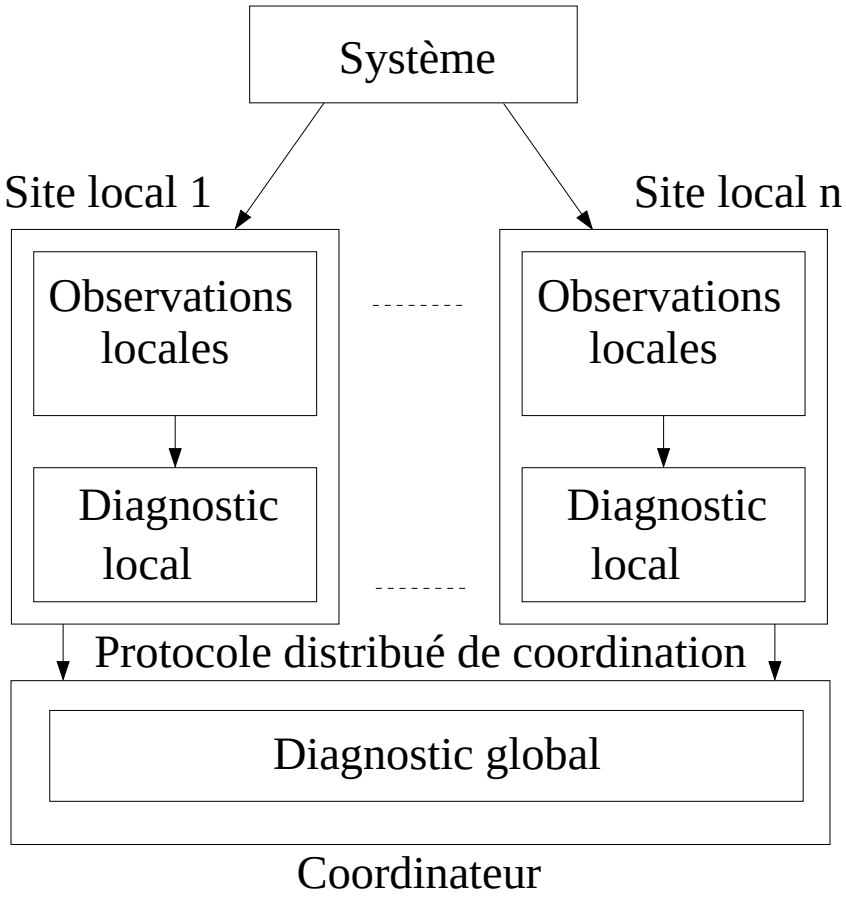


Plan du cours

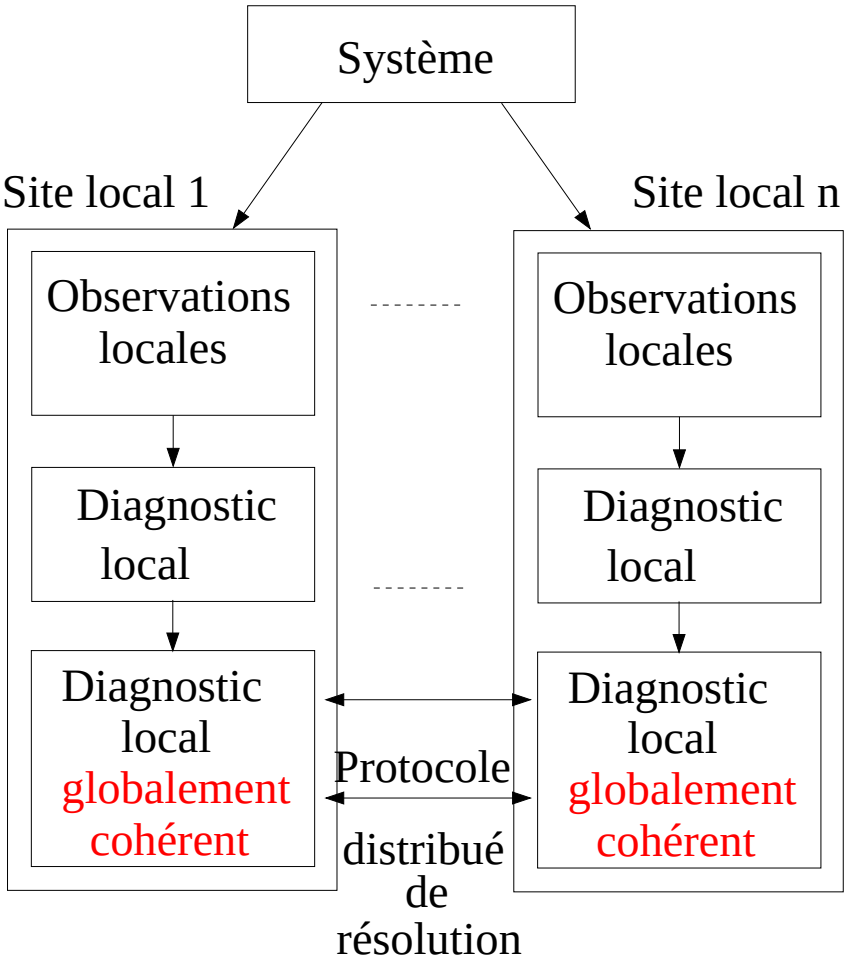
Architecture de diagnostic centralisée



Architecture de diagnostic coordonnée



Architecture de diagnostic distribuée



Plan du cours

Approche coordonnée de diagnostiqueurs globaux

- Un système surveillé par n sites
- Un modèle global $G = (X, T, \Sigma_o \cup \Sigma_{uo}, x_0)$
- Un site i = un sous-ensemble d'observations $\Sigma_i \subseteq \Sigma_o$
- Un diagnostiqueur Δ_i par site i
  - extension d'un diagnostiqueur classique mais ne voit que les observations d'un site

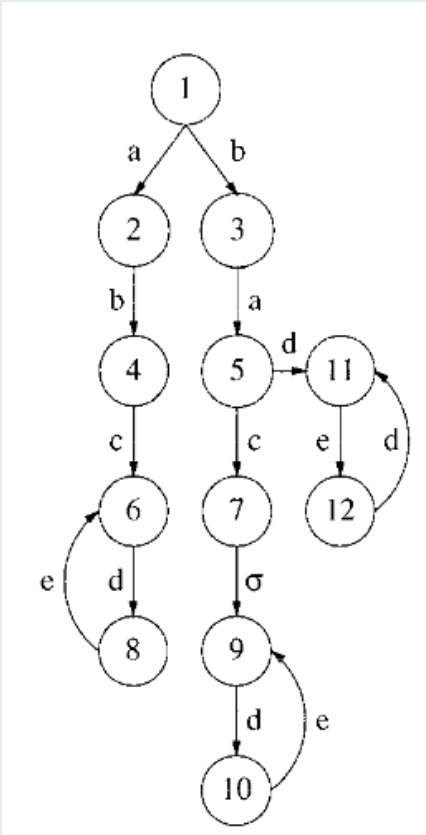
$$L(\Delta_i) = P_{\Sigma_i}(L(G))$$

- Coordinateur : protocole de décision afin d'obtenir le diagnostic global en fonction des propositions locales

Approche coordonnée de diagnostiqueurs globaux (2)

Exemple

Système et deux sites d'observations



Faute F_1 : représentée par l'occurrence de σ (non observable)

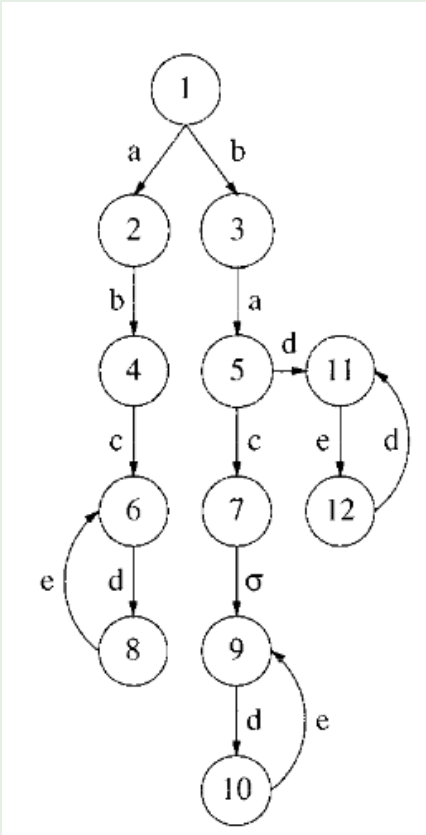
Observations du site 1 : a, c, d, e

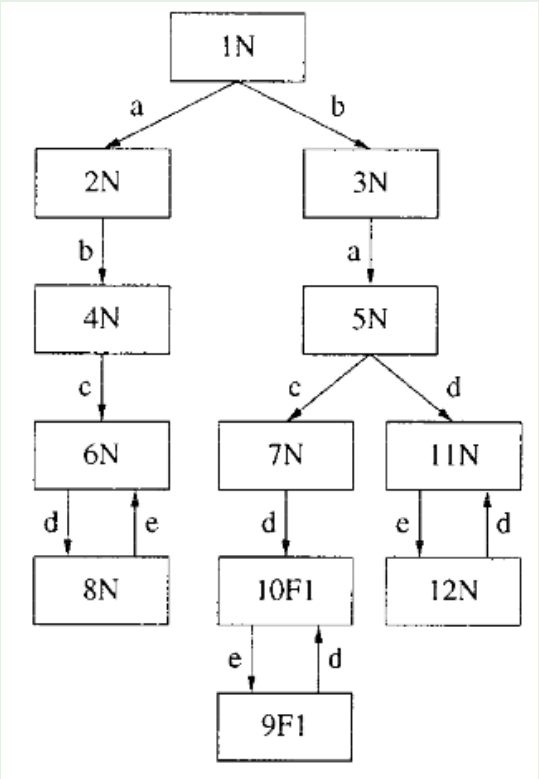
Observations du site 2 : b, d, e

Approche coordonnée de diagnostiqueurs globaux (3)

Exemple

Diagnostiqueur classique

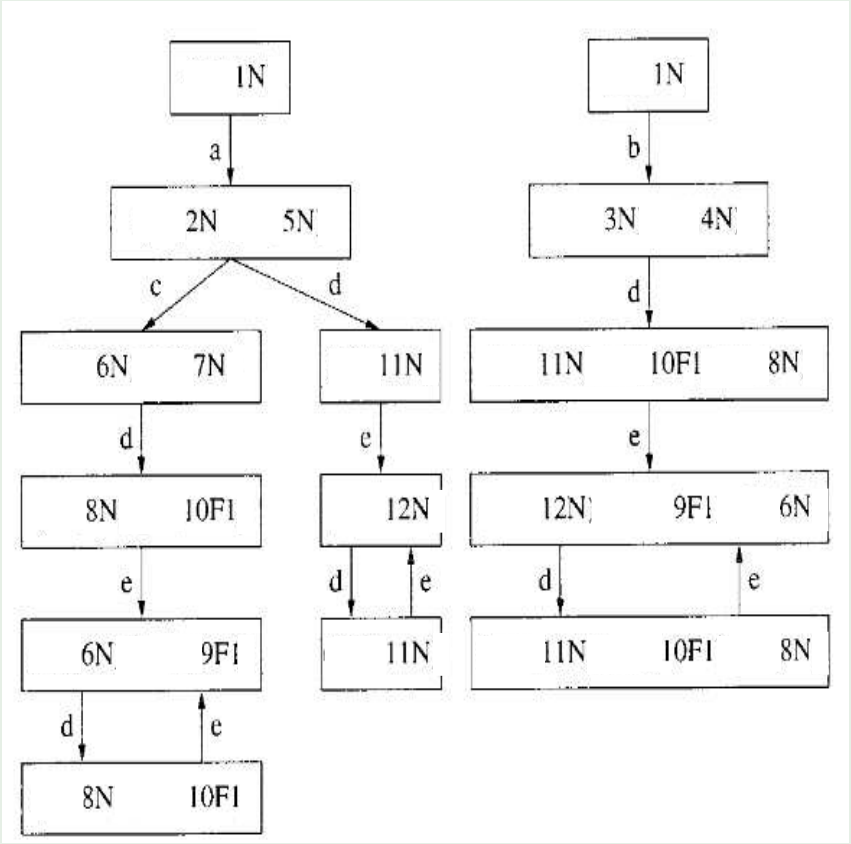




Approche coordonnée de diagnostiqueurs globaux (4)

Exemple

Diagnostiqueurs classiques des deux sites



Approche coordonnée de diagnostiqueurs globaux (5)

Un protocole naïf : attendre un diagnostic local et dire qu'il est global.

Exemple

Le système émet *bacd*.

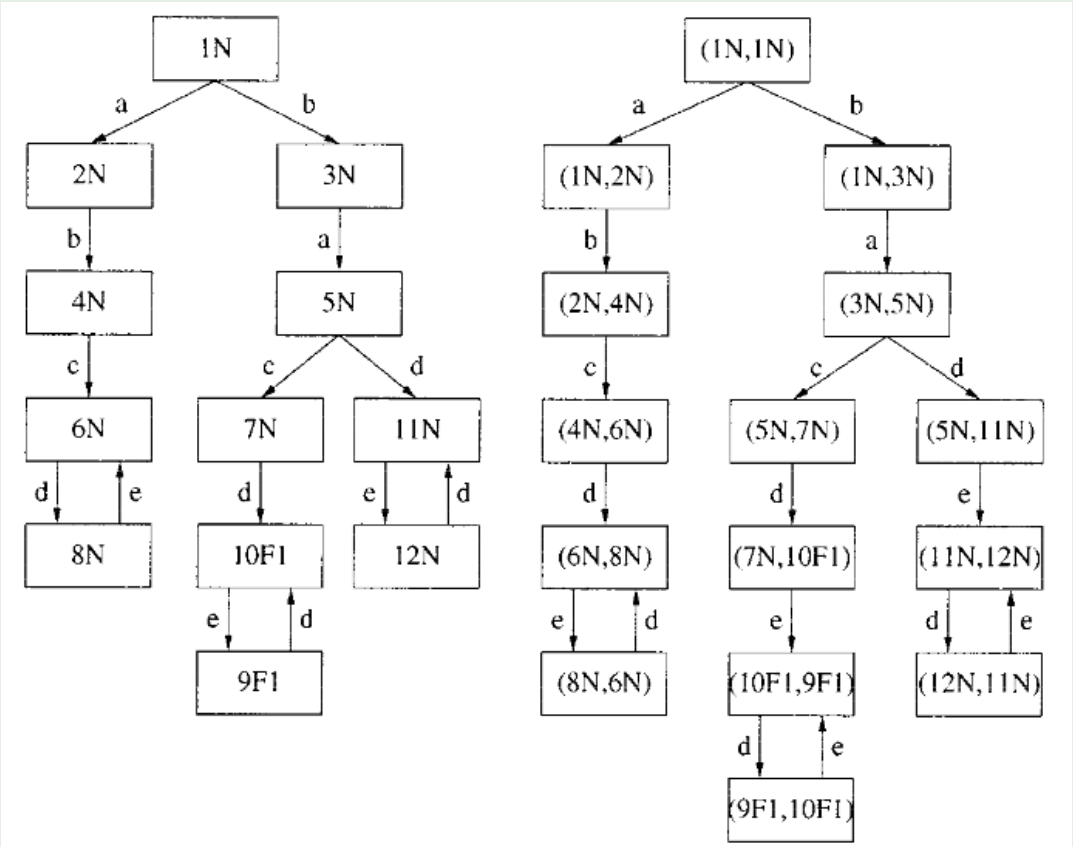
1. Site 2 voit b, il répond $3N, 4N$
2. Site 1 voit a, il répond $2N, 5N$
3. Site 1 voit c, il répond $6N, 7N$
4. Site 1 et 2 voient d, ils répondent $8N, 10F1$ et $11N, 10F1, 8N$
   - Ils s'accordent sur **8N**, **10F1** (donc ambiguité)

Le diagnostiqueur global répond **10F1**. Le protocole est correct mais imprécis.

Approche coordonnée de diagnostiqueurs globaux (6)

Exemple

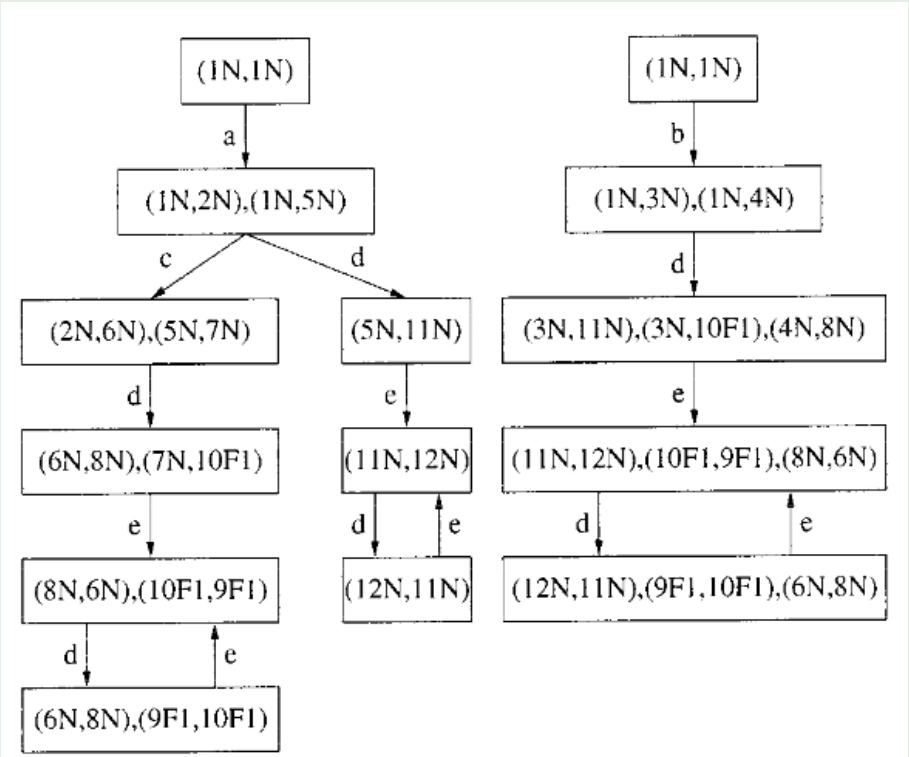
Diagnostiqueur étendu



Approche coordonnée de diagnostiqueurs globaux (7)

Exemple

Diagnostiqueurs étendus par site



Approche coordonnée de diagnostiqueurs globaux (8)

Protocole : utilisation des diagnostiqueurs étendus

Exemple

Le système émet **bacd**.

1. **Site 2 voit b**, il répond $1N \leftarrow 3N, 1N \leftarrow 4N$, le coordinateur dit **1N** $\leftarrow$ **3N** car 1 n'a rien dit et $4N$ n'est possible que si Site 1 a vu a
2. **Site 1 voit a**, il répond $1N \leftarrow 2N, 1N \leftarrow 5N$, le coordinateur dit **3N** $\leftarrow$ **5N** car $2N$ ne succède pas $3N$.
3. **Site 1 voit c**, il répond $2N \leftarrow 6N, 5N \leftarrow 7N$, le coordinateur dit **5N** $\leftarrow$ **7N**.

Approche coordonnée de diagnostiqueurs globaux (9)

Protocole : utilisation des diagnostiqueurs étendus

Exemple

4 **Site 1 et 2 voient d**, ils répondent $6N \leftarrow 8N, 7N \leftarrow 10F1$ et $3N \leftarrow 11N, 3N \leftarrow 10F1, 4N \leftarrow 8N$

- Site 2 répond $3N \leftarrow 11N, 3N \leftarrow 10F1, 4N \leftarrow 8N$. Le coordinateur dit $3N \leftarrow 11N, 3N \leftarrow 10F1$ car $7N$ ne succède pas $4N$.
- Site 1 répond $6N \leftarrow 8N, 7N \leftarrow 10F1$, le coordinateur dit $7N \leftarrow 10F1$
- Ils s'accordent sur **10F1**

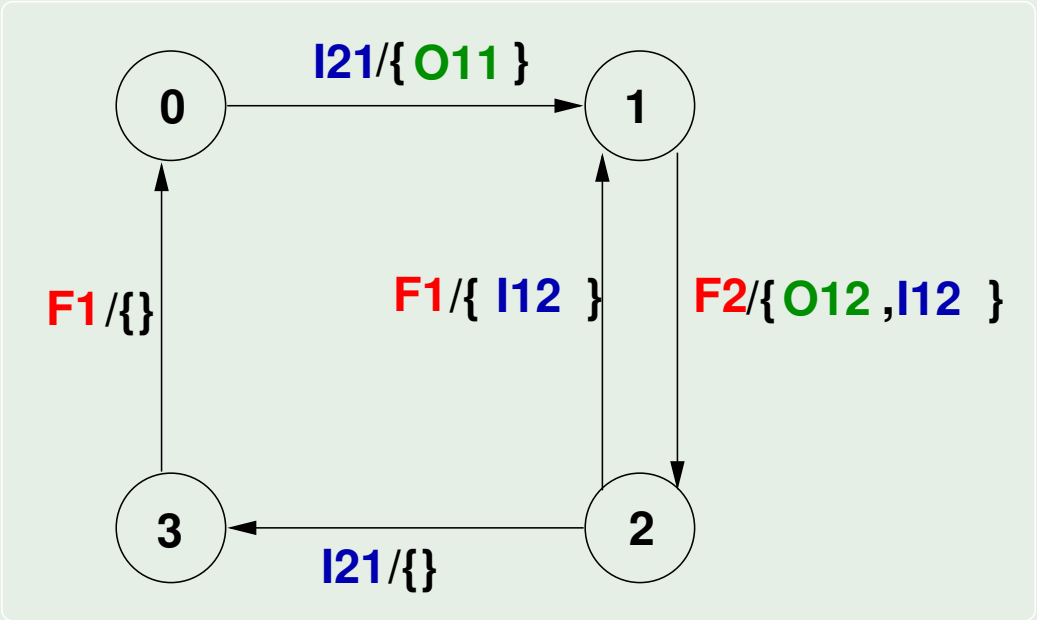
Résumé de l'approche

- Prise d'un nombre de site d'observations
- Chaque site a une connaissance globale du système
  - Problème de complexité spatiale
- Nécessite un protocole de coordinations qui s'appuie sur le modèle global (recherche d'atteignabilité d'états non observables)
- Plusieurs sites mais une horloge globale
  - Le flux d'observations est totalement ordonné

Plan du cours

Modèle de composants : exemple à base d'automates communicants

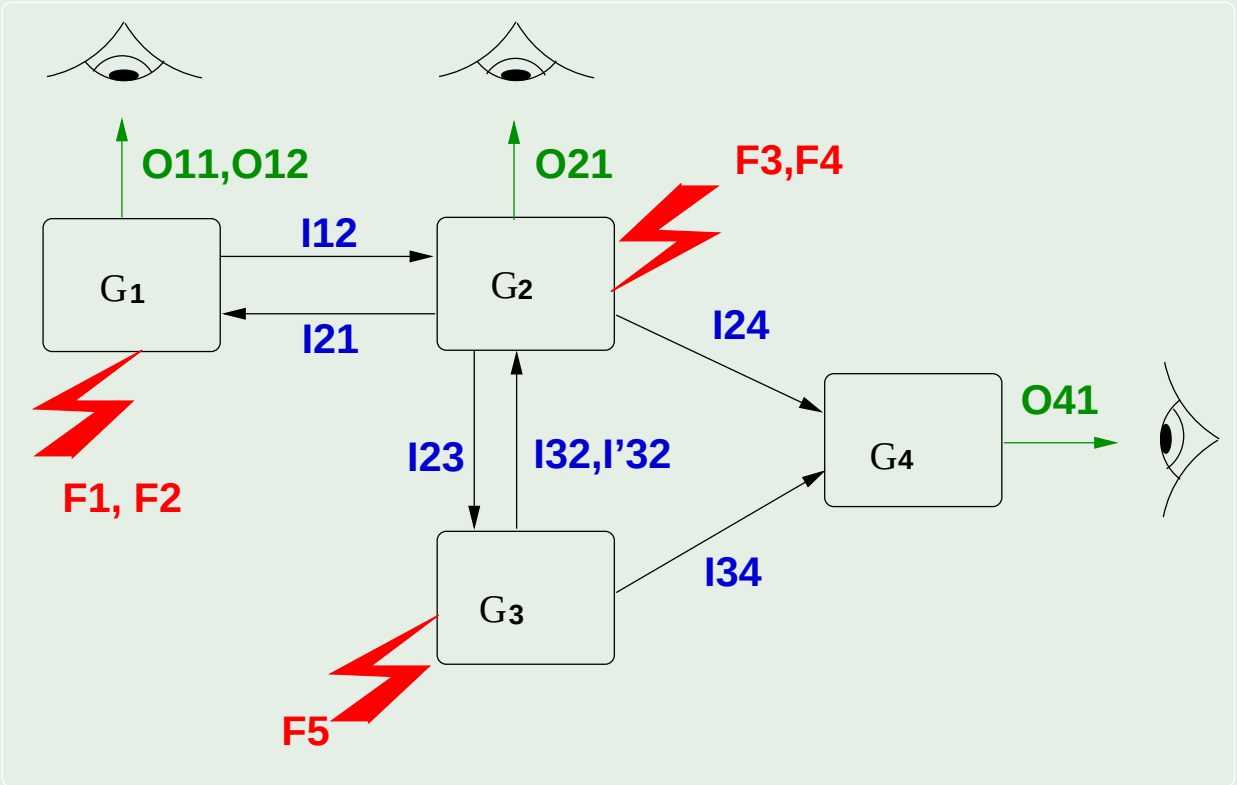
Exemple



- Exo_i : F1 F2, Rcv_i : I21
- $Emit_i$: I12, Obs_i : O11 O12

Modèle du système

Exemple



Modèle global : synchronisation sur les événements internes :

$G = G_1 ||_{\Sigma_{int}} \ldots ||_{\Sigma_{int}} G_n$

Méthodes décentralisées

- Principe : *Diviser pour régner*
- Autrement dit : moins j'en fais mieux je porte !
- *Diviser* :
  - Calcul du diagnostic pour un sous-système seulement $\Delta_{\gamma_1}(\mathcal{O}_{\gamma_1}), \ldots, \Delta_{\gamma_m}(\mathcal{O}_{\gamma_m})$
    - explique les *observations* $\mathcal{O}_{\gamma_i}$ *du sous-système* $\gamma_i = \{G_{i_1}, \ldots, G_{i_k}\}$ par des comportements du sous-système
- *Régner* :
  - Fusion des *diagnostics locaux* : diagnostic global

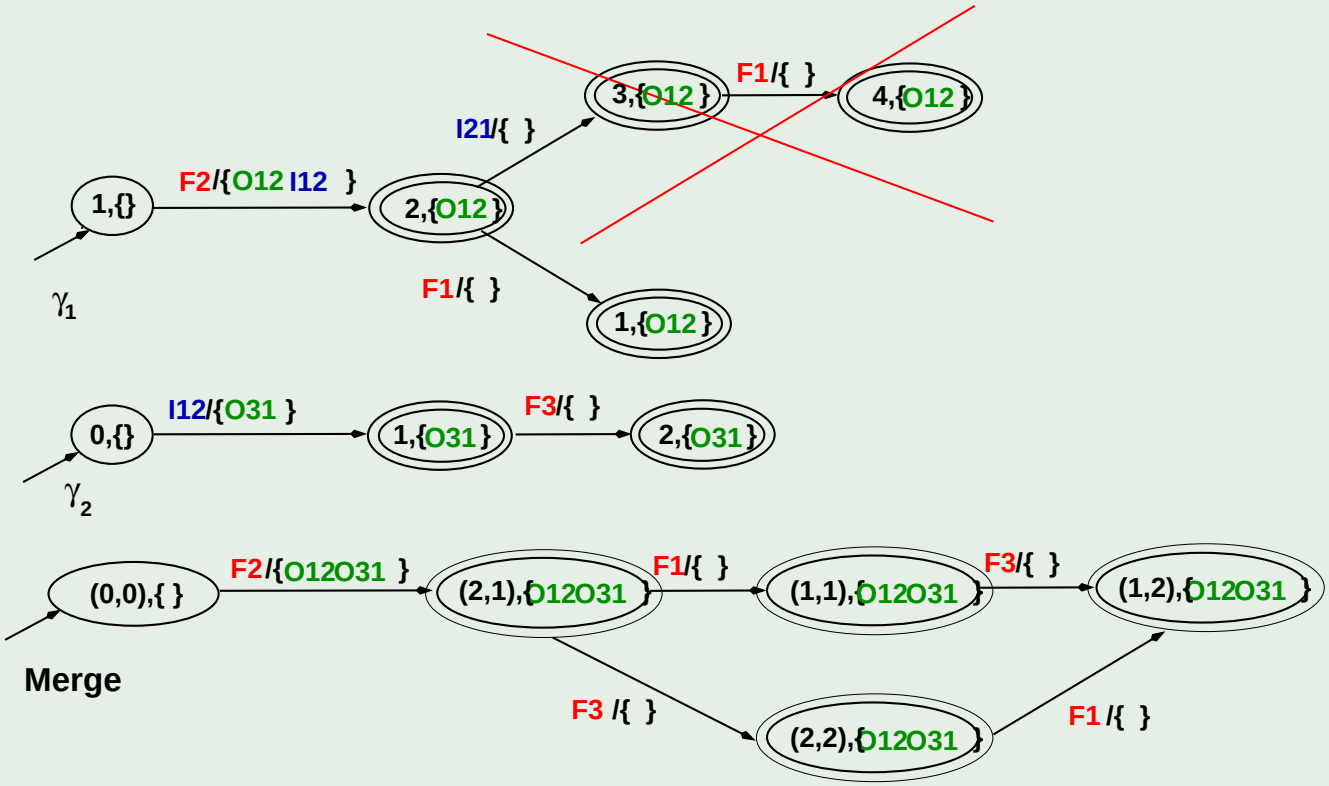
$$\Delta(\mathcal{O}) = \mathit{Merge}(\Delta_{\gamma_1}(\mathcal{O}_{\gamma_1}), \ldots, \Delta_{\gamma_m}(\mathcal{O}_{\gamma_m}))$$

  - Objectif de la fusion : vérifier les interactions entre les diagnostics locaux

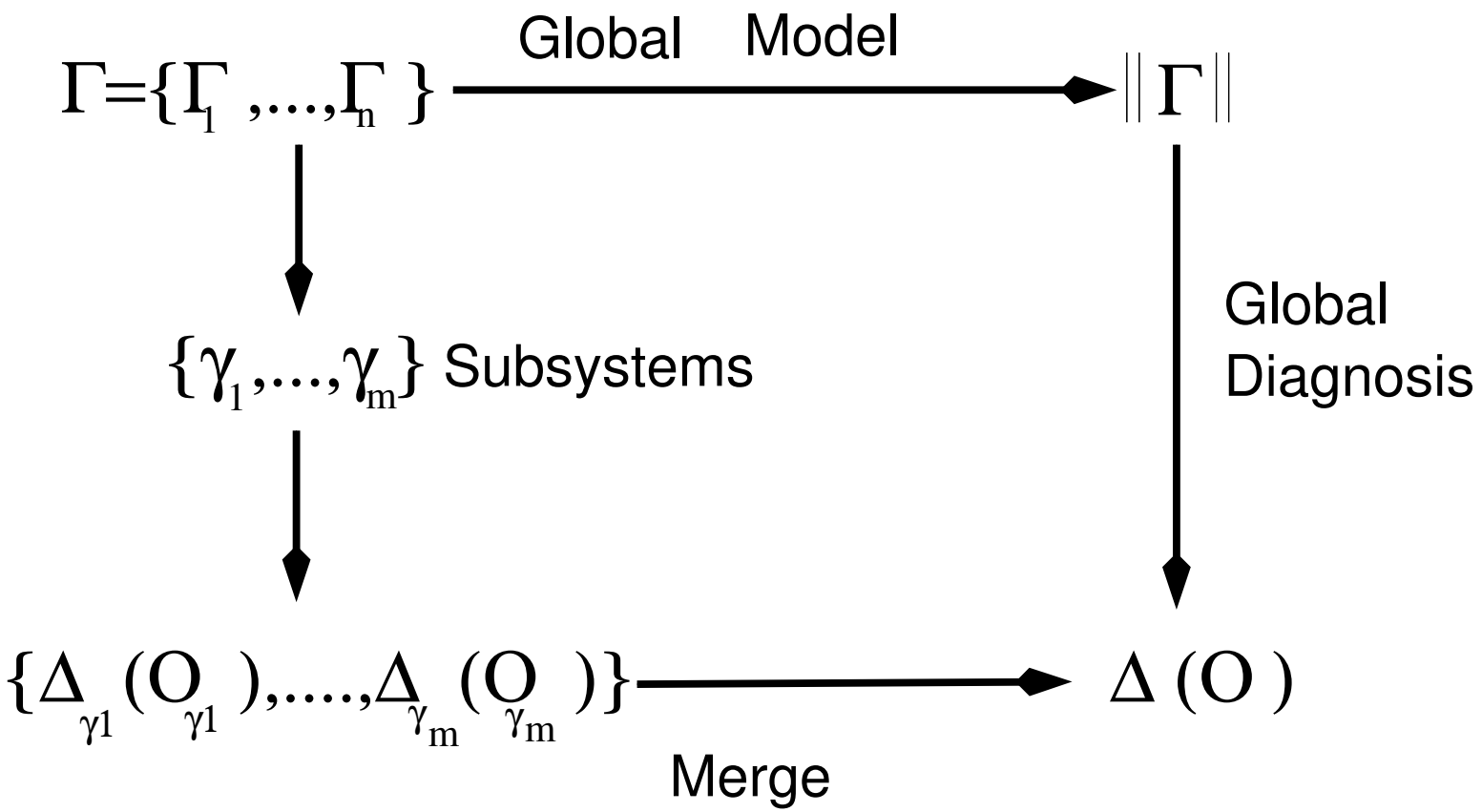
Operation de fusion : exemple

Exemple

J'observe $O12, O31$ sur 2 sites



Centralisé/Decentralisé



Avantages d'une approche décentralisée

- Le *modèle global* n'est pas nécessaire
  - Utilisation de modèles à taille réaliste
- Systèmes supervisés : de nature distribuée
  - Plus adapté à l'évolution à la reconfiguration du système

Méthodes décentralisées : travaux antérieurs

- Diagnostic des "systèmes actifs" (active systems) [Baroni *et al.*]
  - *Simulation* d'un modèle décentralisé Γ prenant en compte les observations
  - *Simulation* par sous-système et généralisation de la simulation (*Merge*)
  - Inconvénient : opération complexe, méthode hors-ligne uniquement

Stratégie de fusion

- L'opération de synchronisation est fondé sur un produit synchronisé : pas efficace !
  - ne l'utiliser que si nécessaire.
- Stratégie fondée sur l'information contenue dans les diagnostics locaux
  - Règle 1 : Détection de chemins incompatibles
  - Règle 2 : Sélection de diagnostics dépendants

Règle 1 : Détection de chemins incompatibles

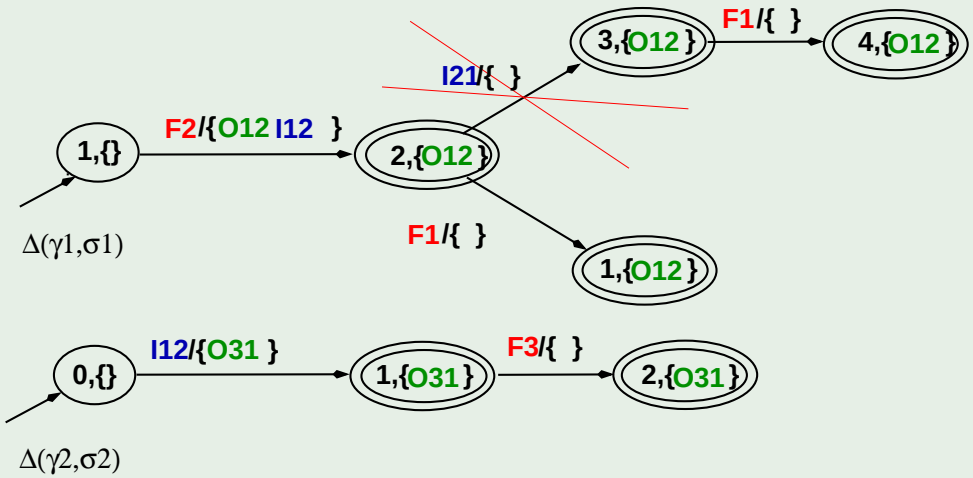
- Soit E_i l'ensemble des événements échangés (interactions) par le sous-système γ_i selon son diagnostic local $\Delta_{\gamma_i}(\sigma_i)$;
- Tout événement échangé entre γ_i et γ_j est nécessairement tel que :

$$e \in E_i \cap E_j$$

- **Règle 1** : si un chemin C de $\Delta_{\gamma_i}(\sigma_i)$ contient une interaction $e \notin E_i \cap E_j$ alors C est incompatible, inutile de synchroniser C avec $\Delta_{\gamma_j}(\sigma_j)$
  - Protocole de coordination : échange d'événements internes

Règle 1 : Détection de chemins incompatibles (2)

Exemple



$E_1 = \{I12, I21\}$
$E_2 = \{I12\}$

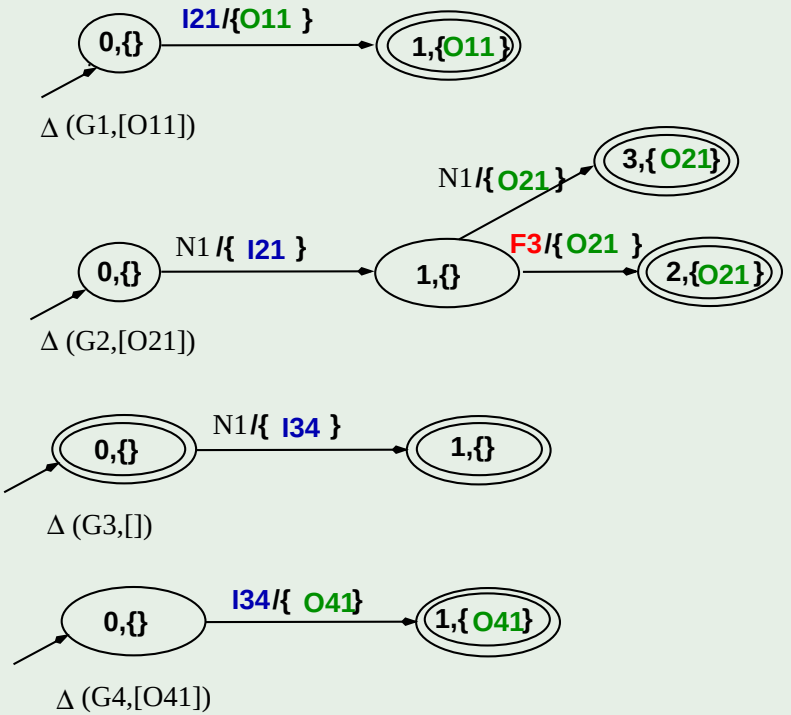
Donc $I21$ n'est pas possible, le diagnostic local de γ_2 ne le permet pas. Tout chemin contenant $I21$ dans Δ_{γ_1} est incompatible. On élimine avant fusion. Δ_{γ_1} est réduit à deux transitions.

Règle 2 : Sélection des diagnostics locaux à fusionner

- Constat : la synchronisation de diagnostics locaux qui n'interagissent pas directement est :
  1. équivalent à faire un produit libre (produit Cartesien)
  2. inutile, car aucune interaction proposée par les deux diagnostics n'est vérifiée
- **Règle 2 :** ne fusionner que les diagnostics qui interagissent !
- Fusion des diagnostics locaux en parallèle (sur plusieurs machines)
- Le résultat est un ensemble de diagnostics indépendants sur des sous-systèmes
  - Chaque diagnostic donne les explications des observations d'une sous parties du système
  - Pas d'intéractions entre les diagnostics résultats

Règle 2 : Sélection des diagnostics locaux à fusionner (2)

Exemple



$\Delta(G1, [O12])$ et $\Delta(G2, [O21])$ intéragissent avec $I21$.

$\Delta(G3, [])$ et $\Delta(G4, [O41])$ intéragissent avec $I34$.

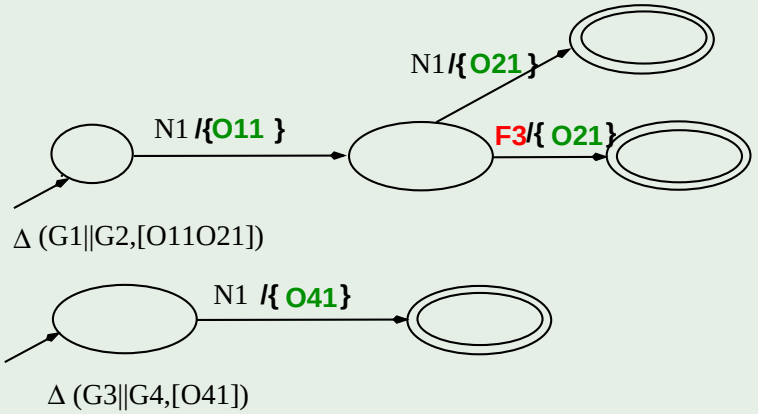
Application de la règle 2 :

Fusion de $\Delta(G1, [O12])$ et $\Delta(G2, [O21])$

Fusion de $\Delta(G3, [])$ et $\Delta(G4, [O41])$

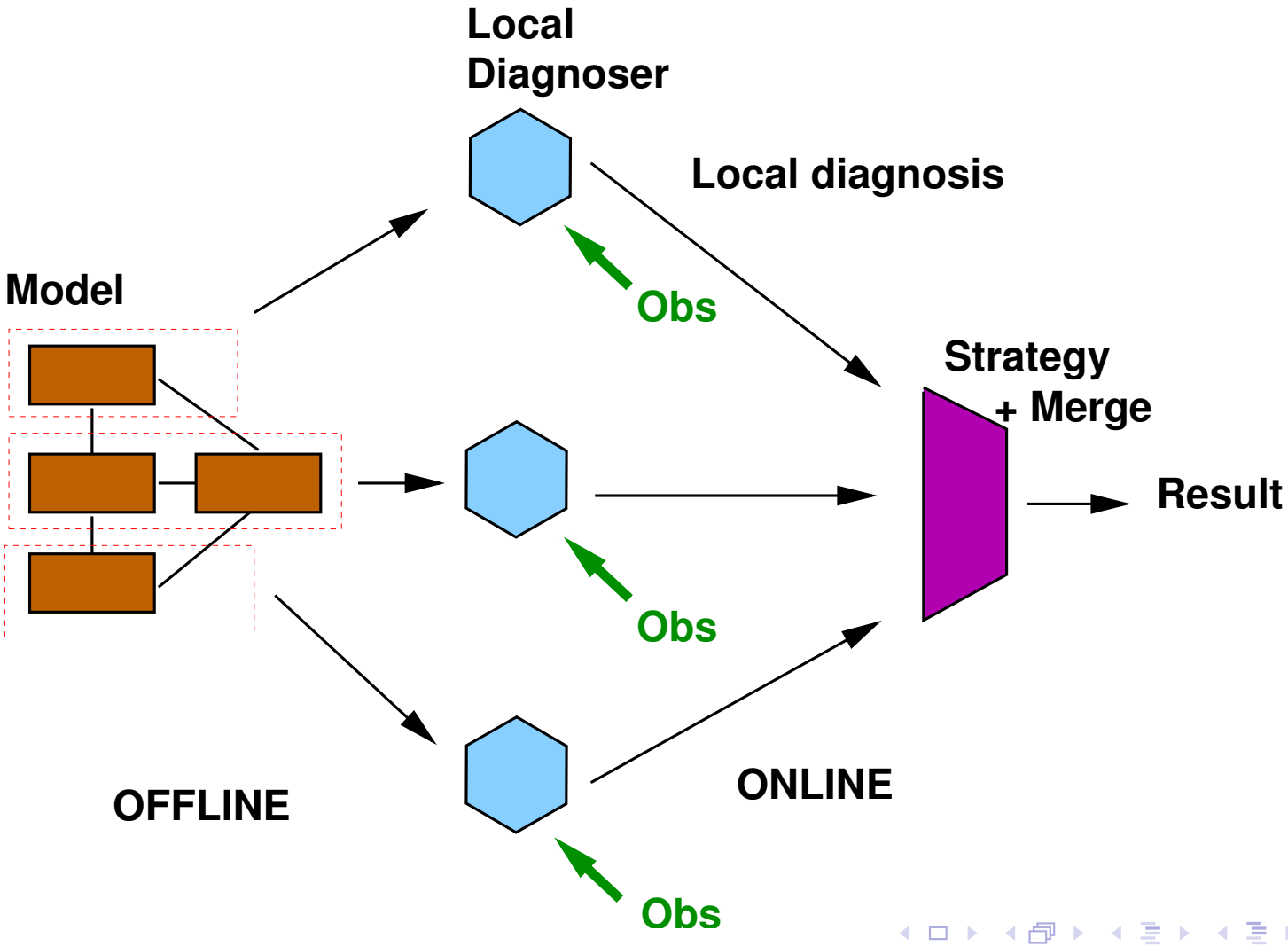
Règle 2 : Sélection des diagnostics locaux à fusionner (2)

Exemple



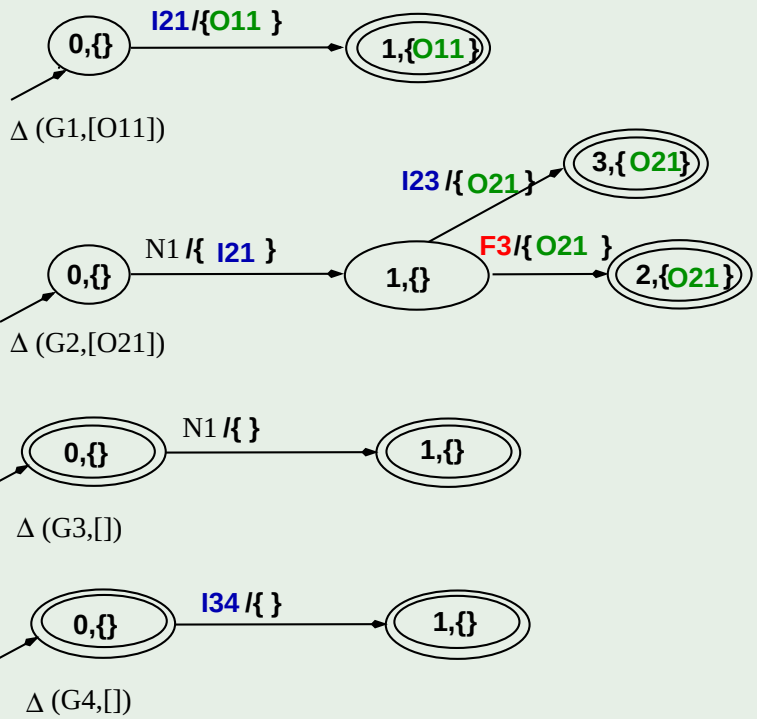
Application de la règle 2 : pas d'interaction entre les deux diagnostics, on arrete les fusions ils sont indépendants. Diagnostic global $= F3, N$

Résumé de ce travail



A vous de jouer !

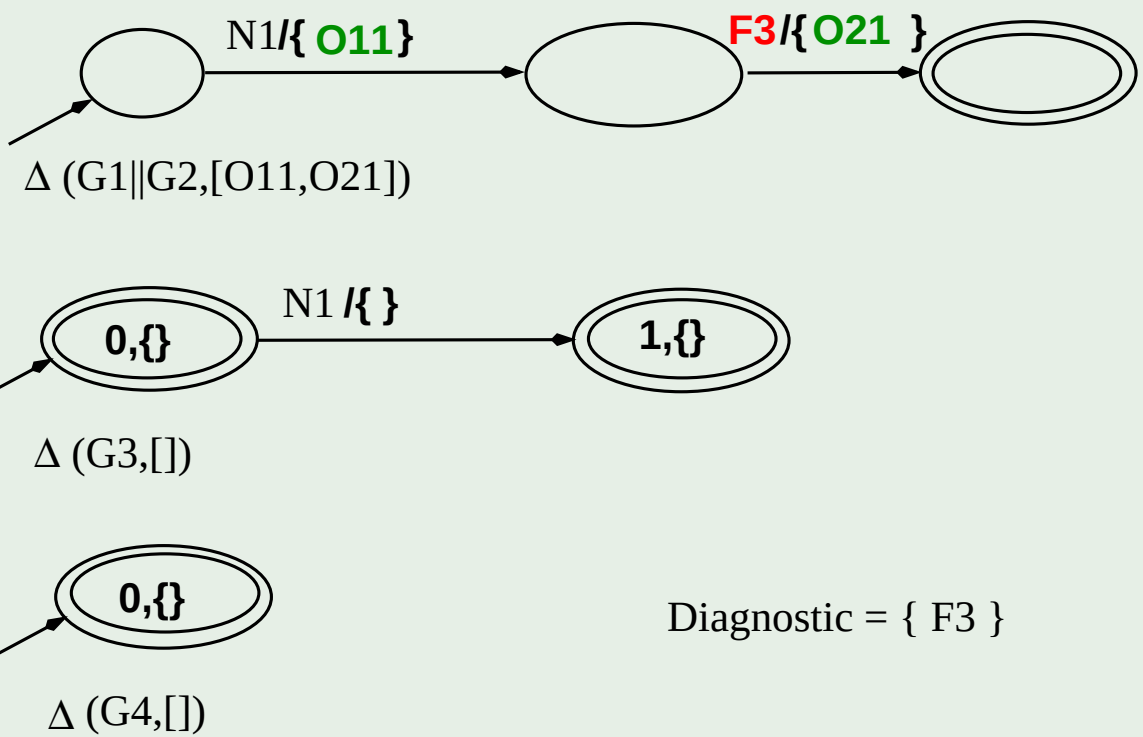
Exemple



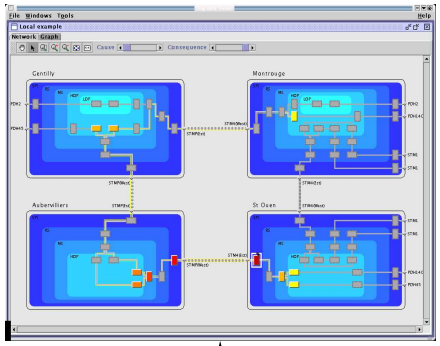
Appliquez les règles 1 et 2 pour trouver le diagnostic global

Resultat du jeu

Exemple



Quelques résultats sur le projet MAGDA



ILOG interface

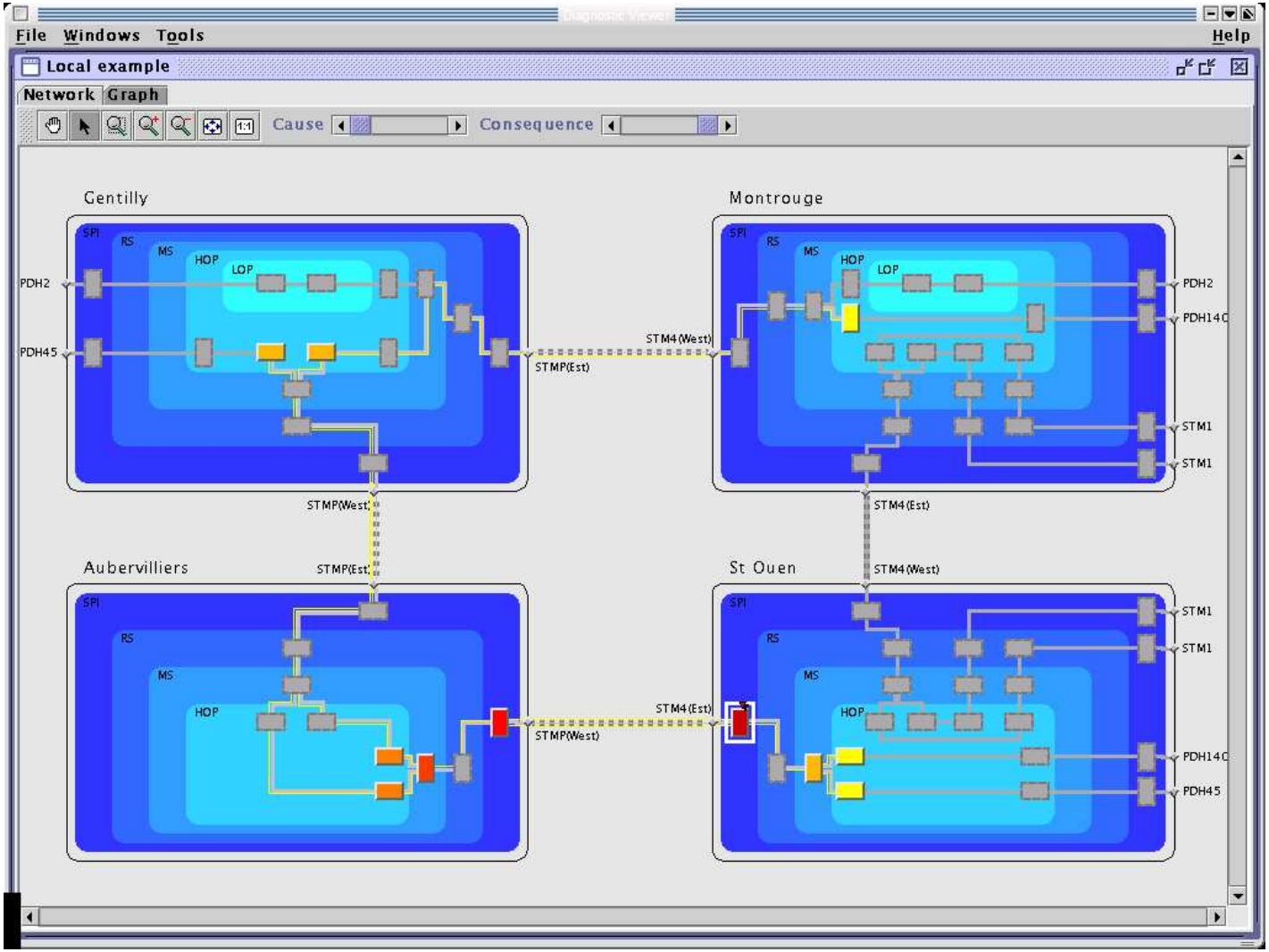
Diagnosis Software — Irisa

Communication via Corba

Alarm Sensors — Alcatel

Network Simulation — Alcatel

Magda : Interface



Quelques résultats sur le projet MAGDA

Scénarios	Stratégie 1	Stratégie 2	Stratégie 3	Stratégie 4
1	3s 590ms	4s 200ms	16s 540ms	>5mn
2	1s 300ms	1s 300ms	1mn 52s 770ms	>5mn
3	1s 780ms	1s 910ms	>5mn	>5mn
4	1s 600ms	2s 30ms	49s 120ms	>5mn
5	2s 620ms	5s 500ms	5s 430ms	3mn 45s 600m
6	1s 780ms	2s 320ms	24s 240ms	57s 440ms
7	1s 480ms	1s 700ms	2mn 54s 920ms	>5mn
8	1s 830ms	3s 90ms	3s 30ms	>5mn

- 8 scénarios
  - Stratégie 1 : Règle 1 + Règle 2
  - Strategie 2 : Règle 1
  - Strategy 3 : Règle 2
  - Strategy 4 : Sans règle

Plan du cours

Approches décentralisées/distribuées

- Points communs avec les approches décentralisées
  - Notion de sites d'observations
  - Connaissance locale à un sous-système (pas de modèle global)
- Différences
  - Pas de coordinateur
  - Pas de diagnostic global mais un ensemble de diagnostics locaux
- En distribué, la décision de réparation est distribuée.
- L'agent en charge de diagnostiquer un sous-système est en charge de le réparer.

Résumé : centralisé-décentralisé-distribué

- Centralisé :

$$\Delta(G_1, \sigma_1, \ldots, G_n, \sigma_n) \triangleq \Delta(G_1 || \ldots || G_n, \sigma_1 \oplus \cdots \oplus \sigma_n)$$

- Décentralisé :

$$\Delta(G_1, \sigma_1, \ldots, G_n, \sigma_n) \triangleq \Delta(G_1, \sigma_1) || \ldots || \Delta(G_n, \sigma_n)$$

- Distribué :

$$\Delta(G_i, \sigma_1, \ldots, \sigma_n) \triangleq P_{G_i}(\Delta(G_1, \sigma_1, \ldots, G_n, \sigma_n))$$

- Liens :

$$\Delta(G_1, \sigma_1, \ldots, G_n, \sigma_n) = \Delta(G_1, \sigma_1, \ldots, \sigma_n) || \ldots || \Delta(G_n, \sigma_1, \ldots, \sigma_n)$$