Évaluation de Performances de Systèmes à Événements Discrets

De la théorie à la pratique

Alain Jean-Marie

ajm@lirmm.fr

INRIA/LIRMM CNRS Université de Montpellier 2

Contexte

Présupposé : l'étude mathématique de systèmes dynamiques est susceptible de servir à la prédiction quantitative du comportement de systèmes réels.

Corollaire : les résultats des recherches peuvent servir aux Praticiens des systèmes en question

- non seulement à comprendre
- mais aussi à prédire, dimensionner, choisir, contrôler.

Systèmes dynamiques réels

Domaines d'application des systèmes à événements discrets stochastiques (s)DEDS :

- réseaux de communication (téléphone, Internet, ...)
- réseaux de calculs, réseaux de tâches
  - planification de projets (RO, PERT)
  - flux de documents
  - calcul parallèle
  - productique
- réseaux de transport (trains, avions, trafic urbain, piétons/évacuations, ...)

Évaluation de Performances

Un système :

- comporte un état évoluant au cours du temps $X(t)$
- est soumis à des sollicitations externes $A(t)$

Les règles de fonctionnement du système permettent de déduire une dynamique

$$X(t+1) = F(X(t), A(t)).$$

Questions :

- Valeur, distribution de $X(t)$, d'une fonctionnelle de X ?
- Comportement asymptotique de $X(t)$? Stabilité, vivacité ?

Systèmes dynamiques du théoricien

Quand nous parlons «systèmes dynamiques», nous voyons :

$$\begin{aligned} \dot{x}(t) &= A\,x(t) + B\,u(t) \\ y(t) &= C\,x(t) \end{aligned}$$

ou en temps discret :

$$\begin{aligned} x(t+1) &= A\,x(t) + B\,u(t) \\ y(t) &= C\,x(t) \end{aligned}$$

Systèmes dynamiques du théoricien

Par exemple :

$$\begin{aligned} x(t) &= A_0 \otimes x(t) \oplus A_1 \otimes x(t-1) \oplus \ldots \\ &\quad \oplus A_p \otimes x(t-p) \oplus B \otimes u(t) \\ y(t) &= C \otimes x(t) \end{aligned}$$

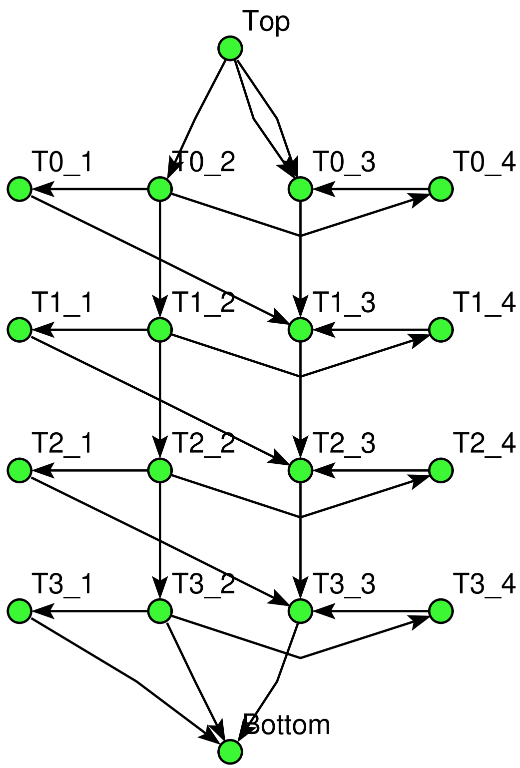
Systèmes dynamiques du théoricien

Ou bien :

$$\begin{array}{rcl} a & \longrightarrow & ab \\ b & \longrightarrow & ac \\ c & \longrightarrow & a \end{array}$$

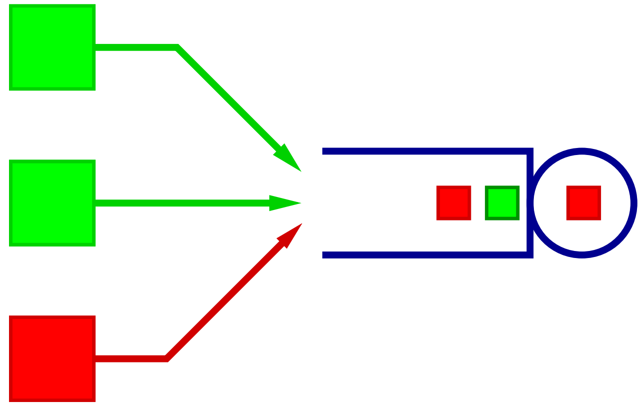
Systèmes dynamiques du théoricien

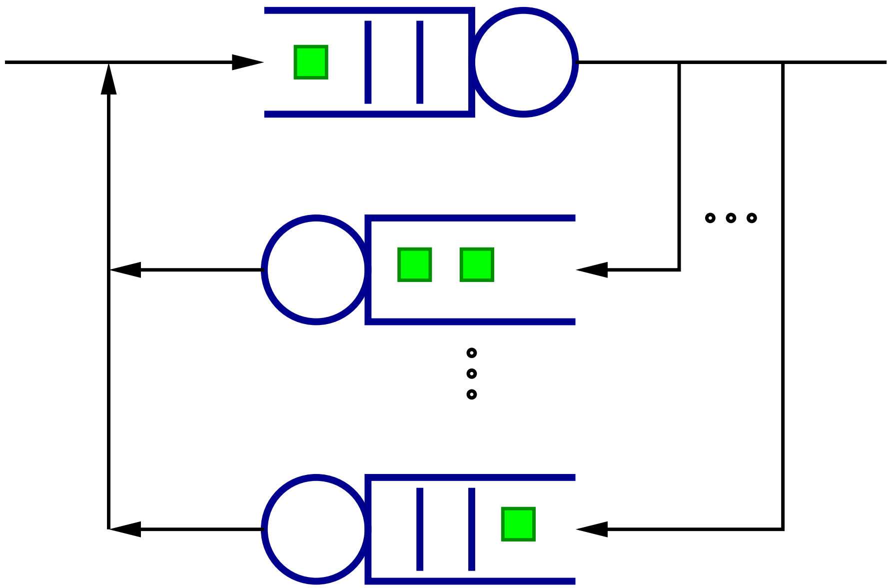
Ou encore :



Systèmes dynamiques du théoricien

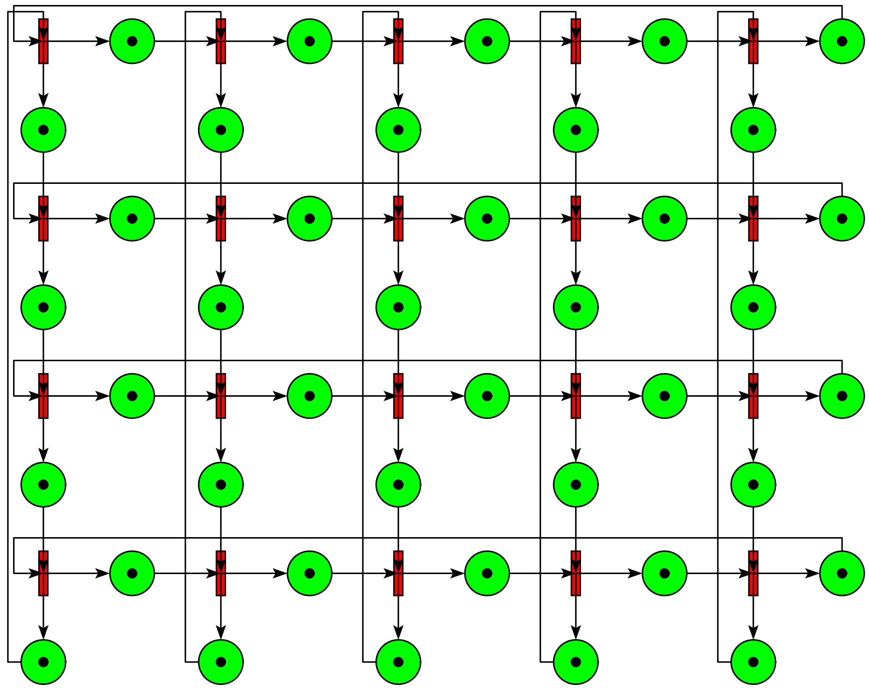
Ou alors :





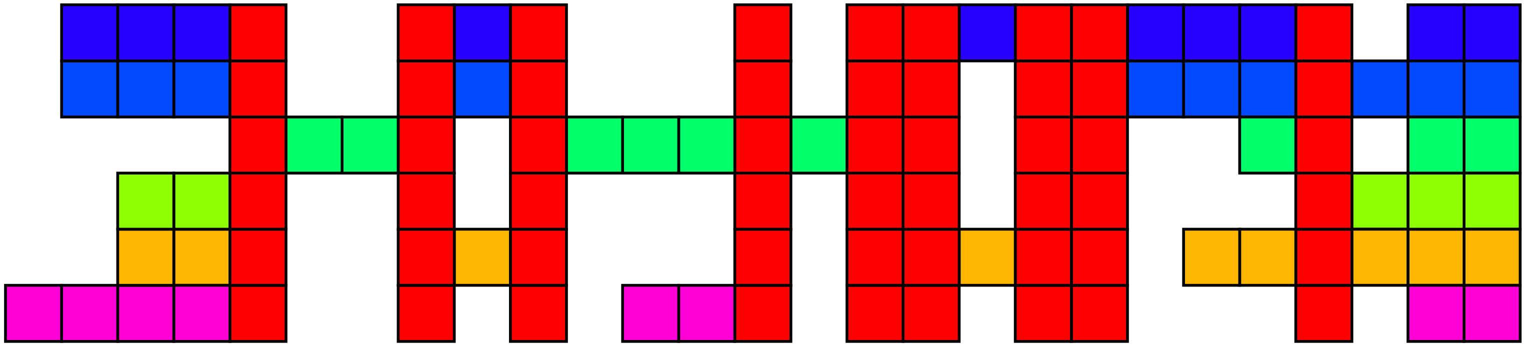
Systèmes dynamiques du théoricien

Ou même :



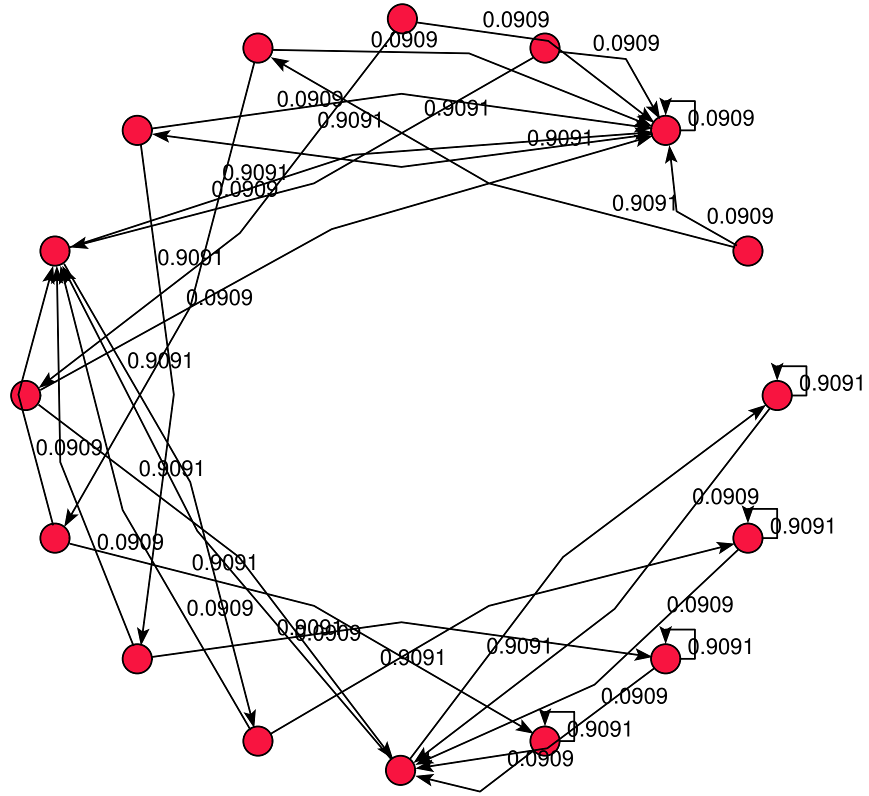
Systèmes dynamiques du théoricien

Ou peut-être :



Systèmes dynamiques du théoricien

À moins que ce ne soit :



Systèmes dynamiques du praticien

Quand un praticien étudie un système dynamique, nous voyons :

```
& QNAP2 code generated by pn2qnap version v6.1
& Origin: ips.pnl
& Date:  05 Nov 97 at 15:41:17
/CONTROL/
    OPTION = NSOURCE;
    OPTION = NRESULT;

/DECLARE/
    QUEUE OBJECT OnOffSrc;
      REAL NextArr;
      REAL NextChg;
    END;
&   transitions
    QUEUE Server;
    OnOffSrc TR_0;
&   other variables
    REAL Tps_Cpu;
    INTEGER Nb_Evts;

&   classes
    CLASS CL_0, CL_1, No_Class;

/CONTROL/
    CLASS = ALL QUEUE;
    TMAX = 1000;

&
& transition
/STATION/
    NAME = TR_0;
    TYPE = SOURCE;
    SERVICE = BEGIN
        TR_0.NextArr := EXP(  10.0000 );
        WHILE ( TR_0.NextArr > TR_0.NextChg )
        DO BEGIN
            TR_0.NextArr := TR_0.NextArr - TR_0.Next
            CST( TR_0.NextChg );
            EXP(   2.0000 );
            TR_0.NextChg := EXP(   3.0000 );
        END;
        CST( TR_0.NextArr );
        TR_0.NextChg := TR_0.NextChg - TR_0.NextArr;
    END;
    TRANSIT =
```

```
        Server, No_Class;

&
& transition
/STATION/
    NAME = Server;
    TYPE = SERVER, SINGLE;
    SERVICE = EXP(   1.0000 );
    TRANSIT = OUT;

/EXEC/
    BEGIN
        TR_0.NextChg := EXP(   2.0000 );
;
        Tps_Cpu := GETCPUT;
        Nb_Evts := 0;
        SIMUL;
        WRITE("Cycle time of transition 0: " )
;        IF( SERVNB(TR_0) > 0 ) THEN WRITE( TIME/SER
          WRITELN;
        Nb_Evts := Nb_Evts + SERVNB(TR_0);
        WRITE("Cycle time of transition 1: " )
;        IF( SERVNB(Server) > 0 ) THEN WRITE( TIME/S
          WRITELN;
        Nb_Evts := Nb_Evts + SERVNB(Server);
        Tps_Cpu := GETCPUT;
        WRITE("Total Execution Time: ", Tps_Cpu );
        WRITELN("      (", Nb_Evts / Tps_Cpu, "evt/s
    END;
/END/
```

Systèmes dynamiques du praticien

Ou bien :

```
# Script M/D/1 selon Chadi Barakat
# Solution du TP de NS pour le DESS TNI

# Initialisation du simulateur
set ns [new Simulator]

# Initialisation des traces (!! doit etre ici sinon
set f [open "md1.trace" w]
set nf [open "md1.nam" w]
set qf [open "md1.qldat" w]
$ns trace-all $f
$ns namtrace-all $nf

# Creation des deux noeuds Source/Destination
set S [$ns node]
set D [$ns node]

# Lien entre les deux
$ns duplex-link $S $D 1Mb 20ms DropTail
$ns queue-limit $S $D 100

# Agents de generation et absobtion de trafic
set udp [new Agent/UDP]
$ns attach-agent $S $udp
set null [new Agent/Null]
$ns attach-agent $D $null
$ns connect $udp $null

set tcp [new Agent/TCP]
$ns attach-agent $S $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $D $sink
$ns connect $tcp $sink

# Loi de trafic attachee a la source
set poisson [new Application/Traffic/Exponential]
$poisson set rate_ 2Mb
$poisson attach-agent $udp
set const [new Application/Traffic/CBR]
$const attach-agent $tcp

# Prise de stats/impressions
set q [[$ns link $S $D] queue]
```

```
# $ns trace-queue $S $D $f
set spy [$ns monitor-queue $S $D $f 0.05]

# Organisation de l'execution
$ns at 0.0 "queueLength 0 0"
$ns at 0.0 "windowTCP $tcp"
$ns at 1.0 "$poisson start"
$ns at 5.0 "puts \"GOO\"; $const start"
$ns at 10 "finish"

proc windowTCP { tcp } {
    global ns
    set time 0.01
    set now [$ns now]
    set wnd [$tcp set cwnd_]
    set rtt [$tcp set rtt_]
    puts "$now $wnd $rtt"
    $ns at [expr $now+$time] "windowTCP $tcp"
}

proc queueLength { sum number } {
    global ns spy qf q
    set time 0.01
    set now [$ns now]
    # set len [$qu length]
    # set sum [expr $sum+$len]
    # incr number
    puts -nonewline $qf "$now [$spy set pkts_] "
    puts $qf "[$spy set parrivals_] [expr [$spy set parrivals_]-[$spy s
    $ns at [expr $time+$now] "queueLength $sum $number"
}

proc finish {} {
    global ns f spy const

    puts "Ayeeee!"
    puts "Avg thput [expr [$spy set bdepartures_]/[$ns now]] oct/s"
    # puts "Queue [$q length]"
    $ns flush-trace
    close $f

    exit 0
}
```

Systèmes à événements discrets académiques

Systémique

Analyse des systèmes complexes par réduction.

- décomposition $\longrightarrow$ structure (blocs + relation)
- analyse des blocs (méthodes opérationnelles)
- agrégation/composition
  - exacte (formes produits, «algèbres», «calculus»)
  - approchée/heuristique
  - avec bornes

Décomposition

- identifier les éléments (sous-systèmes)
- identifier les relations

Comment faire automatiquement ?

- expertise du concepteur
- contraindre par le langage de description ?

Langages de blocs

Langages de description de systèmes faisant appel à des «objets» à mettre en relation.
Exemples :

Matlab/Simulink équations différentielles/récurrentes comme langage de programmation → «simulation» (intégration numérique), et éventuellement des solutions analytiques.

QNAP/RESQ/PAW Langages de files d'attente → simulation («Monte Carlo»), et éventuellement des solutions analytiques.

Réseaux de Petri Langages de réseaux de Petri à objets → exécution, et éventuellement analyse qualitative.

Méthodes opérationnelles

Formalismes se prêtant à une (dé)composition par blocs et à une analyse poussée.

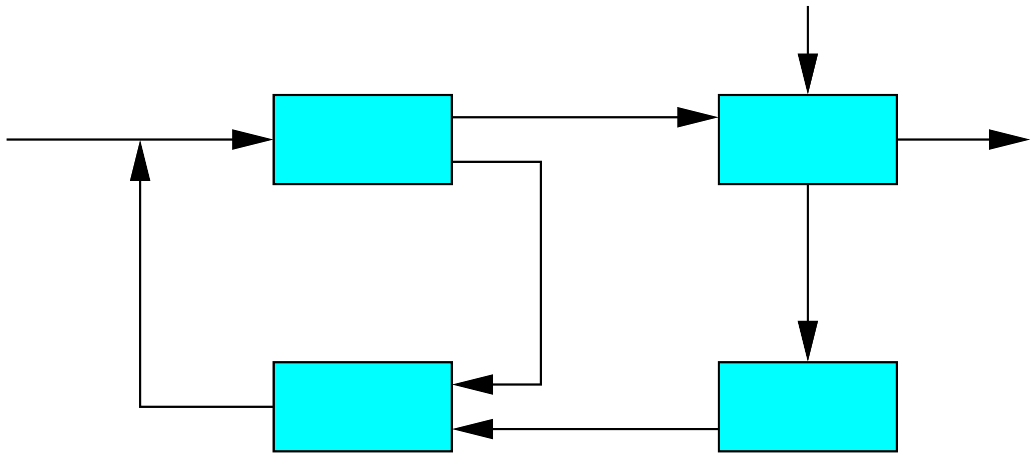
- Chaînes de Markov
- Automates temporisés/stochastiques
- Réseaux de files d'attente
- Réseaux de Petri
- Algèbres de processus
- Algèbres «exotiques»
- ...

Recomposition

Principe du point fixe : coïncidence des «signaux» de sortie et d'entrée.

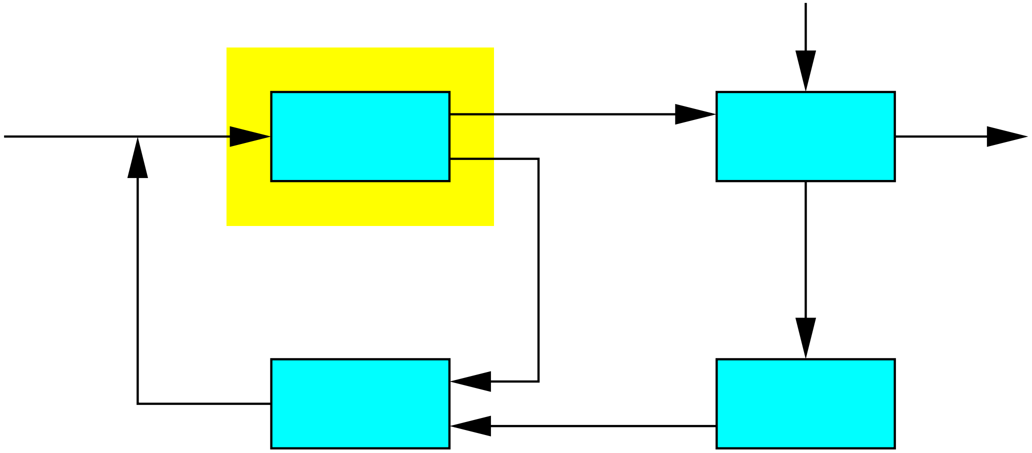
Recomposition

Principe du point fixe : coïncidence des «signaux» de sortie et d'entrée.



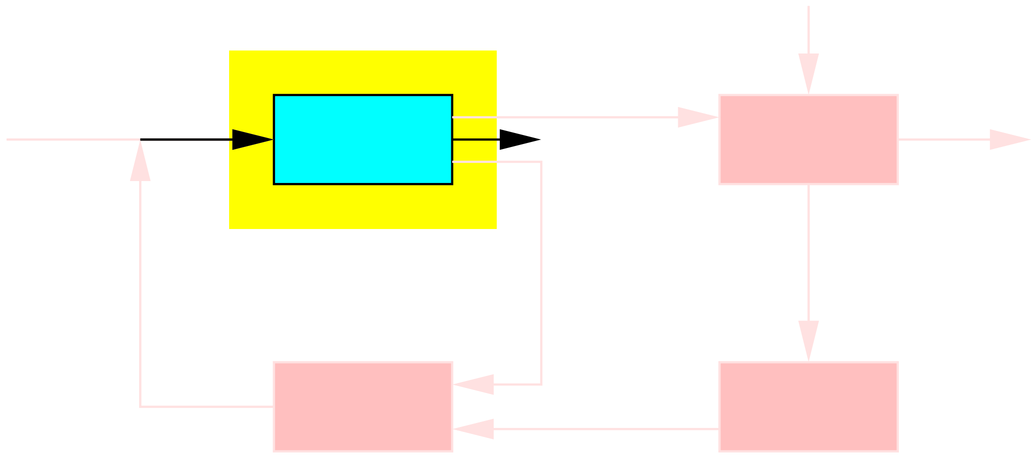
Recomposition

Principe du point fixe : coïncidence des «signaux» de sortie et d'entrée.



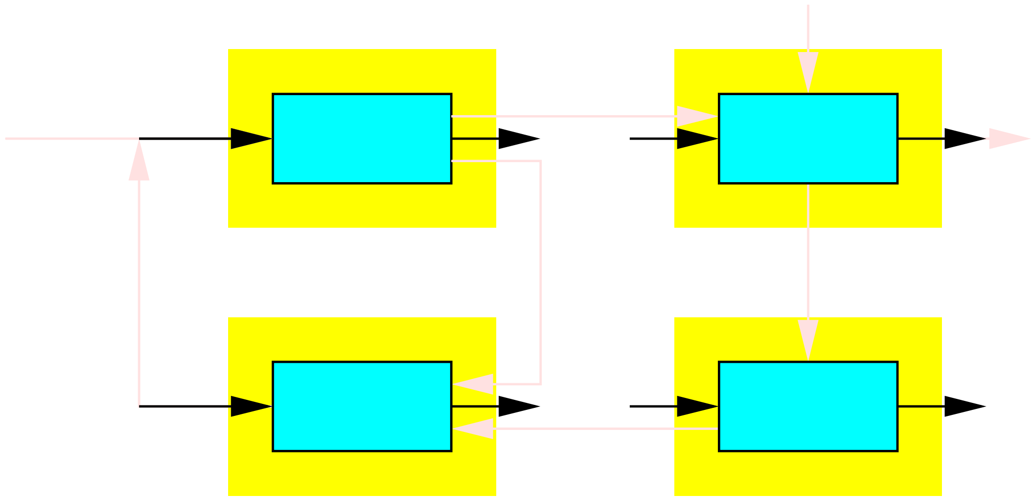
Recomposition

Principe du point fixe : coïncidence des «signaux» de sortie et d'entrée.



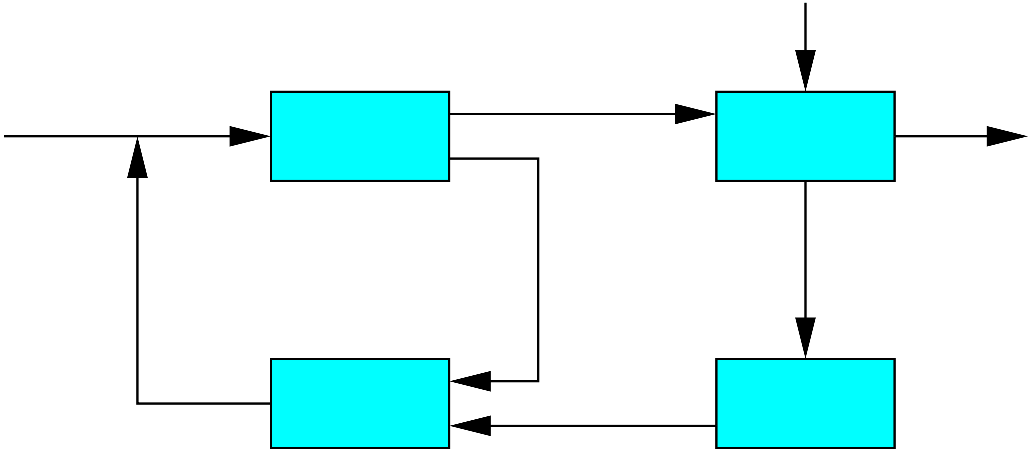
Recomposition

Principe du point fixe : coïncidence des «signaux» de sortie et d'entrée.



Recomposition

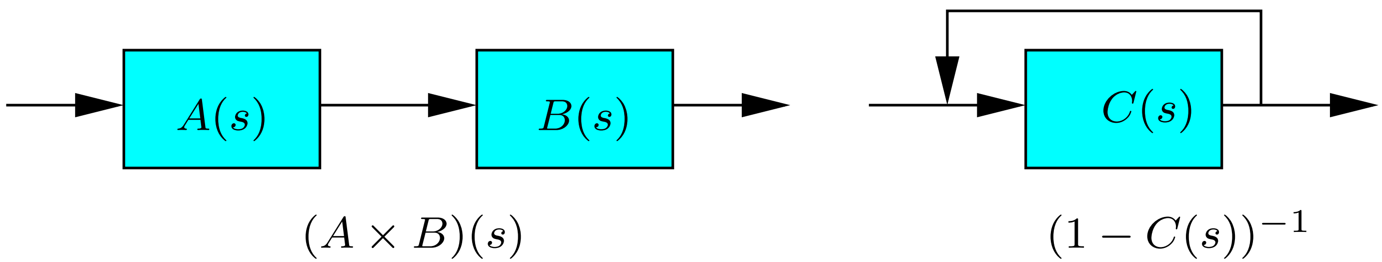
Principe du point fixe : coïncidence des «signaux» de sortie et d'entrée.



Recomposition (suite)

Composition exacte :

- existence d'une fonction de transfert
- composition («algèbre») des fonctions de transfert



- compatibilité des blocs (ex : formes produits pour les files d'attente)

$$P(N_1 = n_1, \ldots, N_p = n_p) \;=\; \prod_{i=1}^{p} P(N_i = n_i)\ .$$

Recomposition (suite)

Composition approchée :

- on ne sait pas calculer la correspondance entrée/sortie pour toute entrée...
- approximation du processus de sortie

  ex. si stochastique : Poisson, i.i.d avec deux moments, Markov-Modulé

  ex. si déterministe : périodique
- itération jusqu'à convergence

Recomposition (fin)

Composition avec bornes ($\longrightarrow$ construction de «pire cas») :

- Comparaisons trajectorielles/stochastiques

$$X \ \geq_{\mathcal{C}} \ Y \qquad \Longrightarrow \qquad Ef(X) \ \geq \ Ef(Y) \ , \quad \forall f \in \mathcal{C} \ .$$

- Association

$$Ef(X)g(Y) \ \geq \ Ef(X) \ Eg(Y) \ .$$

- Majorisation

$$x \prec y \qquad \Longleftrightarrow \qquad \sum_{i=1}^{k} x_i \ \leq \ \sum_{i=1}^{k} y_i, \ \forall k \ .$$

Chaînes de Markov

Méthode universelle. Ingrédients :

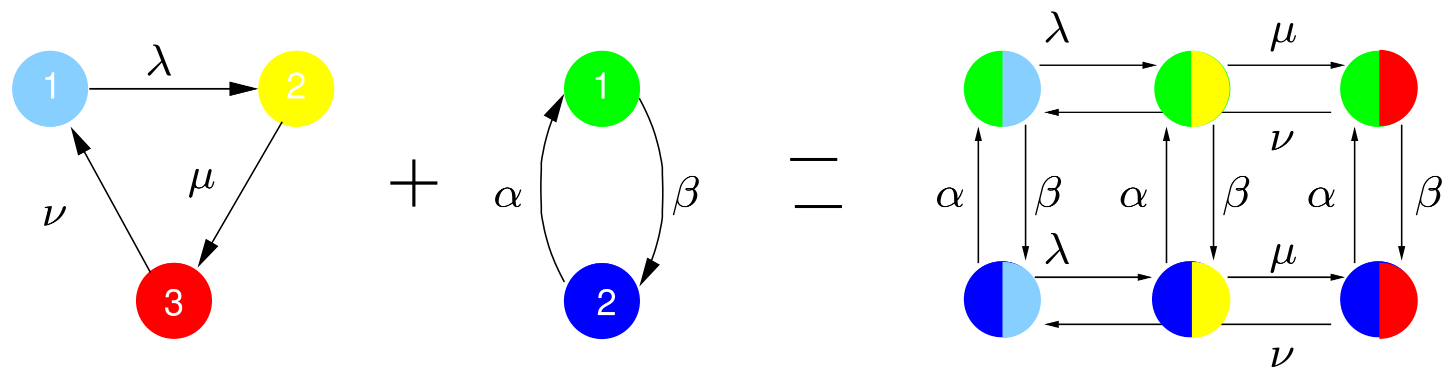
- un espace d'états $\mathcal{E}$
- des transitions probabilisées, dans une matrice P.

Composition de blocs :

$$(\mathcal{E}_1, P_1) \;+\; (\mathcal{E}_2, P_2) \;\longrightarrow\; (\mathcal{E}_1 \times \mathcal{E}_2, P_1 \otimes_f P_2)\;,$$

f fonction décrivant le couplage entre les transitions.

Chaînes de Markov (suite)



$$\begin{pmatrix} -\lambda & \lambda & 0 \\ 0 & -\mu & \mu \\ \nu & 0 & -\nu \end{pmatrix} \oplus \begin{pmatrix} -\alpha & \alpha \\ \beta & -\beta \end{pmatrix} = \left(\begin{array}{ccc|ccc} - & \lambda & 0 & \alpha & 0 & 0 \\ 0 & - & \mu & 0 & \alpha & 0 \\ \nu & 0 & - & 0 & 0 & \alpha \\ \hline \beta & 0 & 0 & - & \lambda & 0 \\ 0 & \beta & 0 & 0 & - & \mu \\ 0 & 0 & \beta & \nu & 0 & - \end{array}\right)$$

Modélisation markovienne

Prise en compte de durées non constantes/non exponentielles

- Modèles semi-markoviens
- Décomposition/approximation de lois par des lois «à phases»

Prise en compte de l'histoire

$$X_{n+1} = f(X_n, \ldots, X_{n-p+1}; \zeta_n)$$

Augmentation de l'espace d'états :

$$Z_n = (X_n, \ldots, X_{n-p+1}) \qquad Z_{n+1} = F(Z_n; \zeta_n) .$$

Solution de modèles Markoviens

- Méthodes de résolution exacte ($\longrightarrow$ numérique)
- Simulation «Monte Carlo» (approchée, exacte)

Solution exacte :

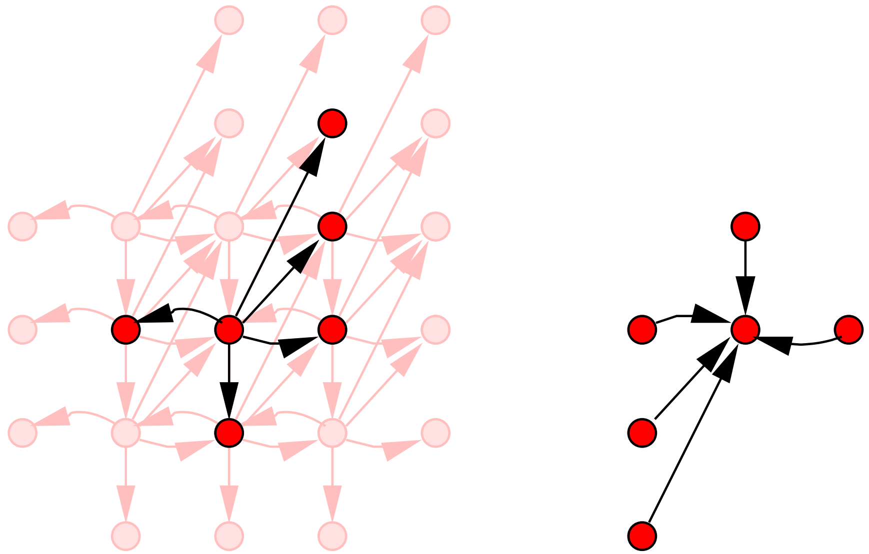
- Espace d'état fini : algébre linéaire

$$\begin{array}{rrcl} \text{probabilités stationnaires} & \pi & = & \pi \times P \\ \text{temps moyen d'atteinte} & T & = & (I - P)^{-1} \times \mathbf{1} \end{array}$$

- Espace d'état infini : séries génératrices

Utilisation des séries génératrices

Exploitation d'une régularité spatiale dans les transitions :



Probabilités stationnaires :

$$\pi(\vec{n}) = \sum_m P_m \, \pi(\vec{n} - \vec{m})$$

Utilisation des séries génératrices (suite)

En passant aux séries génératrices :

$$\Pi(\vec{z}\,) \;=\; \underbrace{P(\vec{z}\,) \times \Pi(\vec{z}\,)}_{\text{intérieur du domaine}} \;+\; \underbrace{\Pi_0(\vec{z}\,)}_{\text{conditions au bord}}$$

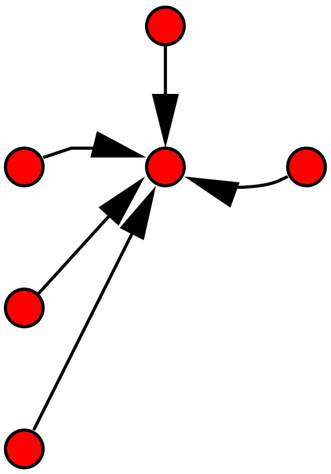
$$\Pi(\vec{z}\,) \;=\; E(\vec{z}^{\;\vec{n}}) \qquad P(\vec{z}\,) \;=\; \sum_m \vec{z}^{\;\vec{m}} P_m \,.$$

$\longrightarrow$ possibilité de traitement automatique, en utilisant des outils formels et numériques.

Utilisation des séries génératrices (suite)

En passant aux séries génératrices :

$$\Pi(\vec{z}\,) = \underbrace{P(\vec{z}\,) \times \Pi(\vec{z}\,)}_{\text{intérieur du domaine}} + \underbrace{\Pi_0(\vec{z}\,)}_{\text{conditions au bord}}$$



$$P(x, y) = x + xy + xy^2 + x^{-1} + y^{-1}$$

Synthèse

La panoplie de l'évaluateur de performances comporte à présent de nombreux outils :

- stochastiques
- analytiques, numériques
- algébriques

+ formalismes et techniques opérationnelles permettant :

- une description modulaire des systèmes
- l'analyse complète des performances chaque bloc séparément
- la reconstitution des performances du système global, exatement ou sous la forme de bornes.

Donc tout va bien...

Opérationnel ?

De nombreuses limitations...

- puissance de modélisation
- taille de l'espace d'états (matrices...)
- dimensions de l'espace d'états (séries génératrices...)
- un bon langage de description ?

Systèmes à événements discrets industriels

Systèmes dynamiques dans l'Industrie

Praticien : de l'Industrie ou recherche finalisée dans un des domaines d'application des (s)DEDS cités plus haut.
Constat : les Praticiens semblent ne pas utiliser de méthodes analytiques et recourrent massivement à la simulation.
Pourquoi ?

Deux groupes d'explications :

- inadéquation des outils
- inadéquation des ouvriers

Limitations techniques

Les outils analytiques ne sont pas utilisables tels quels :

- champ d'application : hypothèses «trop fortes» inadéquates avec la réalité.
- niveau de description inadéquat (files d'attente : où sont-elles ? jetons de réseaux de Petri, matrices (max,+)...)
- mise en œuvre : lecture d'articles, implémentation de formules, algorithmes...

Limitations culturelles

L'environnement du Praticien se caractérise par une forte contrainte temporelle. Les résultats doivent être obtenus vite et sûrement.

Aspect paradoxal :

- les méthodes analytiques permettent souvent d'accélérer le calcul de résulats
- les résultats obtenus peuvent être plus fiables que ceux de simulations.

Mais :

- obtenir un modèle traitable analytiquement et validé par rapport au réel coûte du temps.

Limitations culturelles (suite)

Alors que :

- développer une simulation, conduire les expériences, etc. est un processus considéré comme maîtrisé.

Du point de vue des praticiens :

- ils ne savent pas que des outils analytiques existent
- ils ne savent pas se servir de ces outils
  $\rightarrow$ plus de temps pour l'étude, moins de confiance
- ils ne croient pas aux résultats
- leurs clients ne croient pas aux résultats

Analogies pour l'analyse d'algorithmes ?

Praticiens : les programmeurs.

- aucun de fait d'analyse d'algorithmes, donc
  - ils ne savent pas la complexité spatiale et temporelle de leur programmes
  - ils ne peuvent pas choisir la meilleure méthode
- comment le faire à leur place ?
- complexité algorithmique $\leftrightarrow$ modèle de données ?

Quelles solutions ?

- mettre des outils logiciels à disposition
- les cacher à l'utilisateur

Structure du système ?

- analyse automatique très difficile
- $\longrightarrow$ exploitation de l'expertise du modélisateur

Possibilité : contraindre la description des modèles (langage, interface utilisateur).
Peu populaire parmi les Praticiens. Acceptable si bon rapport flexibilité/puissance d'analyse...

En conclusion

Finalement, que faut-il ?

- des modèles et des résultats plus puissants ?

En conclusion

Finalement, que faut-il ?

- des modèles et des résultats plus puissants ?
- des logiciels plus évolués ?

En conclusion

Finalement, que faut-il ?

- des modèles et des résultats plus puissants ?
- des logiciels plus évolués ?
- des formations plus complètes aux systèmes dynamiques à événements discrets ?

En conclusion

Finalement, que faut-il ?

- des modèles et des résultats plus puissants ?
- des logiciels plus évolués ?
- des formations plus complètes aux systèmes dynamiques à événements discrets ?

?