Simulation parfaite de réseaux de Jackson mixtes par un processus bornant

Ana Bušić   Bruno Gaujal   Florence Perronnin*

INRIA-ENS Paris   INRIA Rhône-Alpes   Université Joseph Fourier

Réunion plénière ANR Marmote

8 octobre 2013

Plan

Motivation

Problème général

Comment échantillonner l'état stationnaire d'une chaîne de Markov ?

Réponses possibles

1. Résoudre $\pi = \pi P$ puis échantillonner $\pi(.)$.
2. Simulation parfaite de la chaîne

Problème 1 (espace d'états)

La matrice de transition P ne tient pas en mémoire. Ni l'espace d'états.

Problème 2 (espace d'états)

L'espace d'états est infini. Il ne tient pas non plus en mémoire...

Problème 3 (structure)

La chaîne n'est peut-être pas monotone.

Motivation

Problème général

Comment échantillonner l'état stationnaire d'une chaîne de Markov ?

Réponses possibles

1. Résoudre $\pi = \pi P$ puis échantillonner $\pi(.)$.
2. Simulation parfaite de la chaîne

Problème 1 (espace d'états)

La matrice de transition P ne tient pas en mémoire. Ni l'espace d'états.

Problème 2 (espace d'états)

L'espace d'états est infini. Il ne tient pas non plus en mémoire...

Problème 3 (structure)

La chaîne n'est peut-être pas monotone.

Motivation

Problème général

Comment échantillonner l'état stationnaire d'une chaîne de Markov ?

Réponses possibles

1. Résoudre $\pi = \pi P$ puis échantillonner $\pi(.)$.
2. Simulation parfaite de la chaîne

Problème 1 (espace d'états)

La matrice de transition P ne tient pas en mémoire. Ni l'espace d'états.

Problème 2 (espace d'états)

L'espace d'états est infini. Il ne tient pas non plus en mémoire...

Problème 3 (structure)

La chaîne n'est peut-être pas monotone.

Motivation

Problème général

Comment échantillonner l'état stationnaire d'une chaîne de Markov ?

Réponses possibles

1. Résoudre $\pi = \pi P$ puis échantillonner $\pi(.)$.
2. Simulation parfaite de la chaîne

Problème 1 (espace d'états)

La matrice de transition P ne tient pas en mémoire. Ni l'espace d'états.

Problème 2 (espace d'états)

L'espace d'états est infini. Il ne tient pas non plus en mémoire...

Problème 3 (structure)

La chaîne n'est peut-être pas monotone.

Motivation

Problème général

Comment échantillonner l'état stationnaire d'une chaîne de Markov ?

Réponses possibles

1. Résoudre $\pi = \pi P$ puis échantillonner $\pi(.)$.
2. Simulation parfaite de la chaîne

Problème 1 (espace d'états)

La matrice de transition P ne tient pas en mémoire. Ni l'espace d'états.

Problème 2 (espace d'états)

L'espace d'états est infini. Il ne tient pas non plus en mémoire...

Problème 3 (structure)

La chaîne n'est peut-être pas monotone.

Motivation

Problème général

Comment échantillonner l'état stationnaire d'une chaîne de Markov ?

Réponses possibles

1. Résoudre $\pi = \pi P$ puis échantillonner $\pi(.)$.
2. Simulation parfaite de la chaîne

Problème 1 (espace d'états)

La matrice de transition P ne tient pas en mémoire. Ni l'espace d'états.

Problème 2 (espace d'états)

L'espace d'états est infini. Il ne tient pas non plus en mémoire...

Problème 3 (structure)

La chaîne n'est peut-être pas monotone.

Contributions

On s'intéresse (d'abord) à des chaînes de Markov particulières : les réseaux de Jackson.

Nouveauté 1 (problème 1)

Simulation parfaite de chaînes de Markov à très grand espace d'états en un temps raisonnable.

Nouveauté 2 (Problème 2)

Algorithme de simulation parfaite pour des chaînes de Markov à espace d'états infini ($N = \infty$)

Nouveauté 3 (Problème 3)

L'algorithme proposé peut être adapté au cas non monotone.

Contributions

On s'intéresse (d'abord) à des chaînes de Markov particulières : les réseaux de Jackson.

Nouveauté 1 (problème 1)

Simulation parfaite de chaînes de Markov à très grand espace d'états en un temps raisonnable.

Nouveauté 2 (Problème 2)

Algorithme de simulation parfaite pour des chaînes de Markov à espace d'états infini ($N = \infty$)

Nouveauté 3 (Problème 3)

L'algorithme proposé peut être adapté au cas non monotone.

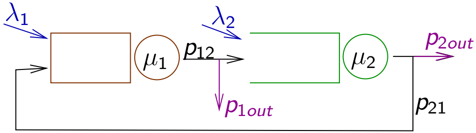
Plan

Modèle

Réseaux de Jackson

- routage probabiliste
- un client arrivant dans une file pleine est perdu.

Exemple démo



- File 1 : capacité finie C
- File 2 : capacité infinie

Plan

Principe du processus bornant

Principe

L'algorithme proposé repose sur la simulation parfaite et la construction d'un processus bornant.

Si une file a une capacité C finie, on la remplace par une file infinie.

- Le réseau de Jackson infini est à forme produit s'il est stable.
- La trajectoire du réseau infini est une borne supérieure de celle du réseau original.
- On sait renverser en temps le processus bornant stationnaire.

Principe du processus bornant

Principe

L'algorithme proposé repose sur la simulation parfaite et la construction d'un processus bornant.

Si une file a une capacité C finie, on la remplace par une file infinie.

- Le réseau de Jackson infini est à forme produit s'il est stable.
- La trajectoire du réseau infini est une borne supérieure de celle du réseau original.
- On sait renverser en temps le processus bornant stationnaire.

Principe du processus bornant

Principe

L'algorithme proposé repose sur la simulation parfaite et la construction d'un processus bornant.

Si une file a une capacité C finie, on la remplace par une file infinie.

- Le réseau de Jackson infini est à forme produit s'il est stable.
- La trajectoire du réseau infini est une borne supérieure de celle du réseau original.
- On sait renverser en temps le processus bornant stationnaire.

Principe du processus bornant

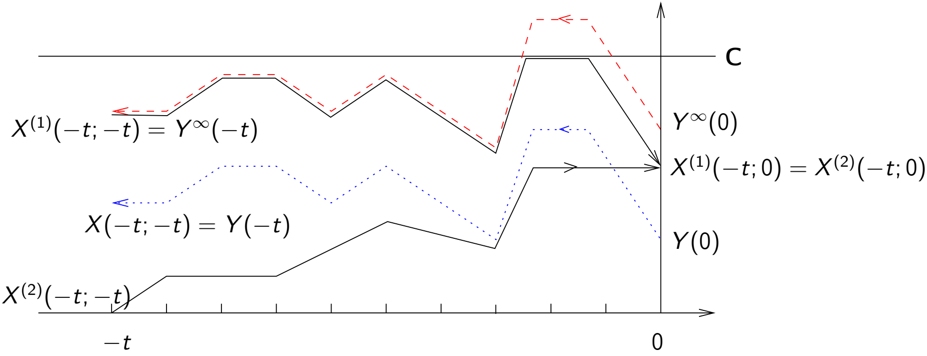
Principe

L'algorithme proposé repose sur la simulation parfaite et la construction d'un processus bornant.

Si une file a une capacité C finie, on la remplace par une file infinie.

- Le réseau de Jackson infini est à forme produit s'il est stable.
- La trajectoire du réseau infini est une borne supérieure de celle du réseau original.
- On sait renverser en temps le processus bornant stationnaire.

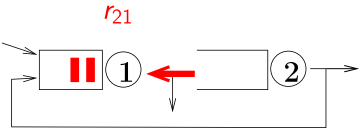
Simulation parfaite avec un processus bornant



Plan

Couplage des événements

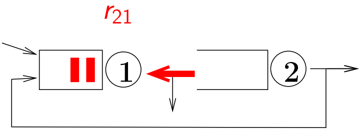
On remonte le temps : $Y^{\infty}(-t)$ trajectoire du réseau infini à l' instant $-t$.

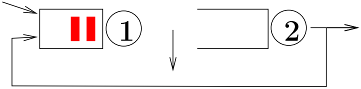


On doit donc choisir les événements pour que la trajectoire soit identique dans les deux sens.

Couplage des événements

On remonte le temps : $Y^{\infty}(-t)$ trajectoire du réseau infini à l' instant $-t$.

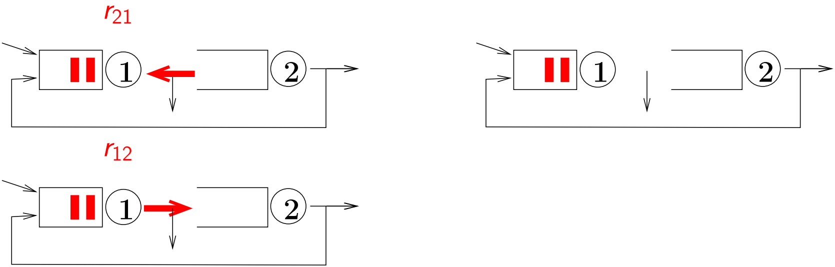




On doit donc choisir les événements pour que la trajectoire soit identique dans les deux sens.

Couplage des événements

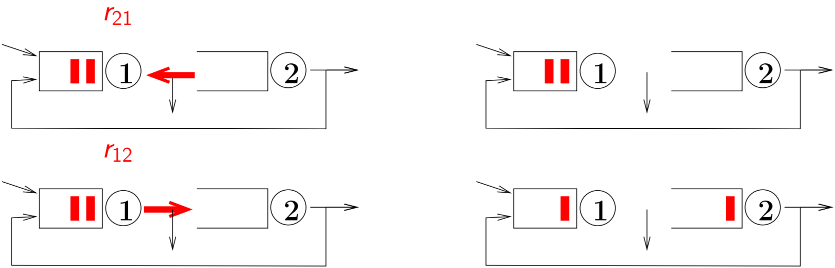
On remonte le temps : $Y^{\infty}(-t)$ trajectoire du réseau infini à l' instant $-t$.



On doit donc choisir les événements pour que la trajectoire soit identique dans les deux sens.

Couplage des événements

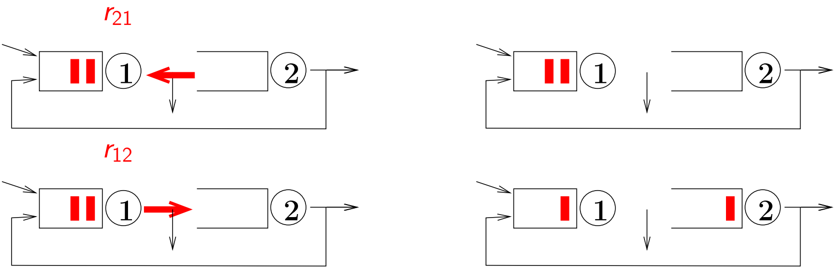
On remonte le temps : $Y^{\infty}(-t)$ trajectoire du réseau infini à l' instant $-t$.



On doit donc choisir les événements pour que la trajectoire soit identique dans les deux sens.

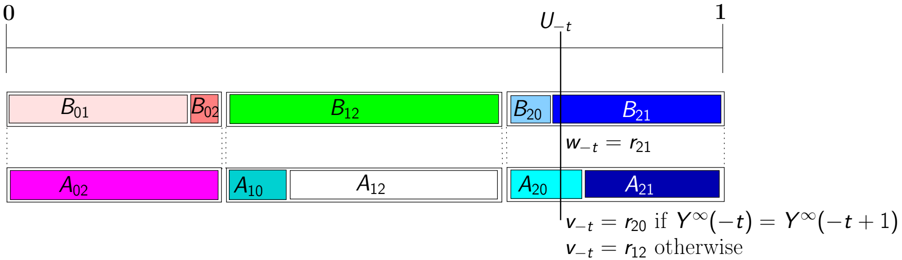
Couplage des événements

On remonte le temps : $Y^\infty(-t)$ trajectoire du réseau infini à l' instant $-t$.



On doit donc choisir les événements pour que la trajectoire soit identique dans les deux sens.

Génération et couplage des événements



Légende :

- B_{ij} (intervalle de) probabilité de l'événement $i \to j$ pour le réseau infini renversé
- A_{ij} (intervalle de) probabilité de l'événement $i \to j$ pour le réseau infini

Plan

Algorithme

Entrées

1. séquence d'innovations
2. un état stationnaire du processus bornant

Sortie

un état distribué selon la loi stationnaire du réseau de Jackson considéré

Algorithme

1. Remonter la trajectoire bornante Y jusqu'à $-2t$
   - et déterminer au passage les événements dans le sens direct.
2. Écrêter l'état obtenu à la capacité réelle du réseau d'origine : $X = Y(-2t) \wedge C$
3. rejouer 2 trajectoires vers l'avant :
   - en partant de X
   - en partant de 0.
4. Doubler t et recommencer depuis l'étape 1 jusqu'à coalescence

Plan

Performances

L'algorithme permet d'échantillonner parfaitement des réseaux de files d'attente avec un espace d'états infini.

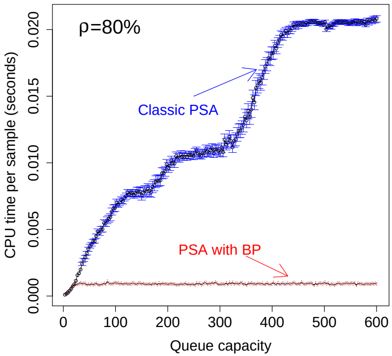
On peut aussi l'utiliser pour simuler des réseaux finis de très grand espace d'état.
Dans ce cas il apporte un gain important sur le temps de simulation, qui n'augmente plus avec la taille de l'espace d'états.

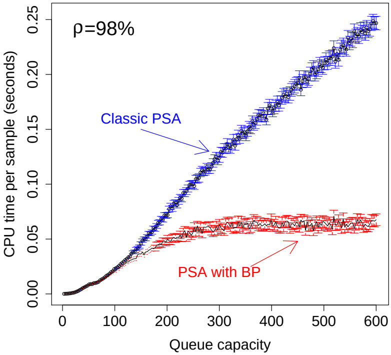
Performances

L'algorithme permet d'échantillonner parfaitement des réseaux de files d'attente avec un espace d'états infini.

On peut aussi l'utiliser pour simuler des réseaux finis de très grand espace d'état.
Dans ce cas il apporte un gain important sur le temps de simulation, qui n'augmente plus avec la taille de l'espace d'états.

Temps d'échantillonnage comparé pour 50 files





Temps de Coalescence

Complexité de l'algorithme proposé :

- Générer un échantillon stationnaire du processus bornant $\mathcal{O}(\text{nombre de files})$
- Générer les événements à partir de la séquence d'innovations (et de la trajectoire) : $\mathcal{O}(1)$
- Construire les trajectoires jusqu'à coalescence

Théorèmes du temps de coalescence

Sous certaines conditions, le temps d'échantillonnage moyen est quadratique en l'occupation moyenne des files :

$$\mathbb{E}\left[\tau_{\text{PSA-BP}}\right] \leqslant \sum_{i=1}^{M} \frac{\Gamma}{\lambda_i} \left(\mathbb{E}\left[Y_i^{\infty}\right]\right)^2$$

Le temps de simulation ne dépend pas de la capacité des files !

Temps de Coalescence

Complexité de l'algorithme proposé :

- Générer un échantillon stationnaire du processus bornant $\mathcal{O}$(nombre de files)
- Générer les événements à partir de la séquence d'innovations (et de la trajectoire) : $\mathcal{O}(1)$
- Construire les trajectoires jusqu'à coalescence

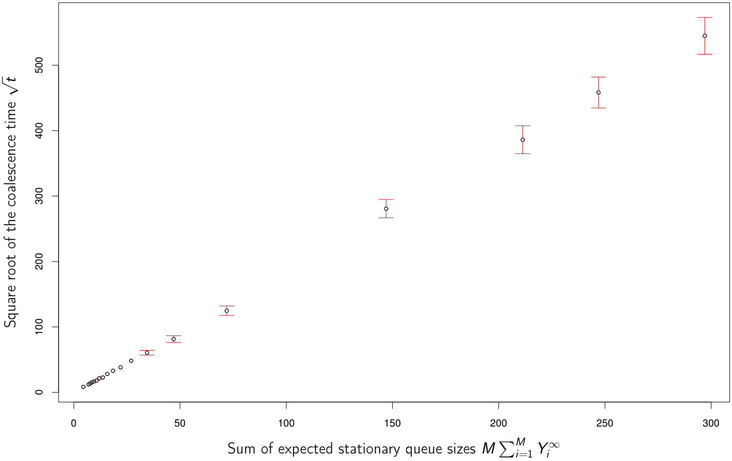
Théorèmes du temps de coalescence

Sous certaines conditions, le temps d'échantillonnage moyen est quadratique en l'occupation moyenne des files :

$$\mathbb{E}\left[\tau_{\text{PSA-BP}}\right] \leqslant \sum_{i=1}^{M} \frac{\Gamma}{\lambda_i} \left(\mathbb{E}\left[Y_i^{\infty}\right]\right)^2$$

Le temps de simulation ne dépend pas de la capacité des files !

Complexité quadratique (occupation moyenne des files)



Plan

Les nouveautés

1. On peut simuler parfaitement certaines CMTC d'espace d'état infini.
2. On peut simuler parfaitement certaines CMTC d'espace d'état très grand en un temps raisonnable.
3. On peut simuler parfaitement certaines CMTC non monotones
4. On a donné l'algorithme directement implémentable et des preuves (partielles) de la complexité.

Sous les hypothèses suivantes :

1. Monotonie du processus bornant
2. on peut échantillonner et renverser le processus bornant
3. la preuve de complexité fait des hypothèses d'hyperstabilité

Les nouveautés

1. On peut simuler parfaitement certaines CMTC d'espace d'état infini.
2. On peut simuler parfaitement certaines CMTC d'espace d'état très grand en un temps raisonnable.
3. On peut simuler parfaitement certaines CMTC non monotones
4. On a donné l'algorithme directement implémentable et des preuves (partielles) de la complexité.

Sous les hypothèses suivantes :

1. Monotonie du processus bornant
2. on peut échantillonner et renverser le processus bornant
3. la preuve de complexité fait des hypothèses d'hyperstabilité

Les nouveautés

1. On peut simuler parfaitement certaines CMTC d'espace d'état infini.
2. On peut simuler parfaitement certaines CMTC d'espace d'état très grand en un temps raisonnable.
3. On peut simuler parfaitement certaines CMTC non monotones
4. On a donné l'algorithme directement implémentable et des preuves (partielles) de la complexité.

Sous les hypothèses suivantes :

1. Monotonie du processus bornant
2. on peut échantillonner et renverser le processus bornant
3. la preuve de complexité fait des hypothèses d'hyperstabilité

Les nouveautés

1. On peut simuler parfaitement certaines CMTC d'espace d'état infini.
2. On peut simuler parfaitement certaines CMTC d'espace d'état très grand en un temps raisonnable.
3. On peut simuler parfaitement certaines CMTC non monotones
4. On a donné l'algorithme directement implémentable et des preuves (partielles) de la complexité.

Sous les hypothèses suivantes :

1. Monotonie du processus bornant
2. on peut échantillonner et renverser le processus bornant
3. la preuve de complexité fait des hypothèses d'hyperstabilité

Les nouveautés

1. On peut simuler parfaitement certaines CMTC d'espace d'état infini.
2. On peut simuler parfaitement certaines CMTC d'espace d'état très grand en un temps raisonnable.
3. On peut simuler parfaitement certaines CMTC non monotones
4. On a donné l'algorithme directement implémentable et des preuves (partielles) de la complexité.

Sous les hypothèses suivantes :

1. Monotonie du processus bornant
2. on peut échantillonner et renverser le processus bornant
3. la preuve de complexité fait des hypothèses d'hyperstabilité

Perspectives

1. Implémenter l'algorithme PSA-BP dans le logiciel PSI
2. Élargir à des chaînes de Markov plus générales
   - Relâcher la condition de stabilité du réseau infini :
     - en remplaçant les files instables par des sources
   - Marche aléatoire sur une grille
   - Autres chaînes de Markov particulières ?

Perspectives

1. Implémenter l'algorithme PSA-BP dans le logiciel PSI
2. Élargir à des chaînes de Markov plus générales
   - Relâcher la condition de stabilité du réseau infini :
     - en remplaçant les files instables par des sources
   - Marche aléatoire sur une grille
   - Autres chaînes de Markov particulières ?

Fin

Perspectives

1. Implémenter l'algorithme PSA-BP dans le logiciel PSI
2. Élargir à des chaînes de Markov plus générales
    - Relâcher la condition de stabilité du réseau infini :
        - en remplaçant les files instables par des sources
    - Marche aléatoire sur une grille
    - Autres chaînes de Markov particulières ?

Fin

Perspectives

1. Implémenter l'algorithme PSA-BP dans le logiciel PSI
2. Élargir à des chaînes de Markov plus générales
   - Relâcher la condition de stabilité du réseau infini :
     - en remplaçant les files instables par des sources
   - Marche aléatoire sur une grille
   - Autres chaînes de Markov particulières ?

Fin