Génération aléatoire de mots dans des langages rationnels: approche markovienne

Adrien Boussicault et **Philippe Duchon**

Aléa 2014

Le problème

- On se donne un langage rationnel L sur un alphabet à k lettres (décrit par un automate déterministe), et un entier n
- On veut un algorithme (efficace) permettant de tirer un mot aléatoire de L, uniforme parmi ceux de longueur exactement n

Le problème

- On se donne un langage rationnel L sur un alphabet à k lettres (décrit par un automate déterministe), et un entier n
- On veut un algorithme (efficace) permettant de tirer un mot aléatoire de L, uniforme parmi ceux de longueur exactement n
- (De manière équivalente : on a un graphe orienté $G = (V, E)$, avec un sommet de départ u_0 et un ensemble $F \subset V$ de sommets d'arrivée possibles ; et on veut un chemin aléatoire uniforme parmi les chemins de longueur n allant de u_0 à un sommet d'arrivée)

Le problème n'est pas nouveau. . .

- Cas particulier de la *méthode récursive* (les séries génératrices sont rationnelles ; la suite des coefficients satisfait une récurrence linéaire)

Le problème n'est pas nouveau...

- Cas particulier de la *méthode récursive* (les séries génératrices sont rationnelles ; la suite des coefficients satisfait une récurrence linéaire)
- Notations : $\delta(u, a)$ destination de l'arc étiqueté a issu de u ; $\ell_{u,v,n}$ nombre de chemins de longueur n de u à v ; $\ell_{u,F,n} = \sum_{v \in F} \ell_{u,v,n}$

Le problème n'est pas nouveau. . .

- Cas particulier de la *méthode récursive* (les séries génératrices sont rationnelles ; la suite des coefficients satisfait une récurrence linéaire)
- Notations : $\delta(u, a)$ destination de l'arc étiqueté a issu de u ; $\ell_{u,v,n}$ nombre de chemins de longueur n de u à v ; $\ell_{u,F,n} = \sum_{v \in F} \ell_{u,v,n}$
- Les algorithmes raisonnables :
  - $u = u_0$, $w = \epsilon$
  - Tant que $n > 0$:
    - Tirer une lettre a avec probabilité $p_{u,n}(a) = \ell_{\delta(u,a),F,n} / \ell_{u,F,n}$
    - $w = w.a$, $u = \delta(u, a)$

Le problème n'est pas nouveau...

- Cas particulier de la *méthode récursive* (les séries génératrices sont rationnelles ; la suite des coefficients satisfait une récurrence linéaire)
- Notations : $\delta(u, a)$ destination de l'arc étiqueté a issu de u ; $\ell_{u,v,n}$ nombre de chemins de longueur n de u à v ; $\ell_{u,F,n} = \sum_{v \in F} \ell_{u,v,n}$
- Les algorithmes raisonnables :
    - $u = u_0$, $w = \epsilon$
    - Tant que $n > 0$:
        - Tirer une lettre a avec probabilité $p_{u,n}(a) = \ell_{\delta(u,a),F,n} / \ell_{u,F,n}$
        - $w = w.a$, $u = \delta(u, a)$
- Tout est dans le **comment on calcule les** $p_{u,n}$.

Le précalcul

- $\ell_{u,v,n}$ est le coefficient (u, v) de M^n où M est la matrice d'adjacence du graphe
- Calcul de tous les $\ell_{u,v,k}$ pour $k \leq n$: $O(nq^3)$ opérations si produits de matrices naïfs ; les entiers sont typiquement exponentiels.
- Algorithme "diviser pour régner" (**Bernardi, Gimenez**) : ne calcule que $O(\log n)$ produits de matrices
- Passage en flottants (**Denise, Zimmermann**) : on calcule des approximations flottantes garanties des $\ell_{u,v,k}$; si cela suffit à la génération aléatoire, super ; sinon, on se rabat sur le calcul en entiers plus coûteux (mais avec bonne probabilité, ça n'arrive pas)
- En combinant les deux, Bernardi et Gimenez obtiennent un algorithme qui fait précalcul et génération aléatoire en complexité (binaire) moyenne linéaire (mais avec, rarement, un temps qui explose)

Et les chaînes de Markov ?

- On aimerait un algorithme à base de simulation d'une chaîne de Markov (homogène en temps), quitte à rajouter un peu de rejet.

Et les chaînes de Markov ?

- On aimerait un algorithme à base de simulation d'une chaîne de Markov (homogène en temps), quitte à rajouter un peu de rejet.
- Autre façon de voir les choses : *est-ce que les mots du langages ressemblent, en un sens à préciser, aux trajectoires d'une chaîne de Markov ?*

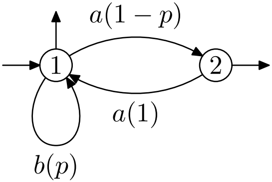
Et les chaînes de Markov ?

- On aimerait un algorithme à base de simulation d'une chaîne de Markov (homogène en temps), quitte à rajouter un peu de rejet.
- Autre façon de voir les choses : *est-ce que les mots du langages ressemblent, en un sens à préciser, aux trajectoires d'une chaîne de Markov ?*
- Tirer un mot aléatoire uniforme dans A^n, et rejeter tant qu'on n'a pas un mot de L, serait exponentiellement lent pour beaucoup de langages.

Et les chaînes de Markov ?

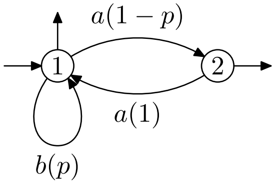
- On aimerait un algorithme à base de simulation d'une chaîne de Markov (homogène en temps), quitte à rajouter un peu de rejet.
- Autre façon de voir les choses : *est-ce que les mots du langages ressemblent, en un sens à préciser, aux trajectoires d'une chaîne de Markov ?*
- Tirer un mot aléatoire uniforme dans A^n, et rejeter tant qu'on n'a pas un mot de L, serait exponentiellement lent pour beaucoup de langages.
- On peut essayer de *biaiser* la marche.

Un exemple jouet : mots de Fibonacci



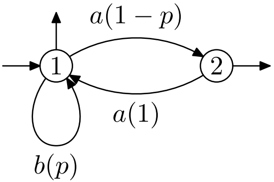
- Au départ de l'état 1, le mot *aa* a probabilité p^2, et le mot *ba* a probabilité $1-p$.

Un exemple jouet : mots de Fibonacci



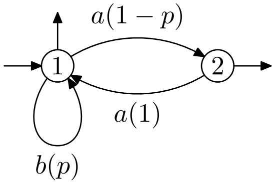
- Au départ de l'état 1, le mot *aa* a probabilité p^2, et le mot *ba* a probabilité $1-p$.
- Au total, les mots a^{2m} et $(ba)^m$ auront des probabilités très différentes, **sauf si** on prend $p^2 = 1-p$.

Un exemple jouet : mots de Fibonacci



- Au départ de l'état 1, le mot *aa* a probabilité p^2, et le mot *ba* a probabilité $1 - p$.
- Au total, les mots a^{2m} et $(ba)^m$ auront des probabilités très différentes, **sauf si** on prend $p^2 = 1 - p$.
- Avec $p^2 + p - 1 = 0$ ($p = \frac{-1+\sqrt{5}}{2} = 1/\Phi$),
  - un chemin de longueur n, de 1 à 1, a probabilité $1/\Phi^n$
  - un chemin de longueur n, de 1 à 2, a probabilité $1/\Phi^{n+1}$

Un exemple jouet : mots de Fibonacci



- Au départ de l'état 1, le mot *aa* a probabilité p^2, et le mot *ba* a probabilité $1-p$.
- Au total, les mots a^{2m} et $(ba)^m$ auront des probabilités très différentes, **sauf si** on prend $p^2 = 1-p$.
- Avec $p^2 + p - 1 = 0$ ($p = \frac{-1+\sqrt{5}}{2} = 1/\Phi$),
  - un chemin de longueur n, de 1 à 1, a probabilité $1/\Phi^n$
  - un chemin de longueur n, de 1 à 2, a probabilité $1/\Phi^{n+1}$
- **Algorithme :** Faire une marche de longueur n ainsi biaisée ; si on termine en 2, accepter le mot ; en 1, accepter avec probabilité $1/\Phi$ (sinon, recommencer).

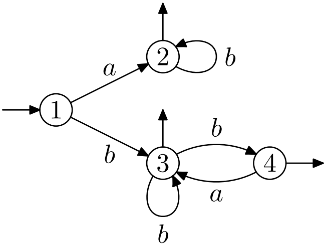
On généralise

On cherche un algorithme de la forme :

- **[Précalcul]** Choisir astucieusement les probabilités de transition d'une marche aléatoire biaisée sur le graphe, et des réels $(q_u)_{u \in V}$ entre 0 et 1 ($q_u = 0$ dès que $u \notin F$)
- **[Tirage]** Simuler une marche de longueur n dans le graphe, terminant en un sommet u ; accepter le mot avec probabilité q_u, sinon recommencer le tirage.

Question : est-ce qu'il existe des paramètres rendant **correcte** (uniformité sur les mots du langage) et **efficace** (linéaire en moyenne) la phase de tirage ?

Mauvaise nouvelle : pas toujours



- Le langage contient $1 + F_{n-1}$ mots de longueur n, dont 1 seul commence par a
- Quelle que soit la probabilité positive qu'on accorde à la transition de 1 vers 2, on accorde exponentiellement trop de poids à l'unique mot commençant par a ; et avec une probabilité nulle, on n'engendre pas tout le langage
- Bref, tel quel notre algorithme ne peut pas marcher pour ce langage (pourtant simple !).

Le cas des automates fortement connexes

- On se restreint au cas où l'automate est **fortement connexe** (en termes de marches : l'ensemble des états sera récurrent)
- **Notre résultat :**
  - Sous l'hypothèse de forte connexité, il existe une unique marche biaisée rendant notre algorithme correct, avec complexité moyenne linéaire ;
  - on sait décrire explicitement les probabilités de transition.

Mise en équations

- Une condition **nécessaire** portant sur les probabilités de transition est la suivante : sur tous les circuits orientés du graphe, la moyenne harmonique des probabilités de transition est la même.

Mise en équations

- Une condition **nécessaire** portant sur les probabilités de transition est la suivante : sur tous les circuits orientés du graphe, la moyenne harmonique des probabilités de transition est la même.
- Autrement dit : il existe un réel $p > 0$ tel que, pour tout circuit orienté (simple, ça suffit), le produit des probabilités de transition est $p^{|C|}$.

Mise en équations

- Une condition **nécessaire** portant sur les probabilités de transition est la suivante : sur tous les circuits orientés du graphe, la moyenne harmonique des probabilités de transition est la même.
- Autrement dit : il existe un réel $p > 0$ tel que, pour tout circuit orienté (simple, ça suffit), le produit des probabilités de transition est $p^{|C|}$.
- (Si on a deux circuits C et C' de moyennes harmoniques distinctes, on peut former un chemin γ, terminant en un état final, et contenant au moins un sommet de C et de C' ; en ajoutant $n|C|$ tours de C', ou $n|C'|$ tours de C, on obtient deux chemins de même longueur et de probabilités différentes)

Mise en équations (bis)

- La condition est **suffisante** : si on pose, pour chaque arête e, $p'_e = p_e/p$, la probabilité de tout chemin γ de longueur n, finissant en un sommet v, est $p_\gamma = p^n p_v$, avec

$$p_v = \prod_{e \in \gamma_v} p'_e,$$

  où γ_v est un chemin simple de u_0 à v (ne dépend pas du chemin) ;
- Il ne reste plus qu'à poser $q_u = \frac{\min_{v \in F} p_v}{p_u}$ pour les sommets $u \in F$, et on a notre marche et nos probabilités d'acceptation.

Récapitulons

On cherche des réels positifs $p, (p_e)_{e \in E}$ satisfaisant un système :

- Pour $u \in V$,

$$\sum_{e=(u,v) \in E} p_e = 1$$

- Pour C circuit orienté (simple) du graphe,

$$\prod_{e \in C} p_e = p^{|C|}$$

Récapitulons

On cherche des réels positifs $p, (p_e)_{e \in E}$ satisfaisant un système :

- Pour $u \in V$,
$$\sum_{e=(u,v) \in E} p_e = 1$$
- Pour C circuit orienté (simple) du graphe,
$$\prod_{e \in C} p_e = p^{|C|}$$
- **Remarque :** pour les équations de circuits, il suffit de poser les équations pour une base des cycles du graphe ; on n'a que $m - n + 1$ équations indépendantes.

Récapitulons

On cherche des réels positifs $p, (p_e)_{e \in E}$ satisfaisant un système :

- Pour $u \in V$,

$$\sum_{e=(u,v) \in E} p_e = 1$$

- Pour C circuit orienté (simple) du graphe,

$$\prod_{e \in C} p_e = p^{|C|}$$

- **Remarque :** pour les équations de circuits, il suffit de poser les équations pour une base des cycles du graphe ; on n'a que $m - n + 1$ équations indépendantes.
- Au total, on se retrouve avec $m + 1$ inconnues, n équations "de sommets" et $m - n + 1$ équations "de cycles"

Résolution

Proposition

Le système admet une unique solution, et les inconnues ont des valeurs strictement positives.

Proposition

Une (la) solution du système est donnée par

- $p = \rho$
- Pour $e = (u, v)$, $p_e = \rho \tilde{S}_{v,u}(\rho)$

où ρ est le rayon de convergence de la série génératrice des chemins du graphe, et $\tilde{S}_{v,u}(t)$ est la série génératrice des chemins issus de v, terminant en u, et n'empruntant aucune arête issue de u

Séries des chemins dans un graphe

(G fortement connexe)

- $S_{u,v}(t)$ série génératrice des chemins de u à v
- Toutes ces séries ont le même rayon de convergence $\rho > 0$, et un pôle simple en ρ

Séries des chemins dans un graphe

(G fortement connexe)

- $S_{u,v}(t)$ série génératrice des chemins de u à v
- Toutes ces séries ont le même rayon de convergence $\rho > 0$, et un pôle simple en ρ
- Si on supprime une ou plusieurs arêtes, tous les rayons de convergence augmentent strictement (et donc les séries se mettent à converger en ρ)

Séries des chemins dans un graphe

(G fortement connexe)

- $S_{u,v}(t)$ série génératrice des chemins de u à v
- Toutes ces séries ont le même rayon de convergence $\rho > 0$, et un pôle simple en ρ
- Si on supprime une ou plusieurs arêtes, tous les rayons de convergence augmentent strictement (et donc les séries se mettent à converger en ρ)
- (En particulier, la solution proposée est bien définie)

Vérification des équations de sommets

- On veut vérifier que pour tout u,

$$\rho \sum_{v:(u,v)\in E} \tilde{S}_{v,u}(\rho) = 1$$

Vérification des équations de sommets

- On veut vérifier que pour tout u,

$$\rho \sum_{v:(u,v)\in E} \tilde{S}_{v,u}(\rho) = 1$$

- $\tilde{S}_u(t) = t \sum_v \tilde{S}_{v,u}(t)$ est la série génératrice de tous les chemins non vides de u à u, arrêtés dès qu'ils reviennent en u

Vérification des équations de sommets

- On veut vérifier que pour tout u,

$$\rho \sum_{v:(u,v)\in E} \tilde{S}_{v,u}(\rho) = 1$$

- $\tilde{S}_u(t) = t \sum_v \tilde{S}_{v,u}(t)$ est la série génératrice de tous les chemins non vides de u à u, arrêtés dès qu'ils reviennent en u
- $S_{u,u}(t) = 1/(1 - \tilde{S}_u(t))$ a un pôle en ρ, donc forcément $\tilde{S}_u(\rho) = 1$.

Vérification des équations de circuits

- On veut vérifier que pour tout circuit $C = (v_0, v_1, \ldots, v_k = v_0)$,

$$\prod_{i=1}^{k} \tilde{S}_{v_i, v_{i-1}}(\rho) = 1$$

- Donc la série $S_{v_0,v_0} - \tilde{S}_C$ des autres chemins de v_0 à v_0, a un pôle en ρ

Vérification des équations de circuits

- On veut vérifier que pour tout circuit $C = (v_0, v_1, \ldots, v_k = v_0)$,

$$\prod_{i=1}^{k} \tilde{S}_{v_i, v_{i-1}}(\rho) = 1$$

- Facile : la série génératrice $\tilde{S}_C(t)$ des chemins de v_0 à v_0 qui n'empruntent **pas**, dans cet ordre, les sommets $v_{k-1}, v_{k-2}, \ldots, v_1$, converge en ρ.
- Donc la série $S_{v_0,v_0} - \tilde{S}_C$ des autres chemins de v_0 à v_0, a un pôle en ρ

Vérification des équations de circuits

- On veut vérifier que pour tout circuit $C = (v_0, v_1, \dots, v_k = v_0)$,

$$\prod_{i=1}^{k} \tilde{S}_{v_i, v_{i-1}}(\rho) = 1$$

- Facile : la série génératrice $\tilde{S}_C(t)$ des chemins de v_0 à v_0 qui n'empruntent **pas**, dans cet ordre, les sommets $v_{k-1}, v_{k-2}, \dots, v_1$, converge en ρ.
- Donc la série $S_{v_0,v_0} - \tilde{S}_C$ des autres chemins de v_0 à v_0, a un pôle en ρ
- Mais cette série s'écrit

$$\frac{1}{1 - \prod_{i=1}^{k} \tilde{S}_{v_i, v_{i-1}}(t)},$$

et donc le dénominateur s'annule en ρ !

Et l'algorithme ?

- Si on oublie la question du calcul des paramètres (c'est un précalcul qu on ne ferait qu'une fois, valable pour toutes les longueurs de mots), la simulation de n pas de marche prend clairement un temps linéaire.
- La probabilité de terminer en un sommet de F est asymptotiquement loin de 0 (sauf si le graphe est périodique et que le langage ne contient pas de mots de longueur n).
- Au total, le temps moyen par génération reste linéaire en la longueur du mot engendré.
- Reste que la constante cachée dans le $\Theta(n)$ peut être peu favorable, en particulier pour de grands automates ou s'il y a peu d'états finaux.

Cas non fortement connexe

- Est-ce que la méthode peut s'étendre au cas plus général d'automates non fortement connexes ?

Cas non fortement connexe

- Est-ce que la méthode peut s'étendre au cas plus général d'automates non fortement connexes ?
- Le cas $1 + F_{n-1}$: on est obligé d'avoir des mots dont la probabilité d'acceptation dépend de n et pas seulement de l'état final

Cas non fortement connexe

- Est-ce que la méthode peut s'étendre au cas plus général d'automates non fortement connexes ?
- Le cas $1 + F_{n-1}$: on est obligé d'avoir des mots dont la probabilité d'acceptation dépend de n et pas seulement de l'état final
- On ne peut pas non plus espérer que la marche donne probabilité $\Theta(\rho^n)$ à chaque mot du langage : on peut facilement construire des langages avec pôle dominant multiple, dont $\Theta(n^\alpha \rho^{-n})$ mots de longueur n.

Cas non fortement connexe

- Est-ce que la méthode peut s'étendre au cas plus général d'automates non fortement connexes ?
- Le cas $1 + F_{n-1}$: on est obligé d'avoir des mots dont la probabilité d'acceptation dépend de n et pas seulement de l'état final
- On ne peut pas non plus espérer que la marche donne probabilité $\Theta(\rho^n)$ à chaque mot du langage : on peut facilement construire des langages avec pôle dominant multiple, dont $\Theta(n^\alpha \rho^{-n})$ mots de longueur n.
- Une piste à creuser : préchoix d'un chemin dans le graphe quotient des composantes fortement connexes, avec un second étage de rejet.