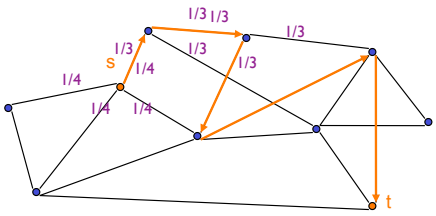
Informatique Quantique

Frédéric Magniez

Cours 6 :
Marches aléatoires et marches quantiques

Définition

- $G = (V,E)$ un graphe (non orienté) sur les sommets V d'arêtes E
- Une marche aléatoire est un déplacement aléatoire
  - sur les sommets V de G
  - en suivant les arêtes E de G
  - tel que $\Pr[\, u \to v \,] = 1/\deg(u)$



Application typique

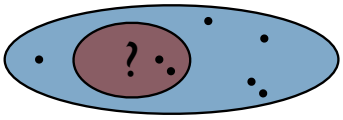
- Recherche de chemin de s à t

Mélange

- Comment mélanger un jeux de cartes et en combien de temps ?

Estimation

- Comptage, surface, volume
- Page rank de Google



Parcours

- Comment envoyer un message au Dalai Lama, le plus rapidement possible ? (pas de recours à l'annuaire)
- Quel est le comportement d'une particule aléatoire
  sur une ligne, dans le plan, l'espace, ... ?
- Comment sortir d'un labyrinthe avec peu de mémoire ?

Modéliser/Analyser des comportements

- Algorithmes probabilistes
- Réseaux, Jeux, Finance

Recherche

- Comment trouver une information dans une base de données gigantesque

Définition

- Un processus stochastique (discret à espace fini) **X** est une suite de variables aléatoires X_t indexée par un paramètre *temps* (discret, $t = 0, 1, 2, 3, ...$) (espace fini, $\{X_t\}_t$ prennent un nombre fini de valeurs)
- Une chaîne de Markov est un processus stochastique tel que

  $\Pr[X_t = a_t \mid X_{t-1} = a_{t-1}, X_{t-2} = a_{t-2}, \dots, X_0 = a_0]$

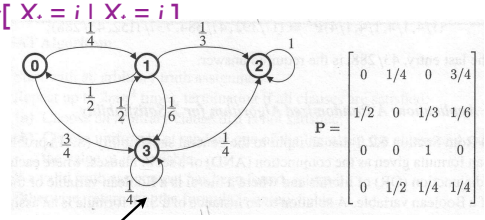
  $= \Pr[X_t = a_t \mid X_{t-1} = a_{t-1}]$ pas d'effet mémoire

  $= P(a_t, a_{t-1})$ indépendant du temps

  en résumé l'évolution ne dépend que de la valeur courante
- Matrice de transition : $P(i,j) = \Pr[X_t = i \mid X_{t-1} = j]$

Remarque.

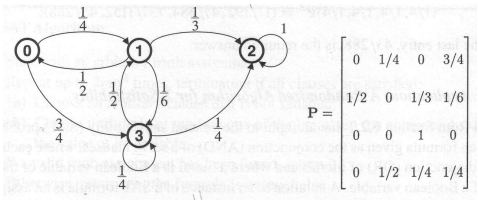
- Etat de X_t : $P_t(i) = \Pr[X_t = i]$
- $P_t = P_0 \times P^t$



Exemples

- Marches aléatoires
- Marches aléatoires avec des boucles : $P' = 1/2\ P + 1/2\ \mathrm{Id}$
- Somme convexe de marches aléatoires : $P'' = 3/4\ P + 1/4\ P'$

Exercice



- Calculer la probabilité d'aller de 0 à 3 en exactement 3 étapes
- Calculer la probabilité d'arriver à 3 en partant d'un état aléatoire et en exactement 3 étapes

SAT

- Entrée : suite de clauses sur n variables

  x_1 ou $\overline{x_2}$ ou $\overline{x_5}$

  $\overline{x_4}$ ou $\overline{x_3}$ ou x_2

- Sortie : une assignation $a \in \{0,1\}^n$ qui satisfait **toutes** les clauses

k-SAT

- Restriction : chaque clause utilise au plus k variables

Théorème

- 3-SAT est NP-complet
- 2-SAT est résoluble en temps linéaire et espace linéaire
- 2-SAT est résoluble en temps quadratique et espace logarithmique

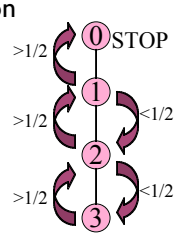
Algorithme

- Choisir une assignation quelconque a
- Répéter $2m \times n^2$ fois, et s'arrêter si a satisfait toutes les clauses
  - Choisir une clause C arbitraire non satisfaite
  - Choisir aléatoirement une des variables x_i de C
  - Changer la valeur du bit i de a correspondant à cette variable
- Si a satisfait toutes les clauses, renvoyer a
- Sinon renvoyer que toutes les clauses ne sont satisfiables simultanément

Analyse de l'algorithme

- Fixer une solution (potentielle) s
- X_t = nombre de bits différents entre a et s à la t-ème itération
- Nécessairement

$$\begin{aligned}\Pr[X_{t+1} = n-1 | X_t = n] &= 1\\ \Pr[X_{t+1} = 0 | X_t = 0] &= 1\\ \Pr[X_{t+1} = j-1 | X_t = j] &\geq 1/2\end{aligned}$$



Suite de l'analyse

- Version pessimiste du comportement de l'algorithme

$$\begin{aligned}Y_0 &= n\\ \Pr[Y_{t+1} = n-1 | Y_t = n] &= 1\\ \Pr[Y_{t+1} = 0 | Y_t = 0] &= 1\\ \Pr[Y_{t+1} = j-1 | Y_t = j] &= 1/2\end{aligned}$$

- h_j : temps moyen d'atteindre l'état 0 en partant de l'état j

$$\begin{aligned}h_n &= 1 + h_{n-1}\\ h_0 &= 0\\ h_j &= \frac{h_{j-1} + h_{j+1}}{2} + 1\end{aligned}$$

- La résolution donne pour $j \neq 0$

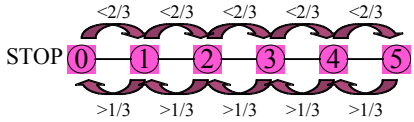
$$h_j = h_{j-1} + 2(n-j) + 1$$

- Et donc $h_n = \sum_{i=0}^{n} (2i+1) = n^2$
- D'après l'inégalité de Markov, La probabilité de ne pas trouver de solution après $2n^2$ étapes est au plus $1/2$, et après $2mn^2$ étapes est au plus $1/2^m$

Que se passe-t-il pour 3-SAT ?

Même algorithme :

- Analyse montre qu'on a tendance à s'éloigner d'une solution



Alors ?

- Idée 1 : partir d'une assignation aléatoire
- Idée 2 : recommencer à partir d'une assignation aléatoire si aucune solution n'est trouvée au bout de $3n$ itérations

Théorème

- L'algorithme ainsi modifié trouve une solution en temps moyen $(4/3)^n$

Remarques

- Mieux que toutes les autres approches déterministe, dans le pire cas
- Toutes les marches aléatoires pour 3-SAT suivent la même structure

Propriétés des chaînes de Markov

Définition

- Une distribution stationnaire est une distribution de probabilité π telle que : $\pi = \pi P$

Définition

- Une marche aléatoire est ergodique si elle est définie sur un graphe
  - connexe : aucun sommet déconnecté du graphe
  - non biparti : les arêtes ne sont pas uniquement entre deux sous-parties disjointes

Remarque : L'ergodicité se généralise aux chaînes de Markov...

Lemme

- Toute combinaison convexe de marches aléatoires connexes est ergodique

Théorème

- Toute chaîne de Markov ergodique (discrète à espace fini)possède une unique distribution stationnaire π. De plus

$\pi_i = \lim_{t\to\infty} P^t(j,i) = 1/h_{ii}$ (h_{ii} : tps moyen pour revenir en i en partant de i)

Exercice

- Montrer que la distribution stationnaire d'une marche aléatoire est
  $\pi(u)=\deg(u)/2|E|$
- Que vaut h_{uu} ?
- Soit h_{vu} le temps moyen d'atteindre u en partant de v

  Montrer que $h_{uu} = \frac{1}{\deg(u)} \sum_{v:(v,u)\in E} (1 + h_{vu})$

  En déduire que $h_{vu} \leq 2|E|$ pour toute arête (v,u) de G

- Soit $c(G)$ le temps moyen de couverture, *i.e.* pour visiter tous les sommets de G

  En utilisant l'existence d'un arbre parcourant tous les sommets de G, montrer que $c(G) \leq 4(|V|-1)|E|$
- Application 1 : Algorithme pour 2-SAT, redémontrer $h_n=n^2$
- Application 2 : Donner un algorithme pour la recherche de chemin dans un graphe en temps n^3 et espace $\log n$

Définition

- Le temps de mélange d'une chaîne de Markov ergodique (discrète à espace fini) est le plus petit temps τ tel que $|(P^\tau)_x - \pi| \leq \frac{1}{e}$

norme L_1

Remarque

- 1/e est fictif, et pourrait être 1/1000
- Il s'agit du temps moyen pour bien *mélanger* en partant de n'importe quel état x de départ

Théorème

- La décroissance de la distance à la distribution stationnaire est monotone et exponentielle :

$$t \geq \tau \times \log(1/\varepsilon) \implies |(P^t)_x - \pi| \leq \varepsilon$$

- Le temps de mélange est en $\tau = O(1/\delta)$ où δ est l'écart spectral

écart entre la valeur propre 1 et la deuxième plus grande valeur propre de P

Exemples

- Pour bien mélanger un jeu de n cartes, il suffit de

  $n \log n$ échanges de 2 cartes prises au hasard

  $\log n$ coupe-mélanges successifs :

t	1	2	3	4	5	6	7	8
ε	1	1	1	1	0.92	0.61	0.33	0,17

Algorithme de recherche

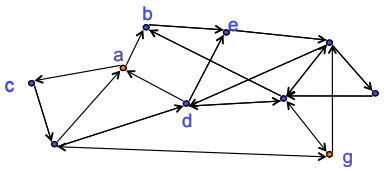
Entrée

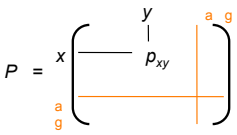
- Un ensemble $X = \{ a, b, c, \dots \}$
- Des éléments marqués M parmis X (disons a et g)
- Une boîte noire qui répond à " $x \in M$? "

Sortie

- Un élément marqué $x \in M$, s'il en existe un

Structure additionnelle : chaîne de Markov





Deux algorithmes très similaires

Algorithme I

- Partir d'un sommet x aléatoire
- Répéter $|X|/|M|$ fois
  - Si l'état courant y est marqué, alors renvoyer y et stopper
  - Sinon simuler τ étapes de la chaîne de Markov M
- Si l'algorithme n'a pas terminé, alors renvoyer "pas d'élément marqué"

Algorithme II

- Partir d'un sommet x aléatoire
- Répéter $\tau \times |X|/|M|$ fois
  - Si l'état courant y est marqué, alors renvoyer y et stopper
  - Sinon simuler **une** étape de la chaîne de Markov M
- Si l'algorithme n'a pas terminé, alors renvoyer "pas d'élément marqué"

Théorème

- Les deux algorithmes trouvent un élément marqué, s'il en existe un, avec grande probabilité

Exercice : Justifier le théorème

Le problème Répétition (Element Distinctness)

Entrée : Une liste de n nombres $\{ x_1, x_2, x_3, \ldots, x_n \}$

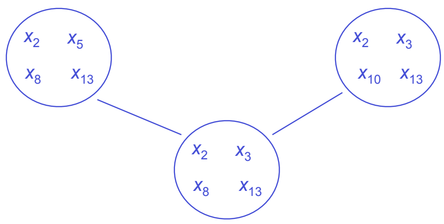
Sortie : Une paire de collision $i \neq j$ telle que $x_i = x_j$

Algorithme déterministe

- Trier les éléments, et vérifier s'il y en a deux consécutifs égaux
- Complexité : $n \log n$

Version marche aléatoire sur le graphe de Johnson

- Etats : sous-ensembles de taille r d'indices de $\{ 1, 2, 3, \ldots, n \}$
- Arêtes entre les sous-ensembles qui diffèrent exactement de 2 éléments
- Idée : marcher sur le graphe de Johnson en maintenant les x_i triés correspondant à chacun des indices du sous-ensemble courant
- Exemple : $n = 15$ et $r = 4$



Cas du problème Répétition

Distribution stationnaire

- π est la distribution uniforme (pourquoi ?)

Ecart spectral du graphe de Johnson

- $\delta \approx 1 / r$

Fractions des états marqués

- Cas d'une unique paire de collision : $\varepsilon = |M| / |X| \approx (r/n)^2$

Complexité

- $r + (n/r)^2 (r^2 \times \log n + r)$

  (init cost : r ; $1/\varepsilon$: $(n/r)^2$; $1/\delta$: r ; update cost : $r \times \log n$; checking cost : r)

  init cost — update cost — checking cost

- Indépendant de r : $n \log n$

Amélioration quantique

- On peut remplacer ε et δ par leurs racines carrées !
- $r + (n/r) (r \times \log n + r)$
- Valeur optimale de $r = n^{2/3}$: $n^{2/3} \log n$

Quantization d'une marche aléatoire *P*

Etats

- Paires de sommets *x*,*y* tq (*x*,*y*) ∈ *E* : $|x\rangle|y\rangle$

Une étape de marche quantique *W(P)*

- Diffuser *y* sur les voisins de *x* :
  symétrie par rapport à l'état $|x\rangle \sum_y \sqrt{P(x,y)}|y\rangle$
- Diffuser *x* sur les voisins de *y* :
  symétrie par rapport à l'état $\sum_x \sqrt{P(x,y)}|x\rangle|y\rangle$

Propriétés spectrale de *W(P)*

- Les valeurs propres de *W(P)* sont reliées à celles de *P* par
  exp(±2i θ) ⟷ cos θ
- Idée : *W(P)* est le produit de deux réflexions...
- La valeur propre 1 est associée au vecteur propre

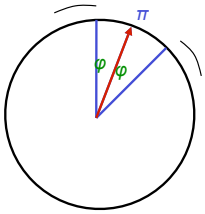
$$|\pi\rangle = |x\rangle \sum_y \sqrt{P(x,y)}|y\rangle \sum_{x,y} \sqrt{\pi_x}\sqrt{P(x,y)}|x\rangle|y\rangle$$

Exercice

- Quel est l'écart spectral *W(P)* de en fonction de celui de *P* ?

Estimation de phase et Grover

Exercice : partie estimation de phase

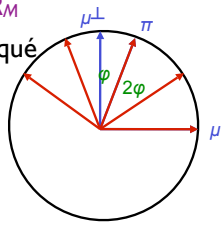


- Montrer qu'il est possible de distinguer $|\pi\rangle$ de tout autre vecteur propre en utilisant O(1/√ δ) fois la marche quantique *W(P)* (avec petite erreur)
- En déduire une réalisation de la symétrie orthogonale R_π par rapport à $|\pi\rangle$ (avec petite erreur)

Exercice : partie Grover

- Soit *Q* la projection orthogonale sur les états marqués sur le premier registre $\mathcal{M} = \mathrm{Vect}\{|x\rangle|y\rangle : x \in M\}$
  Comment réaliser la symétrie orthogonale R_M par rapport à $\mathcal{M}$?
- Soit $|\mu\rangle$ la projection renormalisée de $|\pi\rangle$ par *Q*
  Exprimer $\langle\mu|\pi\rangle$ en fonction de ε = $\Pr_\pi$[x marqué]
- Montrer que le plan $P = \mathrm{Vect}(|\pi\rangle, |\mu\rangle)$ est stable par R_π et R_M
- En déduire un algorithme permettant de trouver un élément marqué
- Montrer que la complexité de votre algorithme est en

$$\text{init cost} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}} \times \text{update cost} + \text{checking cost}\right)$$



D'autres applications (complexité de requêtes)

Vérifier le produit de 2 matrices

- Entrée : 3 matrices A, B et C de taille $n \times n$
- Sortie : accepter ssi $A \times B = C$
- Complexité probabiliste : $\Theta(n^2)$ Complexité quantique : $\Omega(n^{1.5})$ $O(n^{5/3})$

Chercher un triangle dans un graphe

- Entrée : graphe G à n sommets
- Sortie : 3 sommets u, v et w tel que (u,v), (v,w) et (w,u) soient 3 arêtes de G
- Complexité probabiliste : $\Theta(n^2)$ Complexité quantique : $\Omega(n)$ $O(n^{1.3})$

Test de commutativité

- Entrée : n éléments d'un groupe *inconnu* (requête = opération du groupe)
- Sortie : accepter ssi le groupe généré est commutatif
- Complexité probabiliste : $\Theta(n)$ Complexité quantique : $\Theta(n^{2/3})$