IFT2125 - Introduction à l'algorithmique

Algorithmes probabilistes (B&B chapitre 10)

Pierre McKenzie

DIRO, Université de Montréal

Automne 2017

Caractéristiques des algorithmes probabilistes

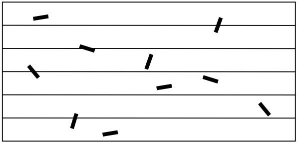
B&B section 10.1

- jouent à pile ou face
- se comportent différemment, exécutés deux fois sur le même exemplaire
- peuvent se tromper
- défient parfois l'intuition.

Première surprise

Le hasard peut être utile

- créer un système solaire (ici toujours en attente de verdict)
- estimer π



- permettre certains protocoles cryptographiques
- vérifier la primalité rapidement
- accélérer une recherche
- réduire l'effet de mauvais exemplaires.

Exemple : pile=succès, face=échec

- $\Pr[\text{un succès, après un essai}] = \frac{1}{2}$
- $\Pr[\text{un succès ou plus, après 2 essais}] =$

Exemple : pile=succès, face=échec

- Pr[un succès, après un essai] $= \frac{1}{2}$
- Pr[un succès ou plus, après 2 essais] $= \frac{3}{4}$
- Pr[un succès ou plus, après 3 essais] $=$

Deuxième surprise

Le hasard peut être précis

Exemple : pile=succès, face=échec

- $\Pr[\text{un succès, après un essai}] = \frac{1}{2}$
- $\Pr[\text{un succès ou plus, après 2 essais}] = \frac{3}{4}$
- $\Pr[\text{un succès ou plus, après 3 essais}] = \frac{7}{8}$
- ...
- $\Pr[\text{un succès ou plus, après } n \text{ essais}] =$

Deuxième surprise

Le hasard peut être précis

Exemple : pile=succès, face=échec

- Pr[un succès, après un essai] $= \frac{1}{2}$
- Pr[un succès ou plus, après 2 essais] $= \frac{3}{4}$
- Pr[un succès ou plus, après 3 essais] $= \frac{7}{8}$
- ...
- Pr[un succès ou plus, après n essais] $= 1 - (\frac{1}{2})^n$

$\Rightarrow$ une suite de 1000 échecs consécutifs est moins probable qu'une erreur interne de l'ordinateur après une seconde de calcul !

Trois types d'algorithmes probabilistes

B&B section 10.2

- Numérique
  - solution approximative à un problème numérique (ex : simulation)
  - plus de temps $\Rightarrow$ plus de précision.
- Monte Carlo
  - toujours une réponse (ex : oui ou non)
  - souvent impossible de vérifier efficacement la réponse
  - plus de temps $\Rightarrow$ meilleure proba de bonne réponse.
- Las Vegas
  - jamais de réponse inexacte, mais parfois sans réponse
  - plus de temps $\Rightarrow$ meilleure proba de réponse.

Trois types

Quand Christophe Colomb a-t-il atteint l'Amérique ?

- Numérique
  Au 15ième siècle
  entre 1493 et 1499
  entre 1489 et 1496

- Monte Carlo
  1492, 1501, 567, 765, 1492, 1487, 1488, 1501, 1500, ...

- Las Vegas
  1492, 1492, sais pas, sais pas, 1492, sais pas, 1492, 1492, ...

Temps moyen vs. temps espéré

B&B section 10.3

- Rappel :

$$t_{\text{moyen}}(n) = \frac{\sum_{|w|=n} \text{temps}(w)}{\#\{w : |w| = n\}}$$

- Temps espéré d'abord défini sur chaque exemplaire :

$$t_{\text{espéré}}(w) = \sum_{\text{suites } \sigma \text{ de piles/faces menant à l'arrêt}} (\text{temps}(w \text{ avec } \sigma)) \times \Pr[\sigma]$$

- puis $t_{\text{espéré}}(n) = \max_{|w|=n} t_{\text{espéré}}(w)$

Nombres pseudo-aléatoires

B&B Section 10.4

Comment générer m bits (pseudo-) aléatoires ?

- pas si simple
- une méthode possible :
  - choisir p et q deux premiers $\equiv 3 \mod 4$ d'une centaine de chiffres
  - former entier z de 200 chiffres en utilisant l'heure en pico-secondes
  - vérifier que pgcd$(z, pq) = 1$
  - pour $i \leftarrow 1$ à m faire [$z \leftarrow z \times z \mod pq$; imprimer parité(z)]
- suite obtenue presque toujours indistingable d'une suite aléatoire, même si la suite se répétera à coup sûr si m très grand
- cf. Pierre L'Écuyer du DIRO

- Pour estimer $I = \int_a^b f(x)\mathrm{d}x$, l'idée :
  - estimer la hauteur $I/(b-a)$ du rectangle

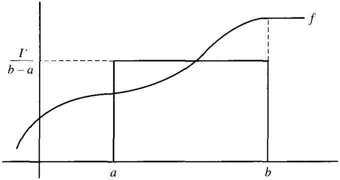


Figure 10.2. Numerical integration

  - multiplier par $b-a$.
- Solution déterministe :
  - prendre m points équidistants entre a et b inclusivement
  - évaluer f à chacun de ces points
  - prendre la moyenne, voilà $I/(b-a)$.
- Solution probabiliste :
  - engendrer les m points entre a et b au hasard
- Quelle méthode est la meilleure ?

Algorithmes de Monte Carlo

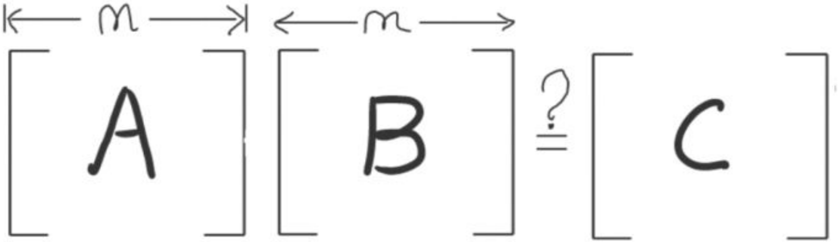
B&B section 10.6

- répond toujours
- peut se tromper
- aucun avertissement en cas d'erreur
- mais réussit avec bonne probabilité sur tout exemplaire

L'algo est *p*-correct, $0 < p < 1$, si $\Pr[\text{bonne réponse}] \geq p$.

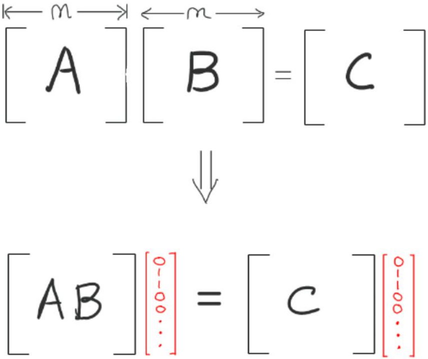
Monte Carlo : exemple 1

Vérifier en $O(n^2)$ que $AB = C$, B&B section 10.6.1



L'idée

Choisir $X \in \{0,1\}^n$ au hasard et exploiter



en répondant oui si cette dernière identité est vérifée, non sinon.

- Alors pas d'erreur possible lorsque $AB = C$: cool !
- Mais quelle probabilité d'erreur si $AB \neq C$?
- Et comment vérifier cette dernière identité en $O(n^2)$?

Mais quelle probabilité d'erreur si $AB \neq C$?

$$\begin{bmatrix} & & \\ & AB - C & \\ & & \end{bmatrix} \overbrace{\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ \vdots \\ \cdot \end{bmatrix}}^{\text{au hasard}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$\exists \; \bullet_{ij} = d_{ij} \neq 0$ et écrivons r_j et $p_i = d_{ij} r_j + y$

Suite en classe. On obtient : $\leq \frac{1}{2}$.

Monte Carlo : exemple 1

(suite)

En résumé :

- sur exemplaire $AB = C$, $\Pr[\text{erreur}] = 0$
- sur exemplaire $AB \neq C$, $\Pr[\text{erreur}] \leq \frac{1}{2}$
- dans tous les cas, $\Pr[\text{erreur}] \leq \frac{1}{2}$
- en prime, biaisé car aucune erreur sur exemplaires positifs
  - plus précisément, faux-biaisé car si l'algo répond “faux” il ne se trompe jamais, i.e., $AB \neq C$.

Monte Carlo : exemple 1
(suite)

Peut-on réduire l'erreur en répétant k fois ? Voyons :

- faux-biaisé $\Rightarrow$

Monte Carlo : exemple 1

(suite)

Peut-on réduire l'erreur en répétant k fois ? Voyons :

- faux-biaisé $\Rightarrow$
  - on peut conclure dès la première réponse "faux"
  - on ne répondra "vrai" qu'après k réponses "vrai"
- l'erreur sur exemplaire $AB = C$?

Monte Carlo : exemple 1
(suite)

Peut-on réduire l'erreur en répétant k fois ? Voyons :

- faux-biaisé $\Rightarrow$
  - on peut conclure dès la première réponse "faux"
  - on ne répondra "vrai" qu'après k réponses "vrai"
- l'erreur sur exemplaire $AB = C$?
  - toutes les réponses seront vrai
  - $\Pr[\text{erreur}] = 0$
- l'erreur sur exemplaire $AB \neq C$?

Monte Carlo : exemple 1

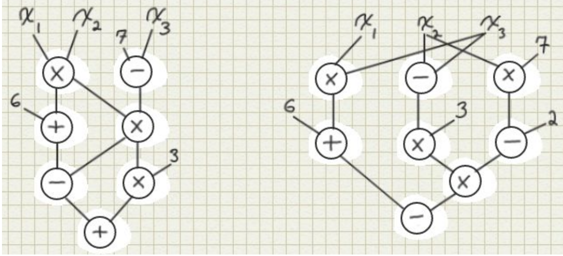
(suite)

Peut-on réduire l'erreur en répétant k fois ? Voyons :

- faux-biaisé $\Rightarrow$
  - on peut conclure dès la première réponse "faux"
  - on ne répondra "vrai" qu'après k réponses "vrai"
- l'erreur sur exemplaire $AB = C$?
  - toutes les réponses seront vrai
  - $\Pr[\text{erreur}] = 0$
- l'erreur sur exemplaire $AB \neq C$?
  - scénario semblable à "pile=succès" et "face=échec"
  - $\Pr[k \text{ échecs consécutifs}] \leq (\frac{1}{2})^k$
- dans tous les cas, $\Pr[\text{erreur}] \leq (\frac{1}{2})^k$
- l'algorithme répété k fois de cette façon est $1 - (\frac{1}{2})^k$-correct

TIP

DONNÉE: deux polynômes décrits par circuit arithmétique
DÉCIDER: si ces deux polynômes sont identiques
Exemple :



En démo : un algo de Monte Carlo fonctionnant en temps polynomial.
Fait notoire : aucun algorithme polynomial non probabiliste résolvant ce problème n'est connu à l'heure actuelle.

Monte Carlo : exemple 3

Test de primalité, B&B section 8.6.2

PRIMALITÉ

DONNÉE: un entier m en binaire
DÉCIDER: si m est premier

- un algo de Monte Carlo en $O(n^3)$ existe ($n = \log_2 m$)
- m est premier $\rightarrow \Pr[\text{erreur}] = 0$
- m n'est pas premier $\rightarrow \Pr[\text{erreur}] \leq \frac{1}{4}$
- amplifier la probabilité de succès par répétitions est donc possible

Cet algo est faux-biaisé ou vrai-biaisé ?

Monte Carlo : exemple 3

Test de primalité, B&B section 8.6.2

<u>PRIMALITÉ</u>

DONNÉE: un entier m en binaire
DÉCIDER: si m est premier

- un algo de Monte Carlo en $O(n^3)$ existe ($n = \log_2 m$)
- m est premier $\rightarrow \Pr[\text{erreur}] = 0$
- m n'est pas premier $\rightarrow \Pr[\text{erreur}] \leq \frac{1}{4}$
- amplifier la probabilité de succès par répétitions est donc possible

Cet algo est faux-biaisé ou vrai-biaisé ?
Faux-biaisé car ne répond jamais "non" quand le nombre est premier

Monte Carlo : exemple 3

Primalité (suite)

- un algo polynomial non probabiliste n'existe que depuis 2002
- l'algo est compliqué
- il requiert temps $\Omega(n^5)$
- l'algo de Monte Carlo est toujours utilisé dans la pratique

Deux cas :

- Algo biaisé
- Algo non biaisé

Cas biaisé

Exemple : un algo $A(x)$ à réponse vrai/faux, vrai-biaisé et $\frac{3}{4}$-correct

Comment amplifier ce $\frac{3}{4}$?

Cas biaisé

Exemple : un algo $A(x)$ à réponse vrai/faux, vrai-biaisé et $\frac{3}{4}$-correct

Comment amplifier ce $\frac{3}{4}$?
Déjà vu :

```
fonction ampli_biaisée(x,k)
    pour i = 1 à k faire
        si A(x) alors
            retourner vrai
    retourner faux
```

Autrement dit : on s'arrête dès le premier **vrai**, sans quoi on répond **faux**.

Cet exemple, avec $k = 3$ répétitions

- Sur un exemplaire x "faux" :
  - $A(x)$ ne répondra jamais "vrai" car $A(x)$ vrai-biaisé
  - ampli_biaisée(x,3) concluera "faux"
  - Pr[ampli_biaisée(x,3) se trompe] $= 0$.

- Sur un exemplaire x "vrai"
  - seule possibilité d'erreur $=$ $A(x)$ répond "faux" 3 fois
  - Pr[err, err, err] $= \frac{1}{4} \times \frac{1}{4} \times \frac{1}{4}$
  - Pr[ampli_biaisée(x,3) se trompe] $= \frac{1}{64} \approx 2\%$.

- Passé de 75%-correct à 98%-correct

Même exemple $A(x)$, mais cette fois $p = \frac{1}{100}$-correct

- Ici $A(x)$ se trompe 99 fois sur 100, pourtant...
- ...amplifier est possible, car seule possibilité d'erreur de $A(x)$ est toujours de répondre "faux" sur un exemplaire vrai :
  - $\Pr[k \text{ erreurs consécutives de } A(x)] = \left(\frac{99}{100}\right)^k$
  - ampli_biaisée(x,10) : est $\approx$ 10%-correct
  - ampli_biaisée(x,30) : est $\approx$ 25%-correct
  - ampli_biaisée(x,140) : est $\approx$ 75%-correct
  - ampli_biaisée(x,300) : est $\approx$ 95%-correct.

Cas biaisé : morale

- vrai-biaisé ou faux-biaisé nous aide
- permet de conclure dès réponse “dans le sens du biais”
- amplifier est possible quel que soit $p > 0$

Cas non biaisé maintenant

Problème à réponse vrai/faux, algo p-correct

Seule option :

- répéter k fois et prendre vote majoritaire
- voici la fonction (prendre k impair pour éviter ambiguité) :

```
fonction ampli_non_biaisée(x,k)
    V = 0
    pour i = 1 à k faire
        si A(x) alors V =+ 1
    si V > k/2 alors
        retourner vrai
    sinon
        retourner faux
```

Cas non biaisé
(suite)

- Mais, “reality check”, considérons :

```
fonction pas_terrible
        si pile alors
            retourner vrai
        sinon
            retourner faux
```

- Cette fonction
  - résout n'importe quel problème à réponse vrai/faux
  - est $\frac{1}{2}$-correcte
  - est (bien sûr !) non biaisée
- Alors quoi ? ?

Cas non biaisé
(suite)

- Mais, “reality check”, considérons :

```
fonction pas_terrible
        si pile alors
            retourner vrai
        sinon
            retourner faux
```

- Cette fonction
  - résout n'importe quel problème à réponse vrai/faux
  - est $\frac{1}{2}$-correcte
  - est (bien sûr !) non biaisée
- Alors quoi ? ?
  - Utopique d'attendre un miracle lorsque p-correct avec $p = \frac{1}{2}$
  - Mais que peut-on espérer au juste ?

Cas non biaisé
(suite)

Probabilité $P[i, k]$ de i succès parmi k tentatives d'un $A(x)$ p-correct :

$$\binom{k}{i} p^i (1-p)^{k-i}$$

Probabilité que ampli_non_biaisé(x, k) soit correct (k impair) :

$$\sum_{i=\lceil \frac{k}{2} \rceil}^{k} P[i, k]$$

Cas non biaisé

Confirmation : $p = \frac{1}{2}$ ne peut être amplifié

k	Pr[ampli_non_biaisé(x, k) correct] $= \sum_{i=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{i} p^i (1-p)^{k-i}$
1	$\frac{1}{2}$
3	$\left[\binom{3}{2} + \binom{3}{3}\right] \cdot (\frac{1}{2})^3 =$

Cas non biaisé

Confirmation : $p = \frac{1}{2}$ ne peut être amplifié

k	Pr[ampli_non_biaisé(x, k) correct] $= \sum_{i=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{i} p^i (1-p)^{k-i}$
1	$\frac{1}{2}$
3	$\left[\binom{3}{2} + \binom{3}{3}\right] \cdot \left(\frac{1}{2}\right)^3 = [3+1] \cdot \left(\frac{1}{2}\right)^3 = \frac{1}{2}$
5	$\left[\binom{5}{3} + \binom{5}{4} + \binom{5}{5}\right] \cdot \left(\frac{1}{2}\right)^5 =$

Cas non biaisé

Confirmation : $p = \frac{1}{2}$ ne peut être amplifié

k	Pr[ampli_non_biaisé(x, k) correct] $= \sum_{i=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{i} p^i (1-p)^{k-i}$
1	$\frac{1}{2}$
3	$\left[\binom{3}{2} + \binom{3}{3}\right] \cdot (\frac{1}{2})^3 = [3+1] \cdot (\frac{1}{2})^3 = \frac{1}{2}$
5	$\left[\binom{5}{3} + \binom{5}{4} + \binom{5}{5}\right] \cdot (\frac{1}{2})^5 = [10+5+1] \cdot (\frac{1}{2})^5 = \frac{1}{2}$
$\vdots$	$\vdots$
$2m-1$	$\underbrace{\left[\binom{2m-1}{m} + \cdots + \binom{2m-1}{2m-1}\right]}_{\frac{1}{2} \cdot \left(\binom{2m-1}{0} + \binom{2m-1}{1} + \cdots + \binom{2m-1}{2m-1}\right)} \cdot (\frac{1}{2})^{2m-1} =$

Cas non biaisé

Confirmation : $p = \frac{1}{2}$ ne peut être amplifié

k	Pr[ampli_non_biaisé(x,k) correct] $= \sum_{i=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{i} p^i (1-p)^{k-i}$
1	$\frac{1}{2}$
3	$\left[\binom{3}{2} + \binom{3}{3}\right] \cdot (\frac{1}{2})^3 = [3+1] \cdot (\frac{1}{2})^3 = \frac{1}{2}$
5	$\left[\binom{5}{3} + \binom{5}{4} + \binom{5}{5}\right] \cdot (\frac{1}{2})^5 = [10+5+1] \cdot (\frac{1}{2})^5 = \frac{1}{2}$
$\vdots$	$\vdots$
$2m-1$	$\underbrace{\left[\binom{2m-1}{m} + \cdots + \binom{2m-1}{2m-1}\right]}_{\frac{1}{2} \cdot \left(\binom{2m-1}{0} + \binom{2m-1}{1} + \cdots + \binom{2m-1}{2m-1}\right)} \cdot (\frac{1}{2})^{2m-1} = \frac{1}{2}$

Cas non biaisé

Aucune amplification possible à moins que $p = \frac{1}{2} + \varepsilon > \frac{1}{2}$

Mais comment calculer un k raisonnable à l'aide de l'horrible

$$\sum_{i=\lceil \frac{k}{2} \rceil}^{k} \underbrace{\binom{k}{i}(\frac{1}{2}+\varepsilon)^i(\frac{1}{2}-\varepsilon)^{k-i}}_{P[i,k]} \quad ?$$

- À bras

Cas non biaisé

Aucune amplification possible à moins que $p = \frac{1}{2} + \varepsilon > \frac{1}{2}$

Mais comment calculer un k raisonnable à l'aide de l'horrible

$$\sum_{i=\lceil \frac{k}{2} \rceil}^{k} \underbrace{\binom{k}{i}(\frac{1}{2}+\varepsilon)^i(\frac{1}{2}-\varepsilon)^{k-i}}_{P[i,k]} \quad ?$$

- À bras
  - $P[i, i+s+1] = P[i-1, i+s] \cdot (\frac{1}{2}+\varepsilon) + P[i, i+s] \cdot (\frac{1}{2}-\varepsilon)$
- À pied

Cas non biaisé

Aucune amplification possible à moins que $p = \frac{1}{2} + \varepsilon > \frac{1}{2}$

Mais comment calculer un k raisonnable à l'aide de l'horrible

$$\sum_{i=\lceil \frac{k}{2} \rceil}^{k} \underbrace{\binom{k}{i} (\frac{1}{2} + \varepsilon)^i (\frac{1}{2} - \varepsilon)^{k-i}}_{P[i,k]} \quad ?$$

- À bras
  - $P[i, i+s+1] = P[i-1, i+s] \cdot (\frac{1}{2} + \varepsilon) + P[i, i+s] \cdot (\frac{1}{2} - \varepsilon)$
- À pied
  - par une formule du genre B&B problème 10.25
- À cheval

Cas non biaisé

Aucune amplification possible à moins que $p = \frac{1}{2} + \varepsilon > \frac{1}{2}$

Mais comment calculer un k raisonnable à l'aide de l'horrible

$$\sum_{i=\lceil \frac{k}{2} \rceil}^{k} \underbrace{\binom{k}{i} (\frac{1}{2} + \varepsilon)^i (\frac{1}{2} - \varepsilon)^{k-i}}_{P[i,k]} \quad ?$$

- À bras
  - $P[i, i+s+1] = P[i-1, i+s] \cdot (\frac{1}{2} + \varepsilon) + P[i, i+s] \cdot (\frac{1}{2} - \varepsilon)$
- À pied
  - par une formule du genre B&B problème 10.25
- À cheval
  - par approximation statistique lorsque k est grand ($k \approx 30$)

Cas non biaisé

Exemples : $p = \frac{1}{2} + \varepsilon$

- Pour obtenir Pr[ampli_non_biaisé(x, k) correct] $=$ 95%
  - statistiques $\Longrightarrow k > 2,706\left(\frac{1}{4\varepsilon^2} - 1\right)$ OK

  - $\varepsilon = 5\% \Rightarrow$ prendre $k \approx 270$
  - $\varepsilon = 1\% \Rightarrow$ prendre $k \approx 6750$
  - $\varepsilon = 0,5\% \Rightarrow$ prendre $k \approx 27000$

- Pour obtenir Pr[ampli_non_biaisé(x, k) correct] $=$ 99,5%
  - statistiques $\Longrightarrow k > 6,636\left(\frac{1}{4\varepsilon^2} - 1\right)$ OK
  - pas tellement pire que pour 95%-correct.

Cas non biaisé : morale

- Il faut ($> \frac{1}{2}$)-correct en partant
- Amplification lente
  - de 1%-correct à 95%-correct (biasé) : $k = 300$
  - de 51%-correct à 95%-correct (non biasé) : $k = 6750$
- Attention : ceci pour problèmes à réponses vrai/faux seulement

Algorithmes de Las Vegas

B&B section 10.7

- utilisent l'aléat pour guider leurs choix
- ne se trompent jamais lorsqu'ils répondent

Las Vegas de type I :

- répond toujours
- mauvais choix $\Rightarrow$ temps plus long
  - sélection et médiane
  - quicksort
  - hashage

Las Vegas de type II :

- mauvais choix $\Rightarrow$ l'algo déclare "pas capable"
  - 8 reines
  - factorisation entière

Rappel : sélection du $k$ième élément d'un tableau $T[1..n]$.

- (vu) avec pseudo-médiane comme pivot, temps pire cas $\Theta(n)$
- (pas vu) pivot trivial $\Longrightarrow$ temps pire cas $\Theta(n^2)$
- (pas vu) pivot trivial $\Longrightarrow$ temps moyen $\Theta(n)$, constante cachée petite.

Un algo Las Vegas de type I choisira le pivot au hasard...et alors ?

Sélection et médiane
(suite)

Fait : temps moyen d'avant = temps espéré maintenant

- par la même preuve (pas vue)
- en jouant de malchance, Las Vegas peut prendre autant de temps que le pire cas de l'algo à choix trivial
- peut même prendre ce pire temps sur un exemplaire qui aurait été facile pour l'algo à choix trivial !

Mais alors, l'intérêt ?

Sélection et médiane

(suite)

Fait : temps moyen d'avant = temps espéré maintenant

- par la même preuve (pas vue)
- en jouant de malchance, Las Vegas peut prendre autant de temps que le pire cas de l'algo à choix trivial
- peut même prendre ce pire temps sur un exemplaire qui aurait été facile pour l'algo à choix trivial !

Mais alors, l'intérêt ?

- Il n'y a plus de mauvais exemplaire !
- l'algo prend aux riches et donne aux pauvres.

Las Vegas de type I

Particulièrement utile quand un algo déterministe existe, qui est :

- bon en moyenne
- mauvais en pire cas

Alors un Las Vegas pourra :

- éliminer les exemplaires pire cas
- uniformiser les exemplaires
- maintenir un bon temps espéré

Autre exemple, quicksort (pas vu) :

- $\Theta(n \log n)$ en moyenne
- quadratique en pire cas
- devient temps espéré $\Theta(n \log n)$.

Las Vegas de type II

Rappel : un tel algo peut échouer, mais détecte alors son échec.

$$\texttt{fonction}\ \mathrm{LV}(x, y, \text{succès})$$

- au retour :
  - succès vrai $\Longrightarrow$ y est solution de l'exemplaire x
  - succès faux $\Longrightarrow$ pas de chance
- $p(x) =$ probabilité de succès
- $(\forall \text{ exemplaire } x)[p(x) > 0]$

Las Vegas de type II

Répétition non bornée d'un Las Vegas de type II

```
fonction obstiné(x)
     répéter
          LV(x, y, succès)
     jusqu'à succès
     retourner y
```

- réponse toujours correcte
- toujours obtenue...
- ...un de ces jours !

Las Vegas de type II

Mais quand obstiné(x) s'arrêtera-t-il ?

Soient

- p : probabilité de succès de LV
- s : temps espéré de LV en cas de succès
- e : temps espéré de LV en cas d'échec
- t : temps espéré de obstiné(x)

Alors

$$t = ps + (1-p)(e+t)$$

d'où

$$t = s + \frac{1-p}{p}e$$

À noter : $s\downarrow$ ou $e\downarrow$ ou $p\uparrow \implies t\downarrow$

Fait expérimental : explorer le graphe des vecteurs ($k \leq 8$)-prometteurs par retour arrière examinait 114 sommets sur 2057 avant de trouver

Observation : les positions des reines qui résolvent le problème ont l'air plutôt arbitraires

Suggère un algo de Las Vegas : parmi les positions qui restent,

- choisir les positions successives à remplir au hasard
- abdiquer tout simplement si impasse atteinte

Las Vegas de type II pour les 8 reines

Avantages :

- conceptuellement plus simple que retour arrière

- plus rapide en principe
  - $p = \Pr[\text{succès}] = 0{,}1293 = \frac{\#\text{ solutions}}{\#\text{ total}}$ (ordinateur)
  - s : temps espéré en cas de succès $=$ coût de générer 9 vecteurs
  - e : temps espéré en cas d'échec $= 6{,}971$ vecteurs (ordinateur)
  - temps espéré de l'algo $= t = s + \frac{1-p}{p} e = 55,93$
  - ce 55,93 à comparer aux 114 par retour arrière !

Las Vegas de type II pour les 8 reines

(suite)

En pratique : coût de générer les nombres aléatoires annule le gain en vecteurs générés

Faire mieux ?
Oui, en ajustant s, e et p.

L'idée : générer les k premières reines aléatoirement, et les $8 - k$ dernières par retour arrière :

- $k = 2$: trois fois plus rapide que retour arrière
- $k = 3$: seulement deux fois plus rapide, même si moins de vecteurs

Las Vegas de type II pour les...39 reines
(suite)

L'avantage de Las Vegas sur le retour arrière croît lorsque n augmente

Exemple de $n = 39$:

- par retour arrière : 10^{10} sommets avant la première solution
- pur Las Vegas : un million de fois plus rapide en implantation réelle
- hybride avec $k = 29$: deux millions de fois plus rapide en implantation, 20 millions moins de vecteurs

Exemple de $n = 1000$:

- bonne idée de choisir $k = 983$:-)

Las Vegas de type II

Exemple : factorisation entière

- problème important
- aucun algo efficace connu (la crypto repose sur sa difficulté !)
- ne semble pourtant pas NP-complet
- choix aléatoires + stratégie judicieuse + estimés sophistiqués tirés de la théorie des nombres permettent parfois de réussir !