Probabilités, Statistiques, Combinatoire

Philippe Duchon

Université de Bordeaux – Licence Informatique

2017-2018

Arbres (binaires) aléatoires

- “arbres binaires aléatoires” : on se fixe un entier $n > 0$, et on va choisir une loi de probabilités sur l'ensemble $\mathcal{B}_n$ des arbres binaires (pas forcément pleins) à n noeuds.

Arbres (binaires) aléatoires

- “arbres binaires aléatoires” : on se fixe un entier $n > 0$, et on va choisir une loi de probabilités sur l'ensemble $\mathcal{B}_n$ des arbres binaires (pas forcément pleins) à n noeuds.
- (Il y a C_n tels arbres, le même nombre que les mots de Dyck de longueur $2n$)

Arbres (binaires) aléatoires

- “arbres binaires aléatoires” : on se fixe un entier $n > 0$, et on va choisir une loi de probabilités sur l'ensemble $\mathcal{B}_n$ des arbres binaires (pas forcément pleins) à n noeuds.
- (Il y a C_n tels arbres, le même nombre que les mots de Dyck de longueur $2n$)
- On a deux lois de probabilités classiques sur $\mathcal{B}_n$:

Arbres (binaires) aléatoires

- "arbres binaires aléatoires" : on se fixe un entier $n > 0$, et on va choisir une loi de probabilités sur l'ensemble $\mathcal{B}_n$ des arbres binaires (pas forcément pleins) à n noeuds.
- (Il y a C_n tels arbres, le même nombre que les mots de Dyck de longueur $2n$)
- On a deux lois de probabilités classiques sur $\mathcal{B}_n$:
  - la loi uniforme (chaque arbre possible a probabilité $1/C_n$) : la plus simple à décrire (mais pas celle qui nous intéresse aujourd'hui) ;

Arbres (binaires) aléatoires

- "arbres binaires aléatoires" : on se fixe un entier $n > 0$, et on va choisir une loi de probabilités sur l'ensemble $\mathcal{B}_n$ des arbres binaires (pas forcément pleins) à n noeuds.
- (Il y a C_n tels arbres, le même nombre que les mots de Dyck de longueur $2n$)
- On a deux lois de probabilités classiques sur $\mathcal{B}_n$:
  - la loi uniforme (chaque arbre possible a probabilité $1/C_n$) : la plus simple à décrire (mais pas celle qui nous intéresse aujourd'hui) ;
  - une autre loi "naturelle" : celle des arbres binaires qu'on obtient en insérant, dans un ordre aléatoire uniforme, n nombres distincts dans un arbre binaire de recherche vide ;

Arbres (binaires) aléatoires

- "arbres binaires aléatoires" : on se fixe un entier $n > 0$, et on va choisir une loi de probabilités sur l'ensemble $\mathcal{B}_n$ des arbres binaires (pas forcément pleins) à n noeuds.
- (Il y a C_n tels arbres, le même nombre que les mots de Dyck de longueur $2n$)
- On a deux lois de probabilités classiques sur $\mathcal{B}_n$:
  - la loi uniforme (chaque arbre possible a probabilité $1/C_n$) : la plus simple à décrire (mais pas celle qui nous intéresse aujourd'hui) ;
  - une autre loi "naturelle" : celle des arbres binaires qu'on obtient en insérant, dans un ordre aléatoire uniforme, n nombres distincts dans un arbre binaire de recherche vide ;
- On va plutôt s'intéresser à cette dernière loi, qui apparaît naturelmenent en algorithmique ; on l'appelle "loi des arbres binaires de recherche aléatoires" (de taille n).

ABR aléatoires

- Une façon d'obtenir un "ABR aléatoire" : prendre une liste comprenant $1, 2, \ldots, n$; "mélanger" (parfaitement) cette liste ; et insérer les valeurs dans un ABR vide, dans l'ordre obtenu.

ABR aléatoires

- Une façon d'obtenir un "ABR aléatoire" : prendre une liste comprenant $1, 2, \ldots, n$; "mélanger" (parfaitement) cette liste ; et insérer les valeurs dans un ABR vide, dans l'ordre obtenu.
- **Première question :** pour un arbre binaire donné, quelle est la probabilité d'obtenir cet arbre ?

ABR aléatoires

- Une façon d'obtenir un "ABR aléatoire" : prendre une liste comprenant $1, 2, \ldots, n$; "mélanger" (parfaitement) cette liste ; et insérer les valeurs dans un ABR vide, dans l'ordre obtenu.
- **Première question :** pour un arbre binaire donné, quelle est la probabilité d'obtenir cet arbre ?
- **Deuxième question :** à quoi ça ressemble, un tel arbre binaire aléatoire ? (pour n grand, disons)

ABR aléatoires

- Une façon d'obtenir un "ABR aléatoire" : prendre une liste comprenant $1, 2, \ldots, n$; "mélanger" (parfaitement) cette liste ; et insérer les valeurs dans un ABR vide, dans l'ordre obtenu.
- **Première question :** pour un arbre binaire donné, quelle est la probabilité d'obtenir cet arbre ?
- **Deuxième question :** à quoi ça ressemble, un tel arbre binaire aléatoire ? (pour n grand, disons)
- **Troisième question :** et concrètement, ça sert à quoi ?

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.
- Peut-on calculer $\text{ins}(t)$? (donner une formule ?)

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.
- Peut-on calculer $\text{ins}(t)$? (donner une formule ?) **Oui !**

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.
- Peut-on calculer $\text{ins}(t)$? (donner une formule ?) **Oui !**
- Supposons que t a un sous-arbre gauche t_1, de taille k, et sous-arbre droit t_2, de taille $n-k-1$ ($0 \leq k \leq n-1$; donc la racine contient $k+1$)

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.
- Peut-on calculer $\text{ins}(t)$? (donner une formule ?) **Oui !**
- Supposons que t a un sous-arbre gauche t_1, de taille k, et sous-arbre droit t_2, de taille $n-k-1$ ($0 \leq k \leq n-1$; donc la racine contient $k+1$)
- **Proposition :** $\text{ins}(t) = \binom{n-1}{k-1}\text{ins}(t_1)\text{ins}(t_2)$.

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.
- Peut-on calculer $\text{ins}(t)$? (donner une formule ?) **Oui !**
- Supposons que t a un sous-arbre gauche t_1, de taille k, et sous-arbre droit t_2, de taille $n-k-1$ ($0 \leq k \leq n-1$; donc la racine contient $k+1$)
- **Proposition :** $\text{ins}(t) = \binom{n-1}{k-1}\text{ins}(t_1)\text{ins}(t_2)$.
- **Corollaire :** $p(t) = \frac{1}{n}p(t_1)p(t_2)$.

Probabilité d'un arbre binaire

- t un squelette d'arbre binaire à n noeuds
- Dans une liste de longueur n "bien mélangée", les $n!$ ordres possibles sont équiprobables (proba. $1/n!$ chacun).
- Donc la probabilité d'obtenir t comme arbre aléatoire, c'est $p(t) = \text{ins}(t)/n!$, où $\text{ins}(t)$ est *le nombre d'ordres d'insertion qui donnent t comme arbre*.
- Peut-on calculer $\text{ins}(t)$? (donner une formule ?) **Oui !**
- Supposons que t a un sous-arbre gauche t_1, de taille k, et sous-arbre droit t_2, de taille $n-k-1$ ($0 \leq k \leq n-1$; donc la racine contient $k+1$)
- **Proposition :** $\text{ins}(t) = \binom{n-1}{k-1}\text{ins}(t_1)\text{ins}(t_2)$.
- **Corollaire :** $p(t) = \frac{1}{n}p(t_1)p(t_2)$.
- **Autre formule :** $p(t) = \prod_s \frac{1}{n(s)}$, où s parcourt l'ensemble des noeuds de t, et où $n(s)$ désigne la taille (nombre de noeuds) de l'arbre dont s est la racine.

Apparence générale d'un arbre

Pour un arbre de taille n, "aléatoire" :

- La taille K du sous-arbre gauche est uniforme sur $\{0, 1, \ldots, n-1\}$

Apparence générale d'un arbre

Pour un arbre de taille n, "aléatoire" :

- La taille K du sous-arbre gauche est uniforme sur $\{0, 1, \ldots, n-1\}$
- La taille du sous-arbre gauche est $n-1-K$, elle aussi uniforme sur $\{0, 1, \ldots, n-1\}$

Apparence générale d'un arbre

Pour un arbre de taille n, "aléatoire" :

- La taille K du sous-arbre gauche est uniforme sur $\{0, 1, \ldots, n-1\}$
- La taille du sous-arbre gauche est $n-1-K$, elle aussi uniforme sur $\{0, 1, \ldots, n-1\}$
- Conditionnellement à $\{K = k\}$, les sous-arbres gauche et droit sont des arbres binaires "aléatoires" de leurs tailles respectives, et indépendants.

Apparence générale d'un arbre

Pour un arbre de taille n, "aléatoire" :

- La taille K du sous-arbre gauche est uniforme sur $\{0, 1, \ldots, n-1\}$
- La taille du sous-arbre gauche est $n-1-K$, elle aussi uniforme sur $\{0, 1, \ldots, n-1\}$
- Conditionnellement à $\{K = k\}$, les sous-arbres gauche et droit sont des arbres binaires "aléatoires" de leurs tailles respectives, et indépendants.
- Également (admis) : **en espérance, la hauteur d'un arbre binaire aléatoire est d'ordre** $O(\log(n))$

Apparence générale d'un arbre

Pour un arbre de taille n, "aléatoire" :

- La taille K du sous-arbre gauche est uniforme sur $\{0, 1, \ldots, n-1\}$
- La taille du sous-arbre gauche est $n-1-K$, elle aussi uniforme sur $\{0, 1, \ldots, n-1\}$
- Conditionnellement à $\{K = k\}$, les sous-arbres gauche et droit sont des arbres binaires "aléatoires" de leurs tailles respectives, et indépendants.
- Également (admis) : **en espérance, la hauteur d'un arbre binaire aléatoire est d'ordre** $O(\log(n))$
- (C'est intéressant : ça montre que pour les ABR, "le hasard fait bien les choses" – les algorithmes dont la complexité est gouvernée par la hauteur de l'arbre, s'exécutent rapidement en moyenne sur de tels arbres)

Apparence générale d'un arbre

Pour un arbre de taille n, "aléatoire" :

- La taille K du sous-arbre gauche est uniforme sur $\{0, 1, \ldots, n-1\}$
- La taille du sous-arbre gauche est $n-1-K$, elle aussi uniforme sur $\{0, 1, \ldots, n-1\}$
- Conditionnellement à $\{K = k\}$, les sous-arbres gauche et droit sont des arbres binaires "aléatoires" de leurs tailles respectives, et indépendants.
- Également (admis) : **en espérance, la hauteur d'un arbre binaire aléatoire est d'ordre** $O(\log(n))$
- (C'est intéressant : ça montre que pour les ABR, "le hasard fait bien les choses" – les algorithmes dont la complexité est gouvernée par la hauteur de l'arbre, s'exécutent rapidement en moyenne sur de tels arbres)
- **Mais** on ne peut pas trop compter sur le fait que les insertions se fassent dans un ordre aléatoire !

Arbres binaires de recherche “randomisés”

(Martinez, Roura, 1999)

- **Scoop :** on peut réaliser les opérations (insertion/suppression) dans les arbres binaires de recherche, **blue**de manière probabiliste, de telle sorte que, **après n'importe quelle séquence d'insertions et de suppressions dans l'arbre**, l'arbre obtenu soit un “arbre aléatoire” (ne dépendant pas de la séquence exacte d'insertions/suppressions : pour toute séquence d'opérations, la probabilité d'obtenir l'arbre t est $p(t)$).

Arbres binaires de recherche “randomisés”

(Martinez, Roura, 1999)

- **Scoop :** on peut réaliser les opérations (insertion/suppression) dans les arbres binaires de recherche, **blue**de manière probabiliste, de telle sorte que, **après n'importe quelle séquence d'insertions et de suppressions dans l'arbre**, l'arbre obtenu soit un “arbre aléatoire” (ne dépendant pas de la séquence exacte d'insertions/suppressions : pour toute séquence d'opérations, la probabilité d'obtenir l'arbre t est $p(t)$).
- Les algorithmes d'insertion et de suppression deviennent “randomisés” (comportement dépend de tirages aléatoires)

Arbres binaires de recherche “randomisés”

(Martinez, Roura, 1999)

- **Scoop :** on peut réaliser les opérations (insertion/suppression) dans les arbres binaires de recherche, **blue**de manière probabiliste, de telle sorte que, **après n'importe quelle séquence d'insertions et de suppressions dans l'arbre**, l'arbre obtenu soit un “arbre aléatoire” (ne dépendant pas de la séquence exacte d'insertions/suppressions : pour toute séquence d'opérations, la probabilité d'obtenir l'arbre t est $p(t)$).
- Les algorithmes d'insertion et de suppression deviennent “randomisés” (comportement dépend de tirages aléatoires)
- **Insertion :** dans l'algorithme classique, à chaque étape récursive on peut décider d'insérer la valeur à la racine courante, en “splittant” l'arbre

Arbres binaires de recherche “randomisés”

(Martinez, Roura, 1999)

- **Scoop :** on peut réaliser les opérations (insertion/suppression) dans les arbres binaires de recherche, **blue**de manière probabiliste, de telle sorte que, **après n'importe quelle séquence d'insertions et de suppressions dans l'arbre**, l'arbre obtenu soit un “arbre aléatoire” (ne dépendant pas de la séquence exacte d'insertions/suppressions : pour toute séquence d'opérations, la probabilité d'obtenir l'arbre t est $p(t)$).
- Les algorithmes d'insertion et de suppression deviennent “randomisés” (comportement dépend de tirages aléatoires)
- **Insertion :** dans l'algorithme classique, à chaque étape récursive on peut décider d'insérer la valeur à la racine courante, en “splittant” l'arbre
- **Suppression :** une fois trouvée la clé à supprimer, on “recolle” aléatoirement les deux sous-arbres gauche et droit

Arbres binaires de recherche “randomisés”

(Martinez, Roura, 1999)

- **Scoop :** on peut réaliser les opérations (insertion/suppression) dans les arbres binaires de recherche, **blue**de manière probabiliste, de telle sorte que, **après n'importe quelle séquence d'insertions et de suppressions dans l'arbre**, l'arbre obtenu soit un “arbre aléatoire” (ne dépendant pas de la séquence exacte d'insertions/suppressions : pour toute séquence d'opérations, la probabilité d'obtenir l'arbre t est $p(t)$).
- Les algorithmes d'insertion et de suppression deviennent “randomisés” (comportement dépend de tirages aléatoires)
- **Insertion :** dans l'algorithme classique, à chaque étape récursive on peut décider d'insérer la valeur à la racine courante, en “splittant” l'arbre
- **Suppression :** une fois trouvée la clé à supprimer, on “recolle” aléatoirement les deux sous-arbres gauche et droit
- La preuve que les algorithmes sont “corrects” est complexe ; les algorithmes correspondants sont très simples.