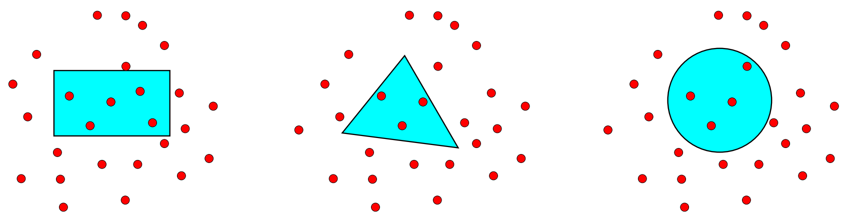
# Geometric Range Searching



1. $S \subset \mathbb{R}^d$ ($n$ points) et $\mathcal{R} \subset \mathcal{P}(\mathbb{R}^d)$ (*domaines*– ranges).
2. Problème :

**Compter (reporter) les points de $S$**
**situés dans le domaine $R$.**





# Recherche Multidimensionnelle

1. $S \subset \mathbb{R}^d$ ($n$ points) et $\mathcal{R} \subset \mathcal{P}(\mathbb{R}^d)$ (*domaines*– ranges).
2. Le problème est celui d'organiser $S$ en une structure de données de telle sorte à pouvoir répondre efficacement à une requête du type:

   **Compter (reporter) les points de $S$ situés dans le domaine $R$.**

3. En général la solution consiste à définir une famille de sous-ensembles "canoniques" de $S$ (phase de prétraitement), puis à chercher dans cette famille une partition de l'ensemble des points à reporter $R \cap S$, dont on extrait l'information requise (phase de réponse à la requête).
4. Les ensembles canoniques sont en général organisés en une structure d'arbre binaire (à plusieurs niveaux) dans laquelle on effectue une recherche guidée par des informations auxilliaires.





# Notes





# Notes

# Arbres binaires

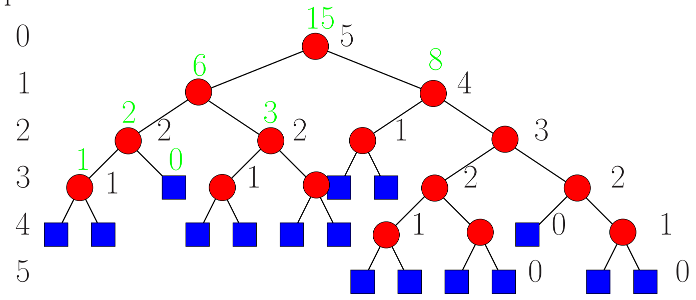
– Définitions – taille, hauteur, profondeur
– Nombre d'arbres binaires
– Génération aléatoire





# Arbres Binaires

profondeur




$$\mathcal{B} = \Box + (\circ, \mathcal{B}, \mathcal{B})$$

**type** *tree* $=\uparrow$ *node*;
*node* = **record**
*k* : *typekey*;
*left*, *right* : *tree*
**end;**





# Taille, Hauteur, Profondeur

$$\begin{gathered}
\text{taille}, \text{hauteur}, \text{lci}, \text{lce}(\Box) = 0 \\
\text{taille}(B) = 1 + \text{taille}(\text{sag}(B)) + \text{taille}(\text{sad}(B)) \\
\text{hauteur}(B) = 1 + \max\{\text{hauteur}(\text{sag}(B)), \text{hauteur}(\text{sad}(B))\} \\
\text{profondeur}(\text{racine}) = 0 \\
\text{profondeur}((\mathit{noeud})) = 1 + \text{profondeur}(\text{pere}(\mathit{noeud}))
\end{gathered}$$

$$\begin{gathered}
\text{lci}(B) = \text{taille}(B) - 1 + \text{lci}(\text{sag}(B)) + \text{lci}(\text{sad}(B)) \\
\text{lce}(B) = \text{taille}(B) + 1 + \text{lce}(\text{sag}(B)) + \text{lce}(\text{sad}(B)) \\
\text{lce}(B) = \text{lci}(B) + 2\text{taille}(B)
\end{gathered}$$

Profondeur moyenne d'un noeud interne = lci/taille
Profondeur moyenne d'un noeud externe = lce/taille





# Nombre d'Arbres Binaires

Le nombre $C_n$ d'arbres binaires de taille $n$ est le n-ième nombre de Catalan

$$C_n = \sum_{i+j=n-1} C_i C_j = \frac{1}{n+1}\binom{2n}{n} \sim \frac{1}{\sqrt{\pi}} \frac{4^n}{n^{3/2}}.$$

$$\begin{aligned}
\sum_n C_n x^n &= \frac{1}{2x}(1 - \sqrt{1-4x}) \\
&= 1 + x + 2x^2 + 5x^3 + 14x^4 + 42x^5 + 132x^6 \\
&\quad + 429x^7 + 1430x^8 + 4862x^9 + \ldots
\end{aligned}$$

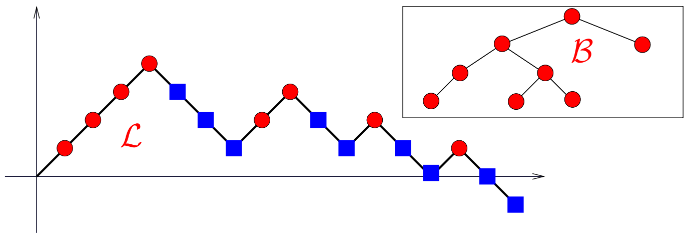
Nombre d'arbres binaires de taille 10, 50, 100:

16796

1978261657756160653623774456

896519947090131496687170070074100632420837521538745909320

# Parcours Préfixe




Le langage de Lukasiewicz $\mathcal{L}$ est le langage engendré par la grammaire

$$S \to aSS + b$$

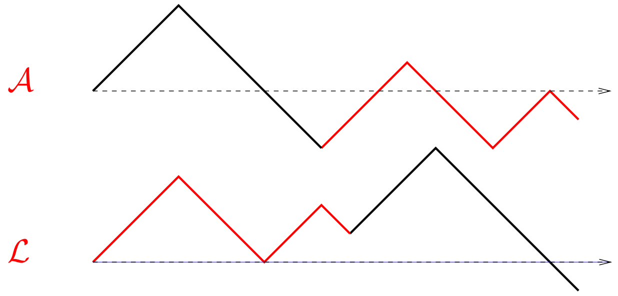
C'est aussi l'ensemble des mots $m \in \{a,b\}^*$ tels que
(1) $|m|_b = 1 + |m|_a$
(2) pour tout préfixe propre $u$ de $m$ on a $|u|_b \leq |u|_a$.

$$\mathcal{B}_n \overset{1-1}{\longleftrightarrow} \mathcal{L}_{2n+1}$$





# Conjugaison



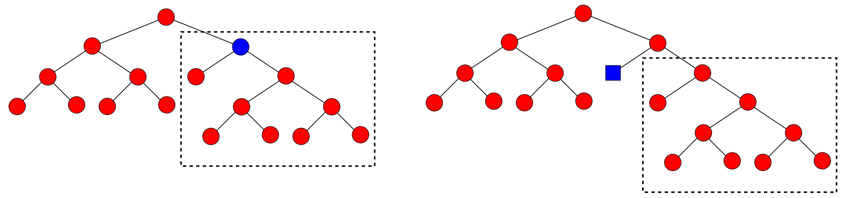

Pour tout mot $w \in \{a,b\}^*$ tel que $|w|_b = 1 + |w|_a$, il existe un et un seul conjugué $u$ de $w$ appartenant à $\mathcal{L}$.

$$\mathcal{A}_{2n+1} \overset{(2n+1)-1}{\longleftrightarrow} \mathcal{L}_{2n+1} \qquad \#\mathcal{L}_{2n+1} = \frac{1}{n+1}\binom{2n}{n}$$





# Génération Aléatoire



- Correspondance 1-2 entre les arbres binaires de taille $n$ dont un noeud est distingué et les arbres binaires de taille $n+1$ dont une feuille est distinguée.

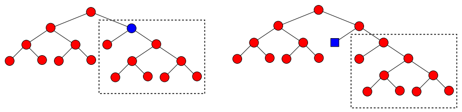
$$2(2n+1)C_n = (n+2)C_{n+1}.$$

- Profondeur moyenne $\sim \sqrt{\pi n}$ – Hauteur moyenne $\sim 2\sqrt{\pi n}$
- G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures: In Pascal and C.* Addison-Wesley, 1991.





# Profondeur Moyenne d'un Noeud Interne



Profondeur moyenne d'un noeud interne :

$$p(n) = \frac{4^n - \frac{3n+1}{n+1}\binom{2n}{n}}{\frac{n}{n+1}\binom{2n}{n}} = \sqrt{\pi n}\left(1 + \frac{9}{8n} + \frac{17}{128n^2} + O(n^{-3})\right) - 3 - \frac{1}{n}$$

| $n$ | 10 | 20 | 50 | 100 | 200 | 500 | $\infty$ |
|---|---|---|---|---|---|---|---|
| $p(n)/\sqrt{n}$ | 0.99 | 1.19 | 1.38 | 1.49 | 1.57 | 1.64 | 1.77 |

## Geometric Range Searching in Dimension One

– THE FUNDAMENTAL DATA STRUCTURING PROBLEM:

– $\mathcal{K}$ un ensemble fini de clés d'un univers totalement ordonné. Recherche/Insertion/Suppression.

– Arbres binaires de recherche aléatoire

– Skip lists

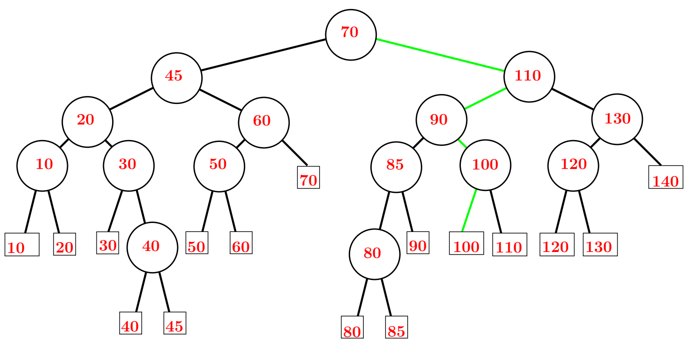
– Hachage

– Autres méthodes (arbres équilibrés, splay trees)





## Arbres Binaires de Recherche




Soit $S$ un ensemble fini de $n$ clés d'un univers totalement ordonné. Un arbre binaire de recherche sur $S$ est un arbre binaire de taille $n-1$ dont les noeuds externes et internes sont étiquetés par les clés de $S$ de telle sorte que (1) la clé d'un noeud interne coincide avec la clé de la feuille la plus à droite de son sous-arbre gauche et (2) un parcours infixe de l'arbre énumére les clés des noeuds internes (externes) dans l'ordre croissant.





## Chemin de Recherche d'une clé

**procedure** *search*(*key* : *typekey*; **var** *t* : *tree*);

**begin**

**if** $t =$ **nil then** *notfound*(*key*)

**else if** $t\uparrow.k = key$ **then** *found*($t\uparrow$)

**else if** $t\uparrow.k < key$ **then** *search*($key, t\uparrow.right$)

  **else** *search*($key, t\uparrow.left$)

**end;**

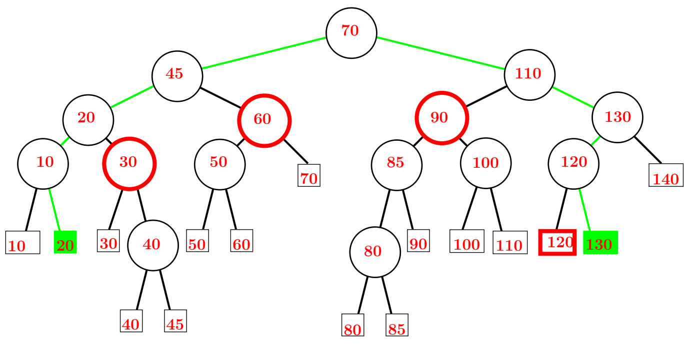
$S[x, x'] = \{k \in S \mid x \leq k \leq x'\}$

$$S[x, x'] = \bigoplus_{\nu} S(\nu)$$

où $\nu$ décrit l'ensemble des racines des sous-arbres pendant respectivement à gauche et à droite des chemins de recherche de $x$ et $x'$ à partir de leur premier ancêtre commun.








$$\begin{aligned}\{K \mid 20 < K < 130\} &= S(30) \cup S(60) \cup S(90) \cup S_{\square}(120)\\ &= \{30, 40, 45\} \cup \{50, 60, 70\} \cup \{80, 85, 90, 100, 110\} \cup \{120\}\end{aligned}$$

# Insertion

```
procedure insert(key : typekey; var t : tree);
begin
if t = nil then t := NewNode(key, nil, nil)
else if t ↑ .k = key then Error
else if t ↑ .k < key then insert(key, t ↑ .right)
   else insert(key, t ↑ .left)
end;
```





# Profondeur d'un ABR Aléatoire

- Statistisque sur le multi-ensemble des $n!$ arbres binaires obtenus en appliquant la procédure d'insertion (étendue) aux $n!$ permutations des entiers de 1 à $n$.
- $E(\text{profondeur}) \sim 1.3863 \log_2 n$ – $E(\text{hauteur}) \sim 4.31107 \log_2 n$.

Fixons une clé requête $K$ et soit $X_j$ la variable aléatoire égale 1 si $K$ est comparée avec la clé, notée $K_j$, insérée dans l'arbre à l'instant $j$, et 0 sinon. La longueur du chemin de recherche de $K$ est $X = \sum_i X_i$.

- $E(X_j \mid \{K_1, \ldots, K_j\} \text{ fixé}) = O(2/j)$
- $E(X_j) = O(2/j)$
- $X = \tilde{O}(\log n)$ Chernoff bound for harmonic random variables
- Borne indépendante de la clé requête.





# Technique de Chernoff

❑ — On dit que $f(n) = \tilde{O}(g(n))$ si $f(n) < cg(n)$ avec probabilité $1 - 1/p(n)$, où $p(n)$ est un polynôme dont le degré dépend de $c$, et ce degré peut être rendu aussi grand que voulu en choisissant la constant $c$ assez grande. Ainsi, $f(n) > cg(n)$ avec probabilité $1/p(n)$ qui est très petit.

❑ — Technique de Chernoff.

$$\begin{aligned} \text{Prob}[X \geq x] &= \text{Prob}[e^{\lambda X} \geq e^{\lambda x}] && (\text{pour } \lambda > 0) \\ &\leq e^{-\lambda x} E[e^{\lambda X}] && (\text{inégalité de Markov}) \end{aligned}$$

[Rappel : $\text{Prob}[X \geq \alpha] \leq \frac{E(X)}{\alpha}$ pour $\alpha > 0$ ]

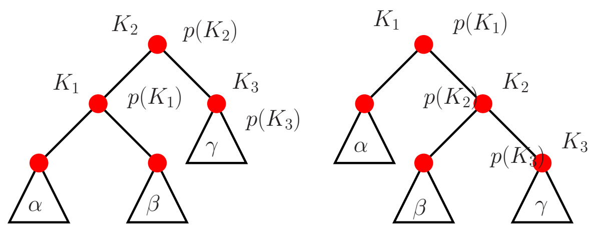
$$\begin{aligned} &\leq e^{-\lambda x} \prod_{i=1}^{n} E[e^{\lambda X_i}] && (\text{indépendance des } X_i) \\ &\leq e^{-\lambda x} e^{H_n(e^{\lambda}-1)} \end{aligned}$$

car $E[e^{\lambda X_i}] = (1 - 1/i) + 1/ie^{\lambda} = 1 + 1/i(e^{\lambda} - 1) \leq \exp[1/i(e^{\lambda} - 1))]$. Il reste à prendre $x = cH_n$ et $\lambda = \ln c$ pour obtenir $\text{Prob}[X \geq cH_n] \leq e^{-H_n(1+c\ln(c/e))}$.





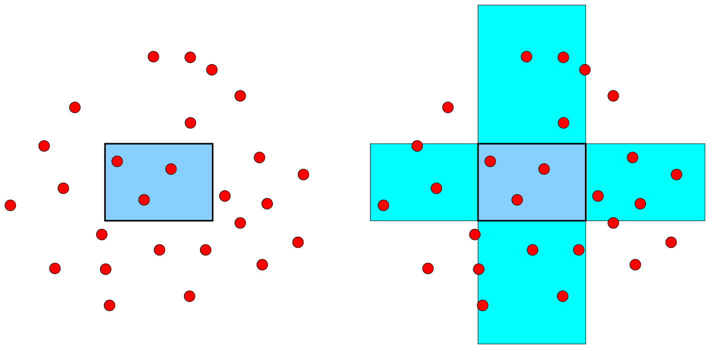
# Treaps




A chaque clé $K$ est associé dynamiquement une priorité $p(K) \in [0, 1]$. L'a.b.r. maintenu est l'arbre binaire associé à la permutation des clés induite par leurs priorités. La diminution/augmentation d'une priorité se traduit par une séquence de rotations dans l'arbre de recherche.

- Recherche/Insertion/Suppression en temps $\tilde{O}(\log n)$
- Insertion/Suppression en $\tilde{O}(2)$ rotations
- C. Aragon and R. Seidel. Randomized Search Trees. In *Algorithmica* 16, pages 464–497, 1996 (FOCS'89).

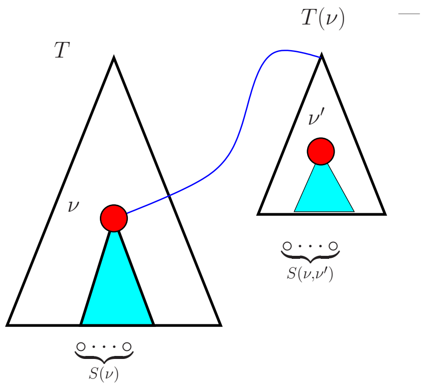
# Orthogonal range Searching



1. $S$ un ensemble de $n$ points du plan
2. $R = [a_1, a_2] \times [b_1, b_2]$ un domaine requête. $S \cap R = ??$
3. $R_1 = [a_1, a_2] \times \mathbb{R}$ et $R_2 = \mathbb{R} \times [b_1, b_2]$.
4. $S \cap R = (S \cap R_1) \cap R_2$







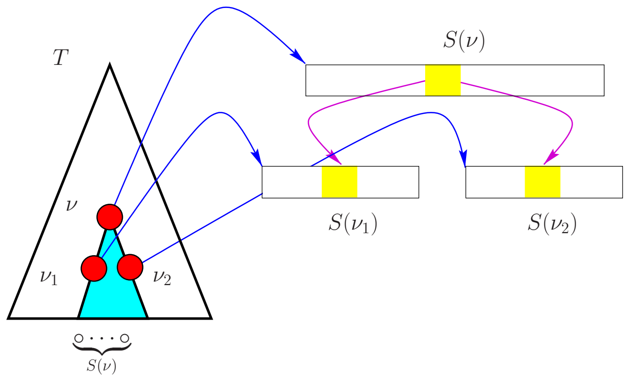

$$
\begin{aligned}
\sum_{\nu \in T} \#T(\nu) &= \sum_{\nu \in T} \#S(\nu) \\
&= \sum_{p \in S} \#\mathrm{CheRech}(T, p) \\
&= O(n \log n).
\end{aligned}
$$

$$(S \cap R_1) \cap R_2 = \bigoplus_{\nu} S(\nu) \cap R_2 = \bigoplus_{\nu} \bigoplus_{\nu'} S(\nu, \nu')$$

où $\nu$ décrit CheRech$(T, [a_1, a_2])$ et $\nu'$ décrit CheRech$(T(\nu), [b_1, b_2])$.







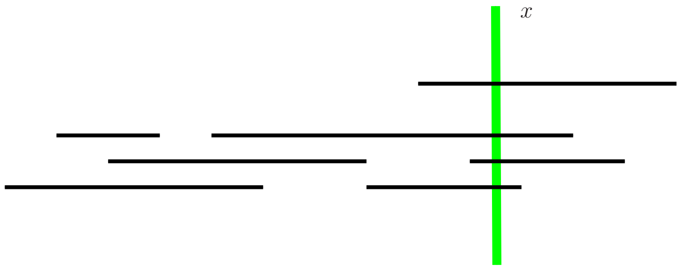





# Fractionnement en Cascade

Soit $T$ un arbre binaire à $n$ noeuds et pour chaque noeud $\nu$ de $T$ une liste totalement ordonnée $L_\nu$. La technique du "fractionnement en cascades" (fractionnal cascading) permet de calculer la position d'une clé dans $L_\nu$ en temps constant si l'on connait la position de la clé dans la liste associée au parent du noeud $\nu$. Le temps nécessaire à la construction de la structure est un $O(n + \sum_{\nu \in T} |L_\nu|)$.

# Arbre de Segments (Bentley 1977)

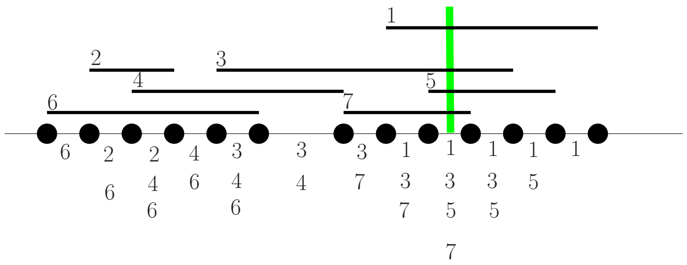



– $\{I_1, \ldots, I_n\}$

– $S(x) = \{I_i \mid x \in I_i\}$





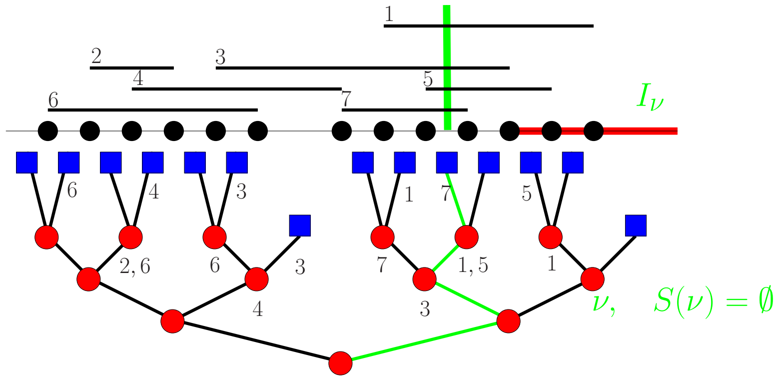







– $S(\nu) = \{\, I_j \supseteq I_\nu \mid I_{\text{parent}(\nu)} \not\subseteq I_j \,\}$.

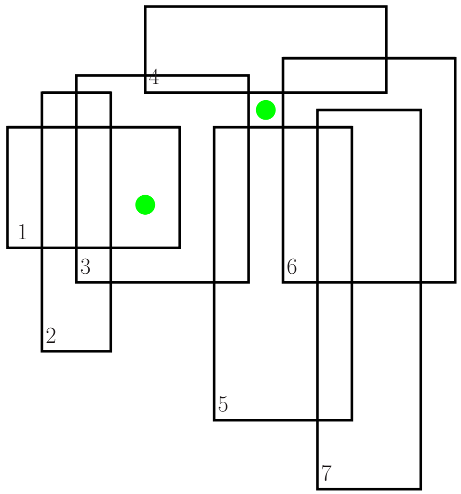
– $\sum_\nu \#S(\nu) = O(n \log n)$




$$S(x) = \biguplus_{\nu \in \mathrm{CheRech}(x)} S(\nu)$$



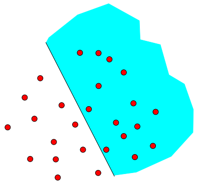





– $\{R_1, \ldots, R_n\}$

– $S(x) = \{R_i \mid x \in R_i\}$

## Recherche dans un Demi-Plan

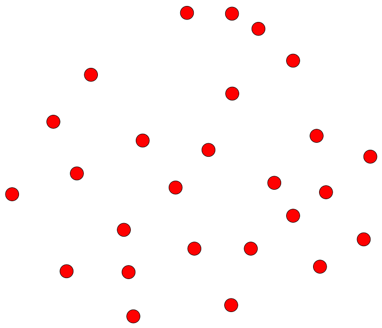


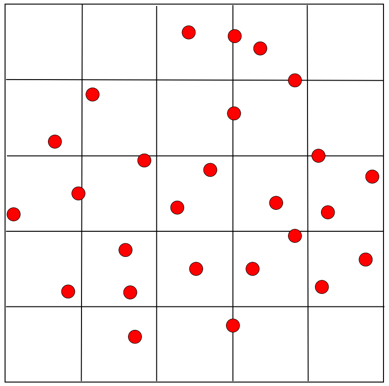
- Méthode de grille
- via les 4-partitions
- partitions simpliciales





## Méthode de Grille









## Méthode de Grille (Suite)

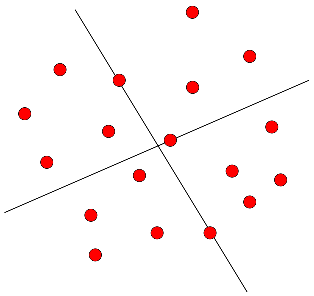
- V. Gaede and O. Günther. Multidimensinal Acces Methods. ACM Computing Surveys. pages 170–231, 30(2),1998
- Samet and co

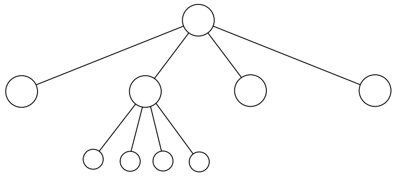




## Notes

# EQUIPARTITIONS





– Via les équipartitions (définition, construction)

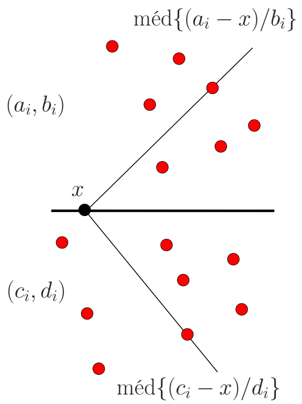
– Temps de requête: $T(n) = O(1) + 3T(n/4) = O(n^{\log_4 3})$.

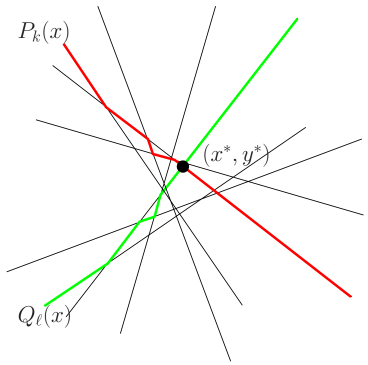
– Nimrod Meggido. Partitionning with Two Lines in the Plane. *J. Algorithms*. 6: 430–433, 1985.





# EQUIPARTITIONS (EXISTENCE)



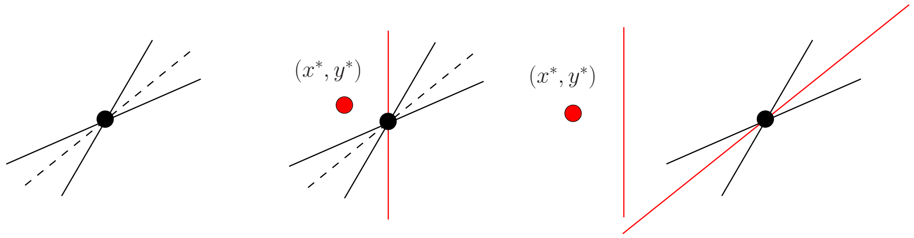

– Soit $P_k(x)$ ($Q_\ell(x)$) la $k$-ième ($\ell$-ième) valeur des $\{A_i x + B_i\}$ avec $A_i > 0$ ($\{C_i x + D_i\}$ avec $C_i < 0$)

– Requête de position : la position de $(x^*, y^*)$ par rapport à n'importe quelle droite est décidable en temps $O(n)$.





# EQUIPARTITIONS (CONSTRUCTION)




– Droites actives

– Itérer: pente médiane, couplage, abscisse médian, requête, intercept médian, requête, élimination.

– Chaque itération désactive 1/8 des droites





# NOTES

❏ — Est-il possible de partitionner une distribution de points de $\mathbb{R}^d$ en $2^d$ parts égales?

❏ — Oui en dimension 3 , non en dimension $\geq 5$, ouvert en dimension 4.

# Préliminaires mathématiques

- Permutations, Combinaisons/Echantillons
- Arbres binaires





# Permutations

Permutation, groupe $S_n$ des permutations, facteurs, décomposition en produit de facteurs, permutation circulaire, transposition.

$$\begin{aligned} \phi &= [7531246] = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 7 & 5 & 3 & 1 & 2 & 4 & 6 \end{pmatrix} \\ &= (1764)(25)(3) \end{aligned}$$

Nombre de permutations, formule de Stirling

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$





# Permutations et transpositions

Toute permutation de $S_n$ s'écrit de manière unique comme un produit de la forme

$$(na_n)\cdots(3a_3)(2a_2)(1a_1)$$

où $a_i \in \{1, 2, \ldots, i\}$.

| | |
|---|---|
| (7 6) | [7 5 3 1 2 4 6] |
| (6 4) | [6 5 3 1 2 4 7] |
| (5 2) | [4 5 3 1 2 6 7] |
| (4 1) | [4 2 3 1 5 6 7] |
| (3 3) | [1 2 3 4 5 6 7] |
| (2 2) | [1 2 3 4 5 6 7] |
| (1 1) | [1 2 3 4 5 6 7] |
| | [1 2 3 4 5 6 7] |

$$[7531246] = (76)(64)(52)(41)(33)(22)(11)$$





# Génération Aléatoire d'une Permutation

**Algorithm** Random-Shuffling ($a[1..n]$);

```
for i := 1 to n do
  begin
    j := random-number(1, i);
    swap(a[i], a[j]);
  end
```

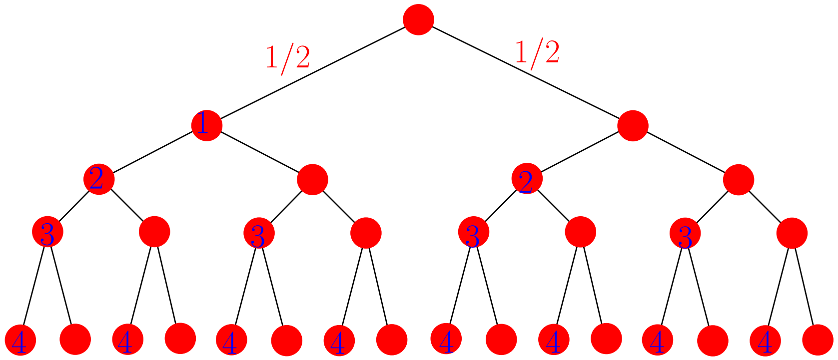
# Combinaisons/Echantillons

1. $\binom{E}{p}$ = l'ensemble des parties de $E$ à $p$ éléments
2. $\binom{n}{p}$ = # parties à $p$ éléments d'un ensemble à $n$ éléments
3. $\binom{n}{0} = \binom{n}{n} = 1$
$$\binom{n}{p} = \binom{n-p}{p}$$
4. $$\binom{n}{p+1}\binom{p+1}{1} = \binom{n}{p}\binom{n-p}{1}$$
5. $$\binom{n}{p}\binom{p}{q} = \binom{n}{p-q}\binom{n-p+q}{q}$$





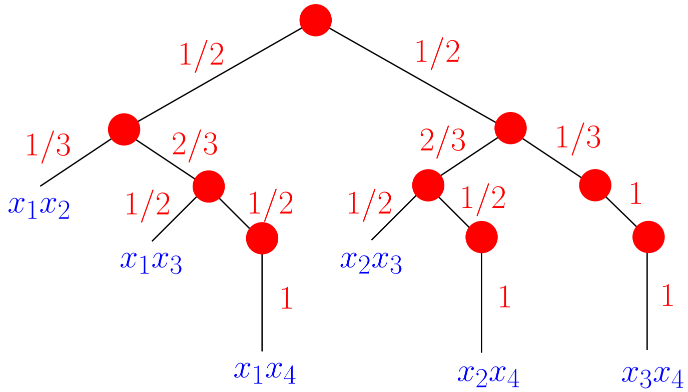
# Génération aléatoire d'un échantillon d'un ensemble à $n$ éléments








# Génération aléatoire d'un échantillon à 2 éléments d'un ensemble à 4 éléments







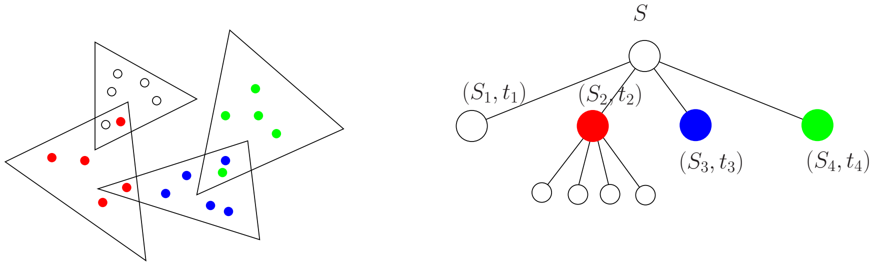

# Génération aléatoire d'un échantillon à $r$ éléments d'un ensemble à $n$ éléments

Un échantillon aléatoire de taille $r$ d'une suite de $n$ éléments est obtenu en tirant le $(k+1)$-ième élément avec probabilité

$$\frac{\binom{n-k-1}{r-r_c-1}}{\binom{n-k}{r-r_c}} = \frac{r-r_c}{n-k}$$

où $r_c$ est la taille de l'échantillon obtenu après tirage des $k$ premiers éléments.

# Partition Simpliciale – Arbre de Partition




1. $\Psi(S) := \{(S_1, t_1), (S_2, t_2), \ldots, (S_r, t_r)\}$ où les $S_i$ forment une partition de $S$ et où $t_i$ est un triangle contenant $S_i$.
2. Une partition simpliciale est *bonne* si $|S_i| \leq 2n/r$.
3. Le *module* de $\Psi(S)$ est le nombre maximal de triangles sécants à une droite.





# Existence & Construction

- Pour tout ensemble de $n$ points du plan, et tout paramètre $1 \leq r \leq n$, il existe une bonne partition simpliciale de module $\sqrt{r}$. De plus une telle partition est constructible en temps $O(n \log r)$.
- Chazelle, Matousek, Welzl, etc
- S. Har-Peled. Constructing planar cuttings in theory and practice. In *SIAM J. Comput.*, 29: 2016–203, 2000.





# Recherche Simpliciale

$$\begin{aligned} T_{\text{ime}}(n) &= O(r) + O(\sqrt{r})T(\frac{2n}{r}) \qquad r = \sqrt{n} \\ &= O\left(\sqrt{n}\, 2^{O(\log\log n)}\right) \\ &= O\left(\sqrt{n}\, \text{polylog } n\right) \end{aligned}$$

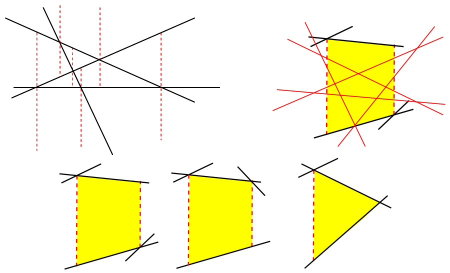
$$\begin{aligned} S_{\text{pace}}(n) &= O(r) + O(r)S(\frac{2n}{r}) \qquad r = \sqrt{n} \\ &= O(n) \end{aligned}$$





# Notes

# Clarkson-Shor technique



❑ — Soit $X$ un ensemble fini de $n$ droites en position générique et soit $R$ un échantillon aléatoire de $X$ de taille $r$ tiré selon une loi uniforme. Que peux-t-on dire du nombre $l(\sigma)$ de droites de $X$ sécantes à une cellule $\sigma$ de l'arrangement $A(R)$?

❑ — $A^{\|}(R)$





❑ — La valeur moyenne sur l'ensemble des trapèzes de $A^{\|}(R)$ du nombre de droites de $X$ sécantes à un trapèze de $A^{\|}(R)$ est $\ll n/r$, soit

$$E\left[\sum_{\sigma\in A^{\|}(R)} l(\sigma)\right] \ll nr. \tag{1}$$

❑ — Le nombre de droites de $X$ sécantes à un trapèze quelconque de $A^{\|}(R)$ est $\ll n/r\log r$ avec probabilité supérieure à $1/2$.

❑ — "divide and conquer" sur les arrangements de droites





# Terminologie et Notations

Nous utilisons les notations suivantes.

- $C_0(R)$ est l'ensemble des trapèzes de l'arrangement $A^{\|}(R)$.
- $C(X) = \bigcup_{R\subseteq X} C_0(R)$.
- Pour $\sigma \in C(X)$ on note $L(\sigma)$ l'ensemble des droites de $X$ sécantes à l'intérieur de $\sigma$. Le cardinal de $L(\sigma)$ est noté $l(\sigma)$.
- Pour $\sigma \in C(X)$ on note $D(\sigma)$ l'ensemble des droites de $X$ tangentes au trapèze $\sigma$. Le cardinal de $D(\sigma)$ est noté $d(\sigma)$.
- Pour $R \subseteq X$ et $i = 0, 1, 2, \ldots$ on note $C_i(R)$ l'ensemble des $\sigma \in C(R)$ tels que $|L(\sigma)\cap R| = i$.
- $\omega_i : R \in \binom{X}{r} \mapsto \sum_{\sigma\in C_0(R)} \binom{l(\sigma)}{i} \in \mathbb{R}$
- $\omega : R \in \binom{X}{r} \mapsto \max_{\sigma\in C_0(R)} l(\sigma) \in \mathbb{R}$.

$$\sigma \in C_0(R) \iff D(\sigma) \subseteq R, L(\sigma)\cap R = \emptyset.$$
$$\sigma \in C_i(R) \iff D(\sigma) \subseteq R, |L(\sigma)\cap R| = i.$$





En particulier la <u>probabilité</u> $p_i(\sigma)$ que $\sigma \in C(X)$ appartienne à $C_i(R)$ est donnée par

$$p_0(\sigma) = \binom{a(\sigma)}{b(\sigma)}/\binom{n}{r}, \qquad p_i(\sigma) = \binom{l(\sigma)}{i}\binom{a(\sigma)}{b(\sigma)-i}/\binom{n}{r}$$

où $a(\sigma) = n - d(\sigma) - l(\sigma)$ et $b(\sigma) = r - d(\sigma)$.

En utilisant $\binom{a}{b}\binom{b}{i} = \binom{a-b+i}{i}\binom{a}{b-i}$ (= # échantillons de taille $b$ dont $i$ éléments sont distingués) on peut écrire que

$$\begin{aligned}
\binom{b(\sigma)}{i}\binom{l(\sigma)}{i}p_0(\sigma) &= \binom{a(\sigma)}{b(\sigma)}\binom{b(\sigma)}{i}\binom{l(\sigma)}{i}/\binom{n}{r} \\
&= \binom{a(\sigma)-b(\sigma)+i}{i}\binom{a(\sigma)}{b(\sigma)-i}\binom{l(\sigma)}{i}/\binom{n}{r} \\
&= \binom{a(\sigma)-b(\sigma)+i}{i}p_i(\sigma).
\end{aligned}$$

# Echantillonnage Aléatoire

❑ — $E[\omega_i] \ll (\frac{n}{r})^i E(|C_i(R)|)$ pour $i = O(1)$

$$\begin{aligned} E[\omega_i] &= \sum_{\sigma \in C(X)} \binom{l(\sigma)}{i} p_0(\sigma) = \sum_{\sigma \in C(X)} \binom{a(\sigma)-b(\sigma)+i}{i} \Big/ \binom{b(\sigma)}{i} p_i(\sigma) \\ &\ll \left(\frac{n}{r}\right)^i \sum_{\sigma \in C(X)} p_i(\sigma) \\ &= (\frac{n}{r})^i E(|C_i(R)|) \end{aligned}$$

❑ — $Q$ un échantillon aléatoire de $R$ obtenu en tirant chaque élément de $R$ avec probababilité $1/2$.

❑ — $|C_{\leq i}(R)| \ll E[|C_0(Q)|]$

$$E[|C_0(Q)|] = \sum_{\sigma \in C(R)} \mathrm{Prob}[\sigma \in C_0(Q)] \geq \sum_{\sigma \in C_{\leq i}(R)} \mathrm{Prob}[\sigma \in C_0(Q)] \geq m|C_{\leq i}(R)|$$

où $m$ est un minorant de $\mathrm{Prob}[\sigma \in C_0(Q)]$ sur $C_{\leq i}(R)$. Pour $\sigma \in C_j(R)$ on a $\mathrm{Prob}[\sigma \in C_0(Q)] = 1/2^{j+d(\sigma)}$ il suit que $m = 1/2^{i+d}$ convient.

❑ — $E[|C_0(Q)|] \ll E[|Q|^2] \ll r^2$





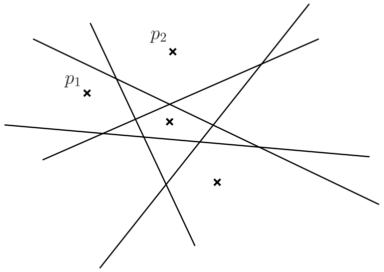
# Echantillonnage Aléatoire

❑ — $\omega(R) \ll n/r \log r$ avec probabilité $\geq 1/2$.





# Complexité de $m$ Cellules d'un Arrangement de $n$ Droites




– $P$ un ensemble de $m$ points et $L$ un ensemble de $n$ droites.

– $K(P, L)$ la complexité cumulée des $m$ cellules de $\mathcal{A}(L)$ définies par les $m$ points de $P$

– $K(P, L) = 4 + 4 + 5 + 4 = 17$

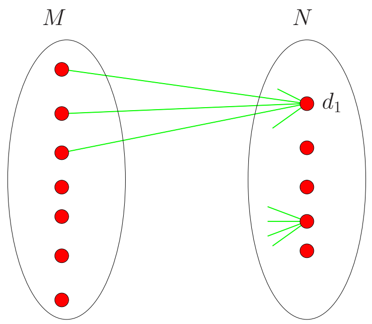
**Theorem 1 (Szemerédi and Trotter (1983) [6, 5]).** $K(m, n) \ll m^{2/3}n^{2/3} + m + n$. □

Borne de Canham : $K(P, L) = O(mn^{1/2} + n)$





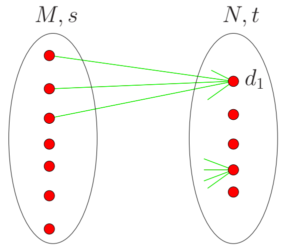
# Extremal Graph Theory




**Theorem 2 (T. Kővàri, V. Sós and P. Turàn (1954, [3])).** *Soit $A \subseteq M \times N$ un graphe bipartite entre les ensembles $M$ et $N$ de cardinaux respectifs $m$ et $n$. Supposons qu'il existe des constantes $s$ et $t$ telles que $A$ ne contienne aucun graphe $K_{s,t}$; alors*

$$|A| = O(\min\{mn^{1-1/s} + n, nm^{1-1/t} + m\}). \quad \square$$

# Lemme de T. Kővári, V. Sós and P. Turán (1954)

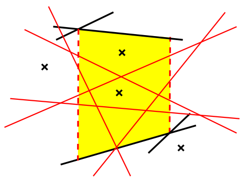


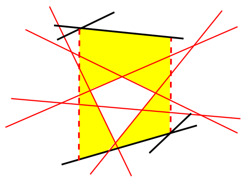

- Soit $\mathcal{H}$ l'ensemble des sous-graphes $K_{s,1}$ du graphe bipartite $A$.
- $$|A| = \sum_{i=1}^{n} d_i$$ $$|\mathcal{H}| = \sum_{i=1}^{n} \binom{d_i}{s} \leq (t-1)\binom{m}{s}$$
- + inégalité de Hölder-Minkowski





# $K(m,n)$ (suite)





- $R$ un échantillon aléatoire de $r$ droites
$$K(P,L) \ll \sum_i m_i n_i^{1/2} + n_i$$
- $\sum_i n_i = O(nr)$
- $E(n_i) = O(n/r)$ $\qquad E(n_i^{1/2}) = O((n/r)^{1/2})$

Il existe un échantillon $R$ tel que

$$\begin{aligned} K(P,L) &= O(m(n/r)^{1/2} + nr), \qquad r = \Theta(m^{2/3}n^{-1/3}) \\ &= O(m^{2/3}n^{2/3} + m + n) \end{aligned}$$





# Cutting

**Theorem 3** (**$1/r$-Cutting de taille $\ll r^2$ (Version pondérée)).** *Soit $X$ un ensemble de $n$ droites du plan en position générique chacune munie d'un poids positif $w(h)$, et soit $r$ un paramètre, $1 \leq r \leq n$. Il existe une triangulation du plan en $t \ll r^2$ triangles $\sigma_1, \ldots, \sigma_t$ de telle sorte que*
$$w(\{h \in X \mid h \cap \mathring{\sigma}_i \neq \emptyset\}) \leq \frac{1}{r} w(X).$$





# Bibliographie

## References


[1] J. Matoušek. Geometric range searching. *ACM Comput. Surv.*, 26:421–461, 1994.

[2] Ketan Mulmuley. *Computational Geometry : An Introduction Through Randomized Algorithms*. Prentice-Hall, 1994.

[3] Janos Pach and Pankaj K. Agarwal. *Combinatorial Geometry*. John Wiley & Sons, New York, NY, 1995.

[4] K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry, II. *Discrete Comput. Geom.*, 4:387–421, 1989.

[5] K. Clarkson, H. Edelsbrunner, Leonidas J. Guibas, Micha Sharir, and Emo Welzl. Combinatorial complexity bounds for arrangements of curves and spheres. *Discrete Comput. Geom.*, 5:99–160, 1990.

[6] E. Szemerédi and W. Trotter Jr. Extremal problems in discrete geometry. *Combinatorica*, 3:381–392, 1983.