Autour des algorithmes de rejet anticipé

Axel Bacher

Coauteurs: Olivier Bodini Alice Jacquot Andrea Sportiello

21 septembre 2017

Sommaire

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

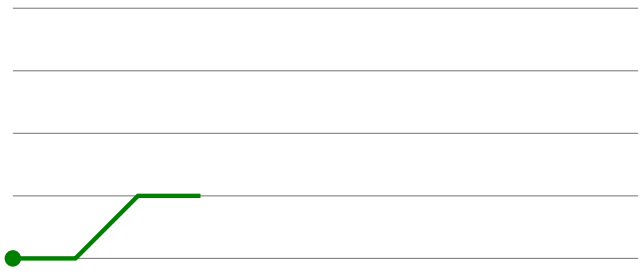
[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]



Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

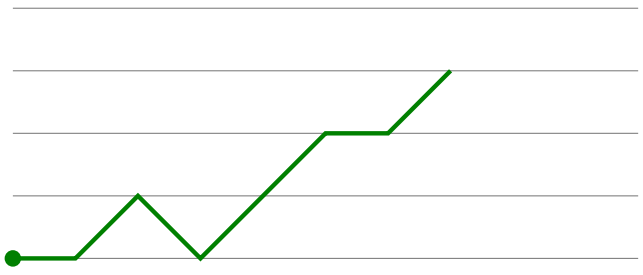
[Barcucci, Pinzani, Sprugnoli 1994, 1995]

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]

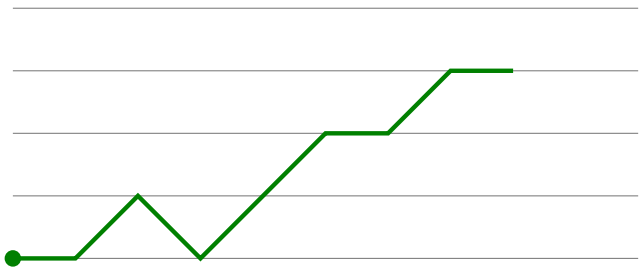
Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]



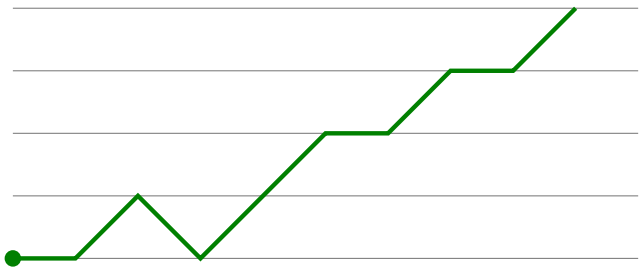
Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]



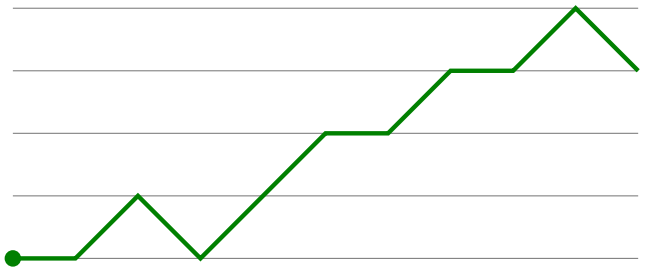
Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]



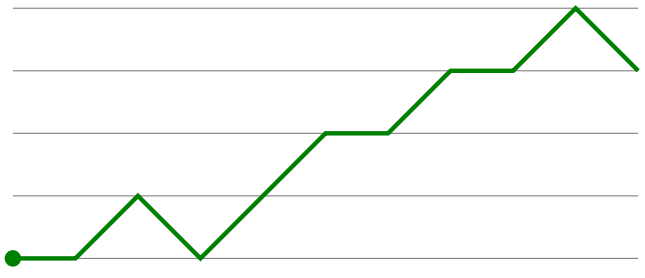
Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]



Algorithme florentin pour les préfixes de Motzkin

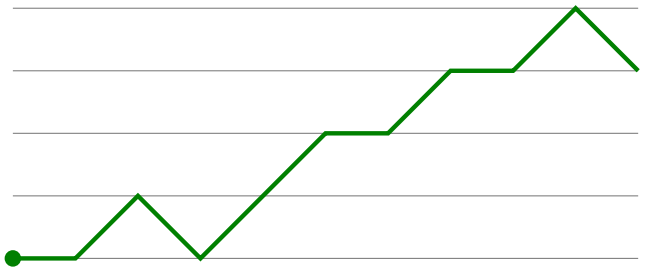
[Barcucci, Pinzani, Sprugnoli 1994, 1995]



- Complexité : $\mathcal{O}(\sqrt{n})$ essais,

Algorithme florentin pour les préfixes de Motzkin

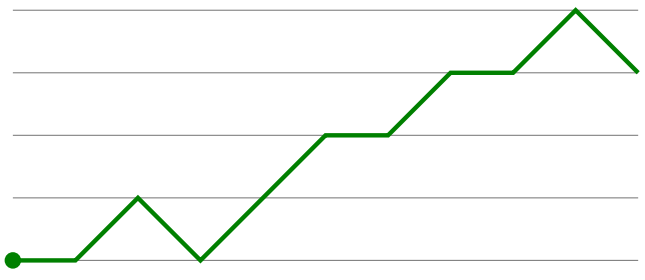
[Barcucci, Pinzani, Sprugnoli 1994, 1995]



- Complexité : $\mathcal{O}(\sqrt{n})$ essais, coût $\mathcal{O}(\sqrt{n})$ par essai
- Analyse en loi limite **[Louchard 1999]**.

Algorithme florentin pour les préfixes de Motzkin

[Barcucci, Pinzani, Sprugnoli 1994, 1995]



- Complexité : $\mathcal{O}(\sqrt{n})$ essais, coût $\mathcal{O}(\sqrt{n})$ par essai $\Rightarrow \mathcal{O}(n)$.
- Analyse en loi limite **[Louchard 1999]**.

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

$\mathbf{P}(\diagup, \diagdown) = \rho$

$\mathbf{P}(——) = \rho^2$

Algorithme florentin pour les préfixes de Schröder

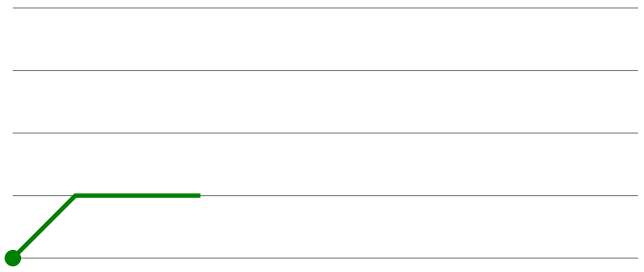
[Penaud, Pergola, Pinzani, Roques 2001]

$\mathbf{P}(\diagup, \diagdown) = \rho$

$\mathbf{P}(\text{——}) = \rho^2$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]



$$\mathbf{P}(/, \backslash) = \rho$$

$$\mathbf{P}(——) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

$$\mathbf{P}(\diagup, \diagdown) = \rho$$

$$\mathbf{P}(\text{——}) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

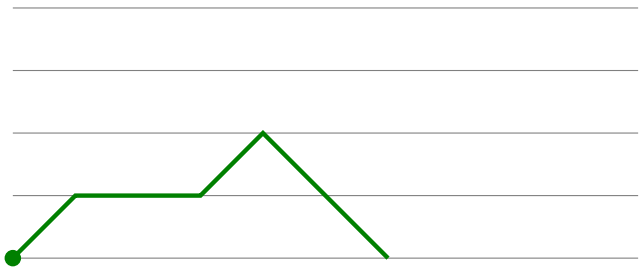
[Penaud, Pergola, Pinzani, Roques 2001]

$$\mathbf{P}(/, \backslash) = \rho$$

$$\mathbf{P}(——) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

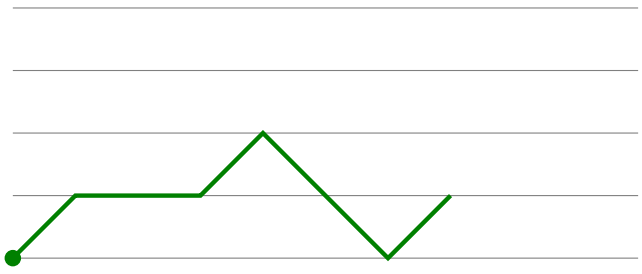


$$\mathbf{P}(\diagup, \diagdown) = \rho$$

$$\mathbf{P}(\text{——}) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

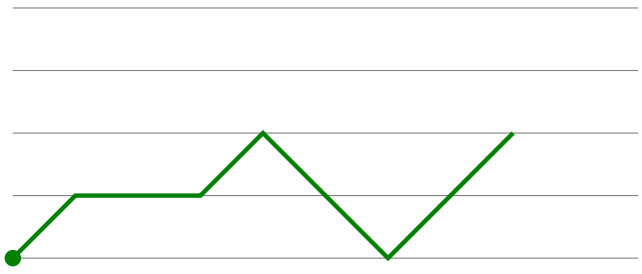


$$\mathbf{P}(\nearrow, \searrow) = \rho$$

$$\mathbf{P}(\longrightarrow) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

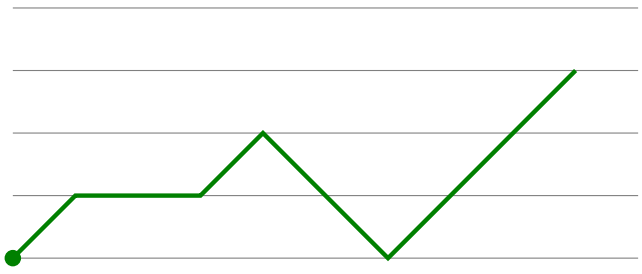


$$\mathbf{P}(\nearrow, \searrow) = \rho$$

$$\mathbf{P}(\longrightarrow) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

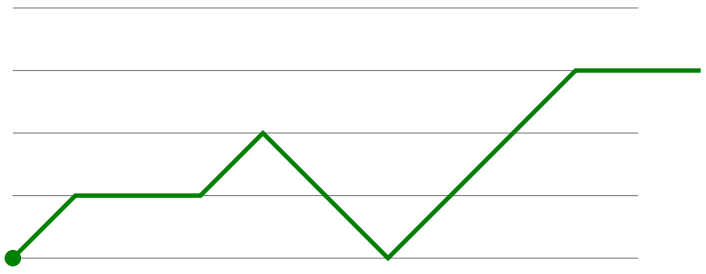


$$\mathbf{P}(\nearrow, \searrow) = \rho$$

$$\mathbf{P}(\longrightarrow) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

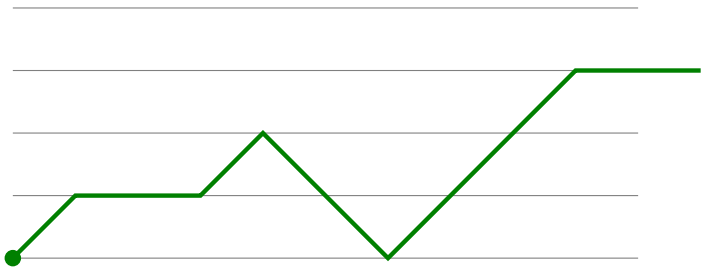


$$\mathbf{P}(\nearrow, \searrow) = \rho$$

$$\mathbf{P}(\longrightarrow) = \rho^2$$

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]



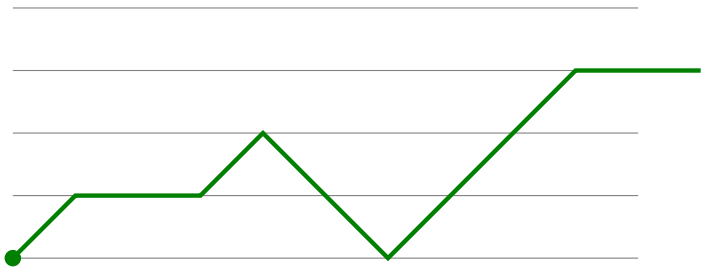
$$\mathbf{P}(/, \backslash) = \rho$$

$$\mathbf{P}(——) = \rho^2$$

- Deuxième tour de rejet si on rate la taille cible.

Algorithme florentin pour les préfixes de Schröder

[Penaud, Pergola, Pinzani, Roques 2001]

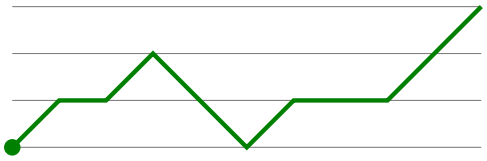


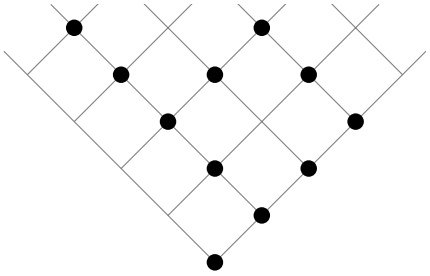
$$\mathbf{P}(\nearrow, \searrow) = \rho$$

$$\mathbf{P}(\text{——}) = \rho^2$$

- Deuxième tour de rejet si on rate la taille cible.
- La probabilité d'accepter tend vers $p = \dfrac{2+\sqrt{2}}{4} \Rightarrow$ complexité $\mathcal{O}(n)$.

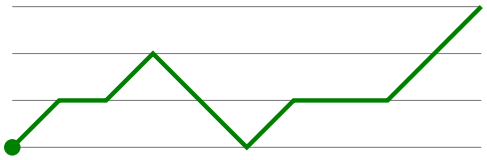
Bijections

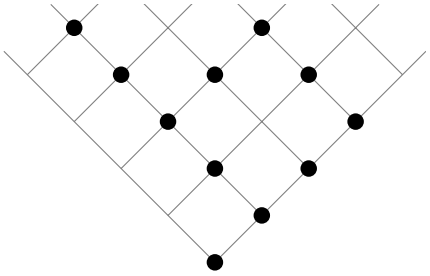




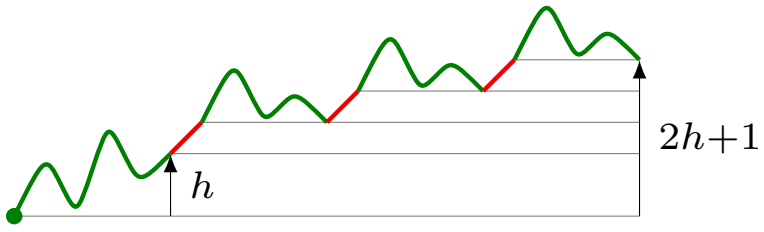
- Bijections avec des animaux dirigés :
  - préfixes de Motzkin $\longleftrightarrow$ AD sur le réseau carré ;
    [Gouyou-Beauchamps, Viennot 1988 ; Bétréma, Penaud 2001]
  - préfixes de Dyck $\longleftrightarrow$ AD sur le réseau tringulaire ;
  - petits préfixes de Schröder $\longleftrightarrow$ AD sur le réseau du roi. **[B. 2013]**

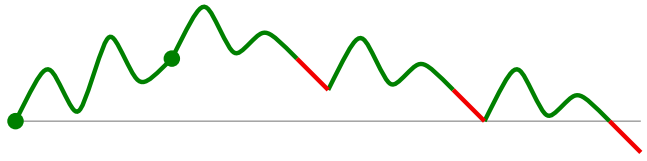
Bijections





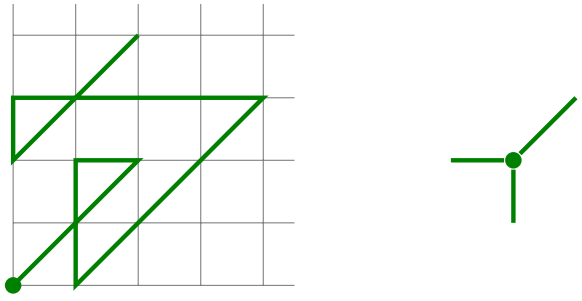
- Bijections avec des animaux dirigés :
  - préfixes de Motzkin $\longleftrightarrow$ AD sur le réseau carré ;
    [Gouyou-Beauchamps, Viennot 1988 ; Bétréma, Penaud 2001]
  - préfixes de Dyck $\longleftrightarrow$ AD sur le réseau tringulaire ;
  - petits préfixes de Schröder $\longleftrightarrow$ AD sur le réseau du roi. **[B. 2013]**



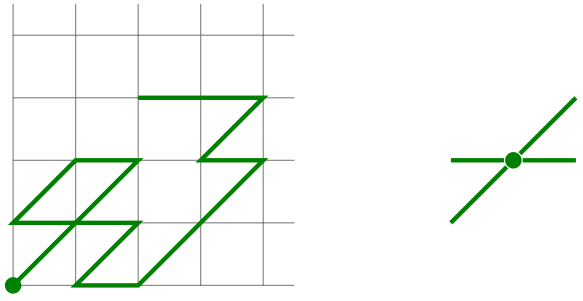


- préfixes de hauteur impaire $\longleftrightarrow$ chemins de Łukasiewicz pointés
  (pour Motzkin, 2e tour de rejet avec $p = 1/2$)

Algorithmes florentins pour d'autres marches

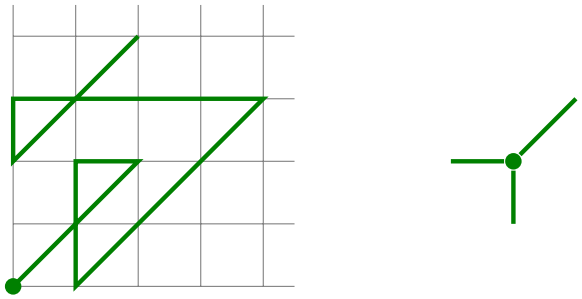


- Nombre d'essais $\mathcal{O}(n^{3/4})$.
- Coût d'un essai $\mathcal{O}(n^{1/4})$.
- Complexité $\mathcal{O}(n)$.

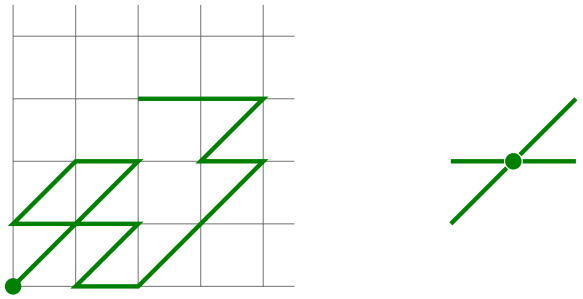


- Nombre d'essais $\mathcal{O}(n^{2/3})$.
- Coût d'un essai $\mathcal{O}(n^{1/3})$.
- Complexité $\mathcal{O}(n)$.

Algorithmes florentins pour d'autres marches



- Nombre d'essais $\mathcal{O}(n^{3/4})$.
- Coût d'un essai $\mathcal{O}(n^{1/4})$.
- Complexité $\mathcal{O}(n)$.



- Nombre d'essais $\mathcal{O}(n^{2/3})$.
- Coût d'un essai $\mathcal{O}(n^{1/3})$.
- Complexité $\mathcal{O}(n)$.

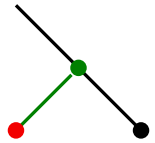
- Il est possible d'engendrer efficacement toutes les familles de marches dans le quart de plan **[Lumbroso, Mishna, Ponty 2016]**.
- Autres possibilités : marches dans un cône, d dimensions, etc.

Algorithme de greffe pour les arbres binaires

[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

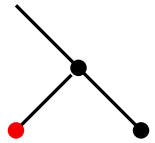
Algorithme de greffe pour les arbres binaires



[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

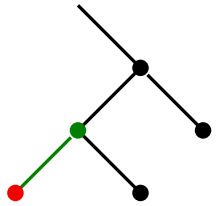
Algorithme de greffe pour les arbres binaires



[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

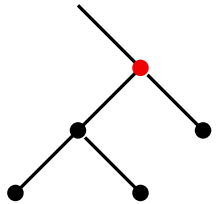
Algorithme de greffe pour les arbres binaires



[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

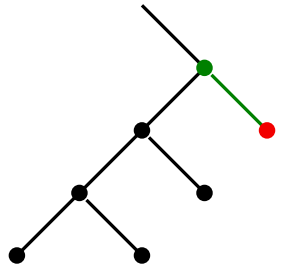
Algorithme de greffe pour les arbres binaires



[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

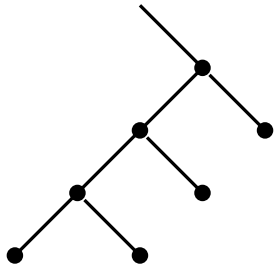
Algorithme de greffe pour les arbres binaires



[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

Algorithme de greffe pour les arbres binaires



[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

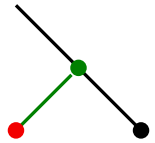
Algorithme de greffe pour les arbres binaires

[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

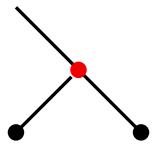


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

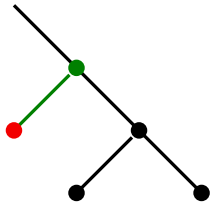


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

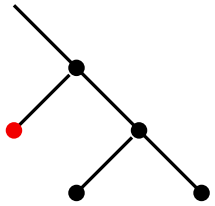


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

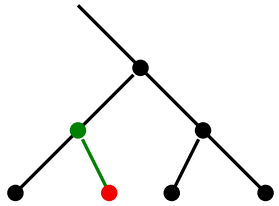


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

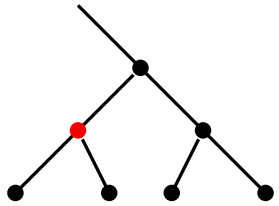


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

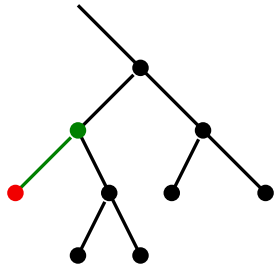


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

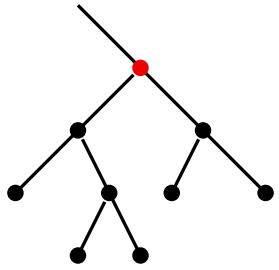


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

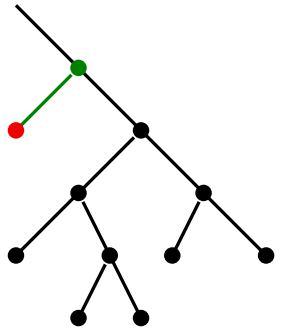


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

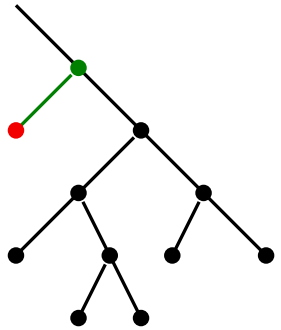


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.

Algorithme de greffe pour les arbres binaires

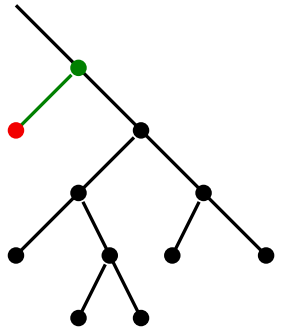


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.
- À chaque étape, l'arbre pointé est distribué uniformément.

Algorithme de greffe pour les arbres binaires

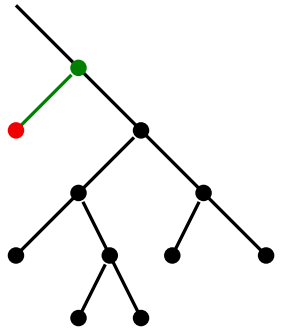


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.
- À chaque étape, l'arbre pointé est distribué uniformément.
- Complexité $\mathcal{O}(\sqrt{n}) \times \mathcal{O}(\sqrt{n}) = \mathcal{O}(n)$.

Algorithme de greffe pour les arbres binaires

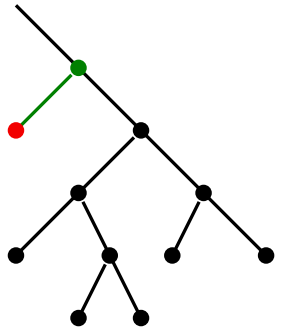


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.
- À chaque étape, l'arbre pointé est distribué uniformément.
- Complexité $\mathcal{O}(\sqrt{n}) \times \mathcal{O}(\sqrt{n}) = \mathcal{O}(n)$.
- Variante de l'algorithme de Rémy (qui coûte $\mathcal{O}(n \log n)$ bits aléatoires).

Algorithme de greffe pour les arbres binaires

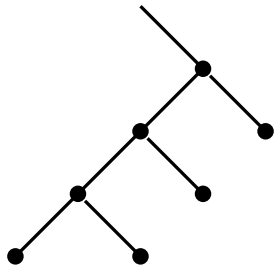


[B., Bodini, Jacquot 2013]

- À chaque étape du processus :
  - On crée un nouveau nœud et une nouvelle feuille à gauche ou à droite ;
  - Avec probabilité $1/2$, on remonte le point.

  Si le point sort de l'arbre, on rejette et on recommence.
- À chaque étape, l'arbre pointé est distribué uniformément.
- Complexité $\mathcal{O}(\sqrt{n}) \times \mathcal{O}(\sqrt{n}) = \mathcal{O}(n)$.
- Variante de l'algorithme de Rémy (qui coûte $\mathcal{O}(n \log n)$ bits aléatoires).
- Fonctionne aussi sur les arbres 1-2, avec un 2e tour de rejet ($p = 3/4$).

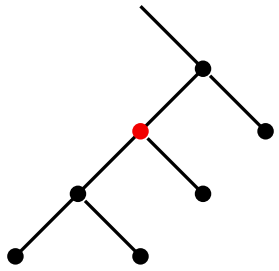
Raffinement de l'algorithme de greffe



[B., Bodini, Jacquot 2013]

- On peut améliorer l'algorithme précédent : si le point sort de l'arbre, au lieu de rejeter, on retire un point aléatoire.

Raffinement de l'algorithme de greffe



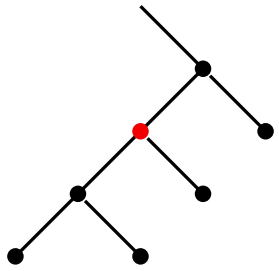
[B., Bodini, Jacquot 2013]

- On peut améliorer l'algorithme précédent : si le point sort de l'arbre, au lieu de rejeter, on retire un point aléatoire.
- Surcoût moyen dû à ces « rattrapages » :

$$\sum_{i=1}^{n} \frac{\log_2 i}{2i} = \mathcal{O}(\log^2 n).$$

Le coût en bits aléatoires est donc de $2n + \mathcal{O}(\log^2 n)$, ce qui est asymptotiquement optimal (algorithme entropique).

Raffinement de l'algorithme de greffe



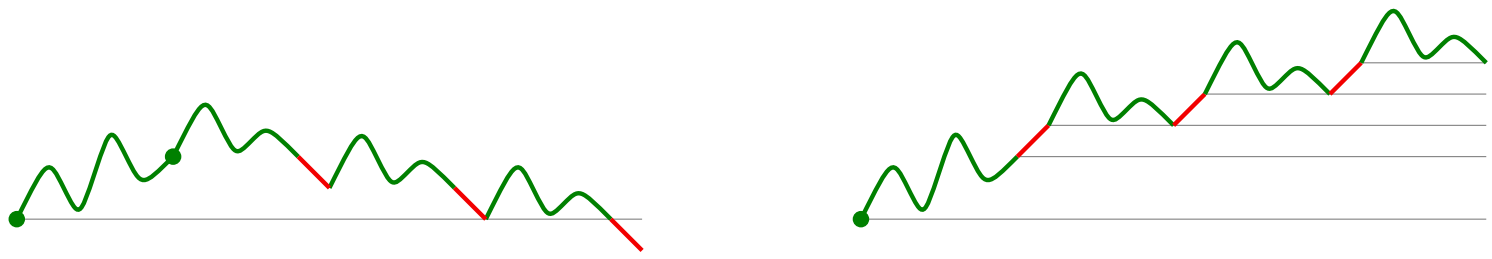
[B., Bodini, Jacquot 2013]

- On peut améliorer l'algorithme précédent : si le point sort de l'arbre, au lieu de rejeter, on retire un point aléatoire.
- Surcoût moyen dû à ces « rattrapages » :

$$\sum_{i=1}^{n} \frac{\log_2 i}{2i} = \mathcal{O}(\log^2 n).$$

  Le coût en bits aléatoires est donc de $2n + \mathcal{O}(\log^2 n)$, ce qui est asymptotiquement optimal (algorithme entropique).
- Ne fonctionne pas pour les arbres 1-2 (on perd l'uniformité).

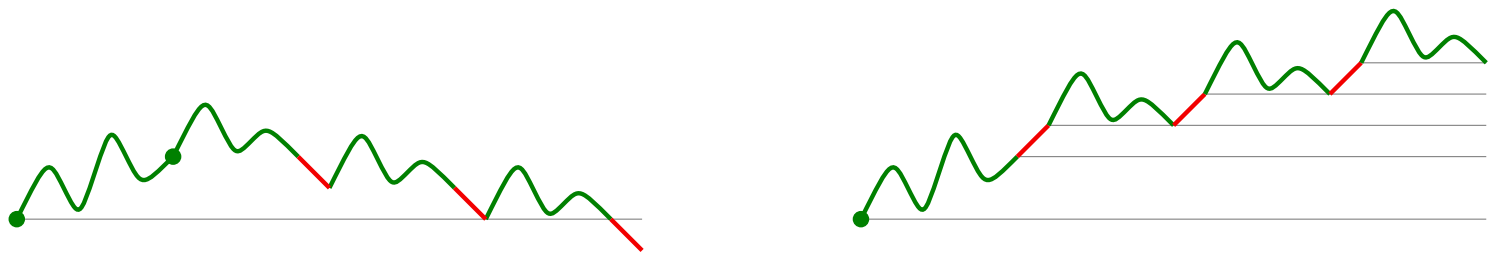
Algorithme de « rattrapage » pour les chemins de Dyck



[B. 2016]

- Pour engendrer un préfixe de Dyck, on tire des pas aléatoires ; si on crève le plancher, on tire un point aléatoire et on déplie.

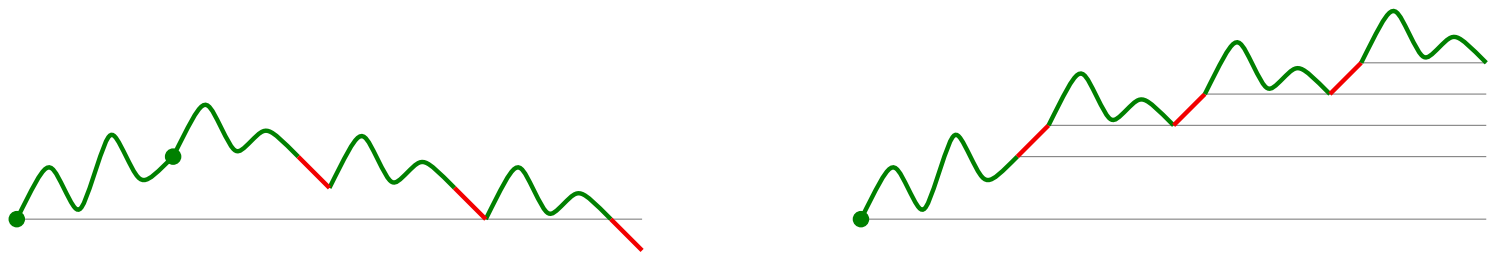
Algorithme de « rattrapage » pour les chemins de Dyck



[B. 2016]

- Pour engendrer un préfixe de Dyck, on tire des pas aléatoires ; si on crève le plancher, on tire un point aléatoire et on déplie.
- À chaque étape, le chemin est uniformément distribué.

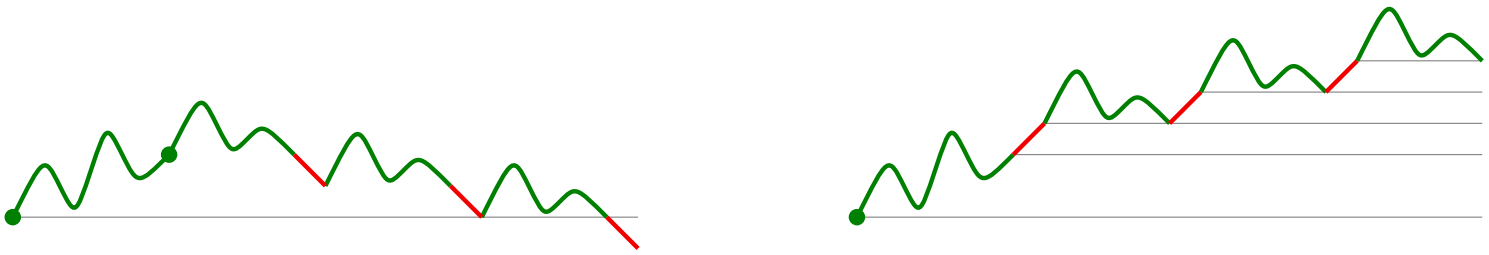
Algorithme de « rattrapage » pour les chemins de Dyck



[B. 2016]

- Pour engendrer un préfixe de Dyck, on tire des pas aléatoires ; si on crève le plancher, on tire un point aléatoire et on déplie.
- À chaque étape, le chemin est uniformément distribué.
- Surcoût en bits aléatoires : $\mathcal{O}(\log^2 n)$ (algorithme entropique).

Algorithme de « rattrapage » pour les chemins de Dyck

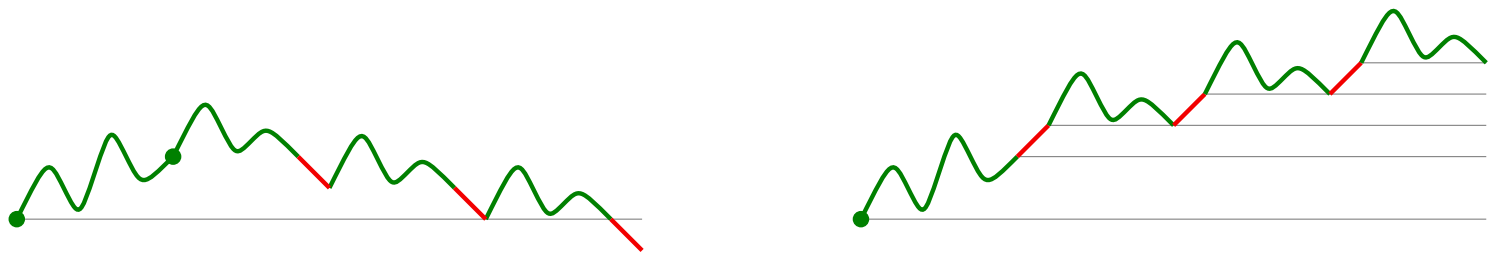


[B. 2016]

- Pour engendrer un préfixe de Dyck, on tire des pas aléatoires ; si on crève le plancher, on tire un point aléatoire et on déplie.
- À chaque étape, le chemin est uniformément distribué.
- Surcoût en bits aléatoires : $\mathcal{O}(\log^2 n)$ (algorithme entropique).
- Surcoût en accès mémoire :

$$\sum_{i=0}^{n-1} \frac{i}{4i} = \mathcal{O}(n).$$

Algorithme de « rattrapage » pour les chemins de Dyck



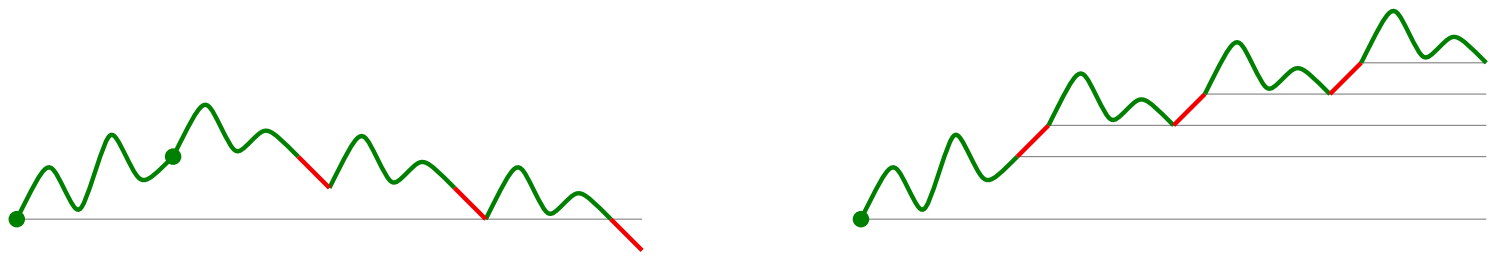
[B. 2016]

- Pour engendrer un préfixe de Dyck, on tire des pas aléatoires ; si on crève le plancher, on tire un point aléatoire et on déplie.
- À chaque étape, le chemin est uniformément distribué.
- Surcoût en bits aléatoires : $\mathcal{O}(\log^2 n)$ (algorithme entropique).
- Surcoût en accès mémoire :

$$\sum_{i=0}^{n-1} \frac{i}{4i} = \mathcal{O}(n).$$

- Extension possible aux chemins de m-Dyck (pas $+1$, $-m$), entropique si on dispose d'une source entropique de $\mathrm{Bernoulli}\left(\frac{1}{m+1}\right)$.

Algorithme de « rattrapage » pour les chemins de Dyck



[B. 2016]

- Pour engendrer un préfixe de Dyck, on tire des pas aléatoires ; si on crève le plancher, on tire un point aléatoire et on déplie.
- À chaque étape, le chemin est uniformément distribué.
- Surcoût en bits aléatoires : $\mathcal{O}(\log^2 n)$ (algorithme entropique).
- Surcoût en accès mémoire :

$$\sum_{i=0}^{n-1} \frac{i}{4i} = \mathcal{O}(n).$$

- Extension possible aux chemins de m-Dyck (pas $+1$, $-m$), entropique si on dispose d'une source entropique de $\mathrm{Bernoulli}\left(\frac{1}{m+1}\right)$.
- Ne fonctionne pas pour Motzkin, ni pour Schröder.

Loi limite du rejet anticipé

- Soient $(X_i)_{i\geqslant 0}$ des v.a. i.i.d. de loi X, telle que pour tout $x > 0$:

$$\frac{\mathbf{P}[X \geqslant xt]}{\mathbf{P}[X \geqslant t]} \xrightarrow[t\to\infty]{} x^{-\alpha} \qquad (0 < \alpha < 1).$$

Loi limite du rejet anticipé

- Soient $(X_i)_{i\geqslant 0}$ des v.a. i.i.d. de loi X, telle que pour tout $x > 0$:

$$\frac{\mathbf{P}[X \geqslant xt]}{\mathbf{P}[X \geqslant t]} \xrightarrow[t\to\infty]{} x^{-\alpha} \qquad (0 < \alpha < 1).$$

- Si $t > 0$, on définit :

$$i(t) = \min\{i \mid X_i \geqslant t\} \quad \text{et} \quad S(t) = X_0 + \cdots + X_{i(t)-1}.$$

Loi limite du rejet anticipé

- Soient $(X_i)_{i\geqslant 0}$ des v.a. i.i.d. de loi X, telle que pour tout $x > 0$:

$$\frac{\mathbf{P}[X \geqslant xt]}{\mathbf{P}[X \geqslant t]} \xrightarrow[t\to\infty]{} x^{-\alpha} \qquad (0 < \alpha < 1).$$

- Si $t > 0$, on définit :

$$i(t) = \min\{i \mid X_i \geqslant t\} \quad \text{et} \quad S(t) = X_0 + \cdots + X_{i(t)-1}.$$

Théorème [B., Sportiello 2015]

Le processus $S(t)/t$ tend en distribution vers la loi D_α telle que :

$$\mathbf{E}\left[e^{zD_\alpha}\right] = \left(1 - \sum_{n=1}^{\infty} \frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}.$$

Loi limite du rejet anticipé

- Soient $(X_i)_{i\geqslant 0}$ des v.a. i.i.d. de loi X, telle que pour tout $x>0$:

$$\frac{\mathbf{P}[X\geqslant xt]}{\mathbf{P}[X\geqslant t]} \xrightarrow[t\to\infty]{} x^{-\alpha} \qquad (0<\alpha<1).$$

- Si $t>0$, on définit :

$$i(t)=\min\{i \mid X_i\geqslant t\} \quad \text{et} \quad S(t)=X_0+\cdots+X_{i(t)-1}.$$

Théorème [B., Sportiello 2015]

Le processus $S(t)/t$ tend en distribution vers la loi D_α telle que :

$$\mathbf{E}\left[e^{zD_\alpha}\right]=\left(1-\sum_{n=1}^{\infty}\frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}.$$

- Éléments de preuve :

$$\mathbf{E}\left[e^{zS(t)}\right]=\frac{\mathbf{P}[X\geqslant t]}{1-\mathbf{E}[e^{zX}\mathbf{1}_{X<t}]}\,; \qquad \frac{\mathbf{E}[X^n\mathbf{1}_{X<t}]}{\mathbf{P}[X\geqslant t]}\sim\frac{\alpha}{n-\alpha}t^n.$$

Loi limite du rejet anticipé

- Soient $(X_i)_{i\geqslant 0}$ des v.a. i.i.d. de loi X, telle que pour tout $x>0$:

$$\frac{\mathbf{P}[X\geqslant xt]}{\mathbf{P}[X\geqslant t]} \xrightarrow[t\to\infty]{} x^{-\alpha} \qquad (0<\alpha<1).$$

- Si $t>0$, on définit :

$$i(t)=\min\{i \mid X_i\geqslant t\} \quad \text{et} \quad S(t)=X_0+\cdots+X_{i(t)-1}.$$

Théorème [B., Sportiello 2015]

Le processus $S(t)/t$ tend en distribution vers la loi D_α telle que :

$$\mathbf{E}\left[e^{zD_\alpha}\right]=\left(1-\sum_{n=1}^{\infty}\frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}.$$

- Éléments de preuve :

$$\mathbf{E}\left[e^{zS(t)}\right]=\frac{\mathbf{P}[X\geqslant t]}{1-\mathbf{E}[e^{zX}\mathbf{1}_{X<t}]}\,; \qquad \frac{\mathbf{E}[X^n\mathbf{1}_{X<t}]}{\mathbf{P}[X\geqslant t]}\sim\frac{\alpha}{n-\alpha}t^n.$$

- La loi D_α est dite de Darling-Mandelbrot **[Darling 1952, Lew 1994]**.

Loi limite du rejet anticipé

- Soient $(X_i)_{i\geqslant 0}$ des v.a. i.i.d. de loi X, telle que pour tout $x>0$:

$$\frac{\mathbf{P}[X\geqslant xt]}{\mathbf{P}[X\geqslant t]} \xrightarrow[t\to\infty]{} x^{-\alpha} \qquad (0<\alpha<1).$$

- Si $t>0$, on définit :

$$i(t)=\min\{i \mid X_i\geqslant t\} \quad \text{et} \quad S(t)=X_0+\cdots+X_{i(t)-1}.$$

Théorème [B., Sportiello 2015]

Le processus $S(t)/t$ tend en distribution vers la loi D_α telle que :

$$\mathbf{E}\left[e^{zD_\alpha}\right]=\left(1-\sum_{n=1}^{\infty}\frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}.$$

- Éléments de preuve :

$$\mathbf{E}\left[e^{zS(t)}\right]=\frac{\mathbf{P}[X\geqslant t]}{1-\mathbf{E}[e^{zX}\mathbf{1}_{X<t}]}\,; \qquad \frac{\mathbf{E}[X^n\mathbf{1}_{X<t}]}{\mathbf{P}[X\geqslant t]}\sim\frac{\alpha}{n-\alpha}t^n.$$

- La loi D_α est dite de Darling-Mandelbrot **[Darling 1952, Lew 1994]**.
- Si $\alpha\geqslant 1$, convergence vers une loi exponentielle.

Loi limite des algorithmes de rattrapage

- On note B_n et M_n les surcoûts en bits aléatoires et accès mémoire dus aux rattrapages.

Théorème [B. 2016]

Le processus B_n se comporte comme une gaussienne, avec :

$$\mathbf{E}[B_n] \sim \frac{\log^2 n}{4 \log 2}, \qquad \mathbf{V}[B_n] \sim \frac{\log^3 n}{6 \log^2 2}.$$

Le processus M_n/n tend en distribution vers la loi $L_{1/2}$, avec :

$$\mathbf{E}\left[e^{zL_\alpha}\right] = \exp\left(\sum_{n=1}^{\infty} \frac{\alpha}{n(n+1)} \frac{z^n}{n!}\right).$$

Loi limite des algorithmes de rattrapage

- On note B_n et M_n les surcoûts en bits aléatoires et accès mémoire dus aux rattrapages.

Théorème [B. 2016]

Le processus B_n se comporte comme une gaussienne, avec :

$$\mathbf{E}[B_n] \sim \frac{\log^2 n}{4 \log 2}, \qquad \mathbf{V}[B_n] \sim \frac{\log^3 n}{6 \log^2 2}.$$

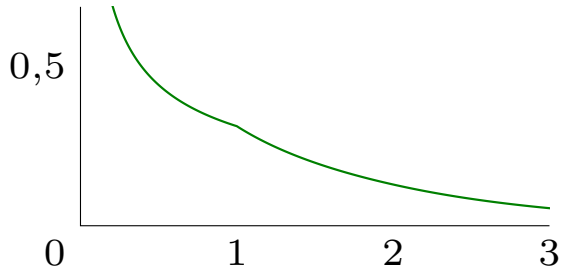
Le processus M_n/n tend en distribution vers la loi $L_{1/2}$, avec :

$$\mathbf{E}\left[e^{zL_\alpha}\right] = \exp\left(\sum_{n=1}^{\infty} \frac{\alpha}{n(n+1)} \frac{z^n}{n!}\right).$$

- Preuve : l'ensemble des points de rattrapage divisés par n tend vers un processus de Poisson sur $(0, 1]$ de densité $\lambda(x) = \dfrac{1}{2x}$.

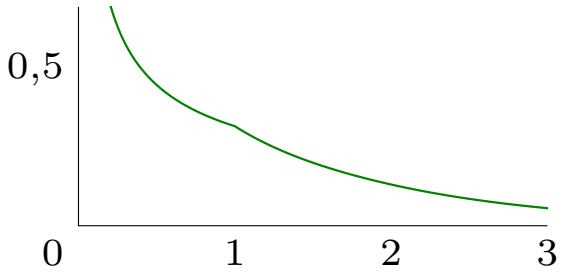
Densité de la loi D_α

$$\mathbf{E}\left[e^{zD_\alpha}\right] = \left(1 - \sum_{n=1}^{\infty} \frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}$$



Densité de la loi D_α

$$\mathbf{E}\left[e^{zD_\alpha}\right] = \left(1 - \sum_{n=1}^{\infty} \frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}$$



- La transformée de Laplace de D_α se met sous la forme :

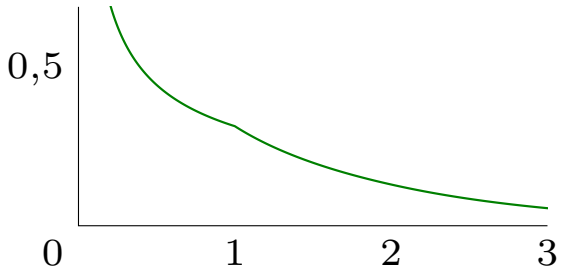
$$\mathbf{E}\left[e^{-zD_\alpha}\right] = \frac{A(z)}{1 - B(z)},$$

$$A(z) = \frac{z^{-\alpha}}{\Gamma(1-\alpha)}$$

$$B(z) = \int_z^\infty \frac{e^{-y}y^{-1-\alpha}}{\Gamma(-\alpha)}\,dy.$$

Densité de la loi D_α

$$\mathbf{E}\left[e^{zD_\alpha}\right] = \left(1 - \sum_{n=1}^{\infty} \frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}$$



- La transformée de Laplace de D_α se met sous la forme :

$$\mathbf{E}\left[e^{-zD_\alpha}\right] = \frac{A(z)}{1-B(z)},$$

$$A(z) = \frac{z^{-\alpha}}{\Gamma(1-\alpha)}$$

$$B(z) = \int_z^\infty \frac{e^{-y}y^{-1-\alpha}}{\Gamma(-\alpha)}\,dy.$$

- Sa densité $f(x)$ vaut donc :

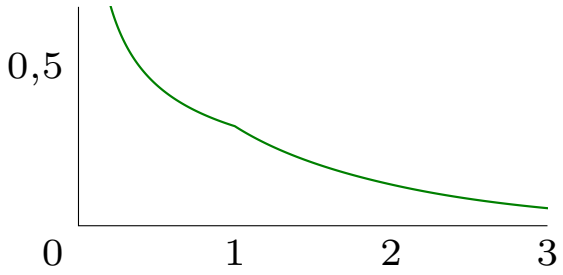
$$f(x) = \sum_{k=0}^{\infty} a * b^{*k}(x),$$

$$a(x) = \frac{\sin(\alpha\pi)}{\pi}x^{\alpha-1}$$

$$b(x) = -\frac{\sin(\alpha\pi)}{\pi}\frac{(x-1)^\alpha}{x}\mathbf{1}_{x>1}$$

Densité de la loi D_α

$$\mathbf{E}\left[e^{zD_\alpha}\right] = \left(1 - \sum_{n=1}^{\infty} \frac{\alpha}{n-\alpha}\frac{z^n}{n!}\right)^{-1}$$



- La transformée de Laplace de D_α se met sous la forme :

$$\mathbf{E}\left[e^{-zD_\alpha}\right] = \frac{A(z)}{1-B(z)},$$

$$A(z) = \frac{z^{-\alpha}}{\Gamma(1-\alpha)}$$

$$B(z) = \int_z^\infty \frac{e^{-y}y^{-1-\alpha}}{\Gamma(-\alpha)}\,dy.$$

- Sa densité $f(x)$ vaut donc :

$$f(x) = \sum_{k=0}^{\infty} a * b^{*k}(x),$$

$$a(x) = \frac{\sin(\alpha\pi)}{\pi}x^{\alpha-1}$$

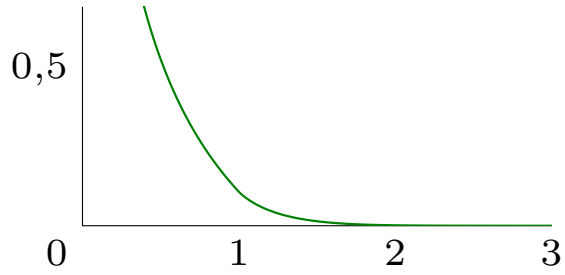
$$b(x) = -\frac{\sin(\alpha\pi)}{\pi}\frac{(x-1)^\alpha}{x}\mathbf{1}_{x>1}$$

et vérifie :

$$xf'(x) + (1-\alpha)f(x) = -\alpha f * f(x-1).$$

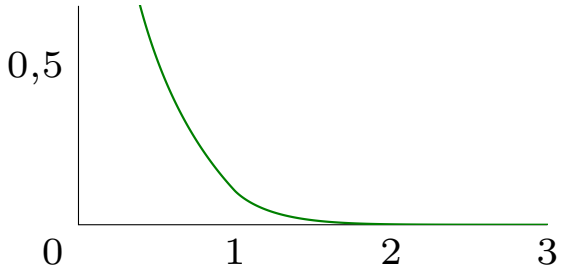
Densité de la loi $L_{1/2}$

$$\mathbf{E}\left[e^{zL_{1/2}}\right] = \exp\left(\sum_{n=1}^{\infty} \frac{1}{2n(n+1)} \frac{z^n}{n!}\right)$$



Densité de la loi $L_{1/2}$

$$\mathbf{E}\left[e^{zL_{1/2}}\right] = \exp\left(\sum_{n=1}^{\infty} \frac{1}{2n(n+1)} \frac{z^n}{n!}\right)$$



- La transformée de Laplace de $L_{1/2}$ se met sous la forme :

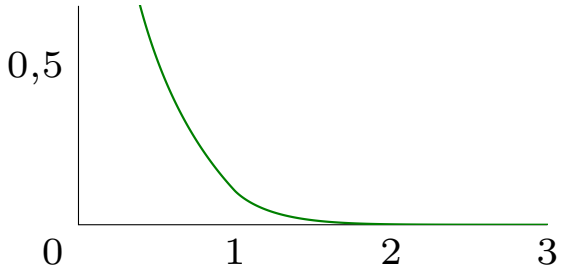
$$\mathbf{E}\left[e^{-zL_{1/2}}\right] = A(z)\exp\big(B(z)\big),$$

$$A(z) = e^{\frac{1-\gamma}{2}} e^{-\frac{1}{2z}} z^{-1/2}$$

$$B(z) = \int_z^{\infty} \frac{e^{-y}}{2y^2}\, dy.$$

Densité de la loi $L_{1/2}$

$$\mathbf{E}\left[e^{zL_{1/2}}\right] = \exp\left(\sum_{n=1}^{\infty} \frac{1}{2n(n+1)} \frac{z^n}{n!}\right)$$



- La transformée de Laplace de $L_{1/2}$ se met sous la forme :

$$\mathbf{E}\left[e^{-zL_{1/2}}\right] = A(z)\exp\big(B(z)\big),$$

$$A(z) = e^{\frac{1-\gamma}{2}} e^{-\frac{1}{2z}} z^{-1/2}$$

$$B(z) = \int_z^{\infty} \frac{e^{-y}}{2y^2}\, dy.$$

- Sa densité $f(x)$ vaut donc :

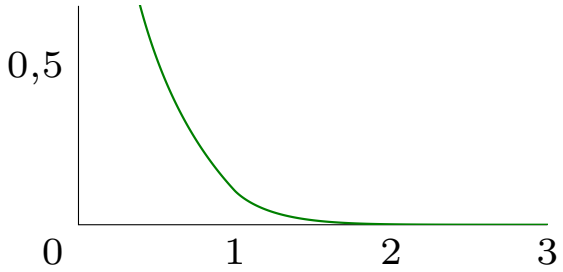
$$f(x) = \sum_{k=0}^{\infty} \frac{a * b^{*k}(x)}{k!},$$

$$a(x) = e^{\frac{1-\gamma}{2}} \frac{\cos\sqrt{2x}}{\sqrt{\pi x}}$$

$$b(x) = \frac{x-1}{2x} \mathbf{1}_{x>1}$$

Densité de la loi $L_{1/2}$

$$\mathbf{E}\left[e^{zL_{1/2}}\right] = \exp\left(\sum_{n=1}^{\infty} \frac{1}{2n(n+1)} \frac{z^n}{n!}\right)$$



- La transformée de Laplace de $L_{1/2}$ se met sous la forme :

$$\mathbf{E}\left[e^{-zL_{1/2}}\right] = A(z)\exp\big(B(z)\big),$$

$$A(z) = e^{\frac{1-\gamma}{2}} e^{-\frac{1}{2z}} z^{-1/2}$$

$$B(z) = \int_z^{\infty} \frac{e^{-y}}{2y^2}\, dy.$$

- Sa densité $f(x)$ vaut donc :

$$f(x) = \sum_{k=0}^{\infty} \frac{a * b^{*k}(x)}{k!},$$

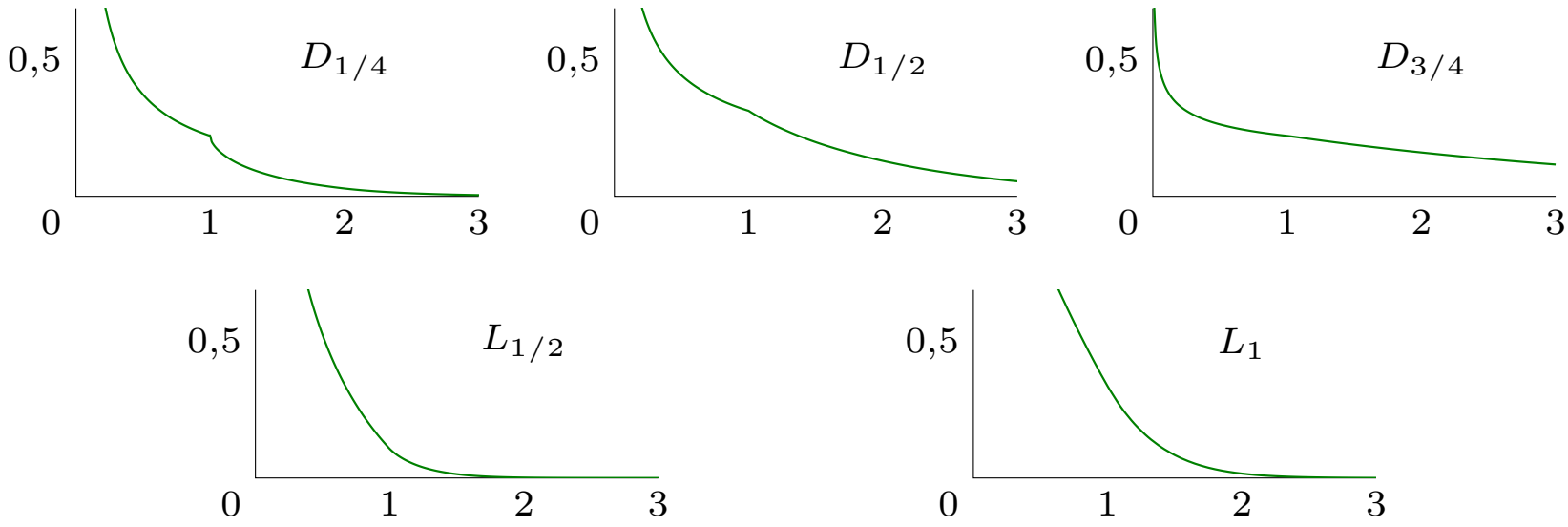
$$a(x) = e^{\frac{1-\gamma}{2}} \frac{\cos\sqrt{2x}}{\sqrt{\pi x}}$$

$$b(x) = \frac{x-1}{2x} \mathbf{1}_{x>1}$$

et vérifie :

$$2xf''(x) + 3f'(x) + f(x) = f(x-1).$$

Autres résultats sur les densités



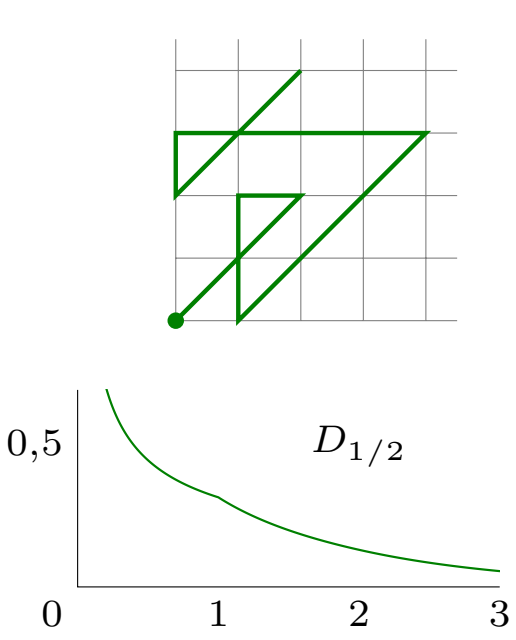
- Queues de distribution **[Lew 94]** :

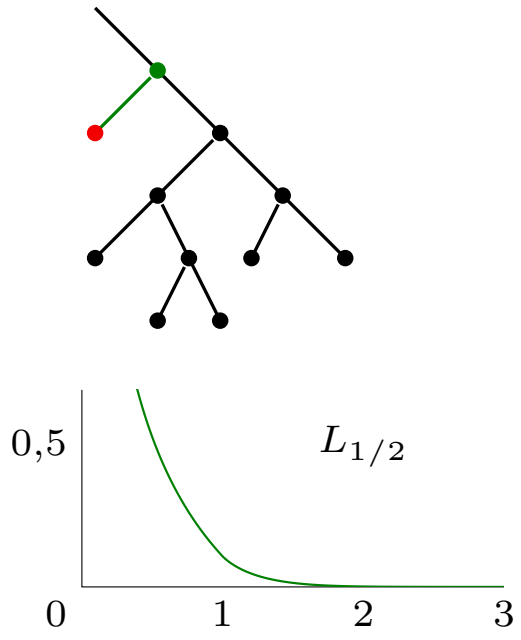
$$\mathbf{P}[D_\alpha \geqslant x] = \frac{e^{-a_0}}{\alpha} e^{-a_0 x} + \mathcal{O}(e^{-a_1 x})$$
$$\mathbf{P}[L_\alpha \geqslant x] = x^{-x} (\log x)^{-2x} (\alpha e)^{x+o(x)}$$

- En cas de deuxième tour de rejet avec $p = 1/(1+\beta)$:

$$\mathbf{E}\left[e^{zD_{\alpha,\beta}}\right] = \left(1 - \sum_{n=1}^{\infty} \frac{\alpha + \beta n}{n - \alpha} \frac{z^n}{n!}\right)^{-1}$$

Perspectives





- Peut-on appliquer ces procédés à d'autres familles de marches ou d'arbres ? ($+a/-b$, etc.)
- Existe-t-il d'autres distributions ayant le même genre de propriétés ? (ex : fonction de Dickman en théorie des nombres)