Dans les 2 parties de ce TD/cours, vous trouverez un exemple de démonstration (à lire attentivement!) dont vous pouvez vous inspirer pour répondre aux questions suivantes.

I Preuve d'équation de récurrence

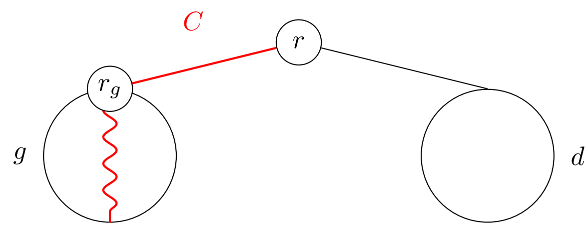
I.1 Exemple : récurrence sur la hauteur

Théorème : Hauteur

Soit a un arbre binaire non vide, de sous-arbre gauche g et sous-arbre droit d.
On note h_a la hauteur de a, définie comme la longueur maximum (en nombre d'arêtes) d'un chemin de la racine de a à une feuille. Alors :

$$h_a = 1 + \max(h_g, h_d)$$

<u>Preuve</u> : Soit C un chemin de longueur maximum de la racine de a à une feuille.
C passe soit dans g, soit dans d (et pas dans les deux). Supposons que C passe dans g, l'autre cas étant symétrique.

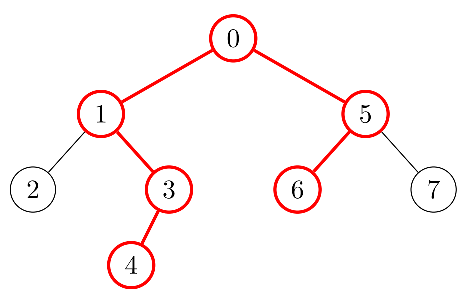


Soit C_g la partie de C qui est incluse dans g.
Supposons que C_g ne soit pas un chemin de longueur maximum de la racine à une feuille dans g. Il existe alors un chemin C'_g plus long que C_g dans g. Mais alors la concaténation de l'arête de r à r_g (racine de g) et du C'_q est plus long que C, ce qui est une contradiction.
Donc C_g est un plus long chemin de r_g à une feuille de g : sa longueur est donc h_g par définition. D'où $h_a = h_g + 1$.
Si C passe par d, on a, par un raisonnement similaire : $h_a = h_d + 1$.
Comme la hauteur est le maximum sur tous les chemins possibles :

$$\boxed{h_a = 1 + \max(h_g, h_d)}$$

I.2 Exercice : Diamètre

Le **diamètre** d_a d'un arbre a est la longueur maximum d'un chemin entre 2 noeuds quelconques de cet arbre. Par exemple, le diamètre de l'arbre ci-dessous est 5, correspondant au chemin $4 \to 3 \to 1 \to 0 \to 5 \to 7$.



1. Soit a un arbre binaire composé d'une racine r_a, d'un sous-arbre gauche g et d'un sous-arbre d. Donner, en la démontrant, une relation de récurrence permettant de calculer d_a.

   Solution : Montrons que $d_a = \max(d_g, d_d, h_g + h_d + 2)$. Pour cela, remarquons qu'il y a 3 possibilités pour un chemin C dans a : soit C passe par r_a, soit C est entièrement dans g ou dans d.

   - La longueur maximum d'un chemin dans g est égal à d_g
   - La longueur maximum d'un chemin dans d est égal à d_d

- Soit C un chemin de a, passant par r_a et de longueur maximum. Notons $l(C)$ sa longueur. Montrons que $l(C) = h_g + h_d + 2$. Pour cela, notons C_g la partie de C dans g. Alors C_g est un chemin maximum de g depuis la racine de g (si ce n'était pas le cas, on pourrait trouver un chemin C_g' plus long et remplacer C_g par C_g' dans C pour obtenir une contradiction). Donc $l(C_g) = h_g$, par définition de la hauteur. De même pour $l(C_d)$.
  Donc $l(C) = l(C_g) + l(C_d) + 2 = h_g + h_d + 2$ (le $+2$ venant des 2 arêtes issues de r_a).

Comme d_a correspond à la longueur du chemin maximum parmi ces 3 possibilités :

$$\boxed{d_a = \max(d_g, d_d, h_g + h_d + 2)}$$

2. En déduire une fonction permettant de calculer le diamètre d'un arbre binaire. Quelle est sa complexité ?

Solution : On choisit de définir le diamètre d'un arbre comme étant `-1`, ce qui est compatible avec l'équation de récurrence trouvée en question précédente.

```
let rec h t = function (* hauteur *)
    | E -> -1
    | N(_, g, d) -> 1 + max (h g) (h d);;

let rec diam t = function
    | E -> 0
    | N(_, g, d) -> max (max (diam g) (diam d)) (h g + h d + 2);;
```

`diam t` effectue un appel récursif sur chaque noeud de `t`, donc effectue n opérations, où n est le nombre de noeuds de t.

3. Pourquoi la fonction précédente n'est pas très efficace ? L'améliorer pour calculer le diamètre en complexité linéaire.

Solution : `diam` calcule plusieurs fois les mêmes hauteurs (une fois dans l'appel `h g` puis à nouveau dans l'appel récursif `d g`). Pour éviter ce problème, on peut renvoyer à la fois hauteur et diamètre :

```
let rec diam t = function (* renvoie diamètre, hauteur *)
    | E -> -1, -1
    | N(_, g, d) -> let dg, hg = diam g in
                    let dd, hd = diam d in
                    max (max dg dd) (hg + hd + 2), 1 + max hg hd
```

On pourrait aussi stocker en mémoire la hauteur de chaque sous-arbre (en créant un nouvel arbre et en mettant la hauteur comme étiquette sur chaque noeud) pour éviter de la calculer plusieurs fois (mémoïsation).

II Preuve de formule sur les arbres

II.1 Exemple : nombre d'arêtes d'un arbre binaire

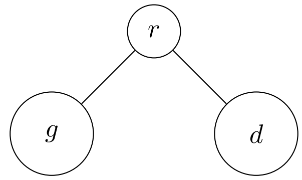
Théorème : Nombre d'arêtes

Soit a un arbre binaire à $n \geq 1$ noeuds.
Alors a possède $n - 1$ arêtes.

Preuve : Démontrons, par récurrence forte :

$$H_n : \text{« un arbre binaire à } n \geq 1 \text{ noeuds possède } n-1 \text{ arêtes »}$$

1. H_1 est clairement vraie : un arbre à 1 sommet possède 0 arête.
2. Supposons H_n vraie et soit a un arbre binaire à $n + 1$ noeuds.
   a se décompose comme une racine r, un sous-arbre gauche g et un sous-arbre droit d :



- Si $g = \emptyset$, alors d a n noeuds donc $n-1$ arêtes d'après H_n. Avec l'arête de r vers la racine de d, il y a donc bien $\boxed{n \text{ arêtes}}$ au total.
- De même si $d = \emptyset$.
- Sinon, soit k le nombre de noeuds de g. d possède alors $n+1-k$ noeuds. D'après H_k, g possède $k-1$ arêtes. D'après H_{n-k}, d possède $n-k-1$ arêtes. Donc a possède :

$$\underbrace{2}_{r} + \underbrace{k-1}_{g} + \underbrace{n-k-1}_{d} = \boxed{n \text{ arêtes}}$$

Remarque : On aurait aussi pu faire une récurrence sur la hauteur de l'arbre, l'important étant que le paramètre sur lequel on fait la récurrence soit strictement plus petit sur les sous-arbres. La plupart des récurrences sur les arbres se font sur le nombre de noeuds ou la hauteur, au choix.

II.2 Exercice : Nombre de feuilles

1. Démontrer que, si a est un arbre binaire avec f_a feuilles et de hauteur h_a :

$$f_a \leq 2^{h_a}$$

Solution : Par récurrence sur la hauteur :

$$P_h : \text{« si } a \text{ est un arbre binaire avec } f_a \text{ feuilles et de hauteur } h_a \leq h \text{ alors } f_a \leq 2^{h_a} \text{ »}$$

Remarque : le fait d'écrire $h_a \leq h$ revient à faire une récurrence sur la hauteur.

- P_{-1} est vraie car si a est de hauteur -1 est a est l'arbre vide qui vérifie bien $f_a = 0 \leq 2^{-1} = h_a$
- Supposons P_h vraie. Soit a un arbre binaire de hauteur $h_a = h+1$.
  Alors a est non-vide donc possède deux sous-arbres g et d.
  Si g et d sont vides alors a est réduit à une feuille et est de hauteur 0, ce qui vérifie bien $f_a = 1 \leq 2^h = 2^0 = 1$.
  Sinon, étant de hauteurs au plus h, on peut leur appliquer P_h : $f_g \leq 2^{h_g}$ et $f_d \leq 2^{h_d}$. Comme $f_a = f_g + f_d$ (car la racine de a n'est pas une feuille) :

$$f_a = f_g + f_d \leq 2^{h_g} + 2^{h_d} \underset{h_g \leq h}{\leq} 2^h + 2^h = 2^{h+1}$$

On rappelle qu'une feuille est un noeud sans fils (dont les deux sous-arbres sont vides) et qu'un noeud interne est un noeud qui n'est pas une feuille. Un arbre est **binaire strict** si ses noeuds ont 0 ou 2 fils.

2. Soit a un arbre binaire **strict**. Conjecturer une formule reliant le nombre f_a de feuilles de a avec son nombre n_a de noeuds internes, puis la prouver par récurrence.

Solution : Montrons par récurrence forte que H_n est vraie pour tout $n \in \mathbb{N}^*$:

$$H_n : \text{Si } a \text{ est un arbre binaire strict non vide avec } n \text{ noeuds alors } f_a = n_a + 1$$

- H_1 est vraie car un arbre avec un noeud est forcément de la forme $a = N(r, E, E)$ (une racine et deux sous-arbres vides) et $f_a = 1$, $n_a = 0$ donc $f_a = n_a + 1$.
- Soit $n \in \mathbb{N}^*$. Supposons H_k, $\forall k \leq n$. Soit a un arbre binaire strict à $n+1$ noeuds.
  a est non vide donc possède deux fils g et d. Comme a possède $n+1 \geq 2$ noeuds, g ou d est non-vide. Comme a est strict, g et d sont non-vides.
  Comme g et d sont binaires, non-vides, stricts et avec moins de noeuds que a, on peut leur appliquer l'hypothèse de récurrence :

$$f_g = n_g + 1 \text{ et } f_d = n_d + 1$$

Donc :

$$\begin{aligned} f_a &= f_g + f_d = n_g + 1 + n_d + 1 \\ &= n_a + 1 \end{aligned}$$

Car $n_a = n_g + n_d + 1$ (la racine de a est un noeud interne).

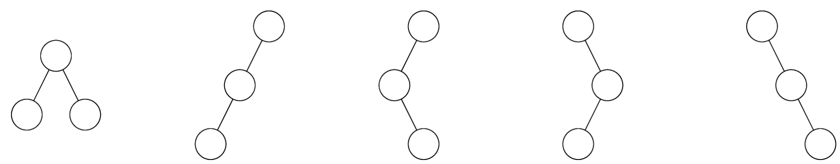
II.3 Exercice : Nombre d'arbres binaires

Dans cet exercice, on veut compter le nombre a_n d'arbres binaires à n noeuds.

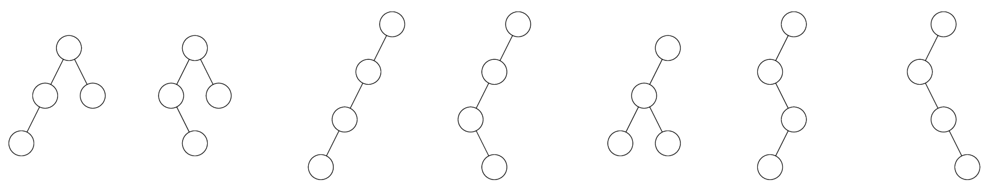
1. Que vaut a_1, a_2, a_3, a_4 ? Dessiner les arbres correspondants.

Solution :

- $a_1 = 1$: `N(_, E, E)`
- $a_2 = 2$: `N(_, N(_, E, E), E)` et `N(_, E, N(_, E, E))`
- $a_3 = 5$:



- $a_4 = 14$:



Ainsi que tous les symétriques (obtenus en échangeant sous-arbre gauche et sous-arbre droit)

2. Montrer que, si $n \in \mathbb{N}$:

$$a_{n+1} = \sum_{k=0}^{n} a_k a_{n-k}$$

Solution : On voit un arbre non-vide comme une couple (g, d) (sous-arbre gauche, sous-arbre droit). Soit A_n l'ensemble des arbres binaires à n noeuds. Alors, comme un arbre à $n+1$ noeuds est obtenu à partir d'un sous-arbre gauche à k noeuds et un sous-arbre droit à $n-k$ noeuds :

$$A_{n+1} = \bigcup_{k=0}^{n} \{(g, d) \mid g \text{ a } k \text{ noeuds et } d \text{ a } n-k \text{ noeuds } \} \qquad = \bigcup_{k=0}^{n} A_k \times A_{n-k}$$

De plus, l'union ci-dessus est disjointe. Donc « le cardinal de l'union est la somme des cardinaux » :

$$|A_{n+1}| = \sum_{k=0}^{n} |A_k \times A_{n-k}|$$

De plus, si A et B sont des ensembles, $|A \times B| = |A| \times |B|$. Donc :

$$|A_{n+1}| = \sum_{k=0}^{n} |A_k||A_{n-k}|$$

$$\boxed{a_{n+1} = \sum_{k=0}^{n} a_k a_{n-k}}$$

L'équation de récurrence ci-dessus permet de montrer (en utilisant, par exemple, des séries entières que vous allez voir plus tard en mathématiques) que a_n est égal au **nombre de Catalan** :

$$\boxed{a_n = \frac{1}{n+1}\binom{2n}{n}}$$