6. Récurrences

Motivations

* Les relations de récurrence munies de conditions initiales permettent de définir des suites de nombres.
* En informatique, elles proviennent essentiellement :
  * des définitions inductives
  * des stratégies du type « diviser pour régner » (divide and conquer) utilisant la récursivité (exemple : la dichotomie)
* Les équations de récurrence sont aux mathématiques discrètes ce que les équations différentielles sont aux mathématiques continues.
* On peut (quelquefois) les résoudre par différentes techniques selon les types de récurrence.

Suite de Fibonacci*

* Léonard de Pise, dit Fibonacci (1175-1240).

* Dans un problème récréatif posé dans un de ses ouvrages, Fibonacci décrit la croissance d'une population de lapins : « Possédant initialement un couple de lapins, combien de couples obtient-on en douze mois si chaque couple engendre tous les mois un nouveau couple à compter du second mois de son existence ? »

* Notons F_n le nombre de couples de lapins au bout du n-ième mois.
  * $F_1 = F_2 = 1$
  * $F_3 = 2$
  * $F_4 = 3$
  * Et, pour $n>2$, $F_n = F_{n-1} + F_{n-2}$.
* 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

Suite de Fibonacci*

* La suite de Fibonacci 1, 1, 2, 3, 5, 8, 13, 21, 34, ... est définie par
  * $F_1 = F_2 = 1$
  * Et, pour $n>2$, $F_n = F_{n-1} + F_{n-2}$.
    * Remarque. On commence quelques fois à $F_0 = F_1 = 1$
* Questions
  * Peut on calculer directement le $n^{ième}$ terme sans calculer les précédents ?
  * Y a t il un algorithme pour calculer F_n rapidement ?
  * Pour n grand, a t on une idée de l'ordre de grandeur de F_n ?

Calcul direct du $n^{ième}$ terme de la suite de Fibonacci

* Le $n^{ième}$ terme de la suite de Fibonacci est donné par la formule

$$F_n = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^n - \left(\frac{1-\sqrt{5}}{2} \right)^n \right)$$

  * En supposant que
    * $F_0 = F_1 = 1$
    * Et, pour $n>1$, $F_n = F_{n-1} + F_{n-2}$.

* Dans le cas le plus général, cela s'appelle résoudre l'équation de récurrence. On verra quelques méthodes pour des cas plus ou moins particuliers, comme par exemple
  * les équations linéaires, comme dans le cas de Fibonacci
  * les équations polynomiales
  * les équations non polynomiales.

Algorithmes de calcul du $n^{ième}$ terme de la suite de Fibonacci

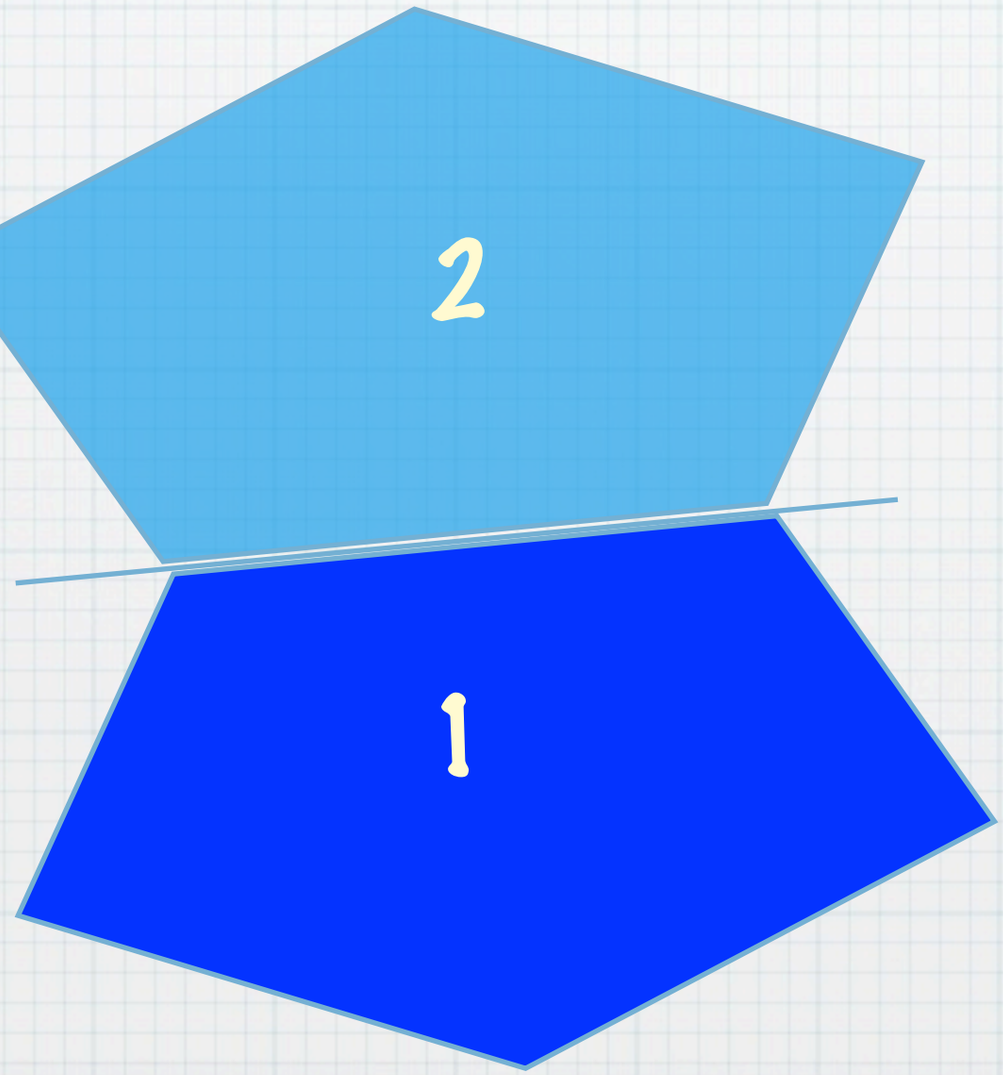
* Programme récursif
  * (define (fibo n)
    (if (< n 2) 1 (+ (fibo (- n 1)) (fibo (- n 2)))))
* En utilisant le calcul matriciel

$$\begin{pmatrix} F_n \\ F_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} F_{n-1} \\ F_{n-2} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{n-1} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

* En utilisant la formule

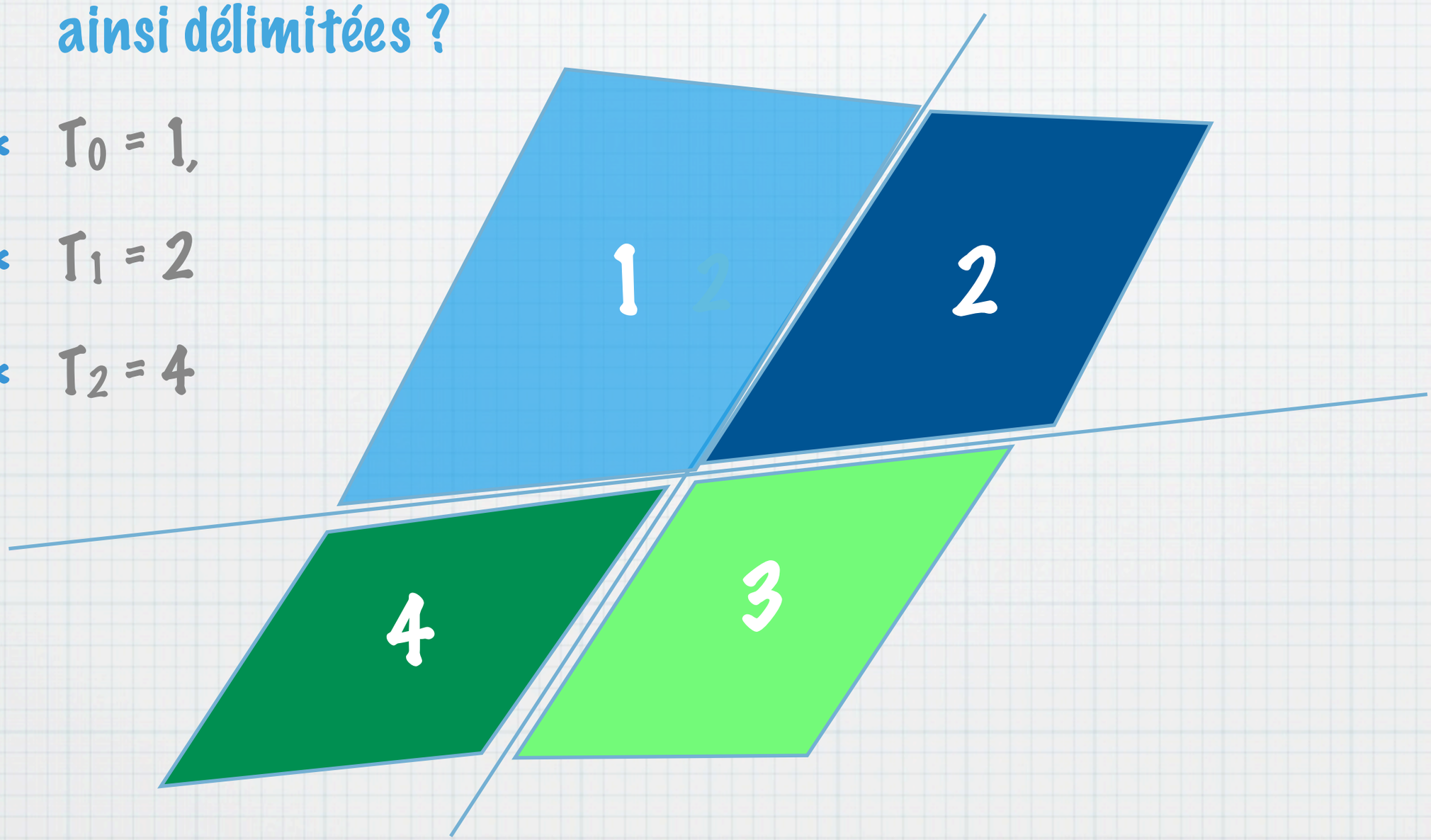
Résolution d'une équation linéaire d'ordre 1

* On trace n droites (2 à 2 sécantes) dans le plan. Elles délimitent un ensemble de régions finies ou infinies. **Quel est le nombre de régions T_n ainsi délimitées ?**
* $T_0 = 1$,
* $T_1 = 2$



Résolution d'une équation linéaire d'ordre 1

* On trace n droites (2 à 2 sécantes) dans le plan. Elles délimitent un ensemble de régions finies ou infinies. **Quel est le nombre de régions T_n ainsi délimitées ?**
* $T_0 = 1$,
* $T_1 = 2$
* $T_2 = 4$



Résolution d'une équation linéaire d'ordre 1

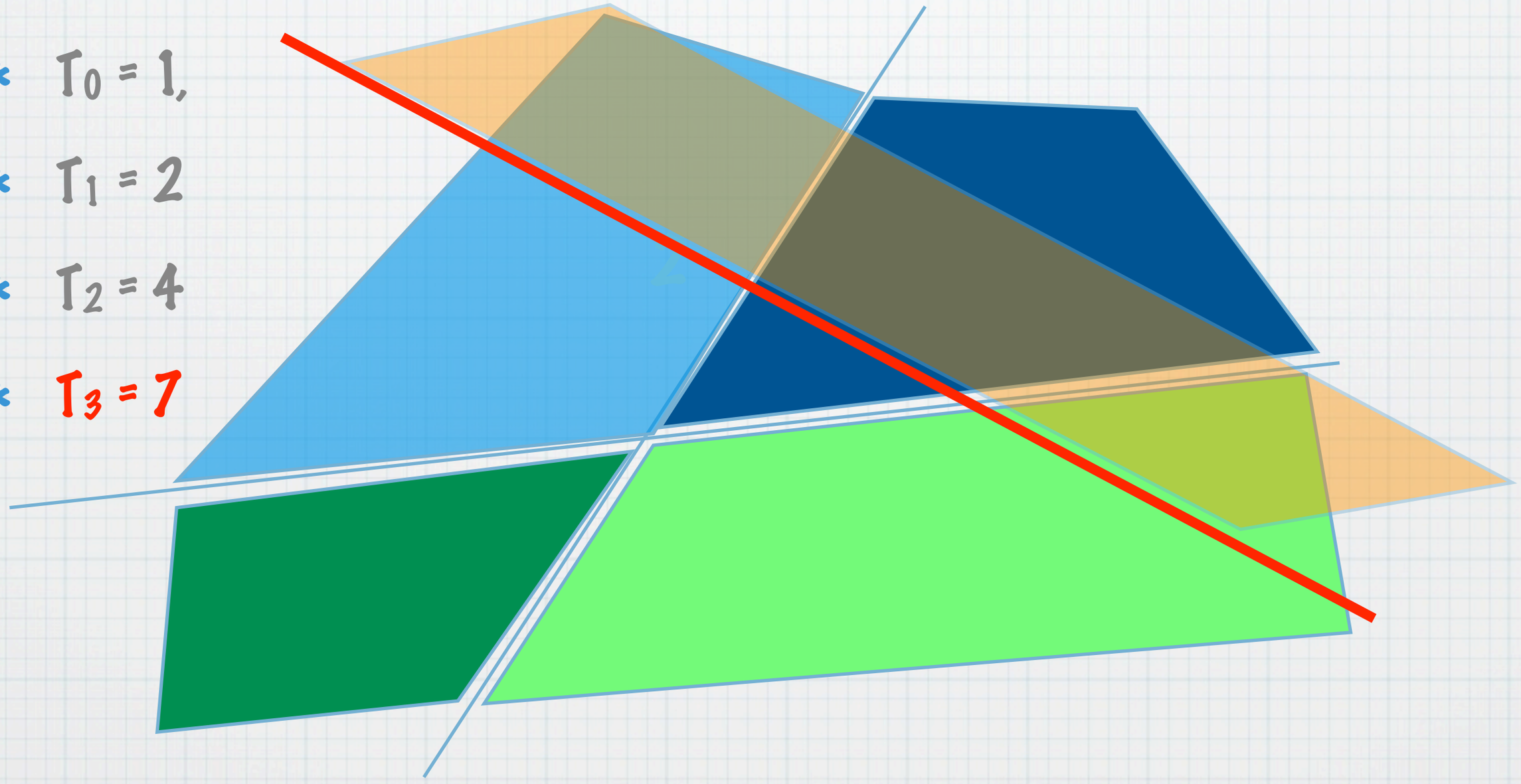
* On trace n droites (2 à 2 sécantes) dans le plan. Elles délimitent un ensemble de régions finies ou infinies. **Quel est le nombre de régions T_n ainsi délimitées ?**

* $T_0 = 1$,

* $T_1 = 2$

* $T_2 = 4$

* $T_3 = 7$

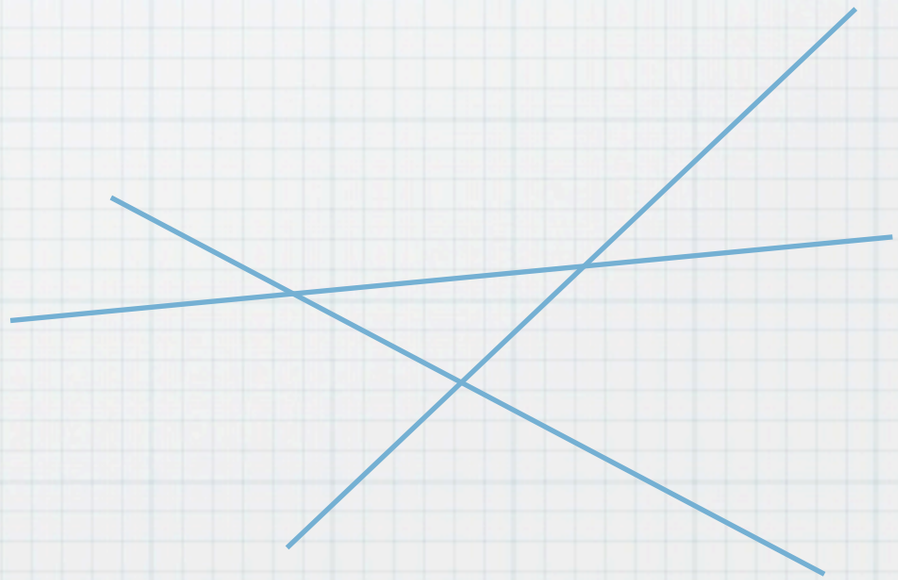


Résolution d'une équation linéaire d'ordre 1

* On trace n droites (2 à 2 sécantes) dans le plan. Elles délimitent un ensemble de régions finies ou infinies. **Quel est le nombre de régions T_n ainsi délimitées ?**
* $T_0 = 1, T_1 = 2, T_2 = 4, T_3 = 7 \ldots$
* On réalise que $T_n = T_{n-1} + n$.

$$
\begin{aligned}
T_n &= T_{n-1} + n \\
T_{n-1} &= T_{n-2} + n-1 \\
T_{n-2} &= T_{n-3} + n-2 \\
&\ldots \\
T_1 &= T_0 + 1
\end{aligned}
$$

$$T_n = T_0 + \sum_{1 \le i \le n} i = 1 + n(n+1)/2$$



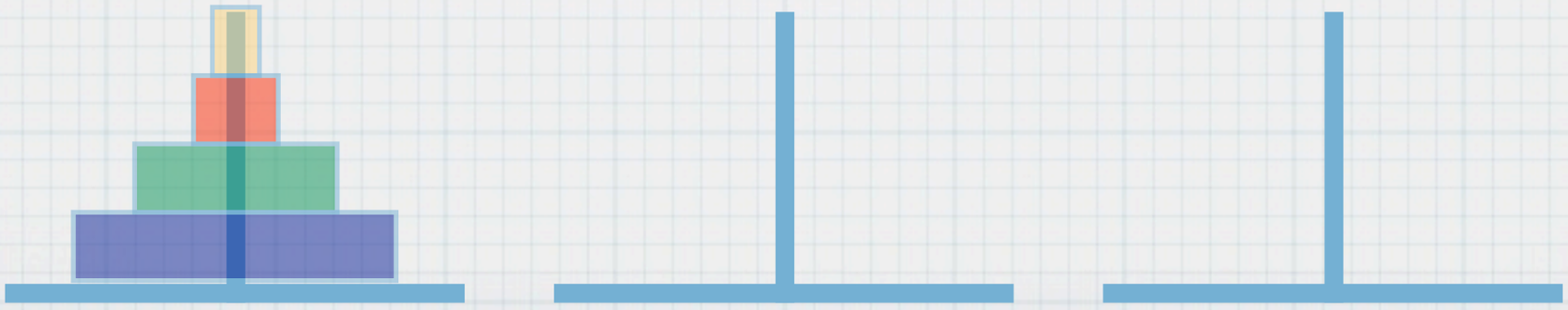
Tours de Hanoï

* Le problème des tours de Hanoï est un jeu de réflexion imaginé par le mathématicien français Édouard Lucas, et consistant à déplacer des disques de diamètres différents d'une tour de « départ » à une tour d'« arrivée » en passant par une tour « intermédiaire » et ceci en un minimum de coups, tout en respectant les règles suivantes :

* on ne peut déplacer plus d'un disque à la fois,

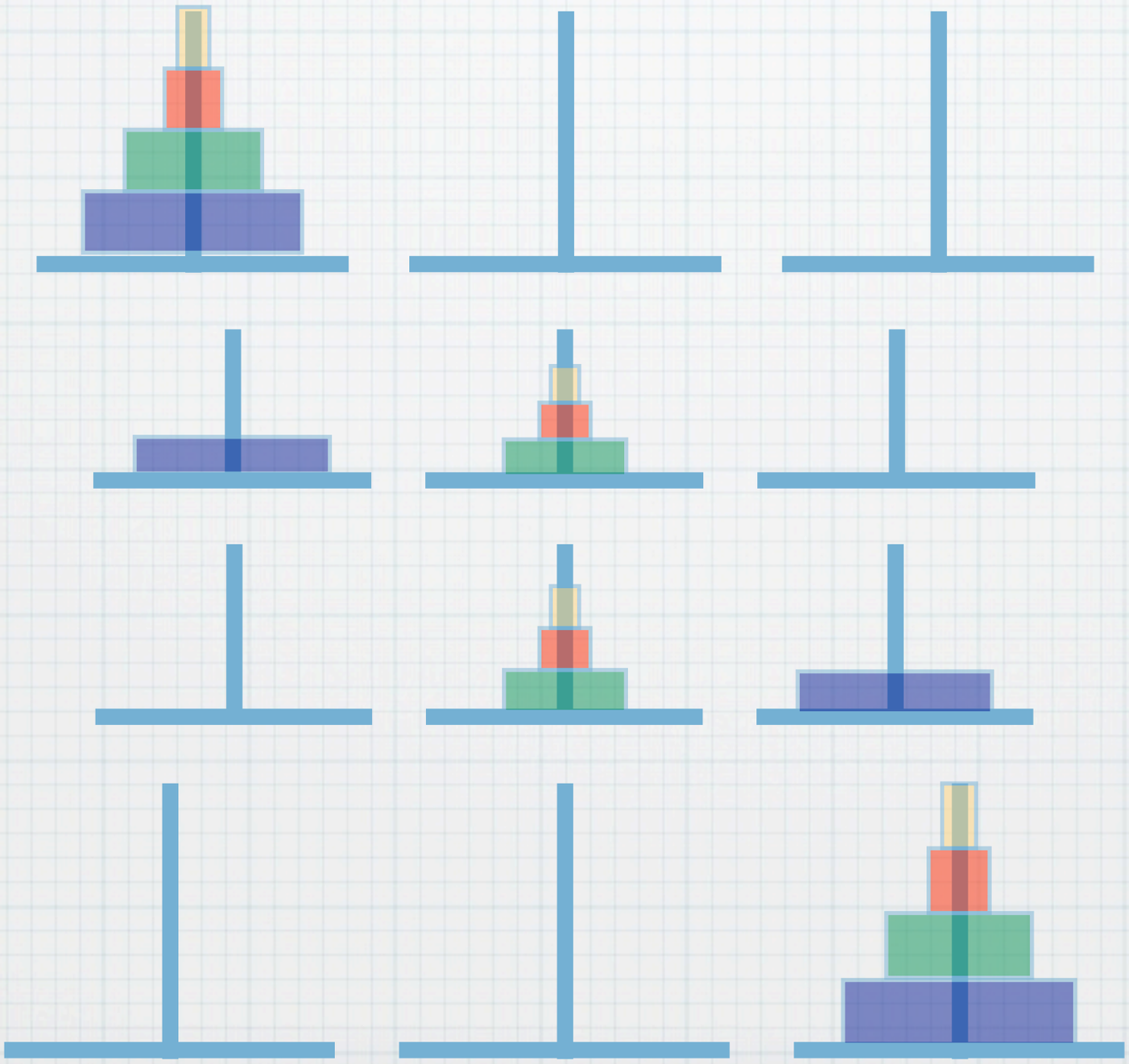
* on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide.

* On suppose que cette dernière règle est également respectée dans la configuration de départ.



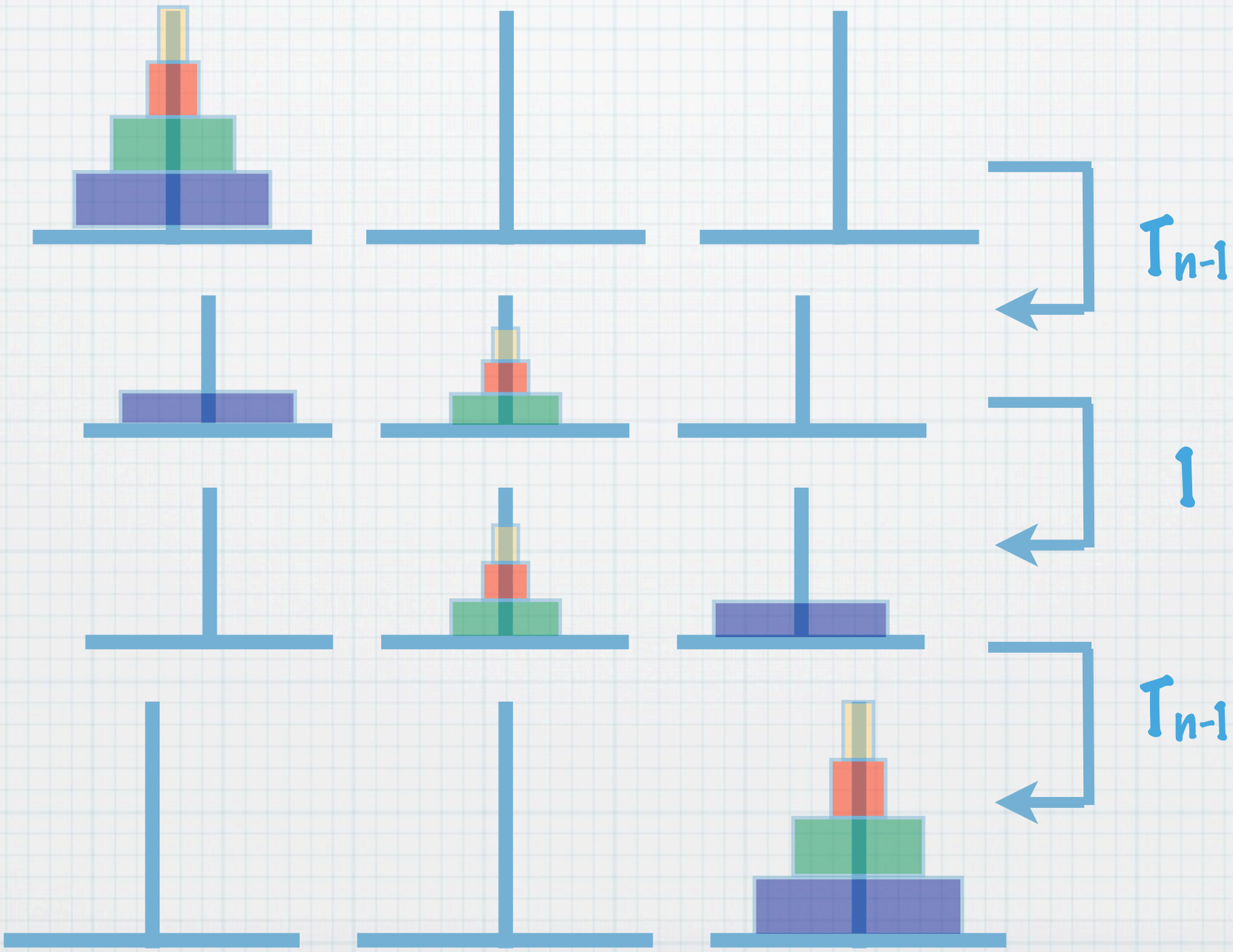
Tours de Hanoï

* On résout le problème par récurrence sur le nombre de disques en imaginant que l'on a résolu le problème pour n-1 disques

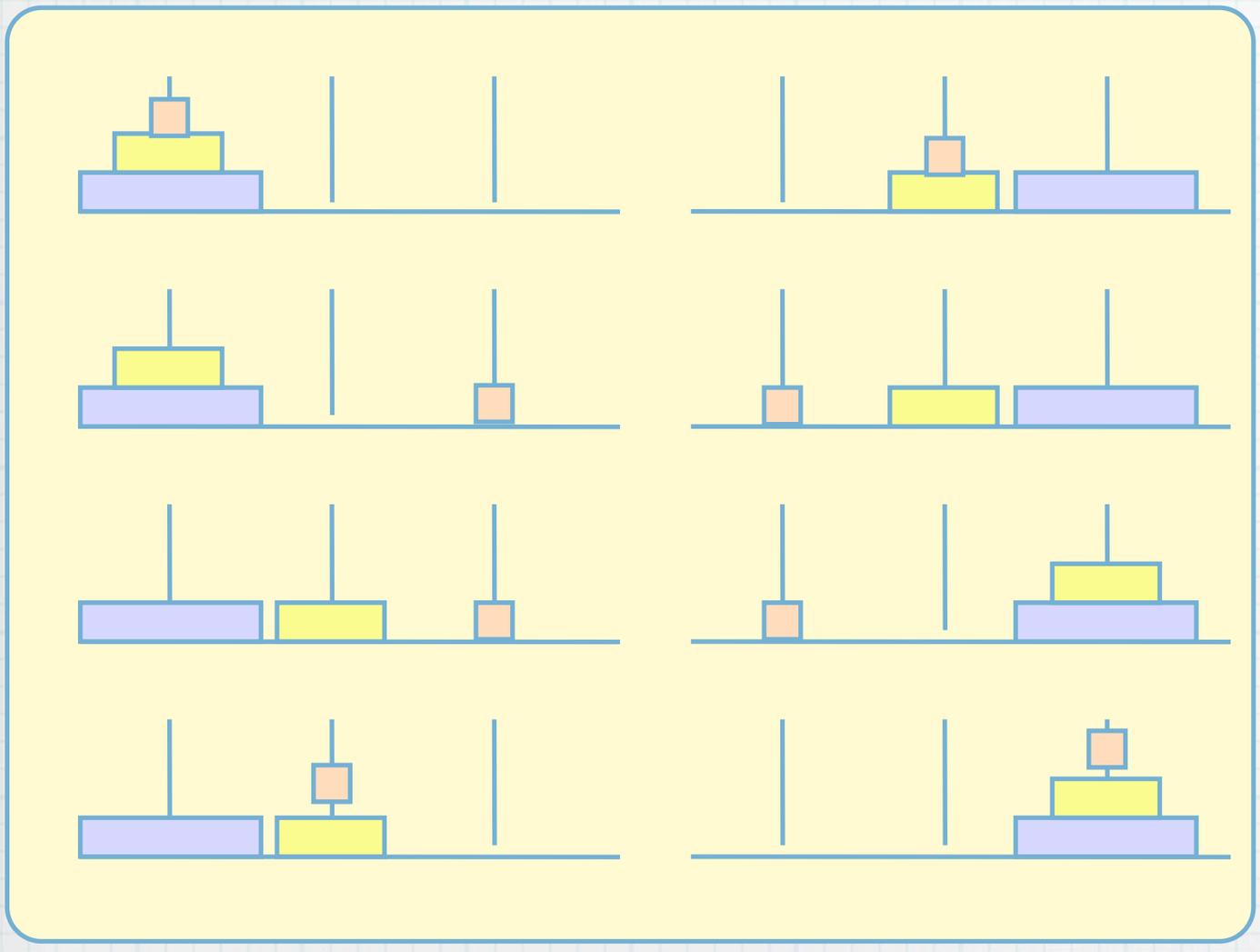


Tours de Hanoï

* Soit T_n le nombre de déplacements élémentaires



Tours de Hanoï : $T_0=0$ et $T_n=2T_{n-1}+1$



$$T_n = 2\,T_{n-1} + 1$$

$$2 \times T_{n-1} = (2\,T_{n-2} + 1) \times 2$$

$$2^2 \times T_{n-2} = (2\,T_{n-3} + 1) \times 2^2$$

...

$$2^{n-1} \times T_1 = (2\,T_0 + 1) \times 2^{n-1}$$

$$T_n = 2^n\,T_0 + \sum_{0 \le i \le n-1} 2^i$$

$$T_n = 2^n - 1$$

Equation de partition

* Une équation de récurrence est dite de partition si elle est de la forme

$$T(n) = f(\{T(n/p), p>0\})$$

* L'étude des algorithmes récursifs produit des équations de partition.
* Parmi eux, les **algorithmes dichotomiques** induisent des équations de partition avec p=2.
  * Pour les résoudre, on opère un changement de variable de sorte à retomber sur une équation linéaire.

Résolution d'une équation de partition

* Appelons T(n) le nombre de comparaisons nécessaires à la recherche dichotomique d'un élément dans un tableau trié de taille n.
  * On suppose $n=2^k$. $T(2) = 1$ et pour $n>1$, $T(n) = 1 + T(n/2)$
* On effectue un changement de variable. On passe de n à k.
  $T(2^1) = 1$ et pour $k>0$, $T(2^k) = 1 + T(2^{k-1})$
* On renomme la relation de récurrence concernant k :
  $S(1) = 1$ et pour $k>0$, $S(k) = 1 + S(k-1)$
* On résout cette équation. Ici, c'est une simple suite arithmétique :
  $S(k) = k$
* Il reste à opérer le changement de variable inverse :
  $T(n) = 1 + \log_2(n)$

Équations linéaires d'ordre 2

* Suite de Fibonacci, l'équation caractéristique a deux racines distinctes réelles
  $F_0 = F_1 = 1$, et $F_n = F_{n-1} + F_{n-2}$ pour $n > 1$
* Cas où les racines sont distinctes, mais complexes
  $G_n = G_{n-1} - G_{n-2}$ pour $n > 1$
* Cas où le polynome caractéristique a une seule racine double
  $H_n = 4H_{n-1} - 4H_{n-2}$ pour $n > 1$

Séries génératrices

* Plutôt que d'étudier une suite de nombres $(a_n)_{n\geq 0}$, on s'intéresse à la fonction définie par

$$x \longrightarrow \sum_{n\geq 0} a_n x^n$$

* Exemples
  * $(a_n)_{n\geq 0}$ est la suite constante et égale à 1: $1/(1-x)$
  * $(b_n)_{n\geq 0}$ est la suite arithmétique $(2n+1)$: $2x/(1-x)^2+1/(1-x)$
  * $(c_n)_{n\geq 0}$ est la suite géométrique (2^n) : $1/(1-2x)$
* Propriétés
  * Série génératrice de $(u_n + v_n)$
  * Série génératrice de (u_{n+1})
  * Série génératrice de (u_{n+2})
  * La suite ayant pour série génératrice $f(x)*g(x)$

Séries génératrices

* Toute récurrence sur la suite $(a_n)_{n\geq 0}$, se traduit alors par une relation fonctionnelle sur sa série génératrice

$$X \longrightarrow \sum_{n\geq 0} a_n X^n$$

* Exemples
  * Suite de Fibonacci : F(x)=1/(1-x-x^2)
  * Nombres de Catalan : C(x) =(1-$\sqrt{1-4x}$)/2x