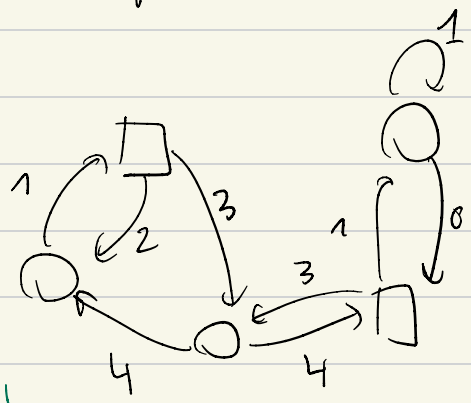
# IV. Jeux de parité.

## 1) Définition.

**Déf:** un **jeu de parité** est un jeu dont les arêtes sont étiquetées par des entiers qu'on appelle **priorités**, et dont l'objectif est "la plus grande priorité qui apparaît souvent est paire".

**Exemple:**




région gagnante pour Eve.

Remarque : les jeux de parité généralisent les jeux de Büchi (prio 1,2) et les jeux de co-Büchi (prio 0,1).

Au programme dans cette section.

1) Algorithme de Zielonka (exponentiel)
2) Positionalité et complexité $NP \cap coNP$
3) Algorithme d'itération de valeur (IV)
4) IV amélioré avec des arbres universels et algo quasi-polynomial.

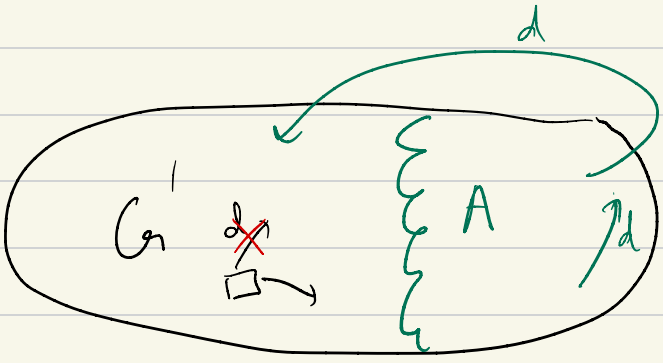
# 1) Algorithme de Zielonka (1998).

Idée : généraliser les techniques d'attracteurs vues par les jeux de Büchi aux jeux de parité.

Soit $d$ la plus grande priorité, on suppose que $d$ est paire.

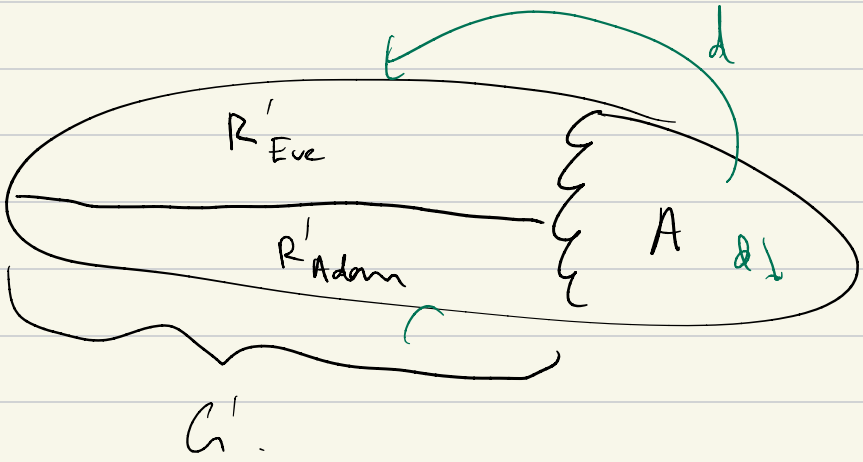
On définit $A = \mathrm{Attr}_0\left(\xrightarrow{d}\right)$, puis $G'$ comme étant le jeu obtenu en

- retirant $A$
- retirant les arêtes de priorité $d$.



La priorité maximale de $G'$ est $< d$, donc on peut le résoudre inductivement (base de l'induction : jeux de Büchi, déjà traités).

On appelle $R'_{Eve}$ la région gagnante pour Eve dans le jeu $G'$, idem pour $R'_{Adam}$.




1er cas : $R'_{Adam} = \emptyset$.

* positionnellement

Lemme 1: si $R'_{Adam}$ est vide, alors Eve gagne* le jeu depuis tous les sommets.

Preuve: On utise la stratégie positionnelle suivante:

- sur $G'$ : stratégie gagnante positionnelle obtenue par induction $\sigma'$
- sur $A$ : stratégie d'attracteur vers priorité $d$.

Montrons que c'est une stratégie gagnante. On considère un chemin consistent avec la stratégie.

Cas 1: le chemin va infiniment souvent dans $A$. Alors à chaque fois qu'on passe dans $A$, on voit priorité $d$. Donc la plus grande priorité vue infiniment souvent est $d$

qui est paire. Donc le chemin satisfait l'objectif.

Cas 2 : on passe un nombre fini de fois dans $A$. Alors on a un suffixe qui reste dans $G'$ et qui est conforme avec la stratégie $\sigma'$. Comme l'objectif de parité ignore les préfixes finis, le chemin est gagnant.

Lemme 2 : Adam gagne le jeu de parité depuis tous les sommets de $R'_{Adam}$.

Preuve : c'est vrai car $R'_{Adam}$ est un piège pour Eve.

On en déduit l'algorithme de Zielonka, qui est un algorithme récursif :

① Calculer $A = \mathrm{Attr}^{G}_{\mathrm{Eve}}\left(\xrightarrow{d}\right)$

② Résoudre récursivement $G_1'$, qui est obtenu à partir de $G_1$ en retirant $A$ et les arêtes de priorité $d$

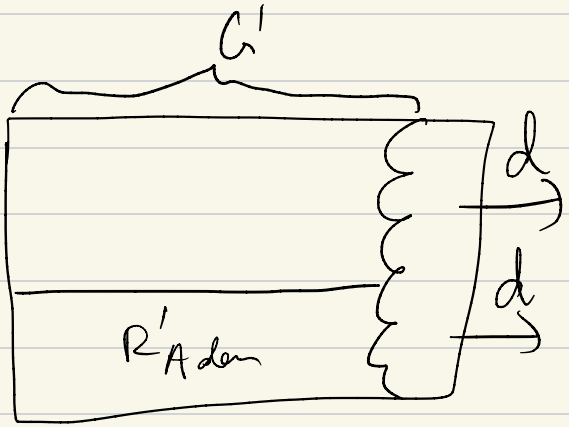
(moins de prio. que $G_1$)

③ Si $R'_{\mathrm{Adam}}$ la région gagnante d'Adam, est vide, alors tout est gagnant pour Eve et on s'arrête

④ Sinon, calculer l'attracteur $B$ pour Adam vers $R'_{\mathrm{Adam}}$, et résoudre récursivement $G_1 \setminus B$.

(plus petit que $G_1$)

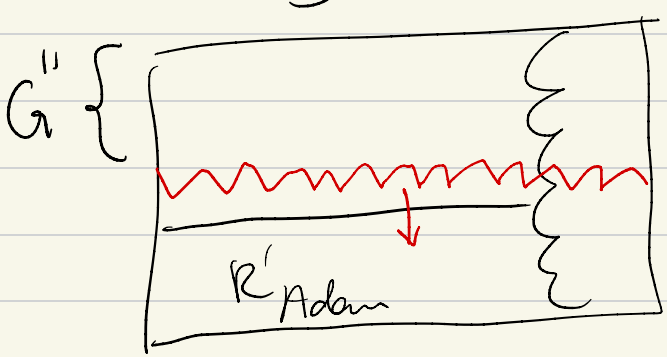
# Schématiquement :




$R'_{Adam} = \phi$

tout est gagnant pour Eve

$R'_{Adam} \neq \phi$




on résout récursivement G''.

Clairement Adam gagne depuis B. Il reste à montrer le lemme suivant.

Lemme : • un sommet gagnant pour Eve dans $G''$ est aussi gagnant pour Eve dans $G'$.
• idem pour Adam.

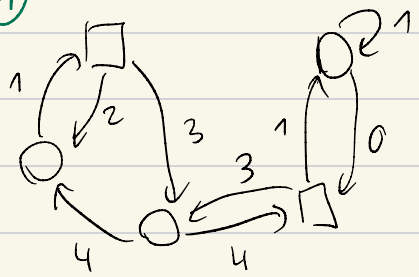
Preuve : • Pour Eve c'est facile, c'est vrai simplement car $G''$ est un piège pour Adam (complémentaire d'un attracteur)

• Pour Adam, on suit la stratégie dans $G''$, et
→ si on reste dans $G''$, c'est bon
→ si on s'échappe, alors on rejoint B qui est aussi gagnant pour Adam, donc c'est bon aussi.

Corollaire : les jeux de parité sont positionnels (pour les deux joueurs).

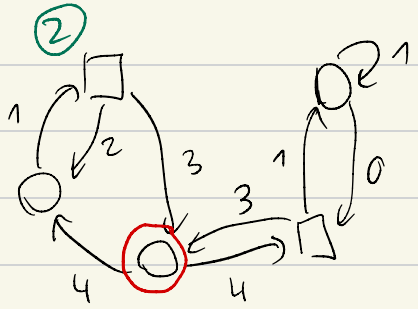
Sur un exemple :

1



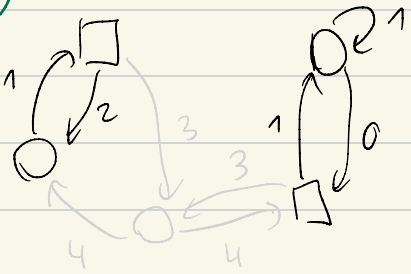
Jeu initial

2



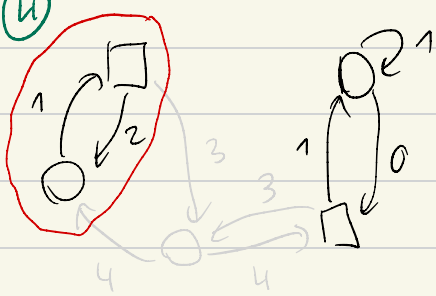
Attracteur pour Eve vers $\xrightarrow{4}$.

3



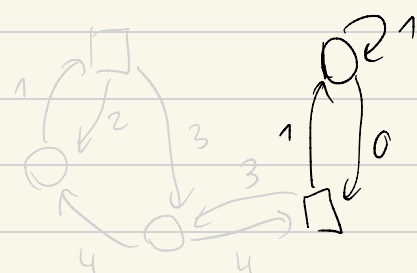
On retire A, la plus grande prio est maintenant 2.

4



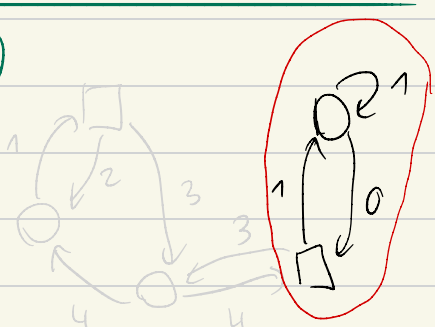
Attracteur pour Eve vers $\xrightarrow{2}$.

5



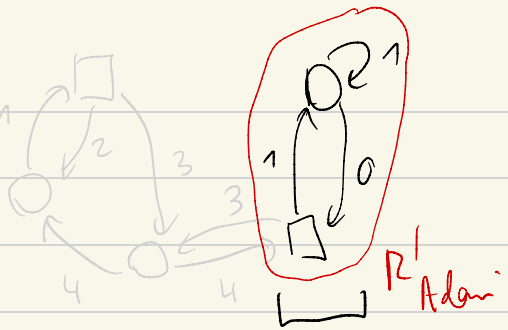
On retire A, plus grande prio 1.

6



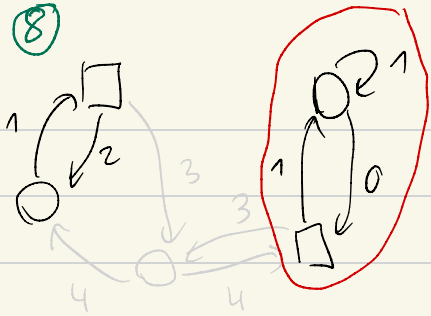
Attracteur pour Adam vers $\xrightarrow{1}$.

(7)



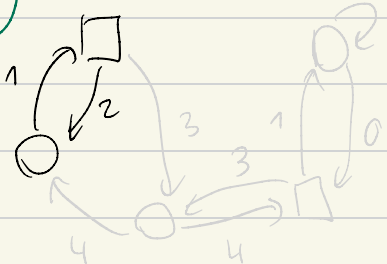

Donc Adam gagne ce sous-jeu partout.

(8)



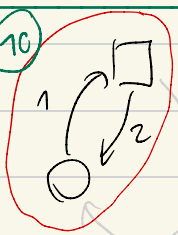
Attracteur pour Adam vers $R'_{Adam}$.

(9)



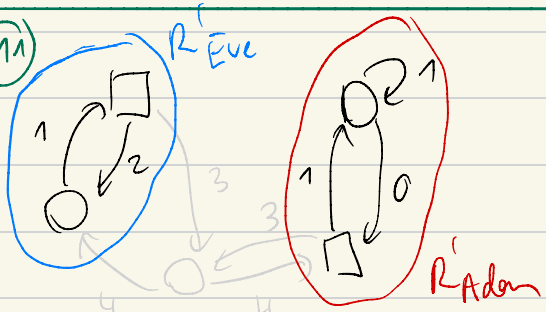
On le retire du jeu, puis on résout à nouveau.

(10)



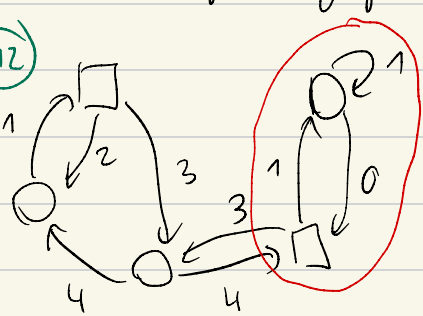
Attracteur pour Eve vers 2. Il n'y a rien d'autre, donc c'est la région gagnante

(11)



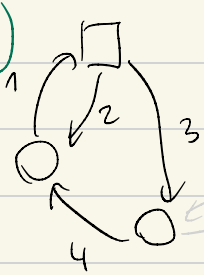

On conclut pour ce sous-jeu, et on dépile l'appel récursif.

(12)



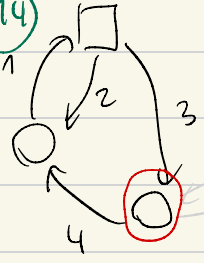
Attracteur pour Adam vers $R'_{Adam}$.

13



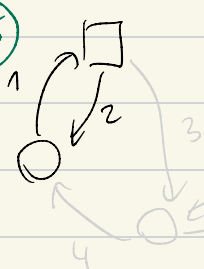

On le retire du jeu, max prio = 4.

14



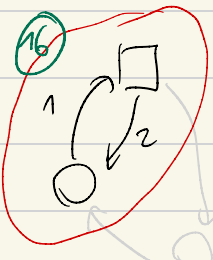

Attracteur pour Eve vers 4.

15



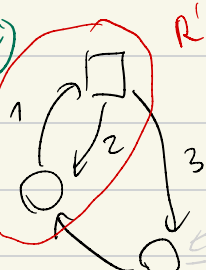

On le retire, mais prio = 2.

16



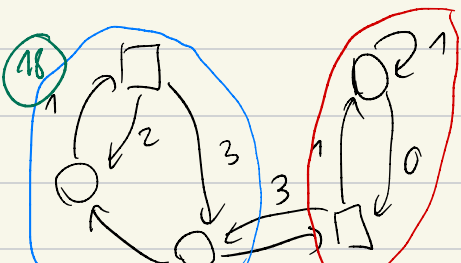

Attracteur pour Eve vers 2, et donc aussi région gagnante.

17




Comme $R'_{Adam} = \emptyset$, Eve gagne partout.

18




Conclusion.

Complexité :

$$f(n,d) \leq O(m) + f(n,d-1) + f(n-1,d)$$

→ borne exponentielle $O(m \cdot (n+d)^{n+d})$.

Exercice : trouver un jeu qui prend un nombre exponentiel d'étapes.

## 2) Complexité des jeux de parité.

PARITY-GAME :

| INPUT : jeu de parité, sommet $v_0$.
| OUTPUT : Eve gagne-t-elle depuis $v_0$ ?

Quelle est la complexité de ce problème ?

**Théorème :** PARITY-GAME $\in$ NP $\cap$ co NP.

**Preuve :** Montrons que c'est dans NP.
On a vu (grâce à l'algo de Zielonka) que c'est positionnel. Cela donne un certificat : une stratégie positionnelle gagnante depuis $v_0$.

Pour vérifier le certificat, il faut déterminer si dans le graphe obtenu une fois que la stratégie est fixée, tous les cycles ont priorité maximale paire. On a donc le problème

EVEN-CYCLES :
| INPUT : graphe avec des priorités.
| OUTPUT : est-ce que sur chaque cycle, la priorité maximale est paire ?

On admet (exercice) que EVEN-CYCLES $\in$ P, et donc PARITY-GAME $\in$ NP.

On procède de la même manière avec le pt de vue d'Adam pour montrer

PARITY-GAME $\in$ co NP. $\square$

Historiquement, on trouve généralement des algos polynomiaux pour les pbs dans NP $\cap$ co NP (factorisation, programmation linéaire, isomorphisme de graphes ...). D'ailleurs certains experts pensent que P = NP $\cap$ co NP.

Malgrés tout ça, on ne connaît pas d'algo polynomial pour les jeux de parité. C'est un domaine de recherche très actif depuis les années 1990.

Depuis 2017, on dispose d'un algo quasi-polynomial en $O(n^{\log d})$ par Calude, Jain, Khoussainov, Li et Stephan.

# 3) Algorithme d'itération de valeurs.

**Idée :** minimiser le nombre de priorités impaires rencontrées avant une priorité plus importante.

**Def :** étant donnée une suite de priorités, on définit sa **signature** comme le vecteur

$$\begin{pmatrix} \text{nombre de } d-1 \text{ avant un } d \\ \text{---} \quad d-3 \quad \text{---} \quad \geq d-2 \\ \vdots \\ \text{nombre de } 1 \text{ avant un } \geq 2 \end{pmatrix}.$$

**Exemple :** avec $d = 6$.

- 1 2 1 1 2 1 3 1 3 3 2 1 4 5 3 1 6 2 3 ---

↳ signature $\begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix}$.

- 3 2 3 2 3 2 3 2 3 2 ...

↳ signature pas définie.

Lemme: pour une suite de priorités qui satisfait parité (la plus grande prio vue infiniment souvent est paire), la signature est bien définie.

Preuve: dans une suite de priorités qui satisfait parité, il existe une position $i_0$ telle que:

- on voit prio $p$ paire à la position $i_0$
- toutes les positions $\geq i_0$ ont prio $\leq p$.

Alors pour toute prio impaire, on la voit au plus $i_0$ fois avant une prio plus grande, et donc la signature est

bien définie. □.

On peut ordonner les signatures :

plus petit ⟷ meilleure pour Eve.

On considère que les 3 sont "largement pires" que les 1, que les 5 sont "largement pires" que les 3, etc.
Cela correspond à l'ordre lexicographique.

Exemple: $\begin{pmatrix} 3 \\ 1 \\ 4 \end{pmatrix} \leq \begin{pmatrix} 4 \\ 0 \\ 0 \end{pmatrix} \leq \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix}$.

Déf: étant donnée une signature $\sigma$ et une priorité $p$, on définit la $p$-troncature de $\sigma$ comme étant la signature obtenue en retirant les coordonnées correspondant à des prio $< p$.

Exemples : · 2-troncature de $\begin{pmatrix} 5 \\ 1 \\ 0 \end{pmatrix}^{5}_{3}_{1} = \begin{pmatrix} 5 \\ 1 \end{pmatrix}^{5}_{3}$

· 5-troncature de $\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}^{7}_{5}_{3}_{1} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}^{7}_{5}$.

On pose ensuite

$\sigma \leq_p \sigma' \iff$ p-troncature de $\sigma \leq$ p-troncature de $\sigma'$.

Idem pour $<_p$ et $=_p$.

Exemples : · $\begin{pmatrix} 1 \\ 0 \\ 4 \end{pmatrix}^{5}_{3}_{1} =_2 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$

· $\begin{pmatrix} 0 \\ 1 \\ 4 \end{pmatrix} =_5 \begin{pmatrix} 0 \\ 18 \\ 0 \end{pmatrix} <_5 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$.

Def : Une mesure dans un jeu ou dans un graphe, et une fonction $\rho$ qui assigne une signature à chaque sommet.

**Déf :** Fixons une mesure $\varphi$. Une arête $v \xrightarrow{p} v'$ est *valide* si

- $p$ est paire et $\varphi(v) \geq_p \varphi(v')$ ou
- $p$ est impaire et $\varphi(v) >_p \varphi(v')$.

**Exemples :**

- $\begin{pmatrix} 2 \\ 1 \\ 4 \end{pmatrix}^{5}_{3}_{1} \xrightarrow{4} \begin{pmatrix} 2 \\ 12 \\ 3 \end{pmatrix}^{5}_{3}_{1}$ valide

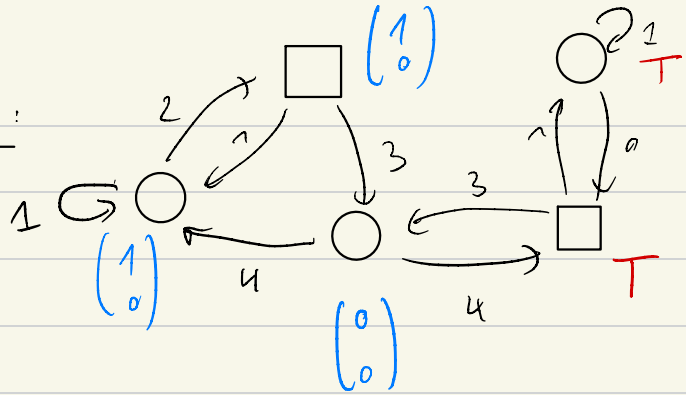
- $\begin{pmatrix} 0 \\ 1 \\ 3 \end{pmatrix}^{5}_{3}_{1} \xrightarrow{3} \begin{pmatrix} 0 \\ 1 \\ 8 \end{pmatrix}^{5}_{3}_{1}$ pas valide.

**Déf :** On dit qu'un sommet est *valide* si :

- c'est un sommet d'Eve avec une arête sortante valide, ou
- c'est un sommet d'Adam dont toutes les arêtes sortantes sont valides.

Une mesure est *valide* si tous les sommets le sont.
Dans le cas d'un graphe, on considère que tous les sommets appartiennent à Adam.

Exemple :




Remarque : par certains sommets (en fait, ce sont les sommets gagnants pour Adam), pas possible de trouver une mesure valide !

## Algorithme d'itération de valeurs :

↳ partir de $\begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$ et progressivement mettre à jour jusqu'à obtenir une mesure valide.

Def : une $n$-mesure est une mesure telle que toutes les coordonnées sont $\leq n$.

**Théorème :** (1) Soit $G_1$ un graphe avec des priorités, de taille $n$. Alors

$G_1$ satisfait parité $\iff$ il existe une $n$-mesure valide.

(2) Soit $G_1$ un jeu de parité de taille $n$. Alors

Eve gagne depuis tous les sommets $\iff$ il existe une $n$-mesure valide.

**Preuve :** on commence par (1).

$\Rightarrow$ : Soit $G_1$ un graphe de taille $n$ qui satisfait parité. Pour chaque sommet, on pose

$\ell(v) =$ signature du chemin $\Pi_v$ depuis $v$ qui a la plus grande signature.

C'est bien défini car $\Pi_v$ satisfait parité. Pour montrer que $\ell$ est valide, il faut vérifier que toutes les arêtes sont valides. Prenons une arête $v \xrightarrow{p} v'$.

Alors $v \xrightarrow{p} \Pi_{v'}$ définit un chemin, donc par définition

$$\ell(v) \geqslant \text{signature de } v \xrightarrow{p} \Pi_{v'}.$$

Si $p$ est impaire, on a

signature de $v \xrightarrow{p} \Pi_{v'} >_p$ signature de $\Pi_{v'} = \ell(v')$

de même si $p$ est paire, on a

signature de $v \xrightarrow{p} \Pi_{v'} \geqslant_p$ signature de $\Pi_{v'} = \ell(v')$

On en déduit dans les deux cas que l'arête $v \xrightarrow{p} v'$ est bien valide.

Il reste à montrer que $\ell$ est bien une $n$-mesure. Supposons par l'absurde que pour un certain $v$, $\ell(v)$ a une coordonnée $> n$, c'est-à-dire que sur le chemin $\pi_v$, plus de $n$ occurrences d'une prio impaire $p$ sont rencontrées avant une prio plus grande.

On en déduit l'existence d'un cycle dont la priorité maximale est $p$, et donc le graphe ne satisfait pas parité, une contradiction.

$\Leftarrow$ : Supposons qu'il existe une mesure valide $\ell$. On montre que $G$ satisfait parité. Prenons un chemin dans $G$

$$v_0 \xrightarrow{p_1} v_1 \xrightarrow{p_2} v_2 \xrightarrow{p_3} \dots,$$

soit $p$ la plus grande priorité qui apparaît infiniment souvent.

Comme toutes les arêtes sont valides, on a pour tout $i$ :

- $p_i$ paire $\Rightarrow \varphi(v_i) \geqslant_{p_i} \sigma(v_{i+1})$
- $p_i$ impaire $\Rightarrow \varphi(v_i) >_{p_i} \sigma(v_{i+1})$.

Supposons par l'absurde que $p$ est impaire, et prenons $i_0$ tel que pour tout $i \geqslant i_0$ on a $p_i \leq p$.

Observons que pour $p_i < p$, on a :

- $\varphi(v_i) \geqslant_{p_i} \varphi(v_{i+1}) \Rightarrow \varphi(v_i) \geqslant_p \varphi(v_{i+1})$
- $\varphi(v_i) >_{p_i} \varphi(v_{i+1}) \Rightarrow \varphi(v_i) \geqslant_p \varphi(v_{i+1})$.

On a donc, pour tout $i \geqslant i_0$,

$$\varphi(v_i) \geqslant_p \varphi(v_{i+1}),$$

et infiniment souvent (dès que $p_i = p$),

$$\varphi(v_i) > \varphi(v_{i+1}).$$

Cela n'est pas possible.

On prouve maintenant (2). Soit $G_1$ un jeu de parité de taille $n$.

$\Rightarrow$ : Supposons que Eve gagne depuis tous les sommets. On prend une stratégie positionnelle gagnante $\tau$. On applique (1) au graphe $G_{1,\tau}$ de $\tau$, ce qui donne une mesure $\mu$ $n$-valide pour $G_{1,\tau}$. Alors la mesure $\mu$ est aussi $n$-valide pour $G_1$.

$\Leftarrow$ : Réciproquement, supposons qu'il existe une mesure valide $\mu$ pour $G_1$. Par chaque sommet d'Eve, on a une arête sortante valide : cela donne une stratégie positionnelle $\tau$ telle que $\mu$ est une mesure valide pour $G_{1,\tau}$, et donc par (1) $G_{1,\tau}$ satisfait parité. On conclut que $\tau$ est une stratégie gagnante. □

Pour gérer les sommets gagnants pour Adam, on rajoute la signature $\top$. Une arête $v \xrightarrow{P} v'$ telle que $\ell(v') = \top$ n'est valide que si $\ell(v) = \top$.

Lemme : soit $G$ un jeu et $\ell$ une mesure valide. Alors l'ensemble des sommets qui ne sont pas envoyés sur $\top$ est un piège pour Adam.

Preuve : soit $R$ cet ensemble. Les arêtes de $R$ vers $^cR$ ne sont pas valides. Comme les sommets d'Ève ont une arête sortante valide et que les sommets d'Adam ont toutes leurs arêtes sortantes valides, $R$ est bien un piège pour Adam.

## Algorithme:

- $\varphi(v) \leftarrow 0$ pour tout $v$.
- Tant que $\varphi$ n'est pas valide :
  - Prendre un sommet $v$ pas valide
  - $\varphi(v) \leftarrow \mathrm{Incr}_n(\varphi(v))$.
- Return $^c\varphi^*(T)$.

Où $\mathrm{Incr}_n(s)$ est la plus petite $n$-signature plus grande que $s$.

**Théorème:** l'algorithme retourne la région gagnante pour Eve.

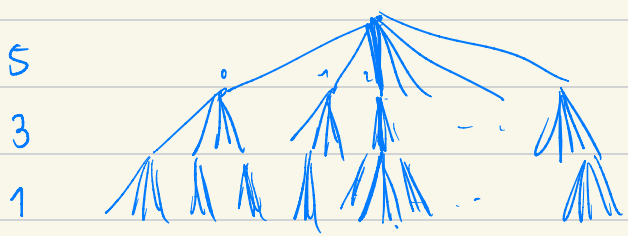
**Preuve:** la mesure calculée ne peut jamais excéder la plus petite

mesure valide. On conclut grâce au Thm précédent.

Complexité : $O(\underbrace{nm}_{\text{calculs validité}} \cdot \underbrace{n.n^{d/2}}_{\text{nombre d'incrémentations}})$

## 4) Arbres universels et algo QP.

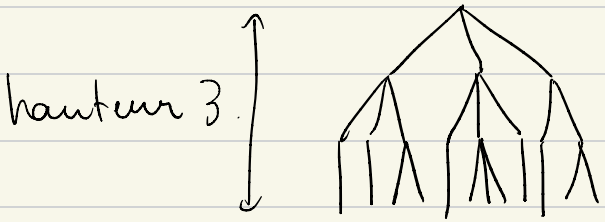
Idée : signature = feuille dans l'arbre complet




$n^{d/2}$ feuilles

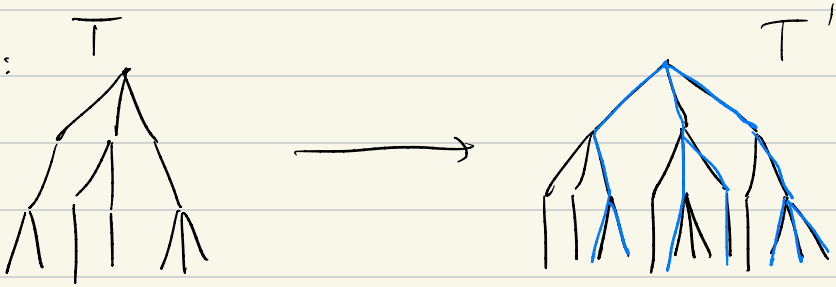
Peut-on retirer des feuilles et garder un algo correct ?

**Def :** arbre de taille $n$ et hauteur $h$, par l'exemple :




**Def :** on dit que $T$ se plonge dans $T'$ si on peut obtenir $T$ en supprimant des feuilles de $T'$. On note $T \longrightarrow T'$.

**Exemple :**




**Def :** un arbre $T$ est $(n,h)$-universel si tous les arbres à $n$ feuilles de hauteur $h$ se plongent dans $T$.

Exemples : • arbre $n$-complet et universel.

•  et $(3,2)$-universel,

car on y plonge 

**Théorème :** Étant donné un arbre $(n, d/2)$-universel, on peut résoudre les jeux de parité en temps

$$O\left( nm \cdot i(n, d/2) \cdot n \cdot t(n, d/2) \right)$$

- $nm$ : calcul de validité — polynomial
- $i(n, d/2)$ : temps de calcul d'un incrément — « polynomial
- $t(n, d/2)$ : taille de l'arbre — ???

**Preuve :** On applique l'algo précédent en remplaçant les signatures par des feuilles de l'arbre. Ça marche car

un arbre correspondant aux signatures d'une stratégie gagnante se plonge par universalité. □

**Théorème** : (1) il existe un arbre $(n,h)$-universel de taille

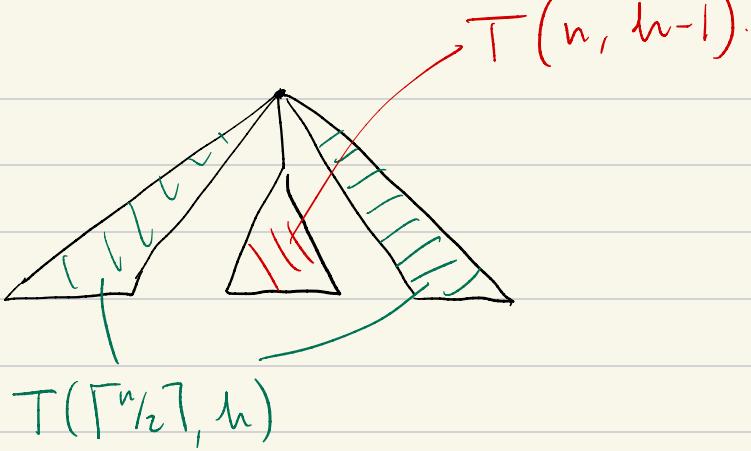
$$n \cdot \binom{h+\log n}{\log n}$$

(2) tout arbre $(n,h)$-universel a taille au moins $\binom{h+\log n}{\log n}$.

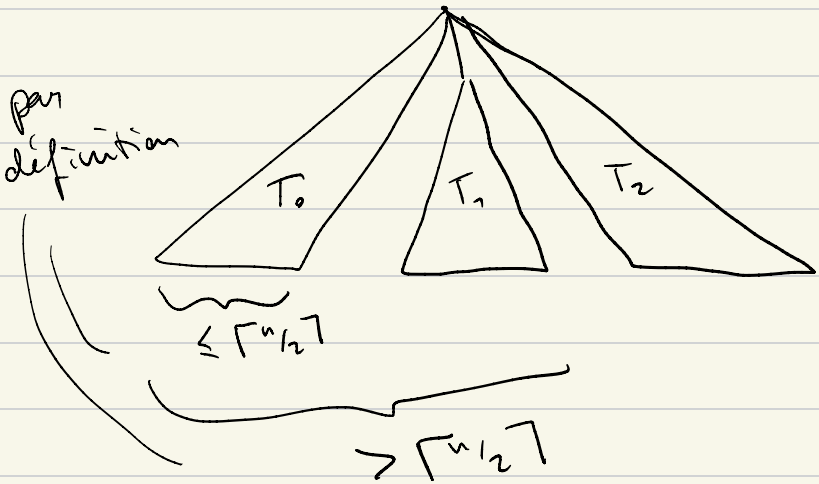
**Remarque** : c'est un $O(n^{2\log n})$.

**Preuve** : on prouve (1). Soit $T(n,h)$ l'arbre défini inductivement par $T(n,1) = \cdot$ et

$T(n,h) =$




On prouve qu'il est $(n,h)$-universel.
Soit $T$ un arbre de taille $\leq n$ et hauteur $h$.
On considère le premier enfant de $T$ tel que il y a plus de $\lceil n/2 \rceil$ feuilles avant cet enfant :



Donc $T_0$ et $T_2$ ont taille $\leq \lceil n/2 \rceil$ et hauteur $h$, et $T_1$ a taille $\leq n$ et hauteur $h-1$. Par induction,

$$T_0, T_2 \longrightarrow T(\lceil n/2 \rceil, h)$$
$$T_1 \longrightarrow T(n, h-1),$$

et donc $T \longrightarrow T(n, h)$, d'où l'universalité.

Par l'analyse de la taille, on pose $k = \log n$, puis il suffit de vérifier que

$$2^k \binom{h+k}{k} \leq 2 \cdot 2^{k-1} \binom{h+k-1}{k-1} + 2^k \binom{h+k-1}{k}$$

ce qui nous est donné par la formule du triangle de Pascal.