1 Dynamique

On a deux type d'element, l'element vide ϕ et des element non vide que l'on representera comme des entier positif $u \in \mathbb{N}$.
Les regles d'arithmetique de nom pour les element non-vide sont:

$$u \wedge v = [u\ v]$$

$$[u\ v]\,.r = v$$

$$[u\ v]\,.l = u$$

Pour un element non-vide et un element vide on a:

$$u \wedge \phi = u$$

$$u.l = u$$

$$u.r = \phi$$

Et entre les element vides on a:

$$\phi.l = \phi$$

$$\phi.r = [\phi]$$

$$\phi \wedge [\phi] = \phi$$

Qui se generalisent pour $\phi_n = [\phi_{n-1}]$ et $\phi_0 = \phi$:

$$\phi_n.l = \phi_n$$

$$\phi_n.r = \phi_{n+1}$$

$$\phi_n \wedge \phi_{n+1} = \phi_n$$

Pour deux noeud x et y qui ne sont pas dans les regles precedente, on suit les même regles que pour deux element non-vides:

$$x \wedge y = [x\ y]$$

$$[x\ y]\,.r = y$$

$$[x\ y]\,.l = x$$

2 Representation

On represente un noeud par trois listes:

- *names* qui represente le nom de chaque element (on pose -1 $\Leftrightarrow$ [ ])
- *bracket_depth* qui represente le nombre de brackets entourant l'element
- *immediatly_left_bracket* qui represente le nombre de bracket "[" imediatement à gauche de l'element

La dynamique peut alors suivre ces regles:

- On cherche d'abord l'indice separant x.l de x.r. Pour cela on cherche l'indice auquel

$$bracket\_depth = immediatly\_left\_bracket + 1$$

- si cette indice est bien strictement dans les limite de la liste, on "deccoupe" alors le noeud en deux (on decremente *immediatly_left_bracket* [0] puis chaque *bracket_depth*)
- si l'indice est à la limite de la liste, on verifie si la liste est un unique element vide, dans ce case x.l = x et x.r = [x] ce que l'on obtient en incrementant *immediatly_left_bracket* et *bracket_depth*.
- Sinon, x.l = x et on retourne x.r = [ ]

Il est ensuite facile d'inverser ces regle pour implementer la fusion de noeud.
Exemple de representation d'un noeud x :

$$\begin{array}{c|ccccc} names & 0 & 1 & 2 & -1 & 3 \\ bracket\_depth & 2 & 2 & 3 & 3 & 2 \\ immediatly\_left\_bracket & 2 & 0 & 2 & 0 & 0 \\ & \underbrace{[[0 \quad 1]}_{x.l} & & \underbrace{[[2 \quad [\,]] \quad 3]]}_{x.r} & & \end{array}$$

3 Ensemble de noeud

Un graph est nommé par une liste de neud $[x_0\ x_1\ ...\ x_n]$.

On initilaisera toujour un graph par $x_i = i$, donc un graph commence par un nom $[0\ 1\ ...\ n]$.

Pour comparer les nom de deux graphs, il faudrait comparer toute les rotation relative possible des les deux liste de graphs. Pour eviter ce probleme, on garde en memoire la variable $zero\_index$ qui est l'index du noeud du graph qui contient l'element 0.

Soit deux graphs nommés $[x_0\ x_1\ ...\ x_n]$ et $[y_0\ y_1\ ...\ y_n]$ qui ont leurs $zero\_index$ notés $zero\_index_x$ et $zero\_index_y$, Pour que ces deux graphs porte le même nom, il faut que $x_{zero\_index_x} == y_{zero\_index_y}$.

On peut alors comparer une seul rotation relative des graphs, en comparant tout les noeud x_i et y_j avec:

$$i - j = offset = zero\_index_x - zero\_index_y$$