Modélisation

Thomas

$Q = 2^D$

Soit un mot $v_1 \dots v_D$, tq "v_i ~~x~~ boules atterrissent à l'instant i."

$v_1 \dots v_d \in Q$.

Si $v_1 = 0$, on ajoute trans° $q \xrightarrow{0} q'$, $q' = v_2 \dots v_{d-1} 0$. (on n'a pas reçu de boule)

Si $v_1 = 1$, on a reçu une boule et il faut la relancer.

↳ $q \xrightarrow{j} q'$, ~~q~~ si $v_{j+1} = 0$, $q' = v_2 \dots \overset{\text{en } j}{1} \dots v_{d-1}$

Temps terminé !!.

[k + jongl[k]] % longueur for k in range(longueur)]

Hector

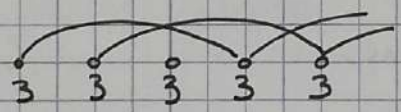
0/ Plan

I/ Modèle
II/ Caractérisation
III/ Enumération

I/ Modèle

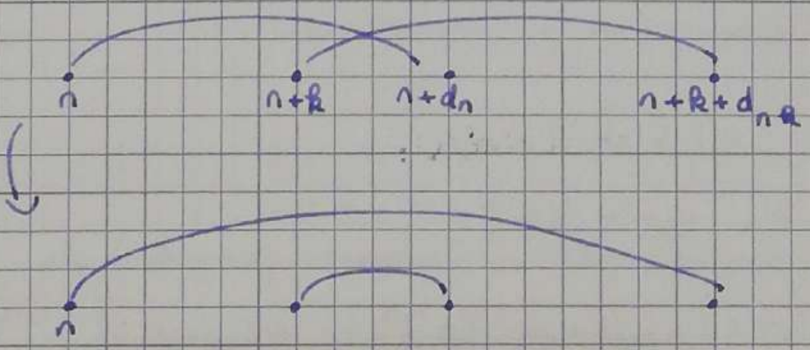
- Discrétisation du problème.
  $d \in \mathbb{N}$ = tps passé en l'air
  $p \in \mathbb{N}$ = période

Ex: 33 →



⇒ les mots représentent ce qu'il faut jongler.

(On veut, étant donné un mot, calculer le mot représentant l'état suivant)



i.e on s'autorise à permuter pour démêler

II / Caractérisation

On veut déterminer si un mot est jonglable

Test de la moyenne

Soit $d_0 \dots d_{p-1}$.

Condition nécessaire : $\frac{1}{p} \sum d_i = b$, b le nombre de balles

Test des permutations

"Sur une période donnée, il y a pas 2 balles qui arrivent au m̂ endroit au m̂ moment."

Soit $a_0 \dots a_{p-1}$ un mot jonglable.
S'il y a pas de collisions sur $a_k + k \, [p]$, on est injectif.
Par cardinalité, on est bijectif.
Donc il n'y a pas de collision.

Test de réécriture

ssi on peut le reconstruire à partir de transitions adjacentes à partir de b^p, i.e le motif où tout est entrelacé.

Présentation de code Python appliquant le test de permutation
Avec des tests !

III / Enumeration

A) b et p fixés

Enumération naïve → on énumère tout et on teste pour éliminer les mots non jonglables (?)

Enumération par de permutations → on énumère les permutations en éliminant les redondances

Enumération par transitions

B) b et d fixés

On suppose $d \geq b$.
On veut créer un automate permettant de lire les mots possibles.

$Q = \{ n \text{ mots de } \{0,1\}^d \text{ tq } \#C_1(n) = b \}$