# TD : HACHAGE – RUSH HOUR

$$\text{hash}(e) = \sum_{\substack{0 \le i < M \\ 0 \le j < M}} d_{i,j} 3^{iM+j}$$

… où pour tout i, j $\in$ [0,M[ :

$$d_{i,j} = \begin{cases} 0 & \text{si la case (i,j) est inocupée} \\ 1 & \text{si la case (i,j) est occupée par un véhicule horizontal} \\ 2 & \text{si la case (i,j) est occupée par un véhicule vertical} \end{cases}$$

**3. Montrer que si $e_1$ et $e_2$ sont deux états accessibles à partir d'un même état initial e et que hash($e_1$) = hash($e_2$), alors $e_1 = e_2$.**

Supposons que hash(e1) = hash(e2) :

$$\text{hash}(e_1) = \sum_{\substack{0 \le i < M \\ 0 \le j < M}} d^{(1)}{}_{i,j} 3^{iM+j} \quad \text{et } \text{hash}(e_2) = \sum_{\substack{0 \le i < M \\ 0 \le j < M}} d^{(2)}{}_{i,j} 3^{iM+j}$$

… avec $d^{(1)}{}_{i,j}, d^{(2)}{}_{i,j} \in \{0,1,2\}$

Alors on a :

$$\sum_{\substack{0 \le i < M \\ 0 \le j < M}} \left(d^{(1)}{}_{i,j} - d^{(2)}{}_{i,j}\right) 3^{iM+j} = 0$$

Cette somme n'est rien d'autre que l'égalité de deux écritures en base 3 où chaque $d_{i,j}$ est un « chiffre » en base 3 (0, 1 ou 2) dont la position est donnée par l'indice (i·M + j) $\in$ {0, 1, 2, …, M, M+1, M+2, …, ($M^2$-1)}.

Or si un entier a deux écritures en base 3 : $\sum a_p 3^p = \sum b_p 3^p$ , alors forcément $a_p = b_p$ pour tout p et donc :

$$\forall i, j, \quad d^{(1)}{}_{i,j} = d^{(2)}{}_{i,j}$$

Les deux états $e_1$ et $e_2$ ont donc exactement le même motif de cases vides / horizontales / verticales. Les véhicules associés à ces deux états ont donc les mêmes positions, mêmes longueurs, et mêmes directions et donc $e_1 = e_2$.

8. **Expliquer en quoi ce serait une mauvaise idée de remplacer le $3^{iM+j}$ par $4^{iM+j}$ dans la définition de hash(e).**

*Première idée : Notion de débordement :*

Si M=6, avec la base 3, on a des « chiffres » $d_{i,j} \in \{0, 1, 2\}$, donc on utilise une base 3 sans débordement dans un `int64` car pour M=6, on a au plus $1 \cdot 3^0 + 3 \cdot 3^1 + \ldots + 3 \cdot 3^{35} < (3^{36} - 1) < 2^{63}-1$.

Si on remplace $3^{iM+j}$ par $4^{iM+j}$ alors pour M=6 on monte à $1 \cdot 4^0 + 3 \cdot 4^1 + \ldots + 3 \cdot 4^{35} > 3 \cdot 4^{35} > 2^{71}$ ce qui dépasse largement la capacité d'un entier signé 64 bits (environ $2^{63}$). On provoquerait donc des débordements arithmétiques et des collisions de hash très nombreuses, ce qui détruirait l'argument d'injectivité.

*Deuxième idée : certains termes valent 0 modulo $2^p$.*

Dans la table de hachage, on ne garde que h = hash(e) mod $2^p$.

Si on utilise la base 4, on a $4^{iM+j} = 2^{2(iM+j)}$ et $h(e) = \sum_{\substack{0 \le i < M \\ 0 \le j < M}} d_{i,j} 2^{2(iM+j)} \mod 2^p$

Ici, dès que $2\cdot(iM+j) \ge p$, on a $4^{iM+j} \equiv 0 \pmod{2^p}$. Cela entraine que toutes les contributions des termes avec $(iM+j) \ge [p/2]$ disparaissent dans le calcul de hash(e)%$2^p$. Autrement dit, le reste modulo $2^p$ ne dépend plus du tout de ce qui se passe sur « les cases hautes » (celles correspondant aux grands $i\cdot M+j$). Deux états qui diffèrent uniquement sur ces cases-là auront le même indice dans la table de hachage, même si on évitait l'overflow avec des entiers arbitraires.

Exemple : grille 2×2, base 4, capacité 4

On a M=2 et p = 2 :

$$\text{hash}(e) = d_{0,0}\cdot 4^0 + d_{0,1}\cdot 4^1 + d_{1,0}\cdot 4^2 + d_{1,1}\cdot 4^3$$

$$h(e) = (d_{0,0}\cdot 4^0 + d_{0,1}\cdot 4^1 + d_{1,0}\cdot 4^2 + d_{1,1}\cdot 4^3) \bmod 4$$

$$h(e) = (d_{0,0}\cdot 4^0 \bmod 8 + d_{0,1}\cdot 4^1 \bmod 8 + d_{1,0}\cdot 4^2 \bmod 8 + d_{1,1}\cdot 4^3 \bmod 8) \bmod 4$$

$$h(e) \equiv d_{0,0}\cdot 4^0 + 0\cdot 4^1 + 0\cdot 4^2 + 0\cdot 4^3 \pmod 4$$

L'indice de la table ne dépend donc que de la case (0,0).