

# Présentation projet de recherche

Par Joffrey Hauw, Erwann Le Juif, Samuel Mauger, Hilan Meyran et William Pignon  
Sous la supervision de Dr. Patrick Derbez

# Sommaire

- Objectifs de notre projet
- Les deux grands types d'attaques sur les chiffrements par bloc
- Rappel AES
- L'attaque
- Organisation du programme
- Problématiques actuelles
- Nos résultats

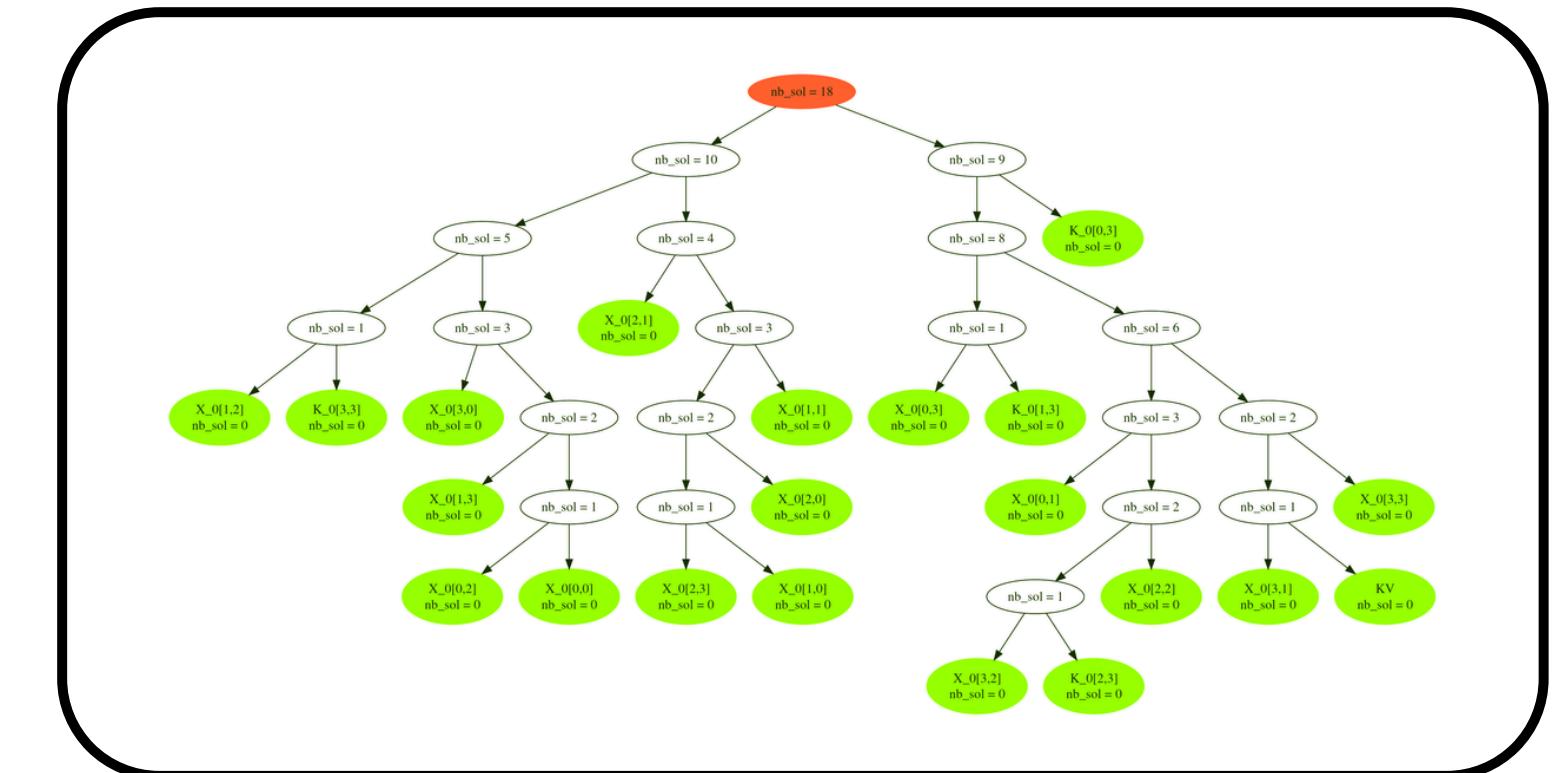
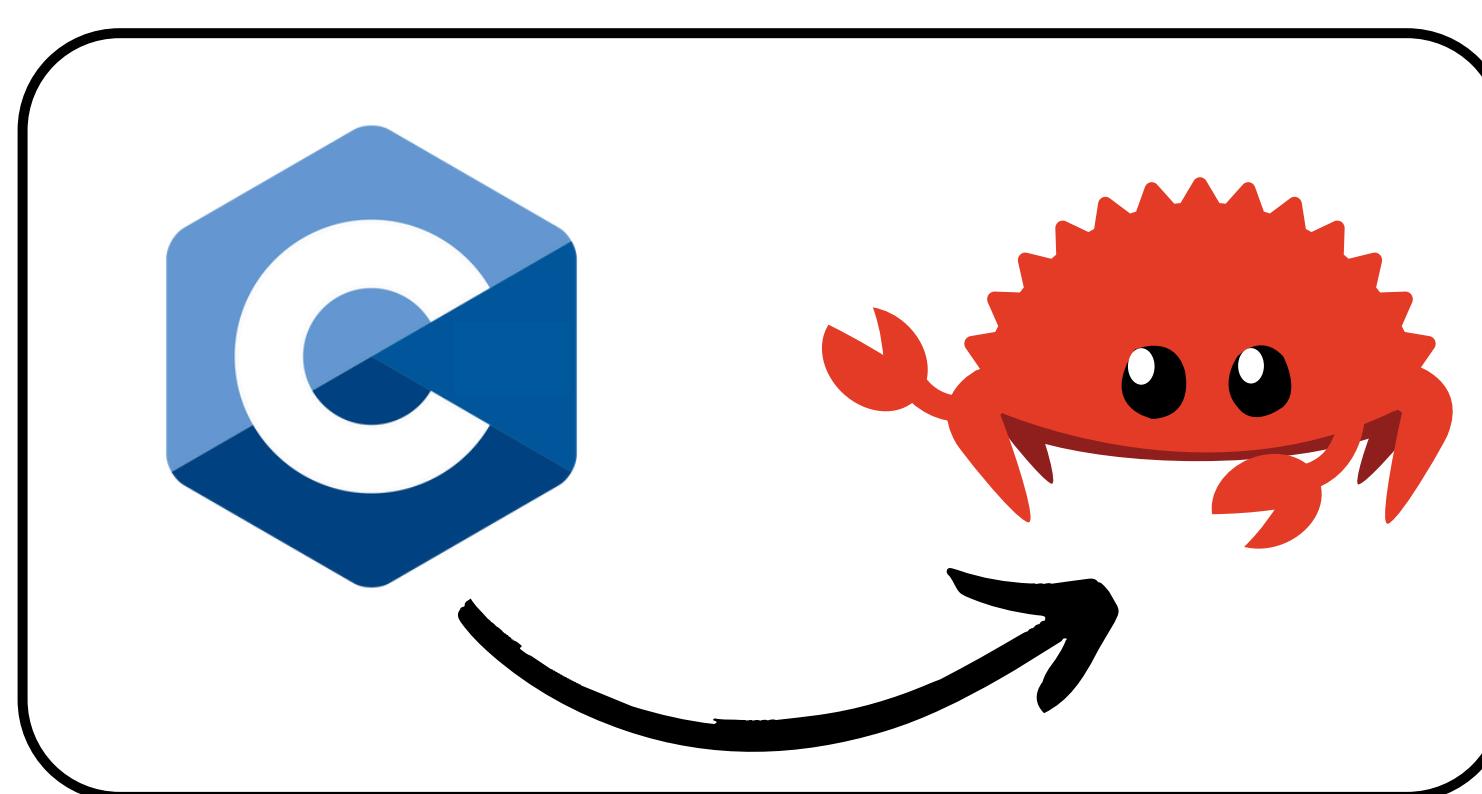


# Objectifs du projet

*“Automatic Search of Attacks on round-reduced AES and Applications”  
Charles Bouillaguet, Patrick Derbez, Pierre-Alain Fouque*

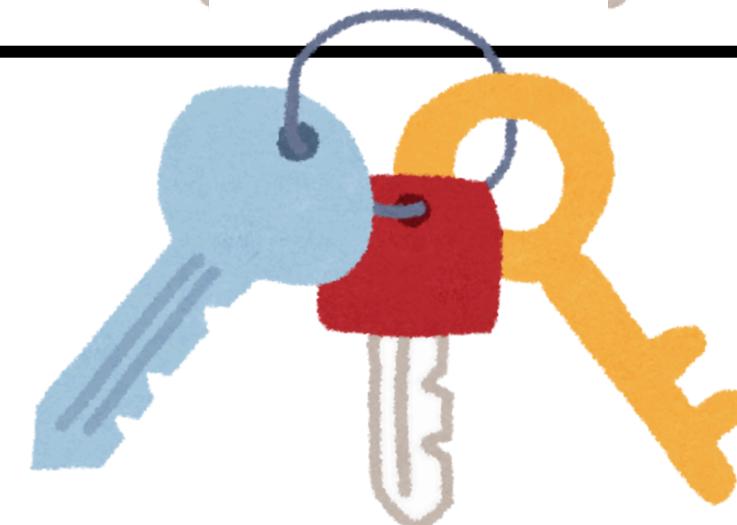
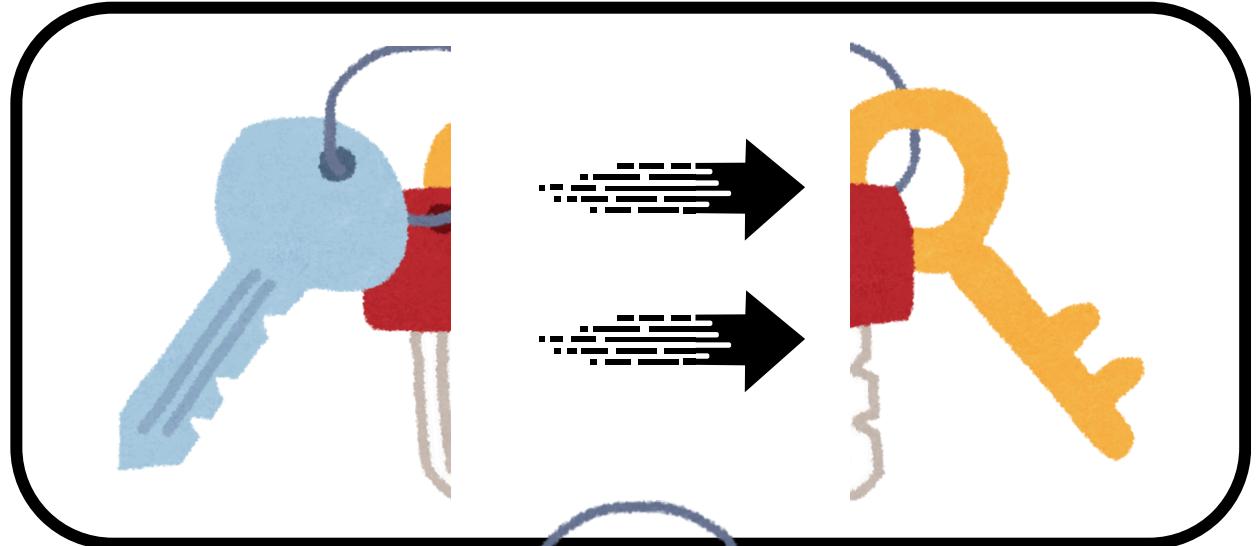
2012

Réimplémentation avec évolutions

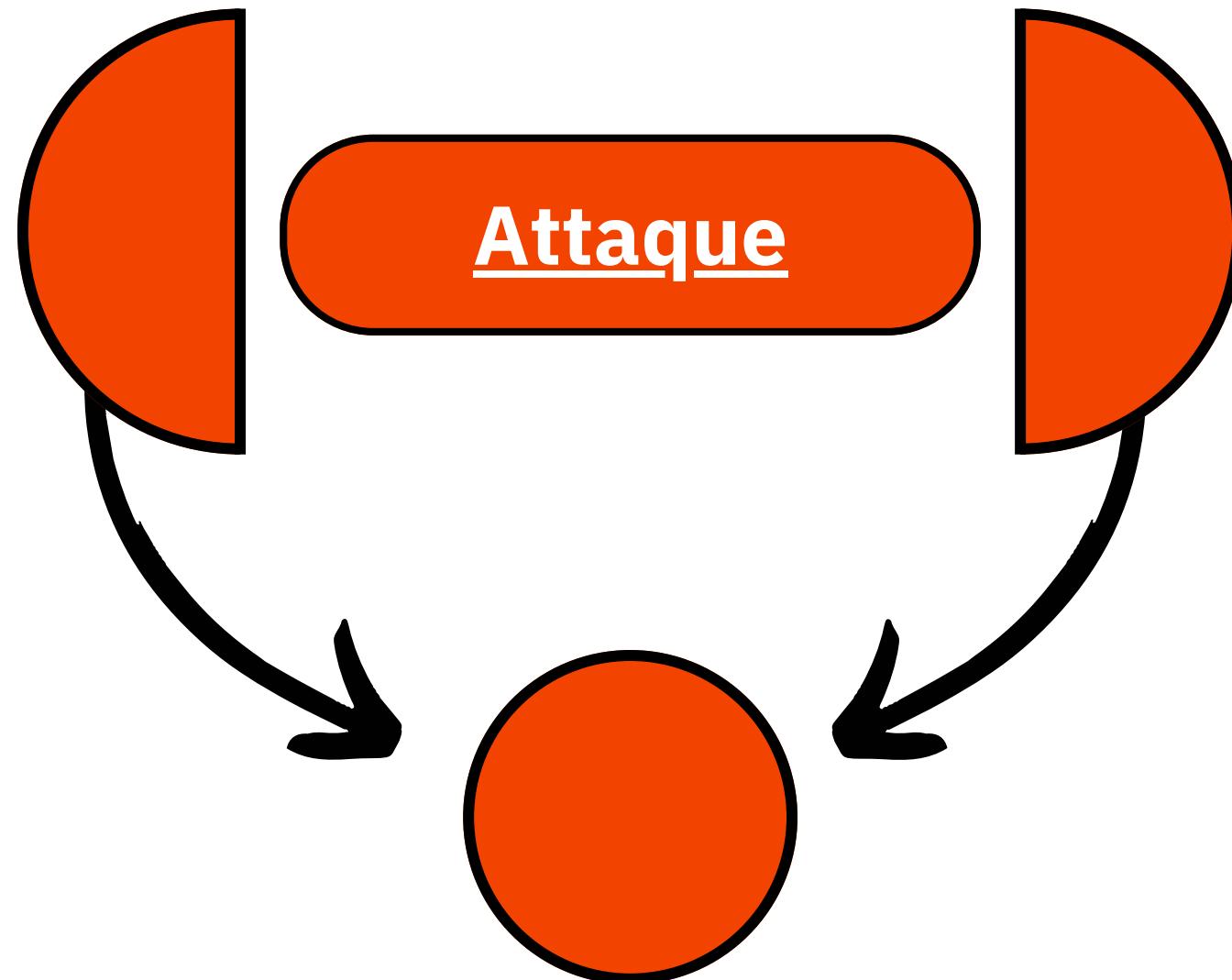


# Types d'attaques

Guess and determine



Meet in the middle



# Généralité de AES

Supporte les clés de taille **128, 192 or 256 bits**



Chaque coefficient correspond à un octet

$$128 \text{ bits} = 8 * 4^2$$

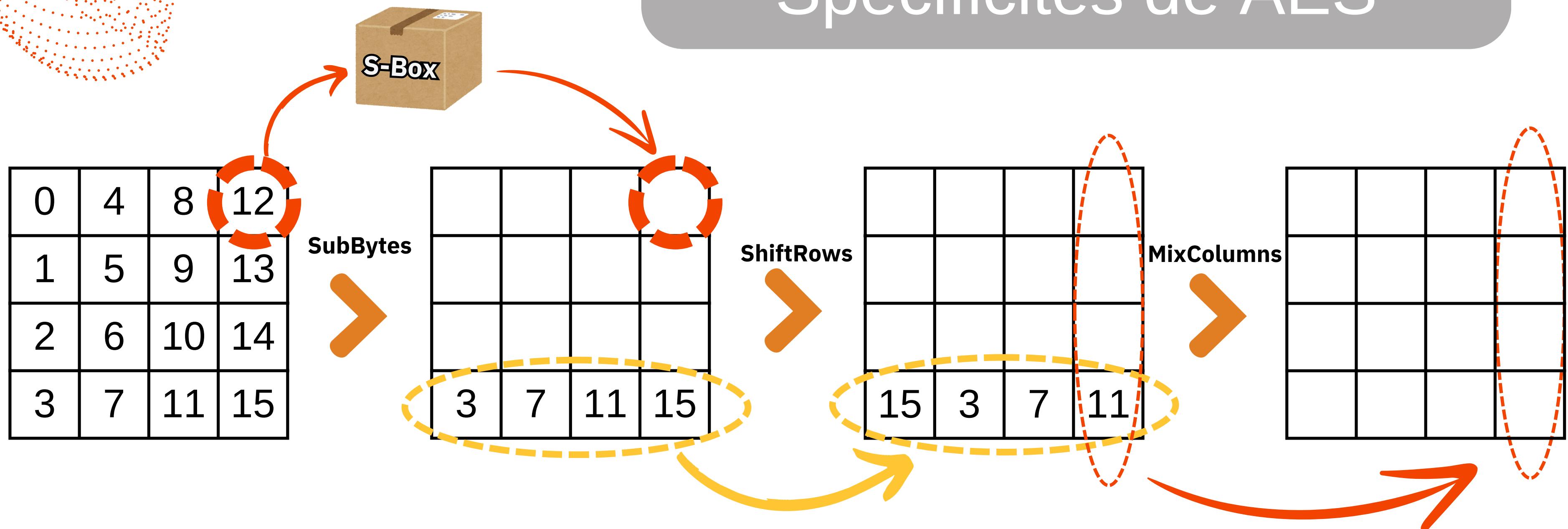
**128 bits** Bonjour à tous,  
Nous espérons que cette présentation :  
. vous plaira et nous amènera à obtenir une bien agréable gradation !  
. vous fera apprendre un maximum de choses !  
**128 bits** . vous fera apprendre un maximum de choses !  
. vous fera apprendre un maximum de choses !  
. vous fera apprendre un maximum de choses ! 

Texte découpé en bloc de **128 bits**

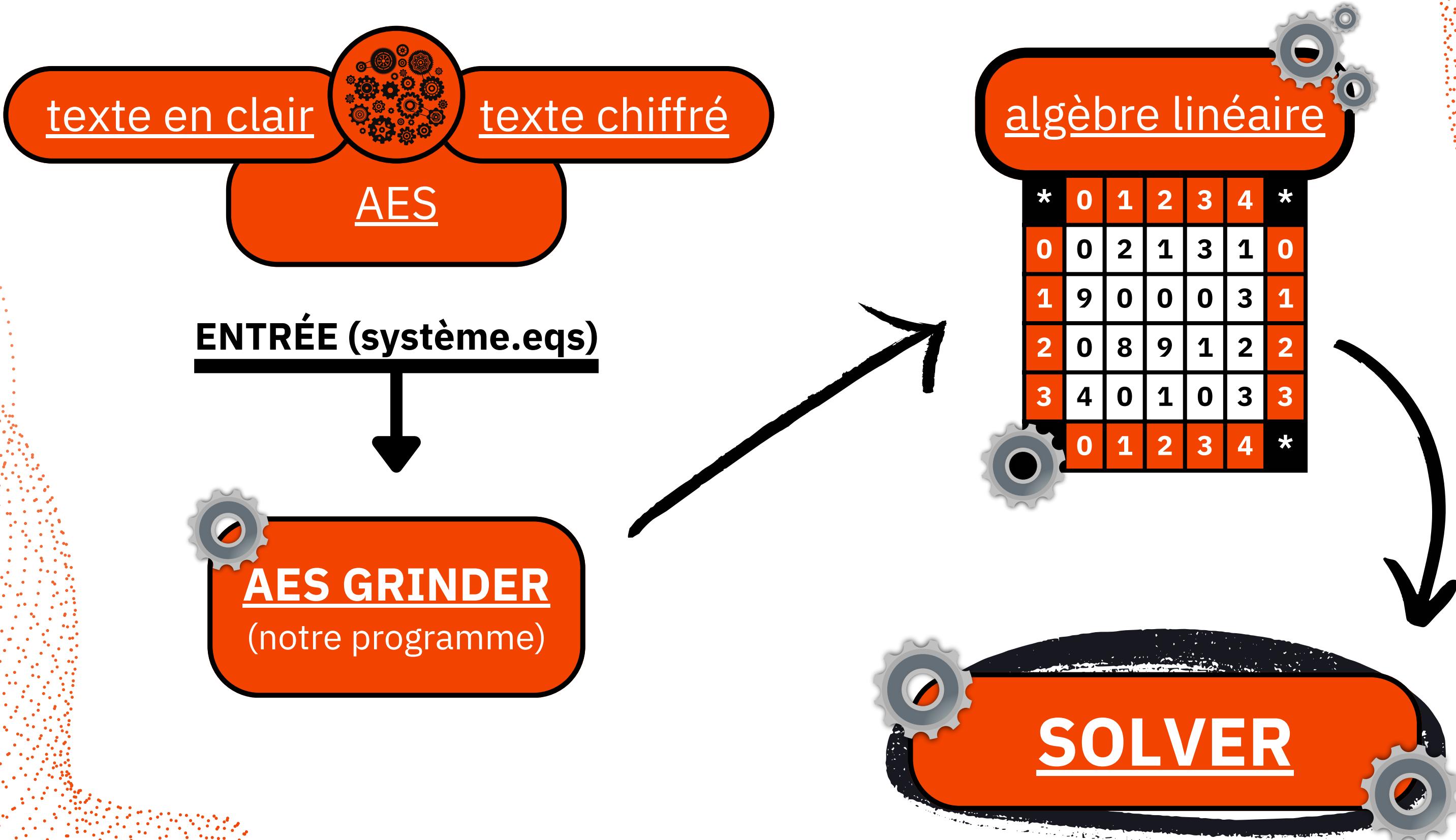
Traité sous forme de matrice

0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15

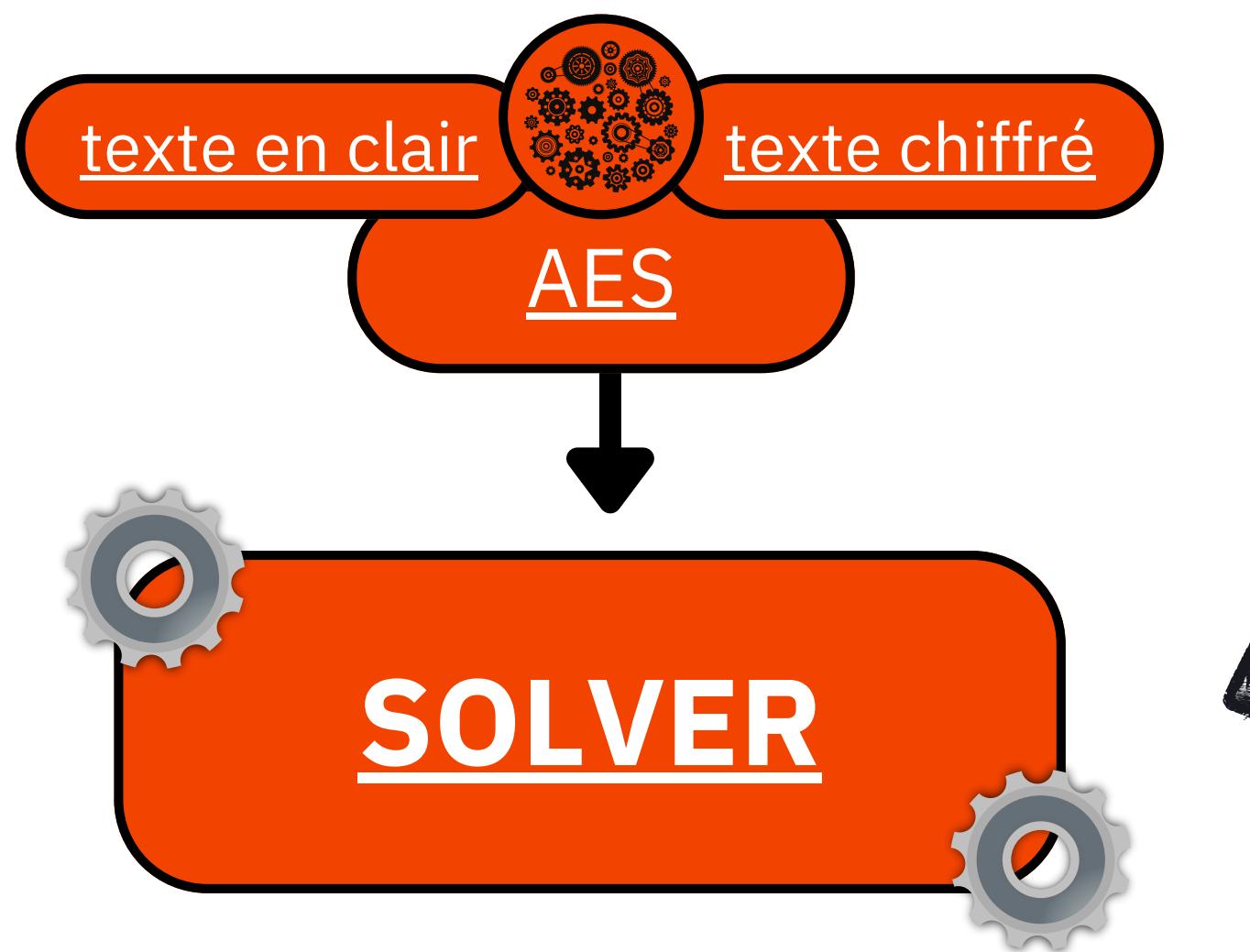
# Spécificités de AES



# Notre attaque sur AES



# Algorithme de recherche

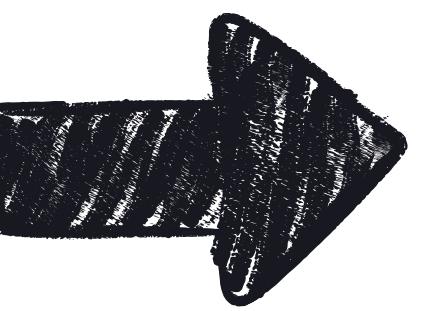


# Opérations sur la matrice

Opérations dans le corps fini GF( $2^8$ )

Polynôme caractéristique AES = 0x11b

- Addition: Opération XOR
- Multiplication : multiplication polynomiale modulo  
0x11b



Pré-calculé  
donc temps  
constant  
de calcul

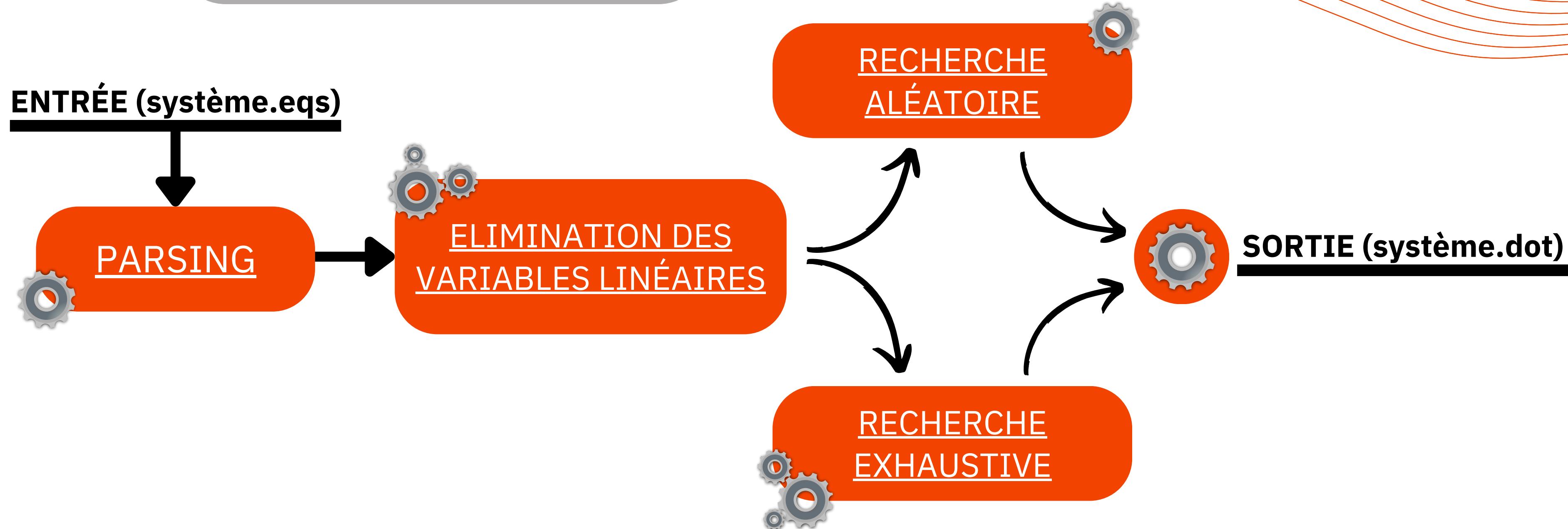
# AES sous forme d'équations

$$KS_i : \begin{cases} k_i[j] + k_i[j-4] + k_{i-1}[j] = 0, & j = 4, \dots, 15 \\ k_i[0] + k_{i-1}[0] + S(k_{i-1}[13]) + RCON_i = 0 \\ k_i[1] + k_{i-1}[1] + S(k_{i-1}[14]) = 0 \\ k_i[2] + k_{i-1}[2] + S(k_{i-1}[15]) = 0 \\ k_i[3] + k_{i-1}[3] + S(k_{i-1}[12]) = 0 \end{cases}$$

$$R_i : \begin{cases} y_i + S(x_i) = 0 \\ w_i + \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \times \begin{pmatrix} y_i[0] & y_i[4] & y_i[8] & y_i[12] \\ y_i[5] & y_i[9] & y_i[13] & y_i[1] \\ y_i[10] & y_i[14] & y_i[2] & y_i[6] \\ y_i[15] & y_i[3] & y_i[7] & y_i[11] \end{pmatrix} = 0 \\ x_{i+1} + w_i + k_{1+i} = 0 \end{cases}$$

$$\begin{aligned} & W_0[3,2] + K_1[3,2] + C[3,2] \\ & W_0[3,3] + K_1[3,3] + C[3,3] \\ & W_0[0,0] + 02*S(X_0[0,0]) + 03*S(X_0[1,1]) + S(X_0[2,2]) + S(X_0[3,3]) \\ & W_0[0,1] + 02*S(X_0[0,1]) + 03*S(X_0[1,2]) + S(X_0[2,3]) + S(X_0[3,0]) \\ & W_0[0,2] + 02*S(X_0[0,2]) + 03*S(X_0[1,3]) + S(X_0[2,0]) + S(X_0[3,1]) \\ & W_0[0,3] + 02*S(X_0[0,3]) + 03*S(X_0[1,0]) + S(X_0[2,1]) + S(X_0[3,2]) \\ & W_0[1,0] + S(X_0[0,0]) + 02*S(X_0[1,1]) + 03*S(X_0[2,2]) + S(X_0[3,3]) \\ & W_0[1,1] + S(X_0[0,1]) + 02*S(X_0[1,2]) + 03*S(X_0[2,3]) + S(X_0[3,0]) \\ & W_0[1,2] + S(X_0[0,2]) + 02*S(X_0[1,3]) + 03*S(X_0[2,0]) + S(X_0[3,1]) \\ & W_0[1,3] + S(X_0[0,3]) + 02*S(X_0[1,0]) + 03*S(X_0[2,1]) + S(X_0[3,2]) \\ & W_0[2,0] + S(X_0[0,0]) + S(X_0[1,1]) + 02*S(X_0[2,2]) + 03*S(X_0[3,3]) \\ & W_0[2,1] + S(X_0[0,1]) + S(X_0[1,2]) + 02*S(X_0[2,3]) + 03*S(X_0[3,0]) \\ & W_0[2,2] + S(X_0[0,2]) + S(X_0[1,3]) + 02*S(X_0[2,0]) + 03*S(X_0[3,1]) \\ & W_0[2,3] + S(X_0[0,3]) + S(X_0[1,0]) + 02*S(X_0[2,1]) + 03*S(X_0[3,2]) \\ & W_0[3,0] + 03*S(X_0[0,0]) + S(X_0[1,1]) + S(X_0[2,2]) + 02*S(X_0[3,3]) \\ & W_0[3,1] + 03*S(X_0[0,1]) + S(X_0[1,2]) + S(X_0[2,3]) + 02*S(X_0[3,0]) \\ & W_0[3,2] + 03*S(X_0[0,2]) + S(X_0[1,3]) + S(X_0[2,0]) + 02*S(X_0[3,1]) \\ & W_0[3,3] + 03*S(X_0[0,3]) + S(X_0[1,0]) + S(X_0[2,1]) + 02*S(X_0[3,2]) \\ & P[0,0] + K_0[0,0] + X_0[0,0] \\ & P[0,1] + K_0[0,1] + X_0[0,1] \end{aligned}$$

# AES GRINDER



## PARSING

### ENTRÉE (système.eqs)

```
var_00 * 0 + var_01 + var_02 + var_03 + var_99 * 0  
var_00 * 1 + var_04 + var_05 + var_06 + var_99 * 1  
var_00 * 2 + var_07 + var_08 + var_09 + var_99 * 2  
var_00 * 3 + var_10 + var_11 + var_12 + var_99 * 3
```

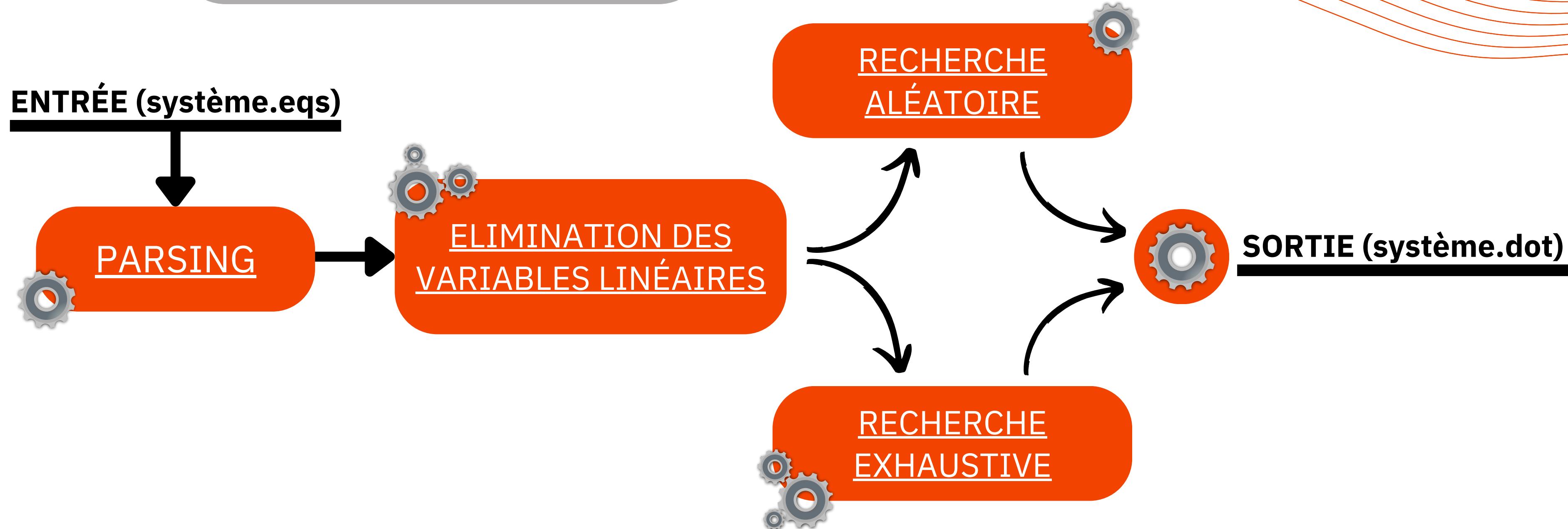
### hashmap

var_00	0
var_99	4
var_01	1
var_04	5
var_05	6
var_06	7
var_07	8
var_08	9
var_10	11
var_11	12
var_12	13

### matrice

*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	*
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	1	1	1	1	0	0	0	0	0	0	1
2	2	0	0	0	2	0	0	0	1	1	1	0	0	0	2
3	3	0	0	0	3	0	0	0	0	0	0	1	1	1	3
*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	*

# AES GRINDER

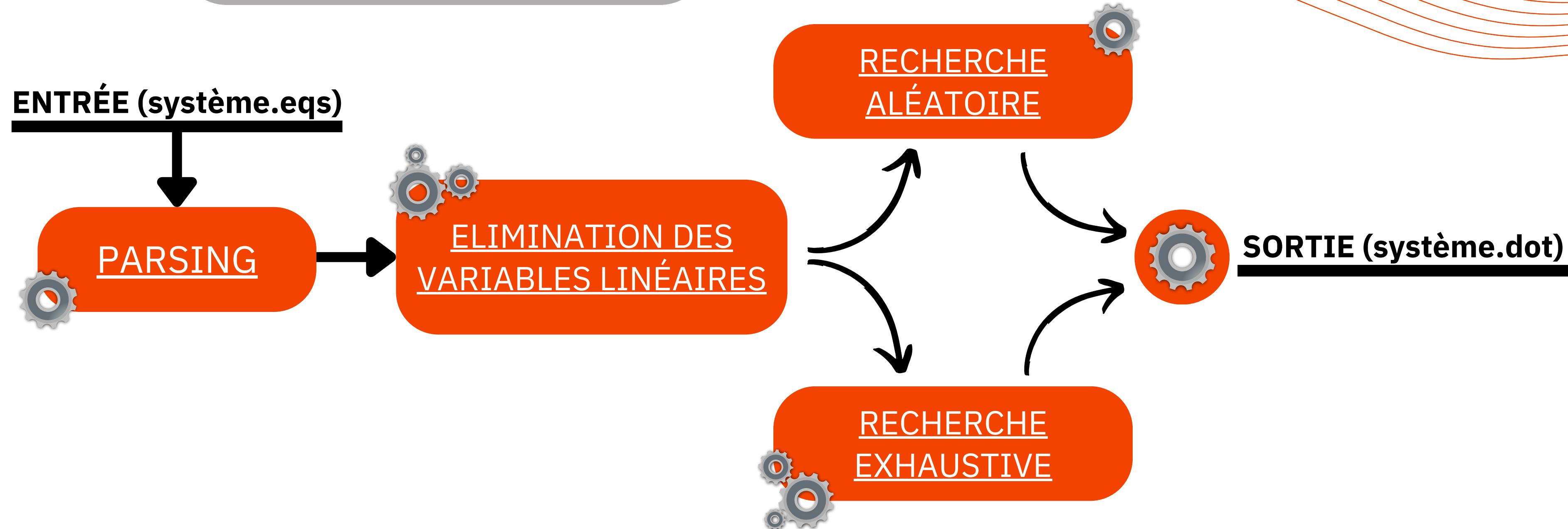


*	x							S(x)	*
*	0	1	1	1	0	0	0	0	*
*	1	0	0	0	1	1	1	1	*
*	2	0	0	0	2	0	0	0	*
*	3	0	0	0	3	0	0	0	*
*	2	6	0	9	1	5	0	8	*
*	5	0	4	4	4	0	7	9	*
*	x							S(x)	*

ELIMINATION DES  
VARIABLES LINÉAIRES

*	x	S(x)	*
*	1	8	*
*	2	4	*
*	4	9	*
*	5	0	*
*	3	7	*
*	4	6	*
*	x	S(x)	*

# AES GRINDER



# Construction récursive d'un solver

## Arborescence

### Noeud Racine

(toutes les variables apparaissent)

### Feuille

(une seule variable apparaît)



# Propriétés sur le système d'équations

Chaque équation du système peut s'écrire sous la forme :

$$f = f_1 + f_2$$

avec  $f$  une équation du système

$f_1$  équation du sous système où n'apparaissent que les variables  $X_1$

$f_2$  équation du sous système où n'apparaissent que les variables  $X_2$

$$X = X_1 \cup X_2$$

avec  $X$  l'ensemble des variables du système d'équations

$$F(x_1, \dots, x_n) = 0 \Leftrightarrow \begin{cases} F(x_1, \dots, x_k) = F(x_{k+1}, \dots, x_n) \\ F(x_1, \dots, x_k) = 0 \\ F(x_{k+1}, \dots, x_n) = 0 \end{cases}$$

# Intuition

Soit,  $X = X_1 \cup X_2$

## Deux Algorithmes de recherche

  $A_1$  résout le problème sur  $E \cap V(X_1)$

Soit,  $S(A_x)$  les solutions d'un algorithme

  $A_2$  résout le problème sur  $E \cap V(X_2)$

Une solution de  $E$ , est aussi une solution de  $E \cap V(X_1)$ , la réciproque n'est pas toujours vraie.

Test des solutions du produits cartésiens  $S(A_1) \times S(A_2)$  sur les équations de  $E$

# Propriétés d'un solver

Complexité en Temps

$T$



$$\max (T(A_1), T(A_2), O(A_1 \text{fusion} A_2))$$

Nombre de Solutions

$O$



$V(A)$

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Ensemble de Variables

$V$

Retourner.  $|V(A)|$  - nombre d'équations à zéro

# Comparaison

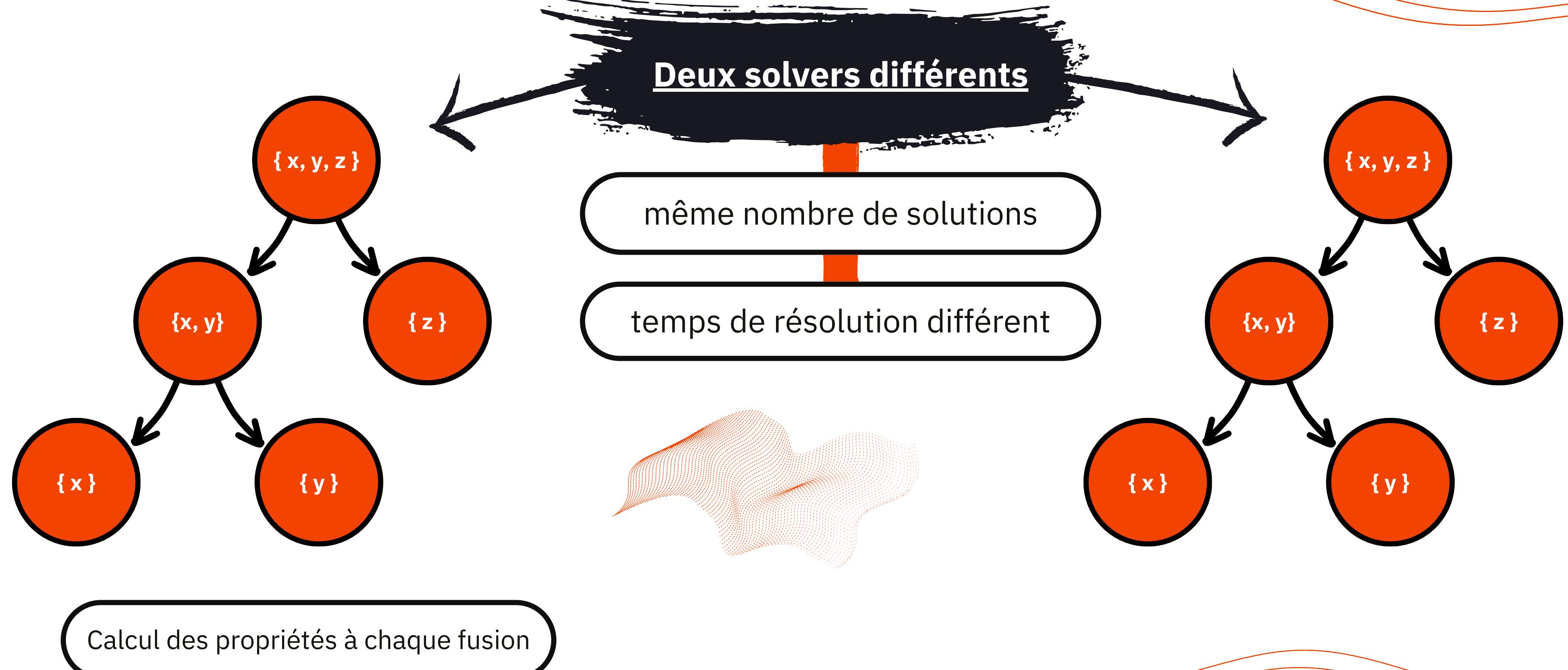
$$A_1 \succeq_1 A_2 \Leftrightarrow \begin{cases} V(A_1) = V(A_2) \\ T(A_1) \leq T(A_2) \end{cases}$$

$$A_1 \succeq_1 A_2 \rightarrow A_1 \bowtie A_3 \succeq_1 A_2 \bowtie A_3$$

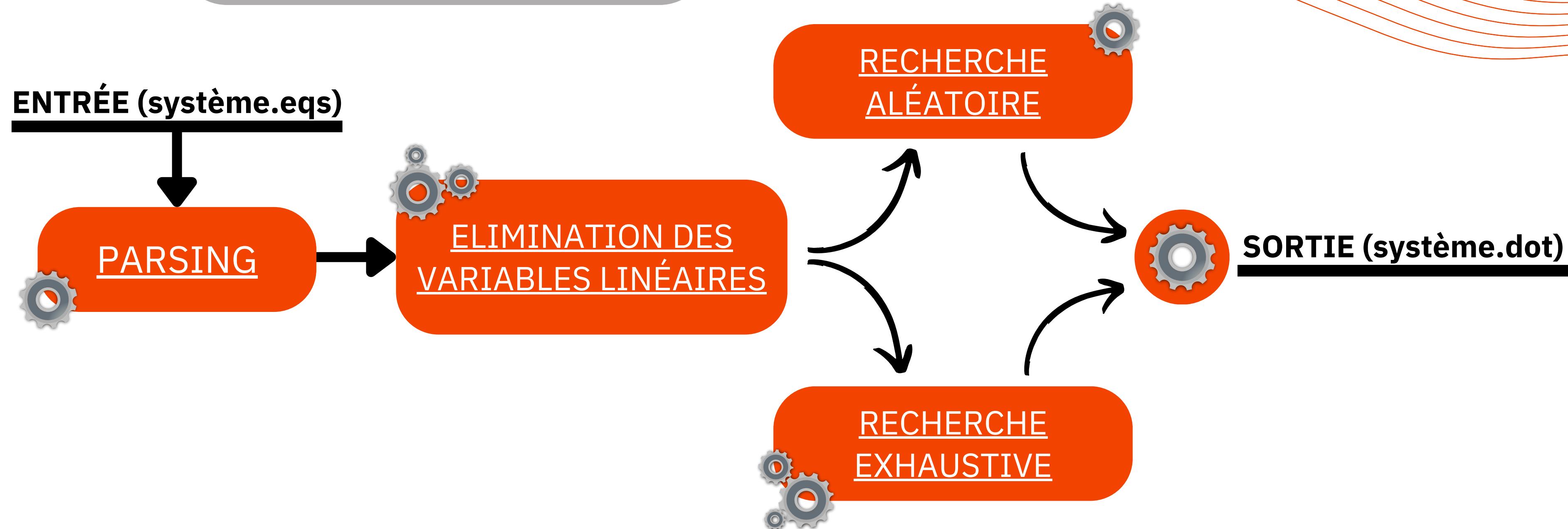
Fusion Compatible

$$\mathcal{A}_1 \succeq_2 \mathcal{A}_2 \Leftrightarrow \begin{cases} T(\mathcal{A}_1) \leq T(\mathcal{A}_2) \\ V(\mathcal{A}_1) \supseteq V(\mathcal{A}_2) \\ O(\mathcal{A}_1) \leq O(\mathcal{A}_2) \end{cases}$$

# Arbre de recherche



# AES GRINDER



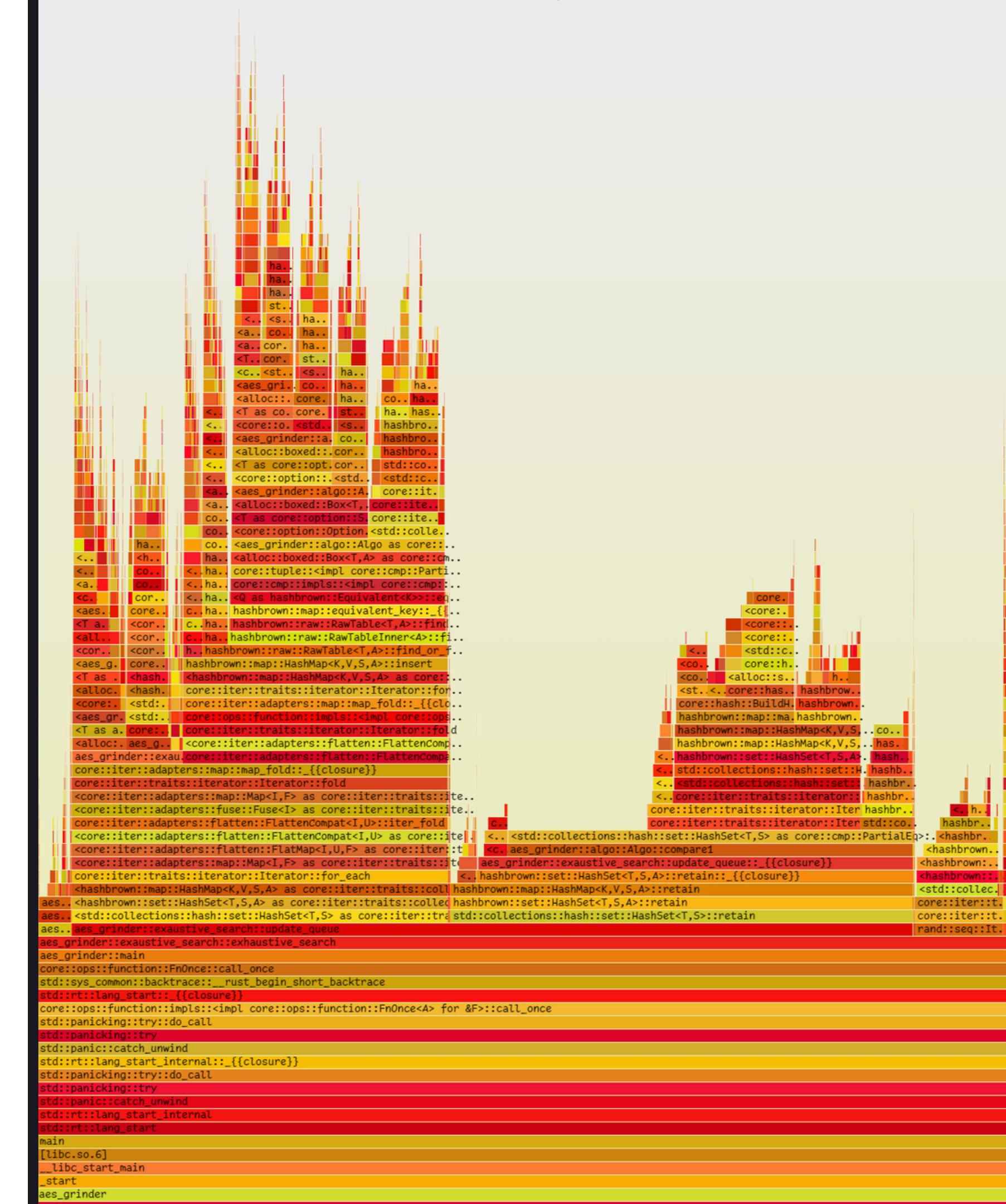
## RECHERCHE EXHAUSTIVE



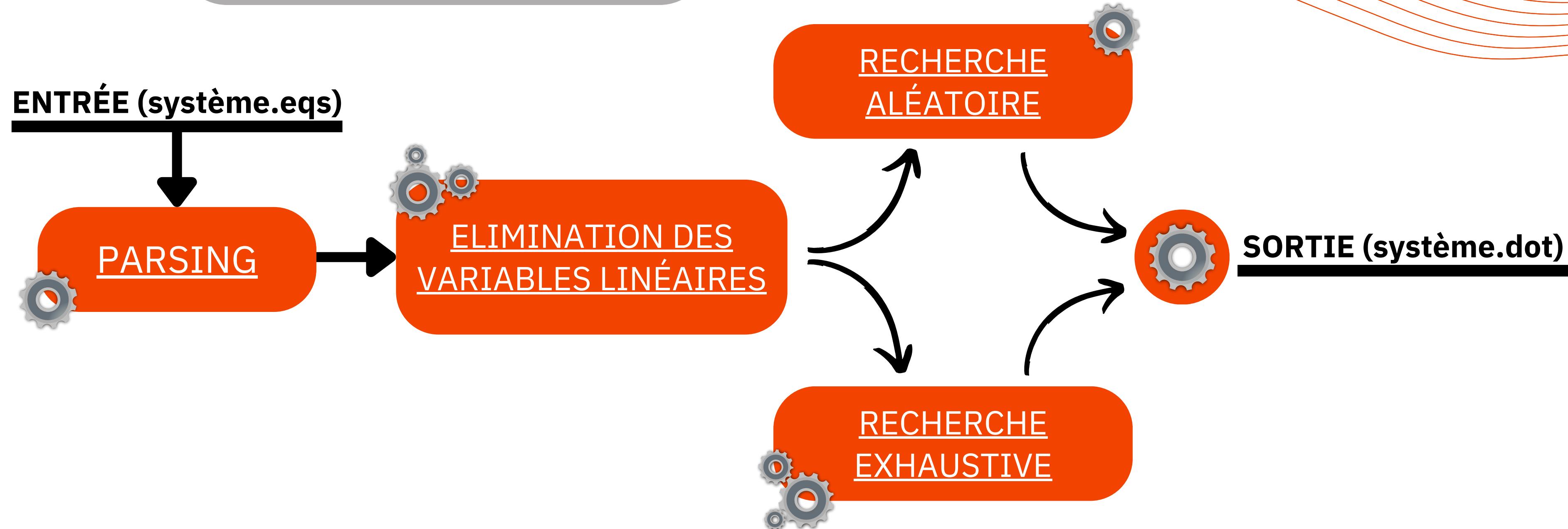
Trouver la meilleure solution

Mais très inefficace  
si le nombre de rounds AES augmente

- Tant que l'on a des algos de base
  - Construction des solvers de bases
  - Fusion de deux solvers de bases
  - Comparaison des algos
    - On garde le meilleur algo
    - On détruit l'algo le moins bon

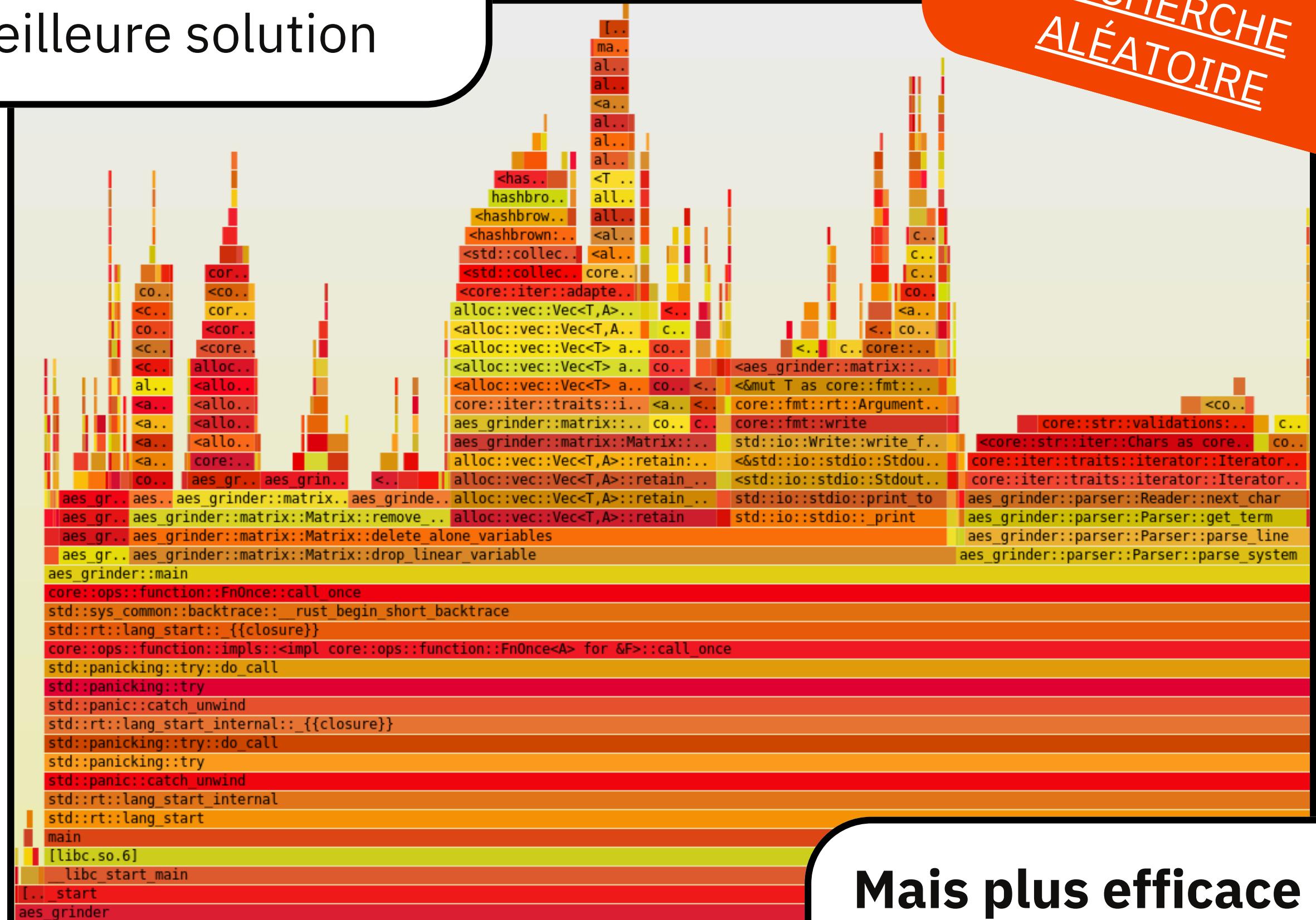


# AES GRINDER



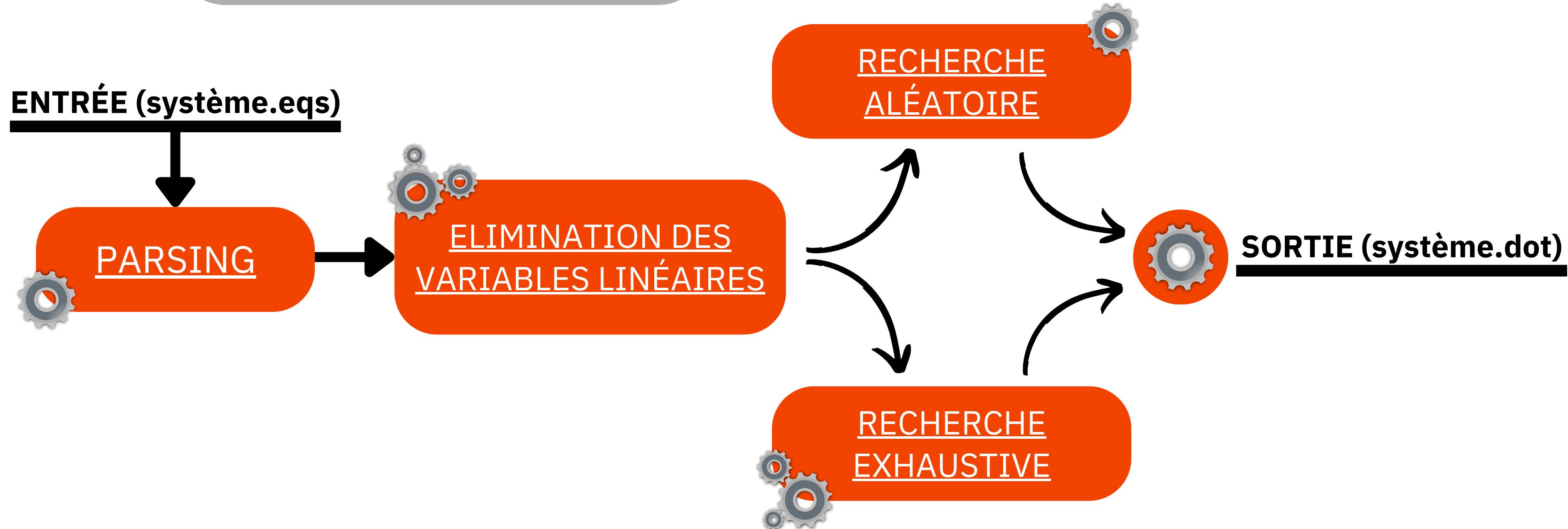
**Pas de garantie de trouver  
la meilleure solution**

**RECHERCHE  
ALÉATOIRE**



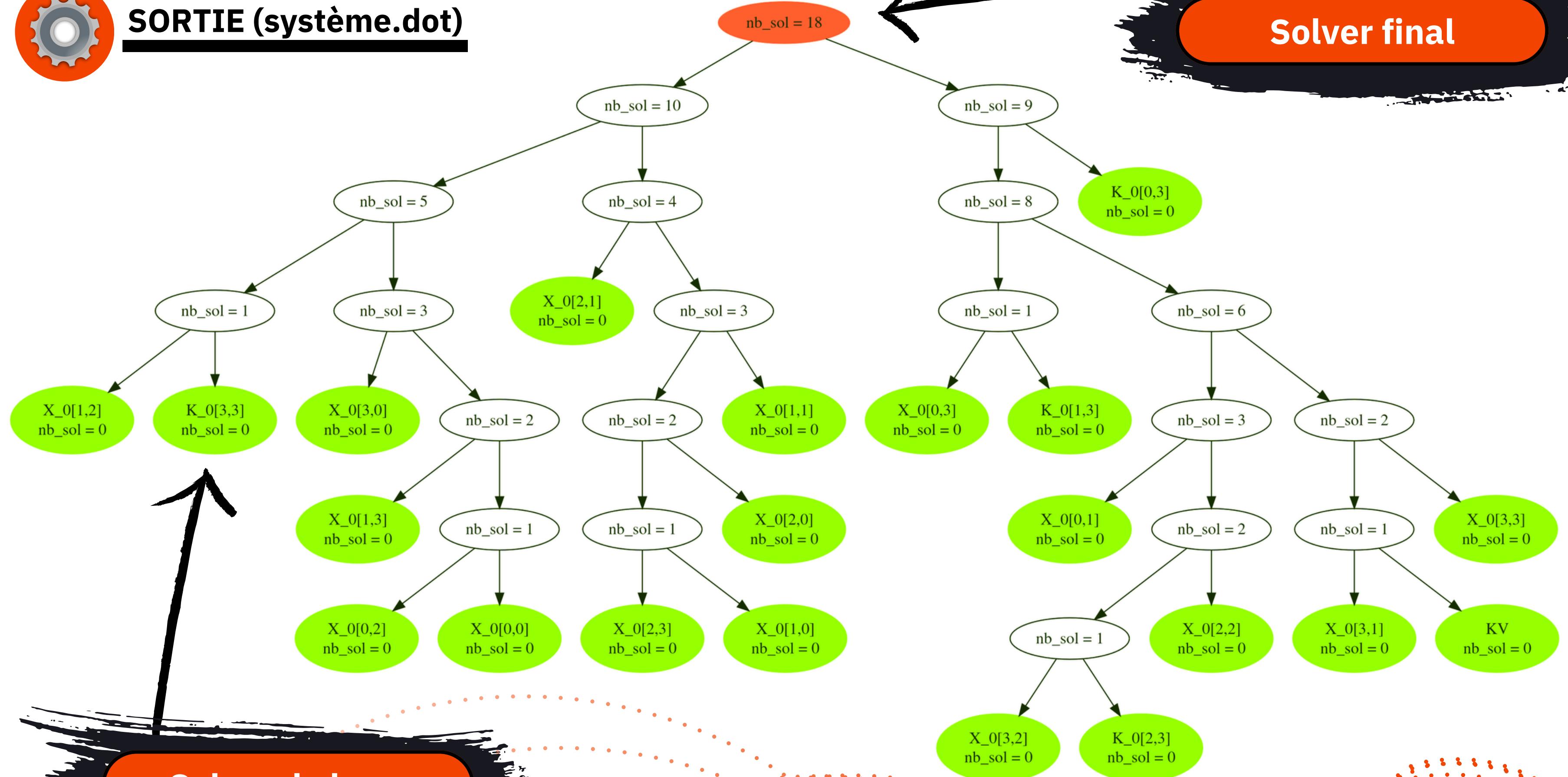
**Mais plus efficace que la  
recherche exhaustive**

# AES GRINDER





## SORTIE (système.dot)



Solver de base

Solver final

# Conclusion

Nos Résultats - Résultats attendus - Limitations

