RETOUR SUR TRACE

À la fin de ce chapitre, je sais :

- ☞ expliquer le principe du retour sur trace
- ☞ donner des exemples d'utilisation
- ☞ coder un algorithme de retour sur trace en OCaml

A Exploration

Soit un problème $\mathcal{P}$ de satisfaction de contraintes tel que les solutions $\mathcal{S}$ se trouvent dans une ensemble fini $\mathcal{E}$ de candidats. On cherche à trouver les solutions de $\mathcal{P}$ en explorant l'ensemble $\mathcal{E}$ tout en respectant les contraintes.

Il est imaginable aujourd'hui d'envisager l'usage de la force brute pour résoudre des problèmes dont la dimension est pourtant élevée.

■ **Définition 1 — Recherche par force brute.** Énumérer tous les éléments candidats de $\mathcal{E}$ et tester s'ils sont solution de $\mathcal{P}$.

Il s'agit donc d'une approche simple à énoncer et à implémenter comme le montre l'algorithme 1.

Algorithme 1 Algorithme de recherche par force brute, problème de satisfaction de contraintes

```
1 : Fonction FORCE_BRUTE(ℰ)
2 :     𝒮 ← ∅
3 :     pour e ∈ ℰ répéter
4 :         si e est un solution de 𝒫 alors
5 :             𝒮 ← 𝒮 ∪ {e}
6 :     renvoyer 𝒮
```

■ **Exemple 1 — Exemple d'algorithmes de recherche par force brute.** Parmi les algorithmes de recherche par force brute utilisés, on note :

- la recherche d'un code secret à quelques chiffres : on teste toutes les permutations possibles jusqu'à trouver la bonne,

- la recherche d'un élément dans un tableau : on teste tous les éléments les uns après les autres jusqu'à trouver le bon,
- le tri bulle : on essaie de placer un élément dans une case, puis on essaie la case du dessus.

■ **Exemple 2 — Problème des huit reines.** On cherche à placer sur un échiquier de 8x8 cases huit reines sans que celles-ci s'attaquent les unes les autres. Une solution est présentée sur la figure 1. On connaît les solutions de ce problème [**ball_eight_1960**] et on peut même les formuler simplement.

En choisissant la recherche par force brute, il est nécessaire de tester $8^8 = 16777216$ configurations. Si le test de la validité de l'échiquier est effectué en moins d'une microseconde [a], l'intégralité des configurations sera examinée en un temps proche de la seconde.

a. ce qui est très réaliste avec une machine standard et un programme non optimisé

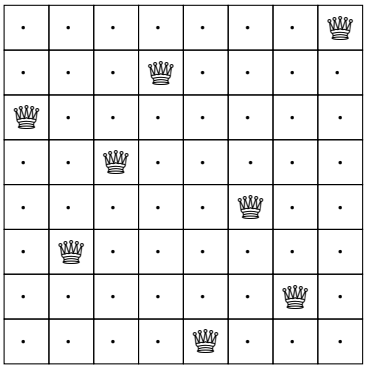

FIGURE 1 – Exemple d'échiquier 8x8 solution au problème des huit reines.

Même si elle est simple à énoncer et à implémenter, la recherche par force brute présente un inconvénient majeur : elle ne supporte pas le passage à l'échelle, c'est à dire qu'elle devient rapidement inutilisable à cause de l'explosion du cardinal de l'ensemble $\mathcal{E}$ à explorer qui induit un temps de calcul nécessaire rédhibitoire.

■ **Exemple 3 — Problème des n reines.** Le problème des n reines est la généralisation du problème des huit reines sur un échiquier de taille $n \times n$ et avec n reines à placer. L'ensemble des candidats est maintenant de taille n^n. Pour $n = 16$, le temps de calcul dépasse déjà la dizaine de milliers d'années. En effet, admettons que le test de validité de l'échiquier soit toujours de l'ordre de la microseconde, on a : $(16^{16} \times 1^{-6})/(60 \times 60 \times 24 \times 365) \approx 584942$ années...

B Principe du retour sur trace

Le retour sur trace est une technique exploratoire utilisée afin guider l'exploration et de ne pas tester toutes les configurations possibles.

■ **Définition 2 — Retour sur trace.** Le retour sur trace construit des ensembles de solutions partielles au problème $\mathcal{P}$. Ces solutions partielles peuvent être complétées de différentes manières pour former une solution au problème. La complétion des solutions se fait de manière incrémentielle.

Le retour sur trace utilise une représentation de l'espace des candidats $\mathcal{E}$ sous la forme d'un arbre de recherche. Chaque solution partielle est un nœud de cet arbre. Chaque solution complète forme un chemin descendant de la racine à une feuille de l'arbre. Au niveau de la feuille, on ne peut plus compléter la solution par quoi que ce soit.

L'algorithme de retour sur trace est un algorithme récursif qui parcours en profondeur l'arbre de recherche en vérifiant qu'il peut compléter la solution partielle par le nouveau nœud trouvé : si c'est le cas, alors il continue l'exploration de cette branche. Si ce n'est pas le cas, cette branche est écartée, car elle ne peut pas donne de solutions.

Vocabulary 1 — Backtracking ⇜⇝ Retour sur trace

Algorithme 2 Algorithme de retour sur trace

```
1 : Fonction RETOUR_SUR_TRACE(v)                ▷ v est un nœud de l'arbre de recherche
2 :     si v est une feuille alors
3 :         renvoyer Vrai
4 :     sinon
5 :         pour chaque fils u de v répéter
6 :             si u peut compléter une solution partielle au problème P alors
7 :                 RETOUR_SUR_TRACE(u)
8 :     renvoyer Faux
```

■ **Exemple 4 — Retour sur trace sur le problème des quatre reines.** L'arbre de recherche nécessaire à l'exécution de l'algorithme de retour sur trace pour le problème des quatre reines est représenté sur la figure 2. La racine de l'arbre est le début de l'algorithme : on n'a pas encore placé de reines. Le première étape est le placement d'une reine sur l'échiquier : sur une même ligne on peut la placer sur quatre colonnes différentes qu'il va falloir tester. Chaque colonne représente donc une solution partielle différente et est un nœud fils de la racine. On peut placer la première reine sur n'importe quelle colonne.

La seconde étape est le placement d'une deuxième reine. Comme on parcours l'arbre en profondeur, on teste la première configuration en premier, c'est à dire une reine sur la première case de la première ligne. On teste alors toutes les solutions partielles possibles. Le premier nœud ne satisfait pas les conditions de validité : on ne peut pas placer une reine

dans la première colonne car il y en a déjà une sur la première ligne. Tout le sous-arbre lié à cette solution partiel est élagué. De même pour le second nœud pour lequel la première reine peut attaquer en diagonale.

On revient donc là où on en était et on continue avec les autres fils valides.

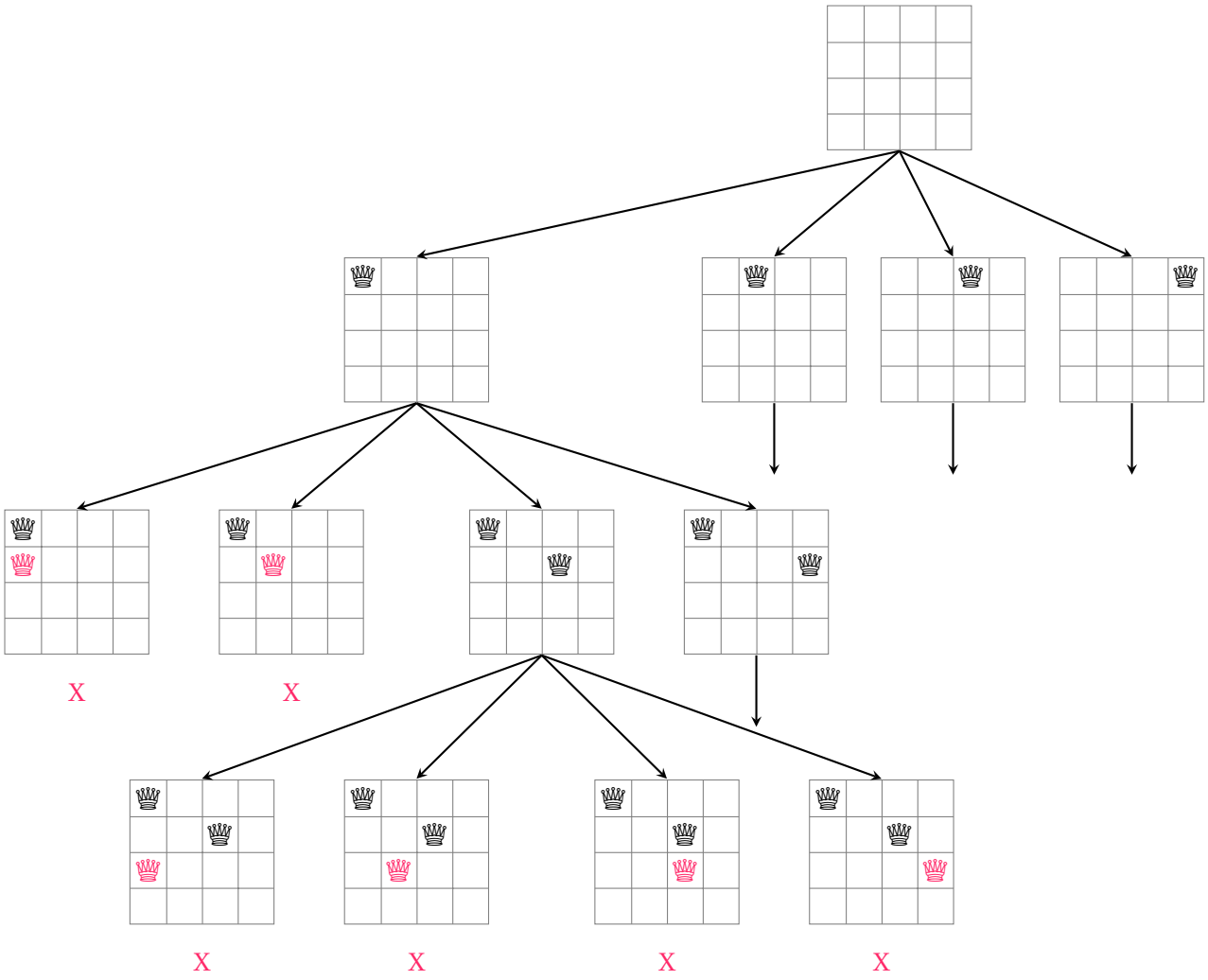

FIGURE 2 – Exemple d'arbre de recherche structurant l'algorithme de retour sur trace. Application au problème de quatre reines.