Algorithmique:
Backtracking 1 - Correction

1 N-Reines

1.1 Implémentation (4 points)

Vous trouverez dans le fichier *nreinesBT.py* disponible sur Moodle une proposition d'implémentation des fonctions du problème.

1.2 Complexité (2 point)

Dans le pire des cas, nous devons faire une recherche en profondeur dans un arbre dont chaque noeud donne naissance à n branches, ceci jusqu'à une profondeur de n, c'est-à-dire une complexité en temps de $O(n^n)$. Les différentes contraintes permettent de réduire le nombre de branche qui doivent être effectivement calculer. La contrainte explicite $T(x, k, n)$ permet de réduire le nombre de branche à chaque niveau. Pour le premier niveau, on a n branches, chacune des ces branches à $n-1$ enfants. Cependant, pour trouver les enfants, la complexité est en $O(n^2)$. On a donc la récursion $T(n) \approx T(n-1) \cdot n + O(n^2)$, ce qui donne une complexité $O(n^2 \cdot n!)$. La contrainte implicite $B(x, k, n)$ permet encore de réduire le nombre de branches qui ne seront pas explorées, cette contrainte ne permet pas de réduire suffisamment le nombre de branche pour se retrouver avec une complexité exponentielle ($O(2^n)$), comme on le voit sur la figure 1, pour l'implémentation proposée.

1.3 Comparaison avec la méthode Greedy (3 points)

La complexité pour la méthode Greedy est très certainement polynomiale. En effet, comme dit au point précédent, la vérification des contraintes se fait en $O(n^2)$; ainsi, si l'algorithme Greedy bouge k reines avant de trouver la solution, on a une complexité de $O(kn^2)$. Cependant, la moyenne de k semble proportionnelle à n, et l'algorithme Greedy a donc une complexité de $O(n^3)$. Cette complexité peut encore être changée, tenant compte du fait que l'algorithme Greedy peut rester coincé dans un minimum local, ce qui demande de relancer une recherche depuis zéro ; si le taux de minimums locaux ne dépend pas de n, la complexité reste néanmoins essentiellement inchangée.

Comme il vient d'être dit, la méthode Greedy présente l'inconvénient de rester coincée dans un minimum local : une seule recherche ne garantit pas de trouver une solution. De plus, la méthode

Greedy ne permet pas de faire une recherche exhaustive des solutions de manière efficace, ce que fait précisément la méthode de Backtracking. Ainsi, la méthode Greedy est particulièrement rapide (complexité polynomiale) par rapport à la méthode de Backtracking (complexité exponentielle ou factorielle) pour trouver *quelques* solutions distinctes, tandis que la méthode de Backtracking est avantageuse pour les trouver toutes.

1.4 Astuce (1 point)

Etant donné une solution, on peut très rapidement en obtenir d'autres grâce aux rotations (on peut tourner l'échiquier de 4 fois 90 degrés avant de revenir à la position initiale), qui sont des transformations changeant la disposition des pièces mais préservant le nombre de conflits, tout comme les réflexions.

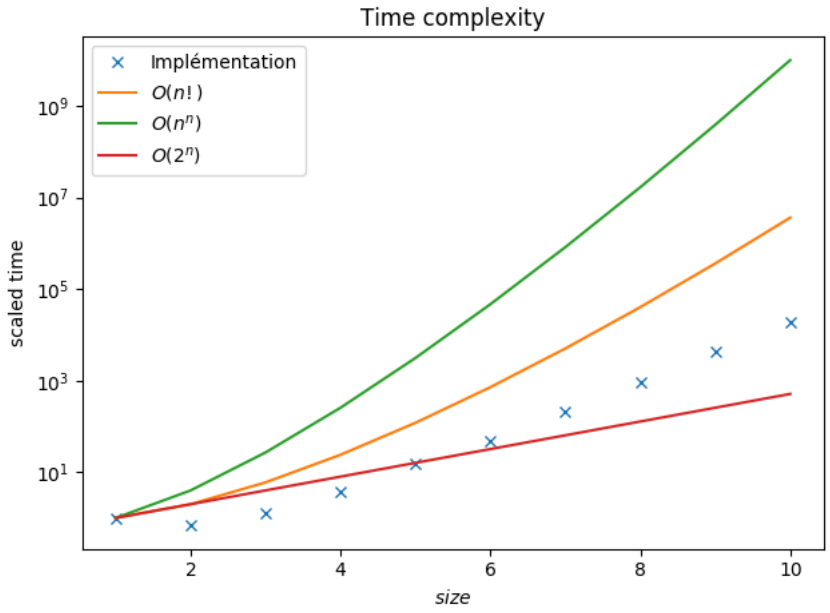

FIGURE 1 – Complexité en temps pour l'implémentation proposée au point 1