



Nathan Filaux
Colas Rémi
1D1

Compte Rendu SAE S2.02

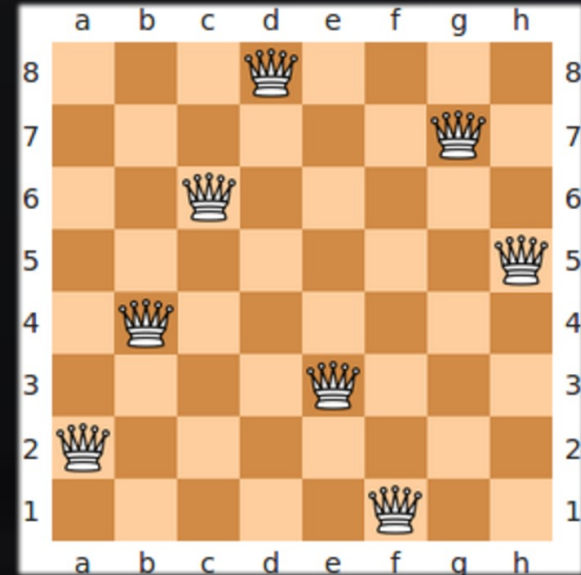
Présentation du problème des 8 dames

Compte Rendu SAE S2.02

Exemple de **solution** pour **$n = 8$**

Dimensions : **$8*8$**

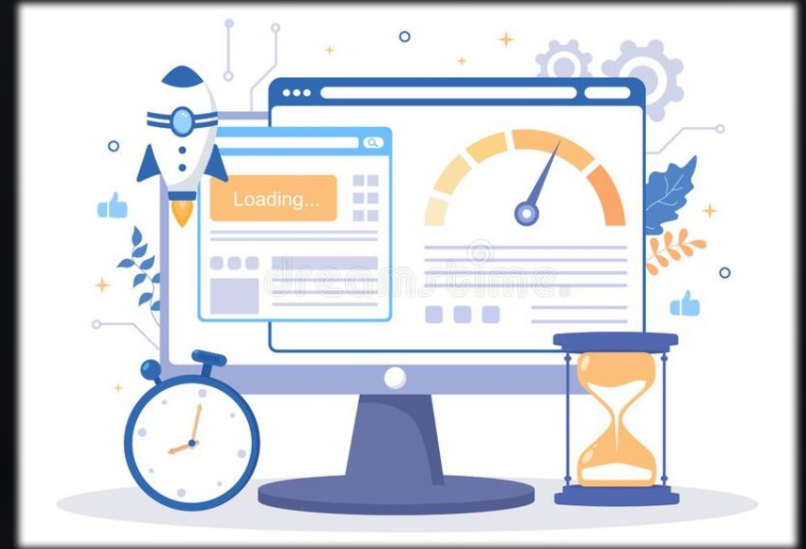
Nombre de **reines** à placer : **8**



Compte Rendu SAE S2.02

Il faut :

- Trouver toutes les solutions donc parcourir **tous les chemins**
- **Optimiser** le code pour diminuer le **temps** d'exécution
- Trouver une **stratégie** pour limiter le nombre de cas à étudier



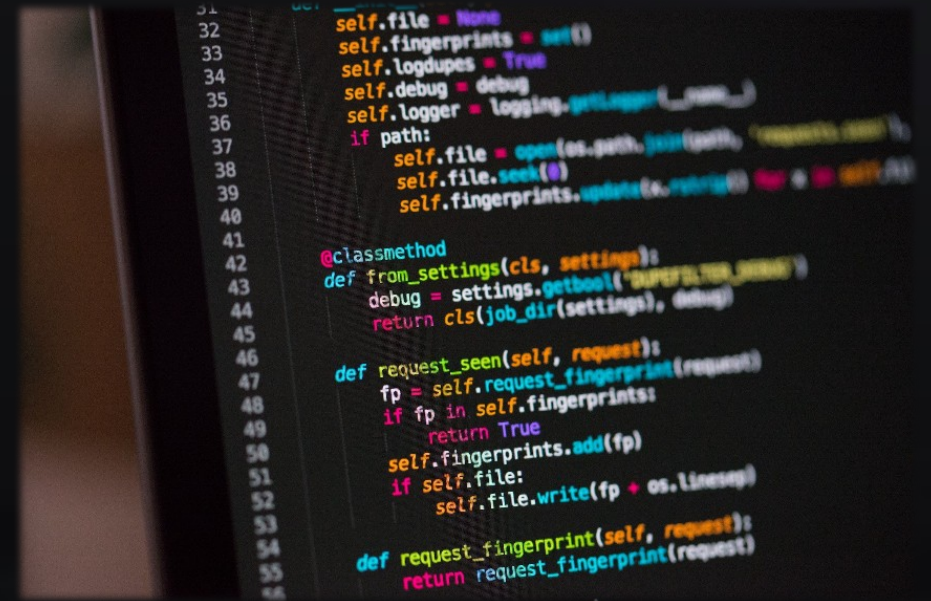
Compte Rendu SAE S2.02

Types d'algorithmes :

- Force brute
- Backtracking
- Recherche locale
- Parcours de graphes

Nos choix :

- **Backtracking**
- 1 en programmation **objet**
- 1 en programmation **procédurale**



```
31
32 self.file = None
33 self.fingerprints = set()
34 self.logdupes = True
35 self.debug = debug
36 self.logger = logging.getLogger(__name__)
37 if path:
38     self.file = open(os.path.join(path, "requests.txt"),
39                     self.file.seek(0)
40     self.fingerprints.update(is_request)
41
42 @classmethod
43 def from_settings(cls, settings):
44     debug = settings.getbool("SUPERFINGER_DEBUG")
45     return cls(job_dir(settings), debug)
46
47 def request_seen(self, request):
48     fp = self.request_fingerprint(request)
49     if fp in self.fingerprints:
50         return True
51     self.fingerprints.add(fp)
52     if self.file:
53         self.file.write(fp + os.linesep)
54
55 def request_fingerprint(self, request):
56     return request_fingerprint(request)
```

Compte Rendu SAE S2.02

1^{er} Programme – BackTracking & Programmation objet

Classe « Plateau » :

- **Tableau** représentant positions **libres**
- Méthodes pour **poser une reine**, trouver **positions libres**, **copier** une instance

Classe « JeuReines » :

- Utilise la classe « **Plateau** »
- Méthodes pour **trouver les solutions**, supprimer les solutions en **doublon**, **afficher**
- Utiliser la **récursivité**

Améliorations : **diviser** chemin au départ, **garder en mémoire** les cas déjà parcourus

Compte Rendu SAE S2.02

2^e Programme – BackTracking & Programmation procédurale

Algorithme procédurale/récuratif :

- Utilisation **grille**
- Fonction « BonnePos » pour **poser une reine**
- Si on place une reine dans la **dernière colonne** → solution
- Sinon mauvais → **BackTracking**

Complexité :

- Mauvaise complexité → **$O(n^2)$**
- Vient de la **nature du problème** des 8 reines
- Augmentation **rapide** de la complexité → courbe **exponentielle**

Améliorations : supprimer **vérification** colonne, ne pas **sauvegarder** mauvais **paternes**

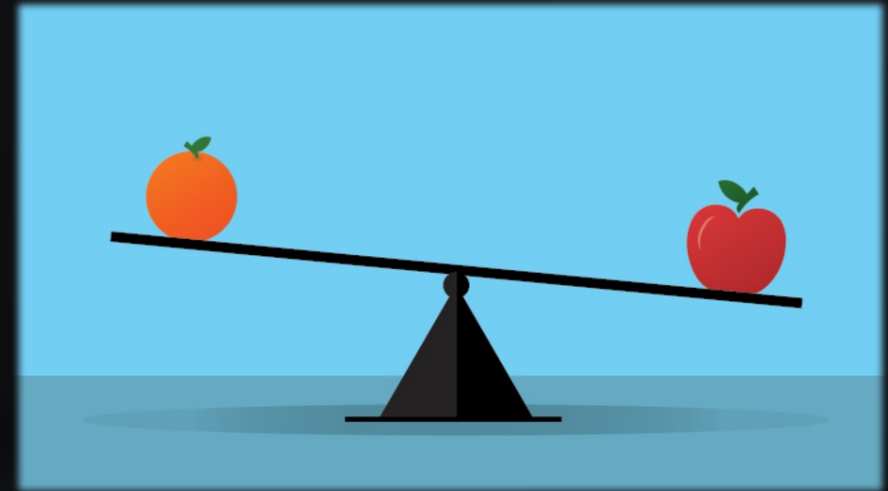
Compte Rendu SAE S2.02

Comparaison

Second algorithme plus **performant** :

- Évite **redondance** d'un même cas
- Moins de **calcul** pour trouver cases libres

→ **Temps d'exécution** bien plus court



Compte Rendu SAE S2.02

Conclusion

Optimisations possibles mais algorithmes à tendance **exponentielle**

→ Temps d'exécution **long**

→ Complexe d'obtenir les solutions **rapidement** sur un échiquier de **grande taille**

