

**SAE 1.02 : Comparaison d’approches algorithmiques**

22/01/2023

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

BASOL Nathan LEDERREY Lussandre

INFO1A2

**Introduction**

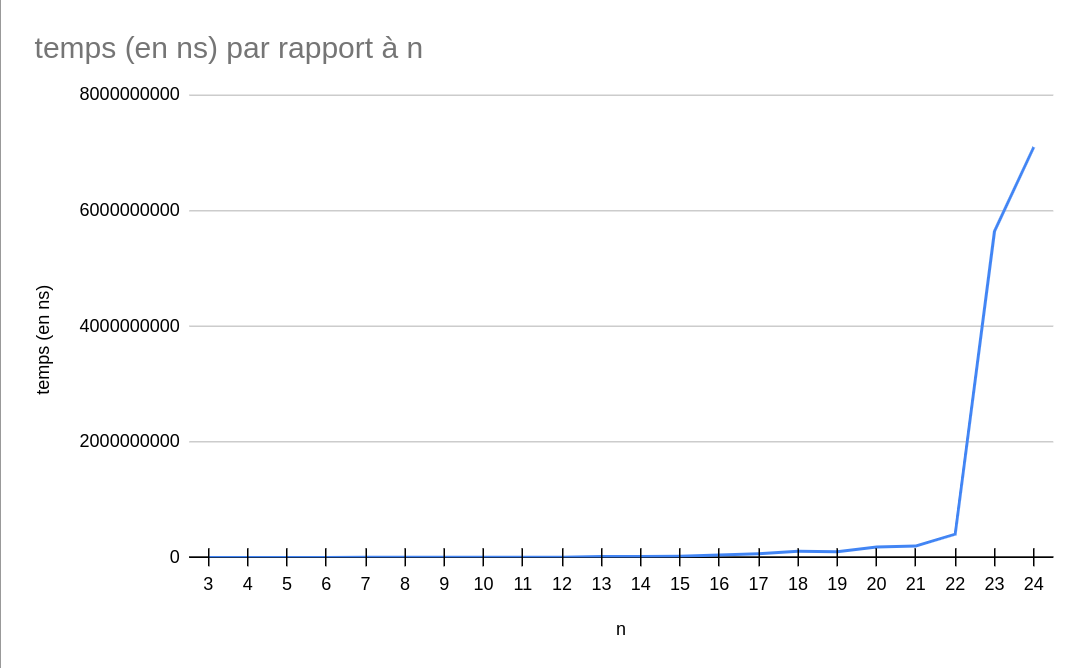
Dans cette SAE, nous reprenons le jeu « Grundy » que nous avions codé pendant la SAE d’octobre (S1.01).

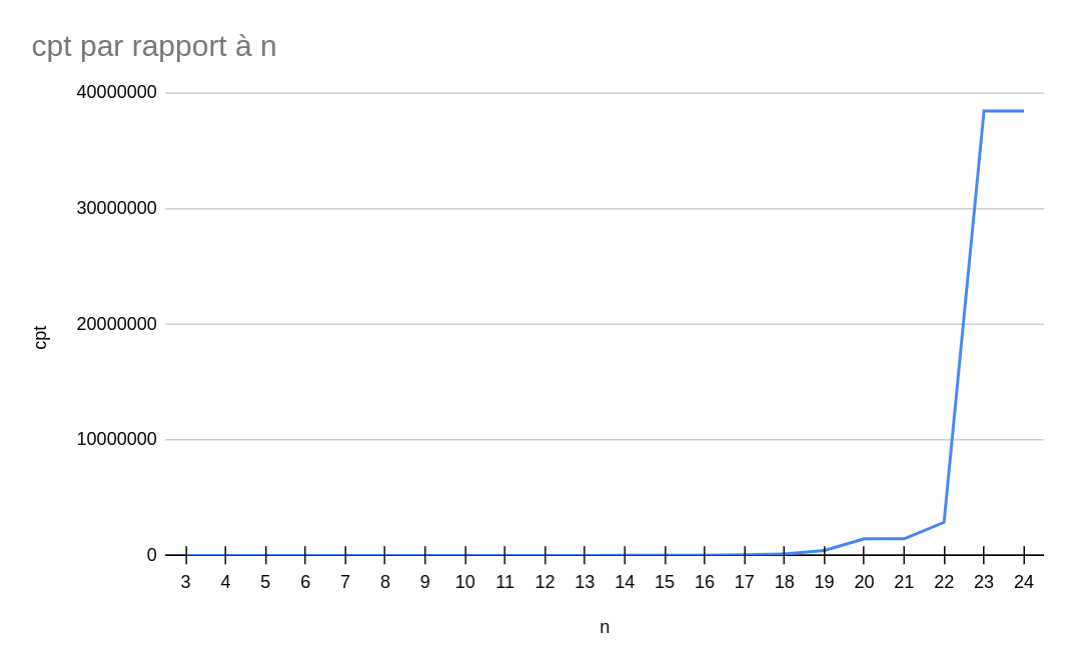
On part d’une classe de départ réalisée par Monsieur Kamp qui contient une méthode jouerGagnant qui apelle une méthode récursive estPerdant. Comme son nom l’indique, cette méthode va choisir le meilleur coup à jouer pour l’IA.

On étudiera à l’aide de graphiques l’efficacité de la méthode estGagnante en fonction du nombres d’allumettes de départ (évolution du temps d’éxécution et de la variable cpt en fonction de n).

Nous allons réaliser plusieurs classes différentes qui vont à chaque fois améliorer la précédente pour optimiser le code afin que l’IA soit plus rapide pour jouer le meilleur coup.

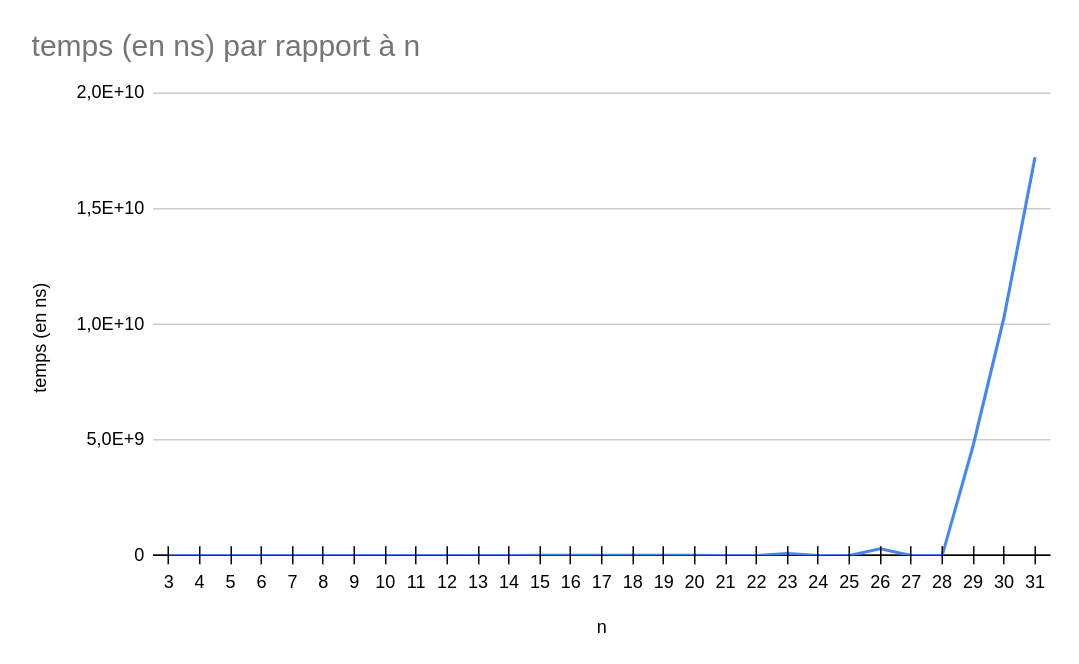
**Grundy version 0**

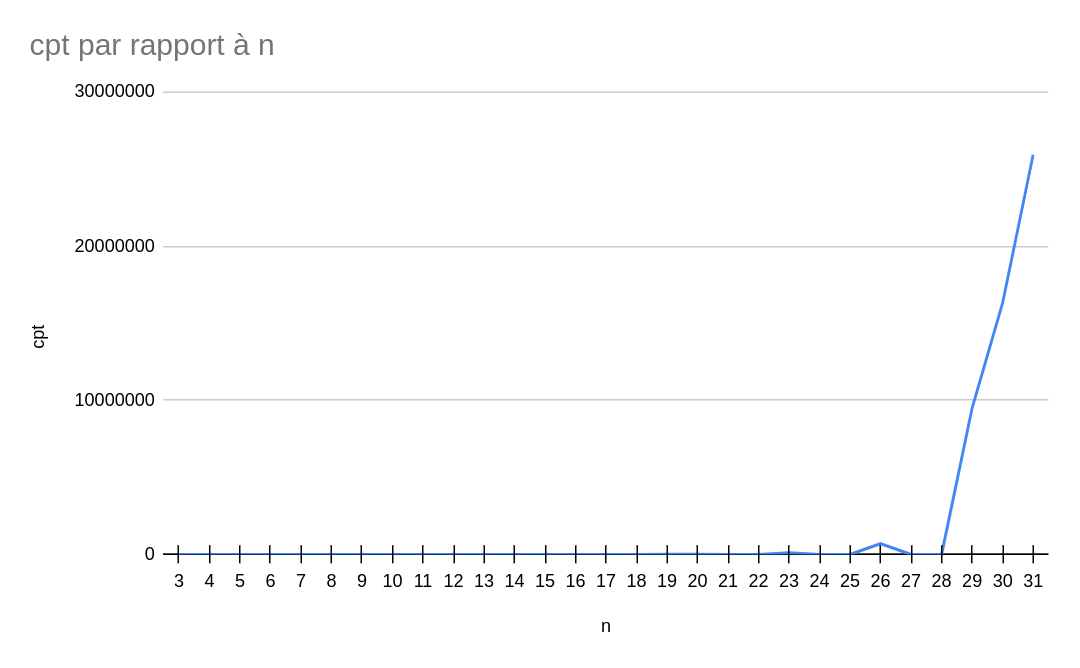




Par ces deux graphiques, on peut voir que le temps d’éxécution est presque instantané jusqu’à 22 allumettes. A partir de 23 allumettes, le temps d’éxécution devient long (6 seconde pour 23 et 24) puis devient TRES LONG pour un nombre d’allumettes encore plus élevé. On voit aussi que la valeur du cpt augmente de la même manière.

**Grundy version 1**

****

****

Cette version améliore le temps d’éxécution pour un nombre d’allumettes de départ plus élevé qu’avant. A partir de 29 allumettes, le temps d’éxécution devient beaucoup trop long. En gros tton ptn tabkeau de ses morts il contientv les positions perdantes, donc quand y’a que des decompositiooooooons perdantes bah il les trouve et bah il fait aucune décompo et il skipe (il a perdu lol)

**Grundy version 2**

**Grundy version 3**

**Grundy version 4**

**Conclusion**

lorem ipsum truc much