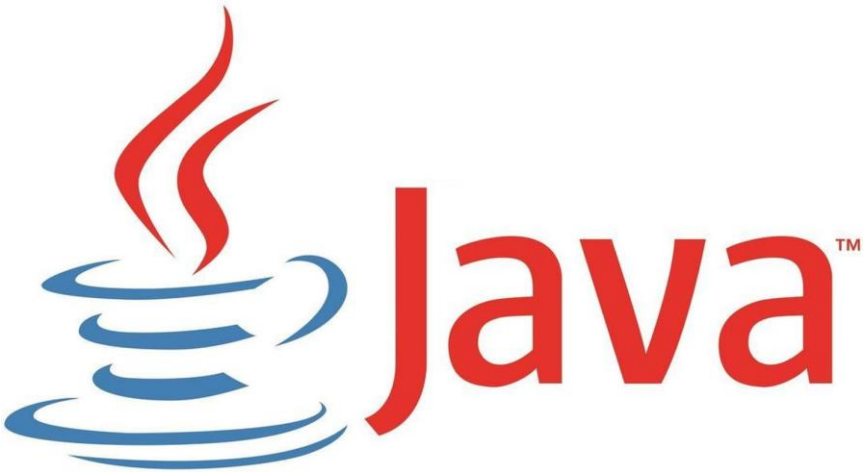
LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

RENDU SAE 1.02 :

Comparaison d'approches algorithmiques



LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

Introduction :

L'objectif principal de ce projet consiste à analyser et comparer différentes versions d'algorithmes pour résoudre le problème du jeu de Grundy, une variante du jeu de Nim. Chaque version explore une stratégie unique et est évaluée en fonction de l'efficacité, notamment le temps d'exécution et la complexité algorithmique. Cette analyse vise à mieux comprendre les mécanismes des algorithmes et à identifier les solutions les plus performantes pour ce type de problème combinatoire.

Le jeu de Grundy met en œuvre une intelligence artificielle conçue pour identifier des coups optimaux à l'aide d'algorithmes récursifs. Il s'agit d'un jeu stratégique où la prise de décision joue un rôle crucial, rendant l'étude des algorithmes particulièrement intéressante. Cinq versions distinctes ont été examinées (version0, version1, version2, version3, version4), intégrant chacune des optimisations progressives. Chaque version a été testée dans différents scénarios pour évaluer sa capacité à résoudre le problème efficacement.

Ce rapport détaille les résultats obtenus, propose une comparaison approfondie entre les versions et tire des conclusions sur l'efficacité de ces algorithmes. En outre, il identifie les points forts et les faiblesses de chaque version, offrant ainsi des pistes d'amélioration pour des développements futurs.

LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

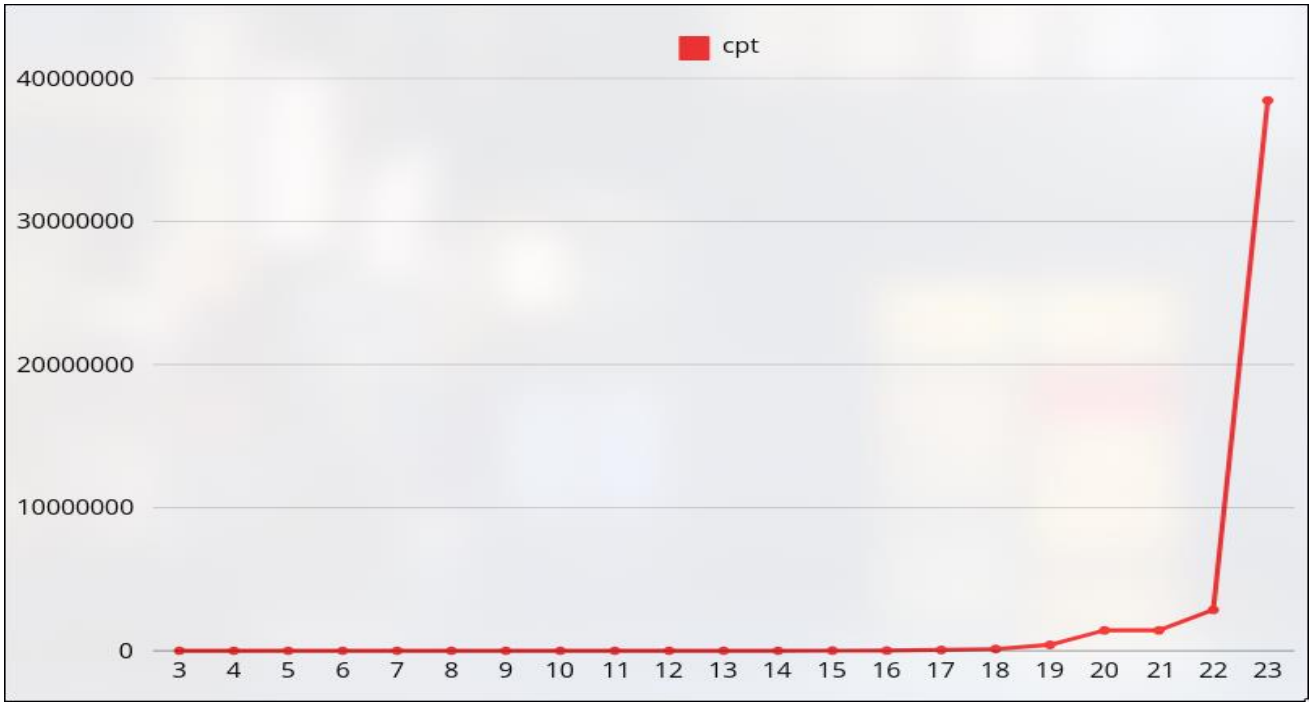
Analyses des Résultats et Comparaisons

Version0 / GrundyRecBruteEff.java :

Tableau version 0 :

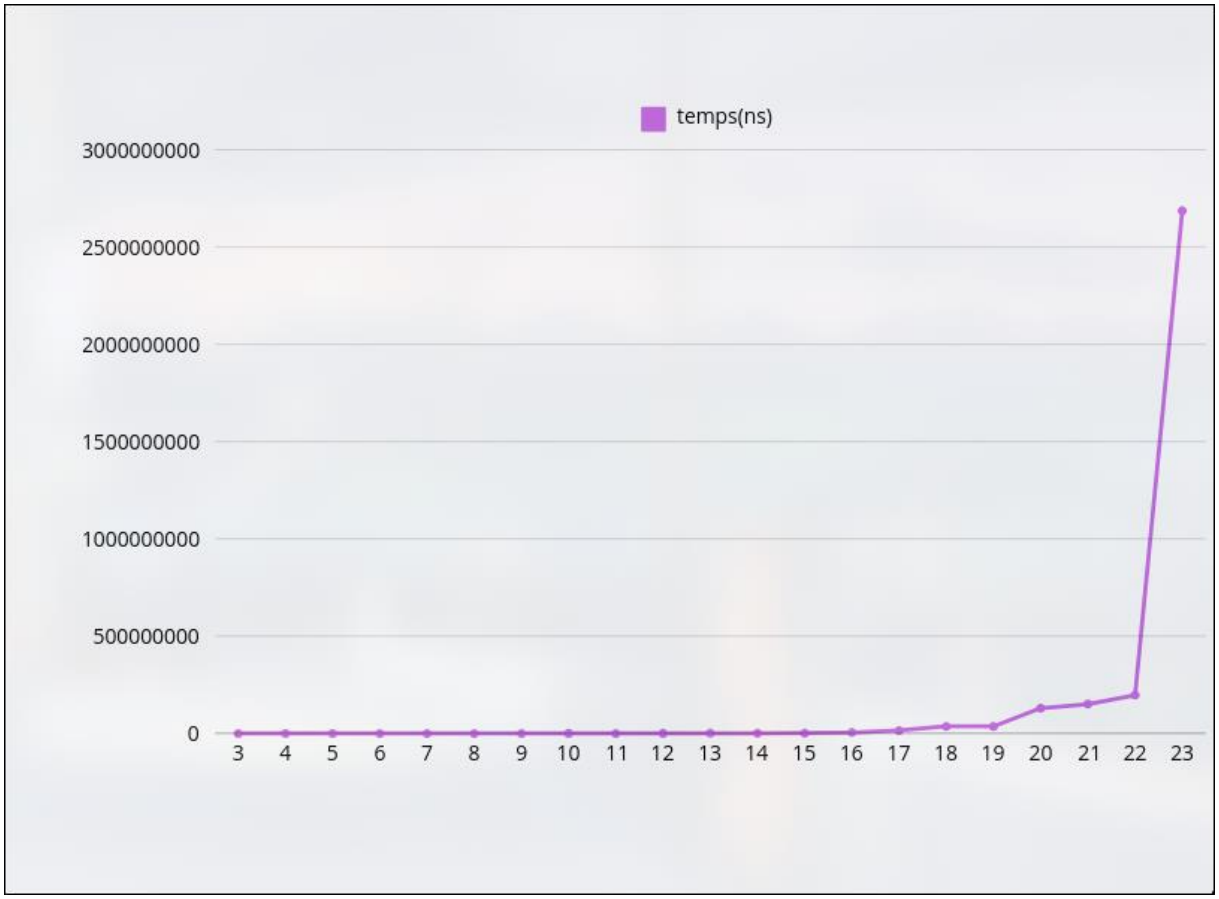
```
Résultats des tests (compteur récursif et temps d'exécution) :
------------------------------------------------------------
Allumettes      cpt                     Temps (ns)
3               1                       6702
4               2                       12052
5               3                       13654
6               7                       50862
7               18                      66850
8               19                      59577
9               39                      67602
10              112                     230856
11              113                     183531
12              227                     296445
13              1267                    810712
14              2559                    645924
15              6527                    1564620
16              20141                   4631981
17              62801                   15345277
18              128101                  36412120
19              422885                  36449938
20              1433262                 130042832
21              1433263                 150609383
22              2866527                 197107948
23              38440186                        2687253283
=== FIN DU TEST ===
```

Graphique version 0 : cpt/ allumettes



LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

Graphique version 0 : temps/ allumettes (n)



La version brute effectue une recherche récursive sans faire appel à aucune forme de mémorisation. Cela implique qu'elle examine minutieusement toutes les configurations possibles pour établir si une position précise est en état de victoire ou de défaite.

Cette mise en œuvre s'avère particulièrement peu efficace pour les valeurs hautes, à cause de la répétition superflue des calculs réalisés. Le programme, à chaque stade, réitère plusieurs fois les mêmes calculs pour des sous-problèmes identiques, entraînant ainsi une croissance exponentielle du nombre d'appels récursifs selon la dimension du problème.Par conséquent, cette méthode devient vite insoutenable lorsqu'il s'agit de gérer des jeux ou des problèmes avec une multitude de configurations possibles.

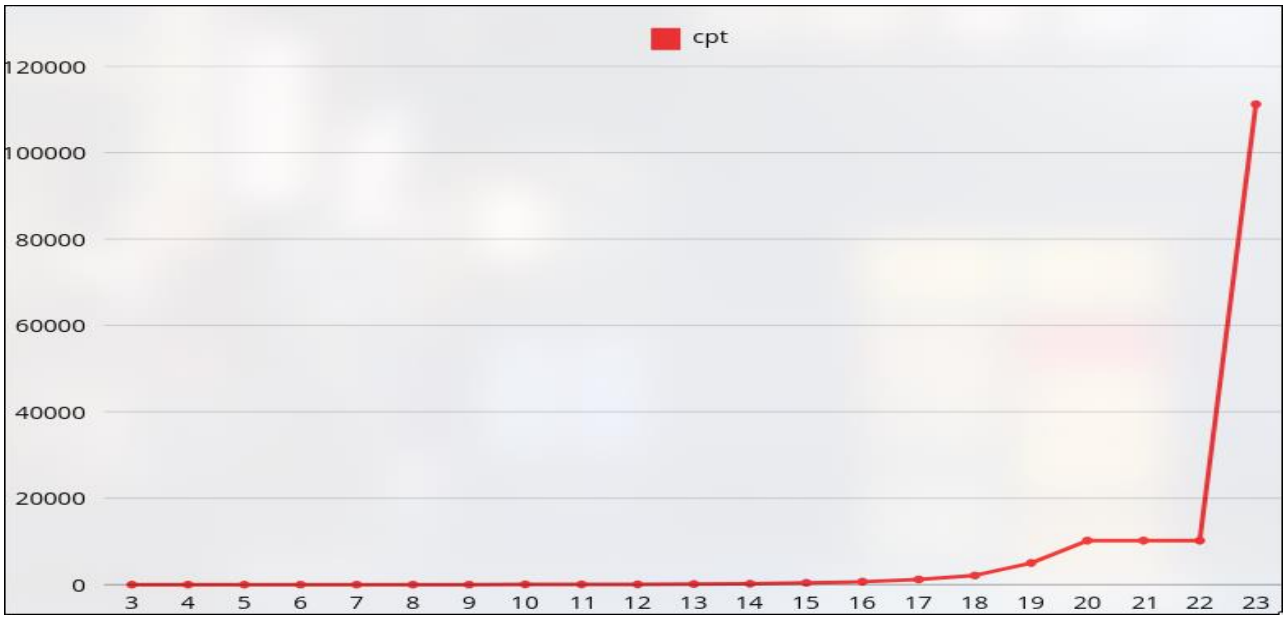
Version1 / GrundyRecPerdantes.java :

Tableau version 1 :

LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

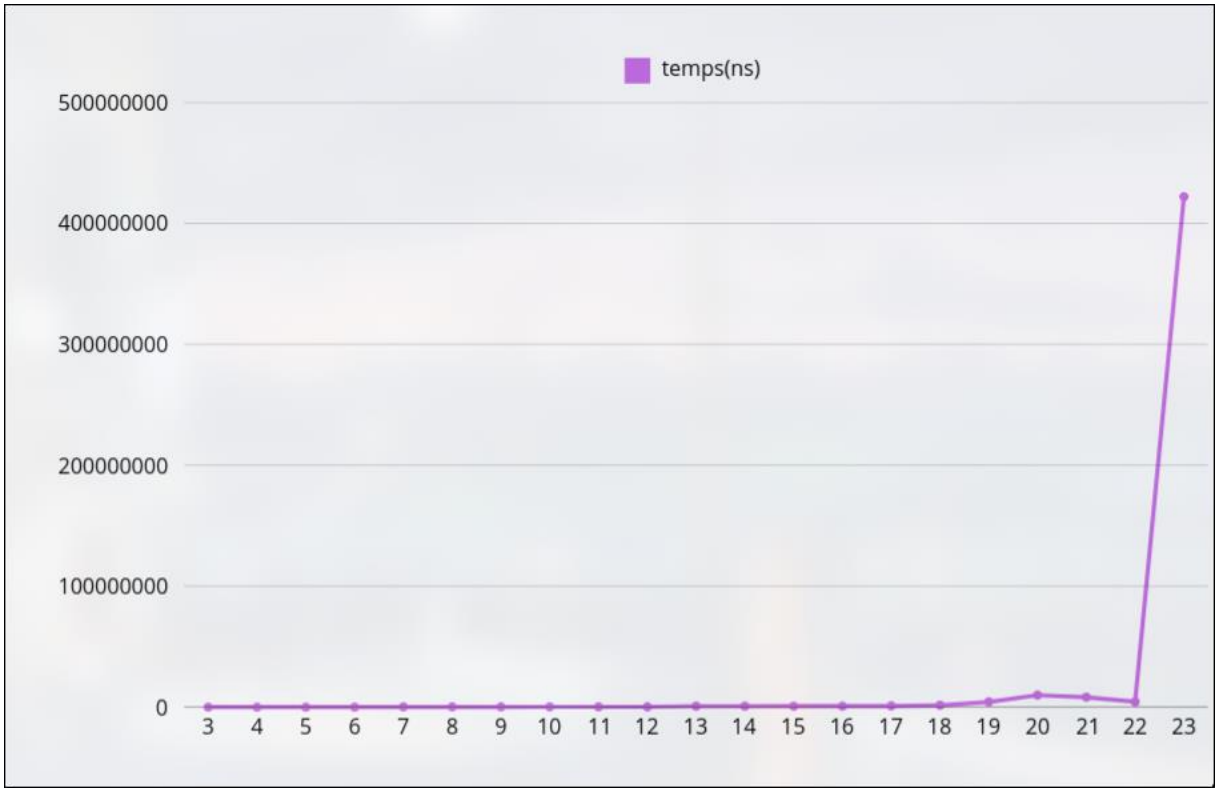
```
=== TEST EFFICACITE : EST GAGNANTE ===

Résultats des tests (compteur récursif et temps d'exécution) :
-------------------------------------------------------------
Allumettes      cpt                         Temps (ns)
3               1                           10810
4               2                           16840
5               3                           23892
6               5                           36956
7               14                          146702
8               15                          107211
9               17                          110958
10              40                          228026
11              41                          180471
12              43                          188136
13              143                         797712
14              187                         805005
15              399                         838274
16              673                         831101
17              1221                        917505
18              2139                        1539373
19              4997                        4238764
20              10176                       9861846
21              10177                       8112288
22              10179                       4433100
23              111212                      42200498
24              111213                      33971572
25              111215                      33580626
26              811220                      231210176
27              811221                      275642723
28              811223                      208815647
29              10298827                            3319594861
=== FIN DU TEST ===
```

Graphique version 1 : cpt/ allumettes(n)



LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

Graphique version 1 : temps/ allumettes (n)



Cette version introduit une optimisation majeure en mémorisant les configurations perdantes déjà calculées, ce qui permet d'éviter des recalculs inutiles, on voit donc dans ces graphiques de la version1, que le temp ou même le cpt a été réduit considérablement comparée aux graphiques de la version 0. Exemple pour n = 23 pour la v1 on a un cpt d'environ 11000 alors que pour la v0, nous avons 38 000 000 de cpt

Commentaire

- L'ajout de la mémorisation réduit considérablement le nombre d'appels récursifs et améliore significativement les temps d'exécution. Cependant, les gains deviennent moins marqués pour des valeurs très élevées de n.

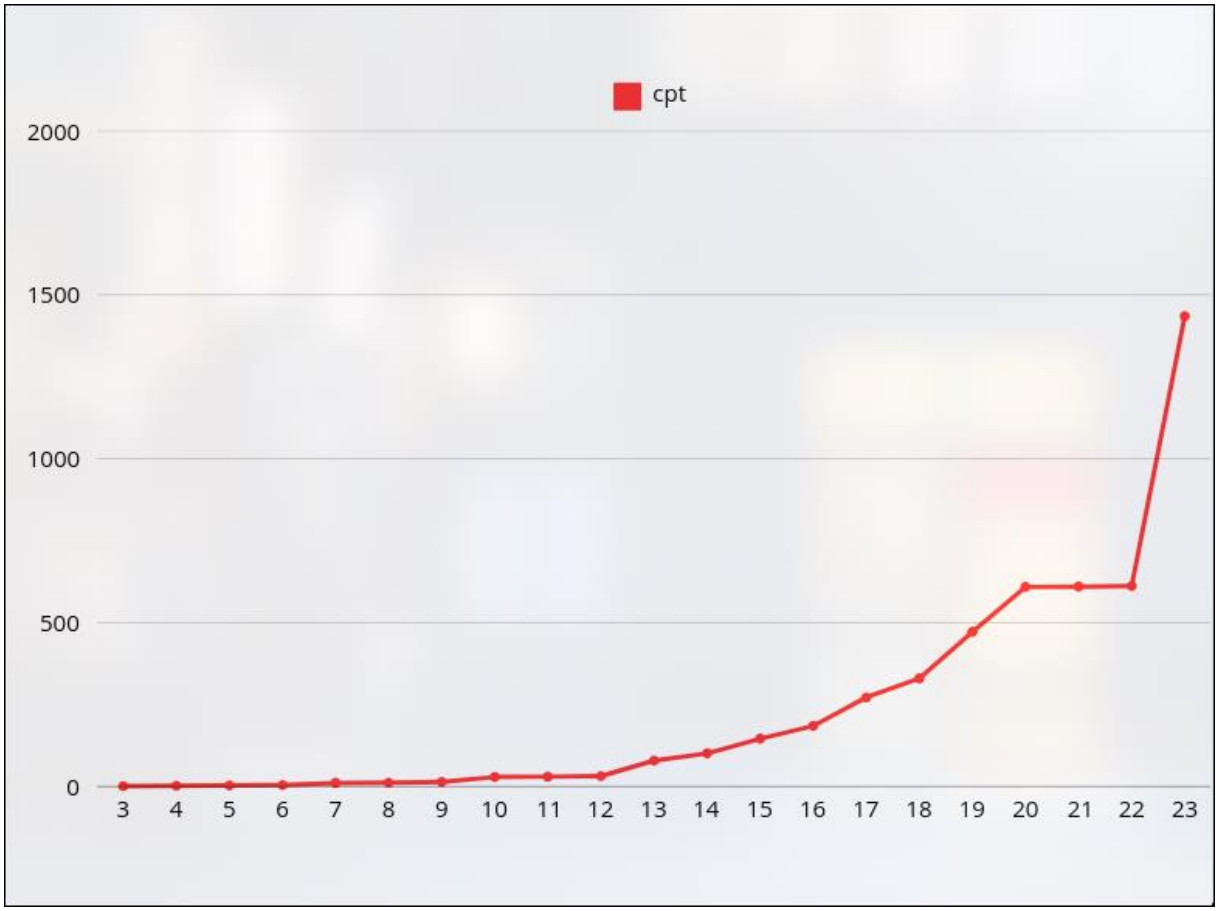
LEJAS-GRENIER Gabin 1D2
PAYEN Titouan 1D2

Version2 / GrundyRecPerdEtGagn.java :

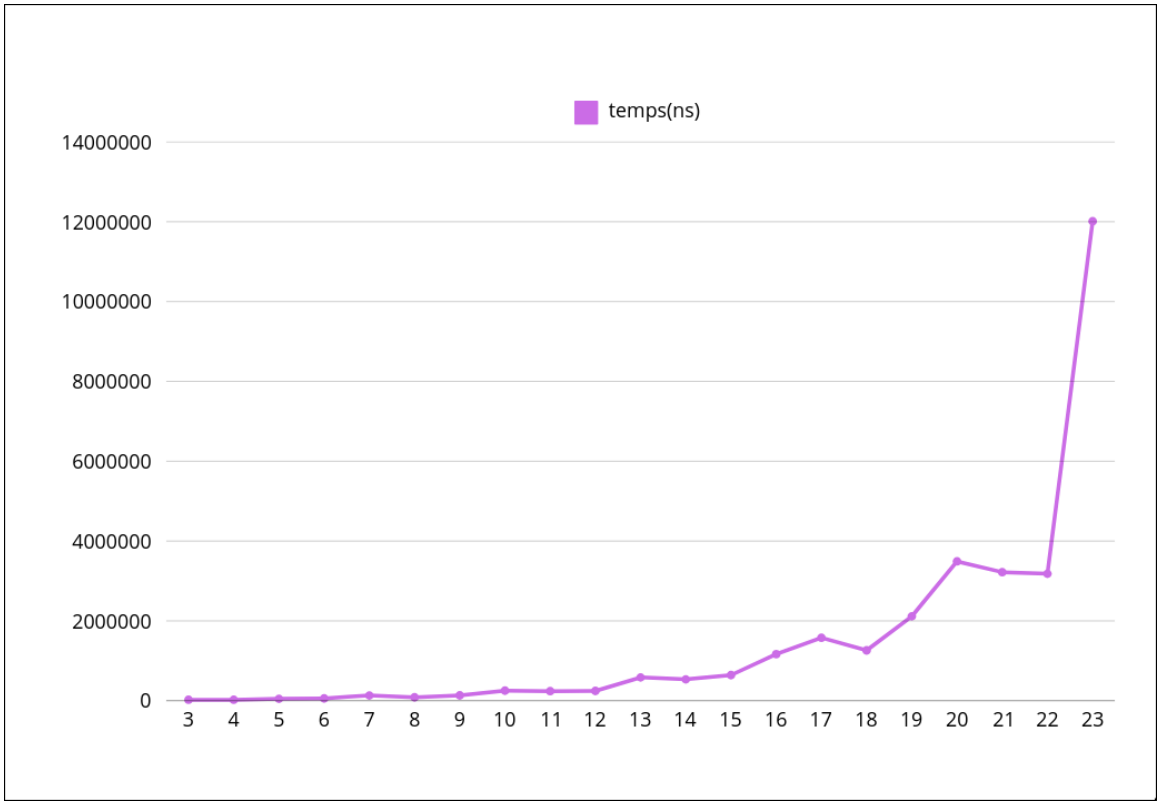
Tableau version 2 :

```
=== TEST EFFICACITE : EST GAGNANTE ===


Résultats des tests (compteur récursif et temps d'exécution) :
-------------------------------------------------------------
Allumettes      cpt                     Temps (ns)
3               1                       23724
4               2                       23644
5               3                       47367
6               5                       57576
7               11                      131613
8               12                      85698
9               14                      133766
10              29                      252586
11              30                      240774
12              32                      245302
13              79                      583856
14              101                     534866
15              146                     642374
16              185                     1167612
17              272                     1577237
18              330                     1261635
19              472                     2115689
20              609                     3491594
21              610                     3220535
22              612                     3181033
23              1435                    12013959
24              1436                    4151110
25              1438                    9709302
26              2863                    36207503
27              2864                    26005302
28              2866                    13846707
29              5894                    100540425
30              6522                    63492563
31              7614                    70674518
32              10883                   143596389
33              13090                   198564923
34              14561                   232603233
35              19843                   422737057
36              24232                   599320644
37              26842                   724189361
38              34986                   1246915335
=== FIN DU TEST ===
```

Graphique version 2 : cpt/ allumettes (n)



Graphique version 2 : temps/ allumettes (n)



Cette version retient à la fois les positions défavorables et favorables, ce qui permet de stopper les calculs plus tôt en identifiant rapidement les configurations reconnues.

En gardant en mémoire les mouvements gagnants, cette version réduit encore la quantité de calculs requise, améliorant ainsi l'efficacité générale. Les configurations complexes sont gérées de manière plus efficace, cependant les délais demeurent importants pour un nombre nombres trop grand d'allumettes.

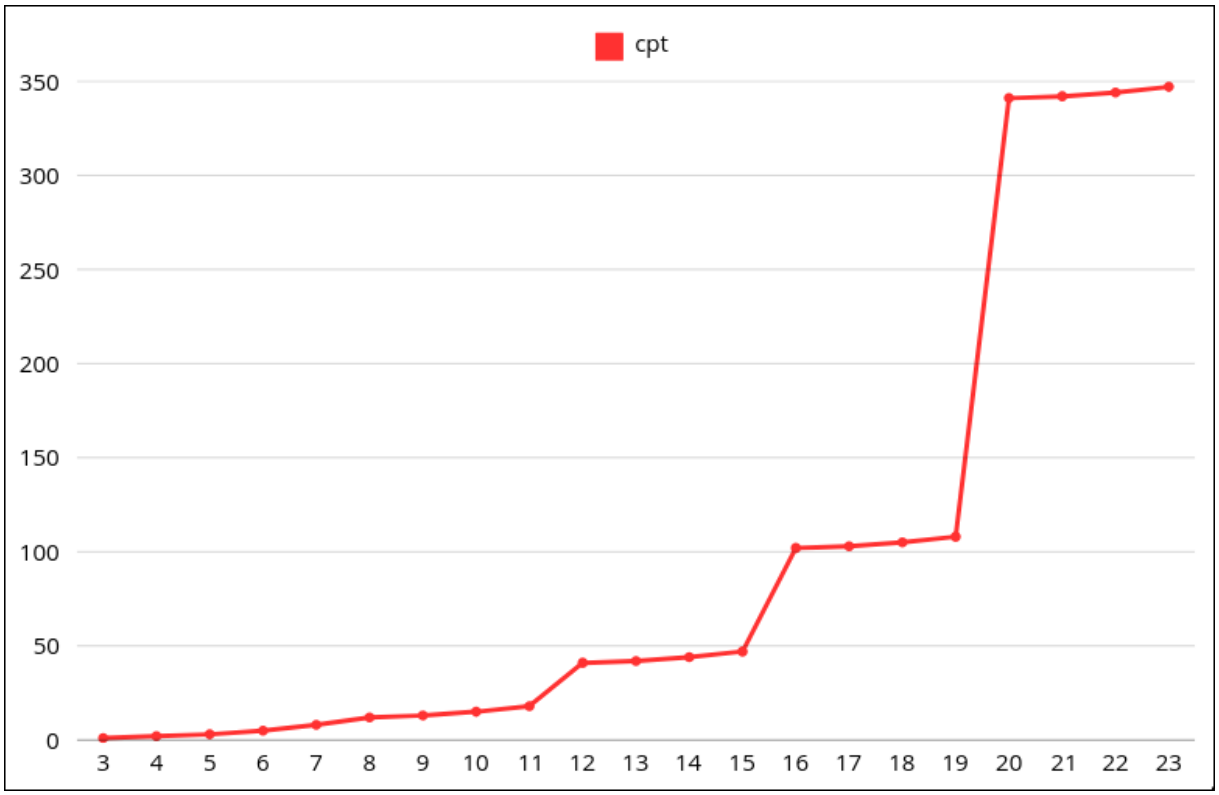
En comparant avec le graphique de la version 1, et en prenant toujours n=23, nous remarquons que, dans le graphique 1, le compteur était à 11 000, tandis que pour la version 2, il est passé à 1 400. Cela montre bien une progression.

Version3 / GrundyRecPerdantNeutre.java :

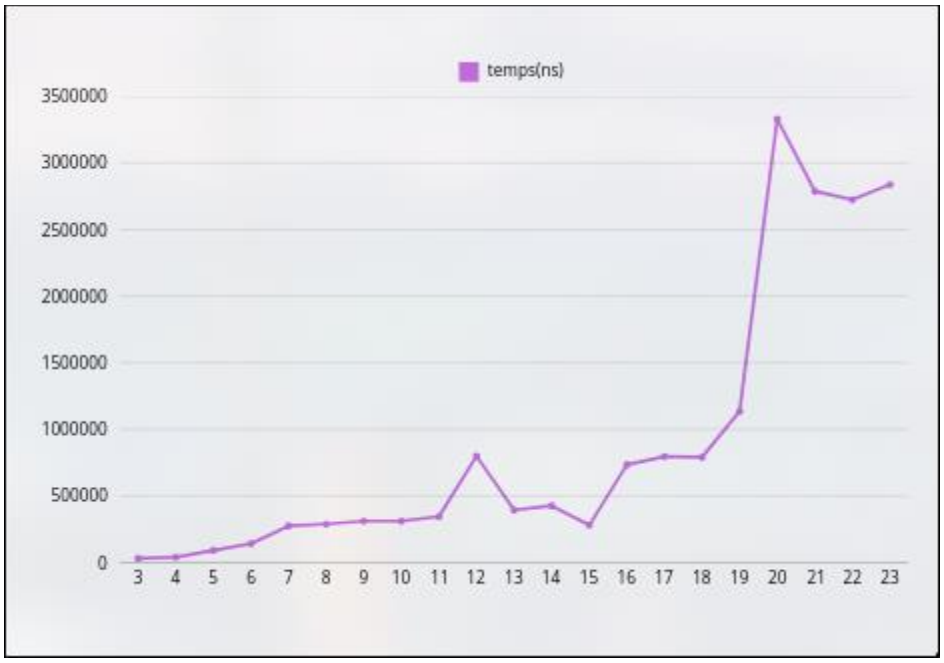
Tableau version 3 :

```
Résultats des tests (compteur récursif et temps d'exécution) :
-------------------------------------------------------------
Allumettes      cpt                     Temps (ns)
3               1                       31116
4               2                       41195
5               3                       92127
6               5                       143370
7               8                       275650
8               12                      288394
9               13                      312257
10              15                      311356
11              18                      345227
12              41                      799271
13              42                      393785
14              44                      426394
15              47                      280740
16              102                     734895
17              103                     796136
18              105                     791647
19              108                     1136975
20              341                     3329335
21              342                     2787542
22              344                     2724026
23              347                     2837502
24              855                     14878675
25              856                     6517363
26              858                     11188254
27              861                     12520483
28              2562                    41332576
29              2563                    34022323
30              2565                    18325673
31              2568                    23805533
32              5570                    133467130
33              5571                    71892501
34              5573                    69658465
35              5576                    73692499
36              13883                   402718792
37              13884                   393880816
38              13886                   399681352
39              13889                   399557116
40              31467                   1801536470
41              31468                   1769205874
42              31470                   1805832704
43              31473                   1816493992
=== FIN DU TEST ===
```

Graphique version 3 : cpt/ allumettes (n)



Graphique version 3 : temps/ allumettes (n)



Cette version utilise le théorème 3.4 utiliser dans ce document, (http://mathenjeans.free.fr/amej/edition/9903grun/grundy2.html). Pour simplifier les configurations en supprimant les piles perdantes, diminuant ainsi les calculs superflus.

En utilisant le théorème 3.4, cette version diminue les calculs répétitifs en éliminant les tas non gagnants des configurations examinées. Ceci facilite considérablement les

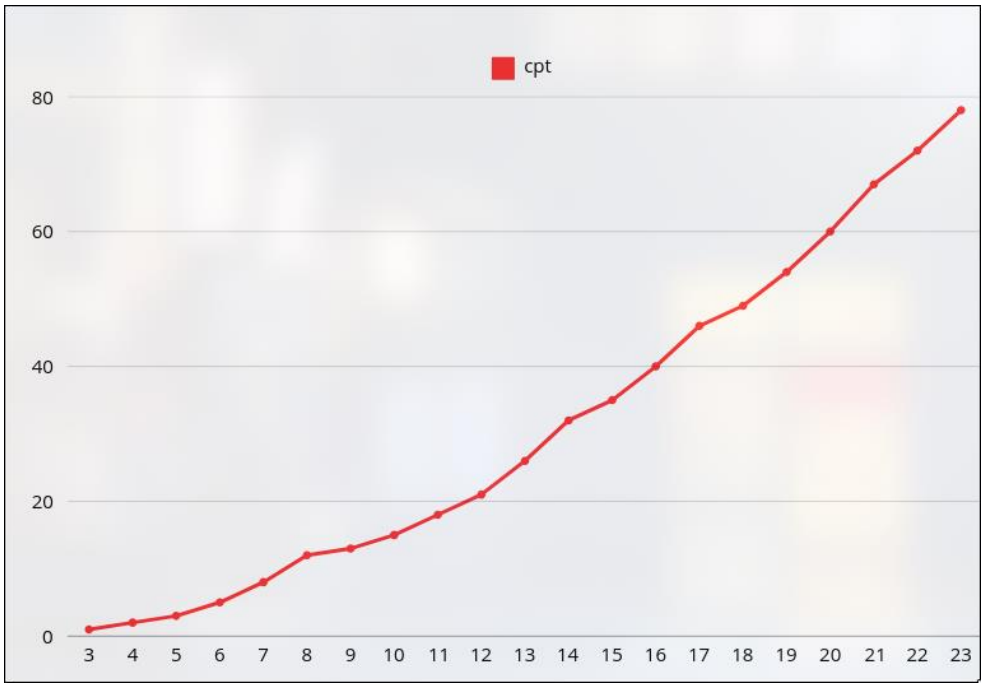
configurations, augmentant ainsi l'efficacité pour des valeurs de n faibles et modérées. Comparé à avant, nous avons gagné quelques compteurs supplémentaires : nous sommes passés de 1400 à 347. Cela représente une belle avancée, mais nous remarquons qu'il devient de plus en plus difficile de gagner en efficacité par rapport aux versions précédentes.

Version4 / GrundyRecGplusGequalsP.java :

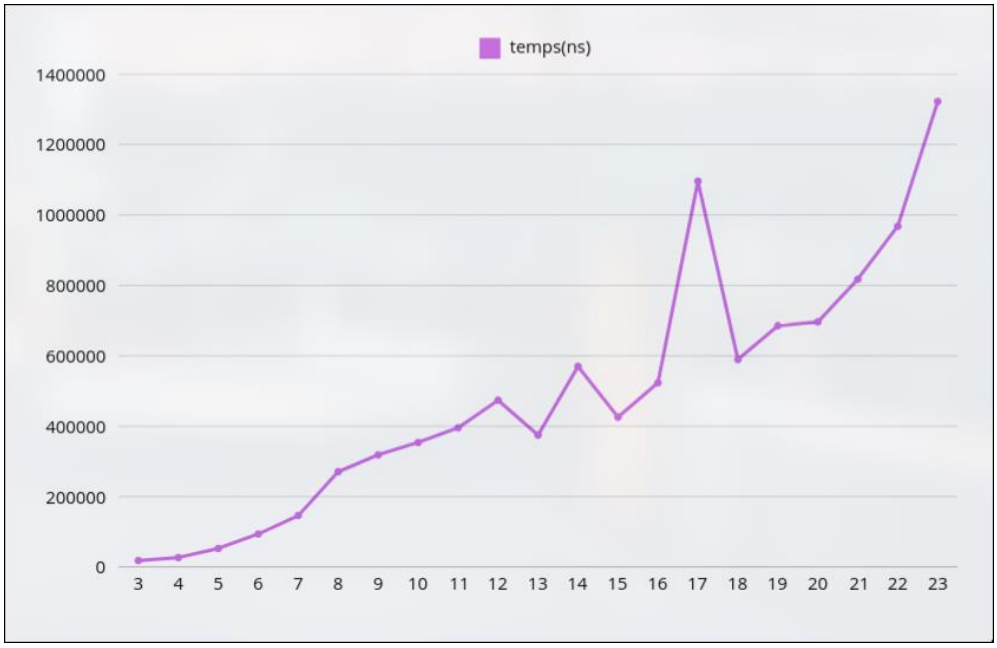
Tableau version 4 :

```
=== TEST EFFICACITE :  EST GAGNANTE ===


Résultats des tests (compteur récursif et temps d'exécution) :
------------------------------------------------------------
Allumettes      cpt                     Temps (ns)
3               1                       18443
4               2                       27209
5               3                       52995
6               5                       93940
7               8                       146075
8               12                      270870
9               13                      318807
10              15                      353940
11              18                      395666
12              21                      473747
13              26                      374818
14              32                      570082
15              35                      425820
16              40                      523457
17              46                      1095553
18              49                      589617
19              54                      685250
20              60                      696580
21              67                      817880
22              72                      968363
23              78                      1322102
24              85                      1295936
25              95                      1344253
26              105                     1567106
27              114                     2230557
28              124                     2288442
29              134                     2884410
30              144                     3002113
31              150                     3234103
32              155                     3476932
33              161                     2531942
34              164                     3821766
35              169                     1985363
36              175                     1617197
37              178                     3217843
38              183                     3212594
39              189                     3153136
40              192                     3408149
41              197                     5841552
42              203                     5761247
```

Graphique version 4 : cpt / allumettes (n)



Graphique version 4 : cpt/ allumettes (n)



Cette version se fonde sur une optimisation qui utilise la règle G + G = P, facilitant la fusion de lots gagnants avec des lots perdants et diminuant ainsi considérablement le nombre de calculs requis. Cette méthode, en identifiant promptement ces combinaisons, optimise la rapidité d'examen des configurations sophistiquées.

Cette version offre une efficacité supérieure en appliquant la règle G + G = P, qui diminue considérablement le nombre de configurations à examiner. Les performances affichent une amélioration significative, même pour des configurations complexes.

Cette version marque une progression significative dans la gestion des configurations gagnantes et perdantes.

Conclusion :

Conclusion sur l'efficacité des algorithmes (Versions 0 à 4)

L'évaluation des différentes versions de l'algorithme du jeu de Grundy montre une amélioration progressive des performances :

- **Version 0 (GrundyRecBruteEff)** : Algorithme brut, inefficace pour n>15-20 en raison d'une explosion du nombre d'appels récursifs.
- **Version 1 (GrundyRecPerdantes)** : Introduction du stockage des configurations perdantes, réduisant les recalculs, mais avec des gains limités pour n élevé.
- **Version 2 (GrundyRecPerdEtGagn)** : Ajout des configurations gagnantes, améliorant les performances, mais toujours tributaire de la récursivité.
- **Version 3 (GrundyRecPerdantNeutre)** : Application du théorème 3.4, simplifiant l'analyse des configurations avec des gains notables pour n modéré.
- **Version 4 (GrundyRecGplusGequalsP)** : Version la plus efficace grâce à la règle G+G=PG+G=PG+G=P, réduisant significativement les appels récursifs et simplifiant les calculs, en utilisant une règle déjà connue.

Chaque version a apporté des optimisations importantes, réduisant progressivement les redondances et augmentant l'efficacité.