Dichotomie sur les indices

On ne prend plus l’étendue des distances entre la distance minimale et la distance maximale, mais on classe l’ensemble des distances existantes par ordre croissant

On applique alors une dichotomie sur les indices.

Dichotomie sur les indices

Si avant la dichotomie choisissait une distance,
elle choisit donc maintenant un indice.
Au lieu de tester la distance 120, on teste l’indice
120 qui correspond à la distance 117 par exemple

Instances	Liste de distances évitées	Étendue de valeurs
pmed1	23 ; [274;276] ; 279 ; 284 ; 286 ; 287 ; 289 ; 291 ; [293;297]	299 → 284
pmed2	13 ; 16 ; 259 ; [269;271] ; 277 ; 283 ; [288;297] ; [299;301] ; [303;315]	316 → 282
pmed3	2 ; 258 ; 267 ; 271 ; 280 ; 284 ; 292 ; 298 ; 300 ; 302 ; 306 ; 309 ; 311 ; 312 ; 319 ; 323 ; [325;328] ; 331 ; 333 ; 334 ; 336 ; [338;367] ; [369;387]	388 → 316
pmed4	1 ; 2 ; 4 ; 21 ; 26 ; 32 ; 283 ; 290 ; 291 ; 293 ; 295 ; 296 ; [298;306] ; [309;326] ; [328;331] ; 333 ; 334	335 → 289
pmed5	239 ; 249 ; 250 ; 255 ; 256 ; 260 ; 263 ; [266;277] ; [279;282] ; [284;311]	312 → 261
pmed6	175 ; 176 ; 184 ; [188;190] ; 193 ; 194 ; 196 ; 197	198 → 188
pmed7	163 ; 167 ; 170 ; 171 ; [173;182]	184 → 170
pmed8	189 ; 192 ; 198 ; 201 ; 203 ; 204 ; 207 ; [209;212] ; [215;219]	220 → 204
pmed9	179 ; 182 ; [187;194] ; 196 ; [199;213]	215 → 189
pmed10	156 ; 157 ; 163 ; 164 ; 165 ; 166 ; 167	169 → 162
pmed11	126 ; 127 ; 128 ; 130 ; 132	134 → 129
pmed12	136 ; 144 ; 152 ; 155 ; [158;166]	167 → 154
pmed13	130 ; 135 ; 137 ; [139;141] ; [143;145] ; 147 ; 149	150 → 139
pmed14	156 ; [158;160] ; [162;164] ; 166 ; 167 ; [169;178]	179 → 160
pmed15	122 ; 128 ; 130 ; [132;134]	136 → 130
pmed16	100 ; 101 ; 104 ; 106	107 → 103
pmed17	98 ; 104	105 → 103
pmed18	[115;117] ; [119;121] ; 123 ; [125;141]	141 → 118
pmed19	100	101 → 100
pmed20	111 ; 112	113 → 111
pmed21	84 ; 88 ; 89	91 → 88
pmed22	105 ; 111	113 → 111
pmed23		94 → 94
pmed24	91 ; [95;97] ; 99	100 → 95
pmed25	[98;100]	102 → 99
pmed26	[82;86]	87 → 82
pmed27	87	91 → 90
pmed28	104 ; [107;109]	110 → 106
pmed29	85	88 → 87
pmed30	95	96 → 95
pmed31		65 → 65
pmed32	75 ; 77 ; 80 ; 117 ; 119 ; 121 ; 123	124 → 117
pmed33	[71;73]	74 → 71
pmed34	93 ; 94 ; 96 ; 97	98 → 94
pmed35	[69;73]	74 → 69
pmed36		87 → 87
pmed37	77	78 → 77
pmed38	76 ; 79 ; 82 ; 83	84 → 80
pmed39	[59;73] ; [75;78] ; 112	115 → 95
pmed40	67	69 → 68

Mais les résultats ne sont pas significativement différents...

Comment l’expliquer ?

Intervalles de valeurs	[0 ; 25%]	[25 ; 50%]	[50 ; 75 %]	[75 ; 100%]		Total
Nombre distances supprimées	10	0	16	402		428
Rapport	2,34 %	0,00 %	3,74 %	93,93 %		100,00 %
Nombre de présences de l’opti	24	16	0	0		40
Rapport	60,00 %	40,00 %	0,00 %	0,00 %		100,00 %

Les valeurs supprimées ne sont que dans la moitié supérieure de l’ensemble des valeurs, tandis que l’optimum est dans la moitié inférieure. Autrement dit, en “une dichotomie”, on ignorait déjà toutes les valeurs inexistantes

Me rapprocher des résultats de Hua

On en était encore loin…
L’hypothèse qui avait été prononcée était la présence d’une upper bound que je ne faisais pas mais que lui avait implémenté.

J’ai donc repris ligne à ligne son code et noté dans un fichier l’état de ce qu’il se passait à chaque ligne.

Verdict

La première différence que j’ai rencontré dans son code impliquerait que la différence de résultat soit “logique” puisqu’il ne suit pas à 100 % ce qui est indiqué dans l’algorithme de son article…

Rappel de son algorithme

for $i = |U|$ to 1 **do**
  Let $totalScore = 0$;
  for each nonempty IS $S_j \in \Pi$ **do**
    $totalScore \leftarrow totalScore + \delta(S_j, u_i)$;
  end for
  if $totalScore \geq 0$ **then**
    $P \leftarrow P \cup \{u_i\}$;
    for each nonempty $S_j \in \Pi$ **do**
      if $\delta(S_j, u_i) > 0$ **then**
        insert u_i into S_j;
      end if
      if $\delta(S_j, u_i) < 0$ **then**
        remove $|N(u_i) \cap S_j|$-1 conf. vertices from S_j
      end if
    end for
    if u_i hasn't been inserted into any nonempty IS **then**
      insert u_i into the first empty IS S_i;
    end if
  end if
end for

The first step to identify P

The essence of the first step of Algorithm 3 is to construct a subgraph $G[P]$ ($P \subseteq U$) with a MDS of size greater than or equal to $|D^*| - |D|$. To derive P, we maintain k ISs in Π, where k is a fixed parameter and each IS is an IS of $G[P]^2$. Let $u_1 < u_2 < \cdots < u_{|U|}$ be the natural ordering of vertices in U. We try to insert each $u_i \in U$ into P from $i = |U|$ to 1. A vertex u_i is allowed to join into many ISs of Π.

Voici l’ordre qu’il utilise dans son code...

```
 CFG={1^ 167 159 115 32 31 23 9 2 40^ 29 39 141 123 121 114 86 55 41 176^ 25 139 140 148 189^ 7 21 28 42 50 67 102 107 110 111 125 129 130 134 149 175 188 191 190 192^ || 46 47 48 49 3
0 53 54 18 56 57 58 59 60 61 62 63 64 65 66 6 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 17 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 5 103 104 105 106 33 108 109 34 35
112 113 16 4 116 117 118 119 120 15 122 14 124 36 126 127 128 37 38 131 132 133 12 135 136 137 138 22 26 13 142 143 144 145 146 147 24 10 150 151 152 153 154 155 156 158 3 160 161 162 1
63 164 165 166 8 168 169 170 171 172 174 19 20 178 179 180 181 182 183 184 185 186 187 11 27 44 43 45 195 196 197 198 199^ }
```

Il utilise les sommets de CFG en commençant bien par le dernier, puis prend tous les non “dominés” comme indiqué mais… CFG n’est pas rangé dans l’ordre

Du minimum Dominating Set au p-center problem

Un stage en recherche c'est quoi ?

-Lire des articles
-Faire des hypothèses
-Élaborer, implémenter les hypothèses
-Analyser les résultats et conclure
-Recommencer

Article

« An Exact Algorithm for the Minimum Dominating Set Problem » Hua Jiang, ZHIFEI Zheng

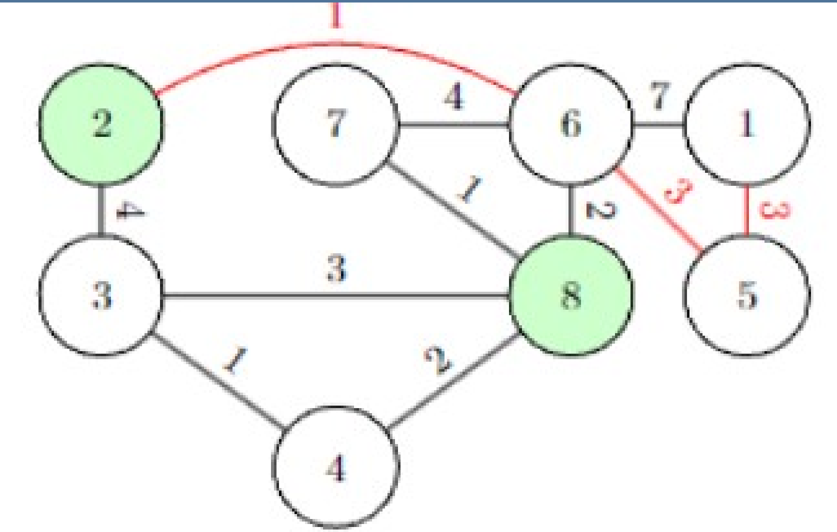
L’hypothèse / L’idée

Est-ce qu'il ne serait pas possible de transformer le problème des p-centres en problème de minimum dominating set et d'utiliser l'algorithme détaillé dans mon article ?

C'est quoi le problème des p-centres ?

Dans un graphe pondéré, l'idée est de marquer un nombre de sommets définis de tel sorte que la plus grande des plus faibles distances séparant un sommet non marqué de son sommet marqué le plus proche soit la plus faible possible

C'est quoi le problème des p-centres ?



C'est quoi le minimum dominating set ?

Dans un graphe non pondéré, un dominating set est un ensemble dont les sommets permettent de dominer tout le graphe.

Un sommet dominé ?

Un sommet est dit dominé s'il est dans l'ensemble dominant, ou s'il est voisin d'un sommet de l'ensemble dominant.

Passer du p-center problem à la résolution de plusieurs mds.

On a besoin de dépondérer le graphe, d'y enlever ses distances.

On va donc effectuer un floyd warshall et choisir une distance. Si la distance qu'on a choisie est plus grande que la distance séparant deux sommets, on a un arc.

Appliquer l'algorithme de l'article étudié

Première étape : Réduire le graphe

On va pour ce faire utiliser les propriétés énoncés dans l'article d'Alber 2006

« Experiments on data reduction for optimal domination in networks » Jochen Alber, Nadja Betzler and Rolf Niedermeier

Deuxième étape : effectuer un branch and bound.

C'est un branch ane bound un peu différent puisqu'il n'y a pas à chaque nœud de l'arbre 2 branches, mais un nombre indéfinis

Troisième étape inclue dans la deuxième : Réduire le nombre de branches

C'est l'idée même de l'algorithme de l'article de mon stage.

Troisième étape inclue dans la deuxième : Réduire le nombre de branches

En suivant divers propriétés, on va construire un ensemble de sommets tel qu'à ce stade de l'arbre, on sait qu'ils ne doivent pas être inséré dans notre dominating set

Troisième étape inclue dans la deuxième : Réduire le nombre de branches

- Independant Set
- Fonction d’évaluation

$$\delta(S_j, u) = \begin{cases} 1 & u \in \mathbb{S}_1 \wedge N_1 = \emptyset \wedge N_2 = \emptyset \\ 0 & u \in \mathbb{S}_1 \wedge N_1 = \emptyset \wedge N_2 \neq \emptyset \\ 1 - |N_1| & u \in \mathbb{S}_1 \cup \mathbb{S}_2 \wedge N_1 \neq \emptyset \\ 0 & u \in \mathbb{S}_2 \wedge N_1 = \emptyset \\ 0 & u \in \mathbb{S}_3 \wedge N_2 \neq \emptyset \\ 1 & u \in \mathbb{S}_3 \wedge N_2 = \emptyset \end{cases}$$

Résultats

Instances	Optimum connu	Bornes finales	Temps (s)
pmed1	127	127	9
pmed2	98	98	38,9
pmed3	93	93	308,7
pmed4	74	74	5,2
pmed5	48	48	0,2
pmed9	37	37	20
pmed10	20	20	0,1
pmed15	18	18	0,3
pmed20	13	13	3,3
pmed25	11	11	21,1
pmed30	9	9	19,6

Résultats avec la nouvelle méthode

instance	TreeSearch
pmed1	253
pmed2	57
pmed3	309
pmed4	0
pmed5	0

Résultats avec l'ancienne