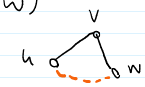
k-centres

15 septembre 2021 14:50

Points P dans un espace d-dimensionnel (d arbitraire)
Pour 2 points $u=(u_1,u_2,\dots,u_d)$ et $v=(v_1,v_2,\dots,v_d)$,
on a une métrique $D(u,v)$ qui donne une distance.
$\Rightarrow \forall u,v,w$, on a

$$D(u,w) \leq D(u,v) + D(v,w)$$

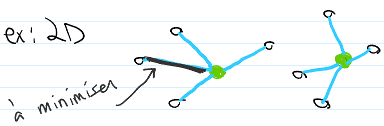


ex: dist. Euclidienne

$$D(u,v) = \sqrt{\sum_{i=1}^{d} (u_i - v_i)^2}$$

But: trouver les k pts de P les plus "centraux"
$\Rightarrow$ choisir k centres en minimisant la distance max à parcourir pour qu'un pt atteigne un centre

ex: 2D



$k=2$

<u>Entrée</u>: points P, entier k
<u>Sortie</u>: $C \subseteq P$ tel que $|C| \leq k$ qui minimise $\max_{p\in P}\left(\min_{c\in C} D(p,c)\right)$

<u>Observation</u>: $\exists p,q \in P$ tel que $OPT = D(p,q)$

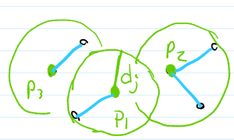
OPT est donc une valeur parmi les $\binom{n}{2}$ dists possibles. ($n=|P|$)

Soit $d_1, d_2, d_3, \dots, d_m$, $m=\binom{n}{2}$ la liste de toutes les dists possibles, triées en ordre ↗.

<u>Idée</u>: voir si $OPT=d_1$ est possible
sinon, voir si $OPT=d_2$ est possible
...
sinon, voir si $OPT=d_m$ est possible

- OPT = le plus petit d_j tel qu'on peut trouver k centres et où chaque $p\in P$ est à dist $\leq d_j$ d'un centre.

- Routine gloutonne pour voir si $OPT=d_j$ est possible




```
getCentres(P, dj)
    C = ∅
    tant que |P| > 0
        soit p ∈ P
        Soit R = { q ∈ P : D(p,q) ≤ dj }
```

Thm : l'algo kCentreApprox est une 2-approx

soit $p \in P$
Soit $R = \{ q \in P : D(p,q) \leq d_j \}$
C.insert(p)
$P = P \setminus R$

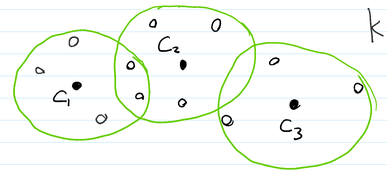
return C

Peut échouer (retourne trop de centres)
mais peut servir à approximer

kCentreApprox(P, k)
Calculer $d_1, d_2, d_3, \ldots, d_m$
pour $i = 1 .. m$
C = getCentres(P, $2 \cdot d_i$)
si $|C| \leq k$
return C

Thm: l'algo kCentreApprox est une 2-approx

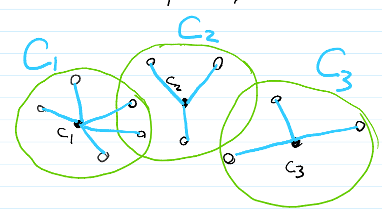
Preuve: soit $C=\{c_1, c_2, \dots, c_k\}$ les centres optimaux.
Soit $OPT = d_j$.
Donc, chaque p_i est à dist $\leq d_j$ d'un point de C.



$k=3$

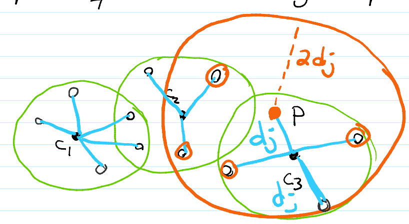
Soit C_1 les pts assignés à c_1 (dist $\leq d_j$ de c_1)
C_2 les pts assignés à c_2 (dist $\leq d_j$ de c_2 pas dans C_1)
...
C_k les pts assignés à c_k

(les C_i sont appelés clusters)



Considérons l'algo quand il reçoit $2d_j = 2OPT$.
Soit p le 1er point qu'il choisit,
et soit C_i le cluster qui contient p.

L'algo choisit p comme centre et retire tout point q à dist $\leq 2d_j$ de p.



Ceci retire tous les pts de C_i, car

$$\forall q \in C_i,\ D(p,q) \leq D(p,c_i) + D(c_i,q)$$
$$\leq d_j + d_j = 2d_j = 2OP$$

Donc, à chaque itération, l'algo retire tous les pts d'un cluster C_i (au plus).

$\Rightarrow$ l'algo fera au plus k itérations car à chaque itération, il retire les pts d'un cluster C_i, et il y a k tels cluste

$\Rightarrow$ l'algo retourne un ensemble de k centr t.q. chaque pt est à dist $\leq 2d_j$ d'un ce

T

ns.

es
entre,

→ l'algo retourne un ensemble de k cen...

t.q. chaque pt est à dist $\leq \underline{2d_j}$ d'un ce...

$APP \leq 2 \cdot d_j = 2 \cdot OPT$. $\blacksquare$

entre,