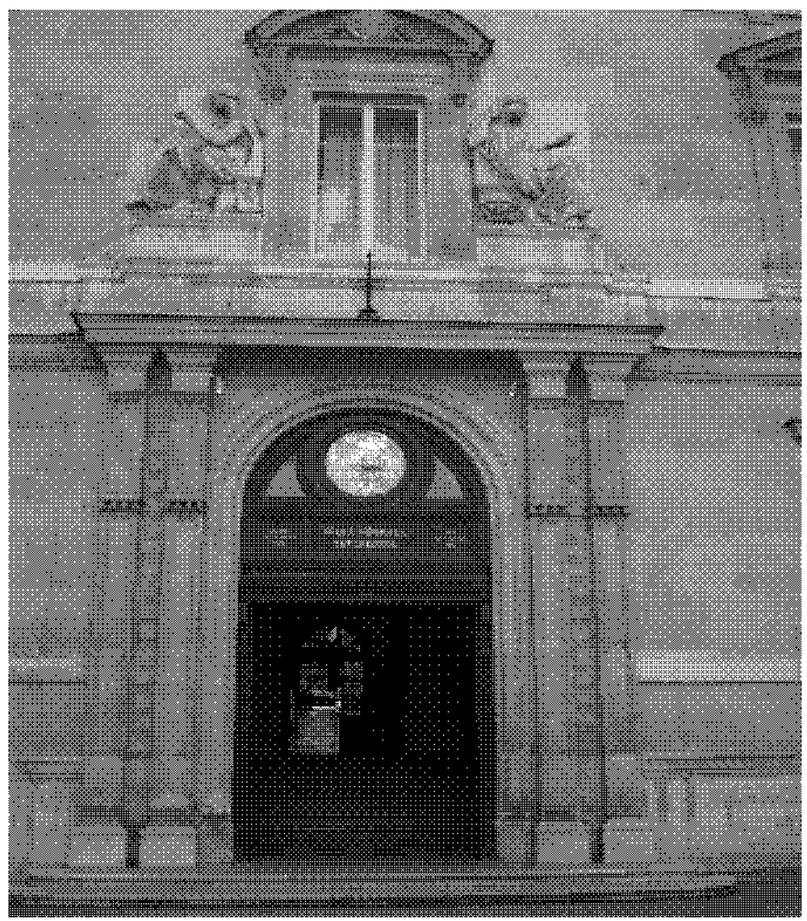
Algorithms for Approximation

Claire Mathieu
CNRS, DI-ENS and Brown



Problème

Modèle

Algorithme

Résolution constructive et efficace de problèmes

- Terminaison de programme ✗
- Arbre couvrant minimum ✓
- Voyageur de commerce ✓ → ∄ known fast solution.
- Plus court chemin ✓ → the solution I outlined is slow. but ∃ fast solution.
- Plus long chemin ✓
- Coupe maximum par recherche locale ✓
- Sac à dos, bin packing ✓
- Tri avec erreurs ✓
- Partitionnement ✓

why?

because

$P = NP \Rightarrow$ (a) we have not found it yet

$P \neq NP \Rightarrow$ (b) it does not exist.

NP-hard problems

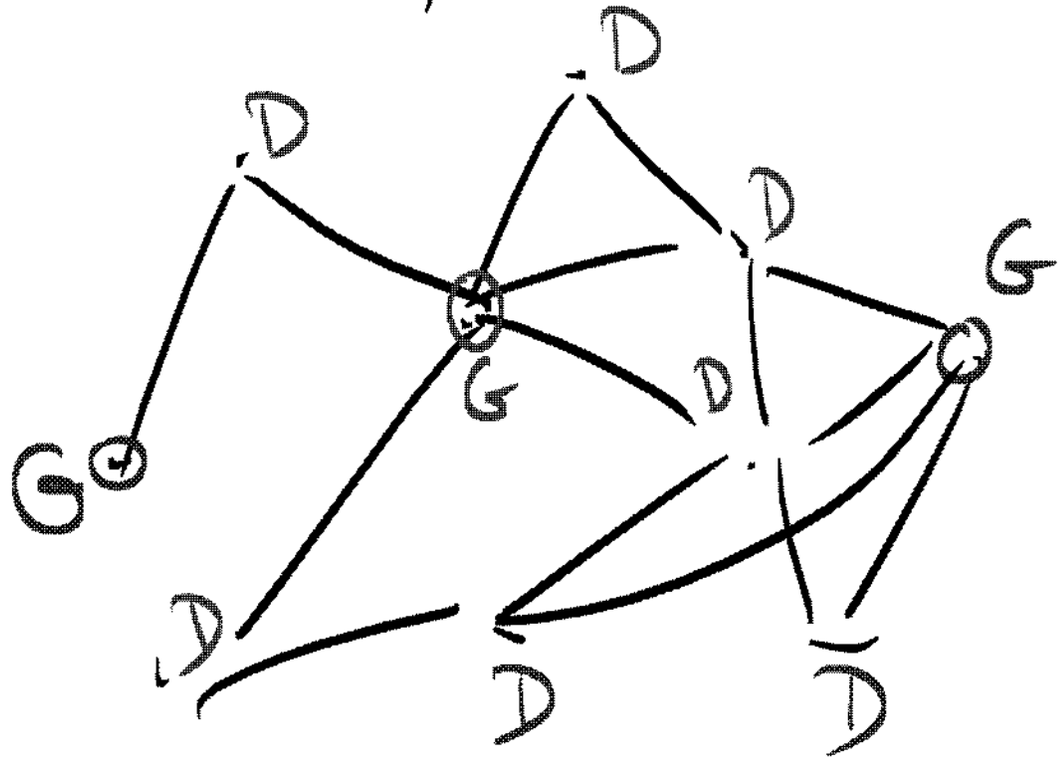
Classification of problems:

Minimum spanning tree	] easy
Shortest path	
2-SAT	
TSP	] NP-hard
longest path	
2-SAT	

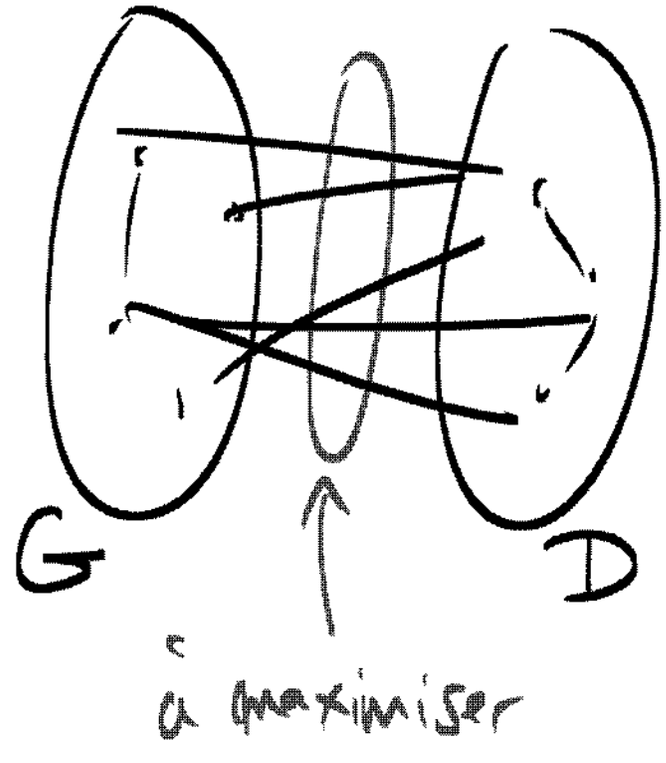
Dealing with NP-hard problems: heuristics, approximation, special cases, average-case analysis

Coupe maximale par recherche locale

graphe $G = (V, E)$



coupe



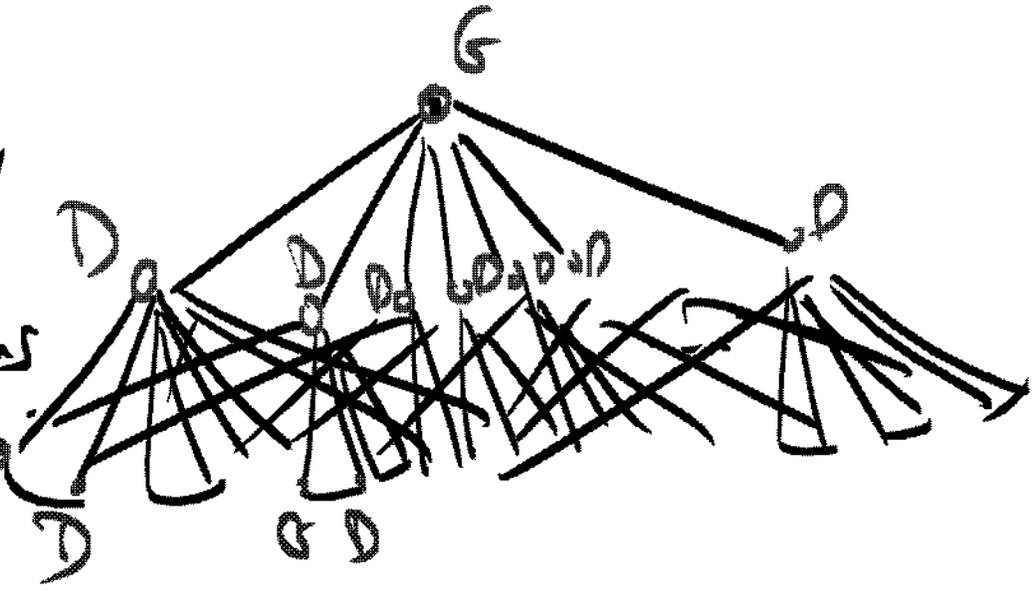
NP-difficile.

But : coup (G, D) tq. $\#\{\text{arêtes in } G \times D\} \geq \alpha \cdot OPT$

α : approximation ratio

Glouton.

Pour chaque sommet v
placer v de G ou D
selon le nb de voisins
de v étiquetés G ou D



Recherche locale

- initialement, partition arbitraire de $V = G \cup D$
- tant qu'$\exists$ vertex v tq changer v de côté améliore la taille de la coupe, changer v.

Aléatoire. $\forall v$ placer v $\begin{cases} \text{dans } G \text{ avec proba } 50\% \\ \text{dans } D \text{ " " } 50\% \end{cases}$

What is α? Valeur(coupe) $\geq \frac{1}{2}$ opt

Pourquoi? $\frac{1}{2}\,\text{opt} \leq \frac{1}{2}|E|$.

1/4	G	G	e n'est pas de la coupe
1/4	G	D	e est
1/4	D	G	e est
1/4	D	D	e n'est pas

$$\Pr(e \text{ est de la coupe}) = 1/2.$$

$$|\text{coupe}| = \sum_{e \in E} \mathbb{1}(e \text{ est de la coupe}).$$

$$\mathbb{E}|\text{coupe}| = \sum_{e \in E} \mathbb{E}(\mathbb{1}(e \text{ est de la coupe}))$$

$$= \sum_{\text{edges } e} \frac{1}{2} = \frac{|E|}{2}.$$

Improving the ratio.

$1/2 = 0.5$

1994: Goemans Williamson

0.878...

2001. Khot defined the "unique games conjecture"

≈2005 if the UGC holds
then no polynomial time algorithm
with ratio better than 0.878...

Sac à dos / Knapsack.

Bin Packing

The movers' problem.

Input: items $0 < a_1, a_2, \ldots, a_n < 1$.

Output: placement of items in bins

such that

{a_i in same bin} has total size ≤ 1.

Goal: minimize # bins used.

bin-packing is NP-hard.

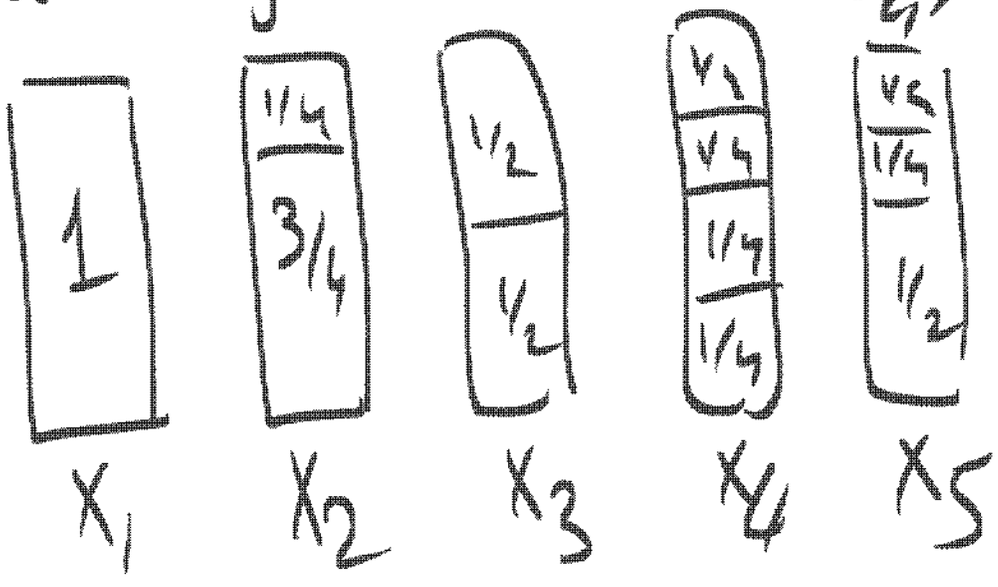
(Knapsack is NP-hard)

idea: rounding.

① Round item sizes up so that every size is an integer multiple of 0.25
② Solve bin packing for rounded instance
③ Hence a packing for initial instance.

Comment: "it doesn't work", b/c if all input items are = 0.0001 but we rounded the to 25%.

If all objects are in $\{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$.



For all $(x_1, \ldots, x_5) \in \{0, \ldots, n\}^5$
check if input fits.
pick solution s.t. # bins is minimum.

$\approx n \log n \left(\frac{1}{\varepsilon}\right)^8$

What is the approximation ratio α?

output bins $\leq \alpha \cdot$ OPT

I want $\alpha = 1.01$.

Comment: "it doesn't work", b/c if all input items are $= 0.0001$ but we rounded them to 25%.

Answer: ignore the really small items.

Comment: "it still doesn't work", b/c:

input: $\frac{1}{2} + 0.00001, \ldots, \frac{1}{2} + 0.00001$

$\frac{1}{2} - 0.00001, \ldots, \frac{1}{2} - 0.00001$.

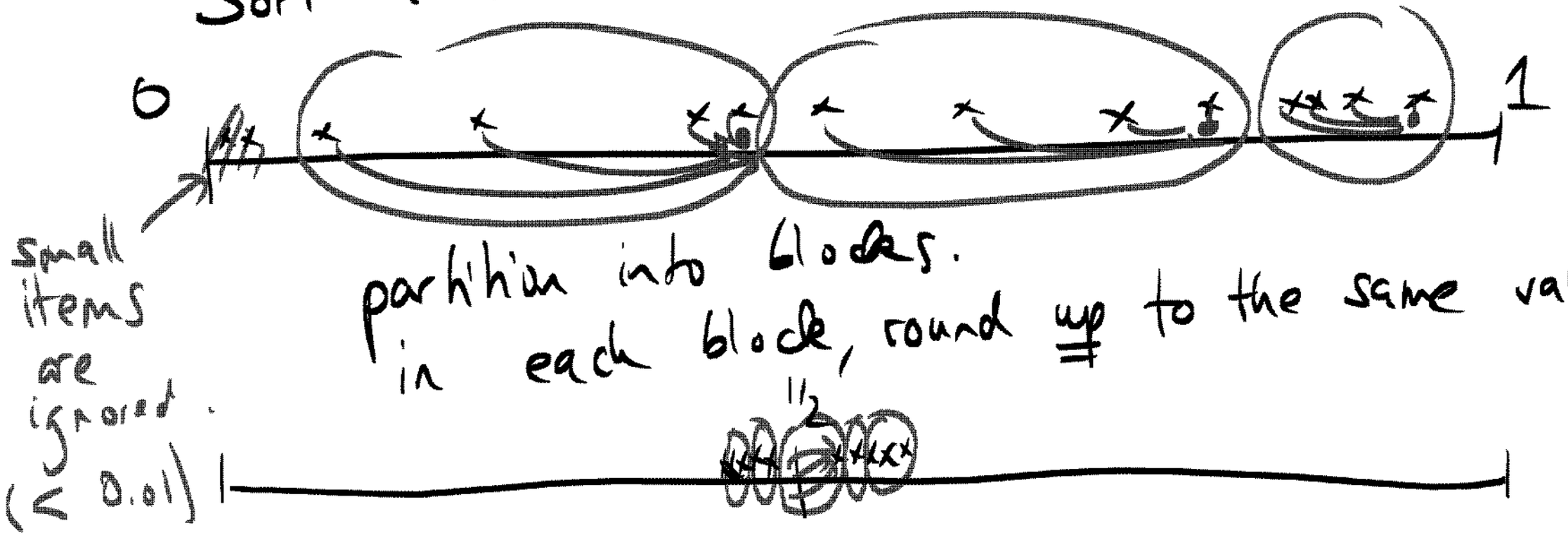
before rounding: $\frac{n}{2}$ bins suffice.

after rounding: $\frac{3n}{4}$ bins are necessary.

Answer (de la Vega Lueker 1981): To get 1.01, use blocks of size $(0.01)^2 \cdot n$

adaptive rounding.

Sort items



partition into blocks.

in each block, round up to the same value.

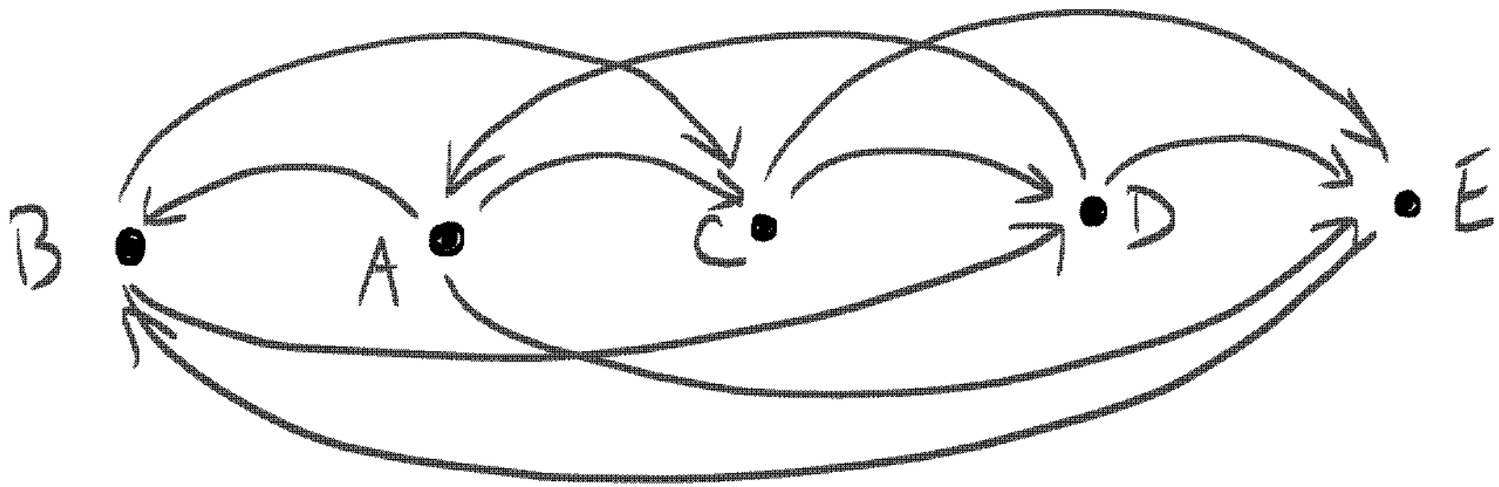
Thm: $\forall \varepsilon > 0$ there is a polynomial-time approximation algorithm with ratio $\leq 1 + \varepsilon$.

Minimum feedback arc set

Input: tournament: complete directed graph.
For every pair of vertices i,j,
either (i,j) is in E ⟵ means: $i < j$
or (j,i) is in E ⟵ means: $j < i$
but not both

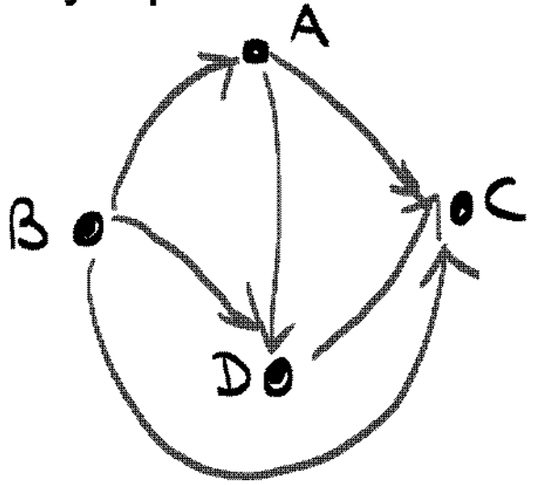
Output:
Ordering of vertices "according to the tournament comparisons"

Objective: $\max \sum_{(i,j) \in E} \mathbb{1}(i \text{ precedes } j \text{ in ordering})$



Theorem [Alon et al. 2007]: The minimum feedback arc set on tournaments is NP-hard.

Easy special case: transitive tournaments



if (i,j) and (j,k) are in E
then (i,k) is in E

For perfect input = transitive tournaments,
the problem is easy: sorting

Algorithm for perfect inputs

Pick a vertex i
Let L = {vertices j such that (j,i) is in E}
and R = {vertices j such that (i,j) is in E}
Recursively solve the problem for L and for R
Output: (ordering of L, i, ordering of R)

What about imperfect inputs?

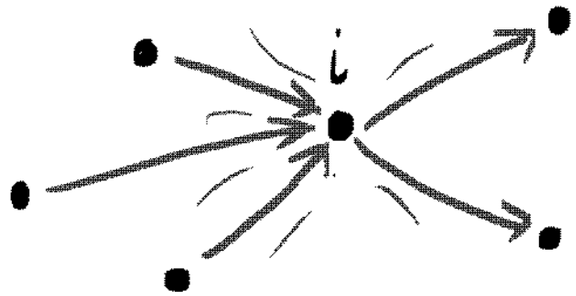
better objective: $\min \sum_{(i,j) \in E} \mathbb{1}(j \text{ precedes } i \text{ in ordering})$

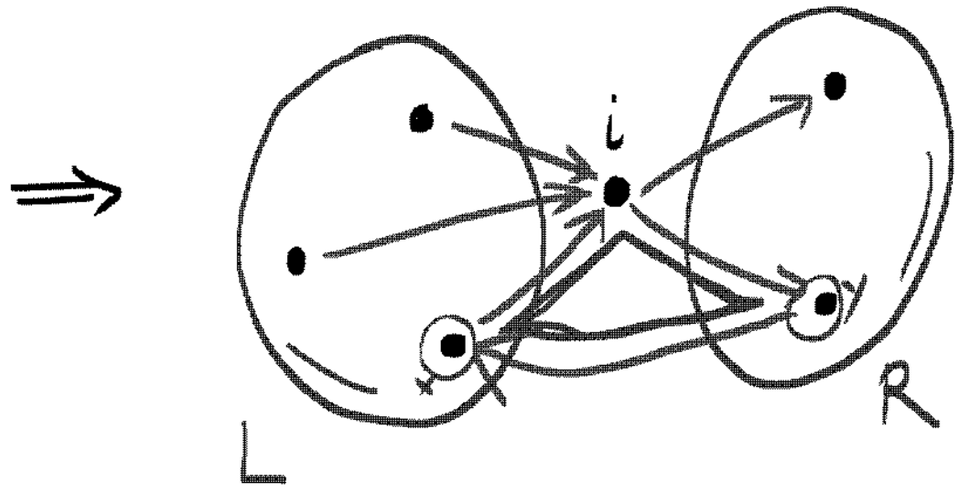
perfect input: value = 0

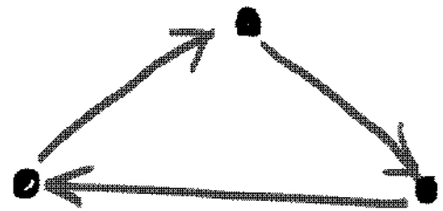
Randomized Quicksoft Algorithm
Pick a random vertex i
Let L = {vertices j such that (j,i) is in E}
and R = {vertices j such that (i,j) is in E}
Recursively solve the problem for L and for R
Output: (ordering of L, i, ordering of R)

Theorem [Ailon Charikar Newman 2005]: It's a 3-approximation

Proof: intuition

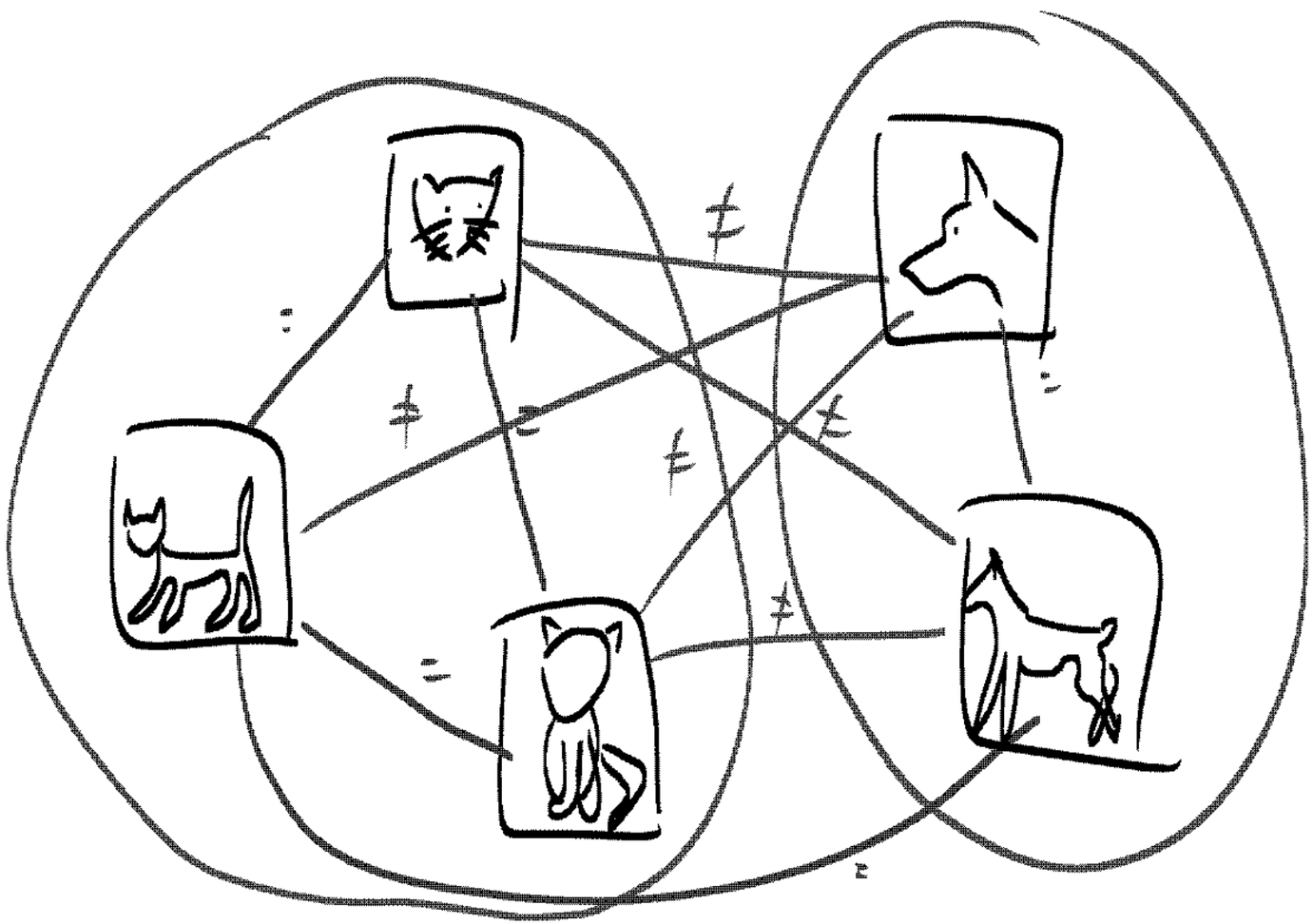






directed cycle of length 3.

Correlation clustering



Input

Undirected graph G=(V,E) with pairs of edge weights

$\forall \{i,j\} \in E \qquad w_{ij}^{+} \geq 0 \qquad w_{ij}^{-} \geq 0$

w_{ij}^{+}: measures similarity between i and j

w_{ij}^{-}: measures dissimilarity between i and j

Output

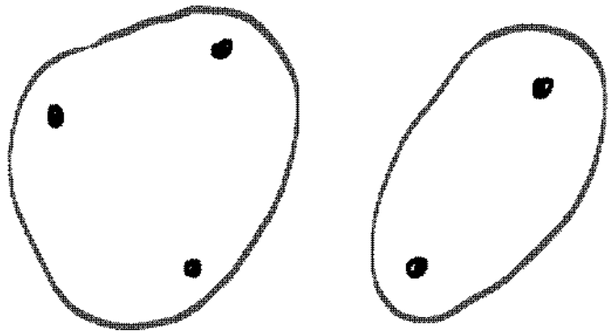
partition V "according to similarity"

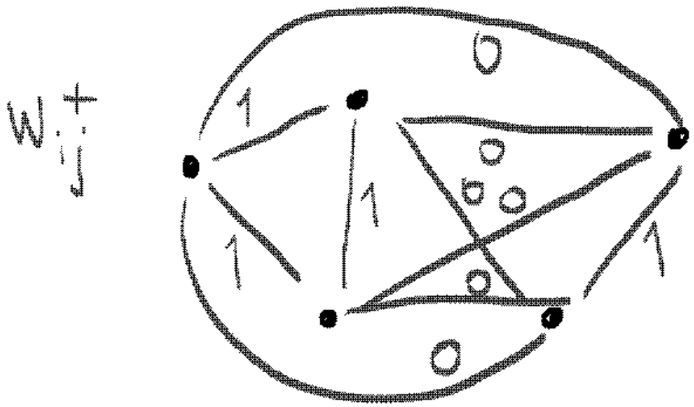
Objective

$$\max \sum_{\{i,j\} \in E} \left[w_{ij}^{+} \mathbb{1}\left(\begin{matrix} i,j \text{ in} \\ \text{same} \\ \text{part} \end{matrix}\right) + w_{ij}^{-} \mathbb{1}\left(\begin{matrix} i,j \text{ in} \\ \text{different} \\ \text{parts} \end{matrix}\right) \right]$$

Easy cases

If input is perfect...
"easy" input: complete graph, $(w_{ij}^+, w_{ij}^-) = (1,0)$ or $(0,1)$
that corresponds exactly to a clustering





Greedy algorithm
Repeat
  Pick a vertex i
  Form a cluster with all vertices j s.t. $w_{ij}^+ = 1$

equivalent objective: $\min \sum_{\{i,j\} \in E} \left[w_{ij}^{+} \mathbb{1}\left(\begin{matrix} i,j \text{ in} \\ \text{different} \\ \text{parts} \end{matrix}\right) + w_{ij}^{-} \mathbb{1}\left(\begin{matrix} i,j \text{ in} \\ \text{same} \\ \text{part} \end{matrix}\right) \right]$

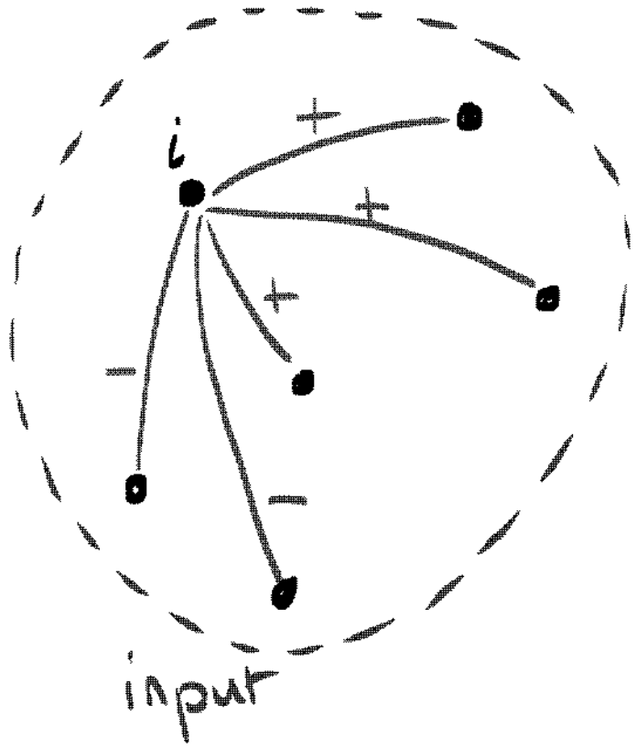
Perfect input: value = 0, solvable by greedy algorithm

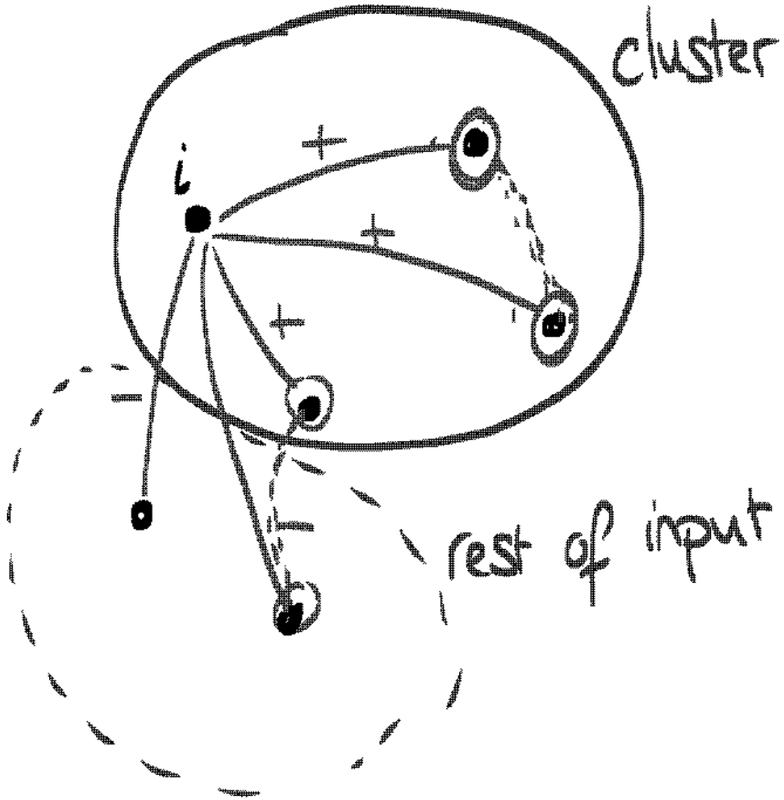
Near-perfect input: if objective is small

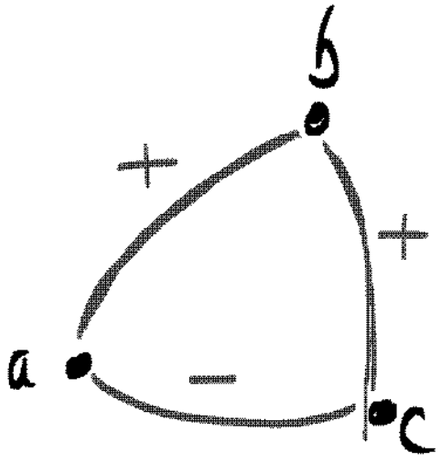
Randomized greedy algorithm
Repeat
  Pick a random vertex i
  Form a cluster with all vertices j s.t. $w_{ij}^{+} = 1$

Theorem [Ailon Charikar Newman 2005]: It's a 3-approximation

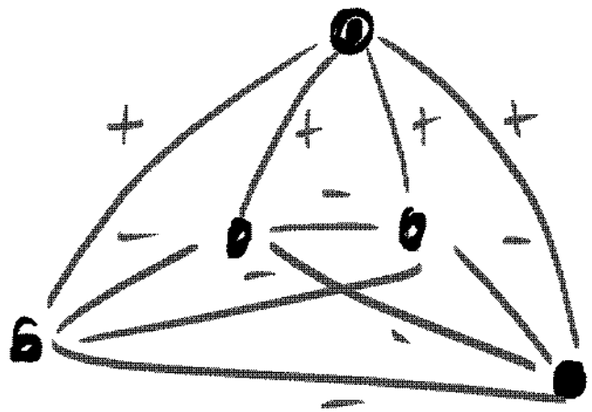
Intuition







Why 3?



Conclusion

importance de résolution effective

outils mathématiques
- algèbre matricielle
- probabilités
- (géométrie)

structure