Complexity of Local Search for Euclidean Clustering Problems

Bodo Manthey [1] Nils Morawietz [2] Jesse van Rhijn [1]
Frank Sommer [2]

[1]University of Twente

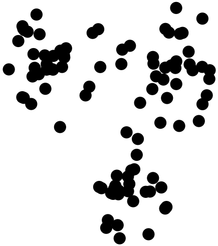
[2]Friedrich-Schiller University Jena

Dutch Optimization Seminar, March 2024

JR supported by NWO grant OCENW.KLEIN.176

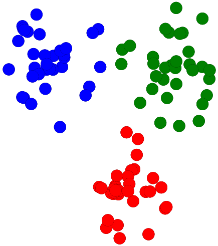
Clustering

Given n points in $\mathbb{R}^d$, group the points into <u>clusters</u>.



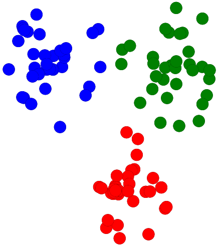
Clustering

Given n points in $\mathbb{R}^d$, group the points into clusters.



Clustering

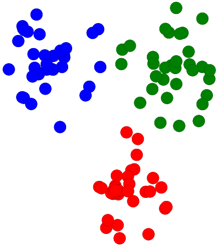
Given n points in $\mathbb{R}^d$, group the points into clusters.



Different objectives:

Clustering

Given n points in $\mathbb{R}^d$, group the points into clusters.

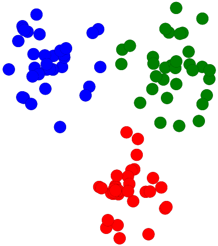


Different objectives:

- *k*-Means: minimize $\sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mathrm{cm}(C_i)\|^2$.

Clustering

Given n points in $\mathbb{R}^d$, group the points into clusters.



Different objectives:

- k-Means: minimize $\sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mathrm{cm}(C_i)\|^2$.
- Squared Euclidean Max Cut: maximize $\sum_{x \in X} \sum_{y \in Y} \|x - y\|^2$.

Flip Heuristics for Clustering

Simple local search method: reassign single points.

Flip Heuristics for Clustering

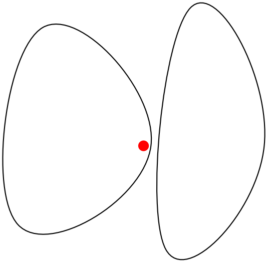
Simple local search method: reassign single points.

Called Flip for Max Cut and Hartigan–Wong method for k-Means.

Flip Heuristics for Clustering

Simple local search method: reassign single points.

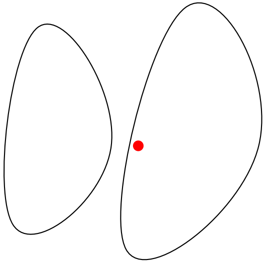
Called Flip for Max Cut and Hartigan–Wong method for k-Means.



Flip Heuristics for Clustering

Simple local search method: reassign single points.

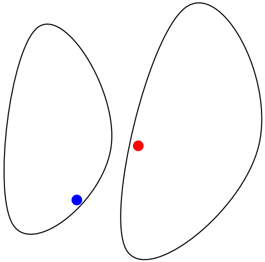
Called Flip for Max Cut and Hartigan–Wong method for k-Means.



Flip Heuristics for Clustering

Simple local search method: reassign single points.

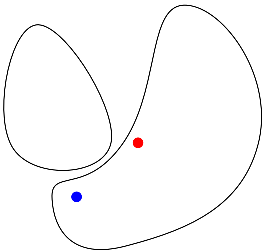
Called Flip for Max Cut and Hartigan–Wong method for k-Means.



Flip Heuristics for Clustering

Simple local search method: reassign single points.

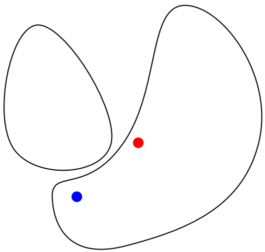
Called Flip for Max Cut and Hartigan–Wong method for k-Means.



Flip Heuristics for Clustering

Simple local search method: reassign single points.

Called Flip for Max Cut and Hartigan–Wong method for k-Means.



Theorem (Etscheid & Röglin, Manthey & R)

There exist instances of both k-Means and Squared Euclidean Max Cut that require $2^{\Omega(n)}$ iterations of Hartigan–Wong and Flip, respectively.

Complexity of Local Search

Class PLS = Polynomial Local Search.

Complexity of Local Search

Class PLS = Polynomial Local Search.

Requires $\exists$ efficient algorithms A, B, C:

Complexity of Local Search

Class PLS = Polynomial Local Search.

Requires $\exists$ efficient algorithms A, B, C:

- A: computes some feasible solution,

Complexity of Local Search

Class PLS = Polynomial Local Search.

Requires $\exists$ efficient algorithms A, B, C:

- A: computes some feasible solution,
- B: evaluates cost of solutions,

Complexity of Local Search

Class PLS = Polynomial Local Search.

Requires $\exists$ efficient algorithms A, B, C:

- A: computes some feasible solution,
- B: evaluates cost of solutions,
- C: computes improving neighbor of solution

Complexity of Local Search

Class PLS = Polynomial Local Search.

Requires $\exists$ efficient algorithms A, B, C:

- A: computes some feasible solution,
- B: evaluates cost of solutions,
- C: computes improving neighbor of solution
  $\rightarrow$ or outputs `locally optimal`.

Complexity of Local Search

Class PLS = Polynomial Local Search.

Requires $\exists$ efficient algorithms A, B, C:

- A: computes some feasible solution,
- B: evaluates cost of solutions,
- C: computes improving neighbor of solution
  $\rightarrow$ or outputs `locally optimal`.

Need a notion of reduction between PLS problems.

PLS-reductions

PLS problems relate via a special type of reduction (f, g):

$\boxed{P}$ $\boxed{Q}$

PLS-reductions

PLS problems relate via a special type of reduction (f, g):

$$\boxed{P} \xrightarrow{\quad f \quad} \boxed{Q}$$

Function f maps instance of P to instance of Q.

PLS-reductions

PLS problems relate via a special type of reduction (f, g):

$$\begin{tikzcd}
\boxed{P} \arrow[r, "f"] & \boxed{Q} \arrow[d] \\
& \boxed{s'}
\end{tikzcd}$$

Function f maps instance of P to instance of Q.

PLS-reductions

PLS problems relate via a special type of reduction (f, g):

$$\begin{tikzcd} \boxed{P} \arrow[r, "f"] & \boxed{Q} \arrow[d] \\ \boxed{s} & \boxed{s'} \arrow[l, "g"'] \end{tikzcd}$$

Function f maps instance of P to instance of Q.

Function g maps solution of Q to solution of P.

PLS-reductions

PLS problems relate via a special type of reduction (f, g):

$$\begin{tikzcd}
\boxed{P} \arrow[r, "f"] & \boxed{Q} \arrow[d] \\
\boxed{s} & \boxed{s'} \arrow[l, "g"']
\end{tikzcd}$$

Function f maps instance of P to instance of Q.

Function g maps solution of Q to solution of P.

Crucial: if s' is locally optimal, then so is $s = g(s')$.

Implications

P is PLS-complete $\implies$ P among hardest problems in PLS.

Implications

P is PLS-complete $\implies$ P among hardest problems in PLS.

PLS-complete P in polytime $\implies$ all $Q \in$ PLS in polytime.

Implications

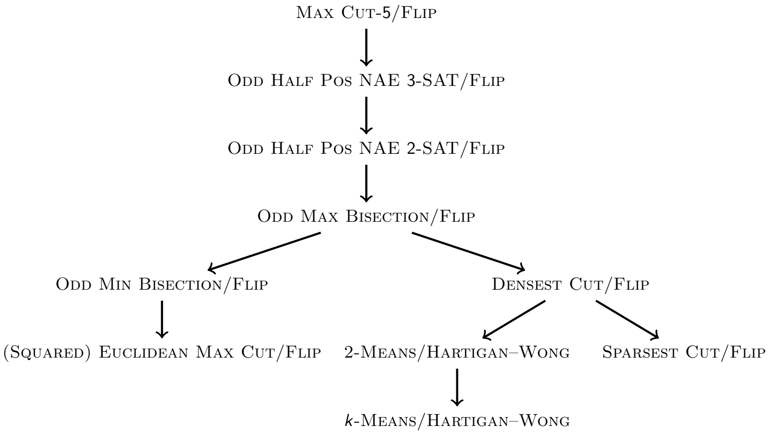
P is PLS-complete $\implies$ P among hardest problems in PLS.

PLS-complete P in polytime $\implies$ all $Q \in$ PLS in polytime.

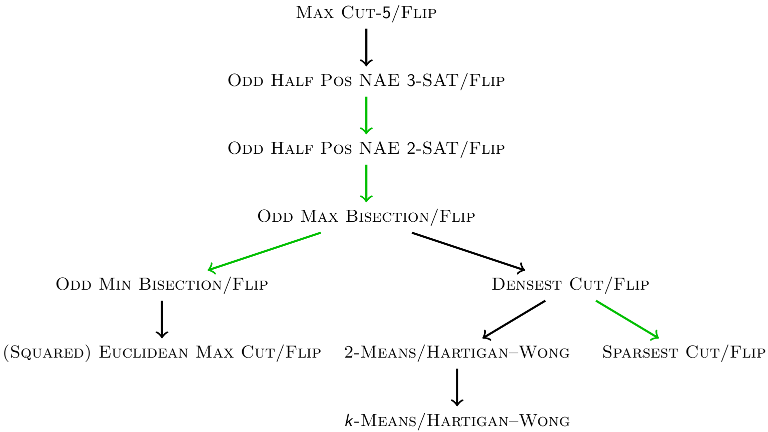
Theorem (Schäffer & Yannakakis)

If for some $P, Q \in$ PLS we have $P \leq_{PLS} Q$ via a tight reduction, then Q inherits any lower bounds on the worst-case running time of P.

Reduction Path



Reduction Path



Reduction Path

Max Cut-5/Flip

$\downarrow$

Odd Half Pos NAE 3-SAT/Flip

$\downarrow$

Odd Half Pos NAE 2-SAT/Flip

$\downarrow$

Odd Max Bisection/Flip

$\downarrow$

Odd Min Bisection/Flip

$\downarrow$

(Squared) Euclidean Max Cut/Flip

Reduction Path

Max Cut-5/Flip

$\downarrow$

Odd Half Pos NAE 3-SAT/Flip

$\downarrow$

Odd Half Pos NAE 2-SAT/Flip

$\downarrow$

Odd Max Bisection/Flip

$\downarrow$

Odd Min Bisection/Flip

$\downarrow$

(Squared) Euclidean Max Cut/Flip

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

Odd Bisection is a Cut, but with $|V_1| = |V_2| \pm 1$.

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

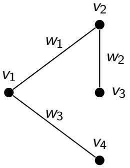
Odd Bisection is a Cut, but with $|V_1| = |V_2| \pm 1$.

Adapt NP-hardness proof of Ageev et al.:

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

Odd Bisection is a Cut, but with $|V_1| = |V_2| \pm 1$.

Adapt NP-hardness proof of Ageev et al.:



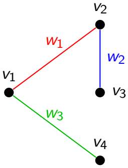
$$\mathcal{X} = \begin{pmatrix} \sqrt{w_1/2} & 0 & \sqrt{w_3/2} & \alpha_{v_1} & 0 & 0 & 0 \\ \sqrt{w_1/2} & \sqrt{w_2/2} & 0 & 0 & \alpha_{v_2} & 0 & 0 \\ 0 & \sqrt{w_2/2} & 0 & 0 & 0 & \alpha_{v_3} & 0 \\ 0 & 0 & \sqrt{w_3/2} & 0 & 0 & 0 & \alpha_{v_4} \end{pmatrix}$$

with $\alpha_v = \sqrt{w(E)/2 - w(\delta(v))/2}$

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

Odd Bisection is a Cut, but with $|V_1| = |V_2| \pm 1$.

Adapt NP-hardness proof of Ageev et al.:



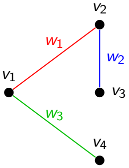
$$\Longrightarrow \quad \mathcal{X} = \begin{pmatrix} \sqrt{w_1/2} & 0 & \sqrt{w_3/2} & \alpha_{v_1} & 0 & 0 & 0 \\ \sqrt{w_1/2} & \sqrt{w_2/2} & 0 & 0 & \alpha_{v_2} & 0 & 0 \\ 0 & \sqrt{w_2/2} & 0 & 0 & 0 & \alpha_{v_3} & 0 \\ 0 & 0 & \sqrt{w_3/2} & 0 & 0 & 0 & \alpha_{v_4} \end{pmatrix}$$

with $\alpha_v = \sqrt{w(E)/2 - w(\delta(v))/2}$

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

Odd Bisection is a Cut, but with $|V_1| = |V_2| \pm 1$.

Adapt NP-hardness proof of Ageev et al.:



$$\Longrightarrow \quad \mathcal{X} = \begin{pmatrix} \sqrt{w_1/2} & 0 & \sqrt{w_3/2} & \alpha_{v_1} & 0 & 0 & 0 \\ \sqrt{w_1/2} & \sqrt{w_2/2} & 0 & 0 & \alpha_{v_2} & 0 & 0 \\ 0 & \sqrt{w_2/2} & 0 & 0 & 0 & \alpha_{v_3} & 0 \\ 0 & 0 & \sqrt{w_3/2} & 0 & 0 & 0 & \alpha_{v_4} \end{pmatrix}$$

with $\alpha_v = \sqrt{w(E)/2 - w(\delta(v))/2}$

Now we are in a purely combinatorial setting $\rightarrow$ more freedom.

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

$$
\begin{array}{c}
\textsc{Max Cut-5/Flip} \\
\downarrow \\
\textsc{Odd Half Pos NAE 3-SAT/Flip} \\
\downarrow \\
\textsc{Odd Half Pos NAE 2-SAT/Flip} \\
\downarrow \\
\textsc{Odd Max Bisection/Flip} \\
\downarrow \\
\textsc{Odd Min Bisection/Flip} \\
\downarrow \\
\textsc{(Squared) Euclidean Max Cut/Flip}
\end{array}
$$

Odd Min Bisection $\leq_{PLS}$ Squared Euclidean Max Cut

$$\begin{array}{c}
\text{Max Cut-5/Flip} \\
\downarrow \\
\text{Odd Half Pos NAE 3-SAT/Flip} \\
\downarrow \\
\text{Odd Half Pos NAE 2-SAT/Flip} \\
\downarrow \\
\text{Odd Max Bisection/Flip} \\
\downarrow \\
\text{Odd Min Bisection/Flip} \\
\downarrow \\
\text{(Squared) Euclidean Max Cut/Flip}
\end{array}$$

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

- boolean variables $x_1, \ldots, x_n$,

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

- boolean variables $x_1, \ldots, x_n$,
- $\#\{\texttt{false}\} = \#\{\texttt{true}\} \pm 1$,

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

- boolean variables $x_1, \ldots, x_n$,
- $\#\{\texttt{false}\} = \#\{\texttt{true}\} \pm 1$,
- weighted Not-All-Equal clauses $C_i = \mathrm{NAE}(x_1, \ldots, x_k)$.

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

- boolean variables $x_1, \ldots, x_n$,
- $\#\{\texttt{false}\} = \#\{\texttt{true}\} \pm 1$,
- weighted Not-All-Equal clauses $C_i = \text{NAE}(x_1, \ldots, x_k)$.

Not-All-Equal clause:

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

- boolean variables $x_1, \ldots, x_n$,
- $\#\{\texttt{false}\} = \#\{\texttt{true}\} \pm 1$,
- weighted Not-All-Equal clauses $C_i = \mathrm{NAE}(x_1, \ldots, x_k)$.

Not-All-Equal clause:

- $\mathrm{NAE}(0, 0) \rightarrow 0$
- $\mathrm{NAE}(1, 1) \rightarrow 0$

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

- boolean variables $x_1, \dots, x_n$,
- $\#\{\texttt{false}\} = \#\{\texttt{true}\} \pm 1$,
- weighted Not-All-Equal clauses $C_i = \text{NAE}(x_1, \dots, x_k)$.

Not-All-Equal clause:

- $\text{NAE}(0, 0) \rightarrow 0$
- $\text{NAE}(1, 1) \rightarrow 0$
- $\text{NAE}(0, 1) \rightarrow 1$
- $\text{NAE}(1, 0) \rightarrow 1$

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Instance of Odd Half Pos NAE k-SAT:

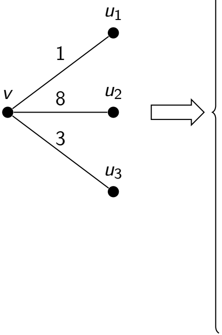
- boolean variables $x_1, \ldots, x_n$,
- $\#\{\texttt{false}\} = \#\{\texttt{true}\} \pm 1$,
- weighted Not-All-Equal clauses $C_i = \text{NAE}(x_1, \ldots, x_k)$.

Not-All-Equal clause:

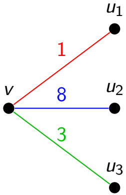
- $\text{NAE}(0,0) \to 0$
- $\text{NAE}(1,1) \to 0$
- $\text{NAE}(0,1) \to 1$
- $\text{NAE}(1,0) \to 1$

Goal: maximize weight of satisfied clauses.

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT



Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT



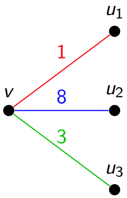
$\text{NAE}(v, u_1)$ weight: M

$\text{NAE}(v, u_2)$ weight: $8M$

$\text{NAE}(v, u_3)$ weight: $3M$

Level 1

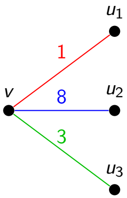
Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT



$\Rightarrow$

NAE(v, u_1)	weight: M	Level 1
NAE(v, u_2)	weight: $8M$	
NAE(v, u_3)	weight: $3M$	
NAE(q_v, u_1)	weight: $-L$	Level 2
NAE(q_v, u_2)	weight: $-8L$	
NAE(q_v, u_3)	weight: $-3L$	

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT



$\mathrm{NAE}(v, u_1)$		weight: M	Level 1
$\mathrm{NAE}(v, u_2)$		weight: $8M$	
$\mathrm{NAE}(v, u_3)$		weight: $3M$	
$\mathrm{NAE}(q_v, u_1)$		weight: $-L$	Level 2
$\mathrm{NAE}(q_v, u_2)$		weight: $-8L$	
$\mathrm{NAE}(q_v, u_3)$		weight: $-3L$	
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_1, u_2, u_3\}$	weight: -1	Level 3
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_1, u_2\}$	weight: -1	
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_1, u_3\}$	weight: 0	
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_2, u_3\}$	weight: -1	
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_1\}$	weight: 0	
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_2\}$	weight: -1	
$\mathrm{NAE}(v, q_v, a_i)$	$\{u_3\}$	weight: 0	
$\mathrm{NAE}(v, q_v, a_i)$	$\emptyset$	weight: 0	

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Max Cut-5/Flip

$\downarrow$

Odd Half Pos NAE 3-SAT/Flip

$\downarrow$

Odd Half Pos NAE 2-SAT/Flip

$\downarrow$

Odd Max Bisection/Flip

$\downarrow$

Odd Min Bisection/Flip

$\downarrow$

(Squared) Euclidean Max Cut/Flip

Max Cut-5 $\leq_{PLS}$ Odd Half Pos NAE 3-SAT

Max Cut-5/Flip

$\downarrow$

Odd Half Pos NAE 3-SAT/Flip

$\downarrow$

Odd Half Pos NAE 2-SAT/Flip

$\downarrow$

Odd Max Bisection/Flip

$\downarrow$

Odd Min Bisection/Flip

$\downarrow$

(Squared) Euclidean Max Cut/Flip

Conclusion

Theorem

Squared Euclidean Max Cut/Flip and k-Means/Hartigan–Wong are PLS*-complete.*

Conclusion

Theorem

Squared Euclidean Max Cut/Flip and k-Means/Hartigan–Wong are PLS*-complete.*

arxiv:2312.14916

Conclusion

Theorem

Squared Euclidean Max Cut/Flip and k-Means/Hartigan–Wong are PLS*-complete.*

Since our reductions are tight, we also get:

arxiv:2312.14916

Conclusion

Theorem

Squared Euclidean Max Cut/Flip and k-Means/Hartigan–Wong are PLS*-complete.*

Since our reductions are tight, we also get:

Corollary

There exist instances with initial solutions for which both Flip and Hartigan–Wong require $2^{\Omega(n)}$ iterations, no matter the implementation.

arxiv:2312.14916

Conclusion

Theorem

Squared Euclidean Max Cut/Flip and k-Means/Hartigan–Wong are PLS-complete.

Since our reductions are tight, we also get:

Corollary

There exist instances with initial solutions for which both Flip and Hartigan–Wong require $2^{\Omega(n)}$ iterations, no matter the implementation.

Other PLS-complete Euclidean optimization problems, e.g. TSP/k-Opt?

arxiv:2312.14916