An Exact ETH-Tight Algorithm for Euclidean TSP

Mark de Berg Hans Bodlaender
Sándor Kisfaludi-Bak Sudeshna Kolay

Shonan Workshop March 8, 2019

Universiteit Utrecht

The Traveling Salesman Problem

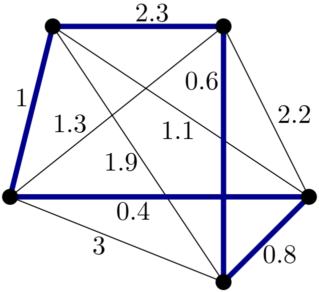
TSP

Given a complete graph with edge weights, find the shortest round trip that visits all vertices exactly once.

The Traveling Salesman Problem

TSP

Given a complete graph with edge weights, find the shortest round trip that visits all vertices exactly once.



TSP History

Who?	When	What	Runtime
Menger	'30	TSP	$O(n!)$

TSP History

Who?	When	What	Runtime
Menger	'30	TSP	$O(n!)$
Held–Karp, Bellmann	'62	TSP	$O(2^n n^2)$

Euclidean TSP

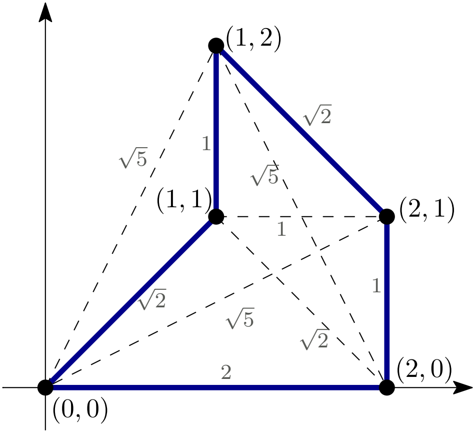
Euclidean TSP

Given n points in $\mathbb{R}^d$, find the shortest round trip that visits all of them.

Euclidean TSP

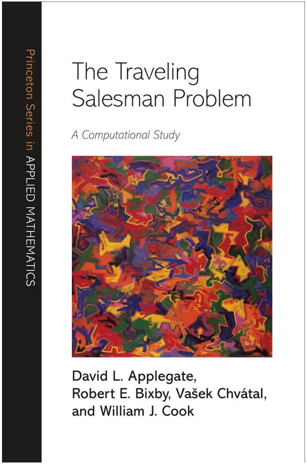
Euclidean TSP

Given n points in $\mathbb{R}^d$, find the shortest round trip that visits all of them.



Applications

- Logistics
- Microchips (printing/drilling)
- Astronomy (pointing the telescope)
- Robotics
- ...



Solving TSP

- Exact methods in practice (e.g., ILP's, matching upper and lower bound heuristics, . . .)
- Approximation: PTAS by Arora and by Mitchell, improved by Rao and Smith ('98–'99)
- This talk focus on worst case time of exact algorithms

Computational model

How hard is it to test for integers $a_1, \ldots, a_r, b_1, \ldots, b_s$ if

$$\sum_{i=1}^{r} \sqrt{a_i} \leq \sum_{j=1}^{s} \sqrt{b_j}$$

??

Exact Euclidean TSP

Who?	When	What	Runtime $\mathbb{R}^2$, $(\mathbb{R}^d)$
Menger	'30	TSP	$O(n!)$
Held–Karp, Bellmann	'62	TSP	$O(2^n n^2)$

Exact Euclidean TSP

Who?	When	What	Runtime $\mathbb{R}^2$, $(\mathbb{R}^d)$
Menger	'30	TSP	$O(n!)$
Held–Karp, Bellmann	'62	TSP	$O(2^n n^2)$
Kann, Hwang–Chang–Lee	'92-'93	ETSP $\mathbb{R}^2$	$n^{O(\sqrt{n})}$

Exact Euclidean TSP

Who?	When	What	Runtime $\mathbb{R}^2, (\mathbb{R}^d)$
Menger	'30	TSP	$O(n!)$
Held–Karp, Bellmann	'62	TSP	$O(2^n n^2)$
Kann, Hwang–Chang–Lee	'92-'93	ETSP $\mathbb{R}^2$	$n^{O(\sqrt{n})}$
Smith–Wormald	'98	ETSP $\mathbb{R}^d$	$n^{O(\sqrt{n})}$, $(n^{O(n^{1-1/d})})$

Contribution

Earlier: $n^{O(n^{1-1/d})} = 2^{O(n^{1-1/d} \log n)}$ algorithm and $2^{\Omega(n^{1-1/d-\epsilon})}$ lower bound.

Contribution

Earlier: $n^{O(n^{1-1/d})} = 2^{O(n^{1-1/d} \log n)}$ algorithm and $2^{\Omega(n^{1-1/d-\epsilon})}$ lower bound.

We have settled the asymptotics of the exponent under the Exponential Time Hypothesis (ETH).

ETH: there is no $2^{o(n)}$ algorithm for 3SAT.

Contribution

Earlier: $n^{O(n^{1-1/d})} = 2^{O(n^{1-1/d} \log n)}$ algorithm and $2^{\Omega(n^{1-1/d-\epsilon})}$ lower bound.

We have settled the asymptotics of the exponent under the Exponential Time Hypothesis (ETH).

ETH: there is no $2^{o(n)}$ algorithm for 3SAT.

Theorem (Main)

For any fixed d, there is a $2^{O(n^{1-1/d})}$ algorithm for Euclidean TSP in $\mathbb{R}^d$. All algorithms need $2^{\Omega(n^{1-1/d})}$ time under ETH.

Contribution

Earlier: $n^{O(n^{1-1/d})} = 2^{O(n^{1-1/d} \log n)}$ algorithm and $2^{\Omega(n^{1-1/d-\epsilon})}$ lower bound.

We have settled the asymptotics of the exponent under the Exponential Time Hypothesis (ETH).

ETH: there is no $2^{o(n)}$ algorithm for 3SAT.

Theorem (Main)

For any fixed d, there is a $2^{O(n^{1-1/d})}$ algorithm for Euclidean TSP in $\mathbb{R}^d$.
All algorithms need $2^{\Omega(n^{1-1/d})}$ time under ETH.

Theorem

There is a $2^{O(\sqrt{n})}$ algorithm for Euclidean TSP in $\mathbb{R}^2$.
All algorithms need $2^{\Omega(\sqrt{n})}$ time under ETH.

On the lower bounds

Theorem (Main)

For any fixed d, all algorithms for Euclidean TSP in $\mathbb{R}^d$ need $2^{\Omega(n^{1-1/d})}$ time under ETH.

Follows from B–B–K–Marx–v.d.Zanden '18 for Ham. Cycle in $\mathbb{Z}^d$.

1. ETH with sparsification
2. Embedding of 3-SAT formula in d-dimensional space, and
3. modifying existing NP-hardness proof of Hamiltonian Circuit: 3-SAT in d-dimensional space → HC in d-dim space
4. Building / modifying gadgets

Contribution

Theorem (Main)

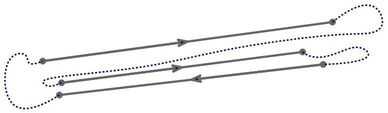
For any fixed d, there is a $2^{O(n^{1-1/d})}$ *algorithm for Euclidean TSP in* $\mathbb{R}^d$*.*
All algorithms need $2^{\Omega(n^{1-1/d})}$ *time under ETH.*

Main ideas:

1. Balanced separating point set with a square (or cube)
2. Recursively separating gives tree structure
3. Packing property guarantees that 'few edges in solution cross cube boundary'
4. Bounding the number of candidate sets of edges across a separator: twiggling square
5. Bounding the number of ways endpoints of these edges are connected (matchings): small representative set with rank based approach

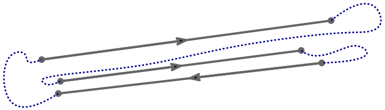
The packing property

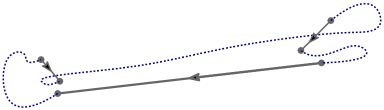
Could this tour be optimal?



The packing property

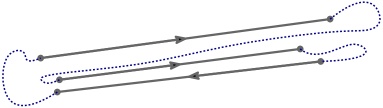
Could this tour be optimal? → No, it can be shortened.

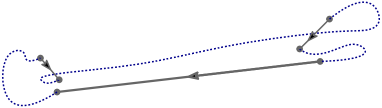




The packing property

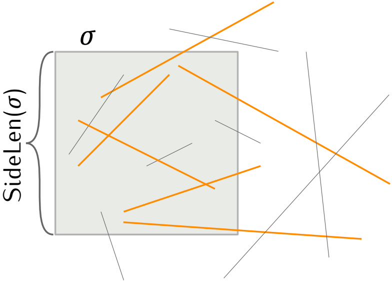
Could this tour be optimal? → No, it can be shortened.





Definition

A segment set has the packing property if for any square σ, there are only $O(1)$ segments of length at least $\mathrm{SideLen}(\sigma)/2$ *intersected by* $\mathrm{int}(\sigma)$.



Using the packing property

Lemma

The segments of an optimal TSP tour in $\mathbb{R}^d$ have the packing property.

Using the packing property

Lemma

The segments of an optimal TSP tour in $\mathbb{R}^d$ have the packing property.

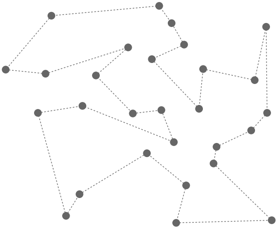
Already observed by Kann ('92) and Smith-Wormald ('98).

Idea behind algorithm:

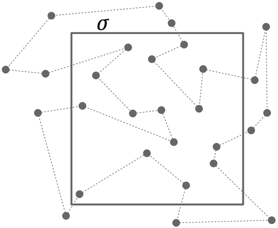
- Find separator square σ intersected by $O(\sqrt{n})$ tour segments
- Solve subproblems recursively

From Packing property: such separator exists!

The separator approach (in $\mathbb{R}^2$)

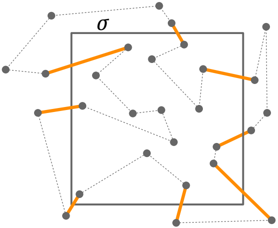


The separator approach (in $\mathbb{R}^2$)



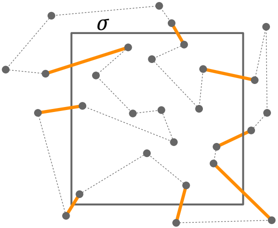
1. Find a square σ such that
   (a) σ partitions P into subsets P_{in} and P_{out} in a balanced way and
   (b) σ intersects $O(\sqrt{n})$ segments of the (unknown) optimal tour

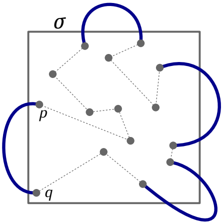
The separator approach (in $\mathbb{R}^2$)



1. Find a square σ such that
   (a) σ partitions P into subsets P_{in} and P_{out} in a balanced way and
   (b) σ intersects $O(\sqrt{n})$ segments of the (unknown) optimal tour
2. For each possible set S of tour segments intersecting σ
   (possible guesses of S are the candidate sets)

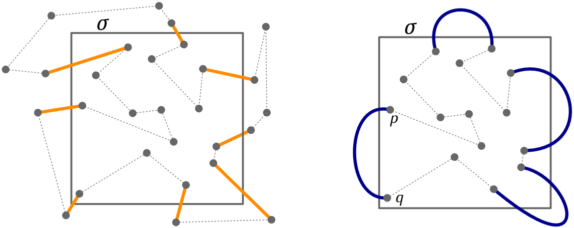
The separator approach (in $\mathbb{R}^2$)





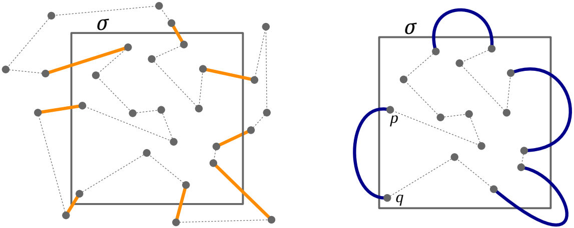
1. Find a square σ such that
   (a) σ partitions P into subsets P_{in} and P_{out} in a balanced way and
   (b) σ intersects $O(\sqrt{n})$ segments of the (unknown) optimal tour
2. For each possible set S of tour segments intersecting σ
   (possible guesses of S are the candidate sets)
3. For all matchings outside, recursively solve inside

The separator approach (in $\mathbb{R}^2$)



1. Find a square σ such that
   (a) σ partitions P into subsets P_{in} and P_{out} in a balanced way and
   (b) σ intersects $O(\sqrt{n})$ segments of the (unknown) optimal tour
2. For each possible set S of tour segments intersecting σ
   (possible guesses of S are the candidate sets)
3. For all matchings outside, recursively solve inside
4. For all matchings inside, recursively solve outside

The separator approach (in $\mathbb{R}^2$)



1. Find a square σ such that
   (a) σ partitions P into subsets P_{in} and P_{out} in a balanced way and
   (b) σ intersects $O(\sqrt{n})$ segments of the (unknown) optimal tour
2. For each possible set S of tour segments intersecting σ
   (possible guesses of S are the candidate sets)
3. For all matchings outside, recursively solve inside
4. For all matchings inside, recursively solve outside

Running time: $\boxed{\text{\# of candidate sets}} \times \boxed{\text{\# of matchings}}$

Bottleneck 1: number of candidate sets

Running time: $\boxed{\text{\# of candidate sets}} \times \boxed{\text{\# of matchings}}$

σ intersects $O(\sqrt{n})$ tour segments.

Bottleneck 1: number of candidate sets

Running time: $\boxed{\text{\# of candidate sets}} \times \boxed{\text{\# of matchings}}$

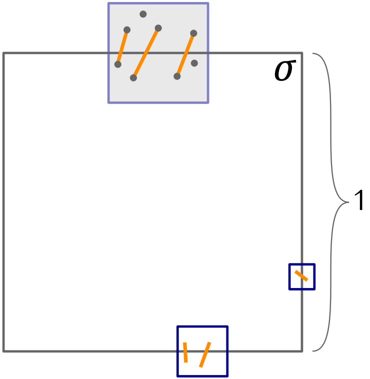
σ intersects $O(\sqrt{n})$ tour segments.

We have

$$\simeq \binom{\binom{n}{2}}{c\sqrt{n}} = 2^{\Theta(\sqrt{n}\log n)} \text{ candidate sets...}$$

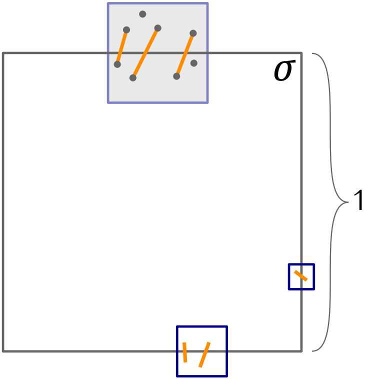
This is tight for known separator theorems.

Resolving Bottleneck 1: Pushing the packing property further



S has the packing property.

Resolving Bottleneck 1: Pushing the packing property further



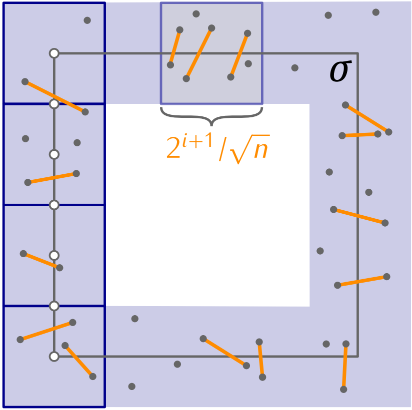
S has the packing property.

Split S into length classes:

$$S_i := \left\{ s \in S \;\middle|\; \frac{2^{i-1}}{\sqrt{n}} \leq s < \frac{2^i}{\sqrt{n}} \right\}$$

Guess each S_i separately.

Resolving Bottleneck 1:
Pushing the packing property further



S has the packing property.

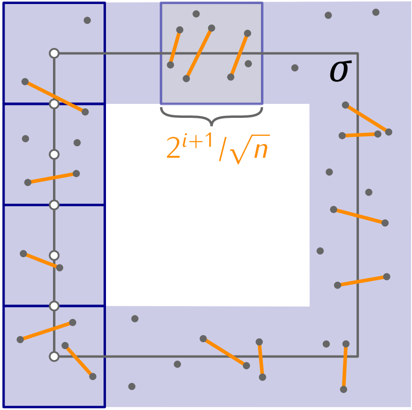
Split S into length classes:

$$S_i := \left\{ s \in S \;\middle|\; \frac{2^{i-1}}{\sqrt{n}} \leq s < \frac{2^i}{\sqrt{n}} \right\}$$

Guess each S_i separately.

S_i is inside annulus of width $\frac{2^{i+1}}{\sqrt{n}}$.

Resolving Bottleneck 1: Pushing the packing property further



S has the packing property.

Split S into length classes:

$$S_i := \left\{ s \in S \;\middle|\; \frac{2^{i-1}}{\sqrt{n}} \leq s < \frac{2^i}{\sqrt{n}} \right\}$$

Guess each S_i separately.

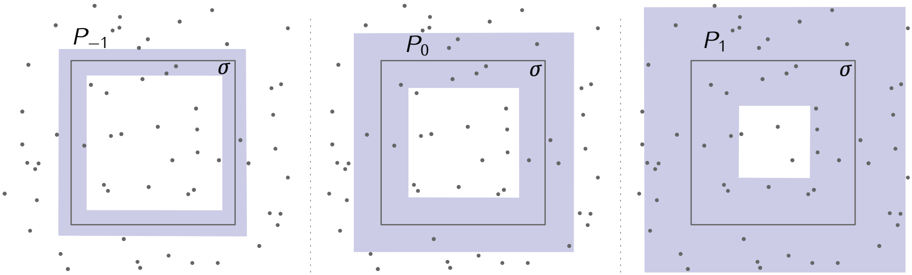
S_i is inside annulus of width $\frac{2^{i+1}}{\sqrt{n}}$.

Few guesses for S_i $\Leftrightarrow$ few pts in the i-th annulus.

We need sparse annuli around σ.

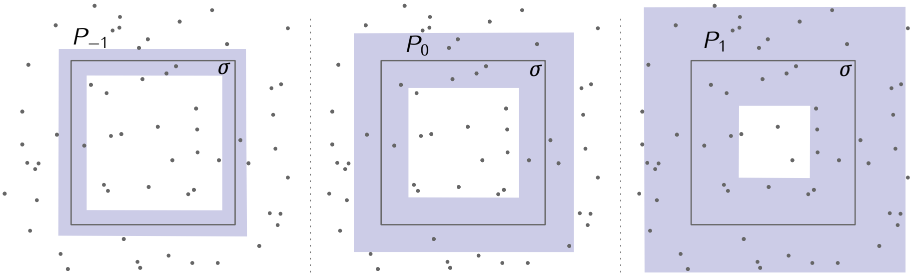
The separator theorem in $\mathbb{R}^2$

$P_i :=$ pts of P at distance $\leq 2^i/\sqrt{n}$ from σ



The separator theorem in $\mathbb{R}^2$

$P_i :=$ pts of P at distance $\leq 2^i/\sqrt{n}$ from σ



Theorem

Given $P \subset \mathbb{R}^2$, there is a balanced separator σ such that $|P_i(\sigma)| \leq c^i\sqrt{n}$, and σ can be found in polynomial time.

Only $2^{O(\sqrt{n})}$ candidates

Theorem

For any set of n points in $\mathbb{R}^2$, there is a balanced separator σ such that

(i) *each candidate set S contains $O(\sqrt{n})$ segments*

Only $2^{O(\sqrt{n})}$ candidates

Theorem

For any set of n points in $\mathbb{R}^2$, there is a balanced separator σ such that

(i) *each candidate set S contains $O(\sqrt{n})$ segments*

(ii) *there are $2^{O(\sqrt{n})}$ candidate sets.*

Only $2^{O(\sqrt{n})}$ candidates

Theorem

For any set of n points in $\mathbb{R}^2$, there is a balanced separator σ such that

(i) *each candidate set S contains $O(\sqrt{n})$ segments*

(ii) *there are $2^{O(\sqrt{n})}$ candidate sets.*

Moreover, σ and the candidates can be computed in $2^{O(\sqrt{n})}$ time.

Only $2^{O(\sqrt{n})}$ candidates

Theorem

For any set of n points in $\mathbb{R}^2$, there is a balanced separator σ such that

(i) *each candidate set S contains $O(\sqrt{n})$ segments*

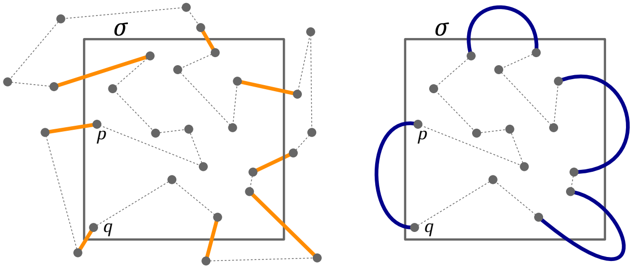
(ii) *there are $2^{O(\sqrt{n})}$ candidate sets.*

Moreover, σ and the candidates can be computed in $2^{O(\sqrt{n})}$ time.

Bottleneck 1 ✓

Bottleneck 2: number of matchings

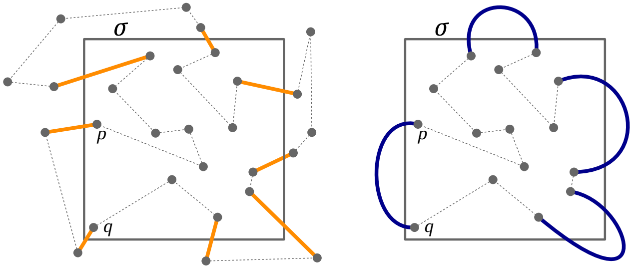
Running time: $\boxed{\text{\# of candidate sets}} \times \boxed{\text{\# of matchings}}$



There are $2^{\Theta(\sqrt{n}\log n)}$ matchings on $c\sqrt{n}$ points...

Bottleneck 2: number of matchings

Running time: $\boxed{\text{\# of candidate sets}} \times \boxed{\text{\# of matchings}}$



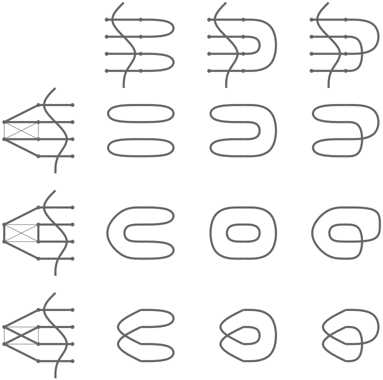
There are $2^{\Theta(\sqrt{n}\log n)}$ matchings on $c\sqrt{n}$ points...

Resolution: adapt Rank Based Approach ('15) by Bodlaender *et al.*

Rank based approach

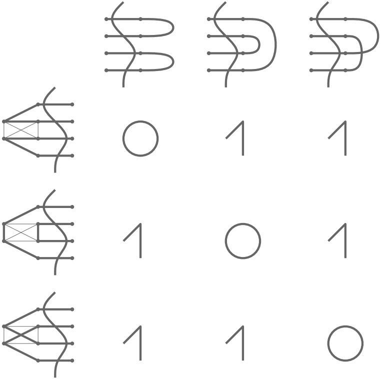
- Introduced for solving connectivity problems like Hamiltonian Circuit, Steiner Tree, Connected Dominating Set, …in $O(2^{O(tw)}n)$ time on graphs of small treewidth tw by B, Cygan, Nederlof, Kratsch (2015)
- Application, better rank bound for HC-like problems by Cygan, Nederlof, Kratsch (2018)
- Experimental evaluation for Steiner tree by Fafianie, B, Nederlof (2015)
- Experimental evaluation for Hamiltonian Circuit by Pilipczuk, Ziobp (2019)

Back to Hamiltonian Circuit on Graphs



If we have two of these in a table, we do not need the third!

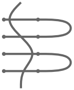
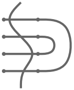
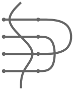
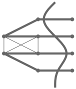
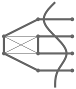
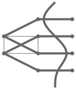
The rank based approach



We can drop a row, when for each 1 in the row, another row has also a 1 in that column

The rank based approach

The connectivity matrix

			
	0	1	1
	1	0	1
	1	1	0

We can drop a row, when for each 1 in the row, another row has also a 1 in that column: a sufficient condition is that the row is a linear combination mod2 of other rows

Rank based approach scheme for HC on graphs of small treewidth

- Do a 'usual' DP on the tree decomposition, BUT
- If a table has more rows that the rank of the connectivity matrix, then REDUCE

REDUCE:
Build the part of the connectivity matrix with
  rows: the entries in the current table
  colums: a basis of the connectivity matrix
Sweep with Gauss elimination (compute mod2)
Remove every row with only 0's
Gives 'representative set', and we end with at most rank(M) number of table entries

Ranks

Connectivity matrix: columns and rows are partitions; 1 if closure of both partition connects all elements, 0 otherwise

Connectivity matrix for matchings: rows and columns are a matching; 1 if combination gives one cycle, 0 otherwise

Theorem (BCKN (see also Lovasz), CKN)

The rank of the connectivity matrix for k elements is 2^{k-1}. The rank of the connectivity matrix for matchings on k elements in $2^{k/2-1}$.

Picture from paper by Cygan, Nederlof, Kratsch

Nr.		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	LC
1		0	0	0	0	1	1	0	1	1	1	1	0	1	1	0	1
2		0	0	0	1	0	1	1	1	0	0	1	1	1	0	1	2
3		0	0	0	1	1	0	1	0	1	1	0	1	0	1	1	1+2
4		0	1	1	0	0	0	0	1	1	1	0	1	1	0	1	4
5		1	0	1	0	0	0	1	0	1	0	1	1	1	1	0	5
6		1	1	0	0	0	0	1	1	0	1	1	0	0	1	1	4+5
7		0	1	1	0	1	1	0	0	0	0	1	1	0	1	1	1+4
8		1	1	0	1	0	1	0	0	0	1	0	1	1	1	0	2+4+5
9		1	0	1	1	1	0	0	0	0	1	1	0	1	0	1	1+2+5
10		1	0	1	1	0	1	0	1	1	0	0	0	0	1	1	2+5
11		1	1	0	0	1	1	1	0	1	0	0	0	1	0	1	1+4+5
12		0	1	1	1	1	0	1	1	0	0	0	0	1	1	0	1+2+4
13		1	1	0	1	1	0	0	1	1	0	1	1	0	0	0	1+2+4+5
14		1	0	1	0	1	1	1	1	0	1	0	1	0	0	0	1+5
15		0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	2+4

Figure 1: The matrix $\mathcal{H}_6$. Letting the baseset be $\{0, \ldots, 5\}$ matching 1 indexing row and column 1 equals $\{\{0,1\},\{2,3\},\{4,5\}\}$. The set $\mathbf{X}_t = \{1,2,4,5\}$ from Definition 3.1 is easily seen to be a row basis; the linear combinations are depicted in the last column.

Weighted solutions

Sort the rows with respect to non-decreasing cost

Gaussian elimination top-to-bottom: eliminate rows that are a linear combination of 'cheaper' rows

Using the rank based approach here

At each step in the recursion:

- For each candidate set of edges across the separating square:
- We have $n^{1-1/d}$ endpoints of the candidate set that can be matched inside and outside
  - Recursively, build representative set of matchings inside
  - Recursively, build representative set of matchings outside
- Make all combinations of inside and outside

In 2d, one can also use that matchings are non-overlapping and use Catalan structures

This resolves Bottleneck 2.

Conclusion

We can solve Euclidean TSP exactly for constant d in $2^{O(n^{1-1/d})}$ time.
This is tight under ETH.

- Rank=-based approach can give practical and theoretical faster algorithms on tree decompositions and similar structures

Conclusion

We can solve Euclidean TSP exactly for constant d in $2^{O(n^{1-1/d})}$ time.
This is tight under ETH.

- Rank=-based approach can give practical and theoretical faster algorithms on tree decompositions and similar structures
- Treewidth-like techniques in geometrc settings

Conclusion

We can solve Euclidean TSP exactly for constant d in $2^{O(n^{1-1/d})}$ time.
This is tight under ETH.

- Rank=-based approach can give practical and theoretical faster algorithms on tree decompositions and similar structures
- Treewidth-like techniques in geometrc settings
- Computational model ...

Conclusion

We can solve Euclidean TSP exactly for constant d in $2^{O(n^{1-1/d})}$ time.
This is tight under ETH.

- Rank=-based approach can give practical and theoretical faster algorithms on tree decompositions and similar structures
- Treewidth-like techniques in geometrc settings
- Computational model ...
- Open: Log shaving the Rectilinear Steiner tree?
  $(n^{O(n^{1-1/d})} \to 2^{O(n^{1-1/d})})$

Conclusion

We can solve Euclidean TSP exactly for constant d in $2^{O(n^{1-1/d})}$ time.
This is tight under ETH.

- Rank=-based approach can give practical and theoretical faster algorithms on tree decompositions and similar structures
- Treewidth-like techniques in geometrc settings
- Computational model ...
- Open: Log shaving the Rectilinear Steiner tree?
  $(n^{O(n^{1-1/d})} \to 2^{O(n^{1-1/d})})$
- Open: Separators with optimal constants?
  (optimal tradeoffs between balance and size?)