Bounded Diameter Decompositions in Mathematics and Theoretical Computer Science

Louis Esperet (CNRS, G-SCOP, Grenoble)

20 ans du GDR-I(F)M

March 18, 2026

A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d

A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications

A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension

A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension
- a clean combinatorial definition !

A notion of dimension for metric spaces

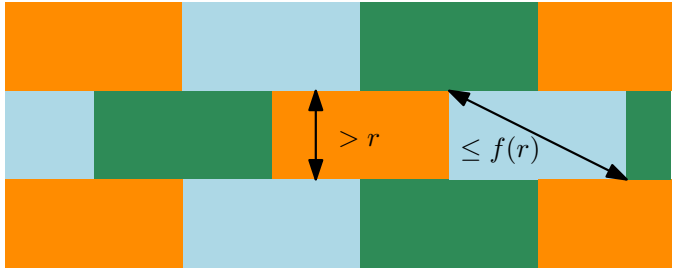
- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension
- a clean combinatorial definition !

A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension
- a clean combinatorial definition !

A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension
- a clean combinatorial definition !



A notion of dimension for metric spaces

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension
- a clean combinatorial definition !

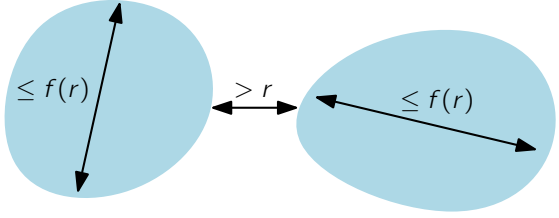
(source: wikipedia - Hex game)

Asymptotic dimension

Gromov (1993). A metric space X has asymptotic dimension at most d if there is a function f such that for every $r > 0$, X can be covered by clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two sets of the same color are at distance $> r$ apart.

Asymptotic dimension

Gromov (1993). A metric space X has asymptotic dimension at most d if there is a function f such that for every $r > 0$, X can be covered by clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two sets of the same color are at distance $> r$ apart.

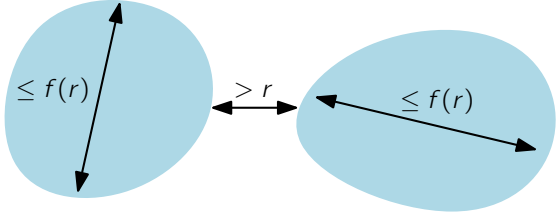


f is called the control function

It can often be taken to be linear

Asymptotic dimension

Gromov (1993). A metric space X has asymptotic dimension at most d if there is a function f such that for every $r > 0$, X can be covered by clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two sets of the same color are at distance $> r$ apart.



f is called the control function

It can often be taken to be linear

- $\mathbb{R}^d$ and d-dimensional grids should have dimension d
- robust and invariant under local modifications
- a continuous space and a sufficiently fine discretization should have the same dimension
- a clean combinatorial definition

Quasi-isometry

Two metric spaces (X, d_X) and (Y, d_Y) are quasi-isometric if there is a map $f : X \to Y$ and constants $A_1, A_2, B_1, B_2, C \geq 0$ such that any element of Y is at distance at most C from some element of $f(X)$, and for every $x_1, x_2 \in X$,

$$A_1 \cdot d_X(x_1, x_2) - B_1 \leq d_Y(f(x_1), f(x_2)) \leq A_2 \cdot d_X(x_1, x_2) + B_2.$$

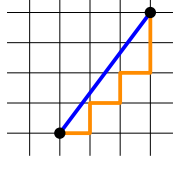
Quasi-isometry

Two metric spaces (X, d_X) and (Y, d_Y) are quasi-isometric if there is a map $f : X \to Y$ and constants $A_1, A_2, B_1, B_2, C \geq 0$ such that any element of Y is at distance at most C from some element of $f(X)$, and for every $x_1, x_2 \in X$,

$$A_1 \cdot d_X(x_1, x_2) - B_1 \leq d_Y(f(x_1), f(x_2)) \leq A_2 \cdot d_X(x_1, x_2) + B_2.$$

Example. $X =$ the 2-dimensional grid, $Y = \mathbb{R}^2$, f : the grid $\to \mathbb{Z}^2$.

$$\frac{1}{\sqrt{2}} d_X(x_1, x_2) \leq d_Y(f(x_1), f(x_2)) \leq d_X(x_1, x_2).$$



Quasi-isometry

Two metric spaces (X, d_X) and (Y, d_Y) are quasi-isometric if there is a map $f : X \to Y$ and constants $A_1, A_2, B_1, B_2, C \geq 0$ such that any element of Y is at distance at most C from some element of $f(X)$, and for every $x_1, x_2 \in X$,

$$A_1 \cdot d_X(x_1, x_2) - B_1 \leq d_Y(f(x_1), f(x_2)) \leq A_2 \cdot d_X(x_1, x_2) + B_2.$$

Observation

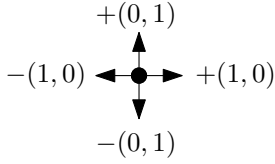
Quasi-isometric spaces have the same asymptotic dimension.

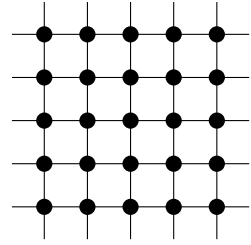
Geometric group theory

Given a finitely generated group G and a finite symmetric set of generators S, the Cayley graph of (G, S) has vertex set x and an edge between any element $x \in G$ and any element $xs \in G$ $(s \in S)$.

Geometric group theory

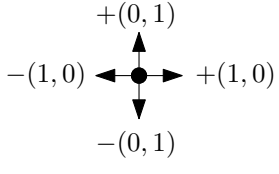
Given a finitely generated group G and a finite symmetric set of generators S, the Cayley graph of (G, S) has vertex set x and an edge between any element $x \in G$ and any element $xs \in G$ $(s \in S)$.

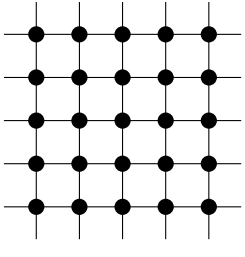




Geometric group theory

Given a finitely generated group G and a finite symmetric set of generators S, the Cayley graph of (G, S) has vertex set x and an edge between any element $x \in G$ and any element $xs \in G$ $(s \in S)$.



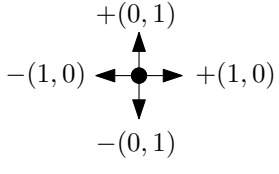


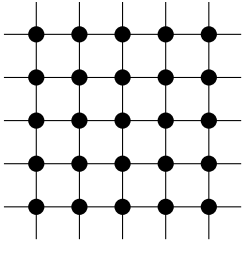
Observation

All Cayley graphs of a finitely generated group G are quasi-isometric, and thus they have the same asymptotic dimension.

Geometric group theory

Given a finitely generated group G and a finite symmetric set of generators S, the Cayley graph of (G, S) has vertex set x and an edge between any element $x \in G$ and any element $xs \in G$ $(s \in S)$.





Observation

All Cayley graphs of a finitely generated group G are quasi-isometric, and thus they have the same asymptotic dimension.

Asymptotic dimension is a group invariant !

A convenient rephrasing

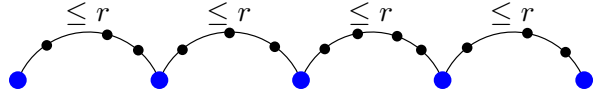
A graph G has asymptotic dimension at most d if there is a function f such that for every $r > 0$, G can be covered by clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two sets of the same color are at distance $> r$ apart.

A convenient rephrasing

A graph G has asymptotic dimension at most d if there is a function f such that for every $r > 0$, G can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.

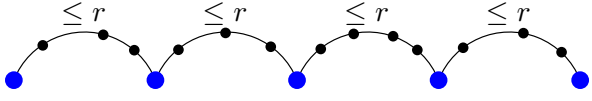
A convenient rephrasing

A graph G has asymptotic dimension at most d if there is a function f such that for every $r > 0$, G can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.



A convenient rephrasing

A graph G has asymptotic dimension at most d if there is a function f such that for every $r > 0$, G can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.



Equivalent definition

A graph G has asymptotic dimension at most d if there is a function f such that for every $r > 0$, G has a $(d + 1)$-coloring, in which each monochromatic sequence as above has diameter at most $f(r)$ in G.

Some results in graph theory

- A graph has asymptotic dimension 0 if and only there is a constant D such that all components have diameter at most D

Some results in graph theory

- A graph has asymptotic dimension 0 if and only there is a constant D such that all components have diameter at most D
- trees have asymptotic dimension 1

Some results in graph theory

- A graph has asymptotic dimension 0 if and only there is a constant D such that all components have diameter at most D
- trees have asymptotic dimension 1

A graph G is K_t-minor-free if one cannot obtain the complete graph on t vertices from G by edge-contraction and vertex-deletion.

Some results in graph theory

- A graph has asymptotic dimension 0 if and only there is a constant D such that all components have diameter at most D
- trees have asymptotic dimension 1

A graph G is K_t-minor-free if one cannot obtain the complete graph on t vertices from G by edge-contraction and vertex-deletion.

- Ostrovskii and Rosenthal (2015) proved that K_t-minor-free graphs have asymptotic dimension at most 4^t

Some results in graph theory

- A graph has asymptotic dimension 0 if and only there is a constant D such that all components have diameter at most D
- trees have asymptotic dimension 1

A graph G is K_t-minor-free if one cannot obtain the complete graph on t vertices from G by edge-contraction and vertex-deletion.

- Ostrovskii and Rosenthal (2015) proved that K_t-minor-free graphs have asymptotic dimension at most 4^t
- Fujiwara and Papasoglu (2020) proved that planar graphs have asymptotic dimension at most 3

Some results in graph theory

- A graph has asymptotic dimension 0 if and only there is a constant D such that all components have diameter at most D
- trees have asymptotic dimension 1

A graph G is K_t-minor-free if one cannot obtain the complete graph on t vertices from G by edge-contraction and vertex-deletion.

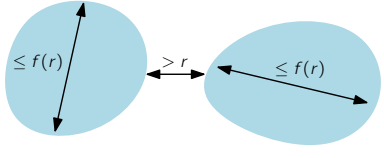
- Ostrovskii and Rosenthal (2015) proved that K_t-minor-free graphs have asymptotic dimension at most 4^t
- Fujiwara and Papasoglu (2020) proved that planar graphs have asymptotic dimension at most 3

Theorem (Bonamy, Bousquet, Esperet, Groenland, Liu, Pirot, and Scott 2020)

For every t, K_t-minor-free graphs have asymptotic dimension at most 2.

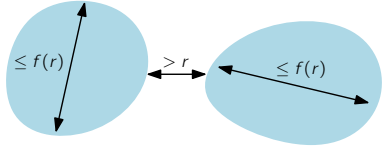
Sparse partitions

A class of graphs has asymptotic dimension at most d if there is a function f such that for every $r > 0$, every graph G in the class can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.



Sparse partitions

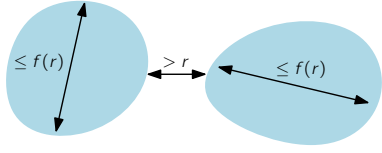
A class of graphs has asymptotic dimension at most d if there is a function f such that for every $r > 0$, every graph G in the class can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.



In many cases we can take $f(r) = c \cdot r$ for some $c > 0$ (d-dimensional grids, trees, planar graphs or more generally excluding a fixed minor).

Sparse partitions

A class of graphs has asymptotic dimension at most d if there is a function f such that for every $r > 0$, every graph G in the class can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.

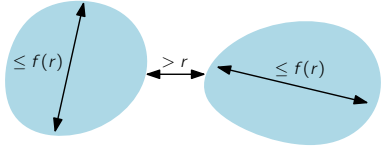


In many cases we can take $f(r) = c \cdot r$ for some $c > 0$ (d-dimensional grids, trees, planar graphs or more generally excluding a fixed minor).

If $f(r) = c \cdot r$ for some $c > 0$ and the partitions can be computed efficiently then this is equivalent to a notion introduced by Awerbuch and Peleg in 1990, under the terminology of sparse partitions/covers.

Sparse partitions

A class of graphs has asymptotic dimension at most d if there is a function f such that for every $r > 0$, every graph G in the class can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two clusters of the same color are at distance $> r$ apart.

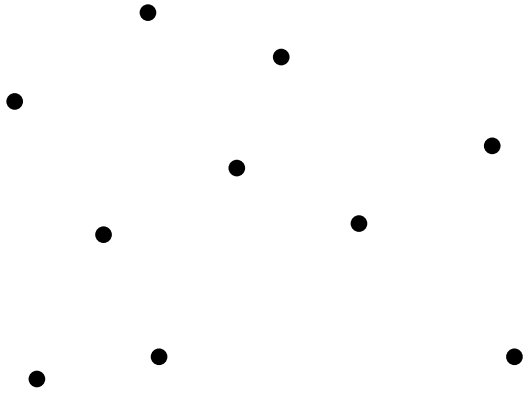


In many cases we can take $f(r) = c \cdot r$ for some $c > 0$ (d-dimensional grids, trees, planar graphs or more generally excluding a fixed minor).

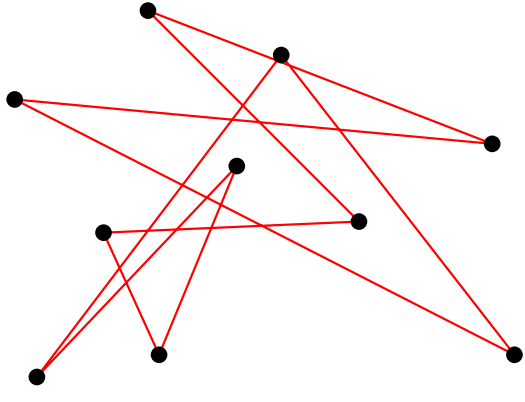
If $f(r) = c \cdot r$ for some $c > 0$ and the partitions can be computed efficiently then this is equivalent to a notion introduced by Awerbuch and Peleg in 1990, under the terminology of sparse partitions/covers.

When c and d are small, several combinatorial optimisation problems can be approximated efficiently.

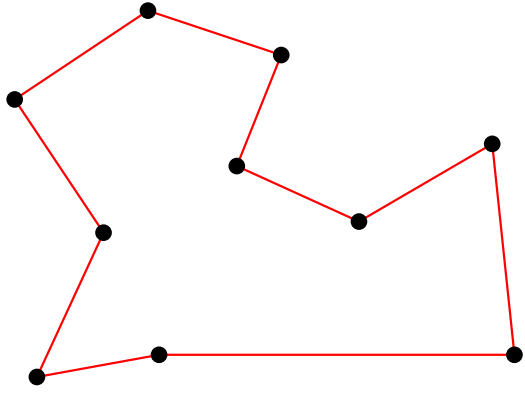
Universal Traveling Salesman Problem



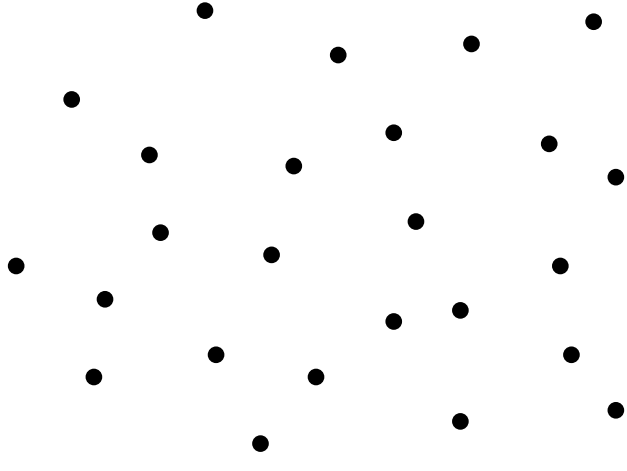
Universal Traveling Salesman Problem



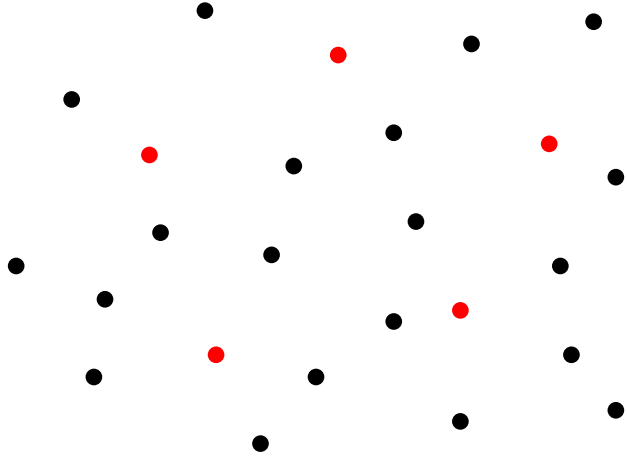
Universal Traveling Salesman Problem



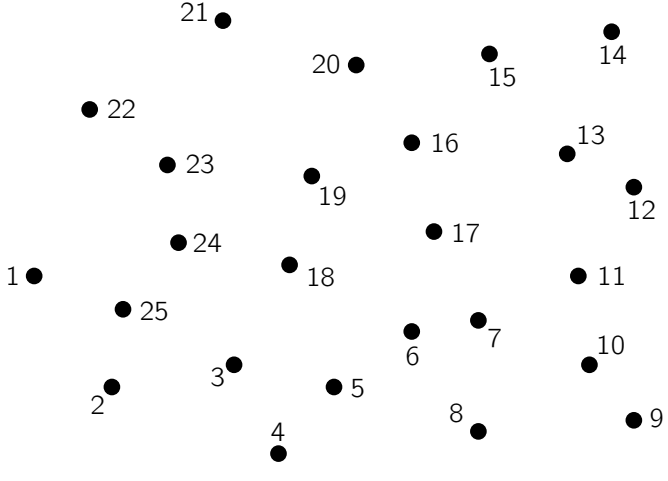
Universal Traveling Salesman Problem



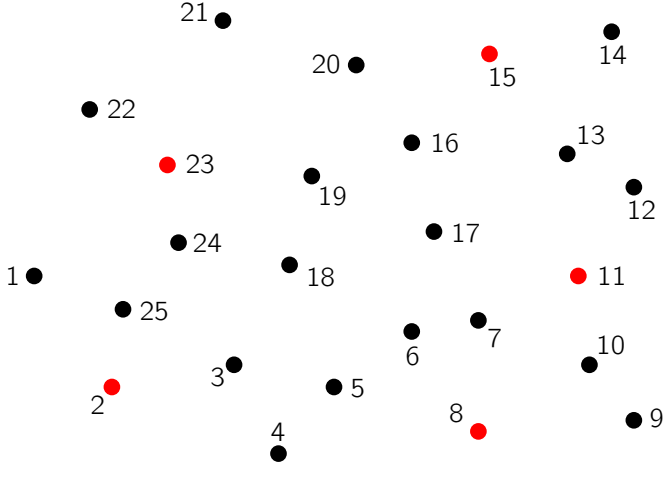
Universal Traveling Salesman Problem



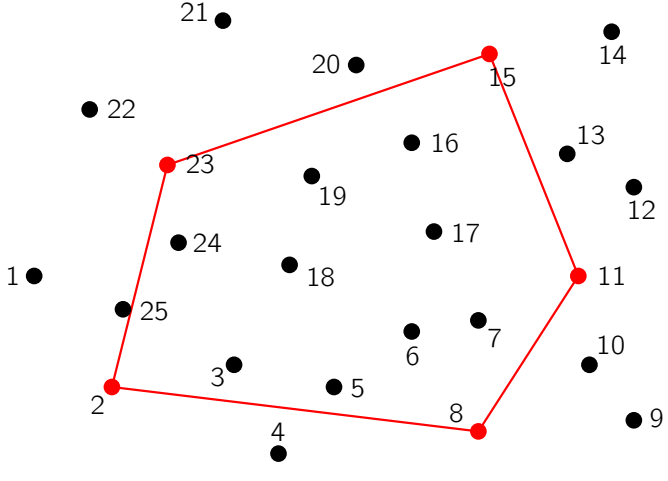
Universal Traveling Salesman Problem



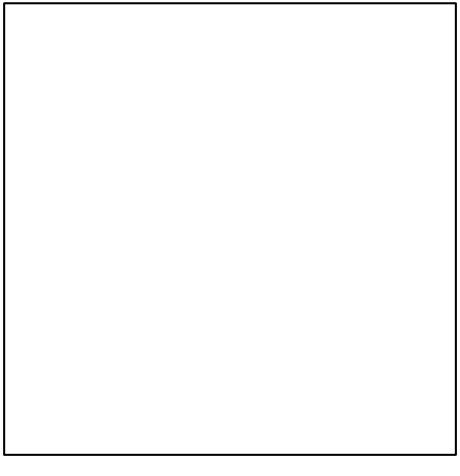
Universal Traveling Salesman Problem



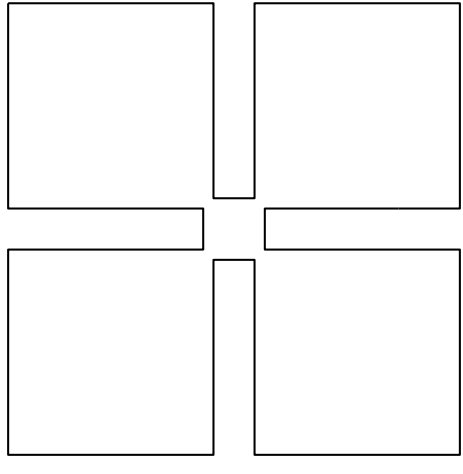
Universal Traveling Salesman Problem



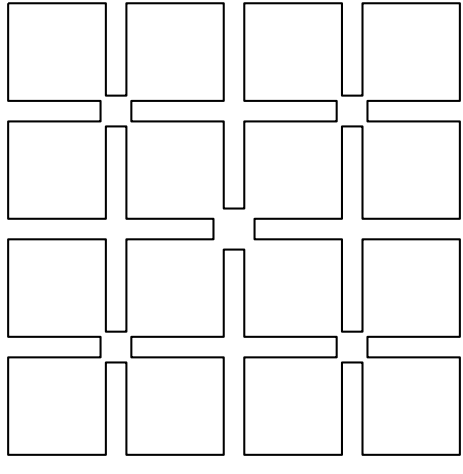
Universal Traveling Salesman Problem



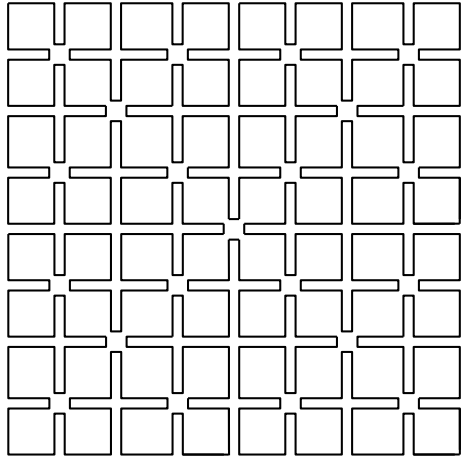
Universal Traveling Salesman Problem



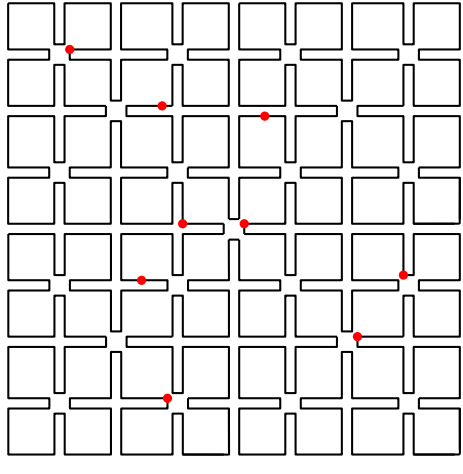
Universal Traveling Salesman Problem



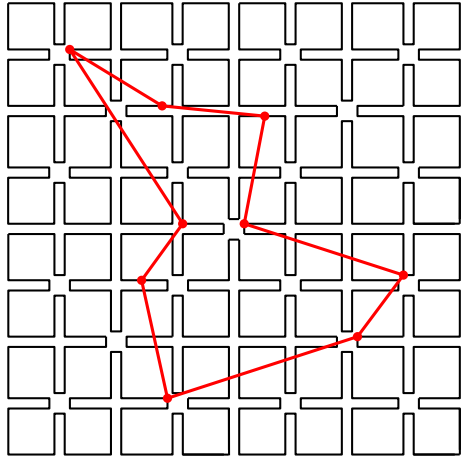
Universal Traveling Salesman Problem



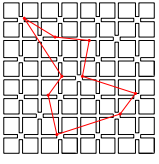
Universal Traveling Salesman Problem



Universal Traveling Salesman Problem



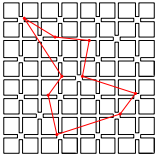
Universal Traveling Salesman Problem



Theorem (Bartholdi and Platzman 1985)

Ordering n points in the plane following the space-filling curve gives a $O(\log n)$-approximation for the TSP for any subset of the points.

Universal Traveling Salesman Problem



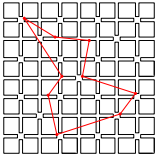
Theorem (Bartholdi and Platzman 1985)

Ordering n points in the plane following the space-filling curve gives a $O(\log n)$-approximation for the TSP for any subset of the points.

Theorem (Jia, Lin, Noubir, Rajaraman and Sundaram 2005)

This works in any metric space that has constant asymptotic dimension with linear control function.

Universal Traveling Salesman Problem



Theorem (Bartholdi and Platzman 1985)

Ordering n points in the plane following the space-filling curve gives a $O(\log n)$-approximation for the TSP for any subset of the points.

Theorem (Jia, Lin, Noubir, Rajaraman and Sundaram 2005)

This works in any metric space that has constant asymptotic dimension with linear control function.

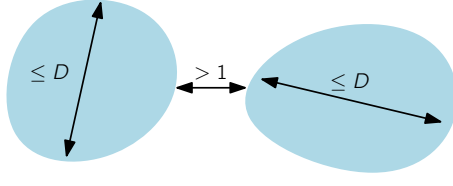
The same holds for several other universal variants of classical combinatorial optimization problems (Universal Steiner Tree, ...)

Network decompositions

A class of graphs G has asymptotic dimension at most d if there is a function f such that for every $r > 0$, every graph G in the class can be partitioned into clusters of at most $d + 1$ colors, each of diameter at most $f(r)$, and any two sets of the same color are at distance $> r$ apart.

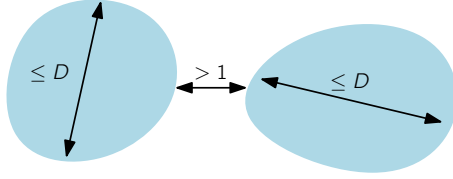
Network decompositions

Taking $r = 1$, for any class of graphs of bounded asymptotic dimension, there exist C and D such that every graph in the class has a partition into clusters of C colors, such that neighboring clusters have different colors and every cluster has diameter at most D.



Network decompositions

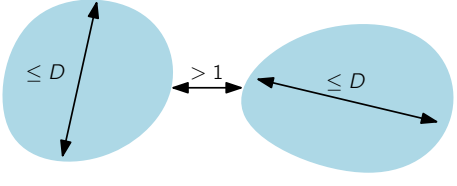
Taking $r = 1$, for any class of graphs of bounded asymptotic dimension, there exist C and D such that every graph in the class has a partition into clusters of C colors, such that neighboring clusters have different colors and every cluster has diameter at most D.



This is a notion introduced by Awerbuch, Goldberg, Luby and Plotkin in 1989, called a network decomposition with C colors and diameter D.

Network decompositions

Taking $r = 1$, for any class of graphs of bounded asymptotic dimension, there exist C and D such that every graph in the class has a partition into clusters of C colors, such that neighboring clusters have different colors and every cluster has diameter at most D.



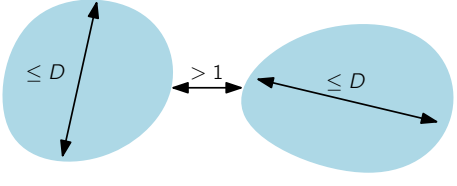
This is a notion introduced by Awerbuch, Goldberg, Luby and Plotkin in 1989, called a network decomposition with C colors and diameter D.

Theorem (Linial and Saks 1993)

Every n-vertex graph has a network decomposition with $O(\log n)$ colors and diameter $O(\log n)$.

Network decompositions

Taking $r = 1$, for any class of graphs of bounded asymptotic dimension, there exist C and D such that every graph in the class has a partition into clusters of C colors, such that neighboring clusters have different colors and every cluster has diameter at most D.



This is a notion introduced by Awerbuch, Goldberg, Luby and Plotkin in 1989, called a network decomposition with C colors and diameter D.

Theorem (Linial and Saks 1993)

Every n-vertex graph has a network decomposition with $O(\log n)$ colors and diameter $O(\log n)$.

This has proved to be a central tool in distributed computing.

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.

Distributed computing

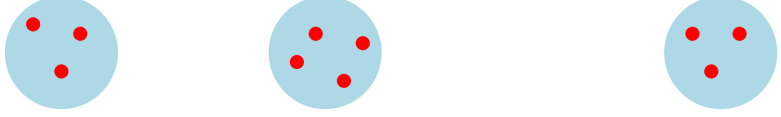
The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.



Distributed computing

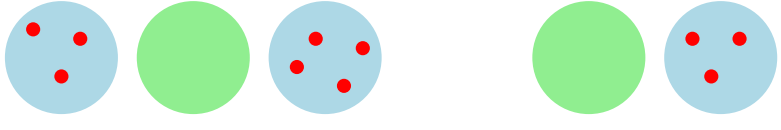
The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.



Distributed computing

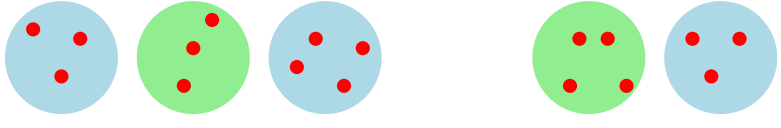
The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.



Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.

Distributed computing

The network is modeled as a graph, in which each node has a processor of infinite computational power and a unique identifier.

At each round, each vertex can exchange a message with each of its neighbors (the process is synchronous, and there are no failure).

At the end, each node outputs its part of the solution, and we want to minimize the number of rounds of communication. $\rightarrow$ LOCAL

Theorem (Ghaffari, Kuhn, and Maus 2017)

If a graph has a network decomposition with C colors and diameter D, any randomized algorithm for a local problem can be derandomized by incurring a multiplicative cost of $O(C \cdot D)$ on the number of rounds.

Theorem (Ghaffari and Grunau 2024)

A network decomposition with $O(\log n)$ colors and diameter $O(\log n)$ can be computed deterministically in $O(\log^2 n)$ rounds.

Generating random spanning trees

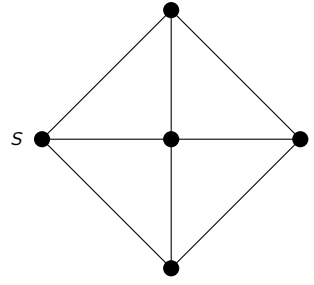
Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

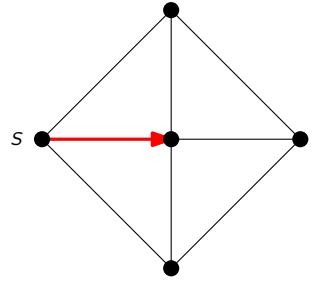
Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

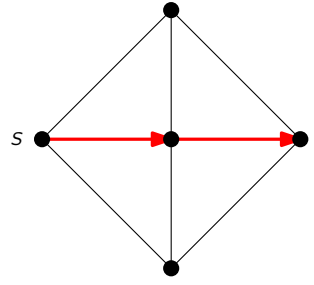
Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

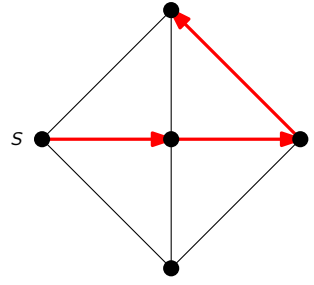
Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

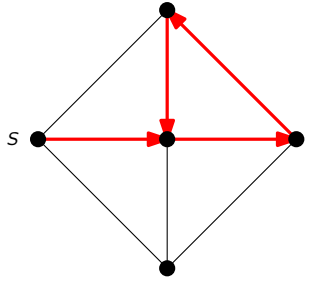
Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

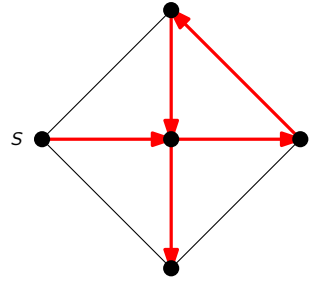
Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

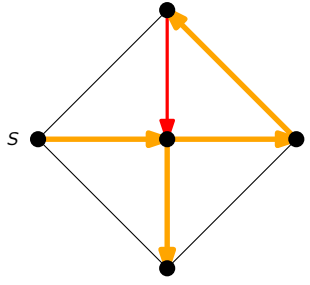
Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

This gives a $O(|E(G)| \cdot |V(G)|)$ time algorithm.

Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

This gives a $O(|E(G)| \cdot |V(G)|)$ time algorithm.

Kelner and Madry (2009) improved this to $O(|E(G)| \cdot \sqrt{|V(G)|})$

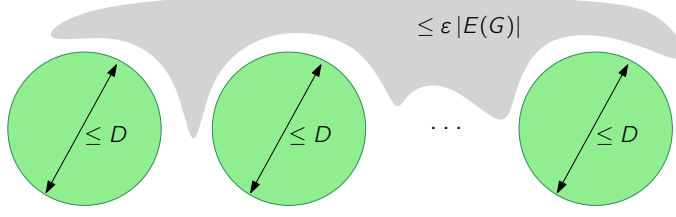
Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

This gives a $O(|E(G)| \cdot |V(G)|)$ time algorithm.

Kelner and Madry (2009) improved this to $O(|E(G)| \cdot \sqrt{|V(G)|})$



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

This gives a $O(|E(G)| \cdot |V(G)|)$ time algorithm.

Kelner and Madry (2009) improved this to $O(|E(G)| \cdot \sqrt{|V(G)|})$

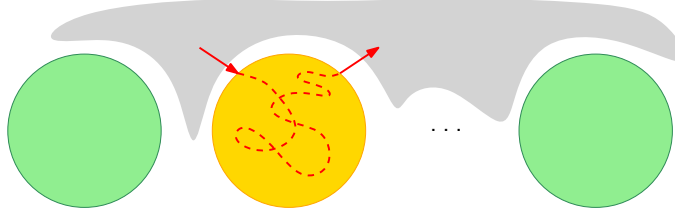
Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

This gives a $O(|E(G)| \cdot |V(G)|)$ time algorithm.

Kelner and Madry (2009) improved this to $O(|E(G)| \cdot \sqrt{|V(G)|})$



Generating random spanning trees

Theorem (Broder 1989 and Aldous 1990)

Perform a random walk W in G, starting at some vertex s, and stop when all the vertices of G have been visited. For each $v \neq s$, consider the first edge incident to v encountered by W. The union of these edges is a uniform random spanning tree of G.

This gives a $O(|E(G)| \cdot |V(G)|)$ time algorithm.

Kelner and Madry (2009) improved this to $O(|E(G)| \cdot \sqrt{|V(G)|})$

Schild (2018), and Anari, Liu, Oveis Gharan, Vinzant, and Vuong (2021) obtained new algorithms running in nearly linear time.

Conclusion

We have seen that low diameter decompositions had been studied and used in several areas, mostly independently between pure math and TCS:

- Geometric group theory
- Graph theory
- Distributed computing
- Approximation algorithms / Combinatorial optimization
- Random generation of combinatorial structures

Conclusion

We have seen that low diameter decompositions had been studied and used in several areas, mostly independently between pure math and TCS:

- Geometric group theory
- Graph theory
- Distributed computing
- Approximation algorithms / Combinatorial optimization
- Random generation of combinatorial structures

(And I haven't mentioned clustering in data analysis and machine learning)

Conclusion

We have seen that low diameter decompositions had been studied and used in several areas, mostly independently between pure math and TCS:

- Geometric group theory
- Graph theory
- Distributed computing
- Approximation algorithms / Combinatorial optimization
- Random generation of combinatorial structures

(And I haven't mentioned clustering in data analysis and machine learning)

In my opinion this illustrates the importance of having organizations & meetings bringing together researchers from very diverse backgrounds in the math-TCS spectrum, and facilitating the flow of ideas between the fields.