Graduate Seminar on Algorithms and Optimization (S4C3)

Metric Embeddings and Applications

Matthias Kaul Wenzheng Li László Végh

2026

Metric Embeddings

Problem

Given a metric space (X, d) *from a class* $\mathcal{X}$ *of metric spaces, and a class of metrics* $\widehat{\mathcal{X}}$*, find a metric* $(\widehat{X}, \widehat{d}) \in \mathcal{X}$ *and a mapping* $f : X \to \widehat{X}$ *such that*

$$\widehat{d}(f(x), f(y)) \approxeq d(x, y) \; \forall x, y \in X.$$

The values of $\mathcal{X}$, $\widehat{\mathcal{X}}$ and $\approxeq$ can vary a lot.

Metric Embeddings

A typical instantiation may look like this:

Problem

Given a finite metric space (X, d) *find a mapping* $f : X \to \mathbb{R}^n$ *such that*

$$\lambda_c \cdot d(x, y) \leq ||f(x) - f(y)||_p \leq \lambda_e \cdot d(x, y) \; \forall x, y \in X.$$

Where λ_e, λ_c are the *expansion* and *contraction* of the embedding respectively. If we can find such an embedding we say (X, d) embedds into $(\mathbb{R}^n, \ell_p)$ with *distortion* $\frac{\lambda_e}{\lambda_c}$.

Why do we want this?

- Allows us to give "simple" representations of a given metric.

Why do we want this?

- Allows us to give “simple” representations of a given metric.
- Computational problems may be easier if the metric is from a sufficiently restricted class $\mathcal{X}$.

Why do we want this?

- Allows us to give "simple" representations of a given metric.
- Computational problems may be easier if the metric is from a sufficiently restricted class $\mathcal{X}$.
- Metrics from $\mathcal{X}$ may require less memory to store.

Why do we want this?

- Allows us to give "simple" representations of a given metric.
- Computational problems may be easier if the metric is from a sufficiently restricted class $\mathcal{X}$.
- Metrics from $\mathcal{X}$ may require less memory to store.
- Rounding Linear (Semidefinite) Programs.

- The set of *all* n-point metrics is a polyhedral cone described by the triangle-inequalities and non-negativity

- The set of *all* n-point metrics is a polyhedral cone described by the triangle-inequalities and non-negativity
- The set of n-point ℓ_1-metrics is a polyhedral cone called CUT_n. A polynomial-time membership oracle for the cone is equivalent to $\mathrm{P} = \mathrm{NP}$

- The set of *all* n-point metrics is a polyhedral cone described by the triangle-inequalities and non-negativity
- The set of n-point ℓ_1-metrics is a polyhedral cone called CUT_n. A polynomial-time membership oracle for the cone is equivalent to $\mathrm{P} = \mathrm{NP}$

If we have a linear optimization problem over CUT_n, we can solve the problem in polynomial time over all metrics, and embed the solution into ℓ_1 with some distortion d. This gives a d-approximation algorithm for the original (presumably hard) problem.

The described approach yields naturally the following questions:

- Which classes of metric spaces can we optimize over?
- Which classes of metrics can we embed into each other with low distortion?

Talk Topics: Introduction

1. Bourgain's Theorem and the Sparsest-Cut Problem Lecture Notes, GNRS Thm 3.2

Talk Topics: Structural Graph Theory

2. Embedding Series-Parallel Graphs GNRS Section 4
3. The Okamura-Seymour Theorem; Embedding Outerplanar Graphs GNRS Section 5, Lecture Notes, Okamura-Seymour
4. Embedding Planar Graphs A face cover perspective to ℓ_1 embeddings of planar graphs
5. Sparse covers for minor-closed graph classes* On Sparse Covers of Minor Free Graphs, Low Dimensional Metric Embeddings, and other applications
6. Optimal Padded Decompositions for Minor-Free Graphs How to Protect Yourself from Threatening Skeletons: Optimal Padded Decompositions for Minor-Free Graphs

Talk Topics: Applications

7. The FRT Algorithm, tree embeddings A tight bound on approximating arbitrary metrics by tree metrics
8. FRT in near-linear time Efficient Construction of Probabilistic Tree Embeddings
9. Approximation algorithm for metrical task systems A polylog(n)-competitive algorithm for metrical task systems
10. Approximation algorithm for Group Steiner Tree A polylogarithmic approximation algorithm for the group Steiner tree problem
11. Embedding negative-type metrics to approximate Sparsest-Cut* Euclidean distortion and the Sparsest Cut
12. Using LDDs to solve Shortest-Path in near-linear time* Negative-Weight Single-Source Shortest Paths in Near-Linear Time

Structure of seminars

Each seminar session is structured as follows:

1. First part of the talk (10-20 minutes)
2. Questions
3. Second part of the talk (55-65 minutes)
4. Discussion

Parts 1 and 3 must not take more than 75 minutes in total.

Recall definitions and results from previous talks when you use them.

What we expect

- Understand every aspect of your topic.
- Prepare your talk on the assigned topic carefully, including questions to the audience
- Prepare a 1- or 2-page summary of your talk, with the most important definitions and results. Distribute hardcopies of this before your talk to the audience.
- Give a rehearsal talk about 2–3 weeks before your main talk.
- Participate actively in the discussions in the seminar.

Besides the text assigned to you, it is usually necessary and always helpful to study further sources (e.g., read other papers).

Topic assignment and registration

If you want to participate in this seminar, send an e-mail to Ulrich Brenner (brenner@dm.uni-bonn.de) with your name and your favorite topics no later than

Tuesday, February 10, before 14:00.

A few days later we will inform you by e-mail about the assignment of the topics.
After the assignment you have one week for the final registration. After that we may give your place to another student.
In addition, you have to register in BASIS in early April (before the seminar begins).
Each participant will be assigned an advisor (Matthias Kaul or Wenzheng Li) who can help with questions.