Data-driven Algorithm Design

Maria-Florina (Nina) Balcan
Carnegie Mellon University
Analysis and Design of Algorithms

Classic algo design: solve a worst case instance.

- Easy domains, have optimal poly time algos.
  E.g., sorting, shortest paths

- Most domains are hard.
  E.g., clustering, partitioning, subset selection, auction design, ...

Data driven algo design: use learning & data for algo design.

- Suited when repeatedly solve instances of the same algo problem.
Data Driven Algorithm Design

Data driven algo design: use learning & data for algo design.

- Different methods work better in different settings.
- Large family of methods - what’s best in our application?

Prior work: largely empirical.

- Artificial Intelligence:
  [Horvitz-Ruan-Gomes-Kautz-Selman-Chickering, UAI 2001]
  [Xu-Hutter-Hoos-LeytonBrown, JAIR 2008]

- Computational Biology: E.g., [DeBlasio-Kececioglu, 2018]

- Game Theory: E.g., [Likhodedov and Sandholm, 2004]
Our Contribution: Data Driven Algorithms with Provable Guarantees.

• Cases studies of widely used algo families. E.g,
  • Data science: clustering, partitioning problems.
  • Comp bio: sequence alignment and protein folding.
  • Auction and pricing problems: e.g., item pricings.
  • AI and OR: branch and bound techniques for solving MIPs.

• General analysis principles:
  • distributional learning (structure of dual class) and online learning (dispersed discontinuities of cost fns).

• Push boundaries of algorithm design and machine learning.
**Example: Clustering Problems**

**Clustering:** Given a set objects organize them into natural groups.

- E.g., cluster news articles, or web pages, or search results by topic.
- Or, cluster customers according to purchase history.
- Or, cluster images by who is in them.

Often need to solve such problems repeatedly.

- E.g., clustering news articles (Google news).
Example: Clustering Problems

Clustering: Given a set objects organize them into natural groups.

Objective based clustering  E.g., k-means, k-median, k-center

\textbf{\textit{k}-means}

**Input:** Set of objects \( S, d \)

**Output:** centers \( \{c_1, c_2, ..., c_k\} \)

To minimize \( \sum_p \min_i d^2(p, c_i) \)

Or minimize distance to ground-truth
Algorithm Design as Distributional Learning

Goal: given family of algos $F$, sample of typical instances from domain (unknown distr. $D$), find algo that performs well on new instances from $D$.

Large family $F$ of algorithms

Sample of typical inputs

Clustering:

Facility location:
Sample Complexity of Algorithm Selection

**Goal:** given family of algos $F$, sample of typical instances from domain (unknown distr. $D$), find algo that performs well on new instances from $D$.

**Approach:** ERM, find $\hat{A}$ near optimal algorithm over the set of samples.

**Key Question:** Will $\hat{A}$ do well on future instances?

**Sample Complexity:** How large should our sample of typical instances be in order to guarantee good performance on new instances?
Sample Complexity of Algorithm Selection

**Goal:** given family of algos $\mathcal{F}$, sample of typical instances from domain (unknown distr. $\mathcal{D}$), find algo that performs well on new instances from $\mathcal{D}$.

**Approach:** ERM, find $\hat{A}$ near optimal algorithm over the set of samples.

**Key tools from learning theory**

- **Uniform convergence:** for any algo in $\mathcal{F}$, average performance over samples “close” to its expected performance.
  - Imply that $\hat{A}$ has high expected performance.
  - $N = O(\text{dim}(\mathcal{F})/\epsilon^2)$ instances suffice for $\epsilon$-close.

$\text{dim}(\mathcal{F})$ (e.g. pseudo-dimension): ability of fns in $\mathcal{F}$ to fit complex patterns.
**Challenge:** “nearby” algos can have drastically different behavior.

**Challenge:** design a computationally efficient meta-algorithm.
Formal Guarantees for Algorithm Selection


Our results:

• New algorithm classes applicable for a wide range of problems (e.g., clustering, partitioning, alignment, auctions).

• General techniques for sample complexity based on properties of the dual class of fns.
Clustering: Linkage + Post-processing

Family of poly time 2-stage algorithms:

1. Greedy linkage-based algo to get hierarchy (tree) of clusters.

2. Fixed algo (e.g., DP or last k-merges) to select a good pruning.
Clustering: Linkage + Post-processing

1. Linkage-based algo to get a hierarchy.
2. Post-processing to identify a good pruning.

Both steps can be done efficiently.
Linkage Procedures for Hierarchical Clustering

**Bottom-Up (agglomerative)**

- Start with every point in its own cluster.
- Repeatedly merge the “closest” two clusters.

Different defs of “closest” give different algorithms.
Linkage Procedures for Hierarchical Clustering

Have a **distance** measure on pairs of objects.

\[ d(x,y) \] - distance between \( x \) and \( y \)

E.g., # keywords in common, edit distance, etc

- **Single linkage:** \( \text{dist}(A, B) = \min_{x \in A, x' \in B} \text{dist}(x, x') \)

- **Complete linkage:** \( \text{dist}(A, B) = \max_{x \in A, x' \in B} \text{dist}(x, x') \)

- **Parametrized family, \( \alpha \)-weighted linkage:**

\[
\text{dist}_{\alpha}(A, B) = (1 - \alpha) \min_{x \in A, x' \in B} d(x, x') + \alpha \max_{x \in A, x' \in B} d(x, x')
\]
**Clustering: Linkage + Post Processing**

**Our Results:** \( \alpha \)-weighted linkage + Post-processing  
[Balcan-Nagarajan-Vitercik-White, COLT 2017]

- Pseudo-dimension is \( O(\log n) \), so small sample complex.
- Given sample \( S \), find best algo from this family in poly time.

**Key Technical Challenge:** small changes to the parameters of the algo can lead to radical changes in the tree or clustering produced.

Problem: a single change to an early decision by the linkage algo, can snowball and produce large changes later on.
Claim: Pseudo-dim of $\alpha$-weighted linkage + Post-process is $O(\log n)$.

Key fact: If we fix a clustering instance of $n$ pts and vary $\alpha$, at most $O(n^8)$ switching points where behavior on that instance changes.

So, the cost function is piecewise-constant with at most $O(n^8)$ pieces.
**Claim:** Pseudo-dim of $\alpha$-weighted linkage + Post-process is $O(\log n)$.

**Key fact:** If we fix a clustering instance of $n$ pts and vary $\alpha$, at most $O(n^8)$ switching points where behavior on that instance changes.

**Key idea:**

- For a given $\alpha$, which will merge first, $\mathcal{N}_1$ and $\mathcal{N}_2$, or $\mathcal{N}_3$ and $\mathcal{N}_4$?

- Depends on which of $\alpha d(p, q) + (1 - \alpha)d(p', q')$ or $\alpha d(r, s) + (1 - \alpha)d(r', s')$ is smaller.

- An interval boundary an equality for 8 points, so $O(n^8)$ interval boundaries.
Clustering: Linkage + Post Processing

Claim: Pseudo-dim of $\alpha$-weighted linkage + Post-process is $O(\log n)$.

Key idea: For $m$ clustering instances of $n$ points, $O(mn^8)$ patterns.

- Pseudo-dim largest $m$ for which $2^m$ patterns achievable.

- So, solve for $2^m \leq m n^8$. Pseudo-dimension is $O(\log n)$. 
Clustering: Linkage + Post Processing

**Claim:** Pseudo-dim of $\alpha$-weighted linkage + Post-process is $O(\log n)$.

For $N = O(\log n / \epsilon^2)$, w.h.p. expected performance cost of best $\alpha$ over the sample is $\epsilon$-close to optimal over the distribution.

**Claim:** Given sample $S$, can find best algo from this family in poly time.

- Solve for all $\alpha$ intervals over the sample.
  
  $\alpha \in [0,1]

- Find $\alpha$ interval with smallest empirical cost.
Learning Both Distance and Linkage Criteria

- Often different types of distance metrics.
  - Captioned images, \(d_0\) image info, \(d_1\) caption info.
  - Handwritten images: \(d_0\) pixel info (CNN embeddings), \(d_1\) stroke info.

Family of Metrics: Given \(d_0\) and \(d_1\), define

\[
d_\beta(x, x') = (1 - \beta) \cdot d_0(x, x') + \beta \cdot d_1(x, x')
\]

Parametrized \((\alpha, \beta)\)-weighted linkage (\(\alpha\) interpolation between SL and CL and \(\beta\) interpolation between two metrics): [Balcan-Dick-Lang, 2019]

\[
dist_\alpha(A, B; d_\beta) = (1 - \alpha) \min_{x \in A, x' \in B} d_\beta(x, x') + \alpha \max_{x \in A, x' \in B} d_\beta(x, x')
\]

Claim: Pseudo-dim. of \((\alpha, \beta)\)-weighted linkage is \(O(\log n)\).

Significant improvement in practice for standard datasets.
Partitioning Problems via IQPs

**IQP formulation**

\[
\text{Max } x^T A x = \sum_{i,j} a_{i,j} x_i x_j \\
\text{s.t. } x \in \{-1,1\}^n
\]

Many of these pbs are NP-hard.

E.g., **Max cut**: partition a graph into two pieces to maximize weight of edges crossing the partition.

**Input**: Weighted graph \( G, w \)

**Output**: \( \text{Max } \sum_{(i,j) \in E} w_{ij} \left( \frac{1-v_i v_j}{2} \right) \\
\text{s.t. } v_i \in \{-1,1\} \)

1 if \( v_i, v_j \) opposite sign, 0 if same sign

var \( v_i \) for node i, either +1 or -1
Parametrized family of rounding procedures

**1. SDP relaxation:**

Associate each binary variable $x_i$ with a vector $u_i$.

$$\max \sum_{i,j} a_{i,j} \langle u_i, u_j \rangle$$

subject to $\|u_i\| = 1$

**2. s-Linear Rounding**

[Feige&Landberg’06]

Inside margin, randomly round

Outside margin, round to -1.

**Algorithmic Approach: SDP + Rounding**

**IQP formulation**

$$\max x^T A x = \sum_{i,j} a_{i,j} x_i x_j$$

s.t. $x \in \{-1, 1\}^n$
Partitioning Problems via IQPs

**Our Results: SDP + s-linear rounding**  [Balcan-Nagarajan-Vitercik-White, COLT 2017]

Pseudo-dimension is $O(\log n)$, so small sample complexity.

**Key idea:** expected IQP objective value is piecewise quadratic in $\frac{1}{s}$ with $n$ boundaries.

Given sample $S$, can find best algo from this family in poly time.

- Solve for all $\alpha$ intervals over the sample, find best parameter over each interval, output best parameter overall.
Other Parametrized Families of Algorithms

**Clustering:** parametrized Lloyd's method  
[Balcan-Dick-White, NeurIPS 2018]

**Alignment pbs (e.g., string alignment):** parametrized dynamic prog.  
[Balcan-DeBlasio-Dick-Kingsford-Sandholm-Vitercik, 2019]

**MIPs:** Branch and Bound Techniques  
[Balcan-Dick-Sandholm-Vitercik, ICML'18]

$$\begin{align*}
\text{Max } & \ c \cdot x \\
\text{s.t. } & \ Ax = b \\
& \ x_i \in \{0,1\}, \forall i \in I
\end{align*}$$

Automated mechanism design for revenue maximization  
Parametrized VCG auctions, posted prices, lotteries.  
[Balcan-Sandholm-Vitercik, EC 2018]
Data-driven Mechanism Design

- **Similar ideas** for sample complex of data-driven mechanism design for revenue maximization. [Balcan-Sandholm-Vitercik, EC'18]

- Pseudo-dim of \( \{\text{revenue}_M : M \in \mathcal{M}\} \) for multi-item multi-buyer settings.
  - Many families: second-price auctions with reserves, posted pricing, two-part tariffs, parametrized VCG auctions, etc.

- **Key insight**: dual function sufficiently structured.
  - For a fixed set of bids, revenue is piecewise linear function of parameters.

---

**2nd-price auction with reserve**

![Graph showing revenue as a function of bids and reserves](Image)

**Posted price mechanisms**

![Graph showing revenue as a function of prices](Image)
General Sample Complexity via Dual Classes

[Balcan-DeBlasio-Dick-Kingsford-Sandholm-Vitercik, 2019]

High level learning theory bit

- Want to prove that for all algorithm parameters $\alpha$:

$$\frac{1}{|S|} \sum_{I \in S} \text{cost}_\alpha(I) \text{ close to } \mathbb{E}[\text{cost}_\alpha(I)].$$

- Function class whose complexity want to control: $\{\text{cost}_\alpha : \text{parameter } \alpha\}$.

- Proof takes advantage of structure of dual class $\{\text{cost}_I : \text{instances } I\}$.

\[\text{cost}_I(\alpha) = \text{cost}_\alpha(I)\]

$\alpha \in \mathbb{R}$
**Thm:** Assume \( \text{cost}_I(\alpha) \): boundary fns \( f_1, f_2, \ldots, f_N \in F \) s.t. within each region, \( \text{cost}_I(\alpha) = g(\alpha) \) for some \( g \in G \).

\[
P\dim(\{\text{cost}_\alpha(I)\}) = \tilde{O}(d_{F^*} + d_{G^*} + d_{F^*} \log N)
\]
Online Algorithm Selection


- **Challenge**: scoring fns non-convex, with lots of discontinuities.

Cannot use known techniques.

- Identify general properties (piecewise Lipschitz fns with dispersed discontinuities) sufficient for strong bounds.

  - Show these properties hold for many alg. selection pbs.
{u₁(·),...,u_T(·)} is (w,k)-dispersed if any ball of radius w contains boundaries for at most k of the uᵢ.
Summary and Discussion

• Strong performance guarantees for data driven algorithm selection for combinatorial problems.

• Provide and exploit structural properties of dual class for good sample complexity.

• Learning theory: techniques of independent interest beyond algorithm selection.
Discussion, Open Problems

• Analyze other widely used classes of algorithmic paradigms.

• Other learning models (e.g., one shot, domain adaptation, RL).

• Explore connections to program synthesis; automated algo design.

• Explore connections to Hyperparameter tuning, AutoML, MetaLearning.

  Use our insights for pbs studied in these settings (e.g., tuning hyper-parameters in deep nets)