Learning to Prune: Speeding up Repeated Computations

Daniel Alabi, Adam Tauman Kalai, Katrina Ligett, Cameron Musco, Christos Tzamos, and Ellen Vitercik

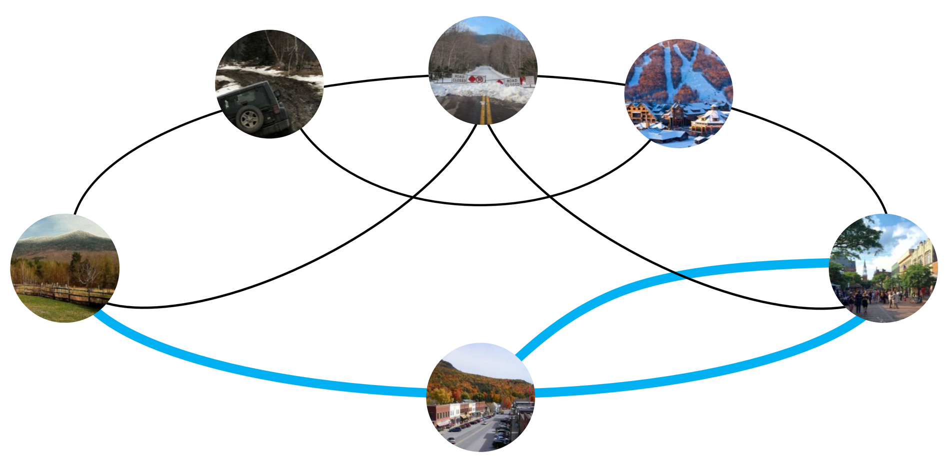
COLT 2019

Speeding up Repeated Computations

Goal: Solve sequence of similar computational problems, exploiting common structure

Typically, large swaths of search space never optimal
Learn to ignore them!

- Shortest path always in specific region of road network
- Only handful of LP constraints ever bind
- Large portions of DNA never contain patterns of interest



Model

Function $f: X \to Y$ maps problem instances x to solutions y

Learning algorithm receives sequence $x_1, \dots, x_T \in X$
E.g., each x_i equals edge weights for a fixed graph

Goal:
Correctly compute f on *most* rounds, minimizing runtime
Worst-case algorithm would compute $f(x_i)$ for each x_i

Assume access to other functions mapping $X \to Y$
- Faster to compute
- Defined by subsets (*prunings*) S of universe $\mathcal{U}$
  - Universe $\mathcal{U}$ represents entire search space
  - Denote corresponding function $f_S: X \to Y$
  - $f_\mathcal{U} = f$

Example:
$\mathcal{U}$ = all edges in fixed graph
S = subset of edges

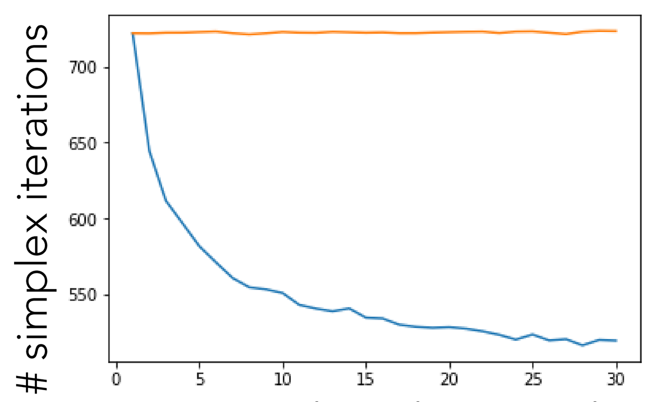
Assume exists $S^*(x) \subseteq \mathcal{U}$ where $f_S(x) = f(x)$ iff $S^*(x) \subseteq S$
- "Minimally pruned set"
- E.g., the shortest path

Algorithm

1. Initialize pruned set $\bar{S}_1 \leftarrow \emptyset$
2. For each round $i \in \{1, \dots, T\}$:
   a. Receive problem instance x_i
   b. With probability $1/\sqrt{i}$, **explore**:
      i. Output $f(x_i)$
      ii. Compute minimally pruned set $S^*(x_i)$
      iii. Update pruned set: $\bar{S}_{i+1} \leftarrow \bar{S}_i \cup S^*(x_i)$
   c. Otherwise (with probability $1 - 1/\sqrt{i}$), **exploit**:
      i. Output $f_{\bar{S}_i}(x_i)$
      ii. Don't update pruned set: $\bar{S}_{i+1} \leftarrow \bar{S}_i$

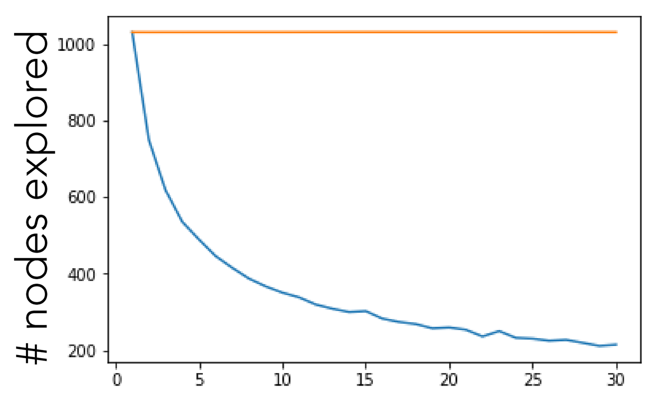
Experiments

Linear programming



Top line: Simplex
Bottom line: Our algorithm
Linear programs:
204 variables, 946 constraints
Fraction of mistakes:
0.018 over 5000 runs with $T = 30$

Shortest path routing



Top line: Dijkstra's algorithm
Bottom line: Our algorithm
Fraction of mistakes:
0.068 over 5000 runs with $T = 30$

Guarantees

Recap: At round i, algorithm outputs $f_{S_i}(x_i)$
S_i depends on $x_{1:i}$

Goal 1: Minimize $|S_i|$
Time it takes to compute $f_{S_i}(x_i)$ typically grows with $|S_i|$

Theorem:
$\mathbb{E}\left[\frac{1}{T}\sum_{i=1}^T |S_i|\right] \le |S^*| + \frac{|\mathcal{U}| - |S^*|}{\sqrt{T}}$, where $S^* = \bigcup_{i=1}^T S^*(x_i)$

Proof:

$$\mathbb{E}[|S_i|] = \frac{1}{\sqrt{i}}|\mathcal{U}| + \left(1 - \frac{1}{\sqrt{i}}\right)\mathbb{E}[|\bar{S}_i|] \le \frac{1}{\sqrt{i}}|\mathcal{U}| + \left(1 - \frac{1}{\sqrt{i}}\right)|S^*|$$

Goal 2: Minimize # of mistakes
Rounds where $f_{S_i}(x_i) \ne f(x_i)$

Theorem:
$\mathbb{E}[\text{\# of mistakes}] \le \frac{|S^*|}{\sqrt{T}}$, where $S^* = \bigcup_{i=1}^T S^*(x_i)$
S^* is smallest set S where $f_S(x_i) = f(x_i)$ for all i

Proof sketch:
- For $e \in S^*$, let $N_T(e)$ be # of times $e \notin S_i$ but $e \in S^*(x_i)$
- When makes mistake, must be $e \in S^*(x_i)$ with $e \notin S_i$
  - Otherwise, $S_i \supseteq S^*(x_i)$, so no mistake
  - This means $N_T(e)$ += 1
- Therefore, $\mathbb{E}[\text{\# of mistakes}] \le \sum_{e \in S^*} \mathbb{E}[N_T(e)]$
- We prove $\mathbb{E}[N_T(e)] \le \sum_{r=1}^T \left(1 - \frac{1}{\sqrt{T}}\right)^r \le \frac{1}{\sqrt{T}}$
  - *If $e \notin \bar{S}_i$, then $e \notin \bar{S}_j$ for $j \le i$*
  - This means $\mathbb{E}[\text{\# of mistakes}] \le \frac{|S^*|}{\sqrt{T}}$

Goal: Route from top to bottom star. **Black nodes:** Pruned subgraph. **Grey nodes:** Nodes Dijkstra explores over 30 rounds.