The (im)possibility of simple search-to-decision reductions for approximation problems

Spencer Peters, and
Optimization Problems

• Any optimization problem comes in (at least!) two flavors.

• Is search (argmin) harder than decision (min)?

• In this talk, we’ll consider limited, black-box access to \( f \).

• **Search**: find \( x^* \) such that
  \[
  f(x^*) = \min_{x \in \{0,1\}^n} f(x)
  \]

• **Decision**: compute
  \[
  \min_{x \in \{0,1\}^n} f(x)
  \]

• “Weak Decision”: decide if
  \[
  \min_{x \in \{0,1\}^n} f(x) \leq r
  \]
A search-to-decision reduction

- Uses only linearly many MIN queries ($2n$).
- In fact, linear time!
- “Instance-wise equivalence”
- Applies directly to many NP-optimization problems, like Max-SAT.

\begin{tikzpicture}
  \node (root) at (0,0) {
    $*$ MIN = 3
  }
  \node (zero) at (-1.5,-1) {
    0... MIN = 5
  }
  \node (one) at (-0.5,-1) {
    1... MIN = 3
  }
  \node (ten) at (0.5,-1) {
    10...
  }
  \node (eleven) at (1.5,-1) {
    11...
  }
  \node (tenone) at (1,-2) {
    10...1 MIN = 3
  }
  \node (psi) at (4,-2) {
    Boolean constraints $\psi$
  }
  \node (psi_zero) at (4,-3) {
    $\psi$'s with $x_i \mapsto 0$
  }
  \node (psi_one) at (4,-4) {
    $\psi$'s with $x_i \mapsto 1$
  }

  \draw[->] (root) -- (zero);
  \draw[->] (root) -- (one);
  \draw[->] (zero) -- (ten);
  \draw[->] (zero) -- (eleven);
  \draw[->] (one) -- (ten);
  \draw[->] (one) -- (eleven);
  \draw[->] (ten) -- (tenone);
  \draw[->] (eleven) -- (tenone);
  \draw[->] (psi) -- (psi_zero);
  \draw[->] (psi) -- (psi_one);
\end{tikzpicture}
What about approximate Optimization?

• $\gamma$-Approximation: find $x$:
  $$f(x) \leq \gamma \cdot \text{MIN}$$

• $\gamma$-Estimation: compute $r$:
  $$\text{MIN} \leq r \leq \gamma \cdot \text{MIN}$$

• Is Approximation harder than Estimation?
The “greedy” reduction

- Let’s say $\gamma = 2$.
- Still linear queries and linear time 😊
- What about the Approximation factor $\gamma'$ we achieve?
The “greedy” reduction

- What about the \textbf{Approximation} factor $\gamma'$ we achieve?
- $\gamma'$ could be as large as $\gamma^n$!
- Can we do better?
The “greedy” reduction

- $k$ branches rather than 2
- Recurse on the leaf with the minimal estimate
- Depth is roughly $n / \log k$.
- $\gamma' \approx \gamma^{n / \log k}$
- Pay in increased number of queries $q = kn / \log k$
- In the typical case $k \gg n$: $\gamma' \approx \gamma^{n / \log q}$
- Still has applications! [Ste16]
- Question: Is greedy optimal?
Branch-and-bound algorithms

- At a high level, a generalization of the “greedy” reduction!
- Practical. (e.g., [MJSE16]). Used for combinatorial optimization problems like TSP, MaxCSPs [BMHW21, Cook16]
- Question: how powerful are “black-box” branch-and-bound algorithms?
Our Model

Let $\mathcal{F}$ be a class of functions $f: \{0, 1\}^n \rightarrow \mathbb{R}_{\geq 0}$. Let $\mathcal{S}$ be a class of “estimable” subsets of the domain.

Given an oracle $h_f: \mathcal{S} \rightarrow \mathbb{R}_{\geq 0}$ satisfying $\min_{S} f \leq h_f(S) \leq \gamma \cdot \min_{S} f$, (and no other access to $f$!)

how many oracle queries $q$ are needed to find $f(x) \leq \gamma' \min_{\{0,1\}^n} f$ ?

(with constant probability, in the worst case over $f$ and $h_f$.)

”Black-box branch and bound” model.

• Both weaker and stronger than real-world BB algorithms
• Weaker: only access to $f$ through the oracle
• Stronger: have access to a powerful oracle!
### Our Results

- For arbitrary $f$, greedy is optimal!
- A tight lower bound for the Traveling Salesperson Problem (TSP).
- A strong lower bound for Max-Constraint Satisfaction Problems.

<table>
<thead>
<tr>
<th>Class $\mathcal{F}$</th>
<th>Queries $\mathcal{S}$</th>
<th>Rough tradeoff</th>
<th>Precise bounds*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arbitrary</td>
<td>Arbitrary</td>
<td>$\gamma' \approx \gamma^{n/\log q}$</td>
<td>$\gamma' = \gamma^{\frac{n}{\ell} + O(1)} \Rightarrow O\left(\frac{2^\ell}{\ell}\right) \leq q \leq O(n \cdot \frac{2^\ell}{\ell})$</td>
</tr>
<tr>
<td>Traveling Salesperson</td>
<td>Partial tours</td>
<td>$\gamma' \approx \gamma n / \log q$</td>
<td>$\Omega((\gamma - 1)n / \log q) \leq \gamma' \leq \gamma n / (\log(q) - 1)$</td>
</tr>
<tr>
<td>Max-CSPs</td>
<td>Partial assignments</td>
<td>$\gamma \approx 1 + \sqrt{\log(q)/n}$</td>
<td>No nontrivial reductions, unless $q \geq \exp(-O((\gamma - 1)^2n))$</td>
</tr>
</tbody>
</table>
Useless Oracles

• **Idea:** Find $\mathcal{D} \in \Delta(\mathcal{F})$ such that for every $S$, the min of $f \leftarrow \mathcal{D}$ over $S$ is overwhelmingly likely to fall in a fixed interval of width $\gamma$.

• Then, for $f \leftarrow \mathcal{D}$, a $\gamma$-estimation oracle $h_f$ is useless! You know in advance what it’s going to tell you.

• How do we make this intuition formal?
Useless Oracles

• Generalizing the intuition from last slide, any oracle $\mathcal{O}$ is useless if most of its answers are predictable!

Useless Oracle Lemma:

• **IF** predictable:

  $\exists$ a fixed function $g$, $\forall x$, $\Pr_{\mathcal{O} \leftarrow \mathcal{D}} [\mathcal{O}(x) = g(x)] \geq 1 - p$,  

• **THEN** useless: $\forall$ oracle algorithms $\mathcal{A}$ making at most $q$ queries,

  \[ d_{TV}(\mathcal{A}^\mathcal{O}(), \mathcal{A}^g()) \leq pq. \]
Useless Estimators

- **Goal**: Find $\mathcal{D} \in \Delta(\mathcal{F})$ such that for every $S$, the min of $f \leftarrow \mathcal{D}$ over $S$ falls in some interval $[z_S, \gamma z_S]$ with large probability $\geq 1 - p$.
- When it does, can set $h_f(S) = g(S) := \gamma \cdot z_S$.
- By the useless oracle lemma, any $\mathcal{A}$ making $\leq q$ queries satisfies
  \[ d_{TV}(\mathcal{A}^{h_f}, \mathcal{A}^{g}) \leq pq. \]
- Since $\mathcal{A}^g$ is independent of $f$, we’re (almost) done!
- Last step: show no fixed $x^*$ independent of $f$ does well.
“Greedy” is optimal

- **Goal**: Find $\mathcal{D} \in \Delta(\mathcal{F})$ such that for every $S$, the min of $f \leftarrow \mathcal{D}$ over $S$ falls in some interval $[z_S, \gamma z_S]$ with probability $\geq 1 - p$.
- For each $x$, set $f(x) = \gamma^i$ independently with some probabilities $p_i$.
- Notice that then the distribution of $\min_S f$ only depends on $|S|$.
- Carefully choose rapidly increasing $p_i \propto 2^i$ so that:
  - For each $|S|$, there is an $i$ such that
    - Very likely to have $\gamma^i \in f(S)$, but
    - Very unlikely to have $\gamma^{i-2} \in f(S)$ (or any smaller value)
    - So can almost always set $h_f(S) = \gamma^i$ independently of $f$. 

Traveling Salesperson

• **Problem:** Given a complete undirected graph on \( n \) nodes along with edge costs, find a Hamiltonian cycle (complete tour) of (approx.) minimum weight.

• **Model:** Queries \( S_p \) consist of all tours extending a path \( p \).

• **Hard Distribution:** For each edge, flip a coin and assign either \( c(e) = 1 \) or \( c(e) \approx \gamma n / \log q \). Short paths don’t move the needle, and long paths have concentrated weight.

• **Matching (inefficient) algorithm:** Query all paths of length \( \ell \approx \log q \) and (inefficiently) find the cycle minimizing the sum of path estimates.
Max-Constraint Satisfaction Problems

• **Problem:** Given constraints from some family on $n$ Boolean variables, find an assignment that satisfies as many as possible.

• **Model:** Queries $S_w$ extending a *partial assignment* $w$.

• **Hard Distribution:** Sample independent constraints consistent with a random planted assignment.

• **Exceptions:** “Trivially unsatisfiable” families—queries can leak the instance $f$. 
Open Questions

• The most obvious direction is to study more function classes $\mathcal{F}$.
• Average-case results?
• More interestingly, could we make richer models of branch-and-bound algorithms that still have provable lower bounds?
Thank you!

• Feel free to follow up with me at speters@cs.cornell.edu
References


• [Ste16]: Noah Stephens-Davidowitz. *Search-to-decision reductions for lattice problems with approximation factors (slightly) greater than one*. In APPROX, 2016