Query Complexity of Tournament Solutions

Palash Dey
Tata Institute of Fundamental Research, Mumbai

February 8, 2017

31st AAAI Conference on Artificial Intelligence (AAAI-17), San Francisco, USA
Tournament Graphs

Directed graph with **exactly one edge** between every pair of vertices
Tournament Graphs: Examples

Example 1
- Vertices: set of items on Amazon
- Edges: $x$ to $y$ if majority of users prefer $x$ over $y$

Example 2
- Vertices: set of drugs for cancer
- Edges: $x$ to $y$ if expected lifetime with $x$ is more than $y$
Tournament Solutions

- $\mathcal{T}(V)$: set of tournaments on the vertex set $V$
- Tournament solutions: $f : \mathcal{T}(V) \to 2^V \setminus \{\emptyset\}$

Examples

- Condorcet non-loser: vertices with out-degree $\geq 1$
- Copeland set: vertices with highest out-degree

$\blacksquare$
Our Model

- Vertex set $V$ known
- Edge set $E$ \textbf{NOT known} — given via oracle

$x, y \in V \quad \xrightarrow{\text{Oracle}} \quad (x \rightarrow y) \in E$

$(y \rightarrow x) \in E$
Our Model

- Vertex set $V$ known
- Edge set $E$ **NOT known** — given via oracle

**Goal:** Query Complexity of Common Tournament Solutions
Our Model

- Vertex set $V$ known
- Edge set $E$ **NOT known** — given via oracle

$\forall x, y \in V \Rightarrow (x \to y) \in E$

$\forall x, y \in V \Rightarrow (y \to x) \in E$

**Goal:** Query Complexity of Common Tournament Solutions

Minimum number of oracle queries needed to compute a tournament solution.
Motivation for Our Model

- Often only vertex set known
- Learning direction of edges can be costly
- Want to minimize the number of queried edges
State-of-the-art

- $n$: the number of vertices

- Query complexity of Condorcet solution is

$$2n - \lfloor \log_2 n \rfloor - 2^{iii}$$

If $\exists x \in V$ with $\text{out-deg}(x) = n - 1$, then $\{x\}$

Otherwise $V$

- Corollary: Query complexity of Condorcet non-loser set is

$$2n - \lfloor \log_2 n \rfloor - 2$$

---


Query Complexity: Copeland set, Slater set

- **Copeland set**: vertices with highest out-degree
- **Slater set**: maximum elements of Slater orders
  - Slater order: A complete order $\succ$ with maximum agreement with $\mathcal{T}$
  - Copeland set: $\{A, C\}$
  - Slater set: $\{A, C\}$; Slater orders are
    - $A \succ C \succ D \succ B$
    - $C \succ D \succ A \succ B$
**Query Complexity: Copeland set, Slater set**

**Observation:** The out-degree of every vertex in Copeland set and Slater set is at least $\frac{n-1}{2}$

**Proof.**

- **Copeland set:** highest out-degree is at least $\frac{n-1}{2}$

- **Slater set:** Let $\text{out-deg}(x) < \frac{n-1}{2}$. Then $x \succ \cdots$ agrees less with $\mathcal{T}$ than $\cdots \succ x$. 


Query Complexity: Copeland set, Slater set

Observation: The out-degree of every vertex in Copeland set and Slater set is at least $\frac{n-1}{2}$

Proof.

- Copeland set: highest out-degree is at least $\frac{n-1}{2}$

- Slater set: Let out-deg($x$) $< \frac{n-1}{2}$. Then $x \succ \cdots$ agrees less with $\mathcal{T}$ than $\cdots \succ x$.

Theorem: Query complexity of Copeland set and Slater set are $\binom{n}{2}$

Proof.
Consider tournament where out-degree of every vertex is $\frac{n-1}{2}$
Query Complexity: Bipartisan set

Tournament induces a zero sum game between two players
- player i chooses vertex $x_i$ simultaneously, $i \in [2]$

- utility of player 1 =
  \[
  \begin{cases} 
  0 & \text{if } x_1 = x_2 \\
  +1 & \text{if } (x_1 \rightarrow x_2) \in E \\
  -1 & \text{if } (x_2 \rightarrow x_1) \in E 
  \end{cases}
  \]

- Bipartisan set of $T$: $\text{BP}(T) = \{x \in V : p_T(x) > 0\}$ called maximal lottery of $T$
Query Complexity: Bipartisan set

Tournament induces a zero sum game between two players

- player $i$ chooses vertex $x_i$ simultaneously, $i \in [2]$

- utility of player 1 = \[
\begin{cases}
0 & \text{if } x_1 = x_2 \\
+1 & \text{if } (x_1 \rightarrow x_2) \in E \\
-1 & \text{if } (x_2 \rightarrow x_1) \in E
\end{cases}
\]

- $\Delta(V)$: set of probability distributions over $V$

- $\mathcal{A}(T)$: adjacency matrix of $T$

- $\mathcal{S}(T)$: skew-adjacency matrix of $T = \mathcal{A}(T) - \mathcal{A}(T)^t$

- Minimax Theorem: \exists unique $p_{T} \in \Delta(V)$ such that

\[\sum_{x, y \in V} p_T(x)q(y)g_{xy} \geq 0 \quad \forall q \in \Delta(V), \mathcal{S}(T) = (g_{xy})_{x, y \in V}\]
Query Complexity: Bipartisan set

Tournament induces a zero sum game between two players

- player $i$ chooses vertex $x_i$ simultaneously, $i \in [2]$

- utility of player 1 =
\[
\begin{cases}
0 & \text{if } x_1 = x_2 \\
+1 & \text{if } (x_1 \rightarrow x_2) \in E \\
-1 & \text{if } (x_2 \rightarrow x_1) \in E
\end{cases}
\]

- $\Delta(V)$: set of probability distributions over $V$

- $\mathcal{A}(T)$: adjacency matrix of $T$

- $\mathcal{G}(T)$: skew-adjacency matrix of $T = \mathcal{A}(T) - \mathcal{A}(T)^t$

- Minimax Theorem: $\exists$ unique $p_T \in \Delta(V)$ such that
\[
\sum_{x,y \in V} p_T(x)q(y)g_{xy} \geq 0 \quad \forall q \in \Delta(V), \mathcal{G}(T) = (g_{xy})_{x,y \in V}
\]

Bipartisan set of $T$: $BP(T) = \{x \in V : p_T(x) > 0\}$
Query Complexity: Bipartisan set

Tournament induces a **zero sum game** between two players

- player $i$ chooses vertex $x_i$ simultaneously, $i \in [2]$

- utility of player 1 = \[
\begin{cases}
0 & \text{if } x_1 = x_2 \\
+1 & \text{if } (x_1 \rightarrow x_2) \in E \\
-1 & \text{if } (x_2 \rightarrow x_1) \in E
\end{cases}
\]

- $\Delta(V)$: set of probability distributions over $V$

- $A(T)$: adjacency matrix of $T$

- $G(T)$: skew-adjacency matrix of $T = A(T) - A(T)^t$

- Minimax Theorem: $\exists$ **unique** $p_T \in \Delta(V)$ such that

$$\sum_{x,y \in V} p_T(x)q(y)g_{xy} \geq 0 \quad \forall q \in \Delta(V), G(T) = (g_{xy})_{x,y \in V}$$

**Bipartisan set of $T$:** $BP(T) = \{x \in V : p_T(x) > 0\}$
Theorem: Query complexity of Bipartisan set is $\Omega(n^2)$

Proof sketch:

Lemma 1: Let $x, y \in V$ and $(x \rightarrow z) \Rightarrow (y \rightarrow z)$ $\forall z \in V$. Then $x \not\in BP(T)$.

Proof.

$y$ strictly dominates $x$. 


Theorem: Query complexity of Bipartisan set is $\Omega(n^2)$

Proof sketch cont.:

**Lemma 2**: A tournament $\mathcal{T} = (V, E)$ be such that:

- $V = V_1 \uplus V_2$ a partition of $V$
- $|E \cap V_1 \times V_2| \leq \frac{|V_1||V_2|}{2} - 1$
- $\text{out-deg}(x) = \frac{|V_1|-1}{2}$ for every $x \in \mathcal{T}[V_1]$  

Then $BP(T) \cap V_2 \neq \emptyset$. 
Query Complexity: Bipartisan set

**Theorem:** Query complexity of Bipartisan set is $\Omega(n^2)$

Proof sketch cont.:

- $V = V_1 \cup V_2$, $|V_1| = |V_2| = n$
- $\mathcal{T}[V_1]$ and $\mathcal{T}[V_2]$ is “fixed.” In degree of every vertex in $\mathcal{T}[V_1]$ and $\mathcal{T}[V_2]$ is $\frac{n-1}{2}$
- For query on $(x, y) \in V_1 \times V_2$, oracle answers $(x \rightarrow y) \in E$
Query Complexity: Bipartisan set

**Theorem:** Query complexity of Bipartisan set is $\Omega(n^2)$

Proof sketch cont.: Suppose algorithm makes $< \frac{n^2}{2}$ queries and outputs $S$

- If $S \cap V_2 \neq \emptyset$, then make $(x \rightarrow y) \in E \forall (x, y) \in V_1 \times V_2$

- Otherwise, make $(y \rightarrow x) \in E \forall$ not queried $(x, y) \in V_1 \times V_2$
Improving Query Complexity: Small top Cycle

**Top cycle:** $B \subset V$ dominant if $(x \rightarrow y) \in E$ for every $x \in B, y \in V \setminus B$. Top cycle is the smallest dominant set.

**Theorem:** Query complexity of top cycle is $\Theta(n^2)$

**Theorem:** If size of top cycle is $k$, then top cycle can be found with $\Theta(nk)$ queries

**Theorem:** If size of top cycle is $k$, then Copeland set, Bipartisan set can be found with $\Theta(nk)$ queries
Conclusion

Summary

- Finding common tournament solutions requires querying almost every edge
- If the tournament has small a top cycle, then query complexity of common tournament solutions can be reduced
Conclusion

Summary

- Finding common tournament solutions requires querying almost every edge
- If the tournament has small a top cycle, then query complexity of common tournament solutions can be reduced

Future work

- Interesting to find average query complexity of common tournament solutions
Thank You!

palash.ju.dey@gmail.com