Learning Voting Trees

Ariel D. Procaccia, Aviv Zohar, Yoni Peleg, and Jeffrey S. Rosenschein

Charalampos S. Nikolaou
charnik@di.uoa.gr

Department of Informatics and Telecommunications
N.K.U.A

January 23, 2008

Outline

- Introductory notes (Computational Social Choice)
- Voting Rules and Voting Trees
- Automated design of Voting Rules by Learning
- Finding a Voting Tree
- Conclusions
- Applications

Notation

- Election: set of **voters** $N = \{1, \ldots, N\}$, **candidates/alternatives** $A = \{x_1, \ldots, x_m\}$.
- Voters have linear preferences R^i.
- Winner: determined by a **voting rule**, a function **f** from rankings to alternatives.

Voting Rules

- Plurality: each voter gives one point to the alternative it ranks first (real-life elections).

Pairwise elections

a **beats** b if the majority of voters prefers a to b.

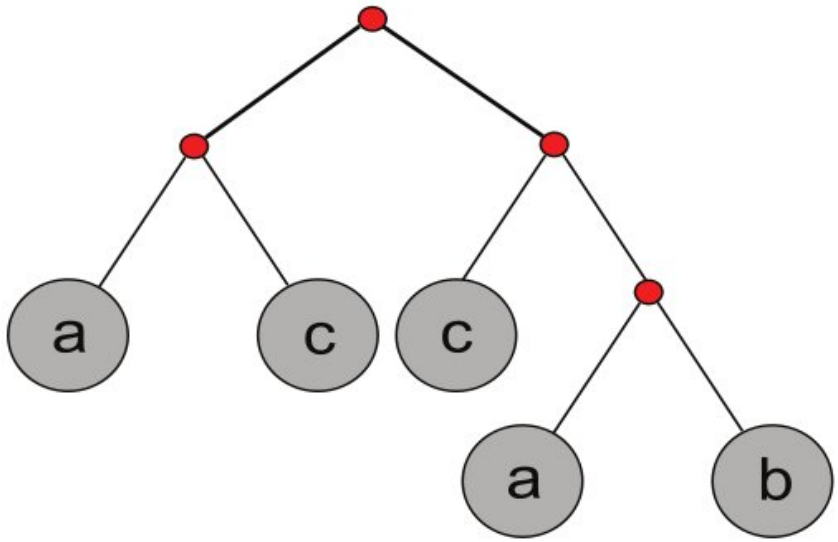
- Copeland: candidate's score is the number of candidates that beats.
- Condorcet Winner: who he beats every other (does not always exist – condorcet paradox).

Tournaments

- A tournament ($\succ$) over A is a complete binary irreflexive relationship ($a \succ b$ iff $\mathrm{a\ beats\ b}$).
- Example: $N = \{1, 2, 3\}, A = \{a, b, c\}$
  - Voter 1: c, b, a
  - Voter 2: b, a, c
  - Voter 3: a, c, b
  - Overall: $b \succ a$, $c \succ b$, $a \succ c$ (Condorcet paradox).
- (Pairwise) **voting rule** is a function from tournaments ($T = T(A)$) to candidates (A).
- $F : T \rightarrow A$, the class of all voting rules.

Binary Voting Trees

- Provide a succinct representation of voting rules.
- Not all voting rules can be represented by a voting tree (Copeland rule can be).



Automated design of Voting Rules by Learning

- Designer:
  - knows the winner of every election, thus, knows the voting rule,
  - does not know how to represent it efficiently.
- Goal:
  - learn the designer's voting rule,
  - find a voting tree.
- Why?
  - Representing a voting rule is EXP(m),
  - voting trees make the procedure easier, more concise, incorporating desired properties (majority, robustness, ...).
- How many examples?

Learning Results (1/2)

Theorem

An exponential training set is needed to learn voting trees.

Restrict to the class of voting trees of polynomial size (at most k leaves).

Lemma

If the size of this class is exponential, the goal is achieved with polynomial training set.

Learning Results (2/2)

Theorem

$|Voting\ trees\ with\ \leq\ k\ leaves| \leq EXP(m,k)$

Proof.

$size \leq (\#\text{possible structures } * \#\text{assignments to leaves}) * k =$
$(\#\text{possible structures } * m^k) * k = ((k-1)!) * m^k * k$ □

Finding a Voting Tree

Definition (TREE-SAT)

- tree with partially labeled leaves,
- a training set

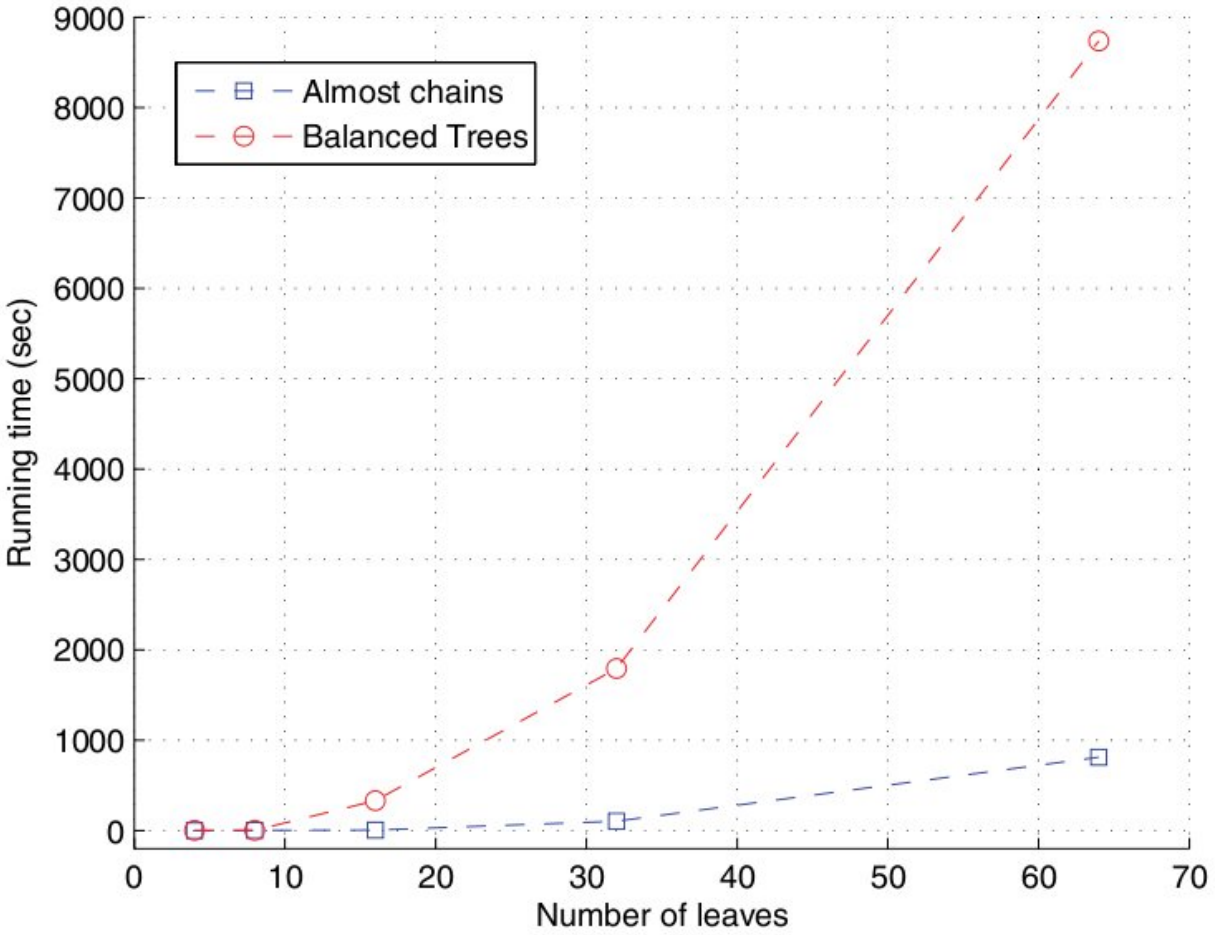
Find a consistent assignment of alternatives to the rest of the leaves.

Theorem

TREE-SAT is NP-complete.

- Imagine the case in which the tree structure was unknown.
- In practice the complexity depends on the structure of the tree.

Experiments



Conclusions

- For those voting rules that can be represented by a tree with polynomial number of leaves the learning process needs only a polynomial number of examples in the number of alternatives.
- Efficiently finding such a tree depends on its structure (almost chain, balanced tree).
- A designer can use this method to translate a cumbersome representation of its voting rule to a concise one which also satisfies desired properties.

Applications

- Compact representation of preferences in combinatorial domains (AI, logic).
- Development of computationally feasible mechanisms, and exploitation of computational intractability as a means against strategic manipulation.

References (1/2)

Ariel D. Procaccia, Aviv Zohar, Yoni Peleg, and Jeffrey S. Rosenschein.
Learning Voting Trees.
In Proceedings of the Twenty-Two AAAI Conference on Artificial Intelligence. July 22-26, 2007.

Ariel D. Procaccia, Aviv Zohar, and Jeffrey S. Rosenschein.
Automated Design of Voting Rules by Learning from Examples.
In Proceedings of the First International Workshop on Computational Social Choice, 436-449. 2006.

References (2/2)

1st International Workshop on Computational Social Choice Amsterdam, 6-8 December 2006.
http://staff.science.uva.nl/~ulle/COMSOC-2006/background.html.

The LaTeX Beamer Class Homepage.
http://latex-beamer.sourceforge.net/.

The End

Thank you!