Fixed Parameter Algorithms for Kemeny Scores

Frances Rosamond
The University of Newcastle, Au

Joint work with
Nadja Betzler, Jiong Guo, and Rolf Niedermeier. Friedrich-Schiller University, Jena, Germany. Michael Fellows, Univ. of Newcastle, AU.

Parameterized Complexity Workshop
University of Newcastle
30 and 31 March, 2010
Motivation
The problem
Parameterized complexity and where to find information
Previous results
Parameterized by the Kemeny Score
Parameterized by the Maximum Kendall-Tau Distance
Parameterized by the Average Kendall-Tau Distance
Summary and Open Problems
The deBorda Experiment

- **UK groups:** de Borda Institute and New Economics Foundation
- **Modified Borda Count:** consensus voting
- **Internet participation:** political science, social science, everyone
- **www.opendemocracy.net/deborda**

Borda Count: give least favoured candidate one point, next best two points, …most favoured candidate m points. Candidate elected may not be anyone’s first choice!

Australia uses preferential voting for almost all elections.
**Ranked ballot methods**

- Ranked ballot methods allow voters to list the candidates in order of preference: first choice, second choice, and so on.

- In Australia, voters are required to rank all of the candidates in order for their ballot to be counted. The effort to reform election abuses led to the widespread use of the Australian ballot, which was adopted in Victoria in 1857, in Great Britain in 1872, and grew increasingly popular in the United States after 1888. The Australian ballot is now used in many European countries and in almost all sections of the United States.

- In US, it gradually replaced earlier methods of voting such as lengthy "tickets" distributed by political parties. In the Australian system all candidates' names are printed on a single ballot and placed in the polling places at public expense, and the printing, distribution, and marking of the ballot are protected by law, thus assuring a secret vote.
### A Complete Online Election in 3 Steps

1. **Create** Name your election: Choose a start date/Choose an election type /Design your ballot: add positions, candidates and questions.

2. **Vote** Unique keys are generated for each voter and sent out by email. Each voter casts their ballot anonymously.

3. **Watch** Watch in real-time as your voters cast their ballots and your election results are tallied. Allow voters to see election results by turning on the 'Public URL' option.

### Three types of elections:
- Preferential Voting (Instant Runoff Voting),
- First Past the Post or Plurality Voting,
- Approval Voting.

---

**LESSON:** People are interested. They want to know about voting.
You Can Vote However You Like
3 min 56 sec - 24 Oct 2008
www.youtube.com

• Homer Simpson tries to vote for Obama
1 min 21 sec - 29 Sep 2008
www.youtube.com

LESSON: A lot of interesting voting is going on.
Can vote over just about anything: political representatives, award nominees, dinner tonight, allocations of tasks/resources, …

- **MTV Movie Awards 2010 Nominees' Voting Now Open!** - 9 hours ago. Well now you can *vote* for your favorite movie moments — starting today (March 29) — for the 2010 MTV Movie Awards, which airs live on June 6 from Los ...

- **WALL STREET JOURNAL, The Australian, 16 Feb.** *Delhi government cites a need to “build consensus” to cultivate of genetically modified eggplant.*

**LESSON:** People are using voting vocabulary “build consensus”.
### Aggregate Paradox

100 voters: Basketball court or a Gym

<table>
<thead>
<tr>
<th>Basketball Court Yes but No gym</th>
<th>Basketball Court No Yes gym</th>
<th>Both Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>49</td>
<td>2</td>
</tr>
</tbody>
</table>
Aggregate Decisions Paradox

100 voters: Basketball court or a Gym

<table>
<thead>
<tr>
<th>Basketball Court</th>
<th>Basketball Court</th>
<th>Both Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes but No gym</td>
<td>No</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Court Yes</th>
<th>Court No</th>
<th>Gym Yes</th>
<th>Gym No</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>49</td>
<td>50</td>
<td>49</td>
</tr>
</tbody>
</table>

Voting separately on each issue gives outcome Yes to both Basketball Court and Gym, even though this option received only 2 votes out of 100.
Manipulation by a voter

- Voter 1: $a > b > c$
- Voter 2: $a > b > c$
- Voter 3: $b > a > c$
- Voter 4: $b > a > c$
- Voter 5: $c > a > b$
Manipulation by a voter

- Voter 1: $a > b > c$
- Voter 2: $a > b > c$
- Voter 3: $b > a > c$
- Voter 4: $b > a > c$
- Voter 5: $c > a > b$

**Voter 5’s true vote has no chance.**

Better for **Voter 5** to vote insincerely and at least get second choice to win.

**Voter 5 should change vote to**

$a > c > b$
Many Election Systems

- Majority
- Condorcet
- Kemeny
- Dodgson
- Young
- Copeland
- Many more

Election
Set of votes $V$, set of candidates $C$.
A vote is a ranking (total order) over all candidates.

Example: $C = \{a, b, c\}$
vote 1: $a > b > c$
vote 2: $a > c > b$
vote 3: $b > c > a$

How to aggregate the votes into a “consensus ranking”? 

What do we want in a consensus?
A ranking “close” to the votes.
Marquis de Condorcet (Marie Jean Antoine Nicolas Caritat). The Condorcet criterion was discovered independently Ramon Llull in 1299.

Condorcet (1785): Find the candidate who wins in a head to head election against every other candidate individually. The election breaks into a series of **pairwise comparisons** between every candidate and every other candidate.
Kendall-Tau distance between two votes

**KT-distance (between two votes \( v \) and \( w \))**

\[
\text{dist}(v, w) = \sum_{\{c,d\} \subseteq C} d_{v,w}(c, d),
\]

where \( d_{v,w}(c, d) \) is 0 if \( v \) and \( w \) rank \( c \) and \( d \) in the same order, else 1.

*Find a distance measure between the input objects:*

**Kendall-Tau distance (KT-dist)** counts the number of pairwise disagreements between two rankings.

“Bubble-sort Metric”

Number of inversions
The Kendall-Tau distance between two votes

<table>
<thead>
<tr>
<th>Votes $V$</th>
<th>Candidates $C$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voter1:</td>
<td>$a &gt; b &gt; c$</td>
</tr>
<tr>
<td>Voter2:</td>
<td>$a &gt; c &gt; b$</td>
</tr>
<tr>
<td>Voter3:</td>
<td>$b &gt; c &gt; a$</td>
</tr>
</tbody>
</table>

The KT-distance between vote 1 and vote 2 is 1.
The KT-distance between vote 1 and vote 3 is 2.

Compare pairwise rankings $ab$, $ac$, $bc$.

<table>
<thead>
<tr>
<th>ab</th>
<th>ac</th>
<th>bc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

 Vote 1&2: $0 + 0 + 1 = 1$

 Vote 1&3: $1 + 1 + 0 = 2$

 Vote 2&3: $1 + 1 + 1 = 3$
Kemeny Score for an order

Votes $V$  
Candidates $C$

Voter1:  $a > b > c$
Voter2:  $a > c > b$
Voter3:  $b > c > a$

<table>
<thead>
<tr>
<th>Order</th>
<th>a &gt; b &gt; c</th>
<th>a &gt; c &gt; b</th>
<th>b &gt; a &gt; c</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ab ac bc</td>
<td>ab ac bc</td>
<td>ab ac bc</td>
</tr>
<tr>
<td>V1:</td>
<td>0 + 0 + 0</td>
<td>0 + 0 + 1</td>
<td>1 + 0 + 0</td>
</tr>
<tr>
<td>V2:</td>
<td>0 + 0 + 1</td>
<td>0 + 0 + 0</td>
<td>1 + 0 + 1</td>
</tr>
<tr>
<td>V3:</td>
<td>1 + 1 + 0</td>
<td>1 + 1 + 1</td>
<td>0 + 1 + 1</td>
</tr>
</tbody>
</table>

Kemeny Score = 3  
Kemeny Score = 4  
Kemeny Score = 5

The **Kemeny Score** for an order is the sum of KT-distances between that order and all votes.
Kemeny Consensus: an order that minimizes the Kemeny Score.

All possible orders for abc: abc, acb, bac, bca, cab, cba

Consider the KT-distances between all votes and all possible orders.

Consensus: a > b > c

Kemeny Score = 3

Voter1: a > b > c
Voter2: a > c > b
Voter3: b > c > a

Kemeny Score = 4

Kemeny Score = 4

Consensus: b > c > a

Kemeny Score = 6

Consensus: c > b > a

Kemeny Score = 5

Kemeny Consensus: an order that minimizes the Kemeny Score.

All possible orders for abc: abc, acb, bac, bca, cab, cba

Consider the KT-distances between all votes and all possible orders.
Kemeny Consensus: order that minimizes the score

Let a and b be two candidates. If a > b in all votes, then every Kemeny consensus has a > b.
Kemeny Winner: In our example, candidate $a$ is Kemeny winner.
John Kemény was born in Budapest, escaped with parents, attended Princeton but left to work at Los Alamos (with Richard Feynman and John von Neumann) during the war. He returned to Princeton, graduated with his BA in 1947, then worked for his doctorate under Alonzo Church. He worked as Einstein’s mathematical assistant during graduate school. Kemeny doctorate in 1949, dissertation entitled "Type-Theory vs. Set-Theory."

Kemeny became President of Dartmouth, which became coed (after 203 years of single-sex education). While president, he taught two courses a year, never missing a class.

He, with Thomas Kurtz, invented BASIC. Kemeny made Dartmouth a pioneer in student use of computers, equating computer literacy with reading literacy.

American colleges and universities experienced a tumultuous period of student protest, Dartmouth enjoyed relative calm due in large part to Kemeny's appeal to students and his practice of seeking consensus on vital college issues.

He invented the Kemeny Election scheme around 1958.
Decision Problem

**KEMENY SCORE**

*Input*: An election \((V, C)\) and a positive integer \(k\).

*Parameter*: \(k\).

*Question*: Is the Kemeny Score of \((V, C)\) at most \(k\)?

**KEMENY WINNER**

*Input*: An election \((V, C)\) a distinguished candidate \(c\).

*Question*: Is there a Kemeny consensus in which \(c\) wins?

Looking for a consensus list that minimizes the sum of the distances to the given votes.
The corresponding complexity class is called FPT. For example, there is an algorithm which solves the vertex cover problem in $O(kn + 1.274^k)$ time, so the corresponding complexity class is called FPT.

Parameterized complexity is a new and promising approach to the central issue of how to cope with other costs being polynomial (called FPT complexity). Parameterized complexity classes and fpt-reduction we refer the reader to the FPT hierarchy are distinct from FPT.

Denote by FPT the class of all fixed parameter tractable problems. First note that by our assumption on the parameter $k$, the problem is in FPT. Parameterized Complexity - an Overview

Of course, to obtain such a complexity, the choice of the parameter is crucial. The general point of view, proposed in the last years, is that such complexity classes can be much better than the class of all polynomial time algorithms (called P complexity).

Lists of related classes: Communication Complexity - Hierarchies - Nonuniform...a third support for parameterized complexity theory...
Meta-Search Engine

How to make an aggregate ranking—“consensus”?
Voters = Internet search engines. Candidates = webpages

Search engines: few voters
Web pages: Huge number of “candidates”

Rank aggregation methods for the web
C. Dwork, R. Kumar, M. Naor, D. Sivakumar
Voters = Internet search engines. Candidates = webpages

Search engines: few voters
Web pages: Huge number of “candidates”

Method of Kemeny minimizes the total disagreement between several input rankings and their aggregation. Unfortunately, computing optimal solutions based on Kemeny is NP-hard, even for only 4 rankings.
Known results

- **Kemeny Score** is NP-complete (even for 4 votes)
  [Dwork & al. WWW 2001]

- **Kemeny Winner** is $P^{NP}_{\parallel}$-complete
  [Hemaspaandra & al. TCS 2005]

Algorithms:

- randomized factor 11/7-approximation [Ailon & al. STOC 2005]
- factor 8/5-approximation [van Zuylen & Williamson WAOA 2007]
- PTAS [Kenyon-Mathieu & Schudy STOC 2007]
- greedy, branch and bound
  [Conitzer & al. AAAI 2006], [Davenport & Kalagnanam AAAI 2004]
Import ideas from CS to solve questions from social choice

Can we design a voting protocol that makes it impossible for a voter to cheat?

Maybe not, but the problem may be intractable, and therefore an acceptable risk.

OK, we can take the risk

Computer Science: Design and analysis of algorithms
These results


- New results:
  - N. Simjour. Improved parameterized algorithms for the Kemeny Aggregation problem. In Proc. 4\textsuperscript{th} IWPEC, v 5917 of LNCS, pp 312-323, 2009.)
Parameterized Complexity

Given a NP-hard problem with input size $n$ and a parameter $k$

**Basic idea:** Confine the combinatorial explosion to $k$

- $O(n^k)$ instead of
- $O(f(k) \cdot n^{O(1)})$

**Definition**

A problem of size $n$ is called *fixed parameter tractable* with respect to a parameter $k$ if it can be solved in $f(k) \cdot n^{O(1)}$ time.

Or additively, in $O(f(k) + n^c)$ time.
Parameterized Complexity

- Database query size tends to be much smaller than the size of the entire database.
- The number of candidates in an election may be much smaller than the number of voters.
- Number of web pages is large compared to the number of search engines.
- Parameters recognize different size magnitudes.

\[ O(f(k)n^c) \]
\[ O(f(k) + n^c) \]
Parameterizations of KEMENY SCORE

1. **Parameter “the Kemeny Score”**. Solvable in $O(1.53^k + m^2 n)$ time, where $k$ denotes the Kemeny Score of the given election.

2. **Parameter “the maximum KT-distance between any two input votes”**. Solvable in $O((3d + 1)! d \left(\log d\right) m n)$ time.

3. **Parameter “the number of candidates”**. Solvable in $O(2^m m^2 n)$ time.

4. **Parameter “the maximum range of candidate positions”**. Solvable in $O^*(32^r)$ time.

5. **Parameter “the average KT-distance”**. Solvable in $O^*(16^{\text{aveKT}})$ time.
Downey-Fellows: Parameterized Complexity, Springer, 1999


Rodney G. Downey, Michael R. Fellows, and Michael A. Langston

Foreword by the Guest Editors
Falk Hüffner, Rolf Niedermeier, and Sebastian Wernicke

Techniques for Practical Fixed-Parameter Algorithms
Michael A. Langston, Andy D. Perkins, Arnold M. Saxton, Jon A. Scharff, and Brynn H. Voy

Innovative Computational Methods for Transcriptomic Data Analysis: A Case Study in the Use of FPT for Practical Algorithm Design and Implementation
Jianer Chen and Jie Meng

On Parameterized Intractability: Hardness and Completeness
Dániel Marx

Parameterized Complexity and Approximation Algorithms
Jens Gramm, Arfst Nickelsen, and Till Tantau

Fixed-Parameter Algorithms in Phylogenetics
Leizhen Cai

Parameterized Complexity of Cardinality Constrained Optimization Problems
Christian Sloper and Jan Arne Telle

An Overview of Techniques for Designing Parameterized Algorithms

Book review
William Gasarch and Keung Ma Kin

Invitation to Fixed-Parameter Algorithms • Parameterized Complexity Theory • Parameterized Algorithmics: Theory, Practice and Prospects
Hans L. Bodlaender and Arie M. C. A. Koster
Combinatorial Optimization on Graphs of Bounded Tree-width
Liming Cai, Xiuzhen Huang, Chunmei Liu, Frances Rosamond, and Yinglei Song
Parameterized Complexity and Biopolymer Sequence Comparison
Erik D. Demaine and MohammadTaghi Hajiaghayi
The Bidimensionality Theory and Its Algorithmic Applications
Georg Gottlob and Stefan Szeider
Fixed-Parameter Algorithms For Artificial Intelligence, Constraint Satisfaction and Database Problems
Petr Hlineny, Sang-il Oum, Detlef Seese, and Georg Gottlob
Width Parameters Beyond Tree-width and their Applications
Gregory Gutin and Anders Yeo
Some Parameterized Problems On Digraphs
Panos Giannopoulous, Christian Knauer, and Sue Whitesides
Parameterized Complexity of Geometric Problems
Iris van Rooij and Todd Wareham
Parameterized Complexity in Cognitive Modeling: Foundations, Applications and Opportunities
FPT NEWS, THE PARAMETERIZED COMPLEXITY NEWSLETTER

email: Frances.Rosamond@newcastle.edu.au if you would like to receive the Newsletter.

The Parameterized Complexity WIKI is located at

http://www.fpt.wikidot.com
<table>
<thead>
<tr>
<th>Problem</th>
<th>$f(k)$</th>
<th>kernel</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex Cover</td>
<td>$1.2738^k$</td>
<td>$2k$</td>
<td>1</td>
</tr>
<tr>
<td>Feedback Vertex Set</td>
<td>$5^k$</td>
<td>$k^3$</td>
<td>2</td>
</tr>
<tr>
<td>Planar DS</td>
<td>$2^{15.13\sqrt{k}}$</td>
<td>$67k$</td>
<td>3</td>
</tr>
<tr>
<td>1-Sided Crossing Min</td>
<td>$1.4656^k$</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Max Leaf</td>
<td>$6.75^k$</td>
<td>$4k$</td>
<td>5</td>
</tr>
<tr>
<td>Directed Max Leaf</td>
<td>$2^{O(k \log k)}$</td>
<td>?</td>
<td>6</td>
</tr>
<tr>
<td>Set Splitting</td>
<td>$2^k$</td>
<td>$2k$</td>
<td>7</td>
</tr>
<tr>
<td>Nonblocker</td>
<td>$2.5154^k$</td>
<td>$5k/3$</td>
<td>8</td>
</tr>
<tr>
<td>3-D Matching</td>
<td>$2.77^{3k}$</td>
<td></td>
<td>9</td>
</tr>
<tr>
<td>Edge Dominating Set</td>
<td>$2.4181^k$</td>
<td>$8k^2$</td>
<td>10</td>
</tr>
<tr>
<td>k-Path*</td>
<td>$4^k$</td>
<td>no $k^{O(1)}$</td>
<td>11</td>
</tr>
<tr>
<td>Convex Recolouring</td>
<td>$4^k$</td>
<td>$O(k^2)$</td>
<td>12</td>
</tr>
<tr>
<td>VC-max degree 3</td>
<td>$1.1899^k$</td>
<td></td>
<td>13</td>
</tr>
<tr>
<td>Clique Cover</td>
<td>$2^{\Delta k}$</td>
<td>$2^k$</td>
<td>14</td>
</tr>
<tr>
<td>Clique Partition</td>
<td>$2^k$</td>
<td>$2^k$</td>
<td>15</td>
</tr>
<tr>
<td>Cluster Editing</td>
<td>$1.83^k$</td>
<td>$4k$</td>
<td>16</td>
</tr>
<tr>
<td>Steiner Tree</td>
<td>$2^k$</td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>3-Hitting Set</td>
<td>$2.076^k$</td>
<td>$O(k^2)$</td>
<td>18</td>
</tr>
<tr>
<td>Minimum Fill/Interval Completion</td>
<td>$O(k^{2k}n^{3m})$</td>
<td></td>
<td>19</td>
</tr>
</tbody>
</table>
Plan: Find reduction rules (preprocessing rules), and search tree bounded by f(k). Algorithms often interleave reductions and branching.

1) Kemeny score is FPT parameterized by the score

Concept of Dirtyness: Dirty Pair

Dirty pair

Two candidates $a$ and $b$ form a *dirty pair* if in $V$ there is one vote with $a > b$ and another vote with $b > a$.

Example:

- vote 1: $a > b > c > d$
- vote 2: $a > d > b > c$
- vote 3: $b > c > a > d$

In optimal consensus: “$a > d$” and “$b > c$”
dirty pairs: $a,b$ and $a,c$ and $b,d$ and $c,d$
Rule 1: Delete all candidates not in a “dirty pair”.

Parameter: Kemeny score

After Reduction Rule: The Election has at most $2k$ candidates. If there are more than $2k$ candidates after Rule 1, then NO.

\[
\begin{align*}
v1 : a &> b \\
v2 : b &> a \\
v3 : & \quad \quad b > c \\
v4 : & \quad \quad c > b \\
v5 : & \quad \quad c > d \\
v3 : & \quad \quad d > c \\
. & \quad \quad . \quad \quad . \\
. & \quad \quad . \quad \quad . \\
v_n : & \quad \quad .
\end{align*}
\]

In an optimal consensus, every dirty pair contributes at least one to the score. A Kemeny score of “$k$” means there can be at most only $k$ dirty pairs; i.e., $2k$ candidates.
Rule 2: If there are more than $2k$ votes identical to a consensus list $\ell$, return YES, otherwise, return NO.

Parameter: Kemeny score

The score of $\ell$ is at most $k$

$v1 : a > b \ldots$
$v2 : b > a \ldots$
$v3 : \ldots$ $b > c \ldots$
$v4 : \ldots$ $c > b \ldots$
$v5 : \ldots$ $c > d \ldots$
$v3 : \ldots$ $d > c \ldots$
$\vdots$ $\vdots$ $\vdots$
$\vdots$ $\vdots$ $\vdots$
$vn : \ldots$

$k$ dirty pairs, $(2k$ votes$)$

$k$ votes identical to the consensus list $\ell$
Exhaustively apply Rules 1 and 2.

Parameter: Kemeny score

Rule 1: Delete all candidates not in a “dirty pair”.
Rule 2: If there are more than $2k$ votes identical to a consensus list $l$, return YES if the score of $l$ is at most $k$; otherwise, return NO.

A YES instance of KEMENY SCORE has at most $2k$ candidates and $2k$ votes.

The Rules can be computed in $O(m^2 n)$ time.
A dirty pair “increases” the Kemeny score at least by 1.

At each search tree node, branch into the two possible relative orders of the dirty pair, and in each case decrease the parameter by one.

\[(S, k)\]

\[a < b\]

\[a > b\]

\[(S - (a<b), k - 1)\]

\[(S - (a>b), k - 1)\]

Branching vector \((1,1)\)
Gives branching number: 2
i.e., \(2^k\) run time

Branching into dirty pairs results in a search tree of size \(O(2^k)\).

All non flip candidates have their relative order with respect to all other candidates already fixed.
Improvement by branching into *Dirty Triples*

Three candidates form a *dirty triple* if they occur in at least two dirty pairs.

Example: \{a, b, c\} form a dirty triple for these 3 votes.

**Case 1:** dirty triple with 3 dirty pairs

- vote 1: $a > b > c$
- vote 2: $a > c > b$
- vote 3: $b > c > a$

![Branching example](example.png)

branching vector: $(3, 4, 4, 5, 5, 6) \Rightarrow$ branching number: 1.52

**Case 2:** dirty triple with 2 dirty pairs

branching vector: $(3, 3, 2) \Rightarrow$ branching number: 1.53

**Case 3:** no more dirty triples

for every dirty pair: make a “majority decision”
(and decrease the parameter accordingly)
KEMENY SCORE can be solved in $O(1.53^k + n^2)$ time, where $k$ is the Kemeny Score.

Outline of the Dynamic Programming Algorithm

• The search tree algorithm enumerates all dirty pairs and then branches according to the dirty triples.

• At a search tree node, in each case of the branching, an order of the candidates involved in the dirty triples processed at this node is fixed and maintained in a set.

• This order represents the relative positions of these candidates in the Kemeny Consensus sought. The parameter is decreased according to this order.

• Since every order of two candidates in a dirty pair decreases the parameter by at least one, the height of the search tree is upper-bounded by $k$.

• At each node of the search tree we exhaustively apply the data reduction rules (interleave kernelization and search trees).
2) Parameter: “the maximum KT-distance between any two input votes”

| Voter1:   | a > b > c |
| Voter2:   | a > c > b |
| Voter3:   | c > b > a |

<table>
<thead>
<tr>
<th>Input Votes</th>
<th>Candidate pairs</th>
<th>KT-distance between voters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ab</td>
<td>ac</td>
</tr>
<tr>
<td>Vote1&amp;2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Vote2&amp;3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Vote1&amp;3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ d = \text{Maximum KT-distance between any two votes} \text{ is 3.} \]
**Lemma.** In an input instance with maximum KT-distance $d$, the *positions* of a candidate in two votes differ by at most $d$.

<table>
<thead>
<tr>
<th>$v_1$</th>
<th>$v_2$</th>
<th>$v_3$</th>
<th>$v_4$</th>
<th>$v_5$</th>
<th>$\vdots$</th>
<th>$v_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>position</th>
<th>$p$</th>
<th>$d$</th>
<th>$p + d + 1$</th>
</tr>
</thead>
</table>
Lemma. In an input instance with maximum KT-distance $d$, every block of width $d + 1$ contains at most $3d + 1$ candidates.
**Lemma.** In an input instance with maximum KT-distance $d$, every block of width $d + 1$ contains at most $3d + 1$ candidates.

<table>
<thead>
<tr>
<th>$v_1$</th>
<th>$v_2$</th>
<th>$v_3$</th>
<th>$v_4$</th>
<th>$v_5$</th>
<th>$v_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| position | $p$ | $d$ | $p + d + 1$ |
**Lemma.** In an input instance with maximum KT-distance $d$, every block of width $d + 1$ contains at most $3d + 1$ candidates.

<table>
<thead>
<tr>
<th>$v_1$</th>
<th>$v_2$</th>
<th>$v_3$</th>
<th>$v_4$</th>
<th>$v_5$</th>
<th>\ldots</th>
<th>$v_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$b$</td>
<td>$c$</td>
<td>$d$</td>
<td>$e$</td>
<td>$f$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>position</th>
<th>$p$</th>
<th>$d$</th>
<th>$p + d + 1$</th>
</tr>
</thead>
</table>
**Lemma.** In an input instance with maximum KT-distance $d$, every block of width $d + 1$ contains at most $3d + 1$ candidates.
KEMENY SCORE can be solved in $O((3d + 1)! \cdot d \log d \cdot n)$ time with $d$ the maximum KT-distance between two input votes.

Outline of Dynamic Programming algorithm

- Range of each candidate is bounded by a block of size $d$.
- Block size is bounded by $3d + 1$.

Algorithm:

- Iterate left to right over the blocks of votes.
- Try all possible orders of the candidates within a block.
- There are $(3d + 1)!$ possible orders of a block.
3) Parameter "the average KT-distance"

**Definition**

For an election \((V, C)\) the average KT-distance \(d_a\) is defined as

\[
d_a := \frac{1}{n(n-1)} \cdot \sum_{\{u,v\} \in V, u \neq v} \text{KT-dist}(u, v).
\]

Sum over all pairwise distances divided by the number of pairs.
Average range of candidate  

Parameter: Average KT-distance

Vote:

<table>
<thead>
<tr>
<th>v_1</th>
<th>v_2</th>
<th>v_3</th>
<th>v_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
</tr>
<tr>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
</tr>
<tr>
<td>c</td>
<td>a</td>
<td>b</td>
<td>f</td>
</tr>
<tr>
<td>c</td>
<td>a</td>
<td>b</td>
<td>f</td>
</tr>
</tbody>
</table>

ave pos(a): 0 + 5 + 1 + 1 = 7/4
ave pos(b): 1 + 0 + 2 + 1 = 4/4
ave pos(c): 2 + 1 + 0 + 0 = 3/4
ave pos(d): 3 + 2 + 5 + 5 = 15/4
ave pos(e): 4 + 3 + 4 + 4 = 11/4
ave pos(f): 5 + 4 + 3 + 3 = 15/4

Average range: 55/4

In every Kemeny consensus, every candidate can only assume a number of consecutive positions bounded by da.
Lesson: Check the size of the parameter, then use the appropriate strategy for your problem.

Example 1: small range, large number of candidates and average distance

\[ a > c > b > e > d > f \ldots \]
\[ b > a > c > d > e > f \ldots \]
\[ b > e > a > c > f > d \ldots \]

Example 2: small average distance, large number of candidates and range

\[ a > b > c > d > e > f \ldots \]
\[ b > c > d > e > f > a \ldots \]
\[ a > b > c > d > e > f \ldots \]

- Number of candidates \( m (O^*(2^m)) \)
- Maximum range \( r \) of candidate positions in the input votes \( (O^*(32^r)) \)
- Average distance of the input votes \( (O^*(16^{da})) \)
Let the average position of a candidate $c$ be $p_a(c)$.

**Lemma**

Let $d_a$ be the average KT-distance of an election $(V, C)$. Then, in every optimal Kemeny consensus $l$, for every candidate $c \in C$ we have $p_a(c) - d_a < l(c) < p_a(c) + d_a$. 
Bounded number of candidates per position

Parameter: average KT-distance

For a position $i$, let $P_i$ denote the set of candidates that can assume $i$ in an optimal consensus.

Lemma

Let $d_a$ be the average KT-distance of an election $(V, C)$. For a position $i$, we have $|P_i| \leq 4 \cdot d_a$.

Proof: Position “range” of every candidate is at most $2 \cdot d_a$.

```
consensus
  1  i - 2d_a  i  i + 2d_a  m
```

Every candidate of $P_i$ must have a position smaller than $i + 2d_a$ and greater than $i - 2d_a$. 
Position $i$, a candidate $c \in P_i$, a subset of candidates $P'_i \subseteq P_i \setminus \{c\}$

**Definition**

$T(i, c, P'_i) :=$ optimal partial Kemeny score if $c$ has position $i$ and all candidates of $P'_i$ have positions smaller than $i$

$P_i = \{a, b, c, d, e, f\}$

<table>
<thead>
<tr>
<th>consensus</th>
<th>{a,b}</th>
<th>c</th>
<th>{d,e,f}</th>
</tr>
</thead>
</table>

$P'_i = \{a, b\}$

Computation of partial Kemeny scores:

- Overall Kemeny score can be decomposed (just a sum over all votes and pairs of candidates)
- Relative orders between $c$ and all other candidates are already fixed
**KEMENY SCORE** can be solved in $O^*(16^{d_a})$ time, with $d_a$ the average KT-distance.

Outline of the Dynamic Programming algorithm

- $T(I, c, P'_i)$ is defined an *optimum partial Kemeny Score* if ‘c’ has position ‘i’ and all other candidates in $P_i$ have positions smaller than i.

- Computation of partial Kemeny Score: Overall KS can be decomposed. Relative orders between c and all other candidates are already fixed.

$n$ votes
$m$ candidates

$$P_i = \{a, b, c, d, e, f\}$$

consensus

<table>
<thead>
<tr>
<th>{a, b}</th>
<th>c</th>
<th>{d, e, f}</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We have $|P_i| \leq 4d_a$, thus there are at most $2^{4d_a}$ subsets of $P_i$.

$\Rightarrow$ Table size is bounded by $16^{d_a} \cdot \text{poly}(n, m)$. 
1. Parameter “the Kemeny Score”. Solvable in $O(1.53^k + m^2n)$ time, where $k$ denotes the Kemeny Score of the given election.

2. Parameter “the maximum KT-distance between any two input votes”. Solvable in $O((3d + 1)! \cdot d \cdot (\log d) \cdot mn)$ time.

3. Parameter “the number of candidates”. Solvable in $O(2^m \cdot m^2n)$ time.

4. Parameter “the maximum range of candidate positions”. Solvable in $O^*(32^r)$ time.

5. Parameter “the average KT-distance”. Solvable in $O^*(16^{\text{aveKT}})$ time.

Work done by Nadja Betzler, Jiong Guo, Rolf Niedermeier. Friedrich-Schiller University, Jena, Germany. Michael Fellows, Frances Rosamond. Newcastle, AU.
## KEMENY SCORE Ties & Incomplete Votes

<table>
<thead>
<tr>
<th>Kemeny score $k$</th>
<th>Kernelization</th>
<th>with Ties</th>
<th>Incomplete Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O^*(1.53^k)$</td>
<td>$O^*(1.76^k)$</td>
<td>$O^*((1.48k)^k)$</td>
<td></td>
</tr>
<tr>
<td>$O^*((3d + 1)!)$</td>
<td>$O^*((6d + 2)!)$</td>
<td>NP-h for $d = 0$</td>
<td></td>
</tr>
<tr>
<td>$O^*(2^m)$</td>
<td>$O^*(2^m)$</td>
<td>$O^*(2^m)$</td>
<td></td>
</tr>
<tr>
<td>$n$</td>
<td>NP-h for $n = 4$</td>
<td>NP-h for $n = 4$</td>
<td></td>
</tr>
</tbody>
</table>

# candidates $m$  
# votes $n$
Open Problems and Future Directions

Notice that the order of the websites given by various search engines (Google, Yahoo, Bai) is almost the same. This implies a small average distance between the sites.

Average distance between points is small. There may be a few outliers. In Geometry called a “one-center” problem. There could be a 2-center problem.

LESSON: Relate the number of dirty elements (involved in dirty pairs) with average distance. Find reduction rules. Bound the number of dirty elements by a function of average distance in reduced instances.
Open Problems and Future Directions

• Investigate typical values for average KT-distance.
• Investigate “similar on average” in other voting systems/settings.
• Consider aggregate parameters.
• The majority of kernelization technique/development has been done with graph problems. Continue to find techniques/tools for consensus type problems. (B, G, K, N: a “partial kernelization”).
• There is a branching algorithm for KEMENY SCORE which runs in $O((5.823)^d \text{poly}(n, m))$ time. (N. Simjour. Improved parameterized algorithms for the Kemeny Aggregation problem. In Proc. 4$^{th}$ IWPEC, v 5917 of LNCS, pp 312-323, 2009.) How can branching continued to be improved?
• The “maximum distance of the input rankings from the solution” is an upper bound for the average distance. In contrast, “the average distance of the input rankings from the solution” is a lower bound for the “average distance between the input rankings”. Open to investigate the parameterized complexity with respect to this parameter.
Thank you