Continuous Computational Social Choice

Martin Koutecký

November 6th, 2025 AGATE Kick-off

CHARLES UNIVERSITY
Faculty of mathematics
and physics

CSI
IÚUK

GAČR
CZECH SCIENCE FOUNDATION

Continuous Computational Social Choice

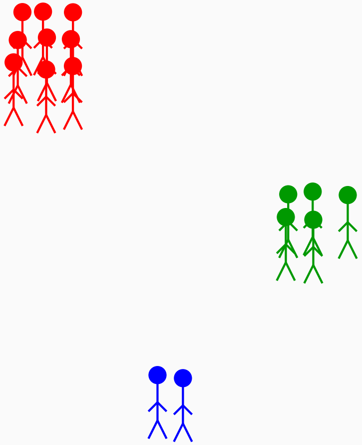
- **Social choice:** voting, matching, allocations, ...

Continuous Computational Social Choice

- **Social choice:** voting, matching, allocations, ...
- **Computational** SoC: algorithms and hardness

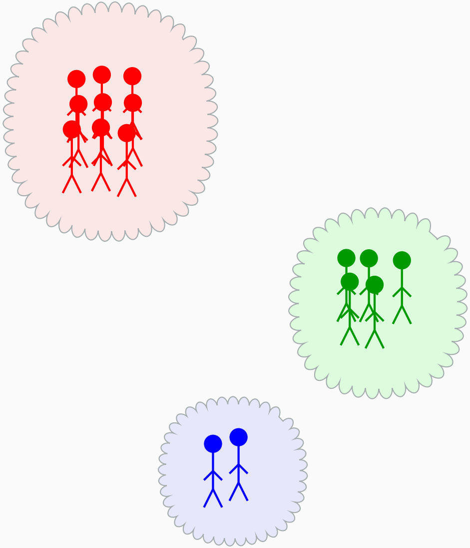
Continuous Computational Social Choice

- **Social choice:** voting, matching, allocations, ...
- **Computational** SoC: algorithms and hardness
- **Traditionally:** discrete agents ⇝ hard



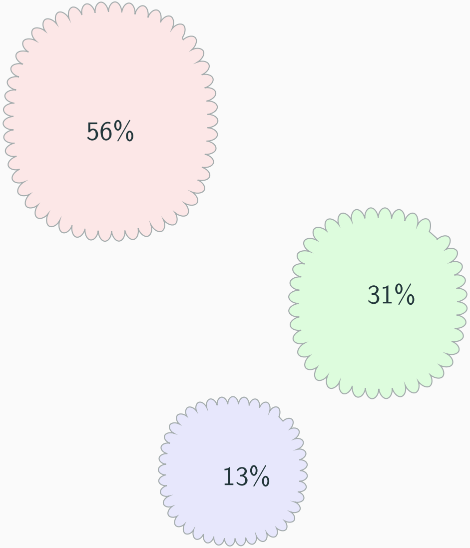
Continuous Computational Social Choice

- **Social choice:** voting, matching, allocations, ...
- **Computational** SoC: algorithms and hardness
- **Traditionally:** discrete agents $\rightsquigarrow$ hard
- **New perspective:**
  - Focus on agent *types*



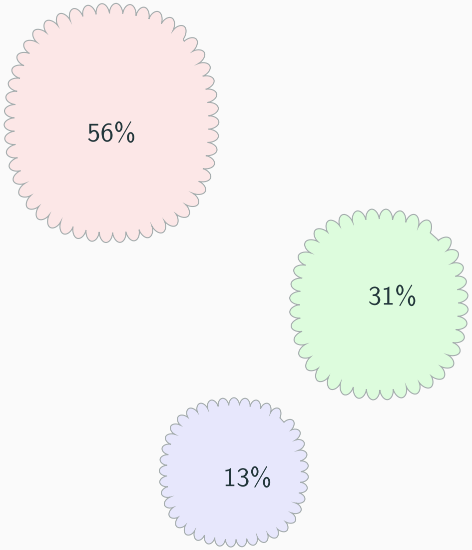
Continuous Computational Social Choice

- **Social choice:** voting, matching, allocations, ...
- **Computational** SoC: algorithms and hardness
- **Traditionally:** discrete agents ⇝ hard
- **New perspective:**
  - Focus on agent *types*
  - Forget individuals ⇝ continuous quantities ⇝ efficient (?)



Continuous Computational Social Choice

- **Social choice:** voting, matching, allocations, …
- **Computational** SoC: algorithms and hardness
- **Traditionally:** discrete agents $\rightsquigarrow$ hard
- **New perspective:**
  - Focus on agent *types*
  - Forget individuals $\rightsquigarrow$ continuous quantities $\rightsquigarrow$ efficient (?)
- **Analogous to:** Mean-field Theory
  - Statistical Physics
  - Mean-field Game Theory
  - “Geometry of Voting”



Case Study: Voting & Bribery

Candidates: $\blacktriangle$, $\blacksquare$, and $\bigstar$.

People: preference (e.g. $\blacksquare \succ \blacktriangle \succ \bigstar$)

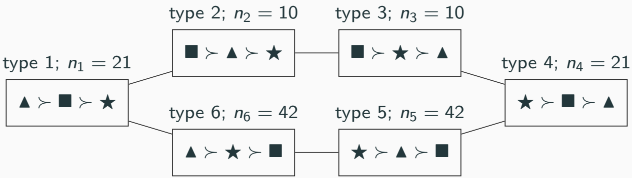
Society: how many people of which type $\Rightarrow$ **Society Graph:**

Case Study: Voting & Bribery

Candidates: ▲, ■, and ★.

People: preference (e.g. ■ $\succ$ ▲ $\succ$ ★)

Society: how many people of which type $\Rightarrow$ **Society Graph:**



Edges $\equiv$ swap distance 1.

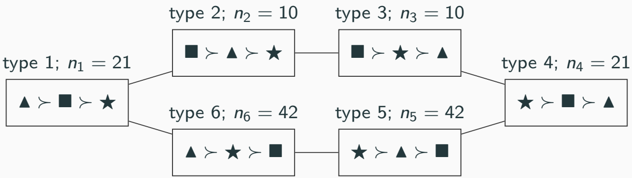
Society $\mathbf{n} = (21, 10, 10, 21, 42, 42)$

Case Study: Voting & Bribery

Candidates: ▲, ■, and ★.

People: preference (e.g. ■ $\succ$ ▲ $\succ$ ★)

Society: how many people of which type $\Rightarrow$ **Society Graph:**



Edges $\equiv$ swap distance 1.

Society $\mathbf{n} = (21, 10, 10, 21, 42, 42)$

Voting rule: given a society, *who should win?*

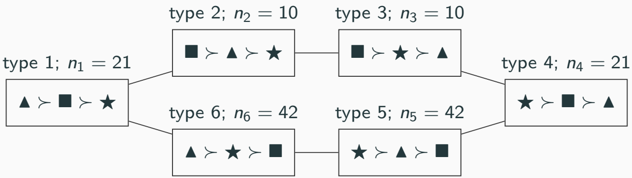
- Plurality = most times first
- Condorcet = beats everyone head-to-head
- Many others (Borda, Kemeny, Dodgson, Approval, STV)

Case Study: Voting & Bribery

Candidates: ▲, ■, and ★.

People: preference (e.g. $\blacksquare \succ \blacktriangle \succ \bigstar$)

Society: how many people of which type ⇒ **Society Graph:**



Edges ≡ swap distance 1.

Society $\mathbf{n} = (21, 10, 10, 21, 42, 42)$

Continuous Society:

$$\mu = \frac{\mathbf{n}}{\|\mathbf{n}\|_1} \approx (.14, .07, .07, .14, .29, .29)$$

Voting rule: given a society, *who should win?*

- Plurality = most times first
- Condorcet = beats everyone head-to-head
- Many others (Borda, Kemeny, Dodgson, Approval, STV)

Young Voting

- Condorcet-consistent rules $\approx$ "candidate *closest* to being Condorcet should win".

Young Voting

- Condorcet-consistent rules $\approx$ "candidate *closest* to being Condorcet should win".
- **Young Score:** #voter deletions to become Condorcet

Young Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Young Score:** #voter deletions to become Condorcet
- c with smallest YS is Young Winner.

Young Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Young Score:** #voter deletions to become Condorcet
- c with smallest YS is Young Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete

Young Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Young Score:** #voter deletions to become Condorcet
- c with smallest YS is Young Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete

n_i = #voters of type i;

x_i = #deleted voters of type i

$$\min \sum_i x_i$$

$$0 \leq x_i \leq n_i \qquad i \in [\tau]$$

$$\sum_{i: c^\star >_i c'} (n_i - x_i) > \sum_{i: c' >_i c^\star} (n_i - x_i) \quad \forall c' \neq c^\star$$

$$\mathbf{x} \in \mathbb{N}^\tau$$

Young Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Young Score:** #voter deletions to become Condorcet
- c with smallest YS is Young Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Polytime! (Linear Programming)

n_i = #voters of type i;

x_i = #deleted voters of type i

$$\min \sum_i x_i$$

$$0 \leq x_i \leq n_i \qquad i \in [\tau]$$

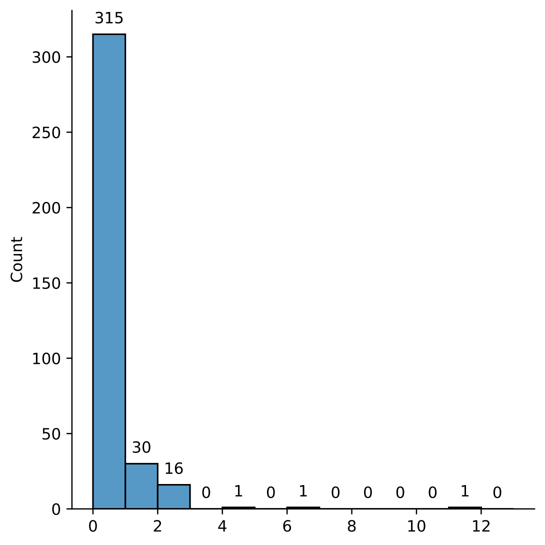
$$\sum_{i: c^\star >_i c'} (n_i - x_i) > \sum_{i: c' >_i c^\star} (n_i - x_i) \quad \forall c' \neq c^\star$$

$$\mathbf{x} \in \mathbb{R}^\tau$$

Young Voting: Preflib (Political Elections)

On political elections of PrefLib ($n = 364$):

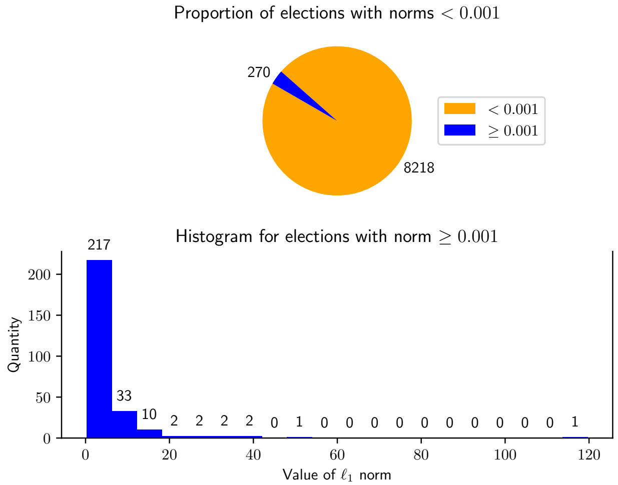
- Young Score vs Young Score$_\infty$ **always** give the same ranking
- On $n = 315$ elections both scores **agree completely**
- On remaining 49 elections never differ by more than 12, or 0.14% in relative terms.



Young Voting: Preflib (All Elections)

On all elections of PrefLib ($n = 8482$ elections):

- Young Score vs Young Score$_\infty$ give the same ranking on 97% instances
- On remaining elections does not differ much



Dodgson Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Dodgson Score:** #adjacent swaps to become Condorcet
- c with smallest DS is Dodgson Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete

Dodgson Voting

- Condorcet-consistent rules $\approx$ "candidate *closest* to being Condorcet should win".
- **Dodgson Score:** #adjacent swaps to become Condorcet
- c with smallest DS is Dodgson Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete

n_i = #voters of type i;

x_{ij} = #voters of type i with j shifts up of $\star$

$$\min \sum_i \sum_j j \cdot x_{ij}$$

$$\sum_j x_{ij} = n_i \qquad i \in [\tau]$$

$$\sum_{t: c^\star >_t c'} y_t > \sum_{t: c' >_i c^\star} y_t \qquad \forall c' \neq c^\star$$

$$x_{ij} \in \mathbb{N}$$

Dodgson Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Dodgson Score:** #adjacent swaps to become Condorcet
- c with smallest DS is Dodgson Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Polytime! (LP)

n_i = #voters of type i;

x_{ij} = #voters of type i with j shifts up of $\star$

$$\min \sum_i \sum_j j \cdot x_{ij}$$

$$\sum_j x_{ij} = n_i \qquad i \in [\tau]$$

$$\sum_{t: c^\star >_t c'} y_t > \sum_{t: c' >_i c^\star} y_t \qquad \forall c' \neq c^\star$$

$$x_{ij} \in \mathbb{R}_{\geq 0}$$

Dodgson Voting

- Condorcet-consistent rules ≈ "candidate *closest* to being Condorcet should win".
- **Dodgson Score:** #adjacent swaps to become Condorcet
- c with smallest DS is Dodgson Winner.
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Polytime! (LP)
- We're lucky: shifts up suffice, o/w $\Theta(m!)$ "output types" to consider

n_i = #voters of type i;

x_{ij} = #voters of type i with j shifts up of $\star$

$$\min \sum_i \sum_j j \cdot x_{ij}$$

$$\sum_j x_{ij} = n_i \qquad i \in [\tau]$$

$$\sum_{t: c^\star >_t c'} y_t > \sum_{t: c' >_i c^\star} y_t \qquad \forall c' \neq c^\star$$

$$x_{ij} \in \mathbb{R}_{\geq 0}$$

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance
  average of all voters

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.
- c is **Kemeny winner** if top of some KR

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.
- c is **Kemeny winner** if top of some KR
- Harder than NP-complete: $P^{NP}_{||}$-complete

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.
- c is **Kemeny winner** if top of some KR
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Still hard!

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.
- c is **Kemeny winner** if top of some KR
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Still hard!

Proof Sketch:

- Each voter type $\equiv$ ranking $+$ weight

Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.
- c is **Kemeny winner** if top of some KR
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Still hard!

Proof Sketch:

- Each voter type $\equiv$ ranking $+$ weight
- Kemeny Ranking is weighted average of voter types

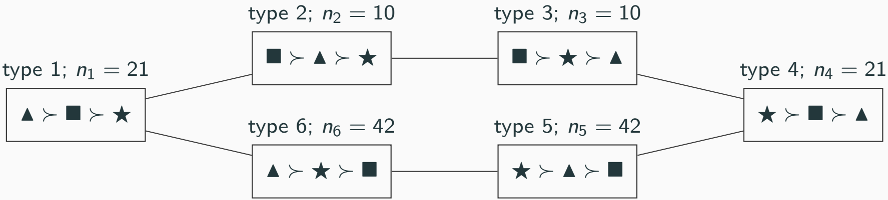
Kemeny Voting

- **Kemeny ranking** $\approx$ swap-distance **average** of all voters
- **Specifically:** ranking $\succ$ minimizing total swap distance from all voters.
- c is **Kemeny winner** if top of some KR
- Harder than NP-complete: $P_{||}^{NP}$-complete
- **Society Continuum:** Still hard!

Proof Sketch:

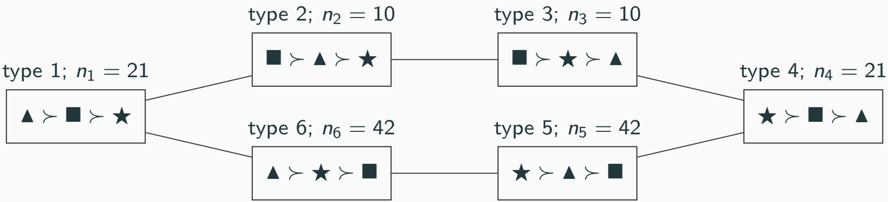
- Each voter type $\equiv$ ranking $+$ weight
- Kemeny Ranking is weighted average of voter types
- : Down-scaling weights by a scalar doesn't change the average!

Bribing



$$\textit{society } \mathbf{n} = (21, 10, 10, 21, 42, 42)$$

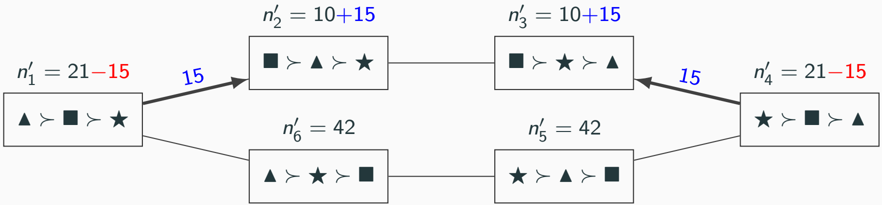
Bribing



society $\mathbf{n} = (21, 10, 10, 21, 42, 42)$

“Bribery:” cheapest way to move voters s.t. ■ wins Plurality? (unit cost per swap)

Bribing



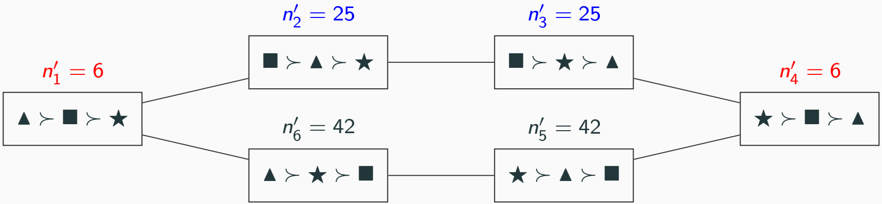
society $\mathbf{n} = (21, 10, 10, 21, 42, 42)$

move $\mathbf{m} = (0, \ldots, 0, +15, +15, 0, \ldots, 0)$ (arc space of complete oriented graph)

change $\boldsymbol{\Delta} = \Delta(\mathbf{m}) = (-15, +15, +15, -15, 0, 0)$

"Bribery:" cheapest way to move voters s.t. ■ wins Plurality? (unit cost per swap)

Bribing

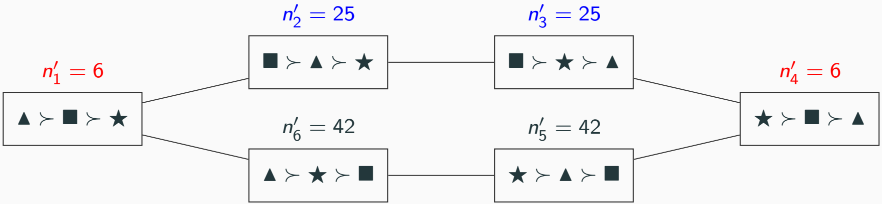


$\mathbf{n}' = \mathbf{n} + \boldsymbol{\Delta}$ with $\boldsymbol{\Delta} = (-15, +15, +15, -15, 0, 0)$

■ **wins:** $48 = n'_1 + n'_6 = n'_4 + n'_5 < n'_2 + n'_3 = 50$

“Bribery:” cheapest way to move voters s.t. ■ wins Plurality? (unit cost per swap)

Bribing



$\mathbf{n}' = \mathbf{n} + \boldsymbol{\Delta}$ with $\boldsymbol{\Delta} = (-15, +15, +15, -15, 0, 0)$

■ **wins:** $48 = n'_1 + n'_6 = n'_4 + n'_5 < n'_2 + n'_3 = 50$

"Bribery:" cheapest way to move voters s.t. ■ wins Plurality? (unit cost per swap)

Actually: Bribery, \$Bribery, Shift Bribery, Swap Bribery, CCDV, etc.

Borda-•-Bribery

- **Borda's Rule:** give $m - 1$ points to 1st candidate, $m - 2$ to 2nd, etc.

Borda-•-Bribery

- **Borda's Rule:** give $m - 1$ points to 1st candidate, $m - 2$ to 2nd, etc.
- $\textsc{Shift-Bribery}_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- $\textsc{Shift-Bribery}_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- $\textsc{Constructive Control by Adding/Deleting Voters}_\infty$: ditto

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, \$Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- $\textsc{Shift-Bribery}_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- $\textsc{Constructive Control by Adding/Deleting Voters}_\infty$: ditto
- Unit cost $\textsc{Swap Bribery}_\infty$: easy (reduces to $\textsc{Shift Bribery}_\infty$)
- $\textsc{Bribery}$, $\$\textsc{Bribery}_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, \$Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints
  - Its dual has $m+\tau$ vars but many constraints

Borda-•-Bribery

- **Borda's Rule:** give $m - 1$ points to 1st candidate, $m - 2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, \$Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m + \tau$ constraints
  - Its dual has $m + \tau$ vars but many constraints
  - Separation $\implies$ Optimization

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, \$Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints
  - Its dual has $m+\tau$ vars but many constraints
  - Separation $\implies$ Optimization
  - Here, separation = sorting

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- $\textsc{Shift-Bribery}_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- $\textsc{Constructive Control by Adding/Deleting Voters}_\infty$: ditto
- Unit cost $\textsc{Swap Bribery}_\infty$: easy (reduces to $\textsc{Shift Bribery}_\infty$)
- $\textsc{Bribery}$, $\$\textsc{Bribery}_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints
  - Its dual has $m+\tau$ vars but many constraints
  - Separation $\implies$ Optimization
  - Here, separation = sorting
- General cost $\textsc{Swap Bribery}_\infty$: ...probably hard?

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, $Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints
  - Its dual has $m+\tau$ vars but many constraints
  - Separation $\implies$ Optimization
  - Here, separation = sorting
- General cost Swap Bribery$_\infty$: ...probably hard?
- "Potentials-cost" Swap Bribery$_\infty$: ...probably easy?

Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, \$Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints
  - Its dual has $m+\tau$ vars but many constraints
  - Separation $\implies$ Optimization
  - Here, separation = sorting
- General cost Swap Bribery$_\infty$: ...probably hard?
- "Potentials-cost" Swap Bribery$_\infty$: ...probably easy?
  - costs like "swapping candidates initially at distance k costs k"

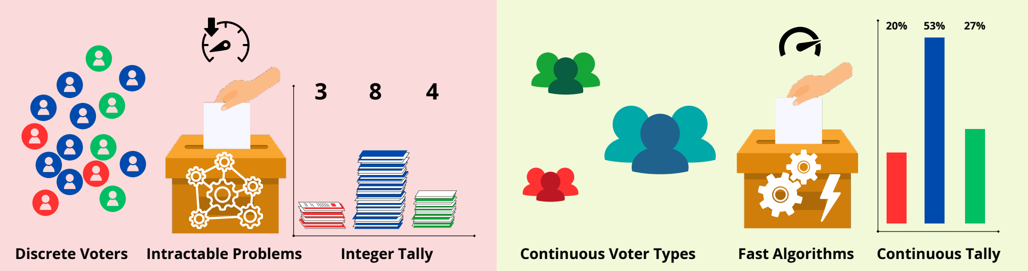
Borda-•-Bribery

- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, $Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m+\tau$ constraints
  - Its dual has $m+\tau$ vars but many constraints
  - Separation $\implies$ Optimization
  - Here, separation = sorting
- General cost Swap Bribery$_\infty$: ...probably hard?
- "Potentials-cost" Swap Bribery$_\infty$: ...probably easy?
  - costs like "swapping candidates initially at distance k costs k"
  - separation problem $\equiv$ special case of Linear Ordering Problem (NP-c)

Borda-•-Bribery

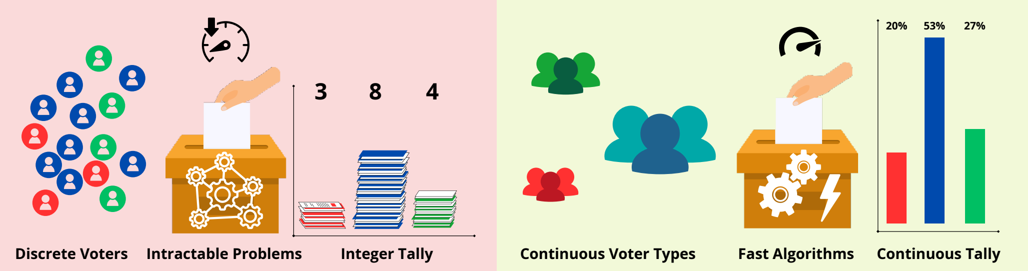
- **Borda's Rule:** give $m-1$ points to 1st candidate, $m-2$ to 2nd, etc.
- Shift-Bribery$_\infty$: easy (LP with $\mathcal{O}(\tau m)$ variables)
- Constructive Control by Adding/Deleting Voters$_\infty$: ditto
- Unit cost Swap Bribery$_\infty$: easy (reduces to Shift Bribery$_\infty$)
- Bribery, \$Bribery$_\infty$: easy-ish (Configuration LP with easy separation problem)
  - LP with $\tau m!$ variables but $m + \tau$ constraints
  - Its dual has $m + \tau$ vars but many constraints
  - Separation $\implies$ Optimization
  - Here, separation = sorting
- General cost Swap Bribery$_\infty$: ...probably hard?
- "Potentials-cost" Swap Bribery$_\infty$: ...probably easy?
  - costs like "swapping candidates initially at distance k costs k"
  - separation problem $\equiv$ special case of Linear Ordering Problem (NP-c)
  - For our costs, optimal face of a known LO relaxation is integral!

Bottom Line



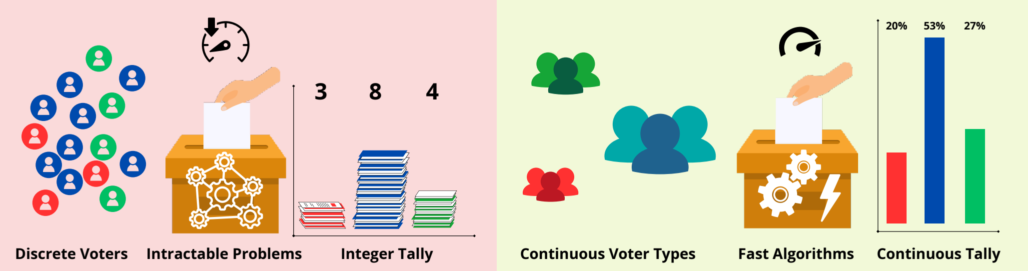
- Est. **half** relevant papers study a problem w/ natural continuous analogue

Bottom Line



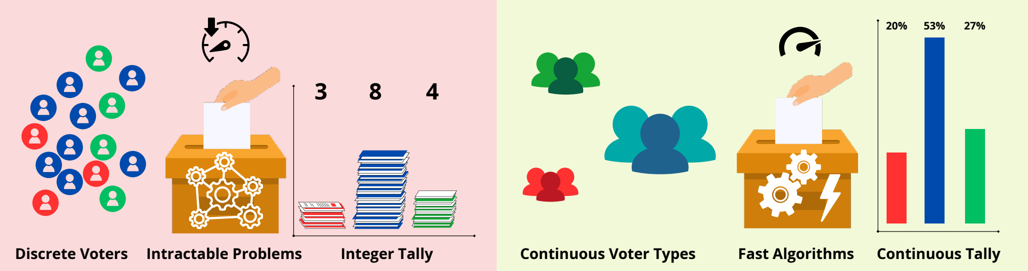
- Est. **half** relevant papers study a problem w/ natural continuous analogue
- Rich and non-trivial new complexity landscape

Bottom Line



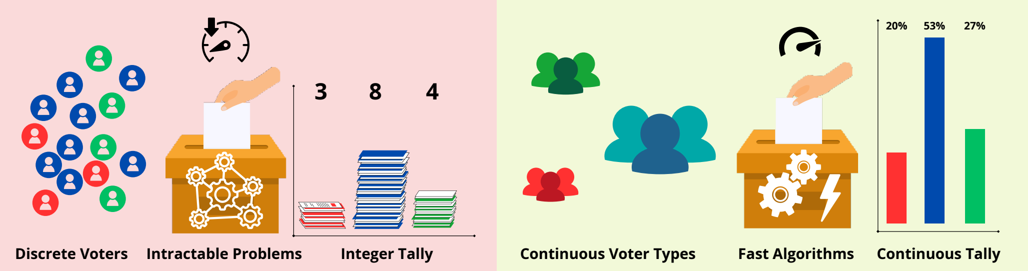
- Est. **half** relevant papers study a problem w/ natural continuous analogue
- Rich and non-trivial new complexity landscape
- Opportunity to use new tools (in ComSoC)

Bottom Line



- Est. **half** relevant papers study a problem w/ natural continuous analogue
- Rich and non-trivial new complexity landscape
- Opportunity to use new tools (in ComSoC)
- May reveal where hardness is a **modeling artifact**

Bottom Line



- Est. **half** relevant papers study a problem w/ natural continuous analogue
- Rich and non-trivial new complexity landscape
- Opportunity to use new tools (in ComSoC)
- May reveal where hardness is a **modeling artifact**

Thank You!