FPT Approximation Schemes for Shift Bribery

Robert Bredereck[1] Jiehua Chen[1] Piotr Faliszewski[2]
André Nichterlein[1] Rolf Niedermeier[1]

[1] Institut für Softwaretechnik und Theoretische Informatik, TU Berlin
[2] AGH University of Science and Technology, Krakow

Motivation

What is the "best" way to cope with NP-hard problems?

Some opinions:

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
(heretical) André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

FPA fixed-parameter approximation

Rand randomized algorithms

Avg average-case analysis

Heur heuristics

Motivation

What is the "best" way to cope with NP-hard problems?

Some opinions:

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
(heretical) André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

FPA fixed-parameter approximation

Rand randomized algorithms

Avg average-case analysis

Heur heuristics

Who wins the poll?

Motivation

What is the "best" way to cope with NP-hard problems?

Some opinions:

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
(heretical) André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

FPA fixed-parameter approximation

Rand randomized algorithms

Avg average-case analysis

Heur heuristics

Who wins the poll? randomized algorithms

Motivation

What is the "best" way to cope with NP-hard problems?

Some opinions:

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
(heretical) André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

FPA fixed-parameter approximation

Rand randomized algorithms

Avg average-case analysis

Heur heuristics

Who wins the poll? randomized algorithms

But: Rolf want fixed-parameter approximation to win.

Motivation

What is the "best" way to cope with NP-hard problems?

Some opinions:

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
(heretical) André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

FPA fixed-parameter approximation
Rand randomized algorithms
Avg average-case analysis
Heur heuristics

Who wins the poll? randomized algorithms

But: Rolf want fixed-parameter approximation to win.
$\rightsquigarrow$ Rolf sends André to workshop "Approximation Algorithms and Parameterized Complexity"

Motivation

What is the "best" way to cope with NP-hard problems?

Some opinions:

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
(converted) André:	FPA	$\succ$	Heur	$\succ$	Avg	$\succ$	Rand

FPA fixed-parameter approximation
Rand randomized algorithms
Avg average-case analysis
Heur heuristics

Who wins the poll? **fixed-parameter approximation**

But: Rolf want fixed-parameter approximation to win.
$\rightsquigarrow$ Rolf sends André to workshop "Approximation Algorithms and Parameterized Complexity"

Some notation

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	FPA	$\succ$	Rand

Some notation

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	FPA	$\succ$	Rand

Election $E = (C, V)$:

- A set of candidates $C = \{c_1, \ldots, c_m\}$.
- A set of voters $V = \{v_1, \ldots, v_n\}$; each voter has a linear order over C.

Some notation

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	FPA	$\succ$	Rand

Election $E = (C, V)$:

- A set of candidates $C = \{c_1, \ldots, c_m\}$.
- A set of voters $V = \{v_1, \ldots, v_n\}$; each voter has a linear order over C.

Voting rule $\mathcal{R}$: $\mathcal{R}(E) \subseteq C$ is the winner set (candidates with most points)

Some notation

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	FPA	$\succ$	Rand

Election $E = (C, V)$:

- A set of candidates $C = \{c_1, \ldots, c_m\}$.
- A set of voters $V = \{v_1, \ldots, v_n\}$; each voter has a linear order over C.

Voting rule $\mathcal{R}$: $\mathcal{R}(E) \subseteq C$ is the winner set (candidates with most points)

Borda $\underline{1}^{\text{st}}$: $m-1$ points, $\underline{2}^{\text{nd}}$: $m-2$ points, $\ldots$, $\underline{m}^{\text{th}}$: 0 points.

Some notation

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	FPA	$\succ$	Rand

Election $E = (C, V)$:

- A set of candidates $C = \{c_1, \ldots, c_m\}$.
- A set of voters $V = \{v_1, \ldots, v_n\}$; each voter has a linear order over C.

Voting rule $\mathcal{R}$: $\mathcal{R}(E) \subseteq C$ is the winner set (candidates with most points)

Borda $\underline{1^{\text{st}}}$: $m-1$ points, $\underline{2^{\text{nd}}}$: $m-2$ points, $\ldots$, $\underline{m^{\text{th}}}$: 0 points.

Copeland$^\alpha$ Head-to-head contest between each two candidates.

Some notation

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	FPA	$\succ$	Rand

Election $E = (C, V)$:

- A set of candidates $C = \{c_1, \ldots, c_m\}$.
- A set of voters $V = \{v_1, \ldots, v_n\}$; each voter has a linear order over C.

Voting rule $\mathcal{R}$: $\mathcal{R}(E) \subseteq C$ is the winner set (candidates with most points)

Borda 1st: $m-1$ points, 2nd: $m-2$ points, …, m^{th}: 0 points.

Copeland$^\alpha$ Head-to-head contest between each two candidates.
Winner: 1 point, loser: 0 points, tie: α points each.

Shift Bribery

Setting:
An external agent may influence voter to shift some preferred candidate forward in the voters' preference list.

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

Shift Bribery

Setting:

An external agent may influence voter to shift some preferred candidate forward in the voters' preference list.

- influence $\in$ {TV-ads, internet-ads, street campaigns, money}

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

Shift Bribery

Setting:

An external agent may influence voter to shift some preferred candidate forward in the voters' preference list.

- influence $\in$ {TV-ads, internet-ads, street campaigns, money}
- external agent has a limited budget

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]

Robert:	FPA	$\succ$	Rand	$\succ$	Avg	$\succ$	Heur
Jiehua:	Rand	$\succ$	FPA	$\succ$	Avg	$\succ$	Heur
André:	Heur	$\succ$	Avg	$\succ$	Rand	$\succ$	FPA

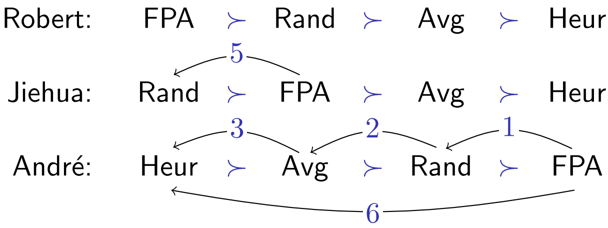
Shift Bribery

Setting:

An external agent may influence voter to shift some preferred candidate forward in the voters' preference list.

- influence $\in$ {TV-ads, internet-ads, street campaigns, money}
- external agent has a limited budget
- each voter has a different price depending on the "shift length"

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]

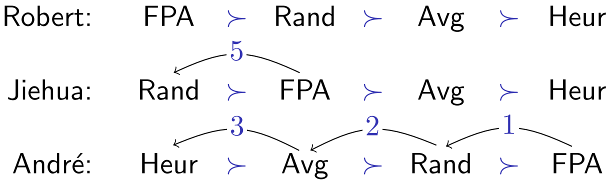


Shift Bribery - Problem Definition

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, a price function list $\Pi = (\pi_1, \ldots, \pi_n)$, and a budget $B \in \mathbb{N}$.

Question: Is there a shift action $\vec{s} = (s_1, \ldots, s_n) \in \mathbb{N}^n$ with $\Pi(\vec{s}) \leq B$ such that p is a winner according to rule $\mathcal{R}$?



$\mathcal{R} :=$ Copeland$^\alpha$ (Head-to-head contest between each two candidates.)

$p :=$ FPA (fixed-parameter approximation) $B := 1$

Shift Bribery - Problem Definition

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, a price function list $\Pi = (\pi_1, \ldots, \pi_n)$, and a budget $B \in \mathbb{N}$.

Question: Is there a shift action $\vec{s} = (s_1, \ldots, s_n) \in \mathbb{N}^n$ with $\Pi(\vec{s}) \leq B$ such that p is a winner according to rule $\mathcal{R}$?

Robert: FPA $\succ$ Rand $\succ$ Avg $\succ$ Heur

5

Jiehua: Rand $\succ$ FPA $\succ$ Avg $\succ$ Heur

3 2 1

André: Heur $\succ$ Avg $\succ$ Rand $\succ$ FPA

$\mathcal{R} :=$ Copeland$^\alpha$ (Head-to-head contest between each two candidates.)

$p :=$ FPA (fixed-parameter approximation) $B := 1$

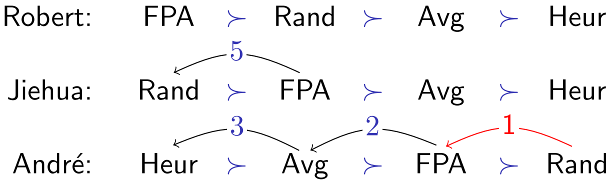
Solution: Yes $\vec{s} = (0, 0, 1)$

Shift Bribery - Problem Definition

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, a price function list $\Pi = (\pi_1, \ldots, \pi_n)$, and a budget $B \in \mathbb{N}$.

Question: Is there a shift action $\vec{s} = (s_1, \ldots, s_n) \in \mathbb{N}^n$ with $\Pi(\vec{s}) \leq B$ such that p is a winner according to rule $\mathcal{R}$?



$\mathcal{R} :=$ Copeland$^\alpha$ (Head-to-head contest between each two candidates.)

$p :=$ FPA (fixed-parameter approximation) $B := 1$

Solution: Yes $\vec{s} = (0, 0, 1)$

What is known about the Shift Bribery?

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]:

- Polynomial-time algorithm for t-Approval voting rule
- NP-hard for Borda, Copeland, and other voting rules
- Polynomial-time 2-approximation for Borda

What is known about the Shift Bribery?

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]:

- Polynomial-time algorithm for t-Approval voting rule
- NP-hard for Borda, Copeland, and other voting rules
- Polynomial-time 2-approximation for Borda

[Elkind and Faliszewski. Approximation algorithms for campaign management. WINE '10]:

- Polynomial-time 2-approximation for all scoring rules (i. e. Borda)
- Polynomial-time m-approximation for Copeland
  ($m =$ number of candidates)

What is known about the Shift Bribery?

[Elkind, Faliszewski, and Slinko. Swap Bribery. SAGT '09]:

- Polynomial-time algorithm for t-Approval voting rule
- NP-hard for Borda, Copeland, and other voting rules
- Polynomial-time 2-approximation for Borda

[Elkind and Faliszewski. Approximation algorithms for campaign management. WINE '10]:

- Polynomial-time 2-approximation for all scoring rules (i. e. Borda)
- Polynomial-time m-approximation for Copeland ($m =$ number of candidates)

[Schlotter, Faliszewski, and Elkind. Campaign management under approval-driven voting rules. AAAI '11]:

- Polynomial-time algorithm for Bucklin and Fallback voting rule
- Problem variant: **Support Bribery** (briber extends incomplete votes)

Our Results

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]:

- There is a factor-$(1+\varepsilon)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** in time $O^*(\lceil n/\varepsilon + 1\rceil^n)$ for $\mathcal{R} \in \{$Copeland, Borda$\}$.

Our Results

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]:

- There is a factor-$(1+\varepsilon)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** in time $O^*(\lceil n/\varepsilon + 1\rceil^n)$ for $\mathcal{R} \in \{$Copeland, Borda$\}$.
- Copeland **Shift Bribery** parameterized by the number n of voters is W[1]-hard for unit prices.

Our Results

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]:

- There is a factor-$(1+\varepsilon)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** in time $O^*(\lceil n/\varepsilon + 1\rceil^n)$ for $\mathcal{R} \in \{$Copeland, Borda$\}$.
- Copeland **Shift Bribery** parameterized by the number n of voters is W[1]-hard for unit prices.
- There is a factor-$(1+2\varepsilon+\varepsilon^2)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** for sortable prices in time $O^*(M^{M\cdot\lceil \ln(M/\varepsilon)\rceil+1})$ (where $M = m \cdot m!$) for $\mathcal{R} \in \{$Copeland, Borda$\}$.

Our Results

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]:

- There is a factor-$(1+\varepsilon)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** in time $O^*(\lceil n/\varepsilon + 1\rceil^n)$ for $\mathcal{R} \in \{$Copeland, Borda$\}$.
- Copeland **Shift Bribery** parameterized by the number n of voters is W[1]-hard for unit prices.
- There is a factor-$(1+2\varepsilon+\varepsilon^2)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** for sortable prices in time $O^*(M^{M\cdot\lceil \ln(M/\varepsilon)\rceil+1})$ (where $M = m \cdot m!$) for $\mathcal{R} \in \{$Copeland, Borda$\}$.

Update: $\mathcal{R}$ **Shift Bribery** for $\mathcal{R} \in \{$Copeland, Borda$\}$ is fixed-parameter tractable with respect to the number m of candidates.

[Bredereck, Faliszewski, Niedermeier, Skowron, and Talmon. Elections with Few Candidates: Prices, Weights, and Covering Problems. ADT' 15]

Further Results

		$\mathcal{R}$ **Shift Bribery**				
parameter	$\mathcal{R}$	unit	convex	all	sort	$0/1$

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]

Further Results

		$\mathcal{R}$ Shift Bribery				
parameter	$\mathcal{R}$	unit	convex	all	sort	$0/1$
#shifts (t)						
budget (B)						
#candidates (m)						
#voters (n)						

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]

Further Results

		$\mathcal{R}$ **Shift Bribery**				
parameter	$\mathcal{R}$	unit	convex	all	sort	$0/1$
#shifts (t)	Borda Copeland$^{\alpha}$					
budget (B)	Borda Copeland$^{\alpha}$					
#candidates (m)	Borda/Copeland$^{\alpha}$					
#voters (n)	Borda Copeland$^{\alpha}$					

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]

Further Results

<table>
<tr><td colspan="2"></td><td colspan="5">$\mathcal{R}$ **Shift Bribery**</td></tr>
<tr><td>parameter</td><td>$\mathcal{R}$</td><td>unit</td><td>convex</td><td>all</td><td>sort</td><td>$0/1$</td></tr>
<tr><td rowspan="2">#shifts (t)</td><td>Borda</td><td colspan="5">FPT</td></tr>
<tr><td>Copeland$^{\alpha}$</td><td colspan="5">W[1]-hard</td></tr>
<tr><td rowspan="2">budget (B)</td><td>Borda</td><td colspan="2">FPT</td><td colspan="3">W[2]-hard</td></tr>
<tr><td>Copeland$^{\alpha}$</td><td colspan="2">W[1]-hard</td><td colspan="3">W[2]-hard</td></tr>
<tr><td>#candidates (m)</td><td>Borda/Copeland$^{\alpha}$</td><td>FPT</td><td colspan="2">XP</td><td colspan="2">FPT-AS</td></tr>
<tr><td rowspan="2">#voters (n)</td><td>Borda</td><td colspan="4">FPT-AS</td><td>FPT</td></tr>
<tr><td>Copeland$^{\alpha}$</td><td colspan="4">FPT-AS / W[1]-hard</td><td>FPT</td></tr>
</table>

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI ’14]

Further Results

		$\mathcal{R}$ Shift Bribery				
parameter	$\mathcal{R}$	unit	convex	all	sort	$0/1$
#shifts (t)	Borda	FPT				
	Copeland$^\alpha$	W[1]-hard				
budget (B)	Borda	FPT		W[2]-hard		
	Copeland$^\alpha$	W[1]-hard		W[2]-hard		
#candidates (m)	Borda/Copeland$^\alpha$	FPT	XP		FPT-AS / FPT*	
#voters (n)	Borda	FPT-AS / W[1]-hard				FPT
	Copeland$^\alpha$	FPT-AS / W[1]-hard				FPT

[Bredereck, Chen, Faliszewski, N., and Niedermeier. Prices matter for the parameterized complexity of shift bribery. AAAI '14]

*[Bredereck, Faliszewski, Niedermeier, Skowron, and Talmon. Elections with Few Candidates: Prices, Weights, and Covering Problems. ADT' 15]

Fixed-Parameter Approximation Scheme I: Overview

Fixed-Parameter Approximation Scheme I: Overview

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, and a price function list $\Pi = (\pi_1, \ldots, \pi_n)$.

Taks: Find a minimum-cost shift action that makes p a winner according to rule $\mathcal{R}$?

Parameter $n := |V|$.

Goal: $(1 + \varepsilon)$ approximation in $f(n, \varepsilon) \cdot \mathrm{poly}$

Fixed-Parameter Approximation Scheme I: Overview

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, and a price function list $\Pi = (\pi_1, \ldots, \pi_n)$.

Taks: Find a minimum-cost shift action that makes p a winner according to rule $\mathcal{R}$?

Parameter $n := |V|$.

Goal: $(1 + \varepsilon)$ approximation in $f(n, \varepsilon) \cdot \mathrm{poly}$

Algorithm outline:

1. Rescale prices to the range $\{0, 1, \ldots, \lceil n/\varepsilon \rceil - 1, \lceil n/\varepsilon \rceil, \infty\}$.
2. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
3. Output cheapest solution.

Fixed-Parameter Approximation Scheme I: Overview

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, and a price function list $\Pi = (\pi_1, \ldots, \pi_n)$.

Taks: Find a minimum-cost shift action that makes p a winner according to rule $\mathcal{R}$?

Parameter $n := |V|$.

Goal: $(1 + \varepsilon)$ approximation in $f(n, \varepsilon) \cdot \mathrm{poly}$

Algorithm outline:

1. Rescale prices to the range $\{0, 1, \ldots, \lceil n/\varepsilon \rceil - 1, \lceil n/\varepsilon \rceil, \infty\}$.
2. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
3. Output cheapest solution.

Observation: Above algorithm works for any voting rule where winner determination is fixed-parameter tractable with respect to n.

Fixed-Parameter Approximation Scheme I: Details

Algorithm outline:

1. Rescale prices to the range $\{0, 1, \ldots, \lceil n/\varepsilon \rceil - 1, \lceil n/\varepsilon \rceil, \infty\}$.
2. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
3. Output cheapest solution.

Fixed-Parameter Approximation Scheme I: Details

Algorithm outline:

1. Rescale prices to the range $\{0, 1, \ldots, \lceil n/\varepsilon \rceil - 1, \lceil n/\varepsilon \rceil, \infty\}$.
2. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
3. Output cheapest solution.

Rescaling:

- Guess maximum cost $\pi_{\max}$ spend at a single voter.
- Set $K = \varepsilon \cdot \pi_{\max}/n$ and

for each $v_i \in V, j \in \{1, \ldots, m\}: \quad \pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$

Fixed-Parameter Approximation Scheme I: Details

Algorithm outline:

1. Rescale prices to the range $\{0, 1, \ldots, \lceil n/\varepsilon \rceil - 1, \lceil n/\varepsilon \rceil, \infty\}$.
2. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
3. Output cheapest solution.

Rescaling:

- Guess maximum cost $\pi_{\max}$ spend at a single voter.
- Set $K = \varepsilon \cdot \pi_{\max}/n$ and

for each $v_i \in V, j \in \{1, \ldots, m\}: \quad \pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$

Running time: Step 1: $O((nm)^2)$
Step 2: $O((\lceil n/\varepsilon \rceil + 1)^n \cdot \text{running time for winner determination})$
In total: $O^*((\lceil n/\varepsilon \rceil + 1)^n)$ for Copeland and Borda voting rule

Fixed-Parameter Approximation Scheme I: Details II

Algorithm outline:

1. Guess maximum cost $\pi_{\max}$ spend at a single voter.
2. Set $K = \varepsilon \cdot \pi_{\max}/n$ and

   for each $v_i \in V, j \in \{1, \ldots, m\}: \quad \pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$
3. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
4. Output cheapest solution.

Fixed-Parameter Approximation Scheme I: Details II

Algorithm outline:

1. Guess maximum cost $\pi_{\max}$ spend at a single voter.
2. Set $K = \varepsilon \cdot \pi_{\max}/n$ and

   for each $v_i \in V, j \in \{1, \ldots, m\}: \quad \pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$
3. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
4. Output cheapest solution.

Approximation factor: algorithm solution: $\vec{s}'$ optimal solution: $\vec{s}$

(A solution is a vector $\in \{0, \ldots, m\}^n$ storing the number of shifts for each voter.)

cost of $\vec{s}'$: $\Pi(\vec{s}')$ cost of optimum solution: $\Pi(\vec{s})$

Fixed-Parameter Approximation Scheme I: Details II

Algorithm outline:

1. Guess maximum cost $\pi_{\max}$ spend at a single voter.
2. Set $K = \varepsilon \cdot \pi_{\max}/n$ and

   for each $v_i \in V, j \in \{1, \dots, m\}$: $\pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$
3. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
4. Output cheapest solution.

Approximation factor: algorithm solution: $\vec{s}'$ optimal solution: $\vec{s}$

(A solution is a vector $\in \{0, \dots, m\}^n$ storing the number of shifts for each voter.)

cost of $\vec{s}'$: $\Pi(\vec{s}')$ cost of optimum solution: $\Pi(\vec{s})$

$$\Pi(\vec{s}') \leq K \cdot \Pi'(\vec{s}')$$

Fixed-Parameter Approximation Scheme I: Details II

Algorithm outline:

1. Guess maximum cost $\pi_{\max}$ spend at a single voter.
2. Set $K = \varepsilon \cdot \pi_{\max}/n$ and

   for each $v_i \in V, j \in \{1,\ldots,m\}: \quad \pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$
3. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
4. Output cheapest solution.

Approximation factor: algorithm solution: $\vec{s}'$ optimal solution: $\vec{s}$

(A solution is a vector $\in \{0,\ldots,m\}^n$ storing the number of shifts for each voter.)

cost of $\vec{s}'$: $\Pi(\vec{s}')$ cost of optimum solution: $\Pi(\vec{s})$

$$\Pi(\vec{s}') \leq K \cdot \Pi'(\vec{s}') \leq K \cdot \Pi'(\vec{s}) \leq \Pi(\vec{s}) + K \cdot n$$

Fixed-Parameter Approximation Scheme I: Details II

Algorithm outline:

1. Guess maximum cost $\pi_{\max}$ spend at a single voter.
2. Set $K = \varepsilon \cdot \pi_{\max}/n$ and

   for each $v_i \in V, j \in \{1, \ldots, m\}: \quad \pi'_i(j) = \begin{cases} \left\lceil \frac{\pi_i(j)}{K} \right\rceil & \text{if } \pi_i(j) \leq \pi_{\max} \\ \infty & \text{otherwise.} \end{cases}$
3. Try all possibilities to spend at most $\lceil n/\varepsilon \rceil$ per voter.
4. Output cheapest solution.

Approximation factor: algorithm solution: $\vec{s}'$ optimal solution: $\vec{s}$
(A solution is a vector $\in \{0, \ldots, m\}^n$ storing the number of shifts for each voter.)

cost of $\vec{s}'$: $\Pi(\vec{s}')$ cost of optimum solution: $\Pi(\vec{s})$

$$\begin{aligned} \Pi(\vec{s}') &\leq K \cdot \Pi'(\vec{s}') \leq K \cdot \Pi'(\vec{s}) \leq \Pi(\vec{s}) + K \cdot n \\ &= \Pi(\vec{s}) + \varepsilon \cdot \pi_{\max} \leq \Pi(\vec{s}) + \varepsilon \cdot \Pi(\vec{s}) = (1+\varepsilon)\Pi(\vec{s}) \end{aligned}$$

Fixed-Parameter Approximation Scheme I: Summary

We saw: There is a factor-$(1+\varepsilon)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** in time $O^*(\lceil n/\varepsilon + 1\rceil^n)$ for $\mathcal{R} \in \{$Copeland, Borda$\}$.

Recall:

- Copeland **Shift Bribery** parameterized by the number n of voters is W[1]-hard for unit prices.
- There is a polynomial-time factor-m approximation algorithm for Copeland **Shift Bribery**. [Elkind and Faliszewski. Approximation algorithms for campaign management. WINE '10]

Open:
Is there a polynomial-time factor-$\log(m)$ (or better) approximation for Copeland **Shift Bribery**?

Fixed-Parameter Approximation Scheme II: Overview

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, and a price function list $\Pi = (\pi_1, \ldots, \pi_n)$.

Taks: Find a minimum-cost shift action that makes p a winner according to rule $\mathcal{R}$?

Parameter $m := |C|$.

Goal: $(1 + \varepsilon)$ approximation in $f(m, \varepsilon) \cdot \mathrm{poly}$

Fixed-Parameter Approximation Scheme II: Overview

$\mathcal{R}$ **Shift Bribery**

Input: An election $E = (C, V)$, a specific candidate $p \in C$, and a price function list $\Pi = (\pi_1, \ldots, \pi_n)$.

Taks: Find a minimum-cost shift action that makes p a winner according to rule $\mathcal{R}$?

Parameter $m := |C|$.

Goal: $(1 + \varepsilon)$ approximation in $f(m, \varepsilon) \cdot \mathrm{poly}$

First Observations:

- at most $m!$ different preference orders
  $\rightsquigarrow$ at most $m!$ "blocks" of voters
- The prices give a sorting of the voters in one block.
  $\rightsquigarrow$ For each two voters of one block, one never shifts p farther in the preference order of the more expensive voter.

Fixed-Parameter Approximation Scheme II: Idea

Brief idea of the algorithm:

Step 1: Spent most of the budget optimally in $O^*(f(m, \varepsilon))$ time:
(exact part of the algorithm)

Step 2: Spent the remaining budget uniformly.
(approximation part of the algorithm)

Fixed-Parameter Approximation Scheme II: Idea

Brief idea of the algorithm:

Step 1: Spent most of the budget optimally in $O^*(f(m, \varepsilon))$ time:
(exact part of the algorithm)

- $B \leftarrow$ available budget
- Repeat several times:

Step 2: Spent the remaining budget uniformly.
(approximation part of the algorithm)

Fixed-Parameter Approximation Scheme II: Idea

Brief idea of the algorithm:

Step 1: Spent most of the budget optimally in $O^*(f(m, \varepsilon))$ time:
(exact part of the algorithm)

- $B \leftarrow$ available budget
- Repeat several times:
  - Guess the voter block where $B/(m \cdot m!)$ budget is spent.
    ($\rightsquigarrow m!$ possibilities)

Step 2: Spent the remaining budget uniformly.
(approximation part of the algorithm)

Fixed-Parameter Approximation Scheme II: Idea

Brief idea of the algorithm:

Step 1: Spent most of the budget optimally in $O^*(f(m, \varepsilon))$ time:
(exact part of the algorithm)

- $B \leftarrow$ available budget
- Repeat several times:
  - Guess the voter block where $B/(m \cdot m!)$ budget is spent.
    ($\rightsquigarrow m!$ possibilities)
  - Guess how far p is shifted in this block with $B/(m \cdot m!)$ budget.
    ($\rightsquigarrow m$ possibilities)
  - $B \leftarrow B - B/(m \cdot m!)$

Step 2: Spent the remaining budget uniformly.
(approximation part of the algorithm)

Fixed-Parameter Approximation Scheme II: Idea

Brief idea of the algorithm:

Step 1: Spent most of the budget optimally in $O^*(f(m, \varepsilon))$ time: (exact part of the algorithm)

- $B \leftarrow$ available budget
- Repeat several times:
  - Guess the voter block where $B/(m \cdot m!)$ budget is spent. ($\rightsquigarrow m!$ possibilities)
  - Guess how far p is shifted in this block with $B/(m \cdot m!)$ budget.
    ($\rightsquigarrow m$ possibilities)
  - $B \leftarrow B - B/(m \cdot m!)$

Step 2: Spent the remaining budget uniformly. (approximation part of the algorithm)

Theorem

There is a factor-$(1 + 2\varepsilon + \varepsilon^2)$ approximation algorithm solving $\mathcal{R}$ **Shift Bribery** for sortable prices in time $O^*(M^{M \cdot \lceil \ln(M/\varepsilon) \rceil + 1})$ (where $M = m \cdot m!$) for $\mathcal{R} \in \{$Copeland, Borda$\}$.

Fixed-Parameter Approximation Scheme II: Remarks

- Algorithm relies on two observations:
  - At most $m!$ different preference orders (voter blocks).
  - Prices induce sorting on voters in one voter block.
- FPT algorithm with respect to parameter m uses the same two observations to construct a mixed integer programming.

Fixed-Parameter Approximation Scheme II: Remarks

- Algorithm relies on two observations:
  - At most $m!$ different preference orders (voter blocks).
  - Prices induce sorting on voters in one voter block.
- FPT algorithm with respect to parameter m uses the same two observations to construct a mixed integer programming.

Open questions:

- What is the parameterized complexity in the unrestricted price functions?
  (Known: **Shift Bribery** is in XP wrt. m)
- Is there a fixed-parameter approximation scheme for unrestricted price functions?

Conclusion

Mostly only polynomial-time approximation algorithms or only fixed-parameter algorithms for voting problems.

- $\rightsquigarrow$ Just few fixed-parameter approximation results for voting problems.
- $\rightsquigarrow$ Search for fixed-parameter approximation algorithms in social choice!

Conclusion

Mostly only polynomial-time approximation algorithms or only fixed-parameter algorithms for voting problems.

- $\rightsquigarrow$ Just few fixed-parameter approximation results for voting problems.
- $\rightsquigarrow$ Search for fixed-parameter approximation algorithms in social choice!

Thank you!