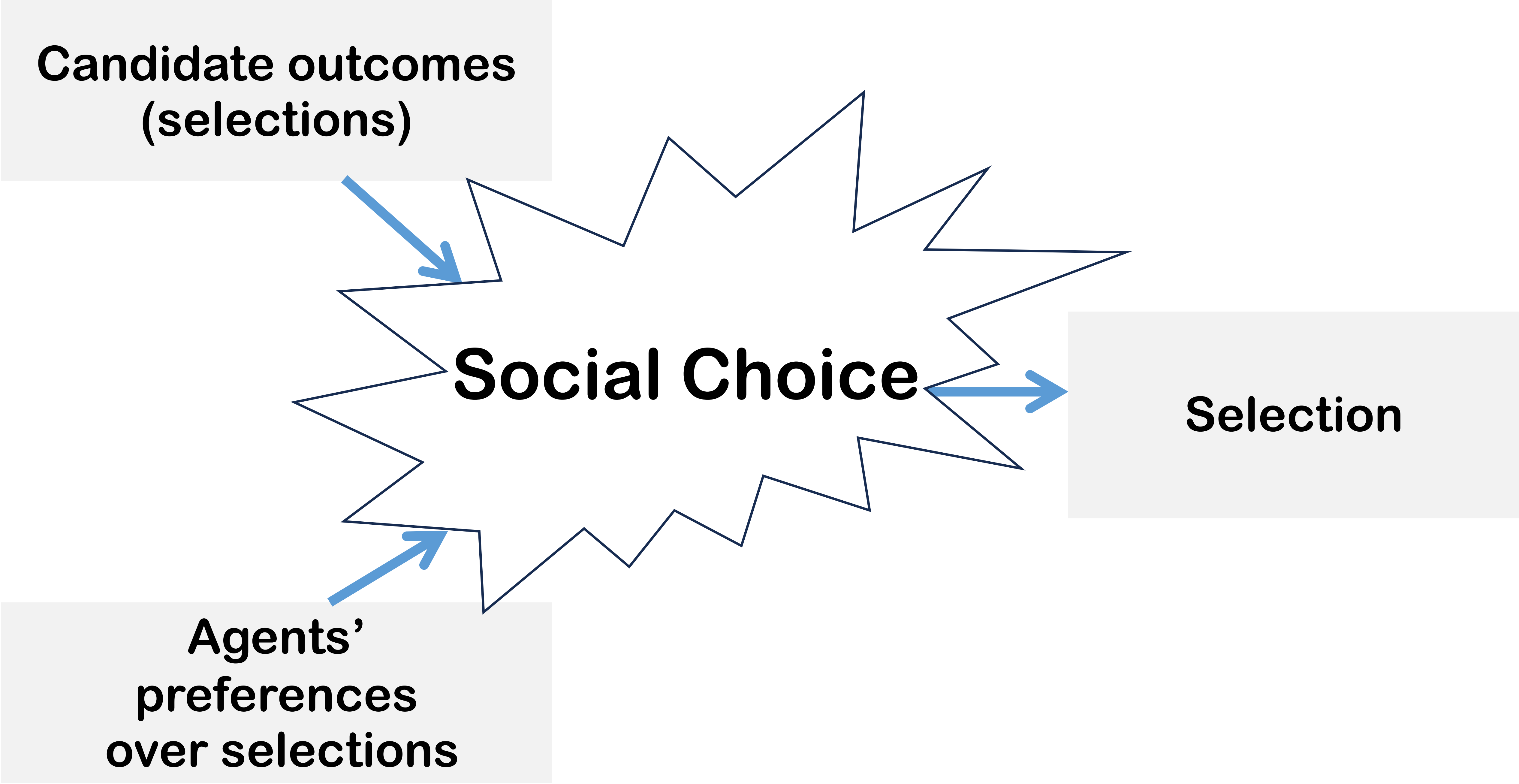
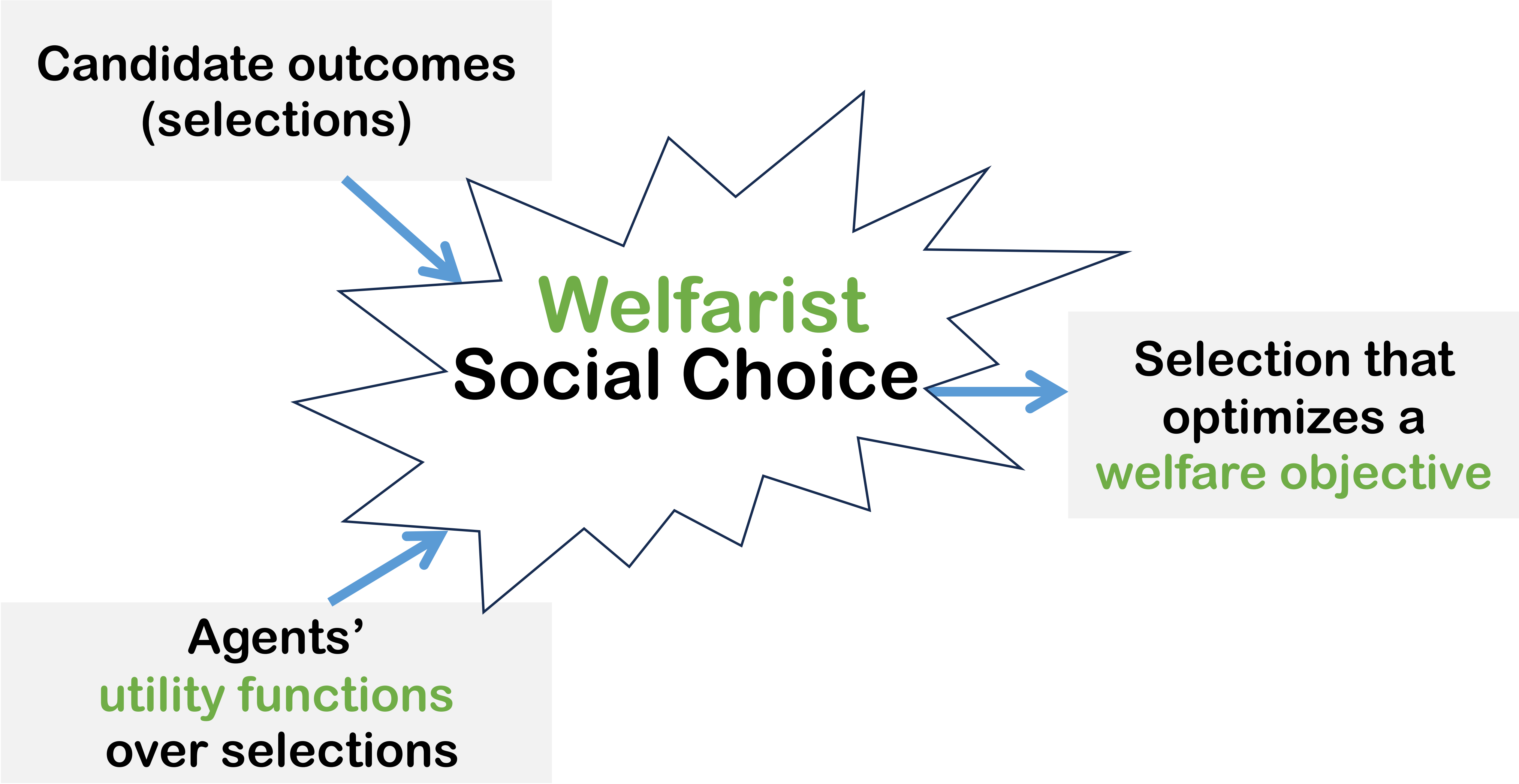
"If your brother becomes poor and cannot maintain himself with you, you shall support though he were a stranger and a sojourner, so that he shall live with you." **(Leviticus 25)**

Reducing Leximin Fairness to Utilitarian Optimization

Accepted to AAAI'25 ☺

Eden Hartman, Yonatan Aumann, Avinatan Hassidim, Erel Segal-HaLevi


Candidate outcomes (selections)
Social Choice
Selection
Agents' preferences over selections


Candidate outcomes (selections)
Welfarist
Social Choice
Selection that optimizes a welfare objective
Agents' utility functions over selections

Welfare objectives

Utilitarian

Maximize the **sum** of agents’ utilities.

Egalitarian

Maximize the **minimum** utility.

Leximin

Maximize the minimum utility.
subject to this, maximize second-smallest utility;
subject to this, maximize third-smallest utility;
and so on.

Welfare objectives – Computation

Utilitarian	Maximize the **sum** of agents' utilities.
Egalitarian	Maximize the **minimum** utility.
Leximin	Maximize the minimum utility. subject to this, maximize second-smallest utility; subject to this, maximize third-smallest utility; and so on.

Welfare objectives – Complexity

Example: Allocating Indivisible Goods Among Agents With Additive Utilities

Utilitarian	Poly: greedily assign each item to the agent who values it most.
Egalitarian	NP-hard Approximation algorithms exist (e.g. Bansal and Sviridenko, 2006).
Leximin	⇒ NP-hard. We are not aware of any approximation algorithms.

The Santa Claus problem. Bansal and Sviridenko, 2006.

Our Contribution

For **any** social choice problem with utilities ≥ 0:

- If you give us a black-box that computes a utilitarian-optimal[1] selection in polytime –
- then we give you a polytime algorithm that computes a leximin-optimal (in expectation) lottery over selections.

Our reduction is **robust**:

- If your black-box is α-approximately utilitarian-optimal -
- then our lottery is α-approximately leximin-optimal[2].

1. for weighted (non-normalized) utilities.
2. for an appropriate definition of 'approximately leximin-optimal'.

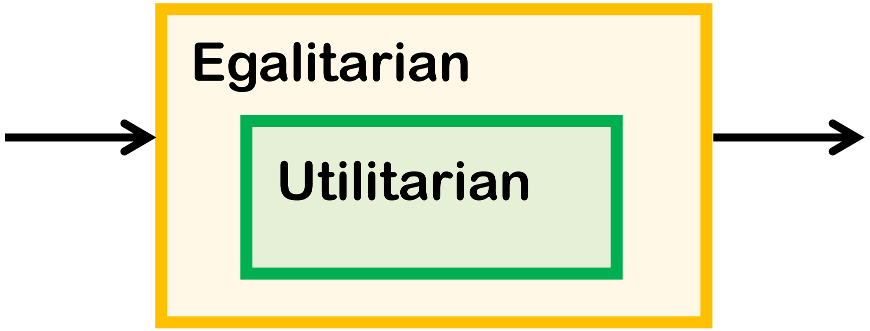
Previous Work (Kawase & Sumita, 2020)

Problem

Stochastic Allocation of Indivisible Goods.

- The output is a lottery over the set of (deterministic) allocations.

<u>Theorem</u>: Given an α-approx. alg. for utilitarian one can obtain an α-approx. alg. for egalitarian in polynomial time.



On the Max-Min Fair Stochastic Allocation of Indivisible Goods. Kawase and Sumita, 2020.

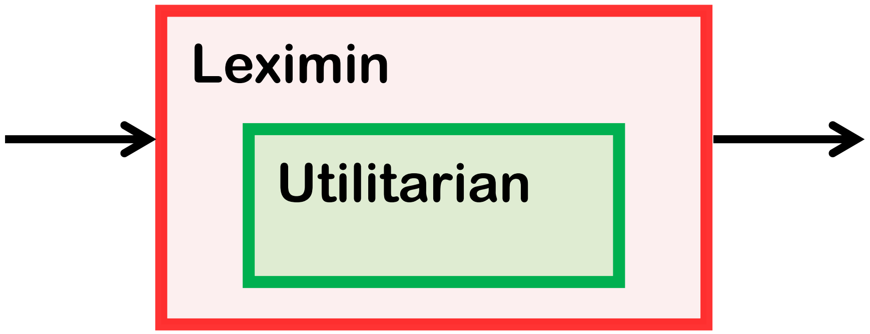
Our Work (2024)

Problem

Any social choice problem with utilities ≥ 0.

- The output is a lottery over the set of (deterministic) selections.

Theorem: Given an α-approx. alg. for utilitarian one can obtain an α-approx. alg. for leximin in polynomial time.



Our Work (2024)

Problem

Any social choice problem with utilities ≥ 0.

- The output is a lottery over the set of (deterministic) selections.

Theorem: Given an α-approx. alg. for utilitarian one can obtain an α-approx. alg. for leximin in polynomial time.

Applications

Stochastic Indivisible Allocation[1]

- Additive utilities $\Rightarrow$ Exact
- Submodular utilities $\Rightarrow$ 0.5-approx (det)
- Submodular utilities $\Rightarrow$ $\left(1-\frac{1}{e}\right)$-approx (rand)

Participatory Budgeting Lottery[2]

- An FPTAS for leximin

Giveaway Lotteries[3]

- An FPTAS for leximin

1. On the Max-Min Fair Stochastic Allocation of Indivisible Goods. Kawase and Sumita, 2020.
2. Fair Lotteries for Participatory Budgeting. Aziz, Lu, Suzuki, Vollen, and Walsh, 2024.
3. Fair and Truthful Giveaway Lotteries. Arbiv and Aumann, 2022.

The Model

Agents	$N = \{1, \dots, n\}$
Selections	$S = \{ s_1, \dots, s_{\lvert S \rvert} \}$ (deterministic outcomes)
Solutions	A solution is a <u>distribution</u> over selections. $X = \left\{ \boldsymbol{x} = \left(x_1, \dots, x_{\lvert S \rvert}\right) \in \mathbb{R}_{\geq 0}^{\lvert S \rvert} \;\middle|\; \sum_{j=1}^{\lvert S \rvert} x_j = 1 \right\}$
Utilities	$u_i : S \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ (a value oracle)
Expected Utilities	$E_i : X \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ $E_i(x) = \sum_{j=1}^{\lvert S \rvert} x_j \cdot u_i(s_j)$

The Model

Agents	$N = \{1, \dots, n\}$
Selections	$S = \{ s_1, \dots, s_{\lvert S \rvert} \}$ (deterministic outcomes)
Solutions	A solution is a <u>distribution</u> over selections. $X = \left\{ \boldsymbol{x} = (x_1, \dots, x_{\lvert S \rvert}) \in \mathbb{R}_{\geq 0}^{\lvert S \rvert} \;\middle\vert\; \sum_{j=1}^{\lvert S \rvert} x_j = 1 \right\}$
Utilities	$u_i : S \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ (a value oracle)
Expected Utilities	$E_i : X \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ $E_i(x) = \sum_{j=1}^{\lvert S \rvert} x_j \cdot u_i(s_j)$

Utilitarian Opt.

A solution $\boldsymbol{x}^{uo} \in X$ s.t $\forall \boldsymbol{x} \in X$:

$$\sum_{i=1}^{\lvert S \rvert} E_i(\boldsymbol{x}^{uo}) \geq \sum_{i=1}^{\lvert S \rvert} E_i(\boldsymbol{x})$$

The Model

Agents	$N = \{1, \dots, n\}$
Selections	$S = \{\, s_1, \dots, s_{\lvert S \rvert} \,\}$ (deterministic outcomes)
Solutions	A solution is a distribution over selections. $X = \left\{ \boldsymbol{x} = (x_1, \dots, x_{\lvert S \rvert}) \in \mathbb{R}_{\geq 0}^{\lvert S \rvert} \;\middle|\; \sum_{j=1}^{\lvert S \rvert} x_j = 1 \right\}$
Utilities	$u_i : S \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ (a value oracle)
Expected Utilities	$E_i : X \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ $E_i(x) = \sum_{j=1}^{\lvert S \rvert} x_j \cdot u_i(s_j)$

Weighted Utilitarian Opt.

Given n constants $c_1, \dots, c_n \geq 0$.

A solution $\boldsymbol{x}^{uo} \in X$ s.t $\forall \boldsymbol{x} \in X$:

$$\sum_{i=1}^{\lvert S \rvert} \boldsymbol{c_i} \cdot E_i(\boldsymbol{x}^{uo}) \geq \sum_{i=1}^{\lvert S \rvert} \boldsymbol{c_i} \cdot E_i(\boldsymbol{x})$$

The Model

Agents	$N = \{1, \ldots, n\}$
Selections	$S = \{ s_1, \ldots, s_{\lvert S \rvert} \}$ (deterministic outcomes)
Solutions	A solution is a <u>distribution</u> over selections. $X = \left\{ \boldsymbol{x} = (x_1, \ldots, x_{\lvert S \rvert}) \in \mathbb{R}_{\geq 0}^{\lvert S \rvert} \;\middle\vert\; \sum_{j=1}^{\lvert S \rvert} x_j = 1 \right\}$
Utilities	$u_i : S \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ (a value oracle)
Expected Utilities	$E_i : X \to \mathbb{R}_{\geq 0} \quad \forall i \in N$ $E_i(x) = \sum_{j=1}^{\lvert S \rvert} x_j \cdot u_i(s_j)$

Leximin Opt.

A solution $\boldsymbol{x}^* \in X$ s.t $\forall \boldsymbol{x} \in X$:

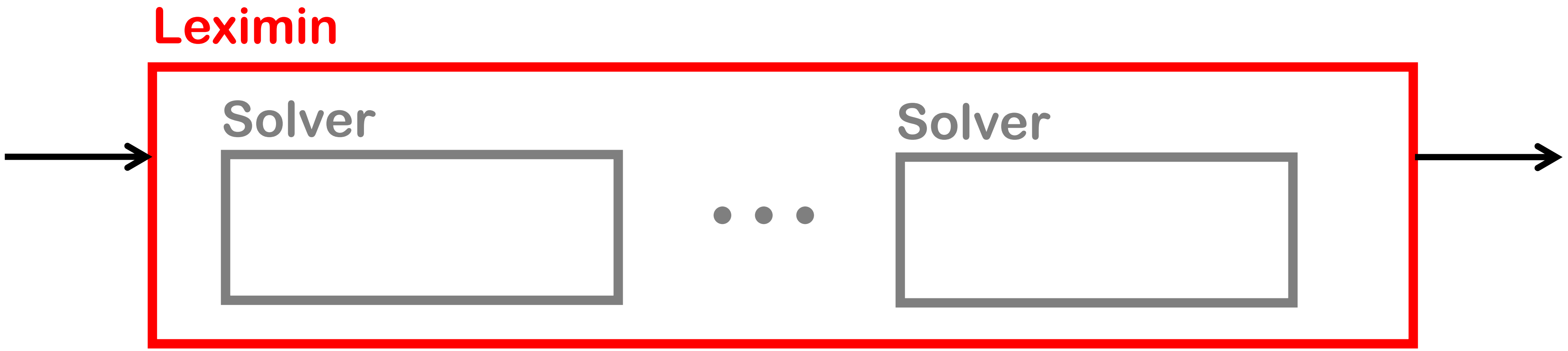
$$\left(E_1(x^*), \ldots, E_n(x^*)\right) \succcurlyeq \left(E_1(x), \ldots, E_n(x)\right)$$

$\boldsymbol{v}^{\uparrow}$	$\boldsymbol{v}$ sorted in weakly-increasing order

Leximin order $\boldsymbol{v} \succcurlyeq \boldsymbol{u}$	if (1) $\boldsymbol{v}^{\uparrow} = \boldsymbol{u}^{\uparrow}$ or (2) $\exists\, 1 \leq k \leq n$ s.t (2.1) $v_i^{\uparrow} = u_i^{\uparrow}$ for $i < k$ (2.2) $v_k^{\uparrow} > u_k^{\uparrow}$

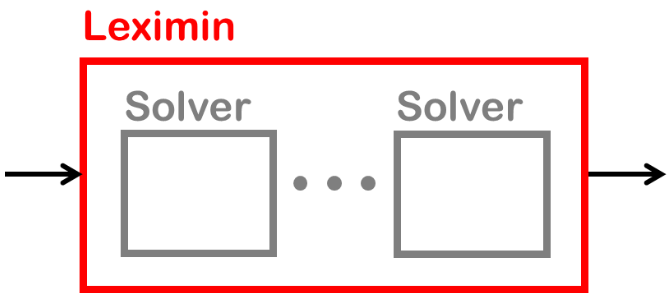
Leximin Optimization

- Leximin is a multi-objective problem.
- Algorithms are usually:
  - Iterative, solve one program in each iteration



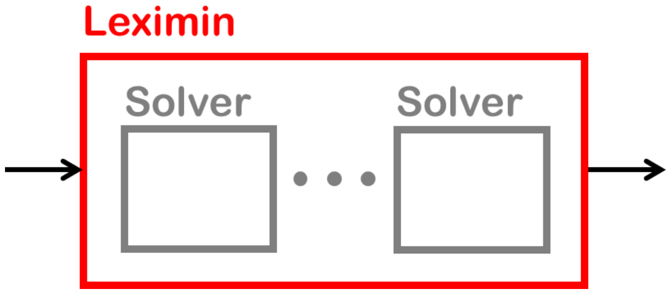
  - Each iteration uses the opt. values from previous iterations as constants.

Leximin Optimization



- Leximin is a multi-objective problem.
- Algorithms are usually:
  - Iterative, solve one program in each iteration
  - Each iteration uses the opt. values from previous iteration as constants.

Leximin Optimization



- Leximin is a <u>multi-objective</u> problem.

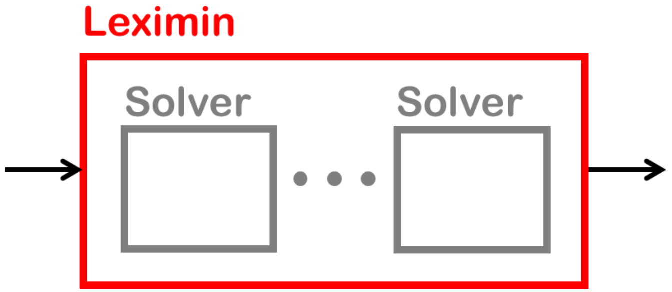
- Algorithms are usually:
  - Iterative, solve one program in each iteration
  - When $t = 1$, solve egalitarian:

$$\max \quad E_{\mathbf{1}}^{\uparrow}(\boldsymbol{x})$$
$$s.t. \quad \boldsymbol{x} \in X$$

$E_{i}^{\uparrow}(\boldsymbol{x})$ The i-th smallest expected-utility obtained by $\boldsymbol{x}$

  - Each iteration uses the opt. values from previous iteration as constants.

Leximin Optimization



- Leximin is a <u>multi-objective</u> problem.

- Algorithms are usually:
  - Iterative, solve one program in each iteration
  - When $t = 1$, solve egalitarian:

$$
\begin{aligned}
\max \quad & z \\
s.t. \quad & (1)\ \boldsymbol{x} \in X \\
& (2)\ E_i(\boldsymbol{x}) \geq z \qquad \forall\, i = 1, \dots, n
\end{aligned}
$$

  - Each iteration uses the opt. values from previous iteration as constants.

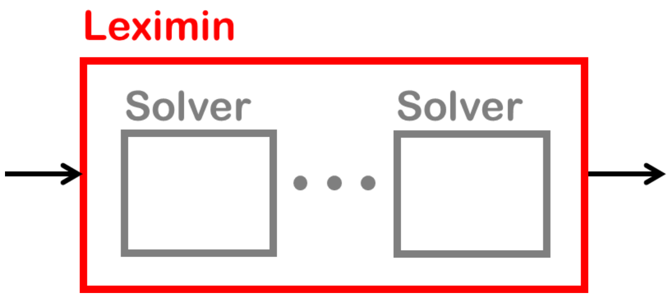
The Ordered Outcomes Algorithm (Ogryczak & Sliwinski, 2006)

- **For t = 1, … n:**
  - **Solve:**

$$
\begin{array}{lll}
\max & \displaystyle\sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i & \text{(P1)} \\
s.t. & \text{(P1.1)} \quad \displaystyle\sum_{j=1}^{|S|} x_j = 1 & \\
 & \text{(P1.2)} \quad x_j \geq 0 & j = 1, \ldots, |S| \\
 & \text{(P1.3)} \quad \displaystyle\sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i & \forall \ell \in [t-1]
\end{array}
$$

  - **Let** x^t **be the solution, and** z^t **be its objective value.**
- **Return** x^n

$E_i^{\uparrow}(\boldsymbol{x})$	The i-th smallest expected-utility at solution $\boldsymbol{x}$



On Direct Methods for Lexicographic Min-Max Optimization. Ogryczak and Sliwinski, 2006.

The Ordered Outcomes Algorithm

- For $t = 1, \dots n$:
  - Solve:

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i & & \text{(P1)} \\
s.t. \quad & \text{(P1.1)} \ \sum_{j=1}^{|S|} x_j = 1 & & \\
& \text{(P1.2)} \ x_j \geq 0 & & j = 1, \dots, |S| \\
& \text{(P1.3)} \ \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i & & \forall \ell \in [t-1]
\end{aligned}
$$

  - Let x^t be the solution, and z^t be its objective value.
- Return x^n

Theorem (Ogryczak & Sliwinski):

Given an exact solver for P1,
the alg. output is leximin-optimal.

In some situations,
solving P1 exactly is NP-hard
(even for $t = 1$)

E.g., stochastic allocation
of indivisible goods among agents
with submodular utilities

Theorem (Kawase & Sumita 2020):
Optimal Egalitarian is NP-hard.

- On Direct Methods for Lexicographic Min-Max Optimization. Ogryczak and Sliwinski, 2006.
- On the Max-Min Fair Stochastic Allocation of Indivisible Goods. Kawase and Sumita, 2020.

The Ordered Outcomes Algorithm

- **For t = 1, … n:**
  - **Solve:**

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i && \text{(P1)} \\
s.t. \quad & \text{(P1.1)} \ \sum_{j=1}^{|S|} x_j = 1 && \\
& \text{(P1.2)} \ x_j \geq 0 && j = 1, \ldots, |S| \\
& \text{(P1.3)} \ \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i && \forall \ell \in [t-1]
\end{aligned}
$$

  - **Let** x^t **be the solution, and** z^t **be its objective value.**
- **Return** x^n

Approximations?

On the Max-Min Fair Stochastic Allocation of Indivisible Goods. Kawase and Sumita, 2020.

The Ordered Outcomes Algorithm

- **For t = 1, … n:**
  - **Solve:**

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i && \text{(P1)} \\
s.t. \quad & \text{(P1.1)} \ \sum_{j=1}^{|S|} x_j = 1 \\
& \text{(P1.2)} \ x_j \geq 0 && j = 1, \ldots, |S| \\
& \text{(P1.3)} \ \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i && \forall \ell \in [t-1]
\end{aligned}
$$

  - **Let** x^t **be the solution, and** z^t **be its objective value.**
- **Return** x^n

Solving P1 Approximately

For $t = 1$:

The alg. of Kawase & Sumita is an approximate solver for (P1).

✓

For $t > 1$:

Their technique no longer works

(due to subtraction in objective)

☹

On the Max-Min Fair Stochastic Allocation of Indivisible Goods. Kawase and Sumita, 2020.

The Ordered Outcomes Algorithm

- For $t = 1, \ldots n$:
  - Solve:

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i && \text{(P1)} \\
s.t. \quad & \text{(P1.1)} \ \sum_{j=1}^{|S|} x_j = 1 \\
& \text{(P1.2)} \ x_j \geq 0 && j = 1, \ldots, |S| \\
& \text{(P1.3)} \ \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i && \forall \ell \in [t-1]
\end{aligned}
$$

  - Let x^t be the solution, and z^t be its objective value.
- Return x^n

What we do:

Define a new type of solver for P1 (even weaker than the approx. one)

- A shallow solver -

Provide a shallow solver for P1 that requires only a black-box for utilitarian.

Prove:

Given a shallow solver for P1, the alg. output is leximin-approx.

Leximin Approximation

Leximin Optimal

A solution $\boldsymbol{x}^* \in X$ s.t $\forall \boldsymbol{x} \in X$:

$$\big(E_1(x^*), \dots, E_n(x^*)\big) \succcurlyeq \big(E_1(x\,), \dots, E_n(x\,)\big)$$

Since leximin has multiple objectives,
defining leximin-approximation is not a trivial task.[1]

In this work we use the following definition:

Leximin Approx

$0 < \alpha \leq 1$

A solution $\boldsymbol{x}^A \in X$ s.t $\forall \boldsymbol{x} \in X$:

$$\big(E_1(x^A), \dots, E_n(x^A)\big) \succcurlyeq \alpha \cdot \big(E_1(x\,), \dots, E_n(x\,)\big)$$

What we do:

Define a new type of solver for P1
(even weaker than the approx. one)

- A shallow solver -

Provide a shallow solver for P1
that requires only
a black-box for utilitarian.

Prove:
Given a shallow solver for P1,
The alg. output is **leximin-approx**.

1. Leximin Approximation: From Single-Objective to Multi-Objective. Hartman, Hassidim, Aumann, and Segal-HaLevi, ECAI 2023.

The Ordered Outcomes Algorithm

- For t = 1, … n:
  - Solve:

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i & & \text{(P1)} \\
s.t. \quad & \text{(P1.1)} \ \sum_{j=1}^{|S|} x_j = 1 & & \\
& \text{(P1.2)} \ x_j \geq 0 & & j = 1, \ldots, |S| \\
& \text{(P1.3)} \ \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i & & \forall \ell \in [t-1]
\end{aligned}
$$

  - Let x^t be the solution, and z^t be its objective value.
- Return x^n

What we do:

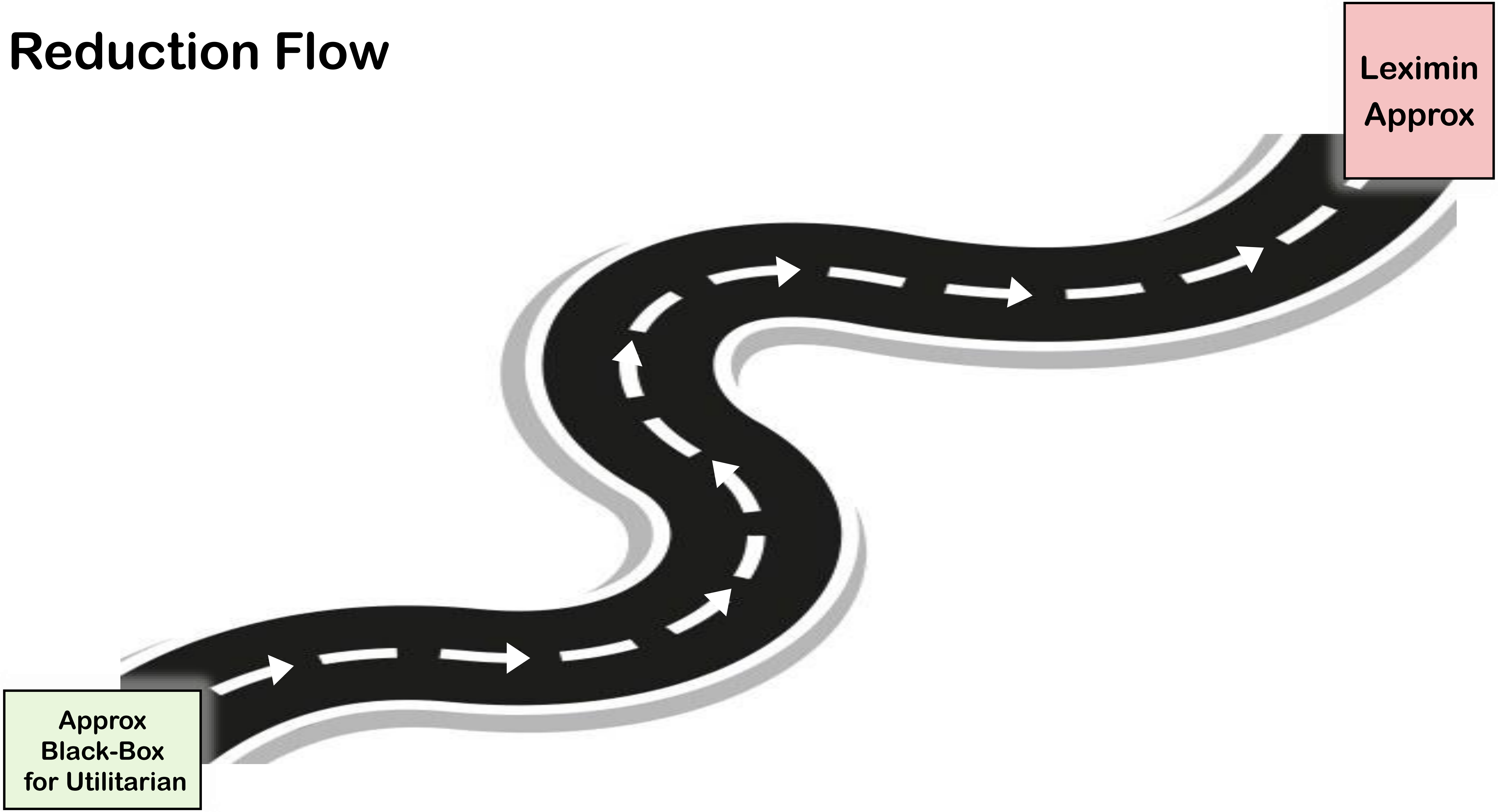
Define a new type of solver for P1
(even weaker than the approx. one)

- A shallow solver -

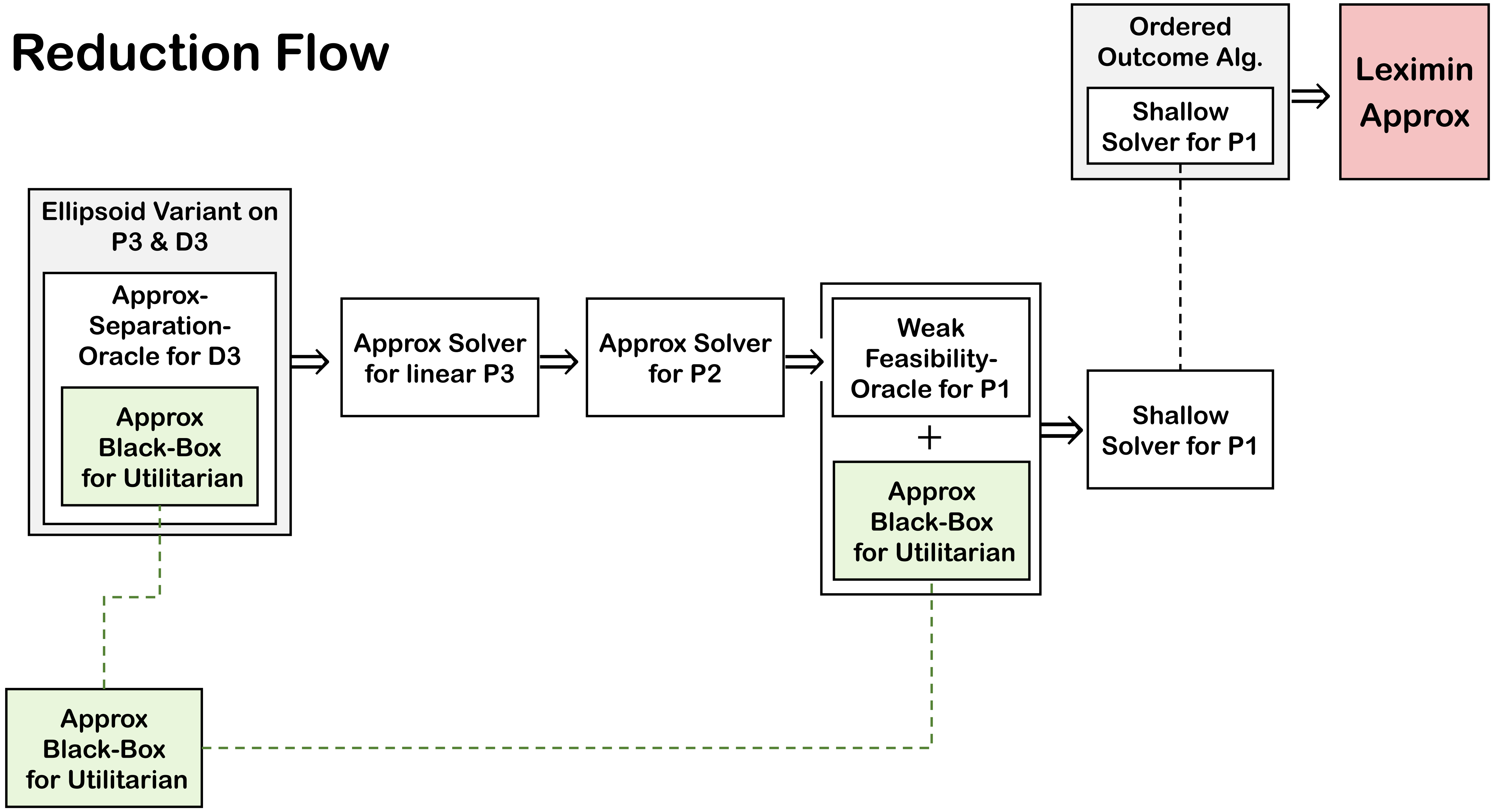
Provide a shallow solver for P1
that requires only
a black-box for utilitarian.

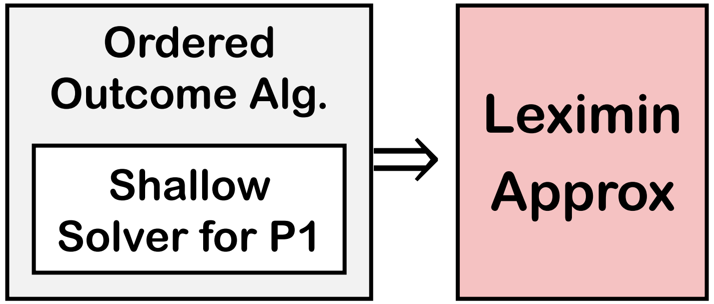
Prove:
Given a shallow solver for P1,
The alg. output is leximin-approx.

Reduction Flow



Reduction Flow





Lemma 5.2 in paper

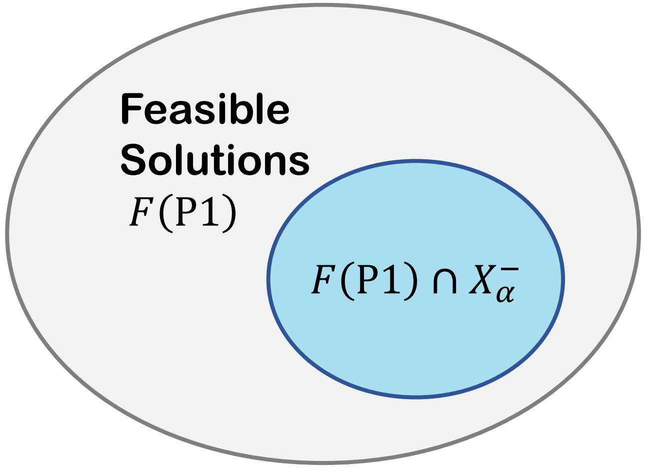
Given an α-shallow-solver for P1,

the output of Ordered Outcomes Alg. is α-leximin-approx.

An α-Shallow Solver: Definition

Solver	returns $\boldsymbol{x}^t \in F(\mathrm{P1})$ (<u>feasible</u> for P1)
Exact	obj($\boldsymbol{x}^t$) $\geq$ obj($\boldsymbol{x}$) for any $\boldsymbol{x} \in F(\mathrm{P1})$
α-Shallow	obj($\boldsymbol{x}^t$) $\geq$ obj($\boldsymbol{x}$) for any $\boldsymbol{x} \in F(\mathrm{P1}) \cap X_\alpha^-$

solutions using $\leq \alpha$ probability[1]



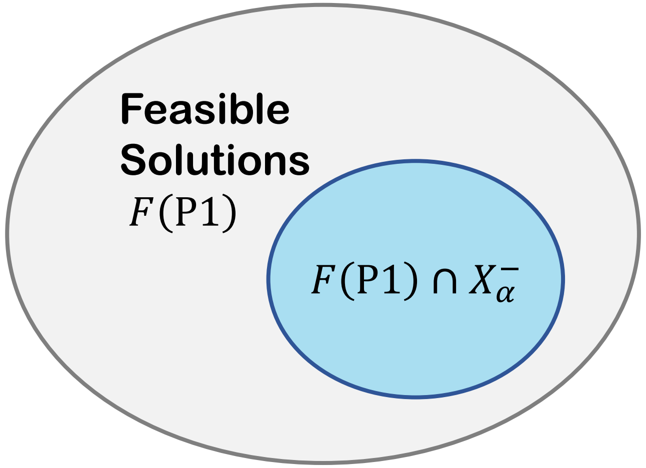
[1] We assume there is a <u>dummy selection</u> that gives all agents utility 0.

An α-Shallow Solver: Definition

Solver	returns $\boldsymbol{x}^t \in F(\mathrm{P1})$ (<u>feasible</u> for P1)
Exact	obj($\boldsymbol{x}^t$) $\geq$ obj($\boldsymbol{x}$) for any $\boldsymbol{x} \in F(\mathrm{P1})$
α-Shallow	obj($\boldsymbol{x}^t$) $\geq$ obj($\boldsymbol{x}$) for any $\boldsymbol{x} \in F(\mathrm{P1}) \cap X_\alpha^-$

$\alpha = 0.9$ solutions using ≤ 0.9 probability[1]

(instead of 1)



[1] We assume there is a <u>dummy selection</u> that gives all agents utility 0, $s_d \in S$.

An α-Shallow Solver for P1

Approx Black-Box for Utilitarian $+$ Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

(a) Approx. black-box for utilitarian
(b) An arbitrary $x_{init} \in F(\mathrm{P1})$ $\quad\Rightarrow\quad$ A shallow-solver for P1
(c) <u>Weak Feasibility-Oracle</u> for P1

An α-Shallow Solver for P1

Approx Black-Box for Utilitarian + Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

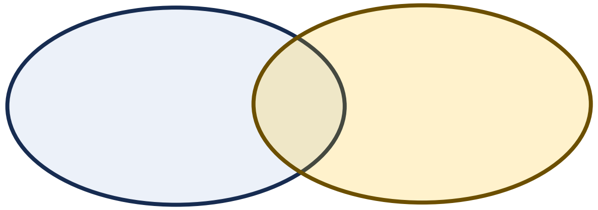
(a) Approx. black-box for utilitarian
(b) An arbitrary $x_{init} \in F(\mathrm{P1})$ $\quad\Rightarrow\quad$ A shallow-solver for P1
(c) Weak Feasibility-Oracle for P1

Weak Feasibility-Oracle for P1 (Definition)

Given a candidate objective value $z_t \in \mathbb{R}_{\geq 0}$, returns one of the following:

- z_t is a feasible objective-value:
  $\exists\, \boldsymbol{x} \in F(\mathrm{P1})$ with objective value $\geq z_t$. In this case, returns such $\boldsymbol{x}$.

- z_t is an infeasible objective-value for solutions using $\leq \alpha$ probability
  $\nexists\, \boldsymbol{x} \in F(\mathrm{P1}) \cap X_\alpha^-$ with objective value $\geq z_t$.



Weak = not mutually exclusive

An α-Shallow Solver for P1

Approx Black-Box for Utilitarian + Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

(a) Approx. black-box for utilitarian

(b) An arbitrary $x_{init} \in F(\mathrm{P1})$ $\Rightarrow$ A shallow-solver for P1

(c) <u>Weak Feasibility-Oracle</u> for P1

Weak Feasibility-Oracle for P1

Given $z_t \in \mathbb{R}_{\geq 0}$ returns one of:

- Feasible objective
- Infeasible objective for sol. using $\leq \alpha$ prob.

An α-Shallow Solver for P1

Approx Black-Box for Utilitarian + Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

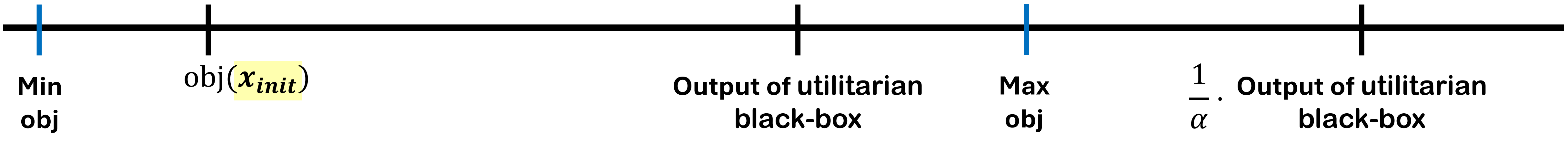
(a) Approx. black-box for utilitarian
(b) An arbitrary $x_{init} \in F(\mathrm{P1})$ $\quad\Rightarrow\quad$ A shallow-solver for P1
(c) <u>Weak Feasibility-Oracle</u> for P1

Weak Feasibility-Oracle for P1

Given $z_t \in \mathbb{R}_{\geq 0}$ returns one of:

- Feasible objective
- Infeasible objective for sol. using $\leq \alpha$ prob.

1. Bound the maximum objective value



An α-Shallow Solver for P1

Approx Black-Box for Utilitarian $+$ Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

(a) Approx. black-box for utilitarian
(b) An arbitrary $x_{init} \in F(\text{P1})$
(c) Weak Feasibility-Oracle for P1

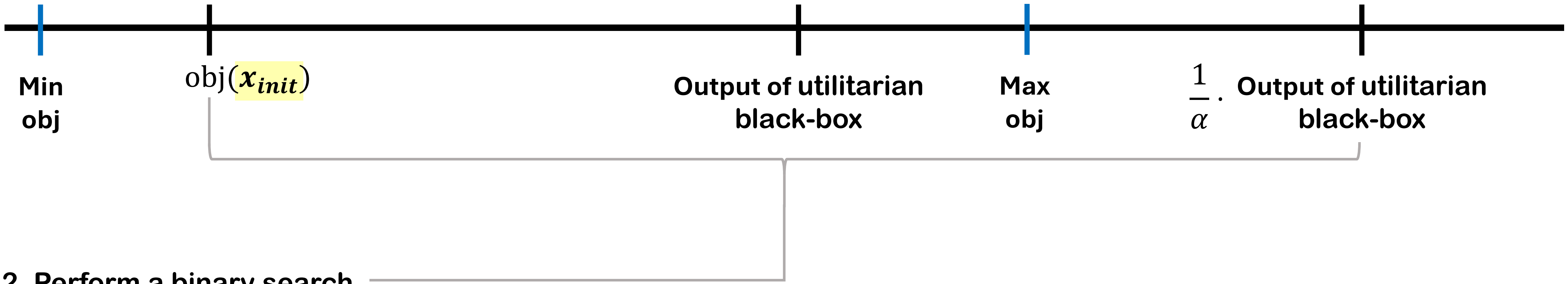
$\Rightarrow$ A shallow-solver for P1

Weak Feasibility-Oracle for P1

Given $z_t \in \mathbb{R}_{\geq 0}$ returns one of:

- Feasible objective
- Infeasible objective for sol. using $\leq \alpha$ prob.

1. Bound the maximum objective value



2. Perform a binary search

An α-Shallow Solver for P1

Approx Black-Box for Utilitarian + Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

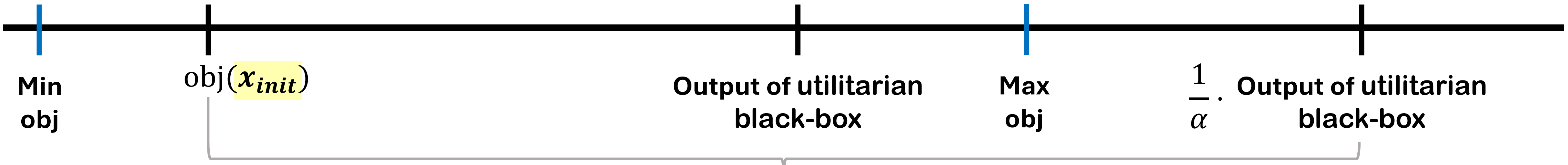
(a) Approx. black-box for utilitarian
(b) An arbitrary $x_{init} \in F(\mathrm{P1})$ $\Rightarrow$ A shallow-solver for P1
(c) Weak Feasibility-Oracle for P1

Weak Feasibility-Oracle for P1

Given $z_t \in \mathbb{R}_{\geq 0}$ returns one of:

- Feasible objective
- Infeasible objective for sol. using $\leq \alpha$ prob.

1. Bound the maximum objective value



2. Perform a binary search

For each z_t, use the Weak Feasibility-Oracle:

- If 'Feasible' go right (higher values)
- otherwise go left (lower values)

An α-Shallow Solver for P1

Approx Black-Box for Utilitarian + Weak Feasibility-Oracle for P1 $\Rightarrow$ Shallow Solver for P1

Lemma 6.1

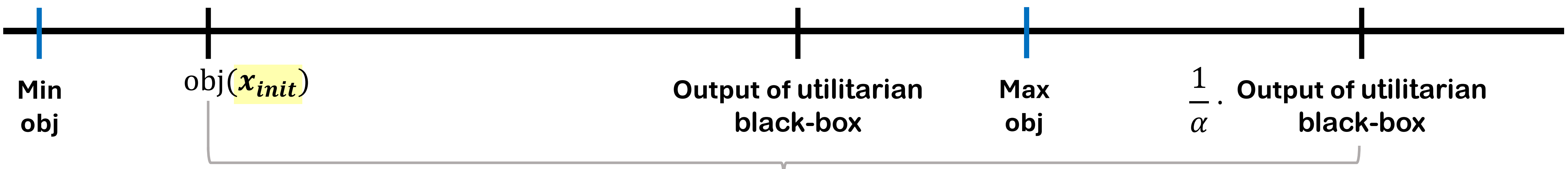
(a) Approx. black-box for utilitarian
(b) An arbitrary $x_{init} \in F(\mathrm{P1})$ $\Rightarrow$ A shallow-solver for P1
(c) Weak Feasibility-Oracle for P1

Weak Feasibility-Oracle for P1

Given $z_t \in \mathbb{R}_{\geq 0}$ returns one of:

- Feasible objective
- Infeasible objective for sol. using $\leq \alpha$ prob.

1. Bound the maximum objective value



2. Perform a binary search

For each z_t, use the Weak Feasibility-Oracle:

- If 'Feasible' go right (higher values)
- otherwise go left (lower values)

3. Return the largest value for which the answer is 'Feasible'.
   ➔ It is maximum w.r.t. all solutions that use $\leq \alpha$ probability.

An α-Weak Feasibility-Oracle for P1

Approx Opt-Solver for P2 $\Rightarrow$ Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for P2 $\Rightarrow$ α-Weak Feasibility-Oracle for P1

P2 is derived from P1 and a candidate objective-value z_t :

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i & & \text{(P1)} \\
s.t. \quad & \text{(P1.1)} \ \sum_{j=1}^{|S|} x_j = 1 & & \\
& \text{(P1.2)} \ x_j \geq 0 & j = 1, \ldots, |S| & \\
& \text{(P1.3)} \ \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i & \forall \ell \in [t-1] &
\end{aligned}
$$

An α-Weak Feasibility-Oracle for P1

Approx Opt-Solver for P2	$\Longrightarrow$	Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$**-Approx-Optimal Solver for P2** $\Rightarrow$ α**-Weak Feasibility-Oracle for P1**

P2 is derived from P1 and a <u>candidate</u> objective-value z_t :

1. **Remove the objective function**

$$\cancel{\max \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i} \qquad \text{(P1)}$$

$$\begin{aligned}
s.t. \quad & \text{(P1.1)} \quad \sum_{j=1}^{|S|} x_j = 1 \\
& \text{(P1.2)} \quad x_j \geq 0 \qquad\qquad j = 1, \ldots, |S| \\
& \text{(P1.3)} \quad \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i \qquad \forall \ell \in [t-1]
\end{aligned}$$

Approx Opt-Solver for P2 $\Longrightarrow$ Weak Feasibility-Oracle for P1

An α-Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for P2 $\Rightarrow$ α-Weak Feasibility-Oracle for P1

P2 is derived from P1 and a <u>candidate</u> objective-value z_t :

1. **Remove the objective function**
2. **Add a constraint to ensure that the objective $\geq z_t$**

$$\cancel{\max \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i} \qquad \text{(P1)}$$

$$
\begin{aligned}
s.t. \quad & \text{(P1.1)} \quad \sum_{j=1}^{|S|} x_j = 1 \\
& \text{(P1.2)} \quad x_j \geq 0 && j = 1, \ldots, |S| \\
& \text{(P1.3)} \quad \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i && \forall \ell \in [t-1] \\
& \boxed{\sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i \geq z_t}
\end{aligned}
$$

An α-Weak Feasibility-Oracle for P1

Approx Opt-Solver for P2 $\Rightarrow$ Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for P2 $\Rightarrow$ **α-Weak Feasibility-Oracle for P1**

P2 is derived from P1 and a <u>candidate</u> objective-value z_t :

1. **Remove the objective function**
2. **Add a constraint to ensure that the objective $\geq z_t$**
3. **Remove Constraint (P1.1)**

$$\cancel{\max \sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i} \qquad \text{(P1)}$$

$$s.t. \quad \cancel{\text{(P1.1)} \sum_{j=1}^{|S|} x_j = 1}$$

$$\text{(P1.2)} \quad x_j \geq 0 \qquad j = 1, \ldots, |S|$$

$$\text{(P1.3)} \quad \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i \qquad \forall \ell \in [t-1]$$

$$\boxed{\sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i \geq z_t}$$

An α-Weak Feasibility-Oracle for P1

Approx Opt-Solver for P2 $\Rightarrow$ Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for P2 $\Rightarrow$ α-Weak Feasibility-Oracle for P1

P2 is derived from P1 and a <u>candidate</u> objective-value z_t :

1. Remove the objective function
2. Add a constraint to ensure that the objective $\geq z_t$
3. Remove Constraint (P1.1)
4. Add new obj: minimize the sum of probabilities

$$\boxed{min \quad \sum_{j=1}^{|S|} x_j} \qquad \text{(P1)}$$

$$s.t. \quad \cancel{\text{(P1.1)} \quad \sum_{j=1}^{|S|} x_j = 1}$$

$$\text{(P1.2)} \quad x_j \geq 0 \qquad j = 1, \ldots, |S|$$

$$\text{(P1.3)} \quad \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i \qquad \forall \ell \in [t-1]$$

$$\boxed{\sum_{i=1}^{t} E_i^{\uparrow}(\mathbf{x}) - \sum_{i=1}^{t-1} z_i \geq z_t}$$

An α-Weak Feasibility-Oracle for P1

Approx Opt-Solver for P2	$\Rightarrow$	Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for P2 $\Rightarrow$ α-Weak Feasibility-Oracle for P1

P2 is derived from P1 and a <u>candidate</u> objective-value z_t :

1. **Remove the objective function**
2. **Add a constraint to ensure that the objective $\geq z_t$**
3. **Remove Constraint (P1.1)**
4. **Add new obj: min the sum over x_j**

$$\min \quad \sum_{j=1}^{|S|} x_j \quad s.t. \tag{P2}$$

$$\text{(P2.1)} \quad x_j \geq 0 \qquad j = 1, \ldots, |S|$$

$$\text{(P2.2)} \quad \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i \qquad \forall \ell \in [t]$$

An α-Weak Feasibility-Oracle for P1

Approx Opt-Solver for P2 $\Rightarrow$ Weak Feasibility-Oracle for P1

Lemma 7.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for P2 $\Rightarrow$ α-Weak Feasibility-Oracle for P1

α-Weak Feasibility-Oracle for P1

Given $\frac{1}{\alpha}$-Approx. solution to P2, $\boldsymbol{x}^A$:

- If its objective value ≤ 1:
  Assert that z_t is feasible, return $\boldsymbol{x}^A$.
- Otherwise:
  Assert that z_t is infeasible for solutions using $\leq \alpha$ probability.

$$\min \sum_{j=1}^{|S|} x_j \quad s.t. \qquad \text{(P2)}$$

$$\text{(P2.1)} \quad x_j \geq 0 \qquad j = 1, \ldots, |S|$$

$$\text{(P2.2)} \quad \sum_{i=1}^{\ell} E_i^{\uparrow}(\mathbf{x}) \geq \sum_{i=1}^{\ell} z_i \qquad \forall \ell \in [t]$$

Weak Feasibility-Oracle for P1

Given $z_t \in \mathbb{R}_{\geq 0}$ returns one of:

- Feasible objective
- Infeasible objective for sol. using $\leq \alpha$ prob.

An $\frac{1}{\alpha}$-Approx-Optimal Solver for P2

Approx Opt-Solver for linear P3 $\Rightarrow$ Approx Opt-Solver for P2

Lemma 8.1

$\frac{1}{\alpha}$-Approx-Optimal Solver for linear P3 $\Rightarrow$ $\frac{1}{\alpha}$-Approx-Optimal Solver for P2

$$
\begin{array}{lll}
\min & \sum_{j=1}^{|S|} x_j \quad s.t. & \text{(P3)} \\
\text{(P3.1)} & x_j \geq 0 & j = 1, \ldots, |S| \\
\text{(P3.2)} & \ell y_\ell - \sum_{i=1}^{n} m_{\ell,i} \geq \sum_{i=1}^{\ell} z_i & \forall \ell \in [t] \\
\text{(P3.3)} & m_{\ell,i} \geq y_\ell - \sum_{j=1}^{|S|} x_j \cdot u_i(s_j) & \forall \ell \in [t], \ \forall i \in [n] \\
\text{(P3.4)} & m_{\ell,i} \geq 0 & \forall \ell \in [t], \ \forall i \in [n]
\end{array}
$$

- A Linear Program which is Equivalent to P2:
  - x^A is a solution for P2 $\Leftrightarrow$ it is part of a solution for P3
  - Auxiliary variables are used to “linearize” the constraints
  - (technique based on Ogryczak and Sliwinski)

- **P3 Cannot be solved directly: exp. number of variables.**

On Direct Methods for Lexicographic Min-Max Optimization. Ogryczak and Sliwinski, 2006.

An $\frac{1}{\alpha}$-Approx-Optimal Solver for P3

Approx Black-Box for Utilitarian $\Rightarrow$ Approx Opt-Solver for linear P3

Corollary 9.1 + Lemma 10.1

α-Black-box for Utilitarian $\Rightarrow$ $\frac{1}{\alpha}$-Approx-Optimal Solver for P3

$$
\begin{array}{llr}
\min \quad \sum_{j=1}^{|S|} x_j \quad s.t. & & \text{(P3)} \\
\text{(P3.1)} \quad x_j \geq 0 & j = 1, \ldots, |S| & \\
\text{(P3.2)} \quad \ell y_\ell - \sum_{i=1}^{n} m_{\ell,i} \geq \sum_{i=1}^{\ell} z_i & \forall \ell \in [t] & \\
\text{(P3.3)} \quad m_{\ell,i} \geq y_\ell - \sum_{j=1}^{|S|} x_j \cdot u_i(s_j) & \forall \ell \in [t], \ \ \forall i \in [n] & \\
\text{(P3.4)} \quad m_{\ell,i} \geq 0 & \forall \ell \in [t], \ \ \forall i \in [n] &
\end{array}
$$

Dual $\longrightarrow$

$$
\begin{array}{llr}
\max \quad \sum_{\ell=1}^{t} q_\ell \sum_{i=1}^{\ell} z_i \quad s.t. & & \text{(D3)} \\
\text{(D3.1)} \quad \sum_{i=1}^{n} u_i(s_j) \sum_{\ell=1}^{t} v_{\ell,i} \leq 1 & \forall j = 1, \ldots, |S| & \\
\text{(D3.2)} \quad \ell q_\ell - \sum_{i=1}^{n} v_{\ell,i} \leq 0 & \forall \ell \in [t] & \\
\text{(D3.3)} \quad - q_\ell + v_{\ell,i} \leq 0 & \forall \ell \in [t], \ \ \forall i \in [n] & \\
\text{(D3.4)} \quad q_\ell \geq 0 & \forall \ell \in [t] & \\
\text{(D3.5)} \quad v_{\ell,i} \geq 0 & \forall \ell \in [t], \ \ \forall i \in [n] &
\end{array}
$$

- **We approximate P3 with its Dual Program and a variant of the ellipsoid method (for approx)**
- **The utilitarian solver is used inside the (approx) separation oracle for the Dual.**

An efficient approximation scheme for the one-dimensional bin-packing problem. Karmarkar, and Karp, 1982.

An $\frac{1}{\alpha}$-Approx-Optimal Solver for P3

Approx Black-Box for Utilitarian $\Rightarrow$ Approx Opt-Solver for linear P3

Corollary 9.1 + Lemma 10.1

α-Black-box for Utilitarian $\Rightarrow$ $\frac{1}{\alpha}$-Approx-Optimal Solver for P3

$$
\begin{array}{lll}
\min & \sum_{j=1}^{|S|} x_j \quad s.t. & \text{(P3)} \\
\text{(P3.1)} & x_j \geq 0 & j = 1, \ldots, |S| \\
\text{(P3.2)} & \ell y_\ell - \sum_{i=1}^{n} m_{\ell,i} \geq \sum_{i=1}^{\ell} z_i & \forall \ell \in [t] \\
\text{(P3.3)} & m_{\ell,i} \geq y_\ell - \sum_{j=1}^{|S|} x_j \cdot u_i(s_j) & \forall \ell \in [t], \ \forall i \in [n] \\
\text{(P3.4)} & m_{\ell,i} \geq 0 & \forall \ell \in [t], \ \forall i \in [n]
\end{array}
$$

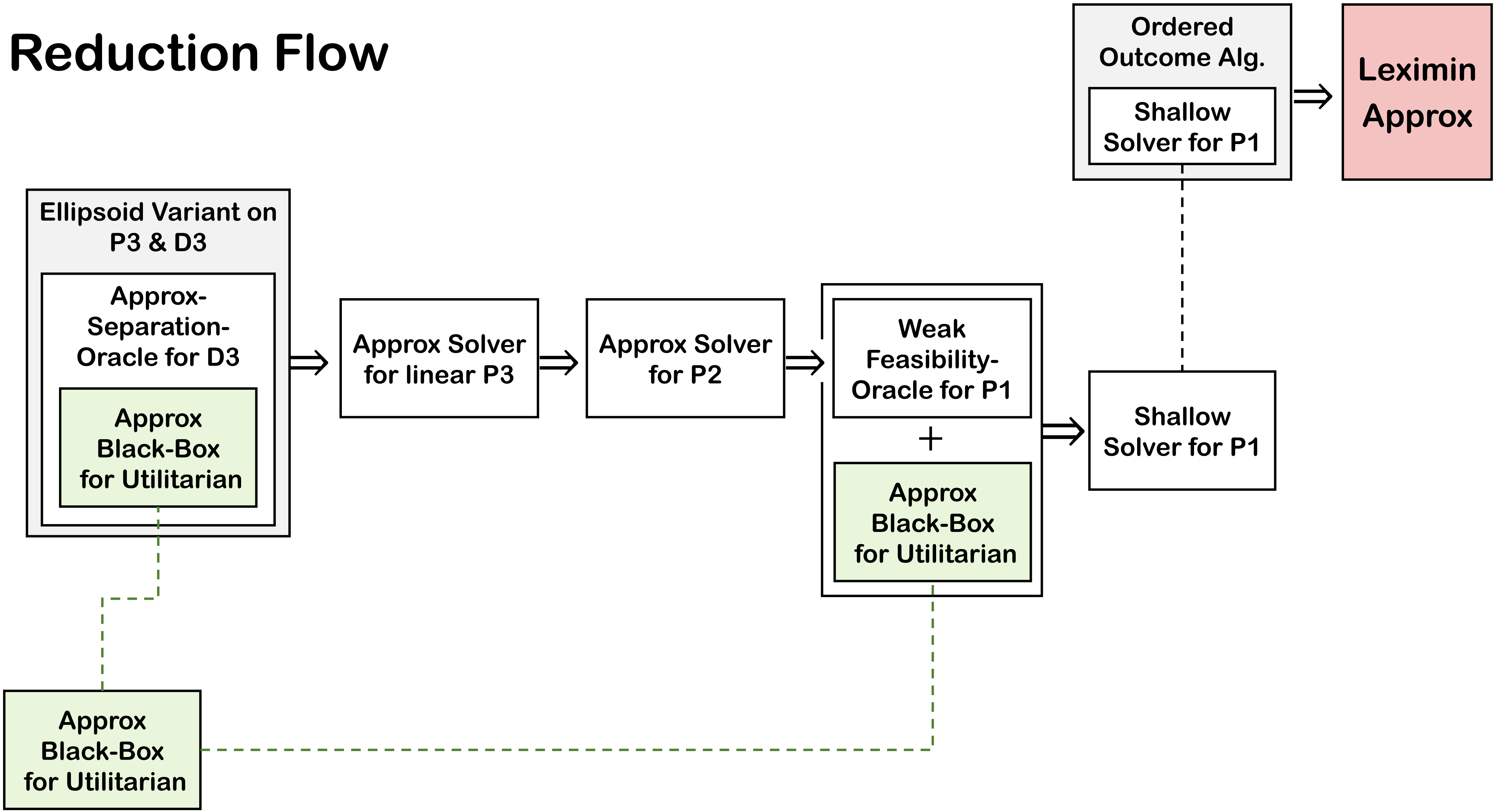
Dual $\longrightarrow$

$$
\begin{array}{lll}
\max & \sum_{\ell=1}^{t} q_\ell \sum_{i=1}^{\ell} z_i \quad s.t. & \text{(D3)} \\
\text{(D3.1)} & \sum_{i=1}^{n} u_i(s_j) \overbrace{\boxed{\sum_{\ell=1}^{t} v_{\ell,i}}}^{c_i} \leq 1 & \forall j = 1, \ldots, |S| \\
\text{(D3.2)} & \ell q_\ell - \sum_{i=1}^{n} v_{\ell,i} \leq 0 & \forall \ell \in [t] \\
\text{(D3.3)} & -q_\ell + v_{\ell,i} \leq 0 & \forall \ell \in [t], \ \forall i \in [n] \\
\text{(D3.4)} & q_\ell \geq 0 & \forall \ell \in [t] \\
\text{(D3.5)} & v_{\ell,i} \geq 0 & \forall \ell \in [t], \ \forall i \in [n]
\end{array}
$$

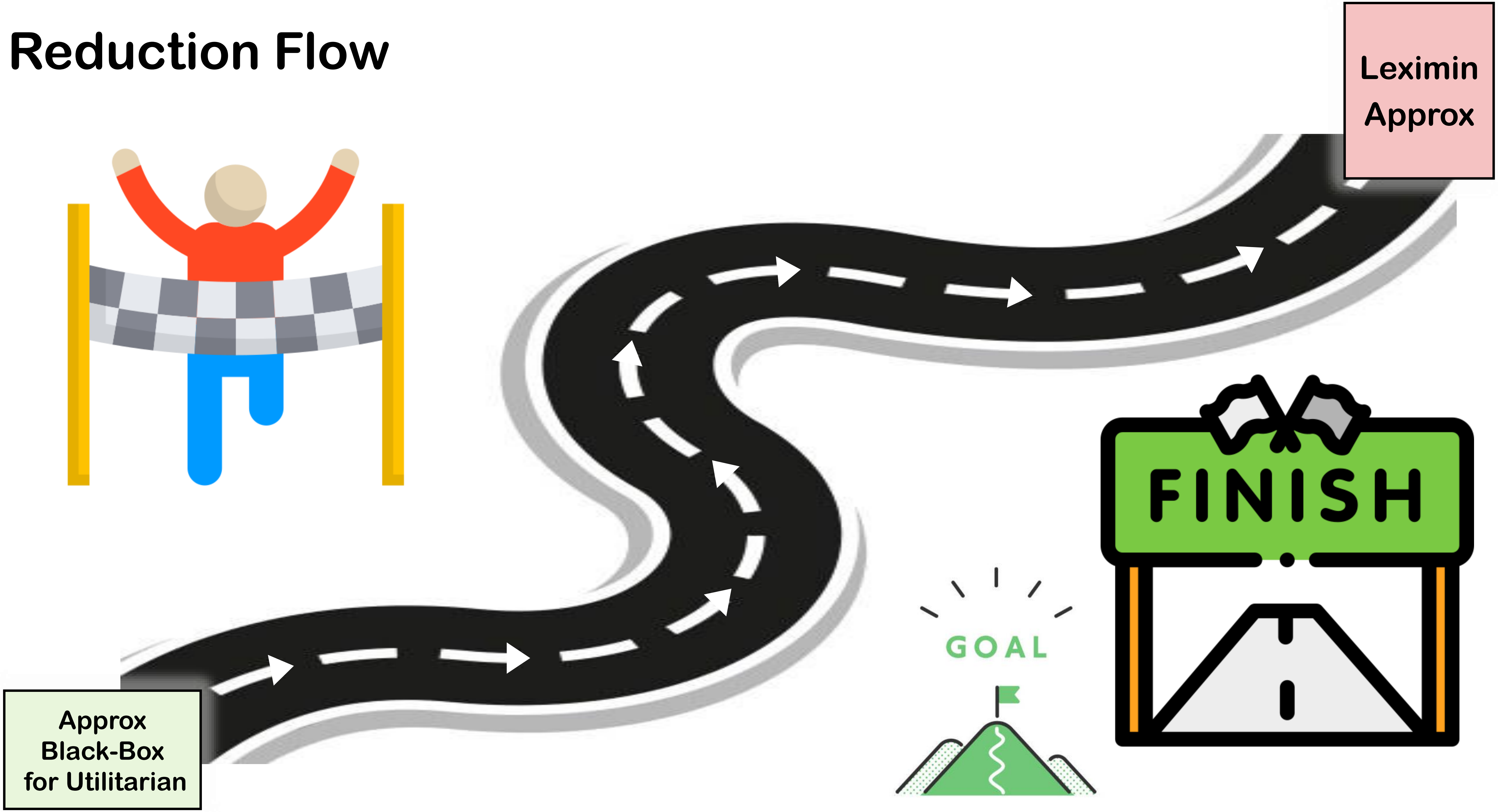
- **We approximate P3 with its Dual Program and a variant of the ellipsoid method (for approx)**
- **The utilitarian solver is used inside the (approx) separation oracle for the Dual.**

An efficient approximation scheme for the one-dimensional bin-packing problem. Karmarkar, and Karp, 1982.

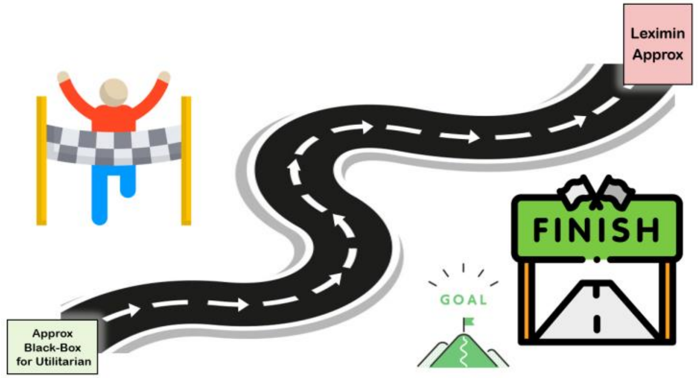
Reduction Flow



Reduction Flow



Open Questions



1. **Social choice problems with utilities ≤ 0**
   (e.g. fair allocation of chores, facility location)

2. **Nash welfare objective** = Maximize the product of utilities.
   - Sweet-spot between utilitarian and egalitarian.
   - Often as hard as egalitarian.
   - Is there an analogous reduction for Nash welfare?

3. **Best of both worlds:** Can we guarantee any ex-post fairness?

Please contact us for any question and any answer!

- erelsgl@gmail.com
- eden.r.hartman@gmail.com