Discrete Cost Uncertainty

Robust Optimization

Stefan Lendl

Institute of Discrete Mathematics
Graz University of Technology

Example: Investments

Example: [Assi et al.; 2009] $\max_{x\in\mathcal{X}} \min_{p\in\mathcal{U}} \sum_{i=1}^{n} p_i x_i$

Input:

b bound on total investment
n investment opportunities
w_i required cash for investment i
p_i profit from investment i (uncertain)

Example: Investments

Example: [Assi et al.; 2009] $\max_{x \in \mathcal{X}} \min_{p \in \mathcal{U}} \sum_{i=1}^{n} p_i x_i$

Input:

b bound on total investment

n investment opportunities

w_i required cash for investment i

p_i profit from investment i (uncertain)

Cash outflows and profits of the investments

i	w_i	p_i^1	p_i^2	p_i^3
1	3	4	3	3
2	5	8	4	6
3	2	5	3	3
4	4	3	2	4
5	5	2	8	2
6	3	4	6	2

$b = 12$

Example: Investments

Example: [Assi et al.; 2009] $\max_{x\in\mathcal{X}} \min_{p\in\mathcal{U}} \sum_{i=1}^{n} p_i x_i$

Input:

b bound on total investment

n investment opportunities

w_i required cash for investment i

p_i profit from investment i (uncertain)

Cash outflows and profits of the investments

i	w_i	p_i^1	p_i^2	p_i^3
1	3	4	3	3
2	5	8	4	6
3	2	5	3	3
4	4	3	2	4
5	5	2	8	2
6	3	4	6	2

$b = 12$

Solutions:

Optimal values and solutions (discrete scenario case)

1	2	3	Optimal solution
17	10	12	Scenario 1: (1,1,1,0,0,0)
11	17	7	Scenario 2: (0,0,1,0,1,1)
16	9	13	Scenario 3: (0,1,1,1,0,0)
15	12	12	Max–min: (0,1,0,1,0,1)

Computational Complexity

Theorem

MIN-MAX *robust optimization is NP-hard for* $\mathcal{X} = \{0,1\}^n$ *if the uncertainty set* $\mathcal{U}$ *contains exactly* $K = 2$ *scenarios.*

Proof.

Reduction from SUBSET SUM. $\square$

K-Approximation

Theorem

Consider an instance of $\textsc{Min-Max}$ *robust optimization for given* $\mathcal{X} \subseteq \{0,1\}^n$ *with discrete uncertainty set* $\mathcal{U}$ *containing exactly* K *scenarios. Consider also an instance of the classic* $\textsc{Min}$ *problem with linear costs* $c_i' = \sum_{s=1}^{k} \frac{c_i^s}{k}$ *for each* $i = 1, \ldots, n$ *and let* x' *be an optimal solution to this instance. It then holds that*

$$\max_{c \in \mathcal{U}} c(x') \leq k \cdot \min_{x \in \mathcal{X}} \max_{c \in \mathcal{U}} c(x)$$

K-Approximation

Theorem

Consider an instance of $\textsc{Min-Max}$ *robust optimization for given* $\mathcal{X} \subseteq \{0,1\}^n$ *with discrete uncertainty set* $\mathcal{U}$ *containing exactly* K *scenarios. Consider also an instance of the classic* $\textsc{Min}$ *problem with linear costs* $c'_i = \sum_{s=1}^{k} \frac{c_i^s}{k}$ *for each* $i = 1, \ldots, n$ *and let* x' *be an optimal solution to this instance. It then holds that*

$$\max_{c \in \mathcal{U}} c(x') \leq k \cdot \min_{x \in \mathcal{X}} \max_{c \in \mathcal{U}} c(x)$$

Same holds also for $c'_i = \max_{c \in \mathcal{U}} c_i$. (Exercise!)

Strong NP-hardness

Theorem

MIN-MAX *robust optimization is strongly NP-hard for* $\mathcal{X} = \{0,1\}^n$.

Proof.

Reduction from SET COVER. $\square$

Algorithm for constant K

Main tool: EXACT problem: Decide if there exists a solution with objective value exactly v!

Theorem (Assi, Bazgun, Vanderpooten; 2005)

There is a pseudo-polynomial algorithm for MIN-MAX *robust optimization with discrete uncertainties $\mathcal{U}$ and $|\mathcal{U}| = K$ is constant if the* EXACT *problem and ists search version are polynomially or pseudo-polynomially solvable.*

Hence the problem is not strongly NP-hard for constant K.

Algorithm for constant K

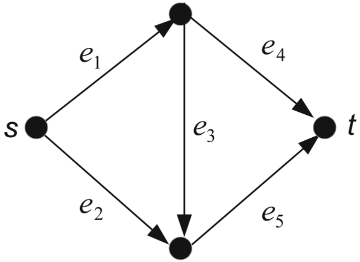
$v \leftarrow 0$
$test \leftarrow false$
while $test \neq true$ **do**
  for $(\alpha_1, \ldots, \alpha_k)$: $\max\{\alpha_1, \ldots, \alpha_k\} = v$ **do**
    $value \leftarrow \sum_{p=1}^{k} \alpha_p (mM+1)^{p-1}$
    if DECIDE_EXACT_$\mathcal{P}$(I',$value$) **then**
      CONSTRUCT_EXACT_$\mathcal{P}$ (I',$value$,x^*)
      $test \leftarrow true$
  if $test \neq true$ **then**
    $v \leftarrow v + 1$
$opt \leftarrow v$
Output opt and x^*

Example: Shortest Path

Example: [Kasperski, Zielinski; 2016]

$$\min_{X \in \mathcal{F}} \max_{c \in \mathcal{U}} c(X)$$

$$c^{s_1} = (2, 10, 3, 1, 1), c^{s_2} = (1, 11, 0, 5, 1), c^{s_3} = (8, 8, 0, 8, 8, 8)$$

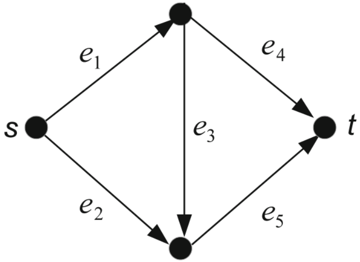


Example: Shortest Path

Example: [Kasperski, Zielinski; 2016]

$$\min_{X \in \mathcal{F}} \max_{c \in \mathcal{U}} c(X)$$

$$c^{s_1} = (2, 10, 3, 1, 1), c^{s_2} = (1, 11, 0, 5, 1), c^{s_3} = (8, 8, 0, 8, 8, 8)$$



	c^{s_1}	c^{s_2}	c^{s_3}
$\{e_1, e_4\}$	3	6	16
$\{e_1, e_3, e_5\}$	6	2	16
$\{e_2, e_5\}$	11	12	16

Connection to multi-objective optimization

Efficient solutions $\mathcal{E}$: A solution x is efficient for $\mathcal{U}$ if no other solution dominates it. A solution x dominates y if $c(x) \leq c(y)$ for all $c \in \mathcal{U}$, with at least one inequality being strict.

Connection to multi-objective optimization

Efficient solutions $\mathcal{E}$: A solution x is efficient for $\mathcal{U}$ if no other solution dominates it. A solution x dominates y if $c(x) \leq c(y)$ for all $c \in \mathcal{U}$, with at least one inequality being strict.

Theorem

At least one solution of MIN-MAX *robust optimization for $\mathcal{X} \subseteq \{0,1\}^n$ is neccessarily an efficient solution.*

Connection to multi-objective optimization

Efficient solutions $\mathcal{E}$: A solution x is efficient for $\mathcal{U}$ if no other solution dominates it. A solution x dominates y if $c(x) \leq c(y)$ for all $c \in \mathcal{U}$, with at least one inequality being strict.

Theorem

At least one solution of MIN-MAX *robust optimization for $\mathcal{X} \subseteq \{0,1\}^n$ is neccessarily an efficient solution.*

Proof.

If x dominates y then $\max_{c \in \mathcal{U}} c(x) \leq \max_{c \in \mathcal{U}} c(y)$.

Connection to multi-objective optimization

Efficient solutions $\mathcal{E}$: A solution x is efficient for $\mathcal{U}$ if no other solution dominates it. A solution x dominates y if $c(x) \leq c(y)$ for all $c \in \mathcal{U}$, with at least one inequality being strict.

Theorem

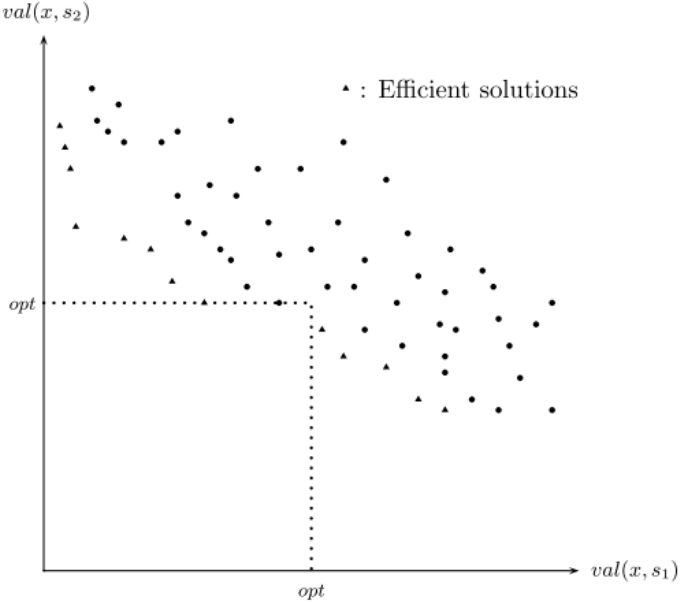
At least one solution of MIN-MAX *robust optimization for* $\mathcal{X} \subseteq \{0,1\}^n$ *is neccessarily an efficient solution.*

Proof.

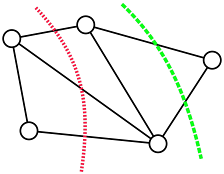
If x dominates y then $\max_{c\in\mathcal{U}} c(x) \leq \max_{c\in\mathcal{U}} c(y)$.
Take $x \in \mathcal{E}$ with minimum $\max_{c\in\mathcal{U}} c(x)$.

$\square$

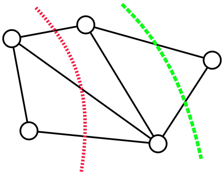
Connection to multi-objective optimization



Robust Global Minimum Cut



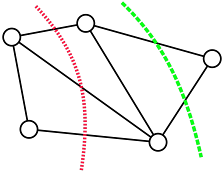
Robust Global Minimum Cut



Algorithm Min-Max($G(V, E, w_1, \ldots, w_k)$):

1. Let $w'(e) = \sum_{i=1}^{k} w_i(e)$, for every $e \in E$.
2. Find all the k-approximate minimum cuts in G with respect to w'.
3. Among all the cuts C found in the previous step find the one for which $\max_{i=1}^{k} w_i(C)$ is minimized.

Robust Global Minimum Cut



Algorithm Min-Max($G(V, E, w_1, \ldots, w_k)$):

1. Let $w'(e) = \sum_{i=1}^{k} w_i(e)$, for every $e \in E$.
2. Find all the k-approximate minimum cuts in G with respect to w'.
3. Among all the cuts C found in the previous step find the one for which $\max_{i=1}^{k} w_i(C)$ is minimized.

Lemma

If C is an optimal $\textsc{Min-Max}$ *cut and D is any other cut in the graph then*

$$w'(C) \leq Kw'(D).$$

Karger's Contraction Algorithm

Procedure Contract(G)

repeat until G has 2 vertices

  choose an edge (v, w) uniformly at random from G

  let $G \leftarrow G/(v, w)$

return G

Lemma

A cut (A, B) is output by the Contraction Algorithm if and only if no edge crossing (A, B) is contracted by the algorithm.

Theorem

A particular minimum cut in G is returned by the Contraction Algorithm with probability at least $\binom{n}{2}^{-1} = \Omega(n^{-2})$.

Karger-Stein: k-Approximate Minimum Cuts

Lemma

The probability that a particular k-approximate minimum cut survives contraction to $2k$ vertices is $\Omega(\binom{n}{2k}^{-1})$.

Theorem

The number of k-approximate minimum cuts is $O((2n)^{2k})$ and they can be found in randomized polynomial time.

Overview: Complexity

Problems	Constant
	Min–max
SHORTEST PATH	*NP*-hard, pseudo-poly [65]
SPANNING TREE	*NP*-hard [43], pseudo-poly [3]
ASSIGNMENT	*NP*-hard [43]
KNAPSACK	*NP*-hard, pseudo-poly [43]
MIN CUT	Polynomial [7]
MIN s–t CUT	strongly *NP*-hard [2]

Problems	Non-constant
	Min–max
SHORTEST PATH	Strongly *NP*-hard [43]
SPANNING TREE	Strongly *NP*-hard [43]
ASSIGNMENT	Strongly *NP*-hard [1]
KNAPSACK	Strongly *NP*-hard [43]
MIN CUT	Strongly *NP*-hard [2]
MIN s–t CUT	Strongly *NP*-hard [2]

A general approximation scheme

For constant K there is an FPTAS computing the efficient solutions $\mathcal{E}$.

Theorem

If for any instance I of a robust $\textsc{Min-Max}$ *robust optimization problem on $\mathcal{X}$ with discrete uncertainty $\mathcal{U}$ it holds that*

1. *a lower and upper bound L and U for the optimal value can be computed in time $p(|I|)$ such that $U \leq q(|I|)L$, where p and q are two polynomials with q non-decreasing and $q(|I|) \geq 1$, and*
2. *there exists and algorithm that finds for I an optimal solution $r(|I|, U)$, where r is a non-decresing polynomial,*

then there exists an FPTAS.

A general approximation scheme

For constant K there is an FPTAS computing the efficient solutions $\mathcal{E}$.

Theorem

If for any instance I of a robust Min-Max *robust optimization problem on $\mathcal{X}$ with discrete uncertainty $\mathcal{U}$ it holds that*

1. *a lower and upper bound L and U for the optimal value can be computed in time $p(|I|)$ such that $U \leq q(|I|)L$, where p and q are two polynomials with q non-decreasing and $q(|I|) \geq 1$, and*
2. *there exists and algorithm that finds for I an optimal solution $r(|I|, U)$, where r is a non-decreasing polynomial,*

then there exists an FPTAS.

When do these conditions hold?

Proposition

If a minimization problem is solvable in polynomial time, then for any instance on a set of k scenarios of Min-Max *robust optimization, there exists a lower and an upper bound L and U computable in polynomial time such that $U \leq kL$.*

When do these conditions hold?

Proposition

If a minimization problem is solvable in polynomial time, then for any instance on a set of k scenarios of Min-Max *robust optimization, there exists a lower and an upper bound L and U computable in polynomial time such that $U \leq kL$.*

Proposition

The second condition can be weakend to require an algorithm that is polynomial in $|I|$ and $\max(I) = \max_{c \in \mathcal{U}, i} c_i$

When do these conditions hold?

Proposition

If a minimization problem is solvable in polynomial time, then for any instance on a set of k scenarios of Min-Max *robust optimization, there exists a lower and an upper bound L and U computable in polynomial time such that $U \leq kL$.*

Proposition

The second condition can be weakend to require an algorithm that is polynomial in $|I|$ and $\max(I) = \max_{c \in \mathcal{U}, i} c_i$

Proof.

If there exists $c \in \mathcal{U}$ such that $c(i) > U$, then $x_i = 0$. □

Overview: Approximation

Problems	Constant	Non-constant
	Min–max	Min–max
SHORTEST PATH	fptas [61]	Not $\log^{1-\varepsilon} k$ approx. [40]
SPANNING TREE	fptas [4,3]	Not $\log^{1-\varepsilon} k$ approx. [40]
KNAPSACK	fptas [4]	Non approx. [4]

Thank you!