The robust knapsack problem with queries

Felix Hauser

12.06.2024

Problem

$W = 10$

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

Problem

$W = 10$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:

Selbst im Worst-case zulässig

Problem

$W = 10$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:

Selbst im Worst-case zulässig

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	6	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	6	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	5	6	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	$\boxed{3}$	$\boxed{6}$	5	9	$\boxed{4}$
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	$\boxed{5}$	$\boxed{6}$	?

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	$\boxed{3}$	$\boxed{6}$	5	9	$\boxed{4}$
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	$\boxed{5}$	$\boxed{6}$	?

Lösung: $\{1, 4\}$ Gesamtwert: 14

$\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Robuste Lösungen:
Selbst im Worst-case zulässig

Q Abfragen erlaubt,
die das exakte Gewicht $\hat{w}_i$ liefern.

Problem - Definition

- Gewichte in Unsicherheitsmenge U unsicher:

$$U = \left[\underline{w}_1, \overline{w}_1\right] \times \ldots \times \left[\underline{w}_n, \overline{w}_n\right]$$

- Bis zu Q Abfragen liefern $\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Zulässige Lösungen:

- Selbst wenn der Worst-case eintritt, darf das Maximalgewicht nicht überschritten werden

Ziel: Möglichst hoher Gesamtwert

Problem - Definition

- Gewichte in Unsicherheitsmenge U unsicher:

$$U = \left[\underline{w}_1, \overline{w}_1\right] \times ... \times \left[\underline{w}_n, \overline{w}_n\right]$$

- Bis zu Q Abfragen liefern $\hat{w}_i \in \left[\underline{w}_i, \overline{w}_i\right]$

Zulässige Lösungen:

- Selbst wenn der Worst-case eintritt, darf das Maximalgewicht nicht überschritten werden

Ziel: Möglichst hoher Gesamtwert

Bewertung eines Algorithmus?

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	3	5	6	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	3	5	6	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	3	5	6	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	3	5	6	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	3	5	6	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	$\boxed{3}$	6	$\boxed{5}$	9	$\boxed{4}$
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	$\boxed{3}$	5	$\boxed{6}$	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Full-knowledge-Problem

$W = 10$
$Q = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	$\boxed{3}$	6	$\boxed{5}$	9	$\boxed{4}$
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	2	$\boxed{3}$	5	$\boxed{6}$	3

$\hat{w}_i$ ist bekannt für alle $i \in \{1, ..., n\}$

$\hat{w}_i$ kann nur für das Packen benutzt werden, falls i abgefragt wurde

Lösung: $\{2, 4\}$ Gesamtwert: 15

Bewertungskriterium

Sei $F^*(I, Q)$ die Optimale Lösung des Full-knowledge Problems auf Instanz I mit Q Abfragen.

Bewertungskriterium

Sei $F^*(I, Q)$ die Optimale Lösung des Full-knowledge Problems auf Instanz I mit Q Abfragen.

Bewertung auf Instanz I mit Q Abfragen:

Ein Algorithmus A ist α-*(query) competitive* falls

$$\frac{F^*(I, Q)}{A(I, Q)} \leq \alpha$$

Bewertungskriterium

Sei $F^*(I, Q)$ die Optimale Lösung des Full-knowledge Problems auf Instanz I mit Q Abfragen.

Bewertung auf Instanz I mit Q Abfragen:

Ein Algorithmus A ist α-*(query) competitive* falls

$$\frac{F^*(I, Q)}{A(I, Q)} \leq \alpha$$

Im Beispiel: $\frac{15}{14}$

Bewertungskriterium

Auf einer Menge $\mathcal{J}$ von Instanzen und Q Anfragen:

Ein deterministischer Algorithmus A ist α-*worst-case (query) competitive* falls:

$$\max_{I \in \mathcal{J}} \frac{F^*(I,Q)}{A(I,Q)} \leq \alpha$$

Bewertungskriterium

Auf einer Menge $\mathcal{J}$ von Instanzen und Q Anfragen:

Ein deterministischer Algorithmus A ist *α-worst-case (query) competitive* falls:

$$\max_{I \in \mathcal{J}} \frac{F^*(I, Q)}{A(I, Q)} \leq \alpha$$

Annahme: $\overline{w}_i \leq W, \underline{w}_i \geq 1 \quad \forall i$

Algorithmen Aufbau

Ein Algorithmus besteht aus zwei Teilen:

Algorithmen Aufbau

Ein Algorithmus besteht aus zwei Teilen:

1. Teil: Entscheide von welchen Gegenständen das Gewicht abgefragt werden soll.

Algorithmen Aufbau

Ein Algorithmus besteht aus zwei Teilen:

1. Teil: Entscheide von welchen Gegenständen das Gewicht abgefragt werden soll.

2. Teil: Packe Gegenstände, sodass eine zulässige Lösung mit maximalem Wert entsteht.

Algorithmen Aufbau

Ein Algorithmus besteht aus zwei Teilen:

1. Teil: Entscheide von welchen Gegenständen das Gewicht abgefragt werden soll.

2. Teil: Packe Gegenstände, sodass eine zulässige Lösung mit maximalem Wert entsteht.

- Löse das Knapsack Problem mit Gewichten

$$w_i = \begin{cases} \widehat{w}_i & \text{falls das Gewicht von } i \text{ abgefragt wurde} \\ \overline{w}_i & \text{sonst} \end{cases}$$

Algorithmen

ALG_1:

- Frage keine Gegenstände ab
- Löse den zweiten Teil

Algorithmen

ALG_1:

- Frage keine Gegenstände ab
- Löse den zweiten Teil

ALG_2:

- Frage die ersten Q Gegenstände ab
- Löse den zweiten Teil

ALG_1 ist W-worst-case query competitive

Sei I, Q beliebig.

ALG_1 ist W-worst-case query competitive

Sei I, Q beliebig.

Sei $i \in \{1, ..., n\}$ der Gegenstand mit maximalen Wert.

ALG_1 ist W-worst-case query competitive

Sei I, Q beliebig.
Sei $i \in \{1, ..., n\}$ der Gegenstand mit maximalen Wert.
Es gilt $F^*(I, Q) \leq W \cdot v_i$, da $\underline{w}_i \geq 1$.

ALG_1 ist W-worst-case query competitive

Sei I, Q beliebig.
Sei $i \in \{1, ..., n\}$ der Gegenstand mit maximalen Wert.
Es gilt $F^*(I, Q) \leq W \cdot v_i$, da $\underline{w}_i \geq 1$.
ALG_1 kann i einpacken, da $\overline{w}_i \leq W$.

ALG_1 ist W-worst-case query competitive

Sei I, Q beliebig.
Sei $i \in \{1, ..., n\}$ der Gegenstand mit maximalen Wert.
Es gilt $F^*(I, Q) \leq W \cdot v_i$, da $\underline{w}_i \geq 1$.
ALG_1 kann i einpacken, da $\overline{w}_i \leq W$.
Also ist $\text{ALG}_1(I, Q) \geq v_i$.

ALG_1 ist W-worst-case query competitive

Sei I, Q beliebig.
Sei $i \in \{1, ..., n\}$ der Gegenstand mit maximalen Wert.
Es gilt $F^*(I, Q) \leq W \cdot v_i$, da $\underline{w}_i \geq 1$.
ALG_1 kann i einpacken, da $\overline{w}_i \leq W$.
Also ist $\mathrm{ALG}_1(I, Q) \geq v_i$.

Also gilt:

$$\frac{F^*(I, Q)}{\mathrm{ALG}_1(I, Q)} \leq \frac{W \cdot v_i}{v_i} = W$$

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

v_i	1	...	1	1	...	1
$\overline{w}_i$	W	...	W	W	...	W
$\underline{w}_i$	1	...	1	1	...	1
$\hat{w}_i$	W	...	W	1	...	1
	$\underbrace{\qquad\qquad}_{\frac{n}{2}}$			$\underbrace{\qquad\qquad}_{\frac{n}{2}}$		

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

v_i	1	...	1	1	...	1
$\overline{w}_i$	W	...	W	W	...	W
$\underline{w}_i$	1	...	1	1	...	1
$\hat{w}_i$	W	...	W	1	...	1

$\underbrace{\qquad\qquad\qquad}_{\frac{n}{2}} \underbrace{\qquad\qquad\qquad}_{\frac{n}{2}}$

Annahme: A fragt nur Gegenstände mit $\hat{w}_i = W$ ab

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

v_i	1	...	1	1	...	1
$\overline{w}_i$	W	...	W	W	...	W
$\underline{w}_i$	1	...	1	1	...	1
$\hat{w}_i$	W	...	W	1	...	1

$\underbrace{\qquad\qquad\qquad}_{\frac{n}{2}}$ $\underbrace{\qquad\qquad\qquad}_{\frac{n}{2}}$

Wieso?

$\downarrow$

$\boxed{\text{Annahme: } A \text{ fragt nur Gegenstände mit } \hat{w}_i = W \text{ ab}}$

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

v_i	1	...	1	1	...	1
$\overline{w}_i$	W	...	W	W	...	W
$\underline{w}_i$	1	...	1	1	...	1
$\hat{w}_i$	W	...	W	1	...	1

$\underbrace{\qquad\qquad}_{\frac{n}{2}}$ $\underbrace{\qquad\qquad}_{\frac{n}{2}}$

Wieso?

$\downarrow$

Annahme: A fragt nur Gegenstände mit $\hat{w}_i = W$ ab

$\implies A(I, Q) = 1$

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

v_i	1	...	1	1	...	1
$\overline{w}_i$	W	...	W	W	...	W
$\underline{w}_i$	1	...	1	1	...	1
$\hat{w}_i$	W	...	W	1	...	1

$$\underbrace{\qquad\qquad}_{\frac{n}{2}} \quad \underbrace{\qquad\qquad}_{\frac{n}{2}}$$

Wieso?

$\boxed{\text{Annahme: } A \text{ fragt nur Gegenstände mit } \hat{w}_i = W \text{ ab}}$

$\implies A(I, Q) = 1$

$F^*(I, Q) = \frac{n}{2}$

Deterministische Algorithmen - Schranken

Sei A ein beliebiger deterministischer Algorithmus.

Sei I wie folgt definiert:

- $W = n$
- $Q = \frac{n}{2}$

v_i	1	...	1	1	...	1
$\overline{w}_i$	W	...	W	W	...	W
$\underline{w}_i$	1	...	1	1	...	1
$\hat{w}_i$	W	...	W	1	...	1

$\underbrace{\qquad\qquad\qquad}_{\frac{n}{2}}\ \underbrace{\qquad\qquad\qquad}_{\frac{n}{2}}$

Wieso?

$\boxed{\text{Annahme: } A \text{ fragt nur Gegenstände mit } \hat{w}_i = W \text{ ab}}$

$\implies A(I, Q) = 1$

$F^*(I, Q) = \frac{n}{2}$

query competitiveness: $\frac{F^*(I,Q)}{A(I,Q)} = \frac{\frac{n}{2}}{1} = \frac{n}{2}$

Algorithmen - Schranken

<table>
<tr><th colspan="2">Parameter</th><th>Schranke</th><th>Scharf?</th></tr>
<tr><td rowspan="4">$W \geq \frac{n}{2}$</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td rowspan="3">$W < \frac{n}{2}$</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
</table>

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q + 1$	ALG_1
$W < \frac{n}{2}$			

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q + 1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n - Q + 1$	ALG_2
$W < \frac{n}{2}$			

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q+1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n-Q+1$	ALG_2
	$Q = \frac{n}{2}, W > n$	$\geq Q+1$	ALG_1
$W < \frac{n}{2}$			

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q + 1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n - Q + 1$	ALG_2
	$Q = \frac{n}{2}, W > n$	$\geq Q + 1$	ALG_1
	$Q = \frac{n}{2}, W \leq n$	$\geq Q$	$\times$
$W < \frac{n}{2}$			

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q + 1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n - Q + 1$	ALG_2
	$Q = \frac{n}{2}, W > n$	$\geq Q + 1$	ALG_1
	$Q = \frac{n}{2}, W \leq n$	$\geq Q$	×
$W < \frac{n}{2}$			

$\text{ALG}_1, \text{ALG}_2$ sind $Q + 1$ worst-case query competitive

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q + 1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n - Q + 1$	ALG_2
	$Q = \frac{n}{2}, W > n$	$\geq Q + 1$	ALG_1
	$Q = \frac{n}{2}, W \leq n$	$\geq Q$	$\times$
$W < \frac{n}{2}$	$0 \leq Q < W$	$\geq Q + 1$	ALG_1

$\text{ALG}_1, \text{ALG}_2$ sind $Q+1$ worst-case query competitive

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q+1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n-Q+1$	ALG_2
	$Q = \frac{n}{2}, W > n$	$\geq Q+1$	ALG_1
	$Q = \frac{n}{2}, W \leq n$	$\geq Q$	×
$W < \frac{n}{2}$	$0 \leq Q < W$	$\geq Q+1$	ALG_1
	$W \leq Q \leq n-W$	$\geq W$	ALG_1

$\mathrm{ALG}_1, \mathrm{ALG}_2$ sind $Q+1$ worst-case query competitive

Algorithmen - Schranken

Parameter		Schranke	Scharf?
$W \geq \frac{n}{2}$	$Q < \frac{n}{2}$	$\geq Q + 1$	ALG_1
	$Q > \frac{n}{2}$	$\geq n - Q + 1$	ALG_2
	$Q = \frac{n}{2}, W > n$	$\geq Q + 1$	ALG_1
	$Q = \frac{n}{2}, W \leq n$	$\geq Q$	$\times$
$W < \frac{n}{2}$	$0 \leq Q < W$	$\geq Q + 1$	ALG_1
	$W \leq Q \leq n - W$	$\geq W$	ALG_1
	$n - W < Q$	$\geq n - Q + 1$	ALG_2

$\longleftarrow$ $\mathrm{ALG}_1, \mathrm{ALG}_2$ sind $Q + 1$ worst-case query competitive

Γ-beschränkte Unsicherheit

Neue Unsicherheitsmenge

$$U = \left\{ \{w_1, ..., w_n\} \,\middle|\, w_i \in \left\{ \underline{w}_i, \overline{w}_i \right\} \text{ und } \left| \left\{ i : w_i \neq \underline{w}_i \right\} \right| = \Gamma \right\}$$

Γ-beschränkte Unsicherheit

Neue Unsicherheitsmenge

$$U = \left\{ \{w_1, ..., w_n\} \,\middle|\, w_i \in \underline{\left\{ \underline{w}_i, \overline{w}_i \right\}} \text{ und } \left| \left\{ i : w_i \neq \underline{w}_i \right\} \right| = \Gamma \right\}$$

Γ-beschränkte Unsicherheit

Neue Unsicherheitsmenge

$$U = \left\{ \{w_1, \ldots, w_n\} \,\middle|\, w_i \in \left\{ \underline{w}_i, \overline{w}_i \right\} \text{ und } \left| \left\{ i : w_i \neq \underline{w}_i \right\} \right| \boxed{=} \Gamma \right\}$$

Γ-beschränkte Unsicherheit

Neue Unsicherheitsmenge

$$U = \left\{ \{w_1, ..., w_n\} \,\middle|\, w_i \in \left\{ \underline{w}_i, \overline{w}_i \right\} \text{ und } \left| \left\{ i : w_i \neq \underline{w}_i \right\} \right| \boxed{=} \Gamma \right\}$$

Rest analog:

- Q-Abfragen erlaubt
- Lösungen müssen robust sein
- Algorithmen in zwei Teilen:
  1. Teil: Entscheide welche Gegenstände abgefragt werden sollen
  2. Teil: Packe Gegenstände robust
- Full Knowledge Problem darf das Wissen über exakte Gewichte nicht zum Packen verwenden

Γ-beschränkte Unsicherheit

Neue Unsicherheitsmenge

$$U = \left\{ \{w_1, ..., w_n\} \,\middle|\, w_i \in \left\{ \underline{w}_i, \overline{w}_i \right\} \text{ und } \left| \left\{ i : w_i \neq \underline{w}_i \right\} \right| \boxed{=} \Gamma \right\}$$

Rest analog:

- Q-Abfragen erlaubt
- Lösungen müssen robust sein
- Algorithmen in zwei Teilen:
  1. Teil: Entscheide welche Gegenstände abgefragt werden sollen
  2. Teil: Packe Gegenstände robust ⟵ Lösung über ein MILP
- Full Knowledge Problem darf das Wissen über exakte Gewichte nicht zum Packen verwenden

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$
$Q = 2$
$\Gamma = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	?	?

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	9	?

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	$\boxed{6}$	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	?	?	9	?

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	$\boxed{6}$	3	9	1
$\overline{w}_i$	3	$\boxed{6}$	5	9	4
$\underline{w}_i$	1	$\boxed{3}$	3	4	1
$\hat{w}_i$	?	$\boxed{6}$	?	9	?

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$\hat{w}_i$	?	6	?	9	?

- $\hat{w}_1 = \underline{w}_1$
- $\hat{w}_3 = \underline{w}_3$
- $\hat{w}_5 = \underline{w}_5$

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	$\boxed{1}$	3	$\boxed{3}$	4	$\boxed{1}$
$\hat{w}_i$	?	$\boxed{6}$	?	$\boxed{9}$	?

- $\hat{w}_1 = \underline{w}_1$
- $\hat{w}_3 = \underline{w}_3$
- $\hat{w}_5 = \underline{w}_5$

Γ-beschränkte Unsicherheit - Beispiel

$W = 10$

$Q = 2$

$\Gamma = 2$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	$\boxed{1}$	3	$\boxed{3}$	4	$\boxed{1}$
$\hat{w}_i$	?	$\boxed{6}$	?	$\boxed{9}$	?

- $\hat{w}_1 = \underline{w}_1$
- $\hat{w}_3 = \underline{w}_3$
- $\hat{w}_5 = \underline{w}_5$

Lösung: $\{1, 2, 3\}$ Gesamtwert: 14

Heuristiken für Intervall und Γ-beschränkte Unsicherheit

Heuristiken

Wieso? Bisherige Algorithmen benutzen Parameter der Gegenstände nicht, um zu entscheiden, welcher Gegenstand abgefragt werden soll

Heuristiken

Wieso? Bisherige Algorithmen benutzen Parameter der Gegenstände nicht, um zu entscheiden, welcher Gegenstand abgefragt werden soll

Mögliche Parameter:

- $\overline{w}_i$
- $\underline{w}_i$
- v_i

Heuristiken

Wieso? Bisherige Algorithmen benutzen Parameter der Gegenstände nicht, um zu entscheiden, welcher Gegenstand abgefragt werden soll

Mögliche Parameter:

- $\overline{w}_i$
- $\underline{w}_i$
- v_i

Γ

- $\overline{a}$:= Anzahl an Gewichten die abgefragt wurden, mit $\hat{w}_i = \overline{w}_i$
- $\underline{a}$:= Anzahl an Gewichten die abgefragt wurden, mit $\hat{w}_i = \underline{w}_i$
- Γ

Heuristiken

Wieso? Bisherige Algorithmen benutzen Parameter der Gegenstände nicht, um zu entscheiden, welcher Gegenstand abgefragt werden soll

Mögliche Parameter:

- $\overline{w}_i$
- $\underline{w}_i$
- v_i

Γ

- $\overline{a}$:= Anzahl an Gewichten die abgefragt wurden, mit $\hat{w}_i = \overline{w}_i$
- $\underline{a}$:= Anzahl an Gewichten die abgefragt wurden, mit $\hat{w}_i = \underline{w}_i$
- Γ

- W
- n

$\mathbb{E}[\hat{w}_i]$ bestimmen

Daraus lässt $\mathbb{E}[\hat{w}_i]$ berechnen, wenn Gewichte gleichverteilt sind.

$\mathbb{E}[\hat{w}_i]$ bestimmen

Daraus lässt $\mathbb{E}[\hat{w}_i]$ berechnen, wenn Gewichte gleichverteilt sind.

Intervall Unsicherheit:

$$\mathbb{E}[\hat{w}_i] = \frac{1}{2}\left(\underline{w}_i + \overline{w}_i\right)$$

$\mathbb{E}[\hat{w}_i]$ bestimmen - Γ-beschränkte Unsicherheit

$\mathbb{E}[\hat{w}_i]$ bestimmen - Γ-beschränkte Unsicherheit

$\Gamma = n - \underline{a}$:

$$\mathbb{E}[\hat{w}_i] = \overline{w}_i$$

$\mathbb{E}[\hat{w}_i]$ bestimmen - Γ-beschränkte Unsicherheit

$\Gamma = n - \underline{a}$:

$$\mathbb{E}[\hat{w}_i] = \overline{w}_i$$

$\Gamma < n - \underline{a}$:

$$\begin{aligned}\mathbb{E}[\hat{w}_i] &= \underline{w}_i P\left[\hat{w}_i = \underline{w}_i \,\middle|\, \overline{a}, \underline{a}\right] + \overline{w}_i P[\hat{w}_i = \overline{w}_i \mid \overline{a}, \underline{a}] \\ &= \ldots \\ &= \underline{w}_i + \left(1 - \frac{n - \overline{a} - \Gamma}{n - \overline{a} - \underline{a}}\right)(\overline{w}_i - \underline{w}_i)\end{aligned}$$

Heuristiken

Heuristiken werden als Funktionen $H : \{1, ..., n\} \to \mathbb{R}$ dargestellt, die jedem Gegenstand einen Wert zuordnet.

Der Algorithmus fragt dann das Gewicht der Q Gegenstände mit dem höchsten Wert $H(i)$ ab.

Heuristiken

Heuristiken werden als Funktionen $H : \{1, ..., n\} \to \mathbb{R}$ dargestellt, die jedem Gegenstand einen Wert zuordnet.

Der Algorithmus fragt dann das Gewicht der Q Gegenstände mit dem höchsten Wert $H(i)$ ab.

Das Auswerten von H muss nach jeder Abfrage wiederholt werden.

Heuristiken

Heuristiken werden als Funktionen $H : \{1, ..., n\} \to \mathbb{R}$ dargestellt, die jedem Gegenstand einen Wert zuordnet.

Der Algorithmus fragt dann das Gewicht der Q Gegenstände mit dem höchsten Wert $H(i)$ ab.

Das Auswerten von H muss nach jeder Abfrage wiederholt werden.

Der Algorithmus löst dann den zweiten Teil des Problems.

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	?	?	?	?	?

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	?	?	?	?

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	?	?	?

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\textcolor{green}{\Gamma}\quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	9	?	?

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	9	36	?

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	9	36	1

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	9	36	1

1.

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	9	36	1
		2.		1.	

Heuristiken - Beispiel

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \;\; \overline{a},\ \underline{a},\ \Gamma,}$ n, W

Beispiel:

$$H(i) = \underline{w}_i \cdot v_i$$

v_i	5	6	3	9	1
$\overline{w}_i$	3	6	5	9	4
$\underline{w}_i$	1	3	3	4	1
$H(i)$	5	18	9	36	1
	4.	2.	3.	1.	5.

Heuristiken - Bewertungen

Sei $\mathcal{I}$ eine Menge von Instanzen, Q die Anzahl an Abfragen

Heuristiken - Bewertungen

Sei $\mathcal{J}$ eine Menge von Instanzen, Q die Anzahl an Abfragen

Worst-case query competitiveness:

$$\max_{I \in \mathcal{J}} \frac{F^*(I, Q)}{A(I, Q)}$$

Heuristiken - Bewertungen

Sei $\mathcal{J}$ eine Menge von Instanzen, Q die Anzahl an Abfragen

Worst-case query competitiveness:

$$\max_{I \in \mathcal{J}} \frac{F^*(I,Q)}{A(I,Q)}$$

Average query competitiveness:

$$\frac{1}{|\mathcal{J}|} \sum_{I \in \mathcal{J}} \frac{F^*(I,Q)}{A(I,Q)}$$

Heuristiken - Testen

Intervall Unsicherheit (100 Instanzen)

- $n = 100$
- $W \in \{51, ..., n\}$ gleichverteilt
- $v_i \in \{1, ..., W\}$ gleichverteilt
- $\overline{w}_i, \underline{w}_i \in \{1, ..., W\}$ gleichverteilt
- $\hat{w}_i \in \left\{\underline{w}_i, ..., \overline{w}_i\right\}$ gleichverteilt

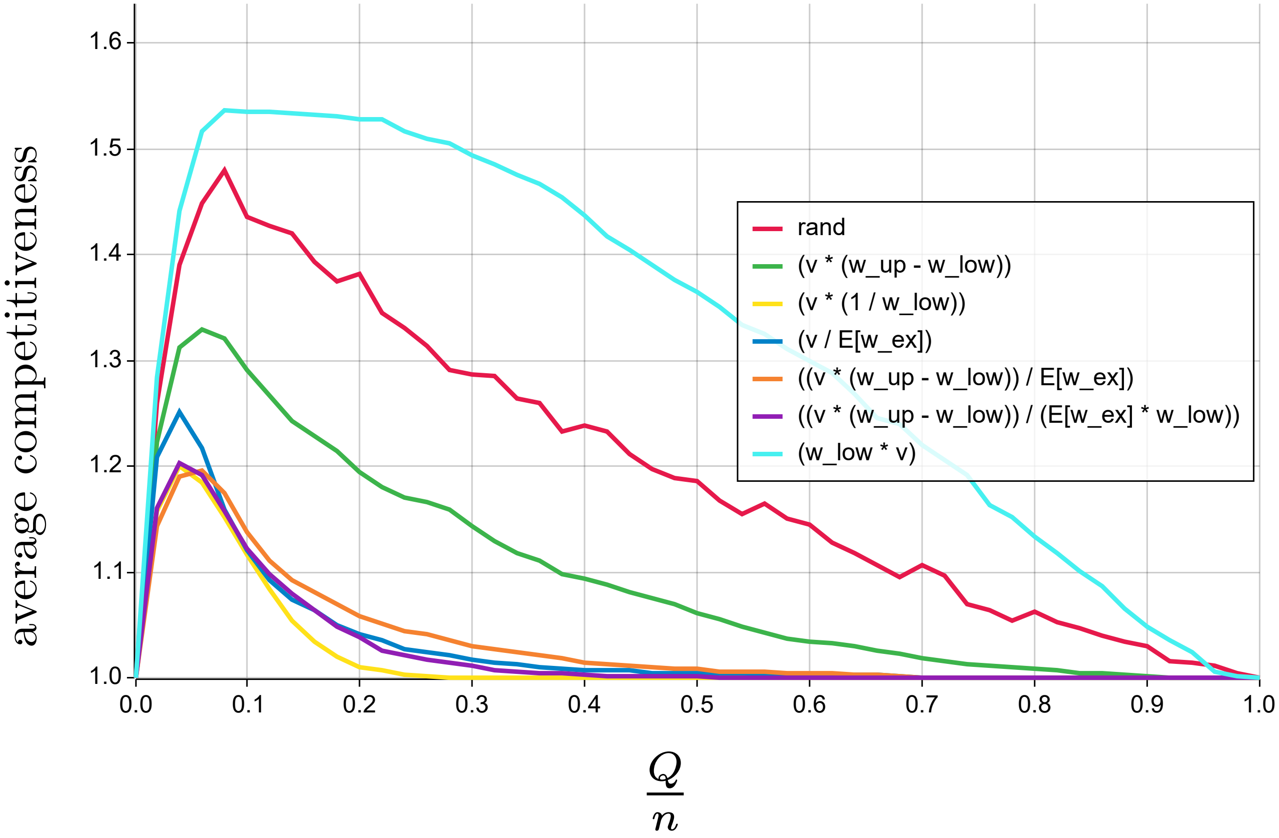
Teste verschiedene Q Werte

Γ-beschränkte Unsicherheit (20 Instanzen)

- $n = 20$
- $\Gamma = 2$
- Rest analog

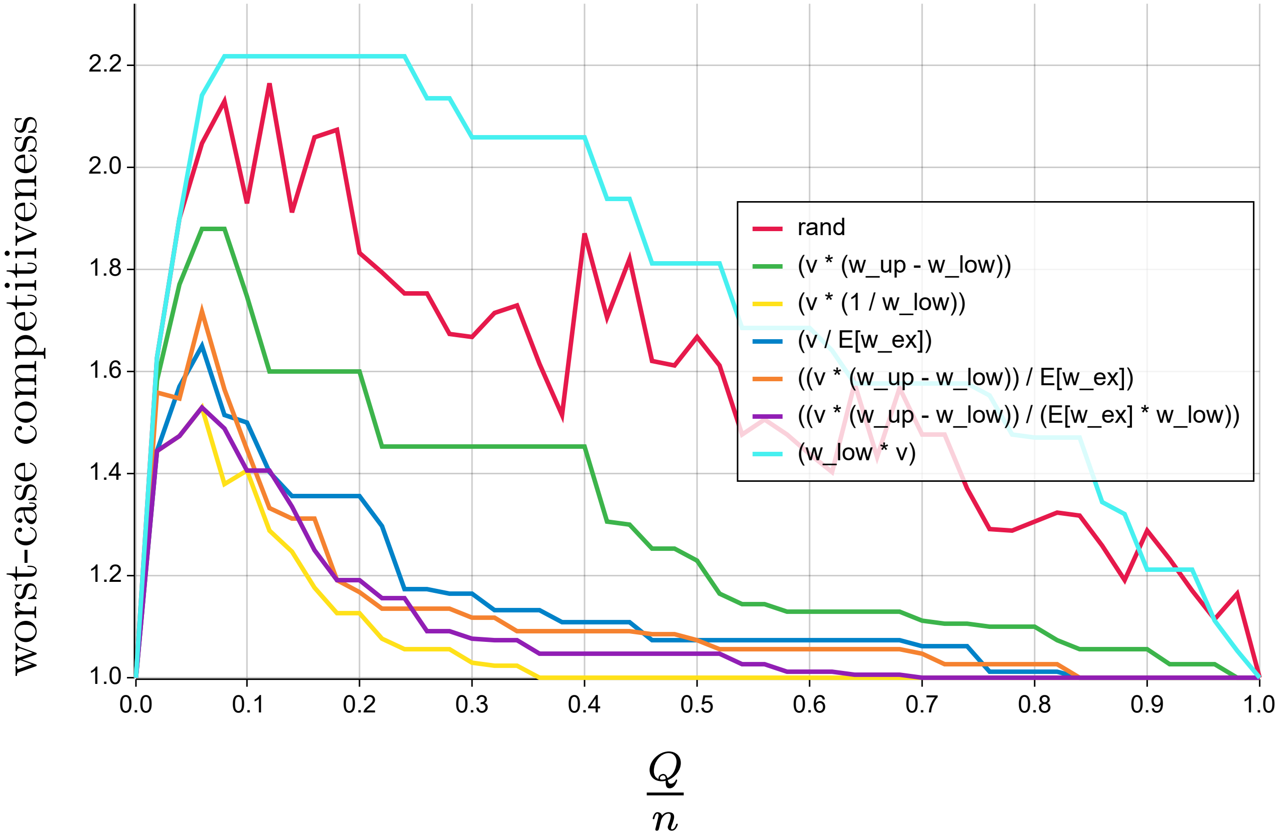
Heuristiken - Intervall Unsicherheit - Vorschläge

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W



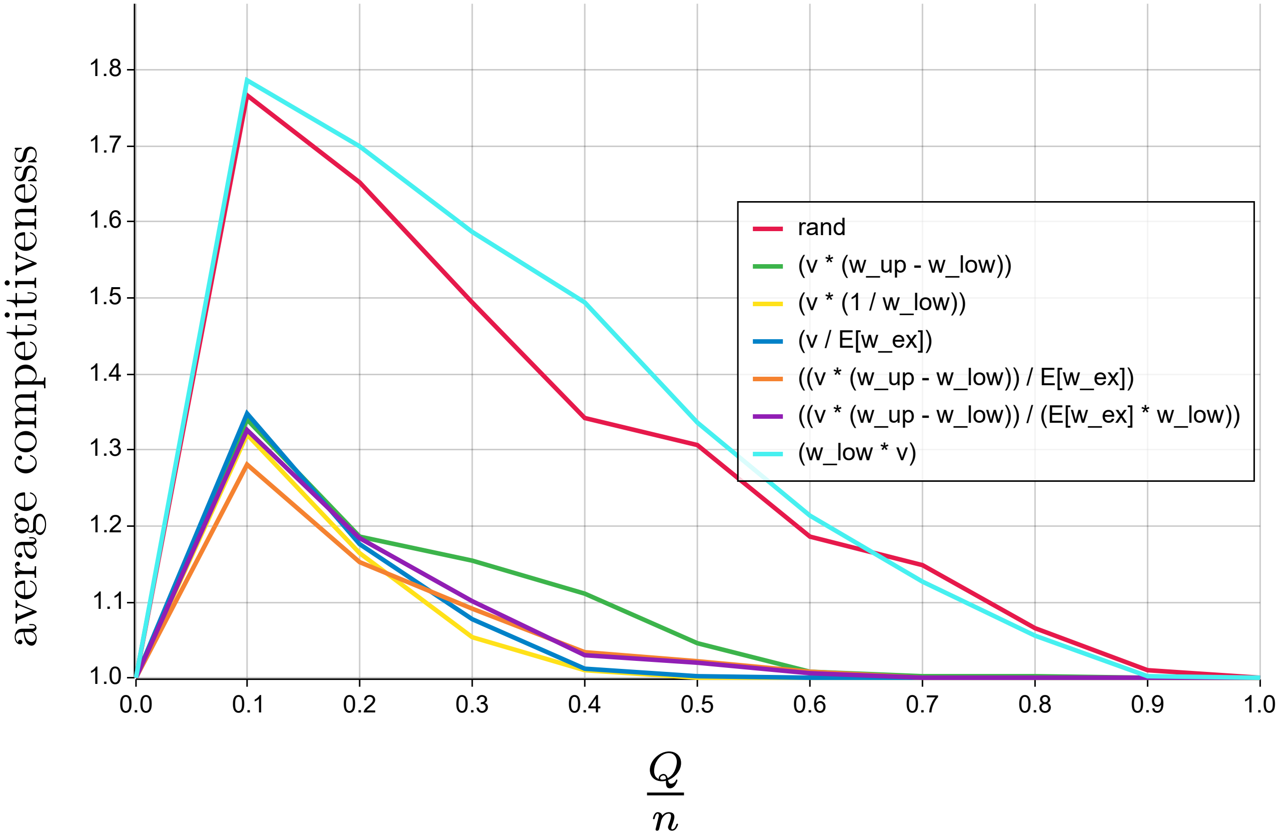
Heuristiken - Intervall Unsicherheit - Vorschläge

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\overline{a},\ \underline{a},\ \Gamma,}$ n, W



Heuristiken - Γ-beschränkte Unsicherheit - Vorschläge

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W



Heuristiken - Γ-beschränkte Unsicherheit - Vorschläge

Parameter: $\overline{w}_i$, $\underline{w}_i$, v_i, $\mathbb{E}[\hat{w}_i]$, $\boxed{\Gamma \quad \overline{a},\ \underline{a},\ \Gamma,}$ n, W