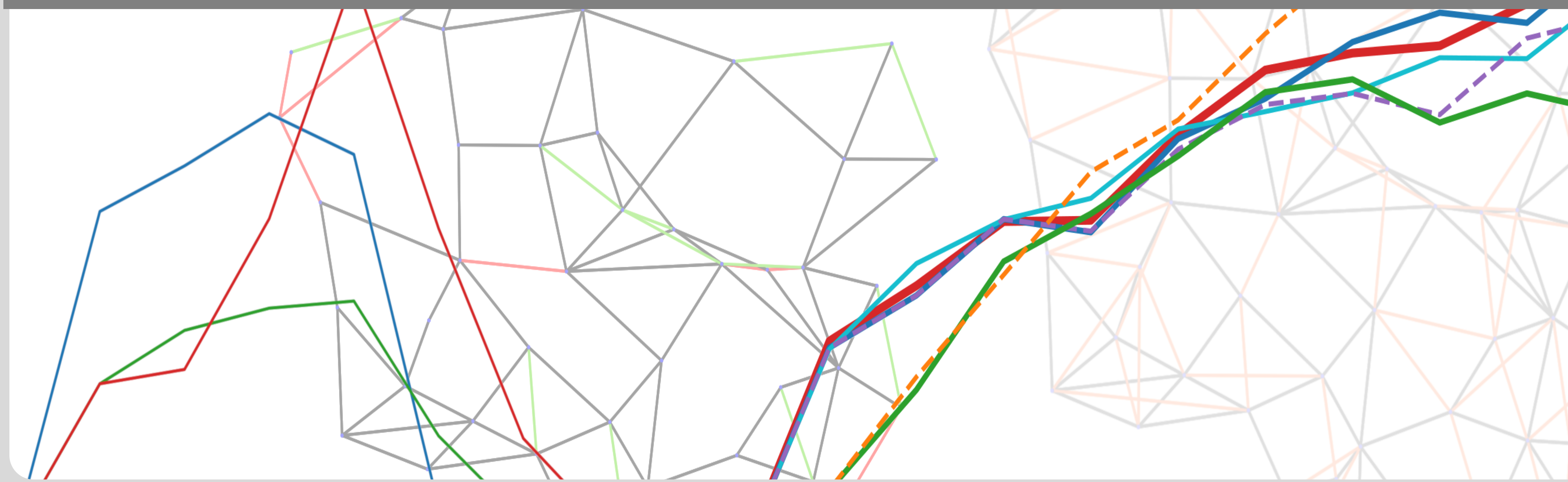
Transmission Network Expansion Planning for Curing Critical Edges

Abschlusspräsentation · 17.Mai 2019

Lena Winter

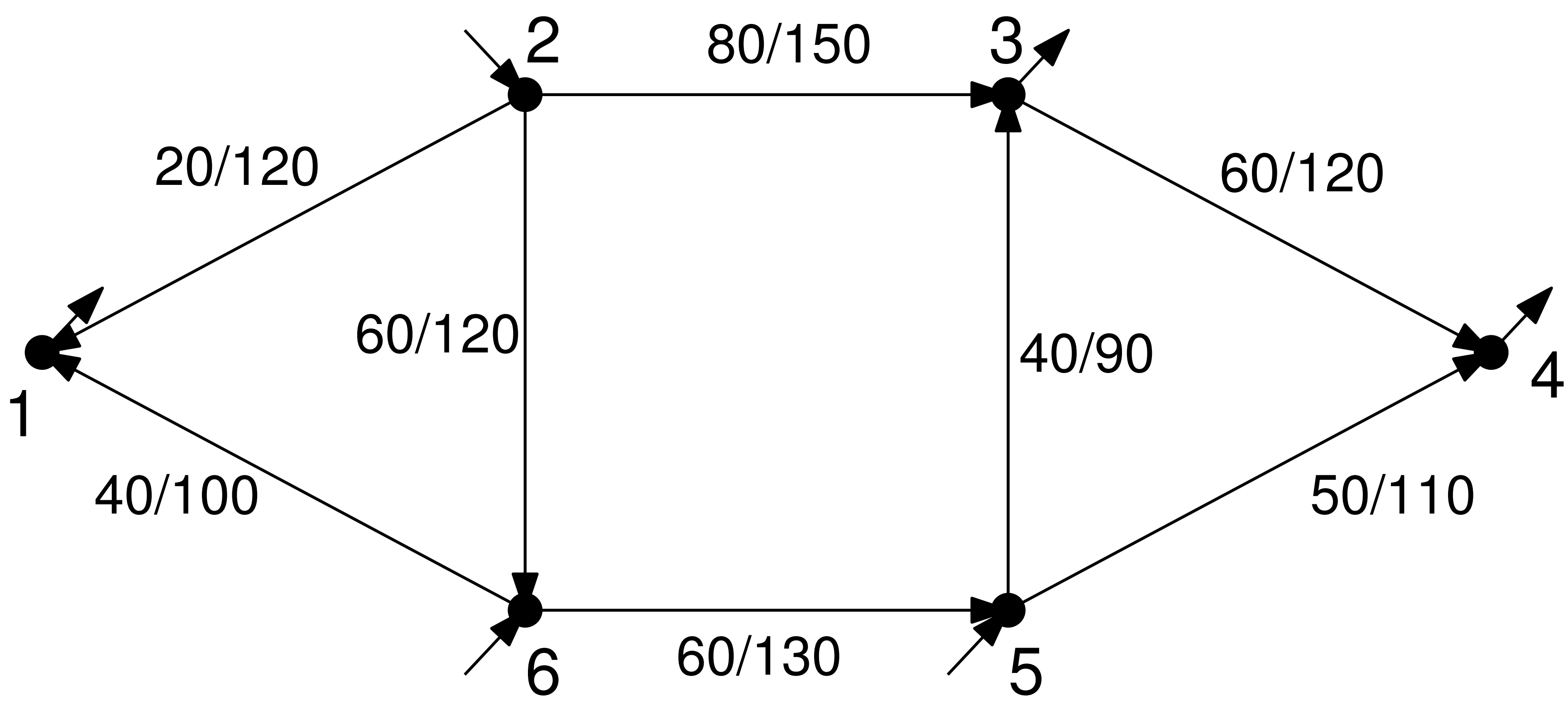
INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP



- **kritische Kante:** Stromleitung durch deren Ausfall ein Stromausfall ausgelöst wird

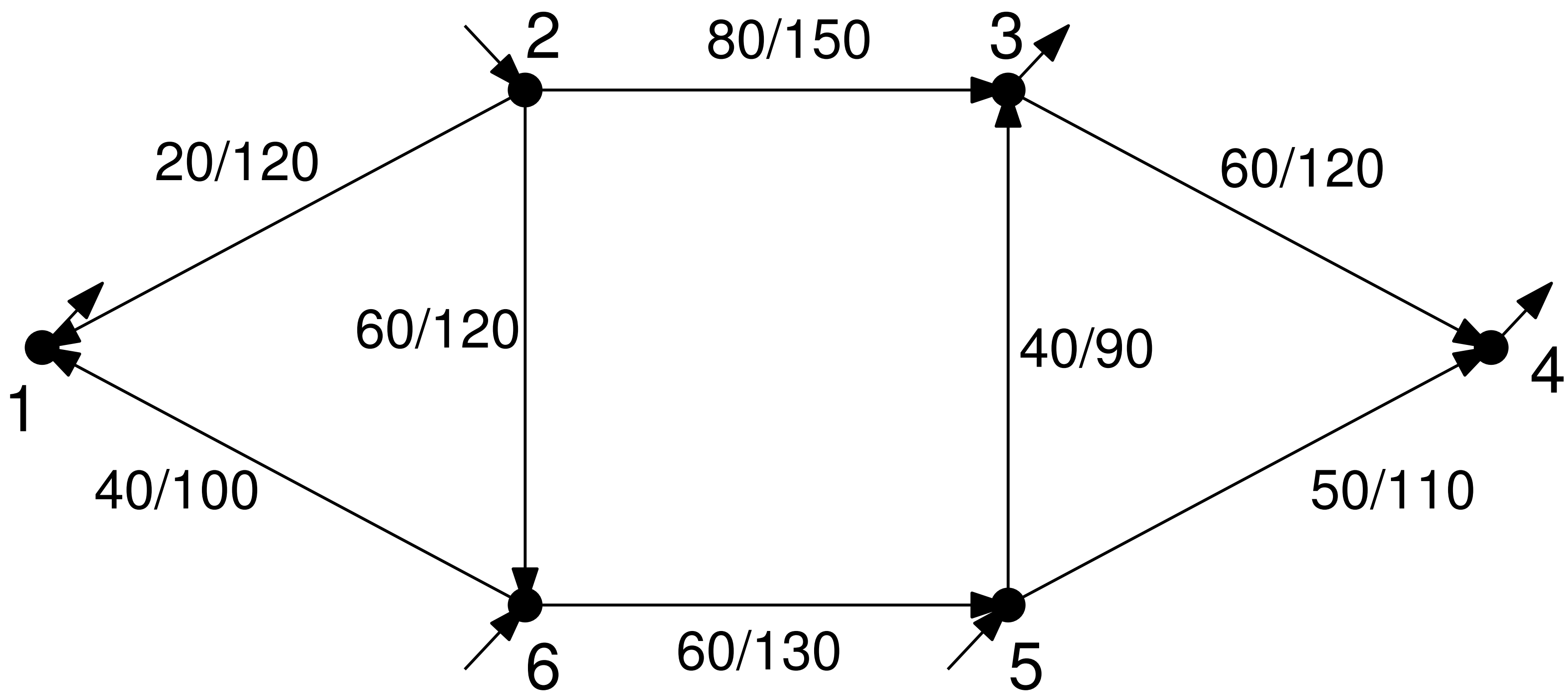
Kritische Kanten

- **Gegeben:** Graph $G = (V, E)$, Kapazität cap(a, b) und Fluss $f(a, b)$ für $(a, b) \in E$



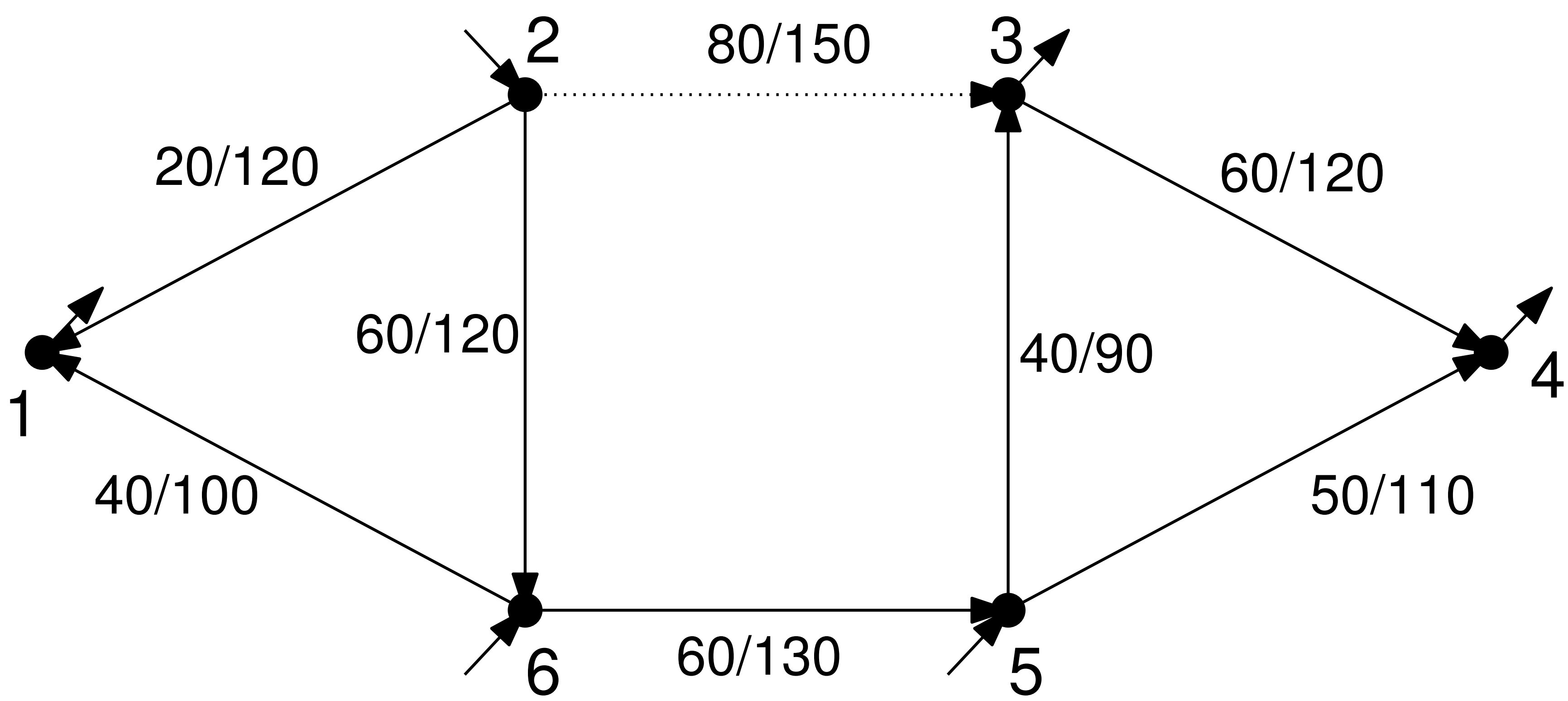
Kritische Kanten

- Ist die Kante $(2, 3)$ kritisch?



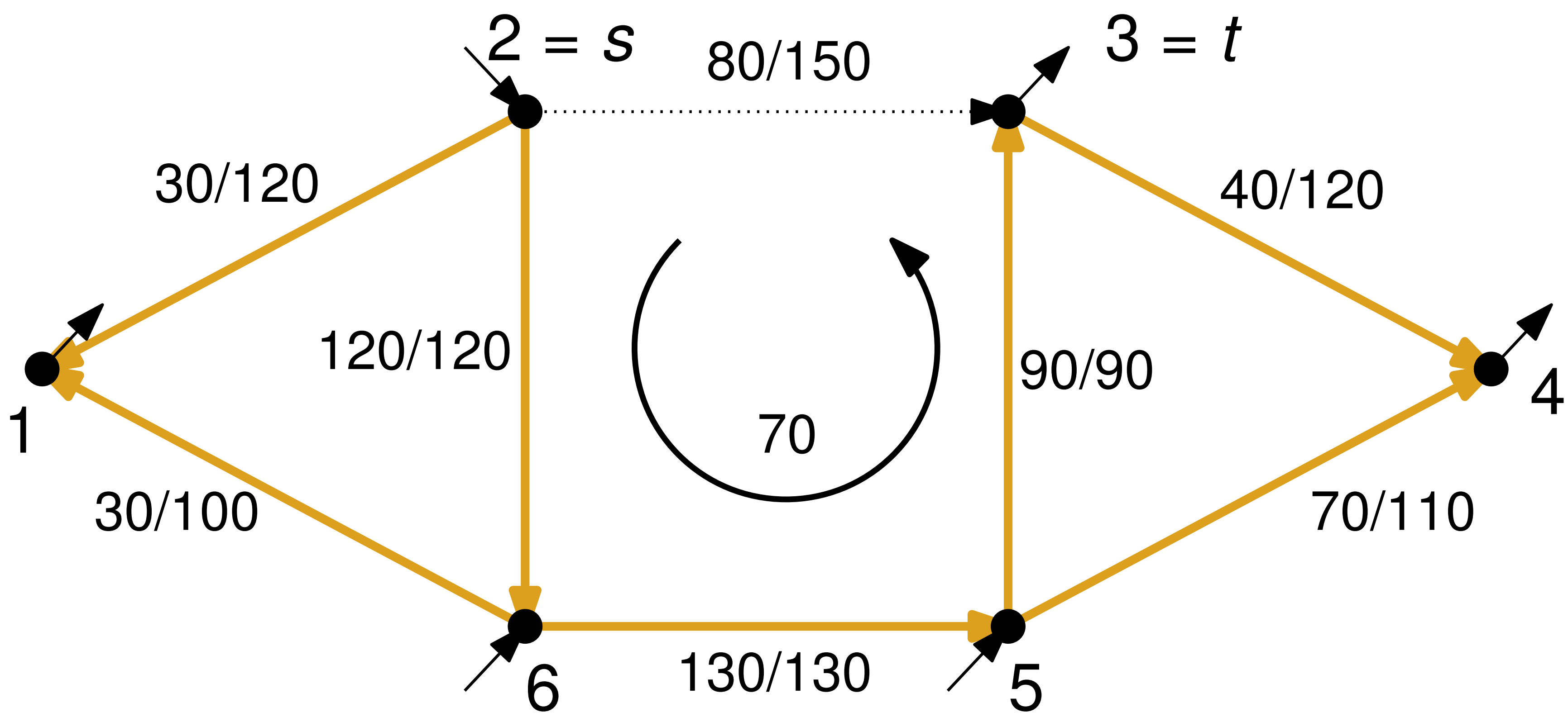
Kritische Kanten

- **Schritt 1:** Lösche Kante $(2, 3)$ aus G



Kritische Kanten

- **Schritt 2:** Bestimme den maximalen Fluss zwischen 2 und 3 $\rightarrow f_{\text{red}}(G, 2, 3)$



Kritische Kanten

- **Schritt 3 :** Kante ist kritisch wenn $\frac{f(a,b)}{f_{\mathrm{red}}(G,a,b)} > 0.614$

Kritische Kanten

- **Schritt 3 :** Kante ist kritisch wenn $\frac{f(a,b)}{f_{\text{red}}(G,a,b)} > 0.614$
- Im Bsp: $f(2,3) = 80$ und $f_{\text{red}}(G,2,3) = 70$

$$\frac{f(2,3)}{f_{\text{red}}(G,2,3)} = \frac{80}{70} = 1.14 > 0.614$$

Kritische Kanten

- **Schritt 3 :** Kante ist kritisch wenn $\frac{f(a,b)}{f_{\text{red}}(G,a,b)} > 0.614$
- Wie viel redundanter Fluss muss hinzugefügt werden?

Kritische Kanten

- **Schritt 3 :** Kante ist kritisch wenn $\frac{f(a,b)}{f_{\text{red}}(G,a,b)} > 0.614$

- Wie viel redundanter Fluss muss hinzugefügt werden?

- Criticality der kritischen Kante (a, b) in Graph G:

$$f_{\text{add}}(G, a, b) = \frac{f(a,b)}{0.614} - f_{\text{red}}(G, a, b)$$

- Im Bsp:

$$f_{\text{add}}(G, 2, 3) = \frac{80}{0.614} - 70 \approx 60.3$$

Motivation TNEP-CCE

- Modellierung von physikalisch korrekten Flüssen ist aufwendig
- Erweiterung nur auf Basis des redundanten Flusses möglich?

Problemstellung: TNEP-CCE

Gegeben:

- Graph $G = (V, E)$
- kritische Kanten E_{crit}
- Budget B
- Menge an Kandidatenkanten E_{cand}
- Kosten für jede der Kandidatenkanten $\text{cost}(a, b)$

Institute of Theoretical Informatics
Algorithmics Group

Problemstellung: TNEP-CCE

Gesucht: $E_{\text{add}} \subset E_{\text{cand}}$ so, dass für $G' = (V, E \cup E_{\text{add}})$:

- $\sum_{(a,b) \in E_{\text{crit}}} f_{\text{add}}(G', a, b)$ minimal
- $\sum_{(a,b) \in E_{\text{add}}} \text{cost}(a, b) \leq B$

Heuristik: Idee

- ähnliches Problem wie Knapsack
  - Budget $\approx$ Rucksackvolumen
  - Kandidatenkanten $\approx$ Gegenstände
  - Kosten $\approx$ Gewicht/Volumen der Gegenstände
  - Crititicality die durch Kante gelöst wird $\approx$ Wert

Heuristik: Wert einer Kandidatenkante

Annahme: nur eine kritische Kante (a, b)

- Criticality : $f_{\text{add}}(G, a, b)$
- Durch Hinzufügen von Kandidatenkante e zu G gelöste Criticality:

$$val(G, e) = f_{\text{add}}(G, a, b) - f_{\text{add}}(G \cup \{e\}, a, b)$$

Heuristik: Wert einer Kandidatenkante

Karlsruhe Institute of Technology

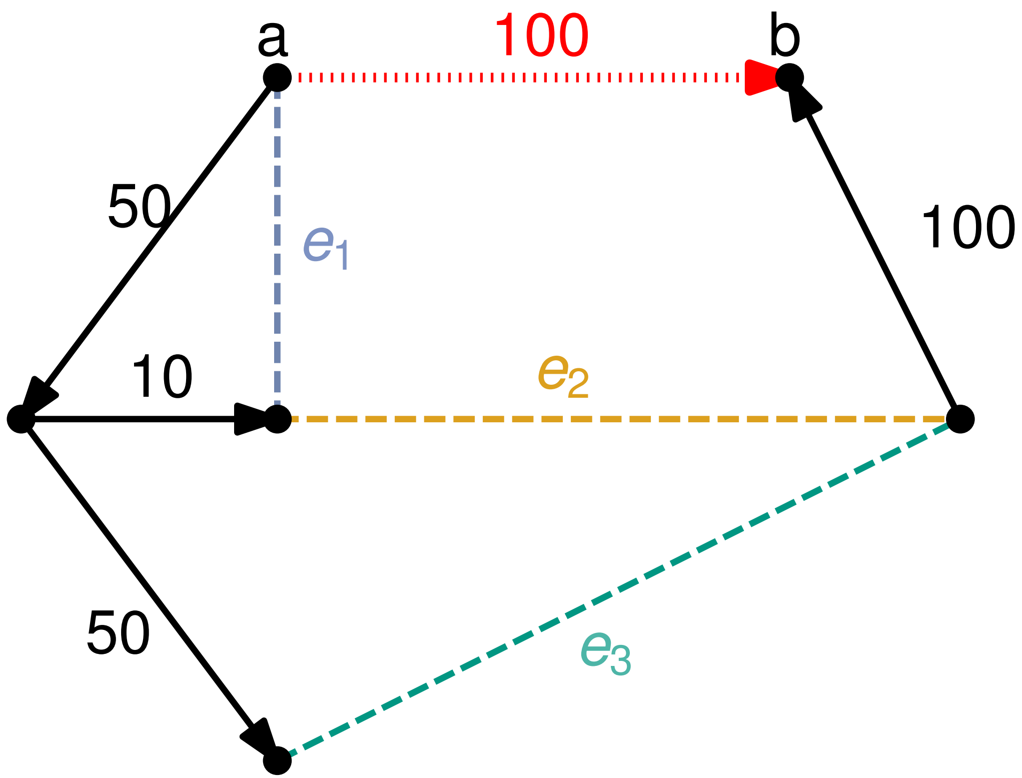
Annahme: nur eine kritische Kante (a, b)

- Criticality : $f_{\text{add}}(G, a, b)$
- Durch Hinzufügen von Kandidatenkante e zu G gelöste Criticality:

$$val(G, e) = f_{\text{add}}(G, a, b) - f_{\text{add}}(G \cup \{e\}, a, b)$$

- Was passiert beim Hinzufügen von einer Kandidatenkante mit dem Wert der anderen?

Heuristik: Wert einer Kandidatenkante



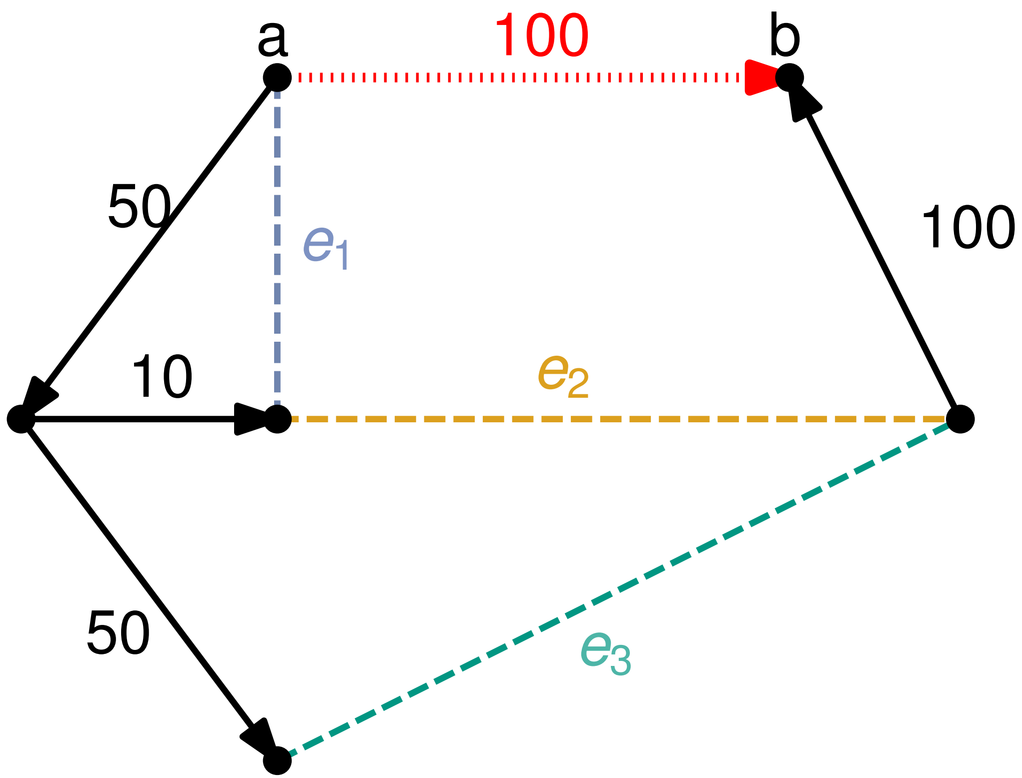
- $\text{cap}(e_n) = 100, n \in \{1, 2, 3\}$
- $\text{cost}(e_n) = 1, n \in \{1, 2, 3\}$
- Werte:

  $\text{val}(G, e_1) = 0$

  $\text{val}(G, e_2) = 10$

  $\text{val}(G, e_3) = 50$

Heuristik: Wert einer Kandidatenkante



- $\text{cap}(e_n) = 100, n \in \{1, 2, 3\}$
- $\text{cost}(e_n) = 1, n \in \{1, 2, 3\}$
- Werte:

  $\text{val}(G, e_1) = 0$

  $\text{val}(G, e_2) = 10$

  $\text{val}(G, e_3) = 50$

$\text{val}(G \cup \{e_1\}, e_2) = 100$	$\text{val}(G \cup \{e_2\}, e_1) = 90$	$\text{val}(G \cup \{e_3\}, e_1) = 0$
$\text{val}(G \cup \{e_1\}, e_3) = 50$	$\text{val}(G \cup \{e_2\}, e_3) = 40$	$\text{val}(G \cup \{e_3\}, e_2) = 0$

Institute of Theoretical Informatics
Algorithmics Group

Heuristik: Wert einer Kandidatenkante

Karlsruhe Institute of Technology

- Keine optimale Substruktur
  - Dynamische Programmierung findet nicht optimalen Lösung

Heuristik: Wert einer Kandidatenkante

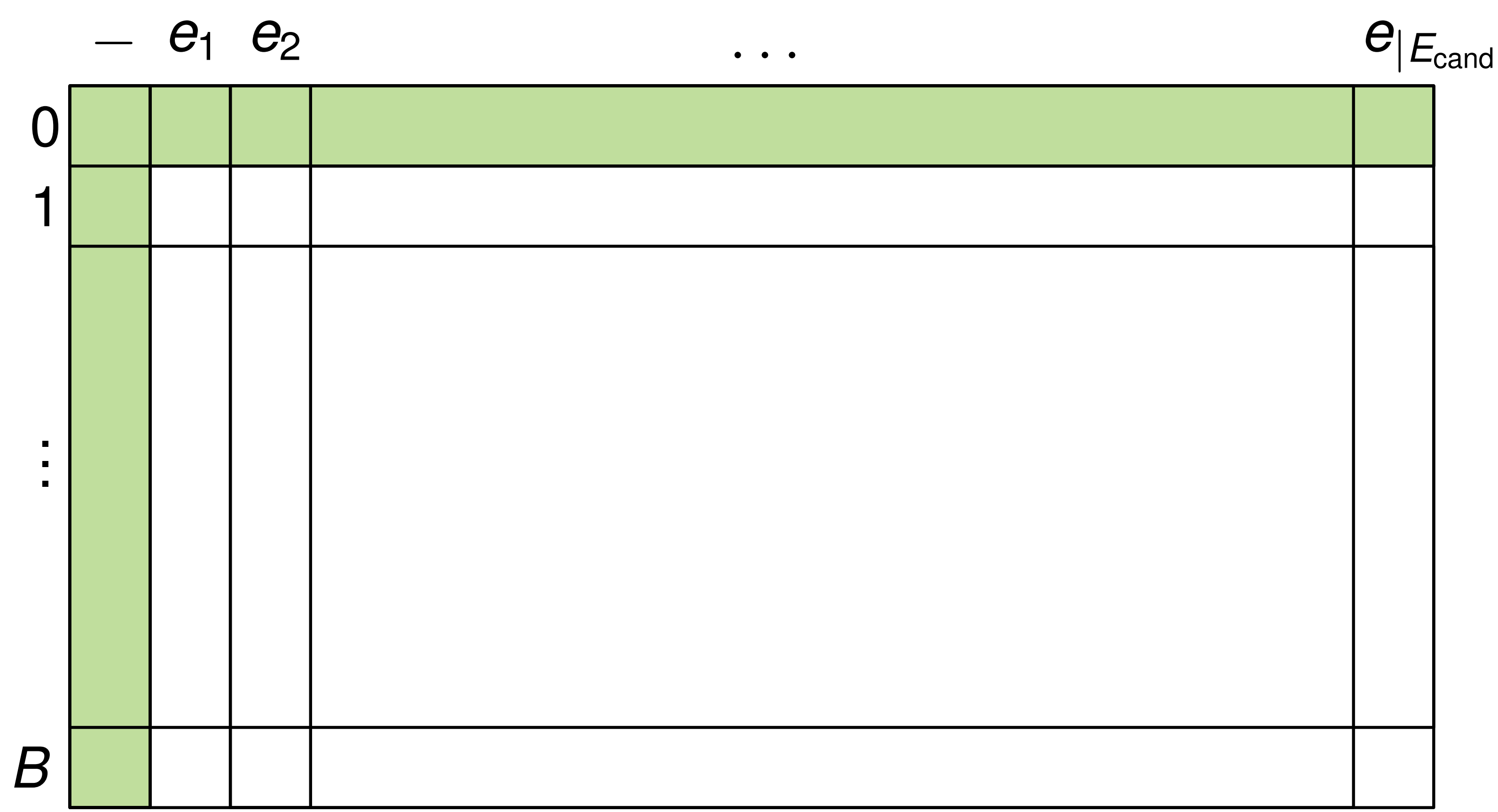
- Keine optimale Substruktur
  - Dynamische Programmierung findet nicht optimalen Lösung

- Werte der Kandidatenkanten verändern sich evtl. stark
  - Neuberechnung der Werte nach jedem Einfügen notwendig

> Die Heuristik ist keine dynamische Programmierung!

Heuristik: Umsetzung

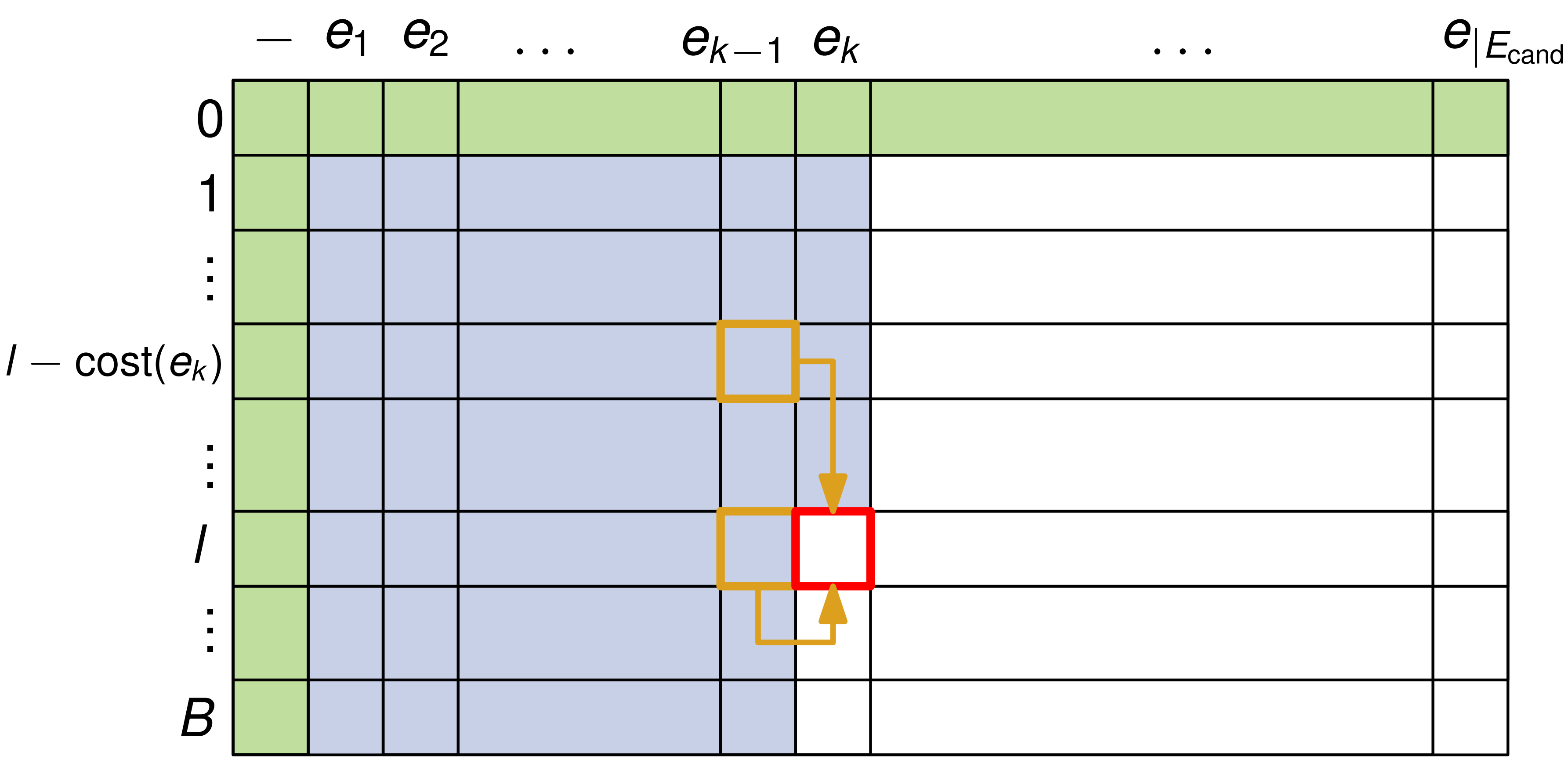
- Arraygröße: $(B + 1) \times (|E_{cand}| + 1)$



Initialisierung als $(\emptyset, 0, 0)$ für (Kandidatenkanten, Kosten, Wert)

Heuristik: Umsetzung

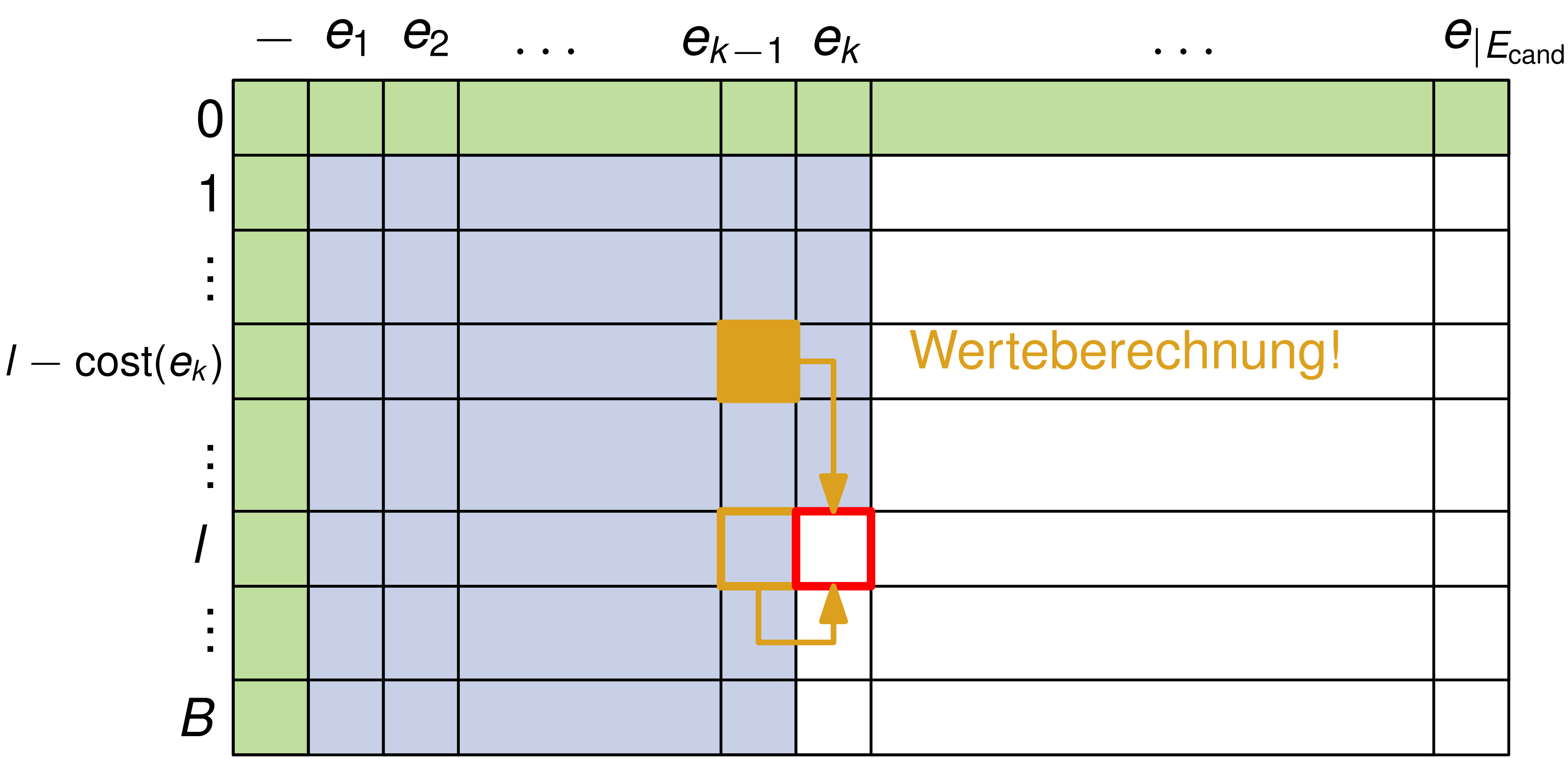
- Arraygröße: $(B + 1) \times (|E_{\text{cand}}| + 1)$



Eintrag in jedes Feld: (ausgewählte Kandidatenkanten, Kosten, Wert)

Heuristik: Umsetzung

- Arraygröße: $(B+1) \times (|E_{cand}| + 1)$



Eintrag in jedes Feld: (ausgewählte Kandidatenkanten, Kosten, Wert)

Institute of Theoretical Informatics
Algorithmics Group

Heuristik: Modifikationen

- Verwendung einer Lookup Tabelle für bereits berechnete Werte
  - HEU_LU
- Verwendung der Initalen Werte ohne Neuberechnung
  - HEU_APPROX

Lösungsansätze

Vergleichsmodell:

- Mixed-Integer Linear Program mit DC-Modell (MMDC_SET)

Modelle zum Lösen von TNEP-CCE:

- Mixed-Integer Linear Programm (MM_GRAPH)
- Heuristik:
  - Mit Lookup table (HEU_LU)
  - Mit Werte Approximation (HEU_APPROX)

Evaluation: Testdaten

Testdaten:

- 17 Übertragungsnetze verschiedener Länder
- Verbrauchsdaten für 2013 im Stundentakt

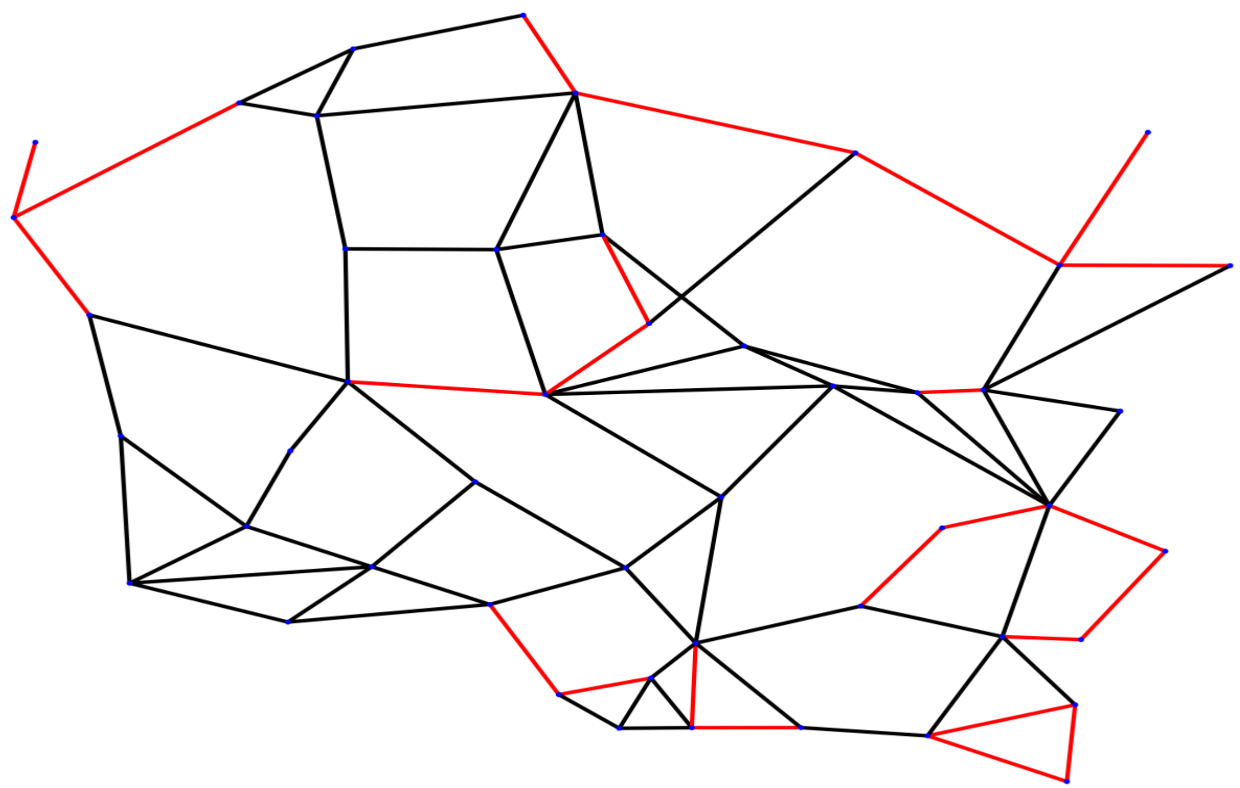
Evaluation: Testdaten

Testdaten:

- 17 Übertragungsnetze verschiedener Länder
- Verbrauchsdaten für 2013 im Stundentakt

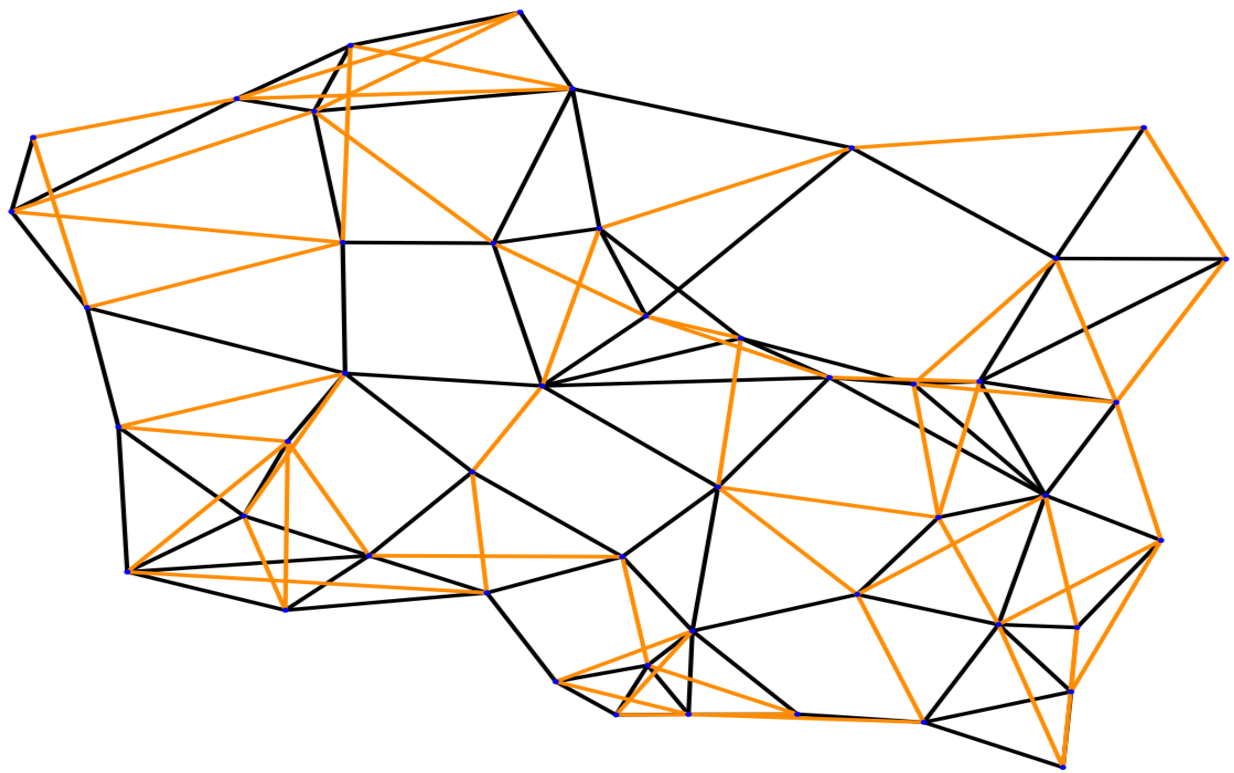
Evaluation: Testdaten

- Als Beispiel polnisches Übertragungsnetz
- kritische Kanten am 1.1.2013, 0 Uhr



Evaluation: Testdaten

- Als Beispiel polnisches Übertragungsnetz
- Kandidatenkanten



Evaluation: Testreihen

Karlsruhe Institute of Technology

Budget-Testreihe:

- Variation des Budgets
- Budget in Prozent der Gesamtkosten der Kandidatenkanten
- Festes Kandidatennetz mit min. 2 Kanididatenkanten pro Knoten
- 48 Snapshots jedes Netzes

Evaluation: Laufzeit

Aufteilung in:

- Zeit zur Erstellung des Modells
- Optimierungszeit

Evaluation: Laufzeit

Aufteilung in:

- Zeit zur Erstellung des Modells
- Optimierungszeit

Erwartungen Optimierungszeit:

- Längere Laufzeit für größeres Budget

Evaluation: Laufzeit

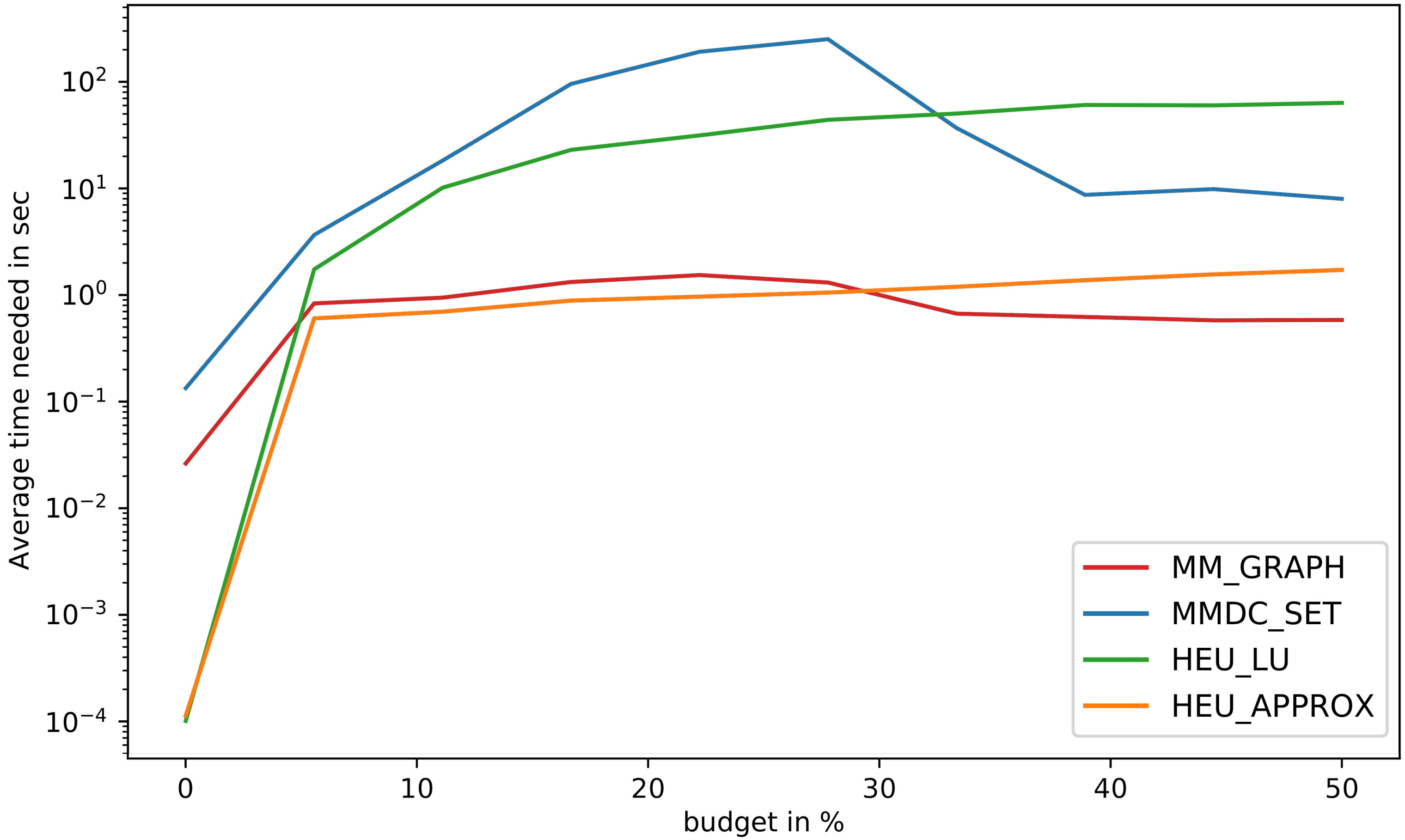
Aufteilung in:

- Zeit zur Erstellung des Modells
- Optimierungszeit

Erwartungen Optimierungszeit:

- Längere Laufzeit für größeres Budget
- HEU_APPROX ist schneller als HEU_LU
- Vergleichsmodell MMDC_SET ist am langsamsten

Evaluation: Optimierungszeit



Optimierungszeiten des polnischen Übertragungsnetzes (48 Knoten, 84 Kanten)

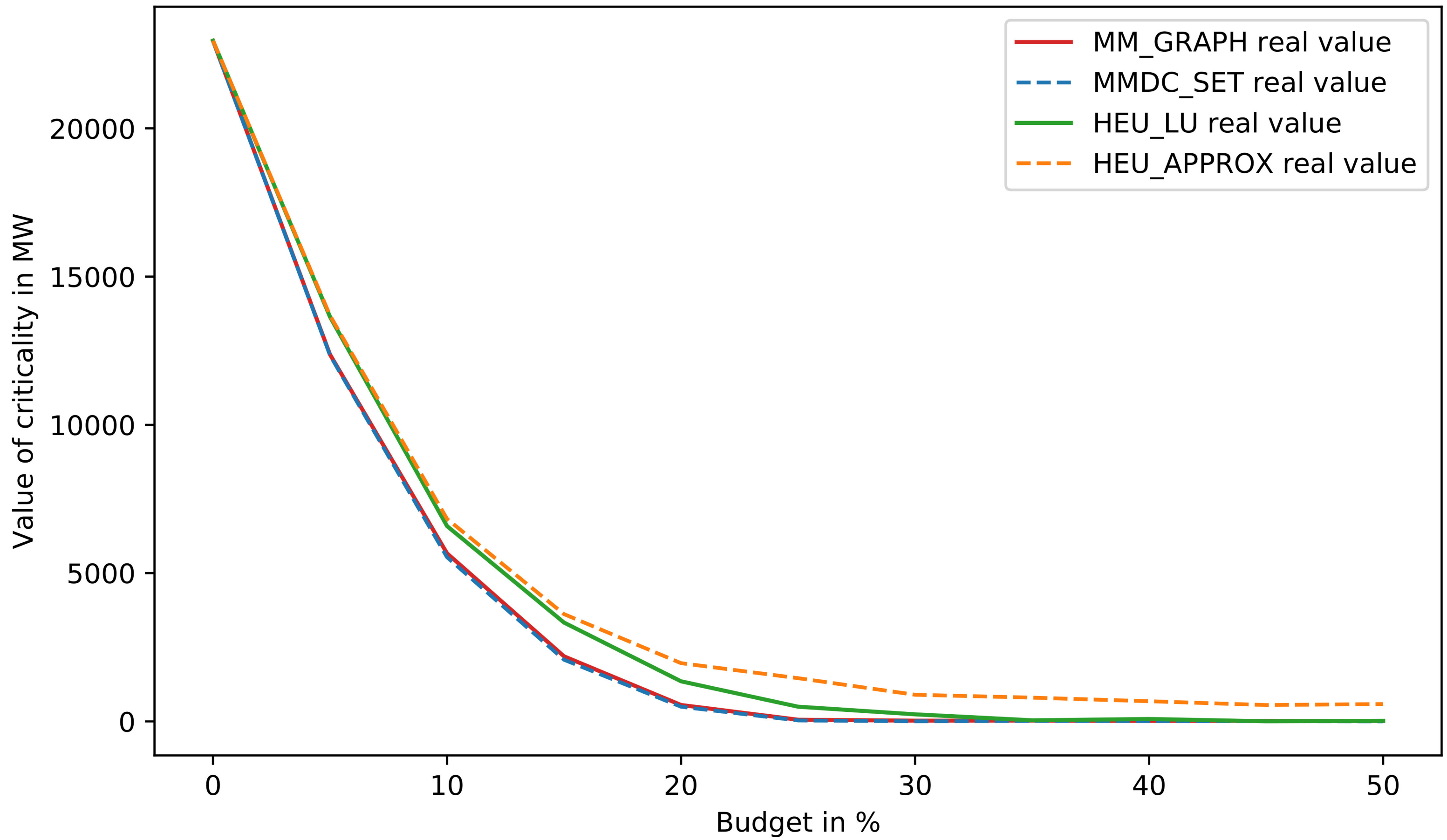
Evaluation: Criticality

Karlsruhe Institute of Technology

Annahmen:

- Mit steigendem Budget wird mehr Criticality gelöst
- Das Vergleichsmodell MMDC_SET löst am meisten Criticality
- Die Heuristik mit Werteapproximation löst am wenigsten Criticality

Evaluation: Criticality



Criticality des polnischen Übertragungsnetzes (48 Knoten, 84 Kanten)

Conclusion

- Die Vereinfachung Lastflüsse mit graph theoretischen Flüssen anzunähren funktioniert sehr gut
- Es kann durch die Vereinfachung zu schlechteren Ergebnissen kommen
- Dafür hohe Beschleunigungen
- Heuristik auf getesten Netzen deutlich schlechter als Vergleichsmethoden

Institute of Theoretical Informatics
Algorithmics Group

Future Work

Karlsruhe Institute of Technology

- Dynamische Formulierung für TNEP-CCE
- Verbesserung Heuristik:
    - Dynamische Flussalgorithmen
    - Einschränkung der zu untersuchenden Kandidatenkanten
- Untersuchen von größeren Netzen
- Verwendung von Metaheuristiken

Institute of Theoretical Informatics
Algorithmics Group