Approximation dynamischer Flüsse mit adaptiver Routenwahl in Netzwerken mit mehreren Senken

Verfasser: Johannes Hagenmaier

Matrikelnummer: 1473547

Studiengang: Mathematik

Abgabe: 09.08.2023

Ausarbeitung

zur

Masterarbeit

Sommersemester 2023

bei

Erstprüfer: Prof. Dr. Tobias Harks

Zweitprüfer: Prof. Dr. Mirjam Dür

Universität Augsburg

Universität
Augsburg
University

Inhaltsverzeichnis

1 Einleitung

Diese Arbeit baut auf auf dem Artikel [4] von Graf, Harks und Sering. Dieser beschäftigt sich mit dynamischen Flüssen, insbesondere einer bestimmten Klasse von dynamischen Flüssen, die auch im Folgenden betrachtet werden, den sog. *IDE-Flüssen*. Im wesentlichen sind dies Flüsse, bei denen Fluss zu jedem Zeitpunkt nur über momentan kürzeste (auch: *aktive*) Wege geschickt wird; eine formale Definition folgt in Kapitel 2. Die Autoren von [4] liefern darin unter anderem Existenzresultate von IDE-Flüssen die für diese Arbeit fundamental sind. Im Fall von Flussnetzwerken mit mehreren ausgezeichneten Quellknoten, einer ausgezeichneten Senke und stückweise konstanten externen Einflussraten wird bewiesen, dass IDE-Flüsse stets existieren und auch terminieren, sofern dem Netzwerk endlich viel Fluss über einen endlich großen Zeitraum zugeführt wird. Des Weiteren wird mit dem *water filling* Algorithmus (siehe online Appendix zu [4]) ein Algorithmus beschrieben, der solche Flüsse direkt berechnet. Durch Ausweiten des Modells, sodass auch mehrere Senken als zulässig gelten, wird diese Berechnung jedoch deutlich komplizierter: Durch die Existenz mehrerer Senken, welche wir mit unterschiedlichen Gütern identifizieren wollen, ergeben sich für die Kanten auch spezifische, gutsabhängige Kostenfunktionen, welche sich untereinander gegenseitig beeinflussen. Dies hat die direkte Folge, dass der *water filling* Algorithmus nicht mehr anwendbar ist: Im Allgemeinen wird es unmöglich, alle Knoten in der Reihenfolge einer topologischen Sortierung auf dem Teilgrpahen der aktiven Kanten abzuarbeiten, da unterschiedliche Güter auch unterschiedliche aktive Kantenmengen implizieren, sodass Kreise entstehen können. Eine weitere Folge ist, dass IDE-Flüsse nicht mehr notwendigerweise terminieren, selbst wenn der Einfluss endlich ist. Hierfür wird in [4] ein Beispiel konstruiert, welches nur eine Quelle, zwei Senken und Kanten mit Kapazität und Reisedauer von jeweils 1 verwendet, in dem sich der eindeutige IDE-Fluss in sich ständig wiederholenden Kreisen verfängt und somit niemals terminiert. Weitere Einblicke über die Folgen dieser Verallgemeinerung liefert [3], welches unter anderem die Komplexität der Berechnung von IDE-Flüssen untersucht. Neben dem Beweis der NP-Schwere einiger damit verwandter Entscheidungsprobleme, ergibt diese Analyse auch, dass die Entscheidung, ob für ein gegebenes Netzwerk mit mehreren Senken ein IDE-Fluss existiert, welcher nach endlicher Zeit terminiert, ebenfalls NP-schwer ist. Diese Ergebnisse werden in Abschnitt 2.5 nochmals aufgegriffen.

Die allgemeine Existenz von IDE-Flüssen bleibt dagegen erhalten: Mit einem ähnlichen Vorgehen wie im Fall mit nur einer Senke kann dies bewiesen werden, dabei wird jedoch der Fixpunktsatz von Kakutani [6] verwendet, womit zwar die Existenz von IDE-Flüssen geklärt ist, nicht aber eine Möglichkeit, diese genau zu bestimmen. Mit Letzterem beschäftigt sich diese Arbeit. Ein erster Ansatzpunkt dafür ist das bereits erwähnte Fixpunkttheorem von Kakutani. Gäbe es eine Möglichkeit, die von Kakutani versprochenen Fixpunkte zu bestimmen, so könnten diese im Beweis von [4] direkt verwendet werden, um zu jedem Zeitpunkt eine im Sinne eines IDE-Flusses zulässige Flussaufteilung zu finden, den Fluss für ein Zeitintervall von positiver Länge zu erweitern und somit das Argument fortzusetzen und im Grenzwert einen vollständigen IDE-Fluss zu erhalten.

Beispielsweise beschäftigen sich einige Arbeiten aus den 1970er-Jahren (siehe [1, 2, 9, 10, 11, 12]) mit einem Ansatz zur Approximation von Fixpunkten basierend auf Brouwer's Theorem. Die Grundidee ist hierbei eine Überdeckung der Definitionsmenge mit Simplizes, die Einführung von Labelfunktionen auf diesen Simplizes, und die iterative Verfeinerung von vollständig gelabelten Teilsimplizes (nach der Idee von Sperner's Lemma), welche die gesuchten Fixpunkte enthalten. Die Umsetzung dieses Vorgehens scheint jedoch schwer praktikabel auf einer Menge deren Dimension nur durch die Anzahl der Kanten, multipliziert mit der Anzahl der Güter, beschränkt werden kann. Ein solcher Algorithmus angewendet auf ein größeres Beispiel benötigt dann einen zu großen Rechenaufwand.

Aktuellere Werke wie [5], [7] oder [8] verwenden Bedingungen an die Funktion deren Fixpunkte zu bestimmen sind, die eine Form von Kontraktion voraussetzen, welche für das hier untersuchte Problem nicht erfüllt ist (siehe Abschnitt 2.3).

Der in dieser Arbeit in Kapitel 3 beschriebene Algorithmus verwendet daher einen neuen Ansatz: Die Approximation von IDE-Flüssen mittels Festlegung und iterativer Verfeinerung von unteren und oberen Schranken an alle Flusswerte. Dies geschieht für alle Güter und Kanten simultan, womit das Problem des *water filling* Algorithmus im Fall der Existenz von mehreren Senken umgangen wird. Der Quellcode dieses Algorithmus ist verfügbar unter `https://github.com/johanneshage/ide-repository/tree/master/multicom%20IDEs`.

Nach der Beschreibung des Algorithmus folgen in Kapitel 4 einige Anwendungsbeispiele, anhand derer die Performance des Algorithmus eingeordnet werden soll.

2 Dynamische Flüsse

Notation 2.1. Es seien $\mathbb{N}$ die Menge der positiven natürlichen Zahlen und $\mathbb{N}_0$ die Menge der nicht-negativen ganzen Zahlen. Weiter bezeichnen $\mathbb{R}_+$, bzw. $\mathbb{R}_{\geq 0}$ die Mengen der positiven, bzw. nicht-negativen reellen Zahlen.

Das Konzept der dynamischen Flüsse basiert auf dem in der Optimierung wohlbekannten Modell von Flüssen in Netzwerken: Gegeben sei ein endlicher Digraph $G = (V, E)$, mit Knotenmenge V und Kantenmenge E. Jede Kante $e \in E$ habe eine Kapazität $\nu_e \in \mathbb{R}_+$, welche angibt, wie viel Fluss von Kante e aufgenommen werden kann. Das Problem besteht darin, eine gegebene Menge Fluss von einem ausgezeichneten Knoten $s \in V$, der Quelle, zu einem anderen Knoten $t \in V \setminus \{s\}$, der Senke, durch das Netzwerk zu navigieren und dabei die Kantenkapazitäten nicht zu übertreffen (oder auch nur zu entscheiden, ob ein solcher Fluss existiert). In diesem Modell sind die betrachteten Flüsse statisch und haben damit weniger realistische Anwendungsbereiche als die in dieser Arbeit betrachteten *dynamischen* Flüsse, welche obiges Konzept folgendermaßen erweitern:

Der Zusatz „dynamisch“ impliziert die Existenz eines Zeitparameters θ, der für einen gegebenen Zeithorizont $T \in \mathbb{R}_+$ einen sich über das Intervall $[0, T]$ erstreckenden, dynamisch veränderbaren Fluss beschreibt. Weiter geben wir für jede Kante $e \in E$ eine positive Reiselänge $\tau_e \in \mathbb{R}_+$ vor, welche die initiale Zeitspanne beschreibt, die von Flusspartikeln für die Überquerung der Kante benötigt wird. Außerdem lassen wir in diesem Modell die Überschreitung von Kantenkapazitäten durch Flusswerte zu, geben jedoch vor, dass sich im Fall einer solchen Überschreitung am Beginn der Kante eine Warteschlange bildet, die den Eintritt des Flusses in die Kante verzögert und somit die tatsächliche Reisedauer verlängert. Dies ist das deterministische Warteschlangenmodell nach Vikrey [13]. Abbildung 1 zeigt eine Veranschaulichung dessen. Gegeben eine Einflussrate von 3 in Knoten s für eine Dauer von 2 Zeiteinheiten, beschrieben durch die Funktion $u : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, ergibt sich die in der Abbildung dargestellte Situation zum Zeitpunkt $\theta = 2.5$. Aufgrund der ausreichend hohen Kapazität der Kante (s, v) kann sämtlicher Fluss die Kante überqueren ohne eine Warteschlange zu verursachen. Jedoch ergibt sich ab Zeitpunkt 1 ein Einfluss von 3 in Knoten v, welcher im selben Moment in Kante (v, t) übergeht, deren Kapazität dadurch überschritten wird. Aufgrund der stückweise konstanten Einflussraten ändern sich die Warteschlangen stückweise linear. Es ergibt sich der Aufbau einer Warteschlange mit Rate 2. Zum Zeitpunkt $\theta = 2.5$ hat diese damit eine Länge von 3 erreicht und wird bis zum Zeitpunkt 3 weiter aufgebaut bis eine Länge von 4 erreicht ist, der Einfluss in v stoppt und die Warteschlange von (v, t) mit Rate 1 abgebaut wird. Zum Zeitpunkt $\theta = 7$ ist diese vollständig abgebaut, womit sämtlicher Fluss bis $\theta = 8$ die Senke t erreicht.

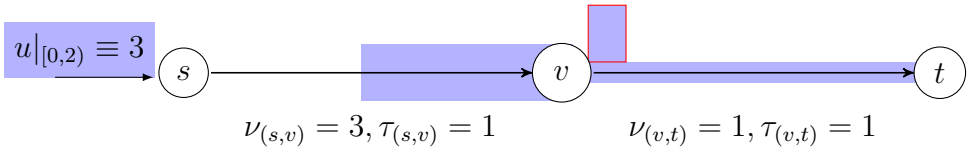


Abbildung 1: Aufbau einer Warteschlange, zum Zeitpunkt $\theta = 2.5$

Im weiteren Verlauf dieser Arbeit wird die Anzahl der Quellen und Senken nicht mehr auf 1 limitiert, stattdessen lassen wir beliebig viele solcher ausgezeichneter Knoten zu. Wir unterteilen den Fluss dabei in Güter, welche wir eindeutig mit ihrem jeweiligen Zielknoten identifizieren. Für eine Menge von Gütern der Form $I = \{1, \dots, k\}$, $k \in \mathbb{N}$, wird der Einfluss von außen in die Quellknoten für jedes Gut $i \in I$ beschrieben durch die Einflussratenfunktion $u_i : S_i \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, wobei $S_i \subseteq V$ die Menge der Quellknoten von Gut i bezeichne. Wegen des in Kapitel 1 erwähnten Beispiels eines IDE-Flusses, welcher niemals terminiert, ist es offensichtlich, dass in diesem Modell nicht mehr notwendigerweise ein endlicher Zeithorizont existieren muss.

Insgesamt wird das Tupel $(G, I, \nu, \tau, (u_i)_{i\in I}, (t_i)_{i\in I})$ bezeichnet als *Flussnetzwerk*. Für ein solches wird im folgenden Kapitel das hier beschriebene Modell formalisiert.

2.1 Das Modell

In diesem Abschnitt werden die grundlegenden Begriffe im Umgang mit dynamischen Flüssen formal eingeführt.

Definition 2.2. Ein *dynamischer Fluss* ist ein Tupel $f = (f^+, f^-)$, wobei f^+, $f^- : I \times E \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ lokal lebesgueintegrierbare Funktionen sind, welche den *Ein-* bzw. *Abfluss* $f^+_{i,e}(\theta)$, $f^-_{i,e}(\theta)$ eines Guts i zum Zeitpunkt θ in bzw. aus Kante e beschreiben. Den *Gesamtfluss* in bzw. aus einer Kante bezeichnen wir mit $f^+_e(\theta) := \sum_{i\in I} f^+_{i,e}(\theta)$, bzw. $f^-_e(\theta) := \sum_{i\in I} f^-_{i,e}(\theta)$.

Auch alle Ein- und Abflussfunktionen $f^+_{i,e}$ und $f^-_{i,e}$ sind stückweise konstant, wenn alle Einflussraten $u_{i,v}$ als stückweise konstant gegeben sind.

Definition 2.3. Der *Zu-* bzw. *Abfluss bis zum Zeitpunkt* $\theta \in \mathbb{R}_{\geq 0}$ eines Guts $i \in I$ von Kante $e \in E$ wird bezeichnet mit $F^+_{i,e}(\theta)$, bzw. $F^-_{i,e}(\theta)$ und ist definiert als

$$F^+_{i,e}(\theta) := \int_0^\theta f^+_{i,e}(\zeta)\, d\zeta, \quad F^-_{i,e}(\theta) := \int_0^\theta f^-_{i,e}(\zeta)\, d\zeta. \tag{2.1}$$

Der *Gesamtzu-* bzw. *abfluss* F^+_e und F^-_e von Kante e *bis zum Zeitpunkt* θ ist definiert als

$$F^+_e(\theta) := \sum_{i\in I} F^+_{i,e}(\theta), \quad F^-_e(\theta) := \sum_{i\in I} F^-_{i,e}(\theta). \tag{2.2}$$

Damit kann die Länge der Warteschlange von Kante e in Abhängigkeit von θ gewählt werden als

$$q_e(\theta) := F^+_e(\theta) - F^-_e(\theta + \tau_e). \tag{2.3}$$

Definition 2.4. Ein dynamischer Fluss f heißt *zulässig*, falls folgende Bedingungen erfüllt sind:

1. Für die Senken t_i gilt
$$\sum_{e \in \delta^-_{t_i}} f^-_{i,e}(\theta) - \sum_{e \in \delta^+_{t_i}} f^+_{i,e}(\theta) \geq 0 \tag{2.4}$$

2. Alle Knoten $v \neq t_i$ erfüllen die Flusserhalungsgleichung
$$\sum_{e \in \delta^+_v} f^+_{i,e}(\theta) - \sum_{e \in \delta^-_v} f^-_{i,e}(\theta) = \begin{cases} u_j(\theta), & \text{falls } v = s_j \text{ für einen Quellkonten } s_j \\ 0, & \text{sonst} \end{cases} \tag{2.5}$$

3. Die Warteschlangen werden schnellstmöglich abgebaut, d.h. für alle Kanten $e \in E$ gilt
$$f^-_e(\theta + \tau_e) = \begin{cases} \nu_e, & \text{falls } q_e(\theta) > 0 \\ \min\{f^+_e(\theta), \nu_e\}, & \text{falls } q_e(\theta) = 0 \end{cases} \tag{2.6}$$

4. Beim Durchlaufen von Warteschlangen und der Überquerung von Kanten genügt der Fluss dem FIFO-Prinzip, d.h. Güter Verlassen die Kanten in dem gleichen Verhältnis, in dem sie diese betreten haben:
$$f^-_{i,e}(\theta) = \begin{cases} f^-_e(\theta) \cdot \frac{f^+_{i,e}(\vartheta)}{f^+_e(\vartheta)}, & \text{falls } f^+_e(\vartheta) > 0 \\ 0, & \text{sonst,} \end{cases} \tag{2.7}$$
wobei $\vartheta := \min\{\vartheta \leq \theta | \vartheta + \tau_e + \frac{q_e(\vartheta)}{\nu_e} = 0\}$.

Bemerkung 2.5. Gleichung (2.6) impliziert in der Tat einen schnellstmöglichen (unter Beachtung der Kantenkapazitäten) Abbau der Warteschlangen: Zusammen mit $q'_e(\theta) = \sum_{i \in I} f^+_{i,e}(\theta) - \sum_{i \in I} f^-_{i,e}(\theta + \tau_e)$ ergibt sich
$$q'_e(\theta) = \begin{cases} f^+_e(\theta) - \nu_e, & \text{falls } q_e(\theta) > 0 \\ \max\{0, f^+_e(\theta) - \nu_e\}, & \text{falls } q_e(\theta) = 0 \end{cases}, \text{ für alle } e \in E, \theta \in \mathbb{R}_{\geq 0}. \tag{2.8}$$

2.2 Kürzeste Wege und IDE-Flüsse

Wir wollen nun eine bestimmte Variante von dynamischen Flüssen betrachten, bei der Güter in jedem Knoten immer nur entlang Kanten, die auf einem momentan kürzesten Weg zu ihrer Senke liegen, geschickt werden. Dynamische Flüsse, welche diese Eigenschaft erfüllen, werden als *IDE-Flüsse* (engl.: *Instantaneous Dynamic Equilibrium*) bezeichnet und bilden den Hauptgegenstand dieser Arbeit. Für die genaue Definition müssen wir jedoch zuerst erklären, wie die Kosten eines Weges aufzufassen sind:

Die Kantenkosten einer Kante $e \in E$ werden beschrieben durch die Funktion

$$c_e : \mathbb{R}_{\geq 0} \to \mathbb{R},\ \theta \mapsto \tau_e + \frac{q_e(\theta)}{\nu_e}, \tag{2.9}$$

wobei der letzte Teil auch als *Wartezeit* bezeichnet wird und für jedes $e \in E$ mit der Funktion

$$w_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0},\ \theta \mapsto \frac{q_e(\theta)}{\nu_e} \tag{2.10}$$

beschrieben wird. Ist dann $v \in V$ ein Knoten, $i \in I$ ein Gut und P ein v, t_i-Pfad, so berechnen sich die Kosten von P durch

$$c_P(\theta) \coloneqq \sum_{e \in P} c_e(\theta). \tag{2.11}$$

Weiter lassen sich mit Hilfe dieser Kosten Distanzen von Knoten $v \in V$ zu den Senken $t_i \in V$ definieren. Diese beschreiben wir mit den zeitabhängigen Knotenlabelfunktionen ℓ_v^i, gegeben durch:

$$\ell_v^i(\theta) \coloneqq \begin{cases} 0, & \text{falls } v = t_i \\ \min_{e=vw \in E}\{\ell_w^i(\theta) + c_e(\theta)\}, & \text{falls } t_i \text{ von } v \neq t_i \text{ in G erreichbar} \\ \infty, & \text{falls } t_i \text{ von } v \text{ aus nicht erreichbar,} \end{cases} \tag{2.12}$$

für alle $\theta \in \mathbb{R}_{\geq 0}$.

Dann heißt eine Kante $e = vw$ *aktiv* für Gut i zum Zeitpunkt θ, wenn gilt:

$$\ell_v^i(\theta) = \ell_w^i(\theta) + c_e(\theta) < \infty \tag{2.13}$$

und die Menge der für Gut i aktiven Kanten zum Zeitpunkt θ wird bezeichnet mit $E_\theta^i \subseteq E$. Im Fall dass nur ein Gut gegeben ist, bezeichnen wir die Knotenlabels auch mit ℓ_v und die Menge der aktiven Kanten mit E_θ. Weiter sei $\delta_{i,v}^+(\theta) \coloneqq \delta_v^+ \cap E_\theta^i$ die Menge der von $v \in V$ ausgehenden, momentan für Gut $i \in I$ aktiven, Kanten.

Definition 2.6. Ein zulässiger dynamischer Fluss f ist ein *IDE-Fluss*, wenn für alle $i \in I$, $\theta \in \mathbb{R}_{\geq 0}$, $e \in E$ gilt:

$$f_{i,e}^+(\theta) > 0 \Rightarrow e \in E_\theta^i. \tag{2.14}$$

In einem IDE-Fluss betreten Spieler also nur für das jeweilige Gut aktive Kanten, entscheiden sich also in jedem Knoten nur für Kanten, die auf momentan kürzesten Wegen liegen.

2.3 Existenz von IDE-Flüssen

Gegeben sei ein dynamisches Flussnetzwerk $(G, I, \nu, \tau, (u_i)_{i\in I}, (t_i)_{i\in I})$ mit rechtskonstanten Einflussratenfunktionen $u_{i,v}$, $i \in I$, $v \in S_i$. Eine Funktion $u_{i,v} : [a, b) \to \mathbb{R}$ ist

dabei *rechtskonstant*, wenn gilt:

$$\forall \theta \in [a,b) : \ \exists\, 0 < \varepsilon < b - \theta : u_{i,v}|_{[\theta,\theta+\varepsilon)} \equiv C \in \mathbb{R}. \tag{2.15}$$

Wir wollen nun das Existenzresultat für IDE-Flüsse aus [4] anführen. Die Idee dabei ist, einen gegebenen IDE-Fluss bis zum Zeitpunkt θ_k durch die richtige Wahl der aktuellen Flussaufteilung um einen Zeitraum der Länge $\alpha_k > 0$ zu erweitern und dabei die Eigenschaften eines IDE-Flusses zu erhalten. Nach der Beschreibung dessen, wie eine solche Erweiterung möglich ist, folgt ein Grenzwertargument und damit das gewünschte Resultat: Die Existenz von IDE-Flüssen unter den Voraussetzungen dieses Kapitels.

Um einen IDE-Fluss bis zum Zeitpunkt θ_k erweitern zu können, benötigen wir den Begriff von *feinen* IDE-Flüssen (aus dem Englischen: IDE thin flows), welche im Anschluss definiert werden, zunächst benötigen wir jedoch noch eine Notation für den aktuellen Einfluss in einen Knoten: Zu einem dynamischen Fluss f bezeichnet der Wert

$$b^-_{i,v}(\theta) \coloneqq \sum_{e \in \delta^-_v} f^-_{i,e}(\theta) + \mathbb{1}_{v=s_i} \cdot u_i(\theta) \tag{2.16}$$

den aktuellen Einfluss von Gut $i \in I$ in Knoten $v \in V$ zum Zeitpunkt θ. Damit können wir definieren:

Definition 2.7. Für einen gegebenen IDE-Fluss bis zum Zeitpunkt θ_k ist ein Tupel $(x,a) \in \mathbb{R}^{I\times E}_{\geq 0} \times \mathbb{R}^{I\times V}$ ein *feiner IDE-Fluss*, wenn gilt:

$$\sum_{e\in\delta^+_v} x_{i,e} = b^-_{i,v}(\theta_k) \qquad \text{f.a. } i \in I, v \in V \setminus \{t_i\}, \tag{2.17}$$

$$x_{i,e} = 0 \qquad \text{f.a. } i \in I, e \in E \setminus E^i_{\theta_k}, \tag{2.18}$$

$$a_{i,t_i} = 0 \qquad \text{f.a. } i \in I, \tag{2.19}$$

$$a_{i,v} = 0 \qquad \text{f.a. } i \in I, v \in V, \text{ mit } \ell^i_v(\theta_k) = \infty, \tag{2.20}$$

$$a_{i,v} = \min_{e=(v,w)\in E^i_{\theta_k}} \frac{g_e(\sum_{j\in I} x_{j,e})}{\nu_e} + a_{i,w} \qquad \text{f.a. } i \in I, v \in V \setminus \{t_i\}, \text{ mit } \ell^i_v(\theta_k) < \infty, \tag{2.21}$$

$$a_{i,v} = \frac{g_e(\sum_{j\in I} x_{j,e})}{\nu_e} + a_{i,w} \qquad \text{f.a. } i \in I, e = (v,w) \in E^i_{\theta_k} \text{ mit } x_{i,e} > 0, \tag{2.22}$$

wobei

$$g_e(x_e) \coloneqq \begin{cases} x_e - \nu_e, & \text{falls } q_e(\theta_k) > 0 \\ \max\{x_e - \nu_e,\ 0\}, & \text{falls } q_e(\theta_k) = 0, \end{cases} \tag{2.23}$$

die aktuelle Änderungsrate der Warteschlange von Kante e bei Gesamtzufluss x_e beschreibt. Wir nennen dann auch x die *Flussaufteilung* und a die *Steigung* zum Zeitpunkt θ_k.

Bemerkung 2.8. Es sei bemerkt, dass mit (2.20) und (2.21) für jedes Gut $i \in I$ und jeden Knoten $v \in V \setminus \{t_i\}$ unterschieden wird, ob $\ell_v^i(\theta_k) < \infty$ oder $\ell_v^i(\theta_k) = \infty$. Nach der Definition in (2.12) differenziert dies die Knoten gerade danach, ob von ihnen aus t_i über Kanten in E erreichbar ist oder nicht. Nach der Definition von aktiven Kanten (2.2) und von den Mengen $E_{\theta_k}^i$ existiert ein v, t_i - Pfad in E genau dann, wenn ein v, t_i - Pfad in $E_{\theta_k}^i$ existiert. Bedingung (2.20) dient alleinig dem Zweck der Wohldefiniertheit und der Eindeutigkeit der $a_{i,v}$ - Werte. Da die Werte aus dieser Bedingung im weiteren Verlauf keine Rolle spielen, ist auch die Wahl des Werts 0 nicht von tieferer Bedeutung.

Für die Existenz von feinen IDE-Flüssen verwenden wir den Fixpunktsatz von Kakutani, erstmals veröffentlicht in [6], in der Version aus [4].

Theorem 2.9 (Fixpunktsatz von Kakutani, [4])**.** Sei $\emptyset \neq K \subseteq \mathbb{R}^n$, mit $n \in \mathbb{N}$, eine kompakte und konvexe Menge. Sei weiter $\Gamma : K \to \mathcal{P}(K)$ eine Funktion in die Potenzmenge von K, sodass für jedes $x \in K$ die Menge $\Gamma(x)$ nichtleer und konvex ist und die Menge $\{(x, y) | x \in K, y \in \Gamma(x)\}$ abgeschlossen ist. Dann existiert ein Fixpunkt x^* von Γ, d.h. $x^* \in \Gamma(x^*)$.

Wir erhalten:

Lemma 2.10 ([4])**.** Es seien $q(\theta_k) \in \mathbb{R}_{\geq 0}^E$ ein Vektor mit möglichen Warteschlangen, $E_{\theta_k}^i \subseteq E$ für jedes $i \in I$ eine azyklische Menge und $b^-(\theta_k) \in \mathbb{R}_{\geq 0}^{I \times V}$ beschreibe alle aktuellen Einflussraten. Weiter gelte für jedes Gut $i \in I$ und jeden Knoten $v \in V$ mit $b_{i,v}^-(\theta_k) > 0$, dass die Senke t_i von v aus mittels Kanten aus $E_{\theta_k}^i$ erreichbar ist. Dann existiert ein feiner IDE-Fluss (x, a).

Beweis. Es sei $x \in \mathbb{R}_{\geq 0}^{I \times E}$ ein Vektor, welcher die Bedingungen (2.17) und (2.18) erfülle. Ein solcher Vektor existiert wegen der Voraussetzung , dass das gegebene Flussnetzwerk sinnvoll ist in dem Sinne, dass für jeden Knoten $v \in V$ mit $b_{i,v}^-(\theta_k) > 0$ immer ein v, t_i - Pfad in $E_{\theta_k}^i$ existiert. Zu diesem Vektor lassen sich Knotenlabels $a \in \mathbb{R}_{\geq 0}^{I \times V}$ bestimmen, die die Bedingungen (2.19) und (2.21) erfüllen: Die Existenz solcher Labels folgt aus der Voraussetzung, dass für jedes relevante Paar $i \in I$, $v \in V$ ein v, t_i - Pfad aus Kanten in $E_{\theta_k}^i$ existiert. Da für jedes Gut $i \in I$ der von $E_{\theta_k}^i$ induzierte Teilgraph azyklisch ist, sind diese Labels sogar eindeutig: Bestimme für jedes Gut $i \in I$ eine topologische Sortierung auf diesem Teilgraphen mit invertierten Kantenausrichtungen, beginnend bei t_i. Setzen wir dann $a_{i,t_i} = 0$ und durchlaufen anschließend die Knoten in Reihenfolge ihrer topologischen Sortierung, so können wir iterativ die relevanten $a_{i,v}$ - Werte passend wählen. Danach sind nur noch die (i, v) - Paare aus (2.20) übrig, welche wir der Vollständigkeit halber auf 0 setzen.

Es bleibt zu zeigen, dass die gewählten a - Werte auch Bedingung (2.22) erfüllen. Dazu verwenden wir Theorem 2.9:

Es sei K die Menge der Vektoren x, die die Bedingungen (2.17) und (2.18) erfüllen, also

$$K := \left\{ x \in \mathbb{R}_{\geq 0}^{I \times E} \;\middle|\; \begin{array}{rll} \sum_{e \in \delta_v^+} x_{i,e} & = b_{i,v}^-(\theta_k), & \text{für alle } i \in I, v \in V \setminus \{t_i\} \\ x_{i,e} & = 0, & \text{für alle } i \in I, e \in E \setminus E_{\theta_k}^i \end{array} \right\}. \quad (2.24)$$

Offensichtlich ist K konvex, kompakt und nicht-leer.

Wir definieren die Funktion $\Gamma : K \to \mathcal{P}(K)$ folgendermaßen:

$$\Gamma(x) \coloneqq \left\{ y \in K : \; y_{i,e} = 0 \text{ für alle } e \in E^i_{\theta_k} \text{ mit } a_{i,v} < \frac{g_e\left(\sum_{j\in I} x_{j,e}\right)}{\nu_e} + a_{i,w} \right\}, \tag{2.25}$$

wobei a die nach obiger Argumentation eindeutigen zu $x \in K$ gehörenden Labels sind. Die Konvexität der Menge $\Gamma(x)$ ist wieder trivialerweise erfüllt und außerdem ist $\Gamma(x)$ nicht-leer, denn für jedes i, v - Paar aus Bedingung (2.21) existiert eine von v ausgehende Kante $e = (v, w)$ über die $a_{i,v}$ definiert wurde und für die daher gilt: $a_{i,v} = g_e(\sum_{j\in I} x_{j,e})/\nu_e + a_{i,w}$. Somit liegt ein y, welches alle $b^-_{i,v}(\theta_k)$ Flusseinheiten in diese Kante schickt (und das für jedes i, v - Paar), in der Menge $\Gamma(x)$.

Für die Abgeschlossenheit der Menge $\{(x, y)|x \in K, y \in \Gamma(x)\}$ sei nun $(x^n, y^n)_{n\in\mathbb{N}}$ eine Folge in dieser Menge, welche in $\mathbb{R}^{I\times E} \times \mathbb{R}^{I\times E}$ konvergiert. Wegen der Folgenkompaktheit von K konvergieren $(x^n)_{n\in\mathbb{N}}$ und $(y^n)_{n\in\mathbb{N}}$ bereits gegen Punkte $x, y \in K$. Es sei dann $(a^n)_{n\in\mathbb{N}}$ die Folge von Labelfunktionen zu den Folgengliedern $(x^n)_{n\in\mathbb{N}}$ und zusätzlich seien a die zu x gehörenden Labels. Da die Berechnung der Knotenlabels zu einer Flussaufteilung nach Obigem eindeutig ist und die Funktionen g_e und min stetig sind, ist auch die Funktion $\Phi : \mathbb{R}^{I\times E}_{\geq 0} \to \mathbb{R}^{I\times V}$, $x \mapsto a$ wohldefiniert und stetig. Wir erhalten damit $\lim_{n\to\infty} a^n = a$. Um zu zeigen, dass $y \in \Gamma(x)$, nehmen wir an es gebe ein Gut $i \in I$ und eine Kante $e = (v, w) \in E^i_{\theta_k}$ mit $y_{i,e} > 0$ und $a_{i,v} < g_e(\sum_{j\in I} x_{j,e})/\nu_e + a_{i,w}$. Mit der Stetigkeit von g_e folgt, dass ein $n_0 \in \mathbb{N}$ existiert, sodass $y^n_{i,e} > 0$ und $a^n_{i,v} < g_e(\sum_{j\in I} x^n_{j,e})/\nu_e + a^n_{i,w}$ für alle $n \geq n_0$. Dies steht im Widerspruch zu $y^n \in \Gamma(x^n)$. Damit ist Theorem 2.9 anwendbar und liefert die Existenz eines Fixpunkts $x^* \in K$ von Γ. Sind dann a^* die zugehörigen Knotenlabels, so bildet (x^*, a^*) einen feinen IDE-Fluss. □

Damit ist die Existenz von feinen IDE-Flüssen geklärt und nun sollen diese verwendet werden, um auch die Existenz von IDE-Flüssen allgemein nachzuweisen. Gegeben sei dazu ein IDE-Fluss f bis zum Zeitpunkt $\theta_k \in \mathbb{R}_{\geq 0}$ mit rechtskonstanten Einflussraten $f^+_{i,e}$, zusammen mit einem feinen IDE-Fluss (x, a). Letzterer wird nun verwendet, um f auf einen größeren Zeitraum zu erweitern. Es sei $\alpha \in \mathbb{R}_+$ die Größe des gewünschten Zeitschritts und für alle $i \in I$, $e \in E$ und $v \in V$ seien

$$f^+_{i,e}(\theta_k + \xi) \coloneqq x_{i,e}, \quad \ell^i_v(\theta_k + \xi) \coloneqq \ell^i_v(\theta_k) + \xi \cdot a_{i,v}, \quad \text{für alle } \xi \in [0, \alpha) \tag{2.26}$$

die erweiterten Einflussraten und Distanzen zu den Zielknoten. Um einen zulässigen Fluss zu erhalten werden die Abflussraten $f^-_{i,e}(\theta_k)$ so erweitert wie von den Zulässigkeitsbedingungen (2.6) und (2.7) diktiert. Damit ergeben sich die Warteschlangenlängen nach (2.3). Wir bezeichnen diese Erweiterung als α-*Erweiterung* und das Intervall $[\theta_k, \theta_k + \alpha)$ als *Phase*. Damit f nach der α-Erweiterung einen IDE-Fluss bis zum Zeitpunkt $\theta_k + \alpha$ bildet, darf α natürlich nicht beliebig groß gewählt werden, sondern so, dass folgende Eigenschaften erfüllt sind:

1. Warteschlangen bleiben nicht-negativ, d.h. der Abbau wird beschränkt durch:

$$q_e(\theta_k) + \alpha \cdot \left(\sum_{j \in I} x_{j,e} - \nu_e \right) \geq 0, \quad \text{für alle } e \in E \text{ mit } q_e(\theta_k) > 0 \tag{2.27}$$

2. Das Aktivwerden einer inaktiven Kante beendet die Phase, es gilt also für alle $i \in I$ und $e = (v, w) \in E \setminus E^i_{\theta_k}$:

$$\ell^i_v(\theta_k) + \alpha \cdot a_{i,v} \leq \tau_e + \frac{q_e(\theta_k)}{\nu_e} + \alpha \cdot \frac{g_e\left(\sum_{j \in I} x_{j,e}\right)}{\nu_e} + \ell^i_w(\theta_k) + \alpha \cdot a_{i,w} \tag{2.28}$$

3. Die aktuellen Einflusswerte aller Knoten bleiben die gesamte Phase über konstant, d.h.:

$$b^-_{i,v}(\theta_k + \xi) = b^-_{i,v}(\theta_k), \quad \text{für alle } i \in I, v \in V \setminus \{t_i\}, \xi \in [0, \alpha) \tag{2.29}$$

Ein $\alpha > 0$ welches diese drei Forderungen erfüllt, wird als *zulässig* bezeichnet. Offensichtlich existiert immer ein α_1, welches (2.27) erfüllt. Wegen $\ell^i_v(\theta_k) < \tau_e + \frac{q_e(\theta_k)}{\nu_e} + \ell^i_w(\theta_k$ für alle $i \in I, e = (v, w) \in E \setminus E^i_{\theta_k}$ existiert auch ein α_2, welches (2.28) erfüllt. Schließlich existiert auch ein α_3, für das (2.29) gilt, denn da alle f^-_e und u_i rechtskonstant, sind auch alle $b^-_{i,v}$ rechtskonstant. Der kleinste Wert von $\alpha_1, \alpha_2, \alpha_3$ ist damit zulässig, es existiert also stets ein zulässiges α.

Bemerkung 2.11. Wegen (2.28) ändert sich innerhalb einer zulässigen Phase α zwar nicht der Aktivitätsstatus inaktiver Kanten, jedoch können aktive Kanten durchaus inaktiv werden: Durch den zum Zeitpunkt θ_k gewählten feinen IDE-Fluss kann eine Kante $e \in E^i_{\theta_k}$ sofort inaktiv werden und somit $e \notin E^i_{\theta_k + \xi}$ für alle $\xi \in (0, \alpha)$. Dies kommt zustande durch Flusswerte in anderen Knoten als dem Startknoten von e, oder durch die Flusswerte anderer Güter von Kanten mit gleichem Startknoten. Damit gilt sicher $x_{i,e} = 0$ für eine solche Kante.

Nachstehendes Lemma begründet die Relevanz von α-Erweiterungen für unsere Zwecke.

Lemma 2.12 ([4])**.** Es seien f ein IDE-Fluss bis zum Zeitpunkt θ_k, (x, a) ein feiner IDE-Fluss zum Zeitpunkt θ_k und $\alpha > 0$ zulässig. Dann liefert die α-Erweiterung nach (2.26) einen IDE-Fluss bis zum Zeitpunkt $\theta_{k+1} := \theta_k + \alpha$ und die erweiterten Distanzfunktionen ℓ geben die korrekten Längen der kürzesten Wege an.

Beweis. Die Zulässigkeitsbedingungen (2.6) und (2.7) sind nach Definition erfüllt. Die Flusserhaltung (2.5) ergibt sich durch die Rechnung:

$$\begin{aligned} \sum_{e \in \delta^+_v} f^+_{i,e}(\theta_k + \xi) = \sum_{e \in \delta^+_v} x_{i,e} = b^-_{i,v}(\theta_k) = b^-_{i,v}(\theta_k + \xi) = \\ = \sum_{e \in \delta^-_v} f^-_{i,e}(\theta_k + \xi) + \mathbb{1}_{v = s_i} \cdot u_i(\theta_k + \xi), \end{aligned}$$

für alle $i \in I, v \in V \setminus \{t_i\}$ und $\xi \in [0, \alpha)$. Schließlich gilt auch (2.4) wegen Eigenschaft (2.18) von feinen IDE-Flüssen und da Kanten in $\delta^+_{t_i}$ niemals aktiv sein können für Gut i.

Um zu zeigen, dass auch die erweiterten Labels ℓ Bedingung (2.12) erfüllen, betrachten wir für ein $\xi \in [0, \alpha)$ und eine Kante $e \in E$ die Änderung der Warteschlange (2.8) zum Zeitpunkt $\theta_k + \xi$:

$$\begin{aligned} q'_e(\theta_k + \xi) &= \begin{Bmatrix} f^+_e(\theta_k + \xi) - \nu_e, & \text{falls } q_e(\theta_k + \xi) > 0 \\ \max\{f^+_e(\theta_k + \xi) - \nu_e, 0\}, & \text{falls } q_e(\theta_k + \xi) = 0 \end{Bmatrix} \\ &= g_e\left(f^+_e(\theta_k + \xi)\right) = g_e\left(\sum\nolimits_{j \in I} x_{j,e}\right). \end{aligned}$$

Es ist also q'_e konstant auf dem gesamten Intervall $[\theta_k, \theta_k + \alpha)$ und somit gilt

$$q_e(\theta_k + \xi) = q_e(\theta_k) + \xi \cdot g_e\left(\sum_{j \in I} x_{j,e}\right), \quad \text{für alle } \xi \in [0, \alpha). \tag{2.30}$$

Da α zulässig, folgt mit (2.28) für alle $i \in I$, $e = (v, w) \in E \setminus E^i_{\theta_k}$:

$$\begin{aligned} \ell^i_v(\theta_k + \xi) &= \ell^i_v(\theta_k) + \xi \cdot a_{i,v} \\ &\overset{(2.28)}{\leq} \tau_e + \frac{q_e(\theta_k)}{\nu_e} + \xi \cdot \frac{g_e\left(\sum_{j \in I} x_{j,e}\right)}{\nu_e} + \ell^i_w(\theta_k) + \xi \cdot a_{i,w} \\ &\underset{(2.30)}{\overset{(2.26)}{=}} \tau_e + \frac{q_e(\theta_k + \xi)}{\nu_e} + \ell^i_w(\theta_k + \xi). \end{aligned}$$

Für die restlichen Kanten $i \in I$, $e = (v, w) \in E^i_{\theta_k}$ gilt:

$$\begin{aligned} \ell^i_v(\theta_k + \xi) &= \ell^i_v(\theta_k) + \xi \cdot a_{i,v} \\ &\underset{(2.21)}{\overset{(2.12)}{\leq}} \tau_e + \frac{q_e(\theta_k)}{\nu_e} + \ell^i_w(\theta_k) + \xi \cdot \left(\frac{g_e\left(\sum_{j \in I} x_{j,e}\right)}{\nu_e} + a_{i,w}\right) \\ &\underset{(2.30)}{\overset{(2.26)}{=}} \tau_e + \frac{q_e(\theta_k + \xi)}{\nu_e} + \ell^i_w(\theta_k + \xi) \end{aligned}$$

und es existiert für jedes Gut i eine Kante $e^* = (v^*, w^*) \in E^i_{\theta_k}$, für die (2.21) mit Gleichheit erfüllt ist und da e^* aktiv, gilt auch $\ell^i_{v^*}(\theta_k) = \ell^i_{w^*}(\theta_k) + \tau_{e^*} + q_{e^*}(\theta_k)/\nu_{e^*}$ und somit Gleichheit in obiger Rechnung, womit (2.12) auch für die erweiterten Labels erfüllt ist.

Der erweiterte Fluss f ist also zulässig mit wohldefinierten Knotenlabels ℓ und sogar ein IDE-Fluss bis zum Zeitpunkt $\theta_{k+1} = \theta_k + \alpha$, denn für alle $i \in I$, $\xi \in [0, \alpha)$ und Kanten $e = (v, w) \in E$ gilt:

$$f^+_{i,e}(\theta_k + \xi) > 0 \;\Leftrightarrow\; x_{i,e} > 0$$

und somit

$$a_{i,v}(\theta_k+\xi) = a_{i,v} \overset{(2.22)}{=} \frac{g_e\left(\sum_{j\in I} x_{j,e}\right)}{\nu_e} + a_{i,w} = \frac{q_e'(\theta_k+\xi)}{\nu_e} + a_{i,w},$$

sodass auch

$$\ell_v^i(\theta_k+\xi) = \tau_e + \frac{q_e(\theta_k+\xi)}{\nu_e} + \ell_w^i(\theta_k+\xi)$$

erfüllt ist. □

Mit diesem Lemma erhalten wir das gewünschte Ergebnis:

Theorem 2.13 ([4])**.** Gegeben sei ein Netzwerk $(G, I, \nu, \tau, (u_i)_{i\in I}, (t_i)_{i\in I})$ mit einer endlichen Menge $I \subset \mathbb{N}$ an Gütern und rechtskonstanten Einflussratenfunktionen $u_{i,v}$. Dann existiert ein IDE-Fluss f mit rechtskonstanten Zuflüssen $f_{i,e}^+$.

Beweis. Es sei $\mathfrak{F}_0$ die Menge aller Tupel (f,θ) mit $\theta \in \mathbb{R}_+ \cup \{\infty\}$ und f ein IDE-Fluss bis zum Zeitpunkt θ, mit allen $f_{i,e}^+$, $f_{i,e}^-$ rechtskonstant. Setze $\hat{\theta}^0 := \sup\{\theta \mid \exists f : (f,\theta) \in \mathfrak{F}_0\}$. Gilt $\hat{\theta}^0 = \infty$ so ist die Aussage gezeigt, sei also $\hat{\theta}^0 < \infty$. Dann existiert ein IDE-Fluss f^1, sodass $(f^1,\theta^1) \in \mathfrak{F}_0$, mit $\theta^1 := \hat{\theta}^1/2$. Dann sei $\mathfrak{F}_1 := \{(f,\theta) \in \mathfrak{F}_0 \mid f\big|_{[0,\theta^1)} = f^1\}$ die Menge der Tupel (f,θ) auf die sich (f^1,θ^1) vervollständigen lässt. Diese Menge ist nicht-leer, denn sie enthält offensichtlich das Tupel (f^1,θ^1), also setze $\hat{\theta}^1 := \sup\{\theta \mid \exists f : \ (f,\theta) \in \mathfrak{F}_1\}$. Wegen Lemma 2.12 gilt $\hat{\theta}^1 > \theta^1$ und somit $\hat{\theta}^1 \in (\theta^1, \hat{\theta}^1]$. Es sei $\theta^1 := (\hat{\theta}^1 - \theta^1)/2 + \theta^1 > \theta^1$. Sukzessive ergibt sich eine streng monoton wachsende Folge $(\theta^k)_{k\in\mathbb{N}}$ und eine monoton fallende Folge $(\hat{\theta}^k)_{k\in\mathbb{N}}$ mit $\theta^k < \hat{\theta}^k$ für alle $k \in \mathbb{N}$ und durch schrittweises Auflösen folgt $\hat{\theta}^k - \theta^k \leq \hat{\theta}^0/2^k \overset{k\to\infty}{\longrightarrow} 0$, also existiert ein gemeinsamer Grenzwert θ^* für diese beiden Folgen. Betrachte dann die punktweisen Limes der Folge $(f^k)_{k\in\mathbb{N}}$ in den Punkten $\theta < \theta^*$. Dies ergibt einen Fluss f^*, mit $(f^*,\theta^*) \in \mathfrak{F}_0$. Nach Lemma 2.12 kann f^* durch ein $\varepsilon > 0$ erweitert werden, dies ist ein Widerspruch zur Definition von $\hat{\theta}^k$, für alle k mit $\hat{\theta}^k \in [\theta^*, \theta^* + \varepsilon)$. □

Zum Abschluss dieses Abschnitts folgt nun die in Kapitel 1 angekündigte Begründung, weshalb Γ nicht kontrahierend ist und demnach die Ansätze zur Fixpunktbestimmung aus [7] und [8] nicht anwendbar sind.

Es sei X ein normierter Vektorraum, $\emptyset \neq C \subseteq X$ eine Teilmenge und $\mathcal{CB}(C) \subseteq \mathcal{P}(C)$ die Menge aller abgeschlossenen und beschränkten Teilmengen von C. Weiter bezeichne für $A,\ B \in \mathcal{CB}(C)$:

$$d_H(A,B) = \max\left\{\sup_{a\in A} d(a,B),\ \sup_{b\in B} d(b,A)\right\}, \tag{2.31}$$

wobei $d(a,B) = \inf_{b\in B}\|a-b\|$, die *Hausdorff Metrik* auf $\mathcal{CB}(C)$ bezeichnet. Damit lässt sich definieren:

Definition 2.14. Es sei $T : C \to \mathcal{CB}(C)$ eine Abbildung. Dann heißt T

(a) *nichtexpansiv*, wenn gilt:

$$d_H(Tx, Ty) \leq \|x - y\|, \text{ für alle } x, y \in C \text{ und} \tag{2.32}$$

(b) *quasinichtexpansiv*, wenn gilt:

$$d_H(Tx, Tp) \leq \|x - p\|, \text{ für alle } x, p \in C \text{ mit } p \in Tp. \tag{2.33}$$

Es ist leicht festzustellen, dass Γ nicht quasinichtexpansiv und damit natürlich weder nichtexpansiv noch kontrahierend ist. Betrachte dazu folgende Situation:

Beispiel 2.15. Es sei $G = (V, E)$ der in Abbildung 2 dargestellte Digraph. Es gelte $I = \{1\}$ und beide Kantenkapazitäten und Reiselängen seien gleich 1. Weiter gebe es einen momentanen Einfluss von $u_s = 2$ Flusseinheiten im Knoten s.

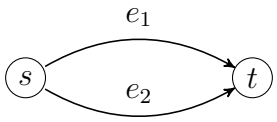


Abbildung 2: $G = (V, E)$, mit $V = \{s, t\}$, $E = \{e_1, e_2\}$

Wir erhalten damit $K = \{x \in \mathbb{R}^2_{\geq 0} : x_{e_1} + x_{e_2} = 2\}$. Die eindeutige IDE-Flussaufteilung schickt jeweils 1 Flusseinheit über beide s verlassenden Kanten und wir erhalten den eindeutigen Fixpunkt $x^* = (1, 1) \in K$ von Γ. Für alle weiteren $x^* \neq x \in K$ gilt dann:

$$\Gamma(x) = \begin{cases} \{(0, 2)\}, & \text{falls } x_{e_1} > x_{e_2}, \\ \{(2, 0)\}, & \text{falls } x_{e_1} < x_{e_2}. \end{cases} \tag{2.34}$$

Infolgedessen gilt $\Gamma : K \to \mathcal{CB}(K)$ und für alle $x^* \neq x \in K$: $d_H(\Gamma(x), \Gamma(x^*)) = \sqrt{2}$. Offensichtlich existieren $x^* \neq x \in K$ mit $\|x - x^*\| < \sqrt{2}$, weshalb Γ nicht quasinichtexpansiv sein kann.

2.4 Termination von IDE-Flüssen

Nachdem nun bekannt ist, dass unter den in dieser Arbeit untersuchten Voraussetzungen IDE-Flüsse immer existieren, stellt sich die Frage, ob diese auch terminieren, wenn wir weiter annehmen, dass die Einflussraten in ihrer Höhe beschränkt und zugleich endlich andauernd sind, es existiere also ein Zeitpunkt $\theta_0 > 0$, sodass für alle Güter $i \in I$ gelte: $\text{supp}(u_i) \subseteq [0, \theta_0)$. Wir definieren:

Definition 2.16. 1. Die Funktionen

$$G_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0},\ \theta \mapsto F_e^+(\theta) - F_e^-(\theta) \tag{2.35}$$

für $e \in E$ heißen *Kantenlast* und bezeichnen die Menge an Fluss auf Kante e zum Zeitpunkt θ.

2. Der *Gesamtfluss* $G(\theta)$ eines Netzwerks zum Zeitpunkt θ wird beschrieben durch

$$G(\theta) = \sum_{e \in E} G_e(\theta). \tag{2.36}$$

Definition 2.17. Ein dynamischer Fluss f *terminiert*, falls es einen Zeitpunkt $\hat{\theta} \geq \theta_0$ gibt, zu dem der Gesamtfluss des Netzwerks gleich 0 ist, d.h. $G(\hat{\theta}) = 0$. Wir sagen dann, f *terminiert bevor* $\hat{\theta}$.

Bemerkung 2.18. Im Fall von IDE-Flüssen wird eine Verringerung des Gesamtflusses genau dann erzielt, wenn Güter ihre jeweilige Senke erreichen. Ein IDE-Fluss terminiert also, sobald sämtliche Güter ihre Senke erreicht haben.

Die benötigen Resultate über die Termination von IDE-Flüssen stammen aus [4] und werden hier lediglich (ohne Beweis) angegeben:

Lemma 2.19 ([4])**.** Es sei G ein azyklischer Graph und die Einflussfunktionen u_j seien beschränkt und endlich andauernd. Dann terminiert jeder zulässige dynamische Fluss.

Theorem 2.20 ([4])**.** Für multi-source single-sink Netzwerke terminiert jeder IDE-Fluss f mit beschränkten, endlich andauernden Einflussraten u_i.

Theorem 2.21 ([4])**.** Es existiert ein Netzwerk mit genau einer Quelle und zwei Senken, in dem jeder IDE-Fluss nicht terminiert.

Bemerkung 2.22. Zum Beweis von Theorem 2.21 wird in [4] ein Beispiel konstruiert, in dem sich der eindeutige IDE-Fluss an verschiedenen Stellen innerhalb des Graphen in Kreisen verfängt, welche sich beliebig oft wiederholen. Insbesondere erreicht kein Fluss jemals eine der angestrebten Senken. Dieses Beispiel wird erneut aufgegriffen und genauer analysiert in Abschnitt 4.2. Von besonderer Relevanz ist Theorem 2.21 für diese Arbeit deshalb, weil es die Auswirkungen des Vorhandenseins mehrerer Zielknoten verdeutlicht: Die Verwendung verschiedener Kostenfunktionen, je nach Gut, kann dazu führen, dass die Güter, welche stets eigensinnig handeln, in Wechselwirkung miteinander einen Fluss erzeugen, der sich im schlimmsten Fall sogar über einen beliebig langen Zeitraum erstreckt.

2.5 Komplexität

In diesem Abschnitt werden einige für diese Arbeit interessante Ergebnisse aus [3] über die Komplexität der Berechnung von IDE-Flüssen zusammengetragen. Wir beginnen mit folgendem Theorem:

Theorem 2.23 ([3])**.** Die Größe der Ausgabe bei der Berechnung von IDE-Flüssen ist nicht-polynomiell in der Größe der Eingabeinstanz, selbst wenn diese auf Periodizität untersucht und gegebenenfalls vereinfacht wird. Dies gilt sogar schon für serien-parallele Graphen.

Die Argumentation der Autoren in [3] verwendet für den Beweis dieses Theorems eine entsprechende Beispielinstanz. Der darin bis auf Nullmengen eindeutige IDE-Fluss umfasst eine in der Codierungsgröße der Eingabeinstanz nicht-polynomielle Anzahl an Phasen. Da ein jeder Algorithmus zur Berechnung von IDE-Flüssen für dieses Beispiel mit Sicherheit alle Phasen in seiner Ausgabe enthalten muss, ist mit dieser Untergrenze an die Ausgabegröße auch eine Untergrenze an die *worst case* Laufzeit gegeben. Dies gilt genauso für jeden Algorithmus, der eine sinnvolle Approximation eines IDE-Flusses berechnen soll. Es wird außerdem zitiert:

Theorem 2.24 ([3])**.** Die folgenden Entscheidungsprobleme sind NP-schwer:

(i) Gegeben seien ein Netzwerk und eine enthaltene Kante: Gibt es einen IDE-Fluss, welcher diese Kante nicht verwendet?

(ii) Gegeben seien ein Netzwerk und eine enthaltene Kante: Gibt es einen IDE-Fluss, welcher diese Kante verwendet?

(iii) Gegeben seien ein Netzwerk und ein Zeithorizont T: Gibt es einen IDE-Fluss, welcher vor T terminiert?

(iv) Gegeben seien ein Netzwerk und ein $k \in \mathbb{N}$: Gibt es einen IDE-Fluss, welcher höchstens k Phasen beinhaltet?

Während also das Problem zu entscheiden, ob zu einem gegebenen Netzwerk ein IDE-Fluss existiert, nach Abschnitt 2.3 immer trivial ist, zeigt dagegen dieses Theorem die Existenz einiger nah verwandter Entscheidungsprobleme auf, die zu den algorithmisch am schwierigsten zu lösenden Problemen der Komplexitätsklasse *NP* gehören.

Damit ergibt sich ein weiteres Argument, weshalb Algorithmen zur Berechnung der Gesamtheit aller IDE-Flüsse im schlimmsten Fall sicher nicht-polynomiell sind: Andernfalls könnten alle Probleme in Theorem 2.24 effizient entschieden werden. Mit dem im nächsten Kapitel vorgestellten Algorithmus wird vorerst nur beabsichtigt, zu einem gegebenen Flussnetzwerk *einen* IDE-Fluss zu bestimmen.

3 Approximation von IDE-Flüssen

Es soll nun ein Algorithmus vorgestellt werden, der auf dem Ansatz des vorigen Kapitels basiert und dessen Ziel es ist, IDE-Flüsse in Netzwerken mit mehreren Senken zu approximieren. Das Grundkonzept ist dargestellt in Algorithmus 1.

Algorithmus 1: `main()`

1 **Eingabe:** Netzwerk $(G, I, \nu, \tau, (u_i)_{i\in I}), (t_i)_{i\in I})$, maximaler Zeitpunkt $T \in \mathbb{R}_+$, bis zu dem der Fluss berechnet werden soll.
2 **Ausgabe:** Approximation f eines IDE-Flusses bis zum Zeitpunkt T.
3 Methode: Initialisiere $f = (f^+, f^-)$ als den Nullfluss, $q(0) = 0$, $c_e(0) = \tau_e$, für alle $e \in E$, und die Labels $\ell_v^i(0)$, für alle $i \in I$, $v \in V$ mit dem Algorithmus von Dijkstra, sowie die korrespondierenden Mengen E_0^i der aktiven Kanten, für alle $i \in I$;
4 $\theta \leftarrow 0$;
5 **while** $\theta < T$ **do**
6 Prüfe, ob vorige Flussaufteilung x, a noch gültig und verwende diese gegebenenfalls (nicht im Fall $\theta = 0$) ;
7 Sonst x, $a \leftarrow$ `fp_approx(`θ`)` ; `// Algorithmus 2`
8 Aktualisiere f^+, f^- und bestimme Schrittweite α ;
9 Setze $q_e(\theta + \alpha) \leftarrow q_e(\theta) + g_e(f_e^+(\theta)) \cdot \alpha$, für alle Kanten $e \in E$;
10 Setze $c_e(\theta + \alpha) \leftarrow \frac{q_e(\theta+\alpha)}{\nu_e} + \tau_e$, für alle Kanten $e \in E$;
11 Setze $\ell_v^i(\theta + \alpha) \leftarrow \ell_v^i(\theta) + \alpha \cdot a_{i,v}$, für alle $i \in I$, $v \in V$;
12 Bestimme alle $E_{\theta+\alpha}^i$;
13 Überarbeite Distanzwerte mit `refine()` ; `// Algorithmus 8`
14 $\theta \leftarrow \theta + \alpha$;
15 **end while**
16 **return:** $f = (f^+, f^-)$;

Der Parameter T ist dabei als ein erzwungener Zeithorizont zu interpretieren. Dieser liefert ein zusätzliches Abbruchkriterium, sodass im Fall in dem der berechnete IDE-Fluss nicht terminiert, zumindest der Algorithmus ein Ende findet. Die Ausgabe ist dann die Approximation eines IDE-Flusses bis zum Zeitpunkt T.

Das Herzstück des Algorithmus bildet die Berechnung (der Approximation) der IDE-Flussaufteilung zu jedem Iterationszeitpunkt θ. Dies geschieht zum Einen in Zeile **7**, mit der später in Abschnitt 3.1 genauer beschriebenen Funktion `fp_approx(`θ`)`. Dies ist auch der rechenaufwendigste Teil des Algorithmus, weshalb dieser auch nur dann ausgeführt werden soll, wenn dies wirklich notwendig ist. Um dies zu gewährleisten wird in Zeile **6** getestet, ob die Flussaufteilung der vorigen Iteration immer noch im Sinne eines IDE-Flusses ist. In zwei der drei Fälle (2.27) - (2.29), welche eine neue Phase einleiten, ist dies ein realistisches Szenario: Zum Einen können Kanten aktiv werden, ohne eine Veränderung der IDE-Flussaufteilung zu erzwingen und genauso können Warteschlangen 0 werden, ohne eine Änderung der Flusswerte mit sich zu ziehen. Dennoch muss in jedem der beiden Fälle eine neue Phase beginnen und somit auch eine neue Iteration des Algorithmus.

Im Anschluss daran wird die Schrittweite α bestimmt, welche so groß wie möglich und gleichzeitig zulässig sein soll. Mit dem Vorgehen dazu beschäftigt sich Abschnitt 3.2. Damit können dann die Ein- und Abflussraten f^+, f^-, die Kantenkosten c, die Warteschlangen q, die Knotendistanzfunktionen ℓ und die Mengen der aktiven Kanten $E^i_{\theta+\alpha}$ für alle Güter $i \in I$ aktualisiert werden.

3.1 Die Berechnung der Flussaufteilung zum Zeitpunkt θ

Die nun präsentierte Funktion `fp_approx()` liefert das Kernstück des behandelten Algorithmus 1. Die generelle Idee dahinter ist inspiriert von der Funktion Γ (2.25): Gegeben eine Flussaufteilung x^k mit Knotenlabels a^k, versucht der Algorithmus für jedes Gut zu beurteilen, ob aus Sicht des jeweiligen Gutes die Flussaufteilung je Knoten im Sinne eines IDE-Flusses ist. Zum einen Teil ist dies die Entscheidung, die in Γ getroffen wird. Gilt für ein Gut $i \in I$ und eine aktive Kante $e = (v, w) \in E^i_\theta$, dass $a^k_{i,v} < g_e\left(\sum_{j\in I} x^k_{j,e}\right)/\nu_e + a^k_{i,w}$, so sagen wir, dass x^k *aus Sicht von i zu viel Fluss über e* schickt. Dabei ist zu beachten, dass nicht notwendigerweise Gut i selbst zu viel über e schickt, sonder nur, dass der Gesamtfluss x_e aus Sicht von i zu hoch ist. Trotzdem reagiert der Algorithmus in einem solchen Fall, indem eine neue Aufteilung x^{k+1} konstruiert wird, mit $x^{k+1}_{i,e} < x^k_{i,e}$ oder $x^k_{i,e} = 0 = x^{k+1}_{i,e}$. Dies geschieht für alle Startknoten und Güter gleichzeitig, womit eine Verringerung des Fehlers erhofft ist. Mit „Fehler“ wird hier die Abweichung der aktuellen Flussaufteilung x^k von der Aufteilung des zu approximierenden feinen IDE-Flusses bezeichnet. Eine genauere Definition und Analyse dieses Fehlers werden in Abschnitt 4.1 behandelt.

Es existieren also i, e-Paare, für die zu viel Fluss geschickt wird, weswegen wir auch davon ausgehen wollen, dass für andere Paare j, $e' = (v, w')$, mit gleichem Startknoten, *zu wenig* Fluss verschickt wird. Wir sagen, x^k schickt *aus Sicht von Gut j zu wenig Fluss* über Kante e', wenn gilt:

- (x^k, a^k) erfüllen die Bedingungen (2.17) - (2.21) eines feinen IDE-Flusses, aber die Bedingung (2.22) ist verletzt, d.h. es existieren ein Gut $j' \in I$ und eine Kante $e'' = (v, w'') \in \delta^+_{j',v}(\theta)$, mit $x^k_{j',e''} > 0$ und $a^k_{j',v} < \frac{g_{e''}(x^k_{e''})}{\nu_{e''}} + a^k_{j',w''}$.
- $e' \in \arg\min_{e=(v,w)\in\delta^+_{j,v}(\theta)} \frac{g_e(x^k_e)}{\nu_e} + a^k_{j,w}$.

Analog folgern wir für solche Paare: $x^{k+1}_{j,e'} > x^k_{j,e'}$, oder $x^k_{j,e'} = b^-_{j,v}(\theta) = x^{k+1}_{j,e'}$.

Algorithmus 2 setzt dies um durch die Verwendung oberer und unterer Schranken für die einzelnen Flusswerte. Dazu bezeichnen $lb^k_{i,e} \in \mathbb{R}_{\geq 0}$, bzw. $ub^k_{i,e} \in \mathbb{R}_{\geq 0}$ die untere, bzw. obere Schranke an den Flusswert $x^k_{i,e}$. Erreicht ein Schrankenpaar für ein Gut i und eine Kante $e = (v, w)$ eine Genauigkeit, welche mit dem Eingabeparameter $\epsilon \in \mathbb{R}_+$ reguliert werden kann, so bleibt der Wert $x^k_{i,e}$ vorläufig unverändert. In diesem Fall sagen wir, Kante e wurde für Gut i *fixiert*. Dann sei $F^i_\theta \subseteq E^i_\theta$ die Menge der aktuell für Gut i fixierten Kanten. Weiter bezeichne

$$b^{nf}_{i,v}(\theta) := b^-_{i,v}(\theta) - \sum_{e\in\delta^+_{i,v}\setminus F^i_\theta} x^k_{i,e} \tag{3.1}$$

den aktuellen, nicht bereits fixierten, Einfluss von Gut i in Knoten v. Dieser Wert ist zu Beginn der Iteration für alle $i \in I$, $v \in V$ initialisiert als $b^-_{i,v}(\theta)$ und wird jedes Mal aktualisiert, wenn ein neuer Flusswert fixiert wird.

Die Liste *coms* enthält alle Knoten $v \in V$, für die $b^{nf}_{i,v}(\theta) > 0$ für mindestens ein $i \in I$ gilt. Weiterhin sei für einen solchen Knoten $v \in coms$ mit $coms_v \subseteq I$ die Menge aller Güter bezeichnet, für die diese Eigenschaft erfüllt ist. Während diese Listen fortwährend zusammen mit b^{nf} aktualisiert werden, beschreibt die Funktion

$$\sigma_\theta : V \to \mathcal{P}(I) \tag{3.2}$$

für jeden Knoten $v \in V$, welche Güter sich generell zum Zeitpunkt θ in v befinden, gibt also die Menge aller $i \in I$ an, für die $b^-_{i,v}(\theta) > 0$ erfüllt ist.

Ergibt sich der Fall, dass $b^{nf}_{i,v}(\theta) \approx 0$[1], so sind alle Kanten mit Startknoten v für Gut i fixiert. Sind die fixierten Flusswerte realistisch, d.h. gilt für alle Kanten $e \in \delta^+_{i,v}(\theta)$ mit positivem Wert $x^k_{i,e}$, dass der Wert $g_e(x^k_e)/\nu_e + a^k_{i,w}$ in einem Toleranzbereich[2] um $a^k_{i,v}$ liegt, so wird das Tupel (i, v) in der Menge $\mathcal{C}$ gespeichert und die Flusswerte für Gut i von Kanten in δ^+_v werden in x als die korrespondierenden fixierten Werte in x^k *approximiert*. Im weiteren Verlauf wird dann die Flussaufteilung $x + x^k$, mit Knotenlabels a^k betrachtet, wobei $x \in \mathbb{R}^{I\times E}_{\geq 0}$ die approximierten Flusswerte enthält (und sonst überall 0 ist), und x^k alle fixierten und alle nicht-fixierten Flusswerte, nicht aber die bereits approximierten Flusswerte, enthält.

Sind dagegen die fixierten Flusswerte nicht in obigem Sinne realistisch und führen somit für die aktuellen Steigungen a^k nicht zu einer gleichmäßigen Kostenänderung, so sind sie wahrscheinlich nicht korrekt, d.h. sie wurden zu früh fixiert. Dann werden die Schranken aller von v ausgehenden Kanten für Gut i relaxiert, diese Kanten werden aus F^i_θ entfernt, d.h die Flusswerte werden nicht mehr als „fixiert“ betrachtet und es wird $b^{nf}_{i,v}(\theta) \leftarrow b^-_{i,v}(\theta)$ gesetzt.

[1] Die genaue Abweichung von 0, die für $b^{nf}_{i,v}(\theta)$ toleriert wird, basiert auf der Wahl der Toleranz in Algorithmus 5: `calc_a()` und wird nach dessen Besprechung, in Gleichung (3.13) nachgeliefert.
[2] Dieser wird ebenfalls bei der Besprechung von Algorithmus 5: `calc_a()` genauer erklärt.

Algorithmus 2: `fp_approx(`θ`)`

Eingabe: Zeitpunkt θ.
Ausgabe: Flussaufteilung x zum Zeitpunkt θ, sowie die dazugehörigen Labeländerungen a^k aus der letzten ausgeführten Iteration.
Methode: Initialisiere x, x^0, sowie *coms* und setze $\mathcal{C} \leftarrow \emptyset$, $k \leftarrow 0$;
while *True* **do**
    `fix_fp_comp(`$\mathcal{C}$`)` ; // Algorithmus 3
    if $k > 0$ **then**
        if $coms = \emptyset$ **then**
            return: x, a^k ;
        end if
        $k \leftarrow k + 1$;
        $x^k \leftarrow$ `calc_flow_by_bounds(`x^{k-1}`)` ; // Algorithmus 4
    end if
    $a^k \leftarrow$ `calc_a(`$x + x^k$`)` ; // Algorithmus 5
    `relax_bounds()` ; // Algorithmus 6
    $x \leftarrow$ `fix_nodes()` ; // Algorithmus 7
    if $coms = \emptyset$ **then**
        return: x, a^k ;
    end if
    while *True* **do**
        Bestimme mit $\Gamma(x + x^k)$ Kanten, für die aus Sicht eines Guts zu viel Fluss geschickt wird ;
        for $v \in coms$ **do**
            for $i \in coms_v$ **do**
                Die oberen und unteren Schranken $ub_{i,e}^{k+1}$, $lb_{i,e}^{k+1}$ werden für alle $e \in \delta_{i,v}^+(\theta)$ in Abhängigkeit von $x_{i,e}^k$ gesetzt. Falls dann $ub_{i,e}^{k+1} - lb_{i,e}^{k+1} < \epsilon \cdot 1/|\sigma_\theta(v)|$, so wird im Anschluss $x_{i,e}^{k+1}$ fixiert und $b_{i,v}^{nf}(\theta)$ wird um diesen Eintrag verringert. Sind für das Paar (i, v) alle möglichen von v ausgehenden Kanten fixiert und die Flusswerte realistisch, so wird (i, v) zu $\mathcal{C}$ hinzugefügt ;
            end for
        end for
        $k \leftarrow k + 1$;
        $x^k \leftarrow$ `calc_flow_by_bounds(`x^{k-1}`)` ;
        $a^k \leftarrow$ `calc_a(`$x + x^k$`)` ;
        if $\mathcal{C} \neq \emptyset$ **then**
            break ;
        end if
        `relax_bounds()` ;
        $x \leftarrow$ `fix_nodes()` ;
        if $coms = \emptyset$ **then**
            return: x, a^k ;
        end if
    end while
end while

Die Initialisierung von x und x^0 in Zeile **3** erfolgt lokal mittels Unterscheidung zweier Fälle. Während dieses Prozesses wird auch die Variable *coms* initialisiert. Letzteres ist einfach durchzuführen und wird hier nicht weiter besprochen.

Wir betrachten also alle Paare (i, v) mit $b^-_{i,v}(0) > 0$. Falls $|\delta^+_{i,v}(0)| = 1$, so setze $x_{i,e} \leftarrow b^-_{i,v}(0)$, wobei e die Kante in $\delta^+_{i,v}(0)$ bezeichne. Dies ist die einzig mögliche Aufteilung im Sinne eines IDE-Flusses, weshalb diese Werte hier direkt in x gespeichert werden und offensichtlich korrekt sind.

Im anderen Fall, in dem $|\delta^+_{i,v}(0)| > 1$ gilt, setze für alle Kanten $e \in \delta^+_{i,v}(0)$:

$$x^0_{i,e} \leftarrow \frac{\nu_e}{\sum_{e' \in \delta^+_{i,v}(0)} \nu_{e'}} \cdot b^-_{i,v}(0). \tag{3.3}$$

An dieser Stelle lässt sich noch nichts über die korrekten IDE-Flusswerte aussagen, weshalb mit (3.3) eine erste Schätzung abgegeben wird. Zwar kommt hier jede zulässige Flussaufteilung gleichermaßen infrage, es wird jedoch die Aufteilung anhand des Verhältnisses der Kantenkapazitäten der aktiven Kanten gewählt, da Kanten mit höherer Kapazität auch mehr Flussvolumen für die gleiche Kostenänderung benötigen. Im Fall, dass unter einer korrekten IDE-Flussaufteilung die Knotenlabels $a^0_{i,w}$ aller Endknoten w zu Kanten in $\delta^+_{i,v}(0)$ gleich sind, und zugleich keine weiteren Güter in v vorhanden sind, ist dies sogar schon eine korrekte Aufteilung.

Alle noch nicht initialisierten Komponenten von x und x^0 werden dann auf 0 gesetzt. In $\mathcal{C}$ werden im weiteren Verlauf alle Paare $(i, v) \in I \times V$ gespeichert, deren zugehörige Flussaufteilung entsprechend der vorgegebenen Toleranz ϵ ausreichend eingegrenzt wurden und daher in x approximiert werden können. Diese Menge wird initialisiert als $\mathcal{C} \leftarrow \emptyset$. In jeder späteren Iteration, mit Ausnahme der ersten, wird $\mathcal{C}$ zu Beginn der Hauptschleife nichtleer sein und es wird Algorithmus 3 ausgeführt.

Algorithmus 3: `fix_fp_comp(`$\mathcal{C}$`)`

```
1  Eingabe: Liste C mit allen zu fixierenden (i,v)-Paaren.
2  Ausgabe: Aktualisierte Flusswerte x und Menge der nicht-fixierten
     Gut-Knoten - Paare coms.
3  Methode: Passe x und coms folgendermaßen an:
4  for (i,v) in C do
5  |  for e in δ⁺_{i,v}(θ) do
6  |  |  x_{i,e} ← x^k_{i,e} ;
7  |  end for
8  |  coms_v ← coms_v \ {i} ;
9  |  if coms_v = ∅ then
10 |  |  coms ← coms \ {v} ;
11 |  end if
12 end for
```

Es werden für jedes (i, v)-Paar in $\mathcal{C}$ alle Flusswerte von Gut i für Kanten in $\delta^+_{i,v}(\theta)$ gesetzt. Dies geschieht in Zeile **6** mithilfe der aus den Schranken lb^k, ub^k bestimmten

Flusswerte. Da diese nur bis zum Wert $\epsilon \cdot 1/|\sigma_\theta(v)|$, für eine vorgegebene Toleranz ϵ verfeinert wurden, entstehen dabei Fehler in der Größenordnung von ϵ. Auch wenn die Größe der Fehler mit ϵ kontrolliert werden kann, führen diese in einem Fall zu verheerenden Konsequenzen: Darf in einem IDE-Fluss für ein Gut über eine aktive Kante *kein* Fluss geschickt werden, so reicht eine Approximation dieses Flusswerts nicht aus, denn falls anstelle von 0 ein Einfluss von $x_{i,e} \approx \epsilon$ in diese Kante entsteht, so ergibt sich die Situation, dass dieses Gut in einem Knoten ankommt, den es unter einem exakten IDE-Fluss möglicherweise niemals erreicht hätte. Dies führt zu erhöhtem Rechenaufwand und weiteren Abweichungen von dem zu approximierenden Fluss.

Diesem Problem kann jedoch leicht Abhilfe verschafft werden, indem solche Flusswerte mit einer zusätzlichen Überprüfung entdeckt und auf den gewünschten Wert 0 gesetzt werden. Infolgedessen ergeben sich Approximationsfehler in der Flusserhaltung, welche zwar von den ursprünglichen approximierten Flusswerten erfüllt ist, aber durch den eben beschriebenen Korrekturschritt verletzt wird. Die Flusserhaltung lässt sich jedoch sofort wiederherstellen, indem der übriggebliebene Fluss, mit Volumen von etwa ϵ, auf die übrigen Kanten mit positivem Flusswert umverteilt wird.

Notation 3.1. Zu einer Kante $e \in E$ bezeichne $e^1 \in V$, bzw. $e^2 \in V$, den Start-, bzw. Endknoten von e. Es gilt also $e = (e^1, e^2)$.

In dem ähnlichen Fall, dass $x_{i,e} > b^-_{i,e^1}(\theta) - \epsilon \cdot 1/|\sigma_\theta(v)|$ gilt, wird der Wert $x_{i,e} \leftarrow b^-_{i,e^1}(\theta)$ gesetzt. Dann ist die Flusserhaltung genau erfüllt.

Nach dem Setzen der Flusswerte in Zeile **6** bleiben diese für den weiteren Verlauf des Algorithmus fest und werden auch nicht mehr in *coms* beachtet. Damit ist der Schritt `fix_fp_comp(`$\mathcal{C}$`)` abgeschlossen und die Hauptschleife in `fp_approx(`θ`)` wird fortgesetzt mit den Zeilen **7** - **9**, in denen auf Vollständigkeit der Flusswerte geprüft und gegebenenfalls terminiert wird. Andernfalls wird x^k mittels Algorithmus 4 aktualisiert. Dieser Schritt wird übersprungen in der ersten Iteration, in der die initialen Werte x^0 verwendet werden.

Algorithmus 4: `calc_flow_by_bounds(`x^{k-1}`)`

```
1  Eingabe: vorige Flussaufteilung x^{k-1}
2  Ausgabe: aktuelle Flussaufteilung x^k, welche lb^k, ub^k respektiert und
     Flusserhaltung gewährleistet.
3  Methode:
4  for v in coms do
5      for i in coms_v do
6          for e ∈ δ^+_{i,v}(θ) do
7              x^k_{i,e} ← (lb^k_{i,e} + ub^k_{i,e})/2, falls existent und x^k_{i,e} nicht fixiert, sonst
                verwende x^{k-1}_{i,e} ;
8          end for
9          Skaliere Flusswerte, falls Flusserhaltung verletzt ;
10     end for
11 end for
12 return: x^k ;
```

Notation 3.2. Für eine potentiell unzulässige Flussaufteilung $x \in \mathbb{R}_{\geq 0}^{I \times E}$, welche nicht notwendigerweise die Flusserhaltung (2.5) erfüllt, ein Gut $i \in I$ und einen Knoten $v \in V$ bezeichnen wir mit $x_{i,v}(\theta) \coloneqq \sum_{e \in \delta_{i,v}^+(\theta)} x_{i,e}$ die Summe aller aktuellen von v ausgehenden Abflüsse von Gut i über aktive Kanten.

Die Flusswerte $x_{i,e}^k$ werden in Zeile **7** auf den Mittelpunkt zwischen ihren jeweiligen Schranken $lb_{i,e}^k$ und $ub_{i,e}^k$ gesetzt. Es besteht die Möglichkeit, dass einzelne (i,e)-Paare an dieser Stelle keine Schranken besitzen. Beispielsweise werden, wenn die Flusswerte für ein (i,v)-Paar für die aktuellen a^k-Werte korrekt sind, jedoch noch nicht fixiert werden können, da sich die a^k-Werte noch ändern könnten, keine Schranken für die korrespondierenden (i,e)-Paare gesetzt. In diesem Fall sollen die selben Werte $x_{i,e}^{k-1}$ erneut verwendet werden.

Wie in Zeile **9** beschrieben, werden die Werte anschließend skaliert, sodass für jedes Paar (i,v) gilt, dass $x_{i,v}^k(\theta) = b_{i,v}^-(\theta)$ und weiterhin alle Schranken eingehalten werden. Dies geschieht natürlich nur an den Stellen, an denen auch Schranken vorhanden sind; wurde dagegen $x_{i,e}^k \leftarrow x_{i,e}^{k-1}$ gewählt, ist dies nicht notwendig. Das Vorgehen ist folgendermaßen:

Es bezeichne $x_{i,v}^{k,0}(\theta)$ für alle $(i,v) \in I \times V$ den anfänglichen Wert von $x_{i,v}^k(\theta)$. Letztere sollen nun korrigiert werden, um Flusserhaltung zu erhalten. Dazu wird für alle vorkommenden (i,v)-Paare zwischen zwei Fällen unterschieden:

Fall 1: $i \in I, v \in V : \; x_{i,v}^{k,0}(\theta) < b_{i,v}^-(\theta)$

Es sei $\Delta x_{i,v}^k \coloneqq b_{i,v}^-(\theta) - x_{i,v}^{k,0}(\theta)$ die in v übriggebliebene Flussmenge von Gut i. Weiter sei $\Delta bd_{i,v}^k \coloneqq \sum_{e \in \delta_{i,v}^+ \setminus F^i} ub_{i,e}^k - x_{i,e}^{k,0}$ der Spielraum, der in Knoten v für Fluss von Gut i nach oben besteht.

Behauptung 1. Es gilt immer: $\Delta bd_{i,v}^k \geq \Delta x_{i,v}^k$.

Beweis. Die Behauptung ist erfüllt, g.d.w. $ub_{i,v}^k \coloneqq \sum_{e \in \delta_{i,v}^+} ub_{i,e}^k \geq b_{i,v}^-(\theta)$ gilt. Während des gesamten Algorithmus wird für jede Kante $e \in \delta_{i,v}^+$ der Wert $ub_{i,e}^{l+1} \in \{x_{i,e}^l,\; x_2 + x_{i,e}^l,\; b_{i,v}^-,\; ub_{i,e}^l\}$ gewählt, wobei für den Index $l < k$ gilt und für den Wert $x_2 > 0$, welcher in der Beschreibung von Algorithmus 6 genauer erklärt wird. Wir erhalten insbesondere $ub_{i,e}^k \geq x_{i,e}^{k-1}$ und mit Flusserhaltung für x^{k-1} auch $ub_{i,v}^k \geq b_{i,v}^-(\theta)$. □

Damit können wir setzen:

$$x_{i,e}^k \leftarrow x_{i,e}^{k,0} + (ub_{i,e}^k - x_{i,e}^{k,0}) \cdot \frac{\Delta x_{i,v}^k}{\Delta bd_{i,v}^k}, \text{ für alle } e \in \delta_{i,v}^+. \tag{3.4}$$

Womit sowohl die Flusserhaltung als auch die Einhaltung der Schranken lb^k, ub^k für das Paar (i,v) auch in Iteration k gewährleistet sind.

Fall 2: $i \in I, v \in V : \; x^k_{i,v}(\theta) > b^-_{i,v}(\theta)$

Dieser Fall funktioniert ähnlich, mit $\Delta x^k_{i,v} := x^k_{i,v}(\theta) - b^-_{i,v}(\theta)$ und $\Delta bd^k_{i,v} := \sum_{e \in \delta^+_{i,v}} x^k_{i,e} - lb^k_{i,e}$, unter Verwendung der Eigenschaft $lb^k_{i,v} \leq b^-_{i,v}(\theta)$. Diese folgt analog zu Obigem mit der Beobachtung, dass in Iteration $l < k$ die Schranke $lb^{l+1}_{i,e} \in \{0,\ x^l_{i,e},\ x^l_{i,e} + (a^l_{i,v} - a^l_{i,e^2}) \cdot \nu_e - g_e(x^{(l)}_e + x^l_e),\ lb^l_{i,e}\}$ gewählt wird, wobei $x^{(l)}_e$ den Wert von x_e während der l-ten Iteration bezeichne, also alle Flusswerte die in den ersten l Iterationen approximiert wurden.

Der Grund für die Wahl von $lb^{k+1}_{i,e}$ als der letzte der obigen Werte wurde bisher noch nicht erläutert, an dieser Stelle ist aber auch nur wichtig, dass dieser den Wert $b^-_{i,v}(\theta)$ nicht überschreitet. Dies ist auch der Fall, da nach Definition 2.7 $(a^l_{i,v} - a^l_{i,e^2}) \cdot \nu_e - g_e(x^{(l)}_e + x^l_e) \leq 0$ gilt. Eine genauere Betrachtung dieser Wahl ist zu finden am Ende dieses Abschnitts in der Beschreibung von Zeile **23** aus Algorithmus 2, oder bei der Beschreibung von `relax_bounds()` nach Algorithmus 6.

Nachdem mit `calc_flow_by_bounds()` die Flusswerte für die aktuelle Iteration von `fp_approx(`θ`)` berechnet wurden, werden die dazu passenden a^k-Werte benötigt, welche in Zeile **13** mit `calc_a(`$x + x^k$`)` bestimmt werden. Dazu sei $\Phi_{i,\theta} : V \to \mathbb{N}$ für jedes Gut $i \in I$ eine topologische Sortierung auf dem Teilgraphen, der von Rückwärtskanten von momentan für Gut i aktiven Kanten induziert ist.

Algorithmus 5: `calc_a(`$x + x^k$`)`

```
1 Eingabe: aktuelle Flussaufteilung x + x^k
2 Ausgabe: Knotenlabeländerungen a^k mit zugehörigen verwendeten Kanten
    A^k, sowie der zweitkleinste erreichte Wert a^{k,min_2} für jedes relevante Paar
    (i, v).
3 Methode:
4 for i in I do
5     for v in Reihenfolge von Φ_{i,θ} do
6         Bestimme a^k_{i,v}-Wert nach (2.21) und merke alle Kanten in A^k_{i,v}, für
            die dieser Wert bis auf eine Fehlertoleranz erreicht wird, sowie den
            nächsthöheren vorkommenden Wert a^{k,min_2}_{i,v}, wobei a^{k,min_2}_{i,v} = a^k_{i,v},
            falls nur ein Wert erreicht wird ;
7     end for
8 end for
9 return: a^k, a^{k,min_2}, A^k ;
```

Die Iteration der Knoten in der Reihenfolge von $\Phi_{i,\theta}$ ist dabei notwendig, damit die berechneten a^k_i-Werte in (2.21) wohldefiniert sind. Da Algorithmus 1 mit approximativen Werten arbeitet, setzen sich die Approximationsfehler auch bei der Berechnung der Steigung a^k fort. Dies hat zur Folge, dass mehrere Kanten mit demselben Startknoten, die im exakten Fall für ein Gut aktiv sind und somit Wege mit exakt gleichen Kosten einleiten, im approximierten Fall leicht unterschiedliche Distanzwerte liefern. Um die Approximation sinnvoll fortsetzen zu können, müssen diese Unterschiede vernachlässigt und die Kanten trotzdem alle als aktiv eingestuft

werden. Zu diesem Zweck erweitern wir bei Betrachtung approximierter IDE-Flüsse das Konzept der aktiven Kanten zu dem der *approximiert aktiven Kanten*. Die Menge der approximiert aktiven Kanten für Gut i zum Zeitpunkt θ wird bezeichnet mit $\widetilde{E}^i_\theta$. Es beschreibt also $\widetilde{E}^i_\theta$ die Menge der Kanten, die, unter Vernachlässigung von Approximationsfehlern, zum Zeitpunkt θ auf kürzesten Wegen zur Senke t_i liegen. Formal halten wir fest:

$$\widetilde{E}^i_\theta := \{e = (v, w) \in E : \ell_{i,v}(\theta) - \ell_{i,w}(\theta) - c_e(\theta) > -\delta\}, \tag{3.5}$$

wobei die exakte Wahl von $\delta > 0$ weitere Einsichten in die Berechnung der approximierten IDE-Flüsse benötigt und in Abschnitt 3.4 nachgeliefert wird. Weiterhin bezeichne

$$\widetilde{\delta}^+_{i,v}(\theta) := \delta^+_v \cap \widetilde{E}^i_\theta \tag{3.6}$$

die Menge der für Gut i momentan von v ausgehenden approximiert aktiven Kanten.

Wir formulieren zunächst folgendes Analogon der Eigenschaft (2.22) aus Definition 2.7 für die approximiert aktiven Kanten:

Definition 3.3. Für einen approximierten IDE-Fluss bis zum Zeitpunkt θ sei $v \in V$ ein Knoten und $(x, a) \in \mathbb{R}^{|coms_v| \times |\delta^+_v|}_{\geq 0} \times \mathbb{R}^{|coms_v|}$ eine Flussaufteilung in v. Dann erfüllt das Tupel (x, a) die *IDE-Eigenschaft im Knoten* v, wenn gilt:

$$a_{i,v} = \frac{g_e(x_e)}{\nu_e} + a_{i,w} \quad \text{f.a. } i \in coms_v,\ e = (v, w) \in \widetilde{\delta}^+_{i,v},\ \text{mit } x_{i,e} > 0. \tag{3.7}$$

Es sei nun für ein festes Paar $(i, v) \in I \times V$ eine Kante

$$e_{min} = (v, w_{min}) \in \underset{e=(v,w)\in\delta^+_{i,v}}{\arg\min}\ g_e(x_e + x^k_e)/\nu_e + a^k_{i,w} \tag{3.8}$$

gegeben. Dies ist demnach eine Kante, über die der Wert $a^k_{i,v}$ exakt erreicht wird. Wir definieren damit die Matrix A^k, welche neben a^k einen weiteren Ausgabeparameter von `calc_a`$(x + x^k)$ darstellt, eintragsweise als:

$$A^k_{i,v} := \left\{ e \in \widetilde{\delta}^+_{i,v}(\theta) : \frac{g_e(x_e + x^k_e)}{\nu_e} + a^k_{i,w} - a^k_{i,v} < \frac{\epsilon}{\nu_e} + \frac{\epsilon}{\nu_{e_{min}}} \right\}. \tag{3.9}$$

Diese Wahl von A^k lässt sich folgendermaßen begründen: Unter der Voraussetzung, dass die aktuellen Knotenlabels $a^k_{i,w}$ aller Endknoten $w \in V$ von Kanten $e = (v, w) \in \delta^+_v$ bereits bestimmt sind und eine Flussaufteilung (x^*, a^*) mit $a^*_{i,w} = a^k_{i,w}$, für $w \in \delta^+_v$, zulassen, welche in v die IDE-Bedingung (3.7) erfüllt und bereits von $x + x^k$ so genau approximiert wird, dass gilt:

$$\forall j \in \sigma_\theta(v) : \forall e \in \widetilde{\delta}^+_{j,v}(\theta) : |x_{j,e} + x^k_{j,e} - x^*_{j,e}| < \epsilon \cdot \frac{1}{|\sigma_\theta(v)|}. \tag{3.10}$$

Dann folgt

$$\frac{g_{e_{min}}(x^*_{e_{min}})}{\nu_{e_{min}}} - \frac{g_{e_{min}}(x_{e_{min}} + x^k_{e_{min}})}{\nu_{e_{min}}} < \frac{\epsilon}{\nu_{e_{min}}}. \tag{3.11}$$

Weiterhin gilt für $e \in \delta^+_v$:

$$\frac{g_e(x_e + x^k_e)}{\nu_e} - \frac{g_e(x^*_e)}{\nu_e} < \frac{\epsilon}{\nu_e}. \tag{3.12}$$

Somit ergibt sich insgesamt für $i \in coms_v$ und $e^* = (v, w^*) \in \arg\min_{e=(v,w)\in\delta^+_{i,v}} g_e(x^*_e)/\nu_e + a^*_{i,w}$:

$$\begin{aligned}
\frac{g_{e^*}(x_{e^*} + x^k_{e^*})}{\nu_{e^*}} + a^k_{i,w^*} - a^k_{i,v} &= \frac{g_{e^*}(x_{e^*} + x^k_{e^*})}{\nu_{e^*}} + a^k_{i,w^*} - \frac{g_{e_{min}}(x_{e_{min}} + x^k_{e_{min}})}{\nu_{e_{min}}} - a^k_{i,w_{min}} \\
&< \frac{g_{e^*}(x^*_{e^*})}{\nu_{e^*}} + \frac{\epsilon}{\nu_{e^*}} + a^*_{i,w^*} - \frac{g_{e_{min}}(x^*_{e_{min}})}{\nu_{e_{min}}} + \frac{\epsilon}{\nu_{e_{min}}} - a^*_{i,w_{min}} \\
&= a^*_{i,v} + \frac{\epsilon}{\nu_{e^*}} - a^*_{i,v} + \frac{\epsilon}{\nu_{e_{min}}} \\
&= \frac{\epsilon}{\nu_{e^*}} + \frac{\epsilon}{\nu_{e_{min}}}.
\end{aligned}$$

Es gilt also $e^* \in A^k_{i,v}$. Es lässt sich zusammenfassend sagen: Ist die Approximation $(x + x^k, a^k)$ für einen Knoten v so genau, dass die Werte a^k lokal (in v) eine Flussaufteilung zulassen, die die IDE-Eigenschaft (3.7) erfüllt und die Werte in $x + x^k$ diese Flussaufteilung hinreichend genau approximieren, so beschreibt $A^k_{i,v}$ aus (3.9) gerade die Menge der Kanten, über die der Wert $a^k_{i,v}$ bis auf einen Approximationsfehler erreicht wird.

Die Forderung (3.10) stellt die gewünschte Situation dar, die es gilt mittels der in Algorithmus 1 berechneten Approximationen zu erreichen. Da die Schranken in Algorithmus 2 bis zur Genauigkeit $\epsilon \cdot 1/|\sigma_\theta(v)|$ verfeinert werden, ist eine Erfüllung der Ungleichung denkbar.

Bei der Berechnung der Steigung für ein (i, v) - Paar verwendet Algorithmus 5 also einen Ausdruck der Form $\epsilon/\nu_e + \epsilon/\nu_{e_{min}}$, für zwei Kanten wie oben. Damit kommen wir auf den zu Beginn des Abschnitts betrachteten Fall zurück, in dem es galt zu entscheiden, ab welcher Entfernung zur 0 der Wert $b^{nf}_{i,v}(\theta)$ auch tatsächlich als 0 interpretiert wird. Wir wählen dazu $\nu^{min}_{i,v} = \min_{e\in\delta^+_{i,v}(\theta)} \nu_e$ und schätzen die in `calc_a()` verwendete Toleranz ab mittels $2 \cdot \epsilon/\nu^{min}_{i,v}$. Bei der Ausführung des Algorithmus geschieht intern also Folgendes:

$$\text{Falls } b^{nf}_{i,v}(\theta) < \frac{2 \cdot \epsilon}{\nu^{min}_{i,v}}, \text{ dann setze } b^{nf}_{i,v}(\theta) \leftarrow 0. \tag{3.13}$$

Allgemein stellt dies ein wichtiges Konzept bei der Abschätzung von Approximationsfehlern einzelner Flusswerte dar: Die Wahl der Toleranz in (3.9) geschieht in Hinblick auf den Fehler der Gesamtflusswerte, da nur diese bei der Berechnung von a^k relevant sind. Bei der Betrachtung der Flusswerte einzelner Güter ist es daher

möglich, dass ein Gut den gesamten Anteil am Fehler des Gesamtflusses ausmacht, wenn dafür alle Flusswerte von anderen Gütern auf der selben Kante exakt sind. Dies ist der Grund, weshalb die Schranken lb^k, ub^k für jeden Knoten $v \in V$ bis zur Genauigkeit von $\epsilon \cdot 1/|\sigma_\theta(v)|$ verfeinert werden: Je mehr Güter in einem Knoten vorhanden sind, desto genauer sollen die Flusswerte sein, damit der maximale gesamte Fehler unter Kontrolle gehalten werden kann. Dies wird bei der Bestimmung der Schrittweite in Abschnitt 3.3 von besonderer Wichtigkeit sein.

Der Parameter A^k wird später aufgrund von zwei verschiedenen Zwecken benötigt: Erstens wird Fluss damit auf Erfüllung der IDE-Eigenschaft (3.7) geprüft und zweitens können damit falsche Schranken in lb^k, ub^k identifiziert werden. Für die Korrektur der fehlerhaften Einträge von lb^k wird zusätzlich der Ausgabeparameter $a^{k,min_2} \in \mathbb{R}^{I \times n}$ benötigt. Diese Matrix dient zur Bestimmung der neuen, relaxierten, lb^k's. Der Relaxierungsprozess geschieht in Algorithmus 2: `fp_approx(`θ`)` in Zeile **14** mithilfe der Funktion `relax_bounds()` und wird nun genauer erklärt.

Algorithmus 6: `relax_bounds()`

1 **Eingabe:** Schranken lb^k, ub^k an die Flusswerte
2 **Ausgabe:** Aktualisierte Werte lb^k, ub^k
3 Methode:
4 **for** v **in** *coms* **do**
5 **for** i **in** $coms_v$ **do**
6 Überprüfe, ob einer von zwei Fällen auftritt, in denen $lb^k_{i,e}$, bzw. $ub^k_{i,e}$ für eine Kante $e \in \delta_v^+$ nicht korrekt sein kann und relaxiere lb^k, ub^k gegebenenfalls ;
7 **end for**
8 **end for**
9 **return** lb^k, ub^k ;

Die Schranken lb^k, ub^k an die einzelnen Flusswerte sind nicht immer „korrekt“, d.h., dass kein Wert innerhalb des Intervalls zwischen einem solchen Schrankenpaar Teil eines IDE-Flusses sein kann. Dieser unerwünschte Fall kann eintreten, da sich die Schranken in Iteration k immer nur an der aktuellen, potentiell ungenauen, Approximation $(x + x^k, a^k)$ an den feinen IDE-Fluss orientieren. Da es insbesondere keinen klaren Anfangspunkt gibt, können vor allem in den ersten Iterationen die vorkommenden Werte sehr ungenau sein, was dazu führt, dass der Algorithmus Flusswerte ausschließt, die sich später bei genaueren a^k-Werten als richtig erweisen. Aus diesem Grund wird darauf geachtet, dass Flusswerte nicht vorschnell als korrekt eingestuft werden, sondern gewartet, bis auch alle anderen Flusswerte für einen Knoten dazu passen, sodass eine Aufteilung im IDE-Sinne realistisch erscheint. Unrealistische Flusswerte dagegen sollen durch Relaxierung der Schranken wieder verworfen werden. Dies geschieht in `relax_bounds()`, in der zwei Fälle aufgedeckt werden, in denen die vorkommenden Schranken für das jeweilige Gut keinen Raum für Verbesserungen zulassen und deshalb angepasst werden sollen.

Im Sinne der Übersichtlichkeit werden im Folgenden die Toleranzbereiche für

die durch Approximationsfehler entstandenen Ungenauigkeiten vernachlässigt. Die Rechnungen werden dadurch anschaulicher und können im Bedarfsfall leicht angepasst werden, um alle vorkommenden Toleranzbereiche zu enthalten. Aus dem gleichen Grund werden Kanten hier als aktiv bezeichnet, auch wenn sie bei exakter Rechnung eventuell nur approximiert aktiv wären.

Es sei nun $v \in V$ ein Knoten und $i \in I$ ein Gut, von welchem zum aktuellen Zeitpunkt Fluss in v vorhanden ist.

Fall 1: Alle für i aktiven, von v ausgehenden Kanten, die einen nicht-minimalen a^k-Wert liefern, sind bereits fixiert.
In diesem Fall gibt es Kanten, über die aus Sicht von i zu wenig Fluss geschickt wird, jedoch keine Kanten, von denen Fluss auf diese Kanten umverteilt werden könnte. Deshalb werden die unteren Schranken zu nicht-minimalen Flusswerten angepasst. Sei dazu $e = (v, w) \in \delta^+_{i,v}$ eine solche Kante, d.h. es gilt: $a^k_{i,v} < g_e(x_e + x^k_e)/\nu_e + a^k_{i,w}$. Sei weiter $\mathcal{I}(e) := \{j \in I : a^k_{j,v} < g_e(x_e + x^k_e)/\nu_e + a^k_{j,w}\} \ni i$.

Fall 1.1: $q_e(\theta) > 0$
Wähle als neue untere Schranke

$$lb^k_{i,e} \leftarrow \max\{0,\ x^k_{i,e} - (x_e + x^k_e - (1 + a^k_{i,v} - a^k_{i,w}) \cdot \nu_e)\}. \tag{3.14}$$

Wird das Maximum in (3.14) im zweiten Teil angenommen, so ist diese Schranke tatsächlich eine Relaxierung, da wegen $i \in \mathcal{I}(e)$ gilt:

$$x_e + x^k_e - (1 + a^k_{i,v} - a^k_{i,w}) \cdot \nu_e = g_e(x_e + x^k_e) + (a^k_{i,w} - a^k_{i,v}) \cdot \nu_e > 0.$$

Des Weiteren ist diese Relaxierung zielführend, da sie eine Setzung $\hat{x}^k$ mit zugehörigem $\hat{a}^k$ zulässt, bei der der Wert $\hat{a}^k_{i,v} = a^k_{i,v}$ auch über Kante e erreicht wird. Wähle dazu $\hat{x}^k_{i,e} = x^k_{i,e} - (x_e + x^k_e - (1 + a^k_{i,v} - a^k_{i,w}) \cdot \nu_e)$ und $\hat{x}^k_{j,e} = x^k_{j,e}$ für alle $j \in I \setminus \{i\}$. Dann gilt:

$$x_e + \hat{x}^k_e = x_e + x^k_e - (x_e + x^k_e - (1 + a^k_{i,v} - a^k_{i,w}) \cdot \nu_e) = (a^k_{i,v} - a^k_{i,w}) \cdot \nu_e + \nu_e.$$

Außerdem stimme $\hat{x}^k$ für alle Kanten mit einem anderen Startknoten als v mit x^k überein und für die restlichen Kanten $e \neq e' \in \delta^+(v)$ gelte ebenfalls $\hat{x}^k_{j,e'} = x^k_{j,e'}$ für $i \neq j \in I$ und das zusätzliche Flussvolumen von Gut i der Größe $(x_e + x^k_e - (1 + a^k_{i,v} - a^k_{i,w}) \cdot \nu_e)$ sei so auf diese Kanten e' aufgeteilt, dass Flusserhaltung auch für dieses Gut im Knoten v gilt.

Insbesondere gilt $\hat{a}^k_{i,w} = a^k_{i,w}$ und somit

$$\begin{aligned} & g_e(x_e + \hat{x}^k_e) = x_e + \hat{x}^k_e - \nu_e = (a^k_{i,v} - a^k_{i,w}) \cdot \nu_e \\ \Leftrightarrow\ & \frac{g_e(x_e + \hat{x}^k_e)}{\nu_e} + \hat{a}^k_{i,w} = a^k_{i,v}. \end{aligned}$$

Es besteht also genügend Spielraum, dass auch über e der Wert $a^k_{i,v}$ erreicht werden kann.

Wird dagegen das Maximum in (3.14) in 0 angenommen, so kann Gut i den

gewünschten Wert nicht aus eigener Kraft erreichen, die Schranke $lb^k_{i,e} = 0$ liefert jedoch die für Gut i größtmöglichen Freiheiten den Fluss auf e zu reduzieren.

Fall 1.2: $q_e(\theta) = 0$
In diesem Fall gilt $g_e(z) \geq 0$ für alle möglichen Einflussraten $z \in \mathbb{R}_{\geq 0}$, weshalb durch eine lokale Änderung von x^k nicht wie in Fall 1.1 der aktuell kleinste a^k-Wert über e erreicht werden kann. Wir wählen dann die kleinstmögliche untere Schranke, nämlich 0. Setze also: $lb^k_{i,e} \leftarrow 0$.

Fall 2: Alle für i aktiven, von v ausgehenden Kanten, die einen minimalen a-Wert liefern, sind bereits fixiert.
Da die Flussaufteilung noch nicht im Sinne eines IDE-Flusses ist, gibt es auch Kanten die von v ausgehen, für die der entsprechende minimale a-Wert nicht erreicht wird. Auch in diesem Fall können die Differenzen nicht ausgeglichen werden, weshalb die oberen Schranken der Kanten mit minimalem a^k-Wert so angepasst werden sollen, dass der nächsthöhere vorkommende a^k-Wert erreicht werden kann. Dieser Wert wurde mit Algorithmus 5 bereits bestimmt und gespeichert als $a^{k,min_2}_{i,v}$. Sei nun $e = (v, w) \in \delta^+_{i,v}(\theta)$ eine Kante, über die der Wert $a^k_{i,v}$ erreicht wird. Wir unterscheiden:

Fall 2.1: $q_e(\theta) > 0$ oder $x^k_e \geq \nu_e$
Wähle als neue obere Schranke

$$ub^k_{i,e} \leftarrow \min\{x^k_{i,e} + (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e - g_e(x_e + x^k_e),\ b^-_{i,v}\}. \tag{3.15}$$

Die Argumentation hinter dieser Wahl ist analog zu Fall 1.1: Wird das Minimum in (3.15) in $b^-_{i,v}(\theta)$ angenommen, so erhält Gut i den größtmöglichen Freiraum für Kante e. Andernfalls wird $\hat{x}^k$ gewählt wie in Fall 1.1, mit den Unterschieden, dass erstens $\hat{x}^k_{i,e} = x^k_{i,e} + (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e - g_e(x_e + x^k_e)$, wobei $(a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e - g_e(x_e + x^k_e) > 0$ nach Konstruktion. Zweitens sei dieser zusätzliche Fluss von Gut i auf Kante e den anderen Kanten $e \neq e' \in \delta^+(v)$ im Verhältnis ihrer Flusswerte in x^k entnommen. Es ergibt sich

$$\hat{x}^k_e = x^k_e + (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e - g_e(x_e + x^k_e) = (1 + a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e,$$

und damit:

$$\begin{aligned} & g_e(x^k + \hat{x}^k_e) = \hat{x}^k_e - \nu_e = (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e \\ \Leftrightarrow\ & \frac{g_e(x^k + \hat{x}^k_e)}{\nu_e} + \hat{a}^k_{i,w} = a^{k,min_2}_{i,v}. \end{aligned}$$

Die Relaxierung erzielt also das gewünschte Ergebnis.

Fall 2.2: $q_e(\theta) = 0$ und $x^k_e < \nu_e$
Es seien

$$ub^k_{i,e} \leftarrow \min\{x^k_{i,e} + (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e + \nu_e - x_e - x^k_e,\ b^-_{i,v}\}.$$

Dies ist essentiell die gleiche Wahl wie in Fall 2.1, nur die Begründung ändert sich. Wird das Minimum im ersten Teil angenommen, so sei $\hat{x}^k$ gewählt wie eben, jedoch mit $\hat{x}^k_{i,e} = x^k_{i,e} + (a^{min_2}_{i,v} - a_{i,w}) \cdot \nu_e + \nu_e - x_e - x^k_e$. Dann gilt:

$$\hat{x}^k_e = x^k_e + (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e + \nu_e - x_e - x^k_e = (1 + a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e - x_e.$$

Weiterhin folgt mit

$$\begin{aligned} x_e + \hat{x}^k_e &= x_e + x^k_e + (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e + \nu_e - x_e - x^k_e \\ &= (a^{k,min_2}_{i,v} - a^k_{i,w}) \cdot \nu_e + \nu_e > \nu_e, \end{aligned}$$

dass $g_e(x_e + \hat{x}^k_e) = x_e + \hat{x}^k_e - \nu_e$. Insgesamt ergibt sich wieder:

$$\frac{g_e(x_e + \hat{x}^k_e)}{\nu_e} + \hat{a}^k_{i,w} = a^{k,min_2}_{i,v}.$$

Nach dem Relaxieren der Schranken werden die entsprechenden Flusswerte so gesetzt, dass sie die relaxierten Schranken respektieren. Dies erfolgt nach dem gleichen Vorgehen wie in Algorithmus 4: `calc_flow_by_bounds()`.

Damit ist die Relaxierung der Schranken mittels `relax_bounds()` geklärt und Zeile **15** von Algorithmus 2 wird erreicht. Die Funktion `fix_nodes()` behandelt nach der Setzung in der Initialisierung von x und der Funktion Algorithmus 3: `fix_fp_comp()` den dritten und letzten Fall, in dem Flusswerte in x approximiert werden, entscheidet also, welche Flusswerte eine ausreichend genaue Approximation darstellen.

Algorithmus 7: `fix_nodes`$(x, x^k, A^k, coms)$

```
1  Eingabe: Flusswerte x, x^k, sowie A^k, coms
2  Ausgabe: Aktualisierte Flusswerte x und coms
3  Methode:
4  for v in coms do
5      if Alle von v ausgehenden, nicht-fixierten, für ein Gut i aktiven Kanten
         mit positivem Flusswert liegen in A^k_{i,v} then
6          if Alle von v ausgehenden Kanten mit für ein Gut i fixiertem,
             positivem Flusswert liegen in A^k_{i,v} then
7              if Alle Knoten, die für ein Gut in coms_v in der topologischen
                 Sortierung dieses Guts nach v kommen, haben keine
                 nicht-fixierten Flusswerte then
8                  Approximiere alle Flusswerte x_{i,e} für i ∈ coms_v, e ∈ δ^+_{i,v}(θ) ;
9                  coms ← coms \ {v} ;
10             end if
11         end if
12     end if
13 end for
14 return: x, coms ;
```

Die Überprüfung geschieht für alle relevanten Knoten, d.h. alle Knoten, in denen sich aktuell Fluss irgendeines Guts befindet. Für einen dieser Knoten v wird dann zuerst geprüft, ob alle von v ausgehenden, für irgendein Gut aktiven, nichtfixierten, Kanten mit positivem Flusswert unter der aktuellen Flussaufteilung die (bis auf Approximationsfehler) gleichen a^k - Werte liefern. Die Knoten, für die dies erfüllt ist, sind die Kandidaten für Flusswerte, die in x approximiert werden können. Für diese wird dann geprüft, ob auch die bereits fixierten Kanten mit positivem Gesamtflusswert den selben a^k - Wert erzeugen. Ist auch dies der Fall, so wird als letztes geprüft, ob alle Knoten die einen Einfluss auf die Labels der Endknoten von Kanten in δ_v^+ haben können, bereits approximiert sind und wenn ja, werden alle von dem betrachteten Knoten ausgehenden Kanten approximiert.

Diese Vielzahl an Tests hat den Vorteil: Werden in `fix_nodes()` Flusswerte approximiert, so sind diese sicher korrekt, unter der Voraussetzung, dass alle vorher approximierten Flusswerte bereits korrekt waren.

Nun ist die zweite **while** - Schleife in Zeile **19** von Algorithmus 2 erreicht, in der die eigentliche Approximation der Flusswerte stattfindet. Dieser Prozess beginnt mit einem Aufruf der Funktion Γ (2.25). Diese liefert Informationen darüber, inwiefern die aktuelle Flussaufteilung $x + x^k$ zu viel Fluss über eine bestimmte Kante schickt (aus Sicht eines oder mehrerer Güter). Des Weiteren ist bereits aus `calc_a`$(x + x^k)$ bekannt, für welche (i, e) - Paare zu wenig Fluss verschickt wurde. Damit können in Zeile **23** die Schranken lb^k und ub^k verfeinert werden, womit eine Verbesserung der Approximation erhofft wird. Dabei ist zu beachten: Während der Durchführung dieser **while** - Schleife rücken einzelne Schranken $lb_{i,e}^k$, $ub_{i,e}^k$ sukzessive näher zusammen, bis die gewünschte Toleranz $\epsilon \cdot 1/|\sigma_\theta(e^1)|$ erreicht ist. Das Erreichen dieser Toleranz reicht jedoch nicht aus, um Korrektheit der approximierten Werte garantieren zu können.

Ein erster Fall, in dem diese nicht gegeben ist, ist wenn $lb_{i,e}^k = ub_{i,e}^k$: Dieser Fall tritt auf, wenn $lb_{i,e}^k = x_{i,e}^k$ gilt und $x + x^k$ aus Sicht von Gut i zu viel Fluss über e schickt (oder umgekehrt: $ub_{i,e}^k = x_{i,e}^k$ und $x + x^k$ schickt aus Sicht von i zu wenig über e). Demnach ist $x_{i,e}^k$ im Allgemeinen nicht der korrekte Wert, weshalb $lb_{i,e}^k$ (bzw. $ub_{i,e}^k$), nach dem selben Vorgehen wie in Algorithmus 6: `relax_bounds()`, relaxiert wird.

Gilt dagegen $0 < ub_{i,e}^k - lb_{i,e}^k < \epsilon \cdot 1/|\sigma_\theta(e^1)|$, so wird $x_{i,e}^k$ fixiert und auch die Werte $lb_{i,e}^k$, $ub_{i,e}^k$ werden vorläufig festgehalten. Geschieht dies für ein (i, v) - Paar für alle Kanten $e \in \delta_{i,v}^+(\theta)$ (bis auf eventuell eine übrige Kante, welche sämtliches restliche Flussvolumen von Gut i erhält), so folgt ein letzter Test, um zu entscheiden, ob die gewählten Flusswerte im Sinne eines IDE-Flusses sind: erzeugen alle von v ausgehenden Kanten mit positivem Flusswert von Gut i den Wert $a_{i,v}^k$, so wird das Paar (i, v) zu $\mathcal{C}$ hinzugefügt und mittels Algorithmus 3: `fix_fp_comp()` werden alle zugehörigen Flusswerte approximiert. Dies geschieht, obwohl auch an dieser Stelle die Korrektheit des Flusses nicht garantiert ist, da sich die Flusswerte anderer Güter oder des selben Gutes mit anderen Startknoten immer noch ändern können. Aufgrund der bei der Konstruktion verwendeten Tests wird jedoch erhofft, dass die Korrektheit dennoch in den meisten Fällen gewährleistet ist. Weiterhin ist dies die einzige Stelle bei der Berechnung des Flusses mittels `fp_approx()`, an der Flusswerte approximiert

werden, deren Korrektheit nicht zugesichert werden kann.

Es besteht auch die Möglichkeit, dass der zuletzt beschriebene Test fehlschlägt. Geschieht dies, so wird $b^{nf}_{i,v}(\theta) \leftarrow b^-_{i,v}(\theta)$ zurückgesetzt, alle von v ausgehenden Kanten werden für Gut i nicht mehr als „fixiert“ betrachtet und die korrespondierenden Schranken $lb^k_{i,e}$, $ub^k_{i,e}$ werden genau wie in `relax_bounds()` relaxiert.

Einsichten in die Konvergenz der Schranken liefern die nun betrachteten Lemmata.

Lemma 3.4. Es seien $v \in coms$, $i \in coms_v$ und für alle nicht-fixierten Kanten $e \in \delta^+_{i,v} \setminus F^i_\theta$ seien $lb^k_{i,e}$, $ub^k_{i,e}$ die aktuellen Schranken an die Flusswerte $x^k_{i,e}$. Weiter sei $\Lambda^k_{i,v} \coloneqq \sum_{e \in \delta^+_{i,v} \setminus F^i_\theta} ub^k_{i,e} - lb^k_{i,e}$ die gesamte momentan verfügbare Intervalllänge für Gut i und Kanten mit Startknoten v. Sind dann $ub^{k+1}_{i,e}$, $lb^{k+1}_{i,e}$ die in Zeile **23** von Algorithmus 2 aktualisierten Schranken und werden diese *nicht* im nächsten Aufruf von `relax_bounds()` relaxiert, so gilt:

$$\Lambda^{k+1}_{i,v} \leq \Lambda^k_{i,v} - \frac{1}{2} \cdot \min_{e \in \delta^+_{i,v} \setminus F^i_\theta} \left(ub^k_{i,e} - lb^k_{i,e}\right) \cdot \left(1 + \frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}\right). \tag{3.16}$$

Beweis. Bekanntermaßen ergeben sich die Schranken ub^{k+1}, lb^{k+1} unmittelbar aus der Flussaufteilung x^k. Diese wiederum wurde zuvor bestimmt mittels `calc_flow_by_bounds()` aus den Schranken ub^k und lb^k. Dies geschah in zwei Schritten: Zuerst durch die Setzung der Werte $x^{k,0}_{i,e}$ als der Mittelpunkt des zugehörigen zugelassenen Intervalls $[ub^k_{i,e}, lb^k_{i,e}]$ und anschließend durch die Skalierung dieser Werte, um die Flusserhaltung zu erfüllen. Im Fall, dass $x^{k,0}_{i,v} < b^-_{i,v}(\theta)$ gilt, geschieht diese Skalierung wie in (3.4) beschrieben:

$$x^k_{i,e} \leftarrow x^{k,0}_{i,e} + \left(ub^k_{i,e} - x^{k,0}_{i,e}\right) \cdot \frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}, \tag{3.17}$$

wobei

$$\Delta x^k_{i,v} = b^-_{i,v} - x^{k,0}_{i,v} \text{ und } \Delta bd^k_{i,v} = \sum_{e \in \delta^+_{i,v} \setminus F^i} ub^k_{i,e} - x^{k,0}_{i,e}. \tag{3.18}$$

Je nachdem, ob für eine Kante $e \in \delta^+_{i,v} \setminus F^i_\theta$ gilt, dass $e \in A^k_{i,v}$, oder $e \notin A^k_{i,v}$, wird entweder $lb^{k+1}_{i,e} \leftarrow x^k_{i,e}$ und $ub^{k+1}_{i,e} \leftarrow ub^k_{i,e}$ gewählt, oder umgekehrt $lb^{k+1}_{i,e} \leftarrow lb^k_{i,e}$ und $ub^{k+1}_{i,e} \leftarrow x^k_{i,e}$. In letzterem Fall beträgt die Verringerung der Intervalllänge der Schranken des Paares (i,e) von Schritt k zu Schritt $k+1$ nach Konstruktion gerade $\left(ub^k_{i,e} - x^{k,0}_{i,e}\right) \cdot \left(1 - \frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}\right) \geq 0$. Im Fall, dass der Wert $ub^{k+1}_{i,e}$ nicht im nächsten Aufruf von `relax_bounds()` relaxiert wird, existiert weiterhin eine Kante $e' \in \left(\delta^+_{i,v} \setminus F^i_\theta\right) \cap A^k_{i,v}$. Für diese Kante werden die Schranken gewählt wie im ersten oben genannten Fall. Die Verringerung des Abstands der zugehörigen unteren und oberen Schranken beträgt dann $\frac{1}{2} \cdot \left(ub^k_{i,e'} - lb^k_{i,e'}\right) \cdot \left(1 + \frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}\right)$. Falls $e' \in \arg\min_{e \in \delta^+_{i,v} \setminus F^i_\theta} \left(ub^k_{i,e} - lb^k_{i,e}\right)$ liegt, ergibt sich immer noch mindestens der Fortschritt aus (3.16).

Die Fälle $x^{k,0}_{i,v} > b^-_{i,v}$ und $x^{k,0}_{i,v} = b^-_{i,v}$ ergeben sich durch analoges Vorgehen, wobei

in letzterem sogar gilt: $\Lambda_{i,v}^{k+1} = \frac{1}{2} \cdot \Lambda_{i,v}^{k}$. □

Lemma 3.5. In der Situation von Lemma 3.4 gilt weiterhin:

$$\Lambda_{i,v}^{k+1} \leq \Lambda_{i,v}^{k} - \min \left\{ \sum_{e \in A_{i,v}^{k}} ub_{i,e}^{k} - lb_{i,e}^{k}, \sum_{e \in \delta_{i,v}^{+} \setminus A_{i,v}^{k}} ub_{i,e}^{k} - lb_{i,e}^{k} \right\} \tag{3.19}$$

$$= \max \left\{ \sum_{e \in A_{i,v}^{k}} ub_{i,e}^{k} - lb_{i,e}^{k}, \sum_{e \in \delta_{i,v}^{+} \setminus A_{i,v}^{k}} ub_{i,e}^{k} - lb_{i,e}^{k} \right\}. \tag{3.20}$$

Beweis. Wir betrachten erneut den Fall $x_{i,v}^{k,0} < b_{i,v}^{-}$, der andere Fall funktioniert analog. Für die Kanten $e^{-} \in A_{i,v}^{k}$ werden $ub_{i,e^{-}}^{k+1} \leftarrow ub_{i,e^{-}}^{k}$ und $lb_{i,e^{-}}^{k+1} \leftarrow x_{i,e^{-}}^{k}$ gesetzt, womit sich für diese ein Fortschritt von insgesamt $\sum_{e^{-} \in A_{i,v}^{k}} \frac{1}{2} \cdot \left(ub_{i,e^{-}}^{k} - lb_{i,e^{-}}^{k} \right) \cdot \left(1 + \frac{\Delta x_{i,v}^{k}}{\Delta bd_{i,v}^{k}} \right)$ ergibt. Für die übrigen Kanten $e^{+} \in \delta_{i,v}^{+} \setminus A_{i,v}^{k}$ ergibt sich mit $ub_{i,e^{+}}^{k+1} \leftarrow x_{i,e^{+}}^{k}$ und $lb_{i,e^{+}}^{k+1} \leftarrow lb_{i,e^{+}}^{k}$ ein Fortschritt von $\sum_{e^{+} \in \delta_{i,v}^{+} \setminus A_{i,v}^{k}} \frac{1}{2} \cdot \left(ub_{i,e^{+}}^{k} - lb_{i,e^{+}}^{k} \right) \cdot \left(1 - \frac{\Delta x_{i,v}^{k}}{\Delta bd_{i,v}^{k}} \right)$.

Betrachten wir den Fall

$$\sum_{e^{-} \in A_{i,v}^{k}} ub_{i,e^{-}}^{k} - lb_{i,e^{-}}^{k} \leq \sum_{e^{+} \in \delta_{i,v}^{+} \setminus A_{i,v}^{k}} ub_{i,e^{+}}^{k} - lb_{i,e^{+}}^{k}, \tag{3.21}$$

so lässt sich der Gesamtfortschritt abschätzen durch

$$\begin{aligned}
&\sum_{e^{-} \in A_{i,v}^{k}} \frac{1}{2} \left(ub_{i,e^{-}}^{k} - lb_{i,e^{-}}^{k} \right) \cdot \left(1 + \frac{\Delta x_{i,v}^{k}}{\Delta bd_{i,v}^{k}} \right) \\
&+ \sum_{e^{+} \in \delta_{i,v}^{+} \setminus A_{i,v}^{k}} \frac{1}{2} \left(ub_{i,e^{+}}^{k} - lb_{i,e^{+}}^{k} \right) \cdot \left(1 - \frac{\Delta x_{i,v}^{k}}{\Delta bd_{i,v}^{k}} \right) \\
&\geq \sum_{e \in A_{i,v}^{k}} \frac{1}{2} \cdot \left(ub_{i,e}^{k} - lb_{i,e}^{k} \right) \cdot \left(1 + \frac{\Delta x_{i,v}^{k}}{\Delta bd_{i,v}^{k}} \right) \\
&+ \sum_{e \in A_{i,v}^{k}} \frac{1}{2} \cdot \left(ub_{i,e}^{k} - lb_{i,e}^{k} \right) \cdot \left(1 - \frac{\Delta x_{i,v}^{k}}{\Delta bd_{i,v}^{k}} \right) \\
&= \sum_{e \in A_{i,v}^{k}} ub_{i,e}^{k} - lb_{i,e}^{k}.
\end{aligned}$$

Diese Rechnung funktioniert analog falls in (3.21) das Zeichen „$\leq$“ durch „$\geq$“ ersetzt wird und steht im Einklang mit (3.19). □

Bemerkung 3.6. Die Abschätzung im Beweis von Lemma 3.5 lässt sich noch leicht verbessern, wenn nicht nur (3.21) gilt, sondern sogar:

$$\left(1+\frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}\right)\cdot\sum_{e^-\in A^k_{i,v}} ub^k_{i,e^-}-lb^k_{i,e^-}\leq\left(1-\frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}\right)\cdot\sum_{e^+\in\delta^+_{i,v}\setminus A^k_{i,v}} ub^k_{i,e^+}-lb^k_{i,e^+}. \tag{3.22}$$

In der Abschätzung des Gesamtfortschritts ergibt sich dann die Untergrenze

$$\left(1+\frac{\Delta x^k_{i,v}}{\Delta bd^k_{i,v}}\right)\cdot\sum_{e\in A^k_{i,v}} ub^k_{i,e}-lb^k_{i,e}. \tag{3.23}$$

Diese beiden Lemmata liefern einen Einblick in das Konvergenzverhalten bei der Bestimmung der Schranken: Zu jedem (i,v) - Paar wird in jedem Verfeinerungsschritt die gesamte Intervalllänge zulässiger Flusswerte von nicht-fixierten, ausgehenden Kanten um die in (3.16) und (3.19) angegebenen Werte reduziert. Aufgrund der Beschaffenheit dieser Untergrenzen wird deutlich, dass falls eine wiederholte Anwendung dieser Lemmata, ohne zwischenzeitliche Relaxierung der Schranken, möglich ist, zwangsläufig alle Flusswerte fixiert werden. Die einzige Möglichkeit, wie die Konvergenz verhindert werden kann, ist indem während der Laufzeit unendlich viele Relaxierungen stattfinden. Auch wenn an dieser Stelle keine Beschränkung der Relaxierungsschritte bekannt ist, wird erhofft, dass aufgrund der Wahl der Relaxierung der Fall der Divergenz nicht auftritt: Die Ursache hinter dem Vorkommen falscher Schranken liegt darin, dass diese zu einem Zeitpunkt gesetzt wurden, als die Approximation des feinen IDE-Flusses zu ungenau war. Demnach scheint es umso wahrscheinlicher, dass keine fehlerhaften Schranken auftreten, je genauer die Approximation des feinen IDE-Flusses ist. Eine genauere Approximation geht einher mit ähnlicheren Werten für die Steigung a, d.h., dass diese eine gleichmäßige Änderung der Kosten verspricht. Die in der Beschreibung von `relax_bounds()` in Algorithmus 6 erklärten Wahlen bei der Relaxierung der Schranken fördern eine solche Angleichung der a-Werte: Die Relaxierung erfolgt immer in eine Richtung, in der der Ausgleich bestmöglich ist. Dennoch halte ich es für wahrscheinlich, dass Spezialfälle existieren, in denen der Algorithmus nicht konvergiert. Mir ist es jedoch bis jetzt nicht gelungen, ein solches Beispiel zu konstruieren.

Nach jedem Aktualisierungsschritt der Schranken lb^k, ub^k wird entweder mittels der Zeilen **29** - **31** die aktuelle **while** - Schleife unterbrochen, da Flusswerte approximiert werden können, oder dies ist nicht der Fall und es wird mit den Zeilen **27** - **33** eine erneute Verfeinerung der Schranken lb^k, ub^k vorbereitet. Dies geschieht bis alle Flusswerte approximiert sind.

3.2 Die Schrittweite α

Haben wir die korrekte Flussaufteilung für einen Zeitpunkt gefunden, so ist das nächste Ziel, die Schrittweite zu bestimmen, mit der der Fluss maximal lange erweitert

werden kann ohne Zulässigkeitsbedingungen oder die IDE-Eigenschaften zu verletzen. Wie bereits in Abschnitt 2.3 angekündigt, müssen dazu drei Fälle berücksichtigt werden, in denen eine Phase beendet wird: Eine Warteschlange wird 0 (2.27), eine inaktive Kante wird aktiv (2.28), oder der Einfluss in einen Knoten ändert sich (2.29).

Ein erschwerendes Element dabei ist, dass das Vorgehen in Algorithmus 1 im Allgemeinen nicht die exakte Flussaufteilung liefert, sondern nur eine Approximation davon. Dies birgt Probleme für das weitere Vorgehen, was an folgendem Beispiel 3.7 veranschaulicht werden soll. Zuvor führen wir noch eine Konvention ein:

Für den Fall, dass hier oder im Folgenden die von Algorithmus 1 berechneten approximierten Flüsse mit den tatsächlichen, exakten IDE-Flüssen für die selbe Instanz verglichen werden, verwenden wir für alle auftretenden Größen im approximierten Modell das Akzentzeichen $\widetilde{\cdot}$, während ein Fehlen einer solchen Tilde auf Zugehörigkeit zum exakten Modell hindeutet. Dies geschieht nur dann, wenn diese Unterscheidung für das bessere Verständnis auch notwendig ist. In Fällen, in denen aus dem Kontext klar ist, dass sich die auftretenden Größen auf das approximierte Modell beziehen, wird die Tilde weggelassen.

Erstmals notwendig wird diese Konvention im nachstehenden Beispiel:

Beispiel 3.7. Gegeben sei der Graph aus Abbildung 3: Kante (a, t_2) habe Reiselänge $\tau_{(a,t_2)} = 3$, alle anderen Reiselängen seien 1. Weiter sei (s_1, b) mit $\nu_{(s_1,b)} = 2$ die einzige Kante mit einer von 1 verschiedenen Kantenkapazität.

Zum Zeitpunkt $\theta = 0$ existiere der in der Abbildung angedeutete Einfluss von 5, bzw. 2 Flusseinheiten von Gut 1 in den Knoten s_1, bzw. c. Die eindeutige Flussaufteilung in Knoten s_1, welche die IDE-Bedingungen erfüllt, schickt vier Flusseinheiten in Kante (s_1, b) und eine Flusseinheit über Kante (s_1, a). Dies geschieht, weil der in c startende Fluss eine Warteschlange bei Kante (c, t_1) verursacht und somit die Kosten dieser Kante mit Rate 1 erhöht. Somit haben beide Wege zu Beginn Kosten 3 und eine Kostenänderung von $+1$.

Der zweite in Abbildung 3 dargestellte Graph zeigt die Situation zum Zeitpunkt $\theta = 1$. Die Warteschlangen von Kanten (s_1, b) und (c, t_1) haben die Länge $2 = \nu_{(s_1,b)}$, bzw. $1 = \nu_{(c,t_1)}$ erreicht und die externen Einflussraten ändern sich wie angezeigt. Für den in s_1 zugeführten Fluss von Gut 2 sind beide ausgehenden Kanten aktiv, aufgrund der durch den Einfluss in e verursachten zusätzlichen Kosten erfolgt der gesamte Einfluss in Kante (s_1, a).

Zum Zeitpunkt $\theta = 1.5$ endet der externe Einfluss in die Knoten d und e, die Flusswerte in den anderen Knoten bleiben unverändert. Zeitpunkt $\theta = 2$ liefert alle drei Ereignisse die für die Erweiterung des IDE-Flusses mittels einer zulässigen Schrittweite relevant sind gleichzeitig: Die Warteschlangen von (s_1, b) und (c, t_1) werden 0, (s_1, b) wird aktiv für Gut 2 und zusätzlich ändert sich der Einfluss in die Knoten a, c, d, t_1 und t_2. All dies sind Gründe dafür, dass $\theta = 2$ der Beginn einer neuen Phase sein muss, oder äquivalent, dass die längste zulässige Schrittweite zum Zeitpunkt $\theta = 1.5$ gerade $\alpha = 0.5$ ist.

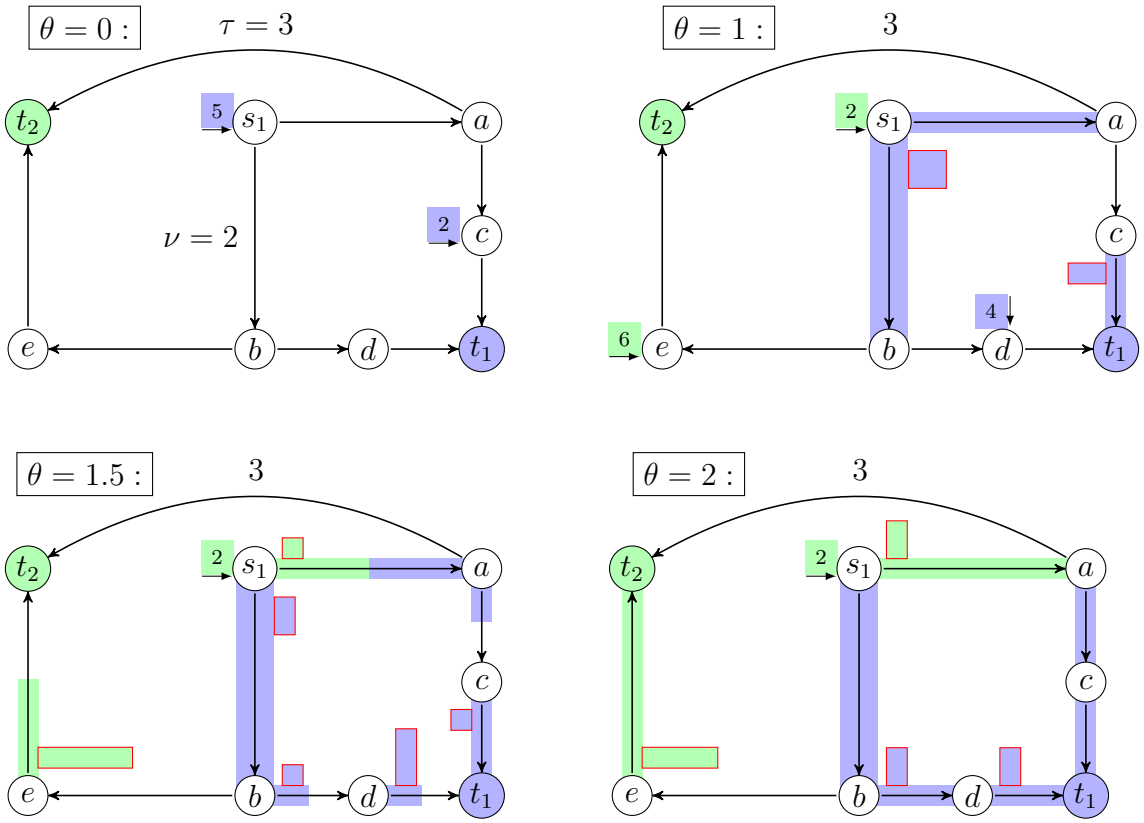


Abbildung 3: Beispiel

Angenommen, anstelle der eben beschriebenen exakten Flusswerte die zu einem IDE-Fluss führen, werden nun die mit `fp_approx(`θ`)` bestimmten Approximationen dieser Flusswerte verwendet um den Fluss zu bestimmen. Diese Abweichungen können mit einer beliebigen Toleranz eingegrenzt werden, da die in diesem Beispiel auftretenden Werte sehr einfach sind, reicht bereits eine Toleranz von beispielsweise $\epsilon = 10^{-4}$ aus, um sinnvolle Ergebnisse zu erhalten. Folgende Tabelle zeigt die entscheidenden Abweichungen der berechneten approximierten Flusswerte, gerundet auf sechs Nachkommastellen:

Phase	$\widetilde{\theta}$	abw. Flusswerte	abw. Warteschlangen
1	0	$\widetilde{f}^+_{1,(s_1,a)}(\widetilde{\theta}) = 0.999929$ $\widetilde{f}^+_{1,(s_1,b)}(\widetilde{\theta}) = 4.000071$	-
2	1	$\widetilde{f}^+_{1,(a,c)}(\widetilde{\theta}) = 0.999929$	$\widetilde{q}_{(s_1,b)}(\widetilde{\theta}) = 2.000035$
3	1.5	$\widetilde{f}^+_{1,(a,c)}(\widetilde{\theta}) = 0.999929$	$\widetilde{q}_{(s_1,a)}(\widetilde{\theta}) = 0.5$ $\widetilde{q}_{(s_1,b)}(\widetilde{\theta}) = 1.000018$ $\widetilde{q}_{(c,t_1)}(\widetilde{\theta}) = 0.5$
4	2.000009	$\widetilde{f}^+_{1,(c,t_1)}(\widetilde{\theta}) = 1$	$\widetilde{q}_{(s_1,a)}(\widetilde{\theta}) = 1.000009$ $\widetilde{q}_{(b,d)}(\widetilde{\theta}) = 1.000009$ $\widetilde{q}_{(d,t_1)}(\widetilde{\theta}) = 0.999991$ $\widetilde{q}_{(e,t_2)}(\widetilde{\theta}) = 1.999991$

Tabelle 1: Approximierte Flusswerte

Durch die Approximationsfehler der berechneten Flusswerte ergeben sich Ungenauigkeiten für den weiteren Verlauf des Flusses. Beispielsweise beträgt die Warteschlange von Kante (s_1, b) zu Beginn der zweiten Phase nicht genau 2 wie in Abbildung 3, sondern $\widetilde{q}_{(s_1,b)}(1) = 2.000035$. Diese geringe Abweichung hat zur Folge, dass die Warteschlange im weiteren Verlauf auch geringfügig später wieder abgebaut ist, weshalb sich eine *zusätzliche* Phase $\widetilde{\theta} \approx 2 + \epsilon$ ergibt. Derartige zusätzliche Phasen sind problematisch aus zweierlei Gründen: Erstens bedeuten sie zusätzlichen Rechenaufwand, da der Algorithmus anstelle einer einzelnen Iteration wie im exakten Fall, zusätzliche Iterationen durchläuft, in denen nahezu identische Berechnungen stattfinden, da zwischen den Iterationen nur Zeitschritte in der Größenordnung von ϵ liegen. Weiter kann der Fall auftreten, dass eine Kante zu früh inaktiv wird, nur um etwa ϵ Zeiteinheiten später wieder aktiv zu werden, da der Fluss diesen Fehler wieder ausgleicht. Es ergeben sich also weitere und teilweise auch größere Ungenauigkeiten.

Um dies zu umgehen, bestimmen wir die eigentliche Schrittweite α_0 bis einer der obigen drei Fälle eintritt, betrachten jedoch *zusätzlich* jedes weitere Vorkommnis eines der drei Fälle innerhalb des Zeitintervalls $[\theta_p + \alpha_0,\ \theta_p + \alpha_0 + \varepsilon_p]$, wobei θ_p den Startzeitpunkt von Phase p, und $\varepsilon_p > 0$ einen weiteren Toleranzparameter in der Größenordnung von ϵ beschreibt, welcher im folgenden Abschnitt 3.3 genauer erklärt wird. Bei allen Ereignissen die in dieses Intervall fallen, wollen wir davon ausgehen, dass die minimal abweichenden Zeitpunkte aufgrund von Approximationsfehlern auftreten und daher als gleichzeitig behandelt werden sollten. Wir verwenden dann als Schrittweite nicht den Wert α_0, sondern den Wert $\alpha = 0.5 \cdot (\alpha_0 + \alpha_{-1})$, wobei α_{-1} den größten, für den Fluss relevanten, Zeitpunkt im Intervall $[\alpha_0,\ \alpha_0 + \varepsilon_p]$ bezeichne[3].

[3] Es gibt eine Ausnahme für diese Wahl von α: Beinhaltet das Intervall $[\theta_p + \alpha_0,\ \theta_p + \alpha_0 + \varepsilon_p]$ einen Zeitpunkt θ_{sp}, zu dem sich eine externe Einflussrate in einen Knoten ändert, so wird $\alpha = \theta_{sp} - \theta$ gewählt

In Beispiel 3.7 wird dies ersichtlich anhand des Anfangszeitpunkts $\widetilde{\theta} = 2.000009$ von Phase 4: Dieser wird in Phase 3 bestimmt, in der $\alpha_0 = 0.5$ gilt. Es ergeben sich folgende zusätzliche Phasen innerhalb des Intervalls $[0.5, 0.5 + \epsilon]$:

- 0.5: Aufgrund der Änderung der Werte $\widetilde{f}^-_{1,(s_1,a)}$, $\widetilde{f}^-_{1,(a,c)}$, $\widetilde{f}^-_{1,(b,d)}$, $\widetilde{f}^-_{1,(d,t_1)}$, $\widetilde{f}^-_{2,(s_1,b)}$, $\widetilde{f}^-_{2,(e,t_2)}$, sowie dem vollständigen Abbau der Warteschlange von Kante (c, t_1)
- 0.500006: Aufgrund des Aktivwerdens der Kante (s_1, b) für Gut 2
- 0.500018: Aufgrund des vollständigen Abbaus der Warteschlange von (s_1, b).

Damit ergibt sich die Schrittweite $\alpha = 0.5 \cdot (0.5 + 0.500018) = 0.5000009$ und somit der Schritt für $\widetilde{\theta}$ von $1.5 \longrightarrow 2.000009$, welcher all diese Ereignisse umfasst.

Die Ursachen der Vergrößerung der Schrittweite sind in dem nachstehendem Schaubild zusammengefasst. Die verwendeten Pfeile sind dabei als Folgepfeile zu verstehen.

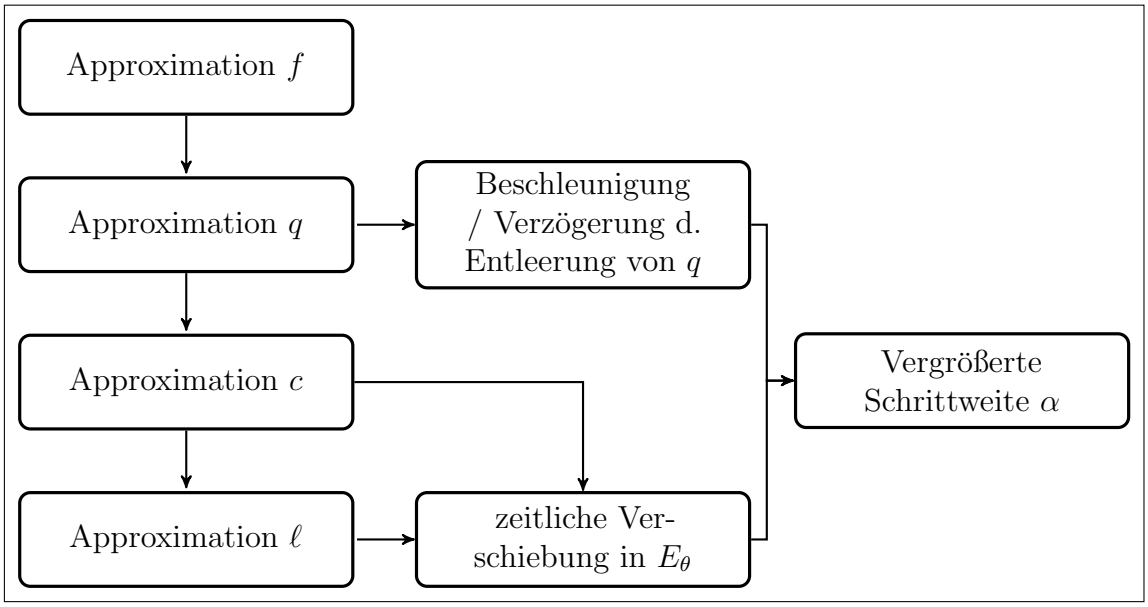


Abbildung 4: Auswirkungen bei der Verwendung von approximierten Flusswerten

Der Ausgangspunkt ist der linke obere Teil der Abbildung: Die Approximationsfehler bei der Wahl der Flusswerte f greifen über auf die Warteschlangenlängen q: Verringerungen der Flusswerte einzelner Kanten führen zu geringeren Auf - und Abbauraten der korrespondierenden Warteschlangen. Analog bringen erhöhte Flusswerte auch erhöhte Änderungsraten der Warteschlangen mit sich. Nach Definition wirkt sich dies auf die Kantenkosten c und damit auch auf die Knotendistanzfunktionen ℓ aus. Da die letzten beiden Größen die Mengen der aktiven Kanten festlegen, ziehen Approximationsfehler in diesen Größen auch zeitliche Verschiebungen der Mengen E_θ mit sich. Die Mengen der approximierten aktiven Kanten $\widetilde{E}_\theta$ sollen zwar im Lauf der Zeit die gleichen Kanten beinhalten wie die Mengen der aktiven Kanten E_θ unter einem exakten IDE-Fluss, wir fordern aber nicht, dass diese zu jedem Zeitpunkt

gleich sind. Es reicht aus, wenn für alle Zeitpunkte θ ein Zeitpunkt $\widetilde{\theta} \approx \theta$ existiert, mit $\widetilde{E}_{\widetilde{\theta}} = E_\theta$.

Insgesamt führen diese Verschiebung des Zeitparameters der aktiven Kanten, sowie die Verzögerung beim Abbau der Warteschlangen notwendigerweise zu einer erhöhten Schrittweite im approximierten Fall.

Infolgedessen wird eine weitere Komplikation bei der Betrachtung approximierter IDE-Flüsse deutlich: Während für exakte IDE-Flüsse nach Definition (2.2) für eine aktive Kante $e = (v, w) \in E^i_\theta$ die Gleichung

$$\ell^i_v(\theta) = \ell^i_w(\theta) + c_e(\theta) \tag{3.24}$$

gilt, ist die Erfüllung einer derartigen Gleichung im approximierten Fall nicht möglich. Wir betrachten dazu ein Beispiel:

Beispiel 3.8. Gegeben sei die Situation in Abbildung 5. Diese zeigt drei Knoten s, v_1 und v_2, sowie die approximierten Knotendistanzlabels von zwei verschiedenen Gütern für die Knoten v_1 und v_2, zu einem festen Zeitpunkt.

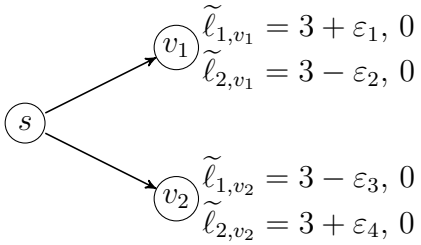


Abbildung 5: Ausschnitt eines Graphen, mit approximierten Knotenlabels

Wir wollen weiter annehmen, dass die exakten Knotenlabels der Knoten v_1, v_2 alle den Wert 3 betragen, sich im approximierten Fall dagegen zu früheren Zeitpunkten die Approximationsfehler ε_1, ε_2, ε_3, $\varepsilon_4 > 0$ ergeben haben, diese jedoch klein genug sind, sodass beide Kanten für beide Güter approximiert aktiv sind. Formulieren wir nun fälschlicherweise die Gleichungen

$$\widetilde{\ell}_{1,s} = \widetilde{\ell}_{1,v_1} + \widetilde{c}_{(s,v_1)}, \quad \widetilde{\ell}_{1,s} = \widetilde{\ell}_{1,v_2} + \widetilde{c}_{(s,v_2)} \tag{3.25}$$

$$\widetilde{\ell}_{2,s} = \widetilde{\ell}_{2,v_1} + \widetilde{c}_{(s,v_1)}, \quad \widetilde{\ell}_{2,s} = \widetilde{\ell}_{2,v_2} + \widetilde{c}_{(s,v_2)} \tag{3.26}$$

analog zu (3.24), so sind die Werte $\widetilde{\ell}_{1,s}$ und $\widetilde{\ell}_{2,s}$ offensichtlich nicht wohldefiniert. Dies liegt daran, dass obige Gleichungen die gutsunabhängige Kostenfunktion $\widetilde{c}$ für die Definition der gutsabhängigen Knotenlabels $\widetilde{\ell}$ verwenden. Wir erhalten demnach unterschiedliche Werte für $\widetilde{\ell}_{1|2,s}$, je nachdem welche der approximiert aktiven Kanten betrachtet wird.

Beachte: Diese Argumentation ist kein Zirkelschluss, da die ursprünglich vorgegebenen Knotenlabels $\widetilde{\ell}_{1|2,v_1|v_2}$ *exakt* sein können für ihre approximierten Flusswerte. Die Approximation der Labels $\widetilde{\ell}_{1|2,s}$ umfasst dann noch einen zusätzlichen Grad der Approximation, sind also selbst nur Approxmationen zu den approximierten Flusswerten.

3.3 Der Toleranzbereich bei der Schrittweitenwahl

Es bleibt die allgemeine Wahl von ε_p: Dieser Parameter soll stets groß genug sein, um alle Abweichungen von Ereignissen durch Approximationsfehler abzudecken, gleichzeitig aber auch möglichst klein, um das Risiko fälschlicher Zusammenführungen von Phasen zu minimieren. Offensichtlich ist, dass die Größe der vergangenen Zeit eine entscheidende Rolle spielt. Während einer bestimmten Phase $[\theta_p, \theta_{p+1})$ bleiben alle Änderungsraten der Approximationsfehler konstant, sodass der resultierende absolute Fehler umso größer wird, je länger die Phase andauert. Wenn dann eine obere Schranke an die Änderungsrate des Fehlers während einer Phase gegeben ist, kann die gesamte Änderung des Fehlers während dieser Phase beschränkt werden durch diese Obergrenze multipliziert mit der Länge der Phase, also der Schrittweite α. Ist weiterhin eine Begrenzung max_dif_p des Fehlers zu Beginn der Phase gegeben, kann durch aufsummieren dieser beiden Werte eine Schranke an den gesamten bis zum Zeitpunkt θ_{p+1} auftretenden Fehler berechnet werden.

Die hier beschriebenen Toleranzen werden im Anschluss zwar ohne formalen Korrektheitsbeweis verwendet, wurden jedoch in mühevoller Kleinarbeit entwickelt und haben sich in allen Experimenten als zweckmäßig erwiesen.

Zuerst wird die Variable max_dif_p eingeführt, welche die durch Approximationsfehler zusätzlich entstandene Wartezeit zum Beginn von Phase p beschränken soll. Wir wählen für $p \geq 1$:

$$max\_dif_p = \max \left\{ max\_dif_{p-1}, \max_{i \in I, v \in V, e \in \delta^+_{i,v}(\theta_p)} |\ell^i_v(\theta_p) - \ell^i_w(\theta_p) - c_{(v,w)}(\theta_p)| \right\}, \tag{3.27}$$

wobei $max\_dif_0 := \epsilon$.

Der Wert max_dif_p wird also gesetzt nach der Berechnung des feinen IDE-Flusses (x, a^k) und der zugehörigen Schrittweite bis zum Zeitpunk θ_p, sowie der resultierenden Aktualisierung der Variablen q, c und ℓ in Phase $p-1$ als die größte betragliche Abweichung des Werts $\ell^i_v(\theta_p) - \ell^i_w(\theta_p) - c_{(v,w)}(\theta_p)$, für $(v,w) \in \delta^+_{i,v}(\theta_p)$, von 0. Dabei ist zu beachten, dass es sich bei den Werten in ℓ und c lediglich um Approximationen der tatsächlichen Knotendistanzen und Kantenkosten handelt (siehe Beispiel 3.8), weshalb Korrektheit nicht garantiert werden kann und sich der heuristische Charakter dieser Methode aufzeigt.

Insgesamt sollen damit drei verschiedene Fehlerquellen abgedeckt werden:

1. Für die Änderung einzelner $\widetilde{f^-}$ - Werte verwenden wir für Gut $i \in I$ in Phase p die Toleranz

$$f\_tol_{i,e} = max\_dif_p + \frac{\left(\frac{\epsilon}{\nu_e} + \frac{\epsilon}{\nu_{e_{a_i}}}\right) \cdot \alpha_0}{\nu_e}, \text{ mit } v \in V, e \in \delta^+_v, \tag{3.28}$$

   wobei $e_{a_i} \in \delta^+_v$ die erste Kante in $A^k_{i,v}$ bezeichne, also insbesondere eine Kante, über die der Wert $a^k_{i,v}$ exakt erreicht wird. Die Toleranz (3.28) setzt sich zusammen aus der Beschränkung max_dif_p an die zu Beginn von Phase p bereits existierende zusätzliche Wartezeit, sowie der Abschätzung der in dieser

Phase weiter entstehenden zusätzlichen Wartezeit von $(\epsilon/\nu_e + \epsilon/\nu_{e_{a_i}}) \cdot \alpha_0/\nu_e$. Der erste Faktor ergibt sich aus (3.9) und wurde bereits in der Beschreibung von Algorithmus 5 erklärt, als die Schranke an den Approximationsfehler in der Einflussrate von Gut i in Kante e. Multipliziert mit α_0 erhalten wir eine Schranke an die tatsächliche Gesamtflussmenge und dividiert durch ν_e die daraus resultierende zusätzliche Wartezeit.

2. Erreicht die Warteschlange einer Kante e mit Startknoten v den Wert 0, so wird der dabei verwendete Toleranzbereich gewählt als

$$q\_tol_e = max\_dif_p + \frac{2 \cdot \frac{\epsilon}{\nu_{min}} \cdot \alpha_0}{-g_e(x_e)}, \quad \text{mit } \nu_{min} = \min_{e' \in \delta_v^+} \nu_{e'}. \tag{3.29}$$

Neben max_dif_p beinhaltet diese Toleranz wieder eine Beschränkung an den in dieser Phase maximal entstehenden Fehler. Da die Warteschlangen gutsunabhängig sind, wird die Toleranz (3.9) hier abgeschätzt mit Hilfe des Werts $2 \cdot \epsilon/\nu_{min}$. Der Nenner des zweiten Summanden beschreibt diesmal die genaue Rate, mit der das zusätzliche Flussvolumen abgebaut wird. Wir setzen dabei voraus, dass $g_e(x_e) < -2 \cdot \epsilon/\nu_{min}$. Diese Annahme ist sinnvoll, da wir davon ausgehen wollen, dass die Warteschlange einer Kante während einer Phase nur dann vollständig erschöpft werden kann, wenn die Abbaurate in dieser Phase um mehr als einen Approximationsfehler von 0 verschieden ist.

3. Beim Test des Aktivwerdens einer für ein Gut $i \in I$, vom Knoten $v \in V$ ausgehenden Kante $e^- = (v, w^-) \in \delta_v^+ \setminus \delta_{i,v}^+(\theta_p)$ sei $e^+ = (v, w^+) \in \delta_{i,v}^+(\theta_p)$ eine in Phase p für Gut i aktive Kante. Wir wählen

$$np\_tol_{i,e^-} = max\_dif_p + \frac{\frac{\left(\frac{\epsilon}{\nu_e} + \frac{\epsilon}{\nu_{e_{a_i}}}\right) \cdot \alpha_0}{\nu_{e^+}} + \frac{\left(\frac{\epsilon}{\nu_e} + \frac{\epsilon}{\nu_{e_{a_i}}}\right) \cdot \alpha_0}{\nu_{e^-}}}{\frac{g_{e^+}(x_{e^+})}{\nu_{e^+}} + a_{i,w^+}^k - \frac{g_{e^-}(x_{e^-})}{\nu_{e^-}} - a_{i,w^-}^k}, \tag{3.30}$$

wobei $e_{a_i} \in \delta_v^+$ wieder die erste Kante in $A_{i,v}^k$ bezeichne. Im Zähler des zweiten Summanden erfolgt die gleiche Abschätzung wie in (3.9), für beide betrachteten Kanten. Damit werden die durch Approximationsfehler möglichen zusätzlichen Kosten für Kante e^-, sowie die potentiell verringerten Kosten von Kante e^+ erfasst. Der Nenner stellt die Änderungsraten der Kosten beider eingeleiteten aktiven Wege dar, mit denen der Fehler abgebaut wird.

Letztendlich wird

$$\varepsilon_p = \max\{\max_{i \in I, e \in E} f\_tol_{i,e}, \max_{e \in E : g_e(x_e) < -2 \cdot \epsilon/\nu_{min}} q\_tol_e, \max_{i \in I, e^- \in E \setminus E_{\theta_p}^i} np\_tol_{i,e^-}\} \tag{3.31}$$

gewählt, sodass keiner der Werte aus (3.28) - (3.30) unterschritten wird. Es sei dann $\alpha_{-1} \in [\alpha_0,\ \alpha_0 + \varepsilon_p]$ der größte Wert, der auch wirklich in einem dieser drei Fälle erreicht wird. Wie in Beispiel 3.7 beschrieben, wird dann die tatsächliche Schrittweite gewählt als $\alpha = 0.5 \cdot (\alpha_0 + \alpha_{-1})$, falls sich im Intervall $[\theta_p + \alpha_0,\ \theta_p + \alpha_0 + \varepsilon_p]$ keine

externe Einflussrate ändert, ansonsten so, dass der Zeitpunkt dieser Änderung exakt erreicht wird.

3.4 Die Aktualisierung der Daten

In diesem Kapitel werden die Zeilen **12** und **13** aus Algorithmus 1 genauer behandelt. Die im vorigen Abschnitt beschriebene Methode zur Schrittweitenwahl wirft die Frage auf, wie dies bei der Aktualisierung der aktiven Kanten beachtet werden muss. Zuerst betrachten wir die Knotenlabels.

Nach Definition (2.2) gilt für eine aktive Kante $e = (v, w)$ eines Guts i zum Zeitpunkt θ die Gleichung

$$\ell_v^i(\theta) - \ell_w^i(\theta) - c_e(\theta) = 0. \tag{3.32}$$

Da der in diesem Kapitel besprochene Algorithmus nur mit approximierten Werten arbeitet, ist diese Gleichung i.A. nicht erfüllbar, weshalb der Begriff der approximiert aktiven Kanten bereits eingeführt wurde. Um die in (3.5) gegebene Definition zu vervollständigen, ist es notwendig, das Auseinanderdriften der Werte in (3.32) durch eine geeignete Größe abzuschätzen. Dies geschieht in Phase p für einen Zeitpunkt $\theta \in [\theta_p, \theta_{p+1}]$ mit Hilfe der Ungleichung

$$\ell_v^i(\theta) - \ell_w^i(\theta) - c_e(\theta) > -\left(max\_dif_p + \frac{2 \cdot \left(\frac{\epsilon}{\nu_e} + \frac{\epsilon}{\nu_{e_{a_i}}}\right) \cdot (\theta - \theta_p)}{\nu_{i,v}^{min}} \right), \tag{3.33}$$

wobei $\nu_{i,v}^{min} := \min_{e \in \delta_{i,v}^+(\theta)} \nu_e$ und $e_{a_i} \in \delta_v^+$ sei die erste Kante in $A_{i,v}^k$. Wir wählen also

$$\delta := \left(max\_dif_p + \frac{2 \cdot \left(\frac{\epsilon}{\nu_e} + \frac{\epsilon}{\nu_{e_{a_i}}}\right) \cdot (\theta - \theta_p)}{\nu_{i,v}^{min}} \right). \tag{3.34}$$

In Beispiel 3.8 haben wir bereits gesehen, dass die Knotenlabels zu einem approximierten IDE-Fluss notwendigerweise auch approximiert werden müssen. Dabei besteht die Gefahr, dass diese Abweichungen so groß werden, dass das Ergebnis nicht mehr sinnvoll mit einem IDE-Fluss identifiziert werden kann. Tritt beispielsweise der unerwünschte Fall ein, dass in (3.33) der Wert max_dif_p zu groß wird, so können auch eigentlich inaktive Kanten als approximiert aktiv eingestuft werden. Um dem entgegenzuwirken passt die in Algorithmus 8 dargestellte Funktion `refine()` die Labels ℓ daher nochmal an. Die Idee dabei besteht darin, die bereits approximierten Knotenlabels korrespondierend zu approximiert aktiven Kanten so zu justieren, dass die Unterschiede möglichst klein gehalten werden. Dies geschieht, indem das Label ℓ_v^i auf den Durchschnittswert aller $\ell_w^i(\theta) + c_e(\theta)$, für $e = (v, w) \in \delta_{i,v}^+(\theta)$, gesetzt wird. Im Fall dass es nur eine solche Kante gibt, wird damit der exakte Wert $\ell_v^i(\theta) = \ell_w^i(\theta) + c_e(\theta)$ erreicht, andernfalls ein bestmöglicher Kompromiss für all diese Kanten.

Im Anschluss daran werden die Warteschlangen $q_e(\theta)$ angepasst um die Abstände von $q_e(\theta)$ zu $q_{i,e} := (\ell_v^i(\theta) - \ell_w^i(\theta) - \tau_e) \cdot \nu_e$ für alle Güter i mit $e \in \delta_{i,v}^+(\theta)$ im Zaum

zu halten. Dazu wird der Mittelwert all dieser Werte $q_{i,e}$ über die Güter verwendet.

Algorithmus 8: `refine()`

Eingabe: Zeitpunkt θ mit zugehörigen Mengen approximiert aktiver Kanten $\widetilde{E}^i_\theta$ für alle Güter i, sowie topologische Sortierungen $\Phi_{i,\theta}$ auf den approximiert aktiven Teilgraphen. Außerdem aktuelle Knotendistanzen ℓ und Kantenkosten c.
Ausgabe: Aktualisierte Warteschlangenlängen q, Kantenkosten c und Knotendistanzlabels ℓ.
Methode:
for i **in** I **do**
  for v **in** $\Phi_{i,\theta}$ **do**
    $\ell^i_v(\theta) \leftarrow \frac{\sum_{e=(v,w)\in\delta^+_{i,v}(\theta)} \ell^i_w(\theta)+c_e(\theta)}{|\delta^+_{i,v}(\theta)|}$;
    $q_{i,e} \leftarrow (\ell^i_v(\theta) - \ell^i_w(\theta) - \tau_e)\cdot\nu_e$, für $e=(v,w)\in\delta^+_{i,v}(\theta)$;
  end for
end for
for e ***in*** E **do**
  $q_e(\theta) \leftarrow \frac{\sum_{i\in I: e\in E^i_\theta} q_{i,e}}{|\{i\in I: e\in E^i_\theta\}|}$;
  $c_e(\theta) \leftarrow \frac{q_e(\theta)}{\nu_e} + \tau_e$;
end for
return: q, c, ℓ ;

Nach der Definition der Knotendistanzlabels (2.12) kann im exakten Fall in der Gleichung (3.32) niemals „>“ gelten. Aufgrund des Aufrufs von `refine()` am Ende jeder Iteration von Algorithmus 1 kann dies in der approximierten Lösung allerdings durchaus passieren. Ergibt sich dieser Fall, so geschieht das aufgrund von Approximationsfehlern und wir betrachten eine solche Kante für ein solches Gut als approximiert aktiv. In der Ungleichung (3.33) ist dieses Szenario bereits enthalten, weshalb diese nicht weiter angepasst werden muss.

Der Aufruf von `refine()` beendet die jeweilige Iteration von Algorithmus 1 und leitet eine neue Phase ein, im Fall dass weiterhin $\theta < T$. Gilt dagegen am Ende einer Iteration, dass $\alpha = T$, so terminiert der Algorithmus mit einem approximierten IDE-Fluss f. Die letzte Möglichkeit, in der $\theta > T$ und $\alpha < T$ gilt, resultiert ebenfalls in der Termination des Algorithmus; die Ausgabe ist dann ein approximierter IDE-Fluss f bis zum Zeitpunkt θ.

4 Rechnerische Analyse

In diesem Kapitel soll der in Kapitel 3 vorgestellte Algorithmus für verschiedene Beispielinstanzen ausgeführt und die resultierenden Ergebnisse analysiert und interpretiert werden.

4.1 Größen der Fehlerbewertung

Um die Leistungsfähigkeit des Algorithmus zu bewerten, werden nun verschiedene Methoden für die Fehlermessung eingeführt.

Definition 4.1. Es sei $(G, I, \nu, \tau, (u_i)_{i\in I}, (t_i)_{i\in I})$ ein Flussnetzwerk und f ein zugehöriger approximierter IDE-Fluss. Gilt für einen Zeitpunkt $\theta \in \mathbb{R}_{\geq 0}$, einen Knoten $v \in V$ und ein Gut $i \in I$, von welchem sich momentan Fluss in v befindet, dass $|\delta^+_{i,v}(\theta)| > 1$, so bezeichnen

$$Err_{i,v}(\theta) := \max_{e^+=(v,w^+)\in\delta^+_{i,v}(\theta):f^+_{i,e^+}(\theta)>0} \ell^i_{w^+}(\theta) + c_{e^+}(\theta) \tag{4.1}$$

$$- \min_{e^-=(v,w^-)\in\delta^+_{i,v}(\theta)} \ell^i_{w^-}(\theta) + c_{e^-}(\theta) \tag{4.2}$$

und

$$Err^{rel}_{i,v}(\theta) := \frac{Err_{i,v}(\theta)}{b^-_{i,v}(\theta)} \tag{4.3}$$

den *absoluten*, bzw. *relativen IDE-Fehler* von Gut i in Knoten v zum Zeitpunkt θ. Weiter bezeichnen

$$Err(\theta) := \sum_{i\in I, v\in V} Err_{i,v}(\theta) \quad \text{und} \tag{4.4}$$

$$Err^{rel}(\theta) := \sum_{i\in I, v\in V} Err^{rel}_{i,v}(\theta) = \frac{Err(\theta)}{\sum_{i\in I, v\in V\setminus\{t_i\}:Err_{i,v}(\theta)>0} b^-_{i,v}(\theta)} \tag{4.5}$$

den *gesamten absoluten* bzw. *gesamten relativen IDE-Fehler* zum Zeitpunkt θ.

All diese Fehlergrößen sind stückweise linear mit möglichen Sprungstellen an den Randpunkten der Phasen. Handelt es sich in der Situation von Definition 4.1 bei f um einen IDE-Fluss, so sind alle definierten Fehler für jeden Zeitpunkt gleich 0. Ziel einer jeden Approximation eines IDE-Flusses wird demnach sein, die betrachteten Fehlermaße in diesem Bereich zu halten. Um dies zu erreichen, wird im Folgenden auch die Änderungsrate dieses Fehlers betrachtet:

$$\Delta Err_{i,v}(\theta) := \max_{e^+=(v,w^+)\in\delta^+_{i,v}(\theta):f^+_{i,e^+}(\theta)>0} a_{i,w^+} + \frac{g_{e^+}(x_{e^+})}{\nu_{e^+}} \tag{4.6}$$

$$- \min_{e^-=(v,w^-)\in\delta^+_{i,v}(\theta)} a_{i,w^-} + \frac{g_{e^-}(x_{e^-})}{\nu_{e^-}}. \tag{4.7}$$

Für die Beurteilung der Approximation der Knotenlabels führen wir folgendes Maß ein:

Definition 4.2. In der Situation von Definition 4.1 sei $\ell^*_{i,v}(\theta)$ die exakte Distanz von Knoten v zur Senke t_i zum Zeitpunkt θ. Dann ist

$$\lambda_{i,v}(\theta) \coloneqq \widetilde{\ell}_{i,v}(\theta) - \ell^*_{i,v}(\theta) \tag{4.8}$$

der *Fehler im Knotenlabel* von Gut i im Knoten v zum Zeitpunkt θ.

Bemerkung 4.3. Die Berechnung des Fehlers im Knotenlabel ist sehr rechenaufwändig, da die exakten Knotenlabels ℓ^* für jede Phase mit dem Algorithmus von Dijkstra neu berechnet werden müssen. Dazu müssen die exakten Kantenkosten c^* aller Kanten zwischengespeichert werden.

Es wäre denkbar, diese Berechnungen stattdessen effizienter durchzuführen, indem stets die exakten Knotenlabels zwischengespeichert, und mithilfe des Produkts $\alpha \cdot a$ von Schrittweite und Labeländerung aktualisiert werden, sodass die Ausführung des Algorithmus von Dijkstra eingespart werden kann. Dieses Vorgehen führt jedoch nicht immer zu den korrekten Distanzwerten, da die Schrittweite α nicht exakt auf die Änderung der Steigung a abgestimmt ist, sodass die exakte Labeländerung in einer Umgebung in der Größenordnung von ϵ um den Randpunkt einer Phase von a abweicht.

4.2 Praktische Anwendung

In diesem Abschnitt wird die Performance von Algorithmus 1 anhand von verschiedenen Beispielen untersucht. Wir beginnen mit einer Beispielinstanz, für die $n = 13$, $m = 24$ und $I = 3$ gilt und für die der untersuchte IDE-Fluss eindeutig ist. Zunächst folgt eine lokale Betrachtung eines kleinen Ausschnitts von diesem Graphen, mit Berechnung der auftretenden IDE-Fehler. Diese wird anschließend ausgeweitet auf den gesamten Graphen und es werden weitere entscheidende Stellen im Verlauf des IDE-Flusses analysiert. Das Beispiel wird abgeschlossen mit einer Darstellung des global maximal auftretenden IDE-Fehlers, sowie der Fehler im Knotenlabel. Im Anschluss daran wird mit dem Beispiel eines nichtterminierenden IDE-Flusses aus [4] das Verhalten der Fehler über einen längeren Zeitraum betrachtet. Zum Abschluss werden die Rechenkapazitäten geprüft anhand eines großen Beispiels in der Form vom Straßennetz Holzkirchen, für das $n = 3052$, $m = 7004$ und $I = 2$ gilt. Wir beginnen mit einer Notation:

Notation 4.4. Für einen rechtskonstanten dynamischen Fluss f sei $[\theta_p, \theta_{p+1})$ eine Phase. Wir schreiben dann

$$\theta^<_{p+1} = \theta_{p+1} - \varepsilon, \text{ wobei } \varepsilon > 0, \tag{4.9}$$

für einen Zeitpunkt unmittelbar vor dem Ende der Phase.

Die erste Beispielinstanz.
Der zu betrachtende Graph G_1 ist dargestellt in Abbildung 6. Die Abbildung zeigt eine Momentaufnahme des eindeutigen IDE-Flusses zum Zeitpunkt $\theta = 0.4$, wobei die Warteschlangen zur besseren Übersichtlichkeit nur angedeutet werden. Während die Breite einer Kante proportional zu ihrer Kantenkapazität ist, stehen die dargestellten Längen jedoch in keinem Verhältnis zu ihrer tatsächlichen Reiselänge. Daher wurden für alle Kanten mit von 1 verschiedener Kapazität oder Reiselänge die jeweiligen Größen in der Abbildung angegeben in der Form „(ν_e, τ_e)“. Es gibt insgesamt sechs Quellknoten mit externen Einflussraten dreier Güter: Gut 1 (rot), Gut 2 (blau) und Gut 3 (grün). Die genauen Einflussraten sind aufgeführt in folgender Tabelle:

Knoten	s			v_2	v_4	v_5	v_7	v_8		
Einflussrate	3	2	2	3	4	4	7	5	5	5
Einflusszeitraum	$[0,1)$			$[0,1)$	$[0,1)$	$[1,2)$	$[0,2)$	$[0,1)$		

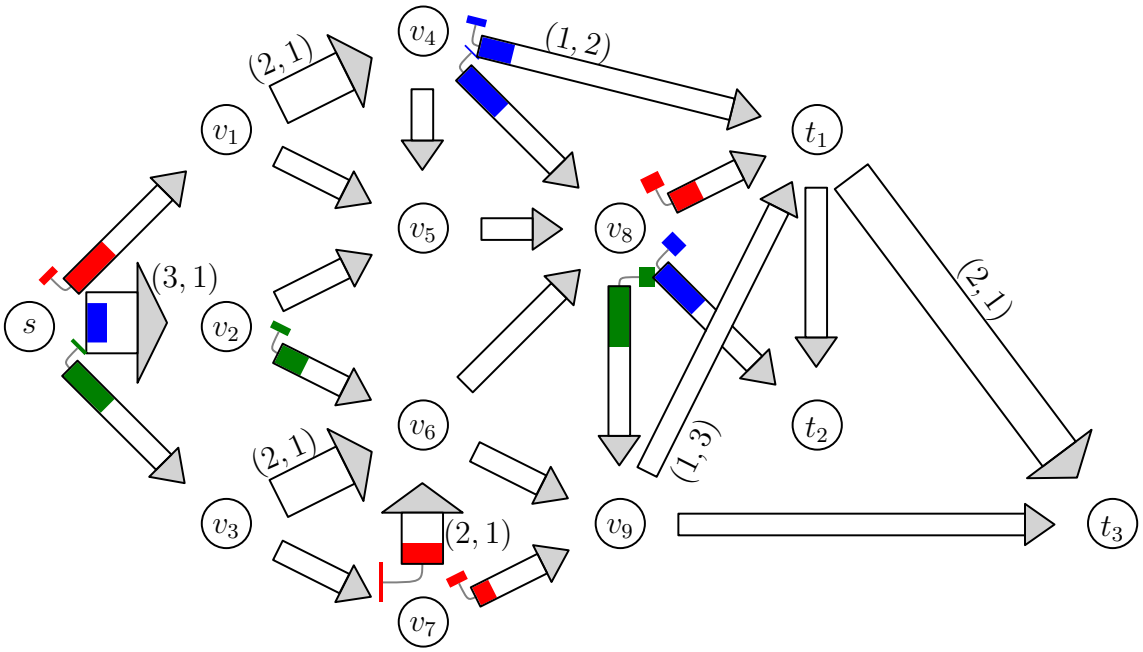


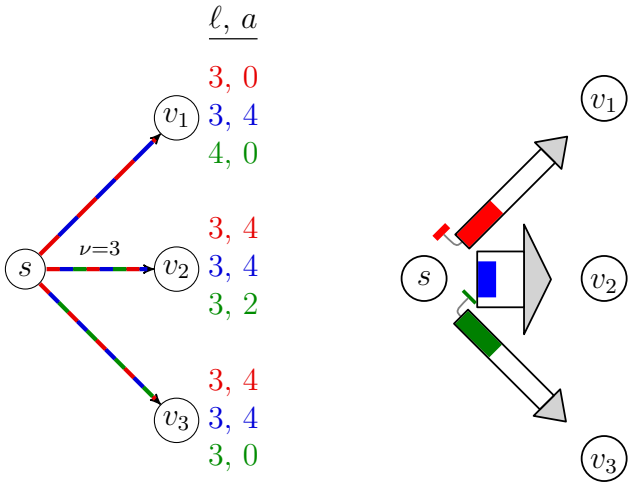
Abbildung 6: Erster Beispielgraph G_1 mit eindeutigem IDE-Fluss zum Zeitpunkt $\theta = 0.4$. Alle Kapazitäten und Reiselängen ungleich 1 sind angegeben in der Form (ν_e, τ_e). Diese und alle weiteren Darstellungen von Flüssen in diesem Kapitel wurden erstellt mit dem Tool zu finden unter https://github.com/ArbeitsgruppeTobiasHarks/dynamic-flow-visualization/tree/main.

Lokale Fehleranalyse.
Wir beschränken uns zunächst auf die Flussaufteilung ausgehend von Knoten s. Betrachte dazu den linken Teil von dem in Abbildung 7 dargestellten Ausschnitt des Graphen. Nach Obigem ist s innerhalb des Zeitintervalls $[0, 1)$ der Quellknoten von 3 Flusseinheiten von Gut 1, 2 Einheiten von Gut 2 und 2 Einheiten von Gut 3. Die erste Spalte der Zahlen neben den Knoten v_1, v_2 und v_3 geben die momentanen Labels ℓ der Knoten, entsprechend der Güter passend zu den verwendeten Farben, an. Die farbliche Markierung der Kanten beschreibt die Menge der Güter, für die die jeweilige Kante zum Zeitpunkt $\theta_1 = 0$ aktiv ist. Weiter nehmen wir an, die momentane Änderungsraten der drei Nachbarknoten von s haben die in der zweiten Spalte von Zahlen dargestellten Werte.

Auch wenn hier zunächst nur der in Abbildung 7 dargestellte Teilgraph betrachtet wird, wurde zur Berechnung der Flusswerte Algorithmus 1 auf das gesamte Netzwerk aus Abbildung 6 angewandt. Die auftauchenden Knotenlabels $\widetilde{\ell}$, Kantenkosten $\widetilde{c}$ und Steigungen $\widetilde{a}$ sind abhängig von dem nicht dargestellten Teil des Graphen und werden hier vorerst als gegeben betrachtet.

Die Tabelle im rechten Teil der Abbildung zeigt für jede Kante und jedes Gut zum Zeitpunkt $\theta_1 = 0$ die eindeutige exakte IDE-Flussaufteilung f^+, die approximierte Flussaufteilung $\widetilde{f}^+$ mit Genauigkeit $\epsilon = 10^{-5}$, sowie die Differenz dieser: $\Delta\widetilde{f}^+_{i,e}(\xi) := f^+_{i,e}(\xi) - \widetilde{f}^+_{i,e}(\xi)$, für $\xi \in [0, \frac{3}{7})$.

Die Grafik im mittleren Teil der Abbildung stellt den resultierenden Fluss zum Zeitpunkt $\theta = 0.4$ dar, wobei die Warteschlangen im Sinne der bestmöglichen Übersichtlichkeit nur angedeutet werden.



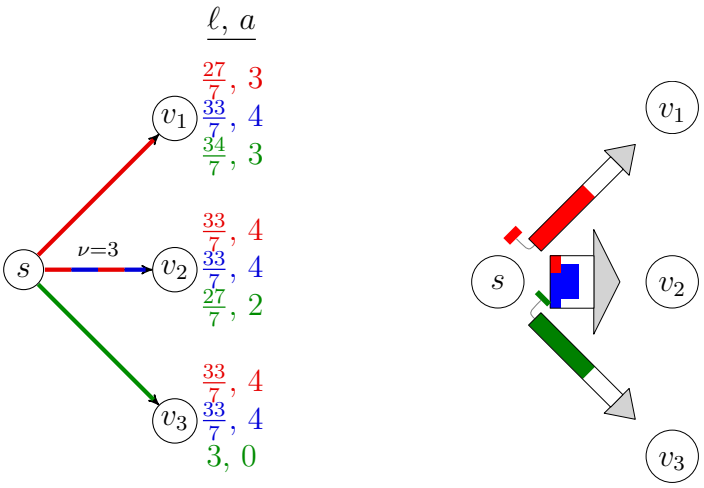
f^+	$\widetilde{f}^+$	$\Delta\widetilde{f}^+$	
3	3	0	
0	0	0	e_1
0	0	0	
0	0	0	
2	2	0	e_2
0	0	0	
0	0	0	
0	0	0	e_3
2	2	0	

Abbildung 7: Aktive Kanten, Knotenlabels und Labeländerungen zum Zeitpunkt $\theta_1 = 0$ (links), resultierender Fluss zum Zeitpunkt $\theta = 0.4$ (Mitte) und Flussaufteilung mit Fehler zum Zeitpunkt θ_1 (rechts).

Für diese erste Phase fällt auf, dass der approximierte Fluss $\widetilde{f}$ mit dem exakten IDE-Fluss f übereinstimmt. Diese Übereinstimmung ist nicht offensichtlich, da alle drei Güter mehrere aktive Kanten zur Verfügung haben und die Flusswerte daher innerhalb des Algorithmus zuerst approximiert werden. Da in diesem Fall alle

Güter sämtliches Flussvolumen über nur jeweils eine Kante schicken, wird dies vom Algorithmus erkannt und die approximierten Werte werden zu den exakten Werten korrigiert.

Der Fluss bleibt für diesen Beispielgraphen erhalten bis zum Zeitpunkt $\theta_2 = \frac{3}{7}$. Während dieser Zeit ändern sich die Änderungsraten a der Knotenlabels der betrachteten Knoten, was jedoch für den untersuchten Graphausschnitt keine sofortige Auswirkungen mit sich bringt, weshalb an dieser Stelle nicht weiter darauf eingegangen wird. Wir betrachten erst wieder den Zeitpunkt θ_2, bei dem sich die in Abbildung 8 dargestellte Situation ergibt:



$f^+(\theta_2)$	$\widetilde{f}^+(\widetilde{\theta}_2)$	$f^+(\theta_2) - \widetilde{f}^+(\widetilde{\theta}_2)$	
2	2.00000000030559	$-3.056 \cdot 10^{-10}$	
0	0	0	e_1
0	0	0	
1	0.999999999694409	$3.056 \cdot 10^{-10}$	
2	2	0	e_2
0	0	0	
0	0	0	
0	0	0	e_3
2	2	0	

Abbildung 8: Aktive Kanten, Knotenlabels und Labeländerungen zum Zeitpunkt $\theta_2 = \frac{3}{7}$ (oben links), resultierender Fluss zum Zeitpunkt $\theta = 0.66$ (oben rechts) und Flussaufteilung mit Fehler zum Zeitpunkt θ_2, bzw. $\widetilde{\theta}_2$ (unten).

Die Mengen der aktiven Kanten werden erneut durch die Färbung der Kanten im oberen linken Teil der Abbildung visualisiert. Ebenso sind die aktuellen Labels und Labeländerungen angegeben. Rechts daneben ist der approximierte Fluss zum Zeitpunkt $\theta = 0.66$ abgebildet und im unteren Teil werden die Flüsse f und $\widetilde{f}$ verglichen:

Die Flusswerte der Güter 2 und 3 sind hier eindeutig: Da beide Güter nur jeweils eine aktive Kante zur Verfügung haben, gibt es nur eine mögliche Flussaufteilung. Die Wahl der Werte von Gut 1 ist dagegen weniger offensichtlich und es treten erstmals Approximationsfehler auf. Damit ergibt sich eine Änderung des IDE-Fehlers von

$$\begin{aligned}\Delta Err_{1,s}(\widetilde{\theta}_2) = & \\ = \widetilde{a}_{1,v_1} + \frac{g_{(s,v_1)}(2 + 3.0559 \cdot 10^{-10})}{\nu_{(s,v_1)}} - \left(\widetilde{a}_{1,v_2} + \frac{g_{(s,v_2)}(3 - 3.0559 \cdot 10^{-10})}{\nu_{(s,v_2)}}\right) & \\ = 3 + 1.000000000305591 - (4 + 0) = 3.05591 \cdot 10^{-10}. &\end{aligned} \tag{4.10}$$

Bemerkung 4.5. Die Steigungen $\widetilde{a}_{1,v_1} = 3 = a_{1,v_1}$ und $\widetilde{a}_{1,v_2} = 4 = a_{1,v_2}$ wurden hier vom Algorithmus exakt bestimmt.

Phase 2 dauert an bis zum Zeitpunkt $\theta_3 = \frac{2}{3}$. Während dieser Zeit ergibt sich ein IDE-Fehler von

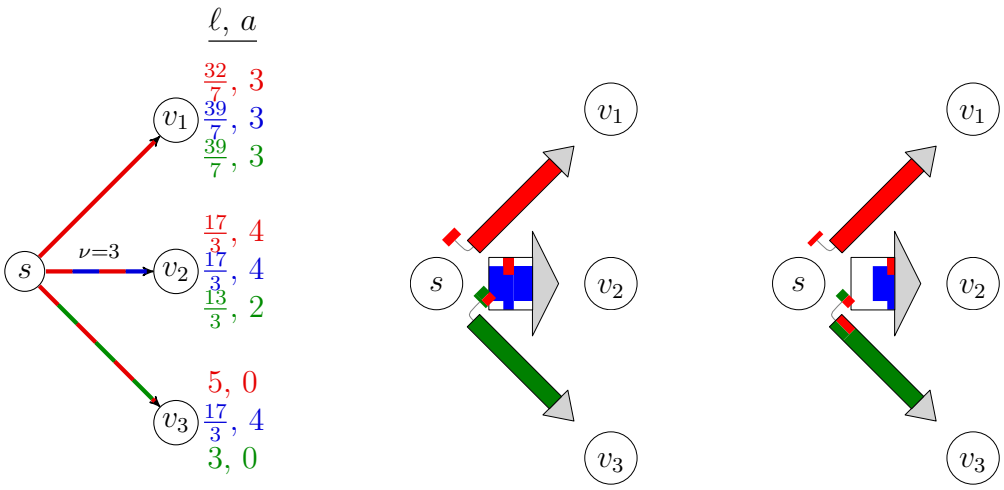
$$\begin{aligned} Err(\widetilde{\theta}_3^<) = Err_{1,s}(\widetilde{\theta}_3^<) &= \widetilde{\ell}_{1,v_1}(\widetilde{\theta}_3^<) + \widetilde{c}_{e_1}(\widetilde{\theta}_3^<) - \left(\widetilde{\ell}_{1,v_2}(\widetilde{\theta}_3^<) + \widetilde{c}_{e_2}(\widetilde{\theta}_3^<)\right) \\ &= 4.571428571383 + 2.095238095268 - (5.666666666606 + 1) \\ &= 4.5474 \cdot 10^{-11} \end{aligned} \tag{4.11}$$

und somit ein relativer IDE-Fehler von

$$Err^{rel}(\widetilde{\theta}_3^<) = \frac{Err(\widetilde{\theta}_3^<)}{b_{1,s}^-(\widetilde{\theta}_3^<)} = 1.5158 \cdot 10^{-11}. \tag{4.12}$$

Die Zwischenergebnisse wurden dabei auf 12 Nachkommastellen gerundet.

Ist $\theta_3 = \frac{2}{3}$ erreicht, so beginnt die dritte Phase und es ergibt sich die in Abbildung 9 dargestellte Situation:



$f^+(\theta_3)$	$\widetilde{f}^+(\widetilde{\theta}_3)$	$f^+(\theta_3) - \widetilde{f}^+(\widetilde{\theta}_3)$	
1	1.0000026816	$-2.682 \cdot 10^{-6}$	
0	0	0	e_1
0	0	0	
0	0	0	
2	2	0	e_2
0	0	0	
2	1.9999973184	$2.682 \cdot 10^{-6}$	
0	0	0	e_3
2	2	0	

Abbildung 9: Aktive Kanten, Knotenlabels und Labeländerungen zum Zeitpunkt $\theta_3 = \frac{2}{3}$ (oben links), resultierender Fluss zum Zeitpunkt $\theta = 1$ (oben mittig), bzw. $\theta = 1.5$ (oben rechts), und Flussaufteilung mit Fehler zum Zeitpunkt θ_3, bzw. $\widetilde{\theta}_3$ (unten).

Erneut sind die Flusswerte der Güter 2 und 3 eindeutig und damit die Approximationen exakt. Für Gut 1 betrachten wir:

$$\Delta Err_{1,s}(\widetilde{\theta}_3) = \widetilde{a}_{1,v_1} + \frac{g_{(s,v_1)}(1.000002682)}{\nu_{(s,v_1)}} - \left(\widetilde{a}_{1,v_3} + \frac{g_{(s,v_3)}(3.999997318)}{\nu_{(s,v_3)}}\right) \tag{4.13}$$

$$= 3 + \frac{0.000002682}{1} - \left(0 + \frac{2.999997318}{1}\right) = 5.3632 \cdot 10^{-6}.$$

Bis zum Zeitpunkt $\theta_4 = 1 = \widetilde{\theta}_4$ ergibt sich:

$$Err(\theta_4^<) = Err_{1,s}(\theta_4^<) = \widetilde{\ell}_{1,v_3}(\theta_4^<) + \widetilde{c}_{e_3}(\theta_4^<) - \left(\widetilde{\ell}_{1,v_1}(\theta_4^<) + \widetilde{c}_{e_1}(\theta_4^<)\right) \tag{4.14}$$

$$= 5 + 2.6666660103 - (5.5714285714 + 2.0952376204)$$

$$= 1.8157 \cdot 10^{-7}.$$

Wir erhalten damit den relativen Fehler

$$Err^{rel}(\theta_4^<) = \frac{Err(\theta_4^<)}{b^-_{1,s}(\theta_4^<)} = 6.052 \cdot 10^{-8}. \tag{4.15}$$

Die Berechneten Flusswerte zeigen bereits in diesem Minimalbeispiel eine Variabilität in der Qualität der approximierten Werte. Obwohl die Toleranz $\epsilon = 10^{-5}$ gewählt wurde, sind die approximierten Flusswerte teilweise deutlich genauer: Der Fehler $Err(\widetilde{\theta}_3^<) \approx 10^{-11}$ ist besonders gering, während auch mit $Err(\widetilde{\theta}_4^<) \approx 10^{-7}$ ein brauchbares Ergebnis vorzuliegen scheint. Die Darstellungen der Flüsse in den Abbildungen 7 - 9 zeigen, dass die approximierten Flusswerte sinnvoll als dynamischer Fluss interpretiert werden können und die Approximation kann sogar noch verbessert werden, indem die Variable ϵ entsprechend angepasst wird.

Im Folgenden wird die Analyse weiter vertieft.

Betrachtung des gesamten Graphen.
Wir betrachten nun den gesamten Graphen G_1 aus Abbildung 6. Einige ausgewählte Flusswerte im Verlauf des zu G_1 berechneten IDE-Flusses und die resultierenden Fehler sind aufgetragen in Tabelle 2. Die erste Spalte enthält die exakten Zeitpunkte θ, während in den folgenden Spalten die approximierten Werte zum approximierten Zeitpunkt $\widetilde{\theta} \approx \theta$ enthalten sind. Die Flusswerte in Spalte 3, sowie die resultierenden Kostenänderungen in Spalte 4 sind in jeder Zeile mit einer Genauigkeit angegeben, sodass die Änderung des IDE-Fehlers in der letzten Spalte deutlich wird.

θ	Kante	$\widetilde{f}^+_{i,e}(\widetilde{\theta})$	$g_e(\widetilde{f}^+_e(\widetilde{\theta}))/\nu_e$	$\widetilde{a}_{i,e^2}$	$\Delta Err_{i,e^1}$
$\frac{2}{13}$	(v_7, v_6)	1.9999923719	0	4	$7.629 \cdot 10^{-6}$
	(v_7, v_9)	5.000007629	4.000007629	0	
$\frac{3}{7}$	(s, v_1)	2.00000000031	1.00000000031	3	$3.1 \cdot 10^{-10}$
	(s, v_2)	0.99999999969	0	4	
$\frac{1}{2}$	(v_7, v_6)	4.6666666666	1.3333333333	0	0
	(v_7, v_9)	2.3333333333	1.3333333333	0	
$\frac{2}{3}$	(s, v_1)	1.000002682	$2.682 \cdot 10^{-6}$	3	$5.363 \cdot 10^{-6}$
	(s, v_3)	1.999997318 2	2.999997318	0	
$\frac{10}{7}$	(v_2, v_5)	0 1	0	3	$3.056 \cdot 10^{-10}$
	(v_2, v_6)	0.99999999969 1	0.99999999969	2	
$\frac{5}{3}$	(v_7, v_6)	4.66667175	1.33333588	0	$7.629 \cdot 10^{-6}$
	(v_7, v_9)	2.33332824	1.333328255	0	
$\frac{8}{3}$	(v_8, v_9)	0 0.83332825	−0.16667175	$\frac{1}{3}$	$1.017 \cdot 10^{-5}$
	(v_8, t_1)	1 0.16667175	0.16667175	0	
$\frac{88}{21}$	(v_6, v_8)	1.249999999924 0.500000000076	0.75	0.5	0
	(v_6, v_9)	1.7499996647994 0.5000003352006	1.25	0	

Tabelle 2: Approximierter IDE-Fluss in G_1

Approximationsfehler treten erstmals für $\theta = \frac{2}{13}$ auf. Während die exakte und eindeutige IDE-Flussaufteilung in Knoten v_7 die Werte $f^+_{1,(v_7,v_6)}(\frac{2}{13}) = 2$, $f^+_{1,(v_7,v_9)}(\frac{2}{13}) = 5$ umfasst, ergibt sich, durch die Abweichungen von $\widetilde{f}$, der eine Änderung des IDE-Fehlers von $\Delta Err_{1,v_7} = 7.629 \cdot 10^{-6}$.

Die nächste Zeile der Tabelle zeigt einen bereits betrachteten, dennoch interessanten Fall: Die approximierten Flusswerte liegen sogar deutlich näher an den exakten Flusswerten $f^+_{1,(s,v_1)}(\frac{3}{7}) = 2$, $f^+_{1,(s,v_2)}(\frac{3}{7}) = 1$, als die verwendete Toleranz $\epsilon = 10^{-6}$ vermuten ließe. Die resultierende Fehleränderungsrate $\Delta Err_{1,s}$ liegt damit in einer Größenordnung von 10^{-10}, also ebenfalls deutlich unter ϵ. Diese erhöhte Genauigkeit ist das Resultat eines günstigen Spezialfalls innerhalb des Algorithmus: Alle Flusswerte außerhalb von s sind bereits fixiert und die Schranken für e_1, e_2 betragen etwa $(lb^k_{1,e_1}, ub^k_{1,e_1}) \approx (2, 2.437)$ und $(lb_{1,e_2}, ub_{1,e_2}) \approx (0.562, 2.312)$.

Die Mittelpunkte dieser Intervalle liegen damit bei etwa 2.218 und 1.437, welche korrigiert werden zu den Flusswerten $x^k_{1,e_1} = 2.087$, $x^k_{1,e_2} = 0.912$. Im weiteren Verlauf ergeben sich die Schranken $(lb^k_{1,e_1}, ub^k_{1,e_1}) \xrightarrow{k} (2, 2)$ und $(lb^k_{1,e_2}, ub^k_{1,e_2}) \xrightarrow{k} (1, 2.312)$. Vorerst wird demnach nur für eine der beiden Kanten eine hohe Genauigkeit der

Schranken erreicht. Da hier jedoch genau zwei aktive Kanten existieren, sind die Genauigkeiten beider Flusswerte aufgrund von Flusserhaltung miteinander korreliert, weshalb sich ebenso genaue Flusswerte für Kante e_2 ergeben, sodass die Bedingungen für Algorithmus 7: `fix_nodes()` erfüllt sind und somit die Werte approximiert werden, bevor auch die Schranke ub^k_{1,e_2} weiter verbessert werden muss.

Ein weiterer interessanter Fall tritt auf bei $\theta = \frac{10}{7}$. Hier entsteht eine positive Änderung des IDE-Fehlers für Gut 2, obwohl die Flussaufteilung für dieses Gut mit $\widetilde{f}^+_{2,e_{(v_2,v_5)}}(\widetilde{\theta}) = 1 = \widetilde{f}^+_{2,e_{(v_2,v_6)}}(\widetilde{\theta})$ exakt berechnet wird. Der Fehler ist zurückzuführen auf den eben besprochenen Fehler von Gut 1 zum Zeitpunkt $\theta = \frac{3}{7}$. Dieser verfälscht die in v_2 ankommende Menge von Gut 1 und somit auch die eindeutige Flussaufteilung auf die einzige aktive Kante (v_2, v_6) für Gut 1 und $\theta = \frac{10}{7}$. Das Resultat ist ein Fehler nicht für Gut 1, sondern Gut 2.

In den nächsten beiden Zeilen ist ein absoluter Anstieg der Fehler zu beobachten: Die Unterschiede der approximierten Flusswerte zu den exakten beginnen bereits in der fünften Nachkommastelle. Diese Differenzen sind zwar immer noch kleiner als ϵ, dennoch ergibt sich mit $\Delta Err_{3,v_8}(\frac{8}{3}) = 1.017 \cdot 10^{-5} > \epsilon$ erstmals eine Änderungsrate des IDE-Fehlers, die ϵ übersteigt.

Dass dieser Anstieg nicht überall gleichermaßen erfolgt, zeigt die letzte Zeile: Hier ist die Approximation der Flusswerte von Gut 1 so genau, dass die Gesamtflusswerte sogar exakt korrekt sind und die Änderung des IDE-Fehlers an dieser Stelle 0 beträgt.

Neben Tabelle 2 liefert Abbildung 10 einen weiteren Überblick über die Fehler bei der Approximation des IDE-Flusses in obigem Flussnetzwerk. Die erste Grafik zeigt die globalen Maxima (nach Knoten) aller IDE-Fehler für jedes Gut innerhalb der ersten 5.5 Zeiteinheiten. Dies ist der gesamte Zeitraum, in dem IDE-Fehler auftreten, auch wenn der Terminationszeitpunkt des IDE-Flusses erst bei etwa $\theta \approx 13.769$ liegt. In der zweiten Grafik ist der resultierende absolute, bzw. relative, Gesamtfehler angegeben.

Der letzte Teil der Abbildung beschreibt den global maximalen, bzw. global minimalen, Fehler im Knotenlabel für den selben Zeitraum. Genauer gesagt gilt beispielsweise: Die dunkelgrüne Linie (Gut 3^+) beschreibt zu jedem Zeitpunkt θ den Wert $\max_{v \in V} \widetilde{\ell}_{3,v}(\theta) - \ell^*_{3,v}(\theta)$, während für die hellgrüne Linie (Gut 3^-) im vorigen Ausdruck der Teil $\max_{v \in V}$ durch $\min_{v \in V}$ ersetzt wird. Gut 2 ist in diesem Teil der Abbildung nicht aufgeführt, da für dieses Gut die Fehler im Knotenlabel betraglich durch den Wert $1.6 \cdot 10^{-9}$ beschränkt werden können und daher keinen positiven Beitrag zur Aussagekraft der Abbildung liefern.

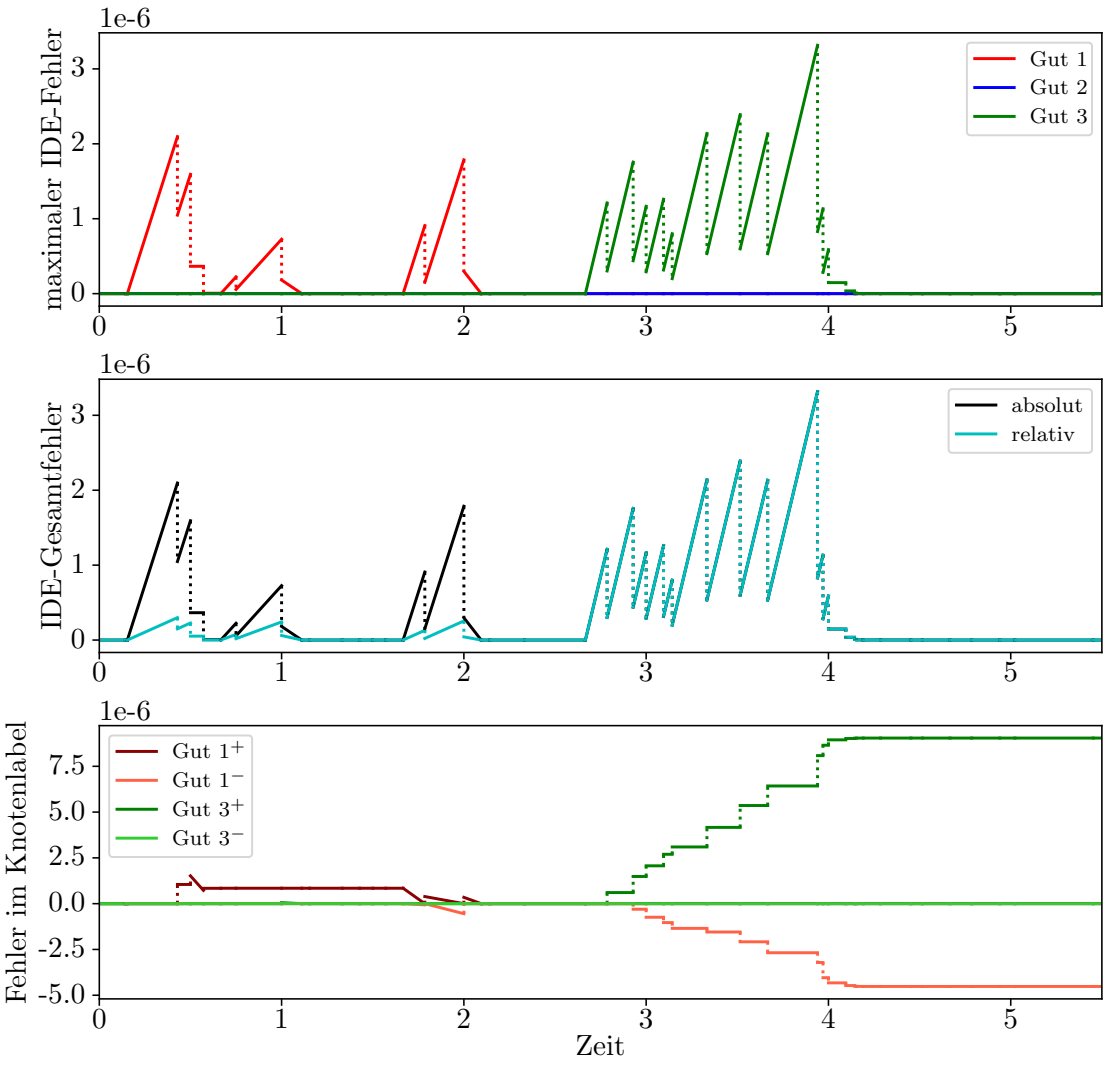


Abbildung 10: Maximaler IDE-Fehler pro Gut, sowie resultierender Gesamtfehler und Fehler im Knotenlabel pro Gut, für $\epsilon = 10^{-5}$

Es ist zu beobachten, dass die Anstiege der IDE-Fehler innerhalb einzelner Phasen sehr steil sind. Durch die Sprungstellen, welche aus Algorithmus 8: `refine()` resultieren, gelingt es jedoch, diese unter Kontrolle zu halten. Dies offenbart ein Problem bei der Approximation von IDE-Flüssen mittels Algorithmus 1: Für ein Beispiel, bei dem einzelne Phasen länger andauern, bleibt auch der potentiell steile Anstieg des IDE-Fehlers über einen längeren Zeitraum bestehen, ohne dass dem mit zwischenzeitlichen Aufrufen von `refine()` entgegengewirkt wird. Es ist jedoch auch der Fall, dass in den meisten Beispielen derartig lange Phasen nicht auftreten, da bereits in diesem einfachen Beispiel der Fluss, der sich über weniger als 14 Zeiteinheiten erstreckt, in insgesamt 104 Phasen aufgeteilt wird. Werden größere Beispiele betrachtet, so ist in den meisten Fällen ein noch deutlich kleineres Verhältnis von Zeithorizont T zur Anzahl Phasen zu erwarten.

Gleichzeitig ist jedoch auch der Nachteil der wiederholten Aufrufe von `refine()` zu beobachten: Viele der Aufrufe resultieren ebenfalls in einem Anstieg des Fehlers

im Knotenlabel, wie die untere Grafik aus Abbildung 10 verdeutlicht. Der Fehler im Knotenlabel für Gut 3 gegen Ende des betrachteten Zeitraums ist positiv, was impliziert, dass für dieses Gut die approximierten Knotenlabels die exakten Distanzwerte übersteigen. Gut 1 zeigt einen gegenläufigen Trend, bei dem die approximierten Knotenlabels niedriger sind als die tatsächlichen Distanzen. Ein solches Resultat war auch zu erwarten, da die Approximation der Knotenlabels durch Algorithmus 8: `refine()` immer dann stattfindet, wenn aus Sicht der Labels verschiedener Güter unterschiedliche Warteschlangenlängen für einzelne approximiert aktive Kanten gefordert werden.

Da `refine()` die Warteschlange auf den Durchschnitt der geforderten Warteschlangenlängen setzt, ist das Resultat im Fall von zwei Gütern größer als der geforderte Wert des einen Guts und kleiner als der des anderen.

Verhalten der Approximationsfehler im Lauf der Zeit.
Wir haben gesehen, dass mit zunehmender Länge der Phasen natürlicherweise auch der IDE-Fehler zunehmen kann. Auch bei gleichbleibender Länge der Phasen ist die Entwicklung des Fehlers interressant. Je mehr Zeit vergeht und je mehr Entscheidungen über Flussaufteilungen getroffen werden, desto mehr kann die Abweichung von dem zu approximierenden exakten IDE-Fluss anschwellen. Dies wird nun anhand einer größeren Instanz getestet. Dazu eignet sich das in [4] entwickelte Beispiel eines Netzwerks, bei dem sich der eindeutige IDE-Fluss in Kreisen verfängt und somit niemals terminiert. Dies ist für die folgende Untersuchung vor allem deswegen nützlich, da der exakte IDE-Fluss bekannt ist und mit der hier berechneten Approximation verglichen werden kann. Weiter können die periodisch auftretenden Flusswerte über einen beliebig großen Zeitraum gegeneinander abgewägt werden.

Der betrachtete Graph besteht aus einer Menge von 1922 Knoten und 3270 Kanten. Die gesamte Einflussmenge teilt sich gleichermaßen auf auf zwei Güter und umfasst pro Gut ein Volumen von 270 Flusseinheiten, ausgehend von je 135 verschiedenen Quellknoten über den Zeitraum $[0, 5]$. Jeder dieser Quellknoten ist Teil eines der A - *Gadgets* nach dem Muster aus dem oberen linken Teil von Abbildung 11. Insgesamt werden mehrere zeitversetzte Versionen dieser Gadgets verwendet, welche im Folgenden mit A^{+k} bezeichnet werden, wobei $k \in \{0, 1, 2, 3, 4\}$ das Ausmaß der Zeitverschiebung bezeichnet. Für eine genauere Beschreibung über den exakten Aufbau des Netzwerks siehe [4]. Der Grund für die Nichtterminierung des IDE-Flusses ist, dass sich der Fluss innerhalb der insgesamt 135 A - Gadgets verfängt, da nach Konstruktion immer wenn der Fluss einen der vier Knoten, über welche der jeweilige Kreis verlassen werden kann, erreicht, der entsprechende Pfad für das jeweilige Gut inaktiv ist.

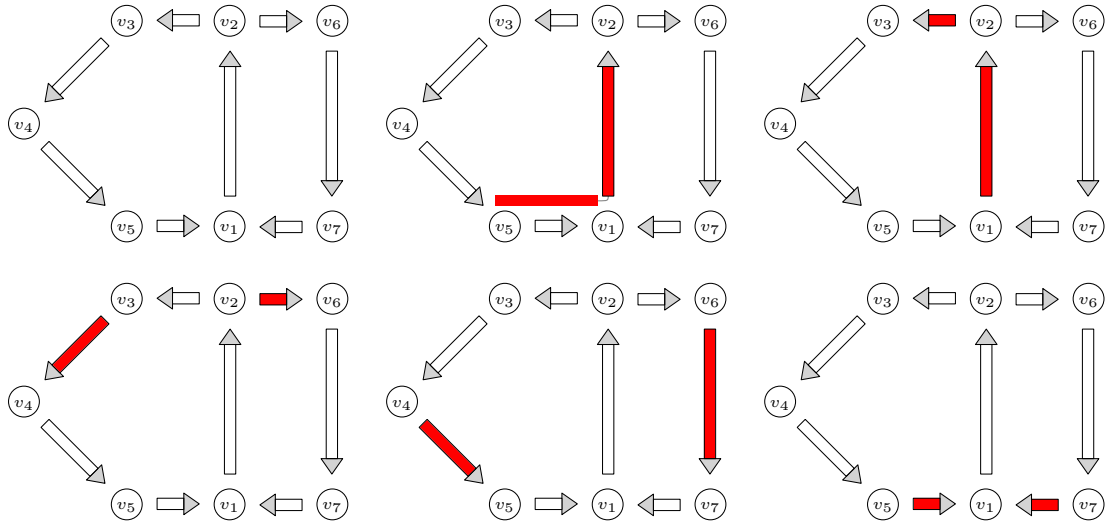


Abbildung 11: Ein Zyklus eines A-Gadgets über einen Zeitraum der Länge 5. Dargestellt sind die Zeitpunkte $\theta \in \{0,\ 5h{+}1,\ 5h{+}2,\ 5h{+}3,\ 5h{+}4,\ 5h{+}5\}$, für ein $h \in \mathbb{N}_0$.

Für die Analyse des Fehlers im Lauf der Zeit betrachten wir nun einen Einzelnen dieser Teilkreise. Für jeden dieser Kreise existieren vier Kanten, die den Kreis verlassen. In dem in Abbildung 11 dargestellten Kreis starten zwei dieser Kanten im Knoten v_2, eine führt zum v_1-Knoten des nächsten A-Gadgets, während die andere einen Pfad in die zweite Graphhälfte einleitet. Ein solcher Pfad wird im Folgenden auch als P_2^k bezeichnet, wobei $k \in \{0, 1, 2, 3, 4\}$ die Zeitverschiebung entsprechend der des betrachteten A-Gadgets bezeichnet. Weiterhin gibt es jeweils einen Pfad nach dem selben Muster ausgehend von den Knoten v_5 und v_7, diese Pfade werden analog auch als P_5^k und P_7^k bezeichnet. In obiger Abbildung sind diese Pfade zur besseren Übersichtlichkeit nicht enthalten und werden an dieser Stelle auch nicht benötigt, da der IDE-Fluss diese niemals verwendet.

Insgesamt zeigt Abbildung 11 fünf Phasen der Länge 1: In der ersten Phase gibt es einen Einfluss von 2 Flusseinheiten in Knoten v_1, welche entlang der einzigen (aktiven) Kante (v_1, v_2) geschickt werden, sodass die Warteschlange mit Rate 1 steigt und eine Zeiteinheit später das zweite Bild erreicht ist. In der zweiten Phase wird der in v_2 ankommende Fluss weitergeschickt über Kante (v_2, v_3), da diese den momentan günstigeren Weg einleitet als Kante (v_2, v_6), sowie die anderen beiden, von v_2 ausgehenden und den Kreis verlassenden Kanten. Damit ergibt sich zu $\theta = 2$ das dritte Bild und die dritte Phase beginnt: Zu diesem Zeitpunkt tauschen (v_2, v_3) und (v_2, v_6) ihren Aktivitätsstatus, sodass der in v_2 ankommende Fluss in dieser Phase den Weg über (v_2, v_6) wählt. Der in v_3 ankommende Fluss hat nur eine Option, sodass bei $\theta = 3$ das vierte Bild erreicht ist. Hier erreicht der Fluss die Knoten v_4 sowie v_6 und hat in beiden Fällen nur jeweils eine Kante zur Verfügung. Im letzten Teil der Abbildung fließt innerhalb des Intervalls $[4, 5)$ sämtlicher Fluss zurück zum Knoten v_1, da die den Kreis verlassenden Wege von v_5 und von v_7 aus inaktiv sind. Damit ergibt sich ein Einfluss von 2 Einheiten in v_1 und der erste Zyklus ist beendet, es vergehen also 5 Zeiteinheiten, bevor sich der Fluss wiederholt.

Die einzigen vorkommenden Flusswerte sind also 0, 1 und 2 und zu keinem Zeit-

punkt wird Fluss auf mehrere von einem Knoten ausgehende, aktive Kanten aufgeteilt. Diese Eigenschaften sind für die Anwendung von Algorithmus 1 sehr wohlwollend, sodass alle auftretenden Flusswerte exakt berechnet werden können und somit alle Änderungsraten der IDE-Fehler stets 0 sind. Eine weitere günstige Eigenschaft ist, dass in allen A-Gadgets mit der selben Zeitverschiebung auch alle Phasenübergänge zum selben Zeitpunkt stattfinden. Dadurch ergeben sich selbst bei Approximation der Schrittweite für alle diese Gadgets die exakt gleichen Abweichungen, sodass auch alle Fehler in den Knotendistanzen, Kantenkosten und Warteschlangen, sowie alle IDE-Fehler konstant bei 0 liegen.

In früheren Versionen des Programms war dies nicht der Fall: Mit einer alternativen Methode der Approximation der Schrittweite, bei der als Schrittweite der Wert α_{-1} aus Abschnitt 3.3 verwendet wurde, kam es auch in diesem Beispiel zu Approximationsfehlern. Die nun verwendete Schrittweite hat sich allgemein als präzisere Wahl erwiesen und scheint der früheren Version in keinen Belangen nachzustehen.

Dieses Beispiel soll nun weiter verwendet werden, um Einsichten in die Laufzeit des Algorithmus zu erlangen. Wir wollen dazu den obigen IDE-Fluss für die ersten 50 Zyklen, also für das Zeitintervall $[0, 250)$, approximieren. Der Zeitraum wird unterteilt in 1250 Phasen, der Algorithmus führt demnach ebenso viele Iterationen aus. Im exakten Fall wird dabei ein einzelner Zeitschritt $[\theta_k, \theta_k + 1)$, $\theta_k \in \mathbb{N}$, gegliedert in nachstehende fünf Phasen. Die Formulierung ist aus Sicht von Gut 1 und funktioniert analog für Gut 2.

1. $[\theta_k, \theta_k + \frac{1}{8})$: Es werden alle 27 $P_5^{+(\theta_k \mod 5)}$ - Pfade von der ersten in die zweite Graphhälfte aktiv, die Flusswerte bleiben unverändert.

2. $[\theta_k + \frac{1}{8}, \theta_k + \frac{1}{6})$: Alle $P_7^{+(\theta_k \mod 5)}$ - Pfade werden aktiv, die Flusswerte bleiben unverändert.

3. $[\theta_k + \frac{1}{6}, \theta_k + \frac{2}{7})$: Alle $P_2^{+(\theta_k+2) \mod 5}$ - Pfade werden aktiv, die Flusswerte bleiben unverändert.

4. $[\theta_k + \frac{2}{7}, \theta_k + \frac{6}{7})$: Die Kanten (v_2, v_3) der $A^{+(\theta_k \mod 5)}$ - Gadgets, sowie die Kanten (v_5, v_1) und (v_7, v_1) der $A^{+(\theta_k+2) \mod 5}$ - Gadgets werden aktiv, die Flusswerte bleiben unverändert. Insbesondere werden unmittelbar nach $\theta_k + \frac{6}{7}$ die Pfade $P_2^{+(\theta_k \mod 5)}$, $P_5^{+(\theta_k+2) \mod 5}$ und $P_7^{+(\theta_k+2) \mod 5}$ inaktiv.

5. $[\theta_k + \frac{6}{7}, \theta_k + 1)$: Es werden die Kanten (v_2, v_6) der insgesamt 54 $A^{+(\theta_k-1) \mod 5}$ - und $A^{+(\theta_k \mod 5)}$ - Gadgets der ersten Graphhälfte aktiv, sowie die sechs Verbindungskanten (v_2, v_1) zwischen $A^{+(\theta_k \mod 5)}$ - und $A^{+(\theta_k+1) \mod 5}$ - Gadgets. Außerdem werden die Pfade $P_5^{+(\theta_k-1) \mod 5}$ und $P_7^{+(\theta_k-1) \mod 5}$ des ersten $A^{+(\theta_k-1) \mod 5}$ - Gadgets innerhalb des Gadgets $B_2^{+(\theta_k-1) \mod 5}$ in der zweiten Graphhälfte aktiv. Gleichzeitig erreichen die Warteschlangen aller $A^{+(\theta_k-1) \mod 5}$ - Gadgets den Wert 0 und die f^- - Werte ändern sich gemäß folgender Tabelle:

Kante	innerhalb des Gadgets	Änderung f^-
(v_1, v_2)	$A^{+(\theta_k \mod 5)}$	$0 \to 1$
(v_2, v_3)	$A^{+(\theta_k-1) \mod 5}$	
(v_2, v_6)	$A^{+(\theta_k-2) \mod 5}$	
(v_3, v_4)	$A^{+(\theta_k-2) \mod 5}$	
(v_4, v_5)	$A^{+(\theta_k-3) \mod 5}$	
(v_6, v_7)	$A^{+(\theta_k-3) \mod 5}$	
(v_5, v_1)	$A^{+(\theta_k-4) \mod 5}$	
(v_7, v_1)	$A^{+(\theta_k-4) \mod 5}$	
(v_1, v_2)	$A^{+(\theta_k-2) \mod 5}$	$1 \to 0$
(v_2, v_3)	$A^{+(\theta_k-2) \mod 5}$	
(v_2, v_6)	$A^{+(\theta_k-3) \mod 5}$	
(v_3, v_4)	$A^{+(\theta_k-3) \mod 5}$	
(v_4, v_5)	$A^{+(\theta_k-4) \mod 5}$	
(v_6, v_7)	$A^{+(\theta_k-4) \mod 5}$	
(v_5, v_1)	$A^{+(\theta_k-5) \mod 5}$	
(v_7, v_1)	$A^{+(\theta_k-5) \mod 5}$	

Tabelle 3: Änderungen der f^--Werte (für beide Güter) zum Zeitpunkt $\theta_k + 1$ für $\theta_k \in \mathbb{N}_{>4}$. Gilt auch für $\theta_k \in \{0, 1, 2, 3, 4\}$, wenn alle Zeilen mit vor Durchführung der *mod* - Operation negativen Zeitwerten weggelassen werden.

Da sich bei den Übergängen zwischen den einzelnen Phasen innerhalb eines solchen Quintetts keine Flusswerte ändern, müssen diese auch nicht immer berechnet werden, weshalb Algorithmus 1 an vier von fünf Interpolationspunkten innerhalb des Intervalls $[\theta_k, \theta_k + 1)$ die Flussberechnungen überspringt. Lediglich zu Beginn von 1, also zum Zeitpunkt θ_k, erfolgt eine solche Berechnung. Die einzige Ausnahme bildet das Intervall $[0, 1)$, welches aus nur vier Teilphasen zusammengesetzt ist:

1. $[0, \frac{1}{2})$: Bei $\theta = \frac{1}{2}$ werden die Kanten (v_2, v_6) aus jedem der 27 A^{+3} - Gadgets aus der ersten (bzw. zweiten) Graphhälfte aktiv für Gut 1 (bzw. Gut 2), alle Flusswerte bleiben gleich.

2. $[\frac{1}{2}, \frac{2}{3})$: Die Phase endet aufgrund des Aktivwerdens der Kanten (v_5, v_1), und (v_7, v_1) aller insgesamt 54 A^{+1} - und A^{+2} - Gadgets aus der ersten (bzw. zweiten) Graphhälfte für Gut 1 (bzw. Gut 2), sowie der Kante (v_2, v_6) aller A^{+4} - Gadgets. Die Flusswerte bleiben gleich.

3. $[\frac{2}{3}, \frac{3}{4})$: Die Kante (v_2, v_3) aller A^{+0}- und A^{+4}-Gadgets, bei gleichbleibenden Flusswerten.

4. $[\frac{3}{4}, 1)$: Die Ausflussrate f^- von Gut 1 (bzw. Gut 2) der Kante (v_1, v_2) aller A^{+0}- Gadgets der linken (bzw. rechten) Graphhälfte ändert sich von 0 zu 1. Außerdem werden folgende Kanten aktiv: Die Kanten (v_2, v_6) der A^{+0} - Gadgets, (v_2, v_3) aus den A^{+4} - Gadgets, die insgesamt sechs Verbindungskanten der

v_2 - Knoten aus A^{+0} - Gadgets mit den Knoten v_1 aus benachbarten A^{+1} - Gadgets, sowie die Kanten der jeweils gegenüberliegenden Graphhälfte zu den Hilfsknoten

Auch in diesem Spezialfall können drei von vier potentiellen Berechnungsschritten übersprungen werden. Insgesamt werden also 999 von 1250 möglichen Berechnungen übersprungen, womit sich erneut die Wichtigkeit von Zeile **6** aus Algorithmus 1 erweist.

Die Berechnungen dieser Flüsse wurden durchgeführt auf einem System mit einem 64-Bit Intel(R) Core(TM) i5-7500, 3.40GHz Prozessor und 16 GB RAM. Der Quellcode wurde in Python Version 3.6 geschrieben. Die damit gemessene Laufzeit beträgt für die Genauigkeit $\epsilon = 10^{-3}$ rund 2717 Sekunden, was etwa 45 Minuten entspricht, und für $\epsilon = 10^{-5}$ rund 3312 Sekunden, also etwa 55 Minuten. Die angegebenen Laufzeiten beinhalten nicht die Berechnung der initialen Knotendistanzlabels mit dem Algorithmus von Dijkstra. Auch wenn diese Laufzeiten bereits deutliche Verbesserungen gegenüber früheren Programmversionen zeigen, ist der Code noch nicht vollständig auf Effizienz optimiert und kann daher wahrscheinlich noch weiter beschleunigt werden.

Der oben beobachtete Unterschied der Laufzeiten bei der Approximation des gleichen Flusses mit zwei verschiedenen Genauigkeiten offenbart die Wichtigkeit einer sinnvollen Wahl für den Parameter ϵ. Besonders in diesem Beispiel wird klar: Die Berechnung mit $\epsilon = 10^{-3}$ reicht vollkommen aus, um eine genaue Approximation zu erhalten. Die Verwendung einer präziseren Genauigkeit resultiert hier lediglich in einer erhöhten Laufzeit. Dies ist jedoch erneut dem glücklichen Umstand zuzuschreiben, dass alle Flusswerte hier exakt berechnet werden können, weshalb in den meisten anderen Beispielen ein kleinerer Wert für ϵ sinnvoll ist.

Die Abbildung auf der folgenden Seite zeigt eine Hälfte des Graphen mit dem berechneten approximierten IDE-Fluss zum Zeitpunkt $\theta = 9$.

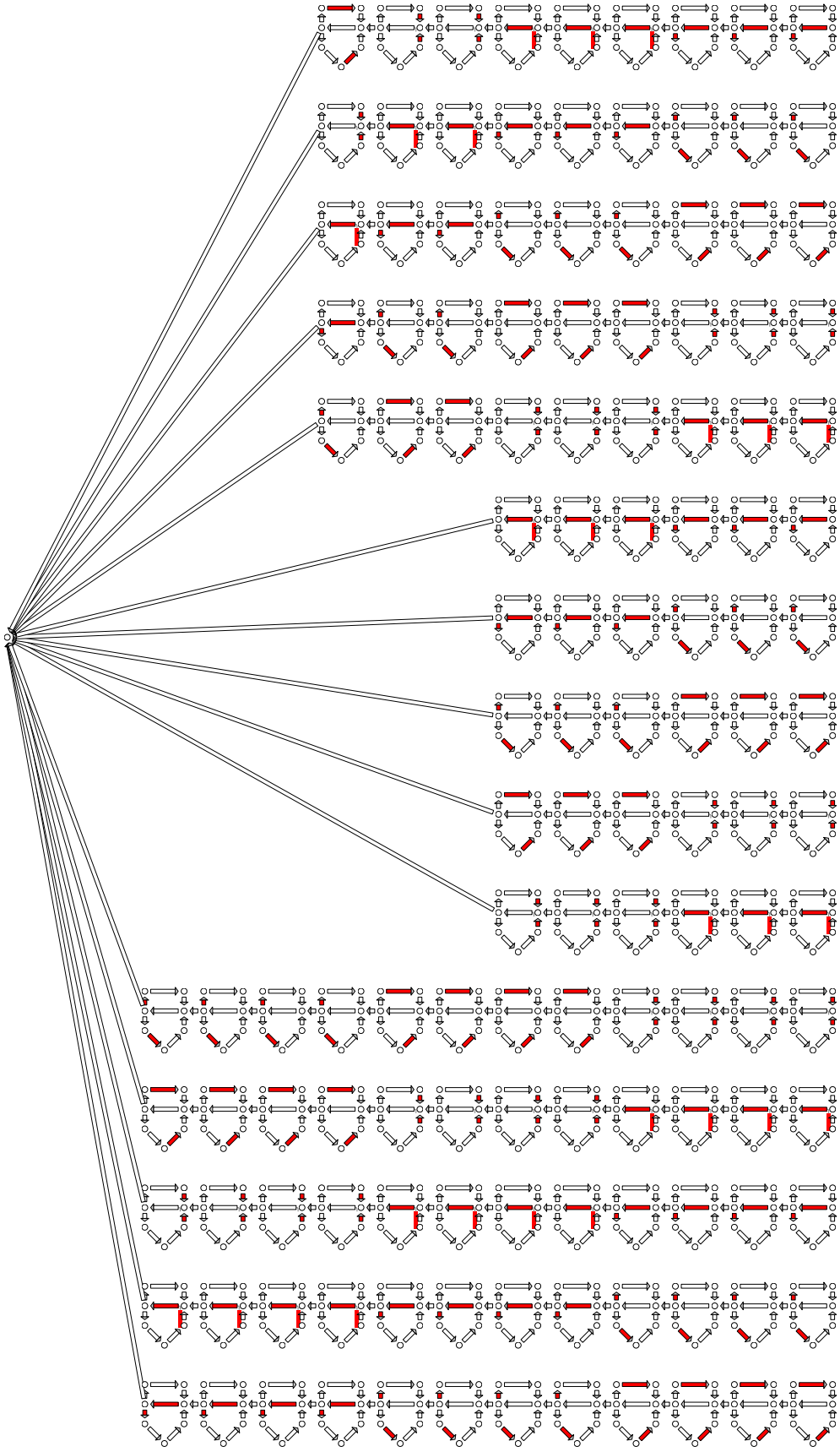


Abbildung 12: Teil des Flussnetzwerks aus [4], zum Zeitpunkt $\theta = 9$.

Test der Rechenkapazitäten.
Es soll nun ein IDE-Fluss in einem noch umfangreicheren Netzwerk bestimmt werden. Das im letzten Abschnitt bearbeitete Beispiel ist zwar von der Knoten- und Kantenzahl bereits recht groß, jedoch ist die Struktur des Graphen und des berechneten IDE-Flusses für die Rechnung relativ wohlwollend. Wir betrachten daher nun eine realistische Modellierung des Marktes Holzkirchen, entnommen von https://github.com/matsim-org/matsim-libs/tree/master/examples/scenarios/holzkirchen. Der darin enthaltene Beispielgraph umfasst eine Menge von 3052 Knoten und 7004 Kanten und ist dargestellt in Abbildung 13. Zur besseren Anwendbarkeit und Übersichtlichkeit werden die Kantenkapazitäten und Reiselängen folgendermaßen angepasst: Die Reiselänge jeder Kante wird durch 100 geteilt und auf 3 Nachkommastellen gerundet. Die Kantenkapazitäten werden eingeteilt in vier Gruppen:

- Kanten mit Kapazität $\nu_e \in [0, 1000]$ in der ursprünglichen Instanz erhalten Kapazität 1,
- Kanten mit Kapazität $\nu_e \in (1000, 2500]$ in der ursprünglichen Instanz erhalten Kapazität 2,
- Kanten mit Kapazität $\nu_e = 6000$ in der ursprünglichen Instanz erhalten Kapazität 3,
- alle anderen Kanten erhalten Kapazität 4.

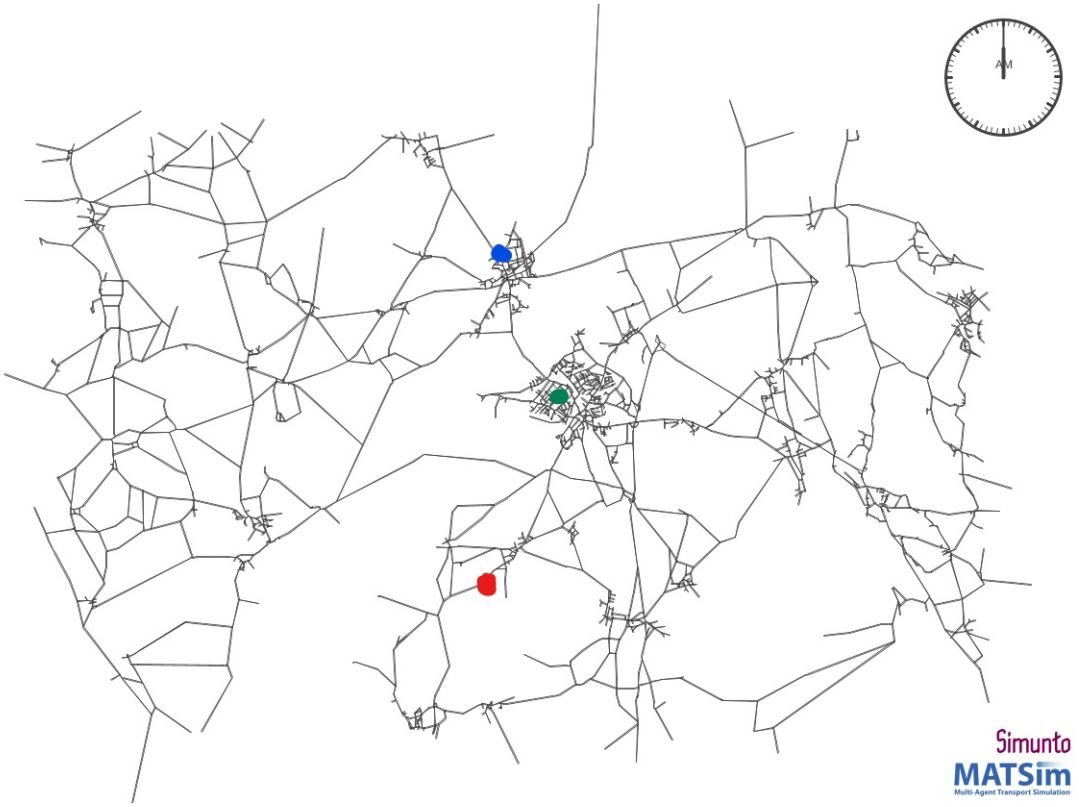


Abbildung 13: Graphnetzwerk Holzkirchen, mit $n = 3052$ und $m = 7004$.
Graphik erstellt mit *Via*: https://simunto.com/via/

Mit diesen Setzungen soll ein erster einfacher IDE-Fluss im Graphnetzwerk Holzkrichen approximiert werden. Beginnend im Zentrum der Stadt (grüne Markierung in Abbildung 13), im Knoten mit Index 2432 starten im Zeitraum $[0, 2)$ zwei Güter mit einer Rate von 15 für Gut 1 und einer Rate von 14 für Gut 2. Der Zielknoten von Gut 1 hat Index 2169 und liegt am südlichen Rand der Stadt unterhalb des Zentrums (rote Markierung). Gut 2 dagegen bewegt sich zum Knoten mit Index 1928 im Norden (blaue Markierung).

Die entstehenden Fehler beider Güter, sowie der absolute und der relative Gesamtfehler sind dargestellt in Abbildung 14.

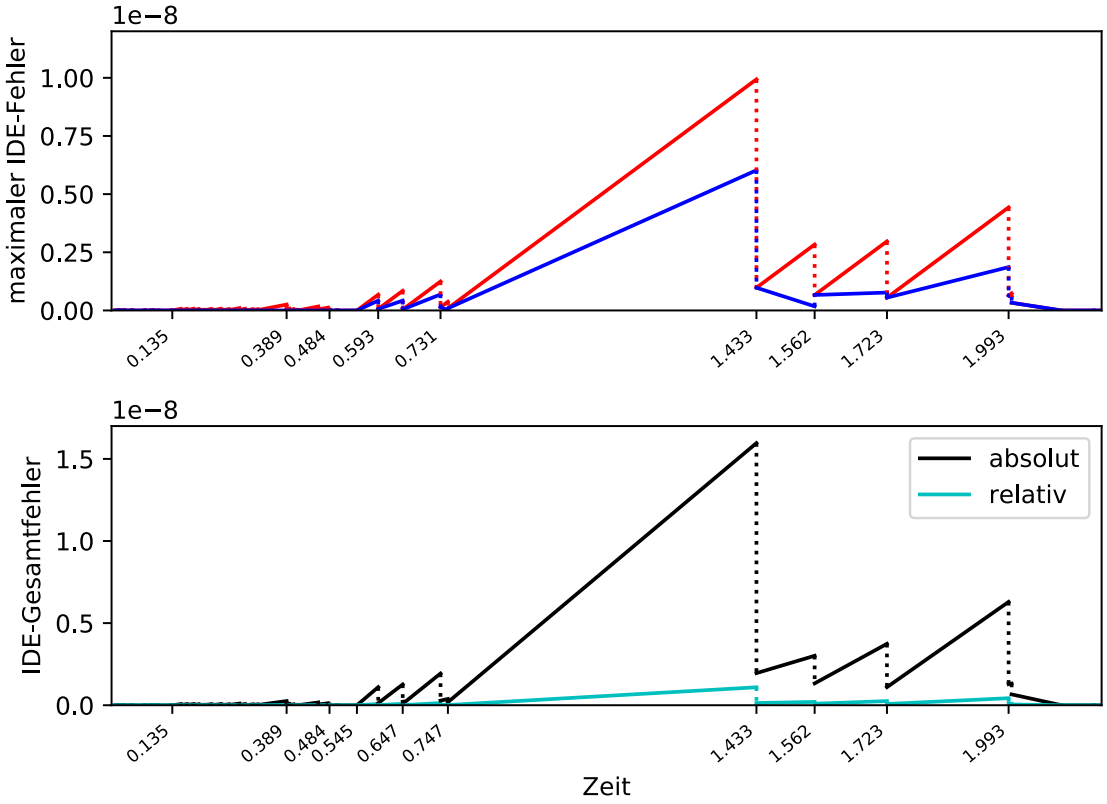


Abbildung 14: IDE-Fehler nach Gut, sowie resultierender Gesamtfehler, für Graphnetzwerk Holzkirchen, mit $\epsilon = 10^{-8}$

Der Fluss terminiert etwa zum Zeitpunkt $\theta = 134.466$. Bis dahin teilt sich der Fluss auf in 824 Phasen, von denen sich in 241 Phasen kein Flusswert ändert, sodass die Berechnungen in diesen Phasen von Algorithmus 1 übersprungen werden. Die Laufzeit für dieses Beispiel liegt mit 3586 Sekunden bei knapp unter einer Stunde. Aufgrund der Einfachheit des berechneten Flusses treten Fehler nur innerhalb der ersten 2.2 Zeiteinheiten auf. Danach ist der Fluss eindeutig und die Approximation exakt.

Die auftretenden Fehler sind dabei zurückzuführen auf die im Startknoten v_{2432} gewählte Flussaufteilung. Der dafür relevante Graphausschnitt ist dargestellt in Abbildung 15:

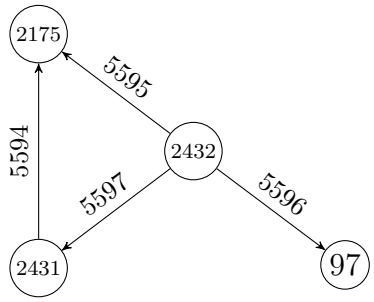


Abbildung 15: Knoten direkt erreichbar vom Startknoten mit Index 2432. Angegeben sind sowohl Knoten- als auch Kantenindizes.

Die Flusswerte, die zu dem in Abbildung 14 dargestellten Fehler führen sind aufgetragen in nachstehender Abbildung:

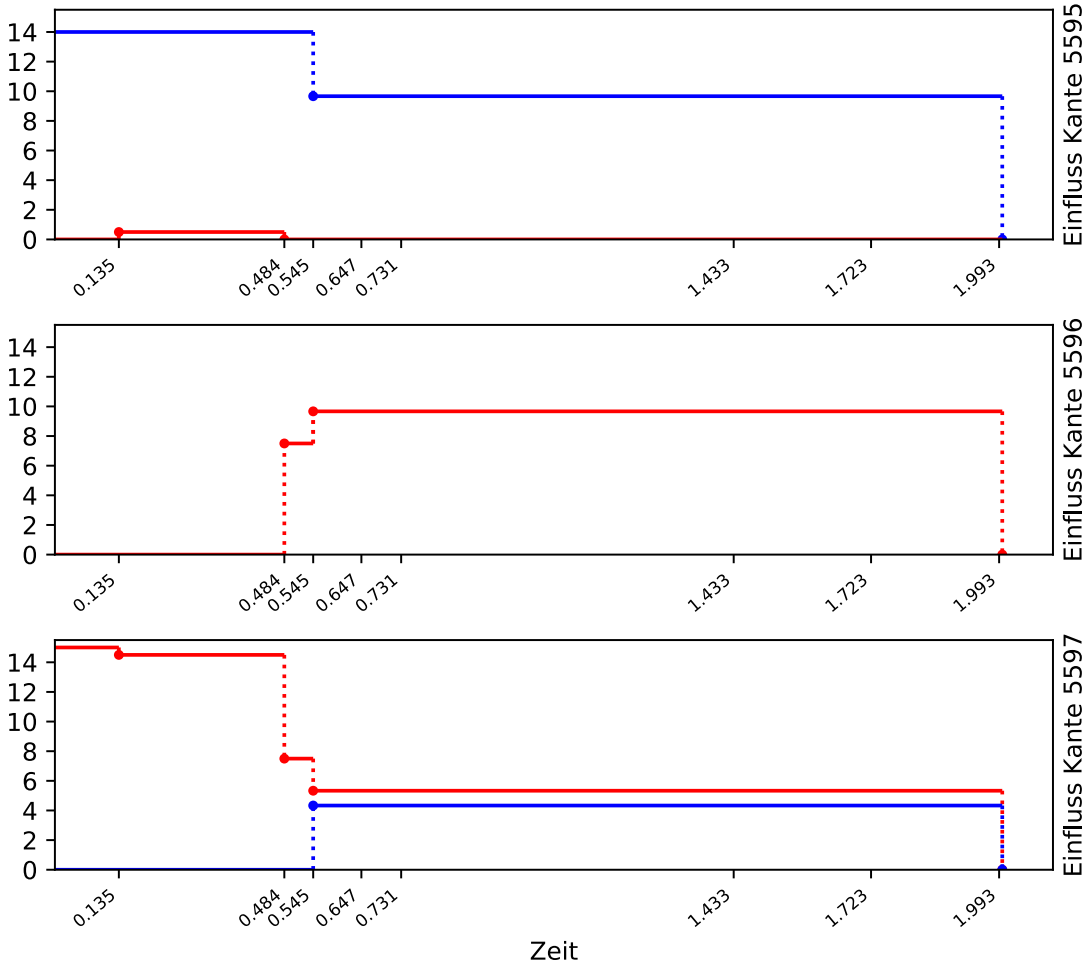


Abbildung 16: Flusswerte dreier ausgewählter Kanten aus dem Graphnetzwerk Holzkirchen, mit $\epsilon = 10^{-8}$. Gut 1: rot, Gut 2: blau.

Es ist zu beobachten: Beginnend beim Zeitpunkt $\theta \approx 0.54526$ bis zum Ende des externen Zuflusses bei $\theta = 2$ bleibt die Flussaufteilung im Knoten v_{2432} konstant.

Die Änderungsraten der Fehler während der Phasen dieses Zeitraums bleiben daher gleichmäßig bei $Err_{1,2432} \approx 7.239 \cdot 10^{-7}$, bzw. $Err_{2,2432} \approx 3.391 \cdot 10^{-8}$. Die Anpassung der Labels durch `refine()` am Ende jeder Phase ist verantwortlich für die auftretenden Sprungstellen. Die Längste auftretende Phase $[0.747, 1.433)$ liefert dann den größten auftretenden Fehler von $Err_{1,2432}(1.433^{<}) \approx 9.4712 \cdot 10^{-9}$ für Gut 1, $Err_{2,2432}(1.433^{<}) \approx 2.0219 \cdot 10^{-9}$ für Gut 2 und somit $Err(1.433^{<}) \approx 1.1493 \cdot 10^{-8}$ insgesamt. Durch Setzen dieser Fehler in Relation zur vorhandenen Flussmenge erhalten wir:

$$Err^{rel}_{1,2432}(1.433^{<}) \approx 6.3141 \cdot 10^{-10}, \quad Err^{rel}_{2,2432}(1.433^{<}) \approx 1.4442 \cdot 10^{-10} \text{ und}$$
$$Err^{rel}(1.433^{<}) \approx 7.7583 \cdot 10^{-10}. \tag{4.16}$$

Diese Fehler legen nahe, dass die berechnete Approximation für dieses Beispiel sehr genau ist. Vor allem wenn der Fehler in Relation zum vorhandenen Flussvolumen gesetzt wird (siehe auch Cyan-farbene Linie in Abbildung 14) scheinen die Abweichungen vernachlässigbar.

Nicht-Eindeutigkeit.
Zum Abschluss betrachten wir noch ein einfacheres Beispiel, bei dem eine ganze Klasse von IDE-Flüssen existiert. Besonders diese Nicht-Eindeutigkeit soll dabei im Augenmerk der Analyse liegen.

Wir betrachten Abbildung 17: Die Kapazitäten der drei von s ausgehenden Kanten sind in der Abbildung angegeben, alle anderen Kapazitäten sowie alle Reiselängen seien gleich 1.

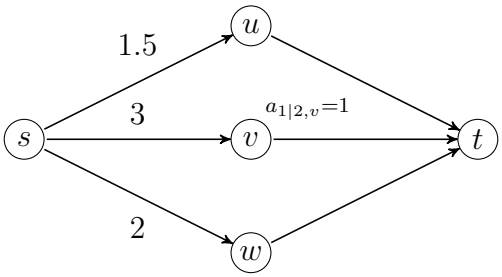


Abbildung 17: Graph G_2

Es soll Fluss zweier Güter von s nach t geschickt werden. Die Güter werden hier ausnahmsweise *nicht* mit ihrer Senke identifiziert. Dies geschieht im Sinne der besseren Übersichtlichkeit und hat hier keine Auswirkungen auf die berechneten Flusswerte. Weiter gelte $a_{1,v}|_{[0,0.5)} = 1 = a_{2,v}|_{[0,0.5)}$ und $a_{1,v}|_{[0.5,1)} = -1 = a_{2,v}|_{[0.5,1)}$. Dies ist beispielsweise zu erreichen mit einer Einflussrate von 2 von Gut 1 im Knoten v über den Zeitraum $[0, 0.5)$. Dieser Teil des Flusses ist dann eindeutig und wird hier nicht genauer betrachtet. Lediglich die Auswirkungen auf die Flussaufteilung im Knoten s sind hier von besonderem Interesse. Tabelle 4 zeigt in den ersten beiden Spalten die Einflussraten in Knoten s von Gut 1 (rot) und Gut 2 (blau) über den Zeitraum $[0, 1)$. In den Spalten 4 und 5 sind die approximierten Flusswerte beider

Güter für $\epsilon = 10^{-5}$, sowie die resultierenden Gesamtflusswerte aufgetragen. Die Werte sind dabei gerundet auf sechs Nachkommastellen. Die letzte Spalte zeigt die exakten Gesamtflusswerte der IDE-Flüsse.

Zeit	Einfluss in s	Kante	Approximation	Gesamtfluss	IDE - Gesamtfluss
$[0, 0.2)$	6.5 1	(s, u)	2.798673 0.201328	3.00000045	3
		(s, v)	0.499998 0	0.499998	0.5
		(s, w)	3.201329 0.798672	4.0000018	4
$[0.2, 0.5)$	7.25 6	(s, u)	2.026826 1.723173	3.749999	3.75
		(s, v)	2.520739 1.979263	4.500002	4.5
		(s, w)	2.702435 2.297564	4.999998	5
$[0.5, 1)$	4 $\frac{10}{3}$	(s, u)	0.545454 0.454545	0.999998	1
		(s, v)	2.727274 2.272729	5.000003	5
		(s, w)	0.727272 0.606059	1.333331	$\frac{4}{3}$

Tabelle 4: Eine Möglichkeit eines approximierten IDE-Flusses

Wie bereits angekündigt ist dies hier nicht der einzig mögliche IDE-Fluss. Tatsächlich ist jeder zulässige Fluss, welcher die in der letzten Spalte beschriebenen Gesamtflusswerte erreicht, ein gültiger IDE-Fluss. Demnach ist obige Approximation in Anbetracht der verwendeten Toleranz $\epsilon = 10^{-5}$ eine korrekte Approximation eines dieser Flüsse. Die Aufteilung der Güter scheint dabei willkürlich, ist jedoch sicher auch abhängig von der initialen Wahl (3.3) der Flussaufteilung in der ersten Iteration des Algorithmus. Ähnliche Ergebnisse sind auch für Beispiele zu beobachten, bei denen nicht einmal die Gesamtflusswerte eindeutig sind.

5 Fazit

In dieser Arbeit wurde ein Algorithmus zur Approximation dynamischer IDE-Flüsse in Netzwerken mit mehreren Senken vorgestellt. Auch wenn die Konvergenz des Algorithmus und die Korrektheit nicht sicher zugesagt werden können, haben wir in mehreren Anwendungsbeispielen gesehen, dass der Algorithmus brauchbare Ergebnisse liefert. Des Weiteren ist mir bis jetzt kein Beispiel bekannt, in dem der Algorithmus eine unerwünschte Ausgabe produziert: Ich halte es dennoch für wahrscheinlich, dass ein solches existiert.

Der Rechenaufwand zeigte sich als überschaubar, sodass auch bei der Anwendung auf größere Netzwerke akzeptable Laufzeiten gemessen werden konnten. In dieser Hinsicht kann der Programmcode aber sicher noch weiter verbessert werden. Beispielsweise scheint es mir sinnvoll, die Berechnung der topologischen Sortierungen zu den aktiven Teilgraphen aller Güter, welche in jeder Phase stattfinden, zu überarbeiten.

Für zusätzliche Informationen zur Implementierung des Algorithmus sei der Leser verwiesen auf https://github.com/johanneshage/ide-repository/blob/master/multicom%20IDEs/Softwareprojekt.pdf.

Herzlichen Dank an Lukas Graf und Michael Markl, die mir während der Anfertigung dieser Masterarbeit bei Fragen hilfsbereit zur Seite standen.

6 Literatur

[1] B Curtis Eaves. An odd theorem. *Proceedings of the American Mathematical Society*, 26(3):509–513, 1970.

[2] B. Curtis Eaves. Computing kakutani fixed points. *SIAM Journal on Applied Mathematics*, 21(2):236–244, 1971.

[3] Lukas Graf and Tobias Harks. A finite time combinatorial algorithm for instantaneous dynamic equilibrium flows. *Mathematical Programming*, 2022.

[4] Lukas Graf, Tobias Harks, and Leon Sering. Dynamic flows with adaptive route choice. *Mathematical Programming*, 183(1):309–335, 2020.

[5] Mohamed Jleli, Bessem Samet, Calogero Vetro, and Francesca Vetro. Fixed points for multivalued mappings in b-metric spaces. *Abstract and Applied Analysis*, 2015:718074, Mar 2015.

[6] Shizuo Kakutani. A generalization of Brouwer's fixed point theorem. *Duke Mathematical Journal*, 8(3):457 – 459, 1941.

[7] Santosh Kumar, Arshad Khan, Muhammad Sarwar, Farhan Khan, Habes Alsamir, and Hasanen A. Hammad. Fixed point results for multivalued mappings with applications. *Journal of Function Spaces*, 2021:9921728, 2021.

[8] Ma'aruf Shehu Minjibir and Chimezie Izuazu. Iterative algorithm for approximating fixed points of multivalued quasinonexpansive mappings in banach spaces. *Fixed Point Theory and Algorithms for Sciences and Engineering*, 2022(1):8, 2022.

[9] M.J. Todd. *The Computation of Fixed Points and Applications*. Lecture Notes in Mathematics; 513. Springer-Verlag, 1976.

[10] G. van der Laan and A. J. J. Talman. A restart algorithm for computing fixed points without an extra dimension. *Mathematical Programming*, 17(1):74–84, 1979.

[11] G. van der Laan and A. J. J. Talman. A restart algorithm without an artificial level for computing fixed points on unbounded regions. In Heinz-Otto Peitgen and Hans-Otto Walther, editors, *Functional Differential Equations and Approximation of Fixed Points*, pages 247–256, Berlin, Heidelberg, 1979. Springer Berlin Heidelberg.

[12] G. van der Laan and A. J. J. Talman. A class of simplicial restart fixed point algorithms without an extra dimension. *Mathematical Programming*, 20(1):33–48, 1981.

[13] William S. Vickrey. Congestion theory and transport investment. *The American Economic Review*, 59(2):251–260, 1969.