# Rekonstruktion von kürzesten Wegen auf Kreisen mittels des erweiterten linearen Komplementaritätsproblems

Masterthesis

vorgelegt von
**Saskia Grabinat**

3. März 2025

Erstbetreuer: Univ.-Prof. Dr. Winfried Hochstättler
Zweitbetreuerin: Dr. Olga Minevich

# Inhaltsverzeichnis

# Abkürzungsverzeichnis

**ELKP** erweitertes lineares Komplementaritätsproblem

**HELKP** homogenes erweitertes lineares Komplementaritätsproblem

**IKWP** inverses Kürzeste-Wege-Problem

**IKWPK** inverses Kürzeste-Wege-Problem auf Kreisen

**LKP** lineares Komplementaritätsproblem

# Abbildungsverzeichnis

# Algorithmenverzeichnis

# 1 Einleitung

Das inverse Kürzeste-Wege-Problem (IKWP) ist eine Variante des klassischen Kürzeste-Wege-Problems. Während beim klassischen Problem die Kantengewichte eines Graphen vorgegeben sind und darauf basierend kürzeste Wege bestimmt werden, verfolgt das IKWP den umgekehrten Ansatz. Zwischen ausgewählten Knotenpaaren sind die kürzesten Distanzen gegeben, und es soll eine Gewichtsfunktion für die Kanten gefunden werden, die sicherstellt, dass die kürzesten Wege mit den vorgegebenen Distanzen übereinstimmen.

In dieser Arbeit setzen wir uns mit einem Spezialfall des IKWP auseinander – das inverse Kürzeste-Wege-Problem auf Kreisen (IKWPK). Hierbei betrachten wir Kreisgraphen, deren Kantengewichte wir so rekonstruieren, dass die kürzesten Wege den gegebenen Distanzen entsprechen. Der Kreisgraph weist eine besondere Struktur auf, da zwischen zwei beliebigen Knoten stets genau zwei verschiedene Wege existieren – einer im Uhrzeigersinn, der andere gegen den Uhrzeigersinn.

Wir untersuchen, ob das IKWPK als lineares Komplementaritätsproblem (LKP) gelöst werden kann. Es zeigt sich jedoch, dass dies nicht der Fall ist. Daher überführen wir das IKWPK in das erweiterte lineare Komplementaritätsproblem (ELKP), welches als Verallgemeinerung des LKP gilt und uns eine alternative Modellierung des Problems ermöglicht.

Es ist anzumerken, dass die Thesis primär auf dem von De Schutter und De Moor verfassten Artikel „*The extended linear complementarity problem*“, siehe [DSDM95], basiert. In dieser Arbeit wird das ELKP formal eingeführt und ein Lösungsalgorithmus beschrieben. Im Verlauf der Thesis übertragen wir die darin aufgeführten Methoden und Algorithmen auf das IKWPK. Unser Ziel ist es, zu untersuchen, ob wir das IKWPK als ELKP lösen können. Die Effizienz des Lösungsalgorithmus überprüfen wir anhand von Simulationen mit verschiedenen Testinstanzen.

Um das Thema zu bearbeiten, ist die Thesis wie folgt aufgebaut:
Das Kapitel 2 ist das Grundlagenkapitel. In diesem führen wir graphentheoretische Begriffe ein und formulieren das IKWP sowie den Spezialfall, das IKWPK. Außerdem analysieren wir die Möglichkeit, das IKWPK in ein LKP zu transformieren. Das Kapitel 3 bildet den Hauptteil der Thesis. Wir führen zunächst das ELKP und dessen homogene Variante (HELKP) ein. Anschließend überführen wir das IKWPK in dieses Problemstellungen. Danach passen wir den Lösungsalgorithmus aus [DSDM95] auf unsere Problemstellungen an und fügen weitere Vorverarbeitungsschritte hinzu. In Kapitel 4 untersuchen wir die Effizienz des ursprünglichen Algorithmus und der angepassten Lösungsverfahren. Dazu führen wir Simulationen mit verschiedenen Testinstanzen des IKWPK durch und vergleichen im Anschluss insbesondere die Laufzeiten der Verfahren. Abschließend fassen wir die Ergebnisse unserer Arbeit im Kapitel 5 zusammen.

# 2 Mathematische Grundlagen

In diesem Kapitel entwickeln wir ein gemeinsames Verständnis der für die Thesis relevanten Begrifflichkeiten. Zunächst führen wir zentrale graphentheoretische Definitionen ein, die die Grundlage für die Formulierung des allgemeinen inversen Kürzeste-Wege-Problems (IKWP) und seines Spezialfalls, des inversen Kürzeste-Wege-Problems auf Kreisen (IKWPK), bilden. Anschließend behandeln wir das lineare Komplementaritätsproblem (LKP) und untersuchen die Möglichkeit, das IKWPK in ein LKP zu transformieren. Hierbei zeigen wir, dass dieses in seiner ursprünglichen Form nicht lösbar ist. Dies bildet den Ausgangspunkt für den Hauptteil der Thesis.

## 2.1 Graphentheoretische Definitionen

Die Definitionen 2.1 bis 2.4 basieren auf [BGW21, vgl. S. 1 ff.]. Den Definitionen 2.5 bis 2.10 liegen [Mol95, vgl. S. 6 f.] und [Sau23, vgl. S. 4 ff.] zugrunde.

**Definition 2.1.** Ein **Graph** $G(V, E)$ ist ein Tupel, welches aus einer nichtleeren endlichen **Kontenmenge** $V$ und einer endlichen **Kantenmenge** $E$ besteht. Die Kanten $e \in E$ sind ungeordnete Paare von Knoten $\{v_i, v_j\} \subset V$. Wir geben diese auch mit $e = v_i v_j$ an.

**Definition 2.2.** Die Knoten $v_i$ und $v_j$ der Kante $e = v_i v_j$ sind die **Endknoten** der Kante. Die Kante $e$ **verbindet** die Knoten $v_i$ und $v_j$. Ist $v_i$ ein Endknoten der Kante $e$, so heißt $v_i$ **inzident** mit $e$. Zwei Knoten $v_i$ und $v_j$, die durch eine Kante verbunden sind, heißen **adjazent**. Zwei Kanten $e_i$ und $e_j$ heißen **adjazent**, wenn sie einen gemeinsamen Knoten besitzen.

**Definition 2.3.** Eine Kante $e = \{v\}$ mit übereinstimmenden Endknoten ist eine **Schleife**. Haben zwei Kanten $e_i$ und $e_j$ dieselben Endknoten, dann heißen diese Kanten **parallel**. Einen Graphen bezeichnen wir als **schlicht** oder **einfach**, wenn dieser weder Schlingen noch parallele Kanten enthält.

**Definition 2.4.** Ein **Spaziergang** $W(V, E)$ ist ein Graph, dessen Knoten und Kanten eine alternierende Folge $v_1, e_1, v_2, \ldots, e_k, v_k$ bilden, in dem jede Kante $e_i$ die Knoten $v_i$ und $v_{i+1}$ verbindet. Der Spaziergang **verbindet** $v_1$ mit $v_k$, und heißt $v_1$-$v_k$-Spaziergang. Ein Spaziergang heißt **geschlossen**, wenn $v_1 = v_k$ gilt, also der erste und letzte Knoten des Spaziergangs übereinstimmen. Ein **Weg** oder **Pfad** $P(V, E)$ ist ein Spaziergang $v_1, e_1, v_2, \ldots, e_k, v_k$, in dem kein Knoten mehrmals vorkommt, das heißt $v_i \neq v_j$ für $1 \leq i \leq j \leq k$. Ein **Kreis** $C(V, E)$ ist ein nicht-trivialer geschlossener Spaziergang $v_1, e_1, v_2, \ldots, e_k, v_1$, in dem kein Knoten außer dem ersten und letzten mehrfach vorkommt, somit gilt $v_i \neq v_j$ für $2 \leq i \leq j \leq k - 1$.

Da in einem schlichten Graphen keine parallelen Kanten zugelassen sind, ist ein Spaziergang $v_1, e_1, v_2, \ldots, e_k, v_k$ eindeutig durch seine Knoten bestimmt. In diesem Fall können wir den $v_1$-$v_k$-Spaziergang ausschließlich durch seine Knotenfolge $v_1, v_2, \ldots, v_k$ beschreiben.

Im weiteren Verlauf werden wir stets Kreise als Grundlage unserer Fragestellungen betrachten. Diese weisen das Merkmal auf, dass zwischen zwei beliebigen verschiedenen Knoten $v_i$ und $v_j$ genau zwei $v_i$-$v_j$-Wege existieren. Einer der Wege verläuft im Uhrzeigersinn entlang des Kreises und der andere entgegen des Uhrzeigersinns. Damit wir die gegenläufigen Wege voneinander unterscheiden können, definieren wir die aufsteigenden und absteigenden Wege, und illustrieren diese anhand des Beispiels in Abbildung 1.

**Definition 2.5.** Seien $v_i$ und $v_j$ zwei beliebige verschiedene Knoten eines Kreises $G(V, E)$. Die Kante mit dem größten Index im Kreis sei $e_{\max} = e_{|E|} \in E$. Wir bezeichnen den $v_i$-$v_j$-Weg, der die Kante $e_{\max}$ nicht enthält, als **aufsteigenden Weg** $P^+(V^+, E^+)$. Entsprechend bezeichnen wir den $v_i$-$v_j$-Weg, der die Kante $e_{\max}$ enthält, als **absteigenden Weg** $P^-(V^-, E^-)$.

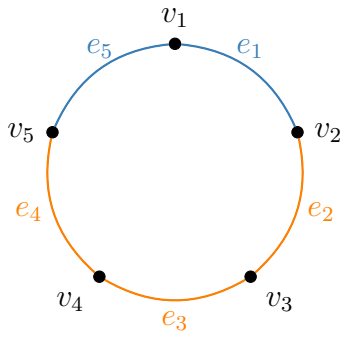



**Abbildung 1:** Beispiel für einen aufsteigenden und absteigenden $v_2$-$v_5$-Weg in einem Kreisgraphen $G(V, E)$. Der aufsteigende Weg $P^+(\{v_2, v_3, v_4, v_5\}, \{e_2, e_3, e_4\})$ ist orange markiert, während der absteigende Weg $P^-(\{v_2, v_1, v_5\}, \{e_1, e_5\})$ in Blau dargestellt ist. Der absteigende Weg $P^-$ enthält die Kante mit dem größten Index, $e_{\max} = e_5$.

Um Graphen mit zusätzlichen Informationen zu codieren, erweitern wir sie durch die Zuweisung von Kantengewichten.

**Definition 2.6.** Ein Graph $G(V, E, \omega)$ mit einer Gewichtsfunktion $\omega : E \to \mathbb{R}$, die jeder Kante $e \in E$ eine reelle Zahl $\omega(e)$ als Kantengewicht zuordnet, heißt **gewichteter Graph**.

**Definition 2.7.** Sei $G(V, E, \omega)$ ein gewichteter Graph und $P(V_P, E_P, \omega_P)$ ein Weg in $G$. Die **Länge des Weges** $|P|$ ist gleich der Summe seiner Kantengewichte $|P| = \sum_{i=1}^{|E_P|} \omega_P(e_i)$. Der **kürzeste Weg** zwischen zwei verschiedenen Knoten $v_i$ und $v_j$ in einem Graphen ist der Weg, der die beiden Knoten mit minimaler Länge verbindet.

Im Zusammenhang mit kürzesten Wegen betrachten wir ein spezielles Paar von Graphen, das für die spätere Problemformulierung des IKWP von zentraler Bedeutung ist. Die beiden Graphen basieren auf der gleichen Knotenmenge, weisen jedoch im Allgemeinen unterschiedliche Kantenmengen auf. Charakteristisch ist, dass die Kantengewichte des einen Graphen den Längen der kürzesten Wege des anderen Graphen entsprechen.

**Definition 2.8.** Sei $G(V, E)$ ein Graph und $G_d(V, D, \delta)$ ein gewichteter Graph mit nichtnegativer Gewichtsfunktion $\delta : D \to \mathbb{R}_0^+$. Wir bezeichnen $G_d$ als **Distanzgraph** auf $G$ und die Elemente $d \in D$ als **Distanzkanten**. Die Gewichtsfunktion $\omega : E \to \mathbb{R}_0^+$ heißt $G_d$**-erfüllende Gewichtsfunktion** auf $G(V, E, \omega)$, wenn für alle Distanzkanten $d = v_i v_j \in D$ das Kantengewicht $\delta(d)$ mit der Länge des kürzesten $v_i$-$v_j$-Weges in $G$ übereinstimmt.

Vorausgesetzt, es existiert eine $G_d$-erfüllende Gewichtsfunktion $\omega$, entsprechen die Kantengewichte des Distanzgraphen $G_d$ den Längen der kürzesten Wege im gewichteten Graphen $G$.

Betrachten wir Kreise statt beliebiger Graphenstrukturen als zugrunde liegende Graphen, so können wir die Distanzkanten den kürzesten auf- und absteigenden Wegen innerhalb des Kreises zuordnen. Hierzu definieren wir zwei zusätzliche Matrizen. Diese beschreiben, welche Kanten des Kreises jeweils zum aufsteigenden bzw. absteigenden Weg einer bestimmten Distanzkante gehören.

**Definition 2.9.** Sei $G(V, E, \omega)$ ein Kreis mit $G_d$-erfüllender Gewichtsfunktion $\omega$ und $G_d(V, D, \delta)$ ein Distanzgraph auf $G$. Eine Distanzkante $d = v_i v_j \in D$ heißt **aufsteigende Distanzkante**, wenn der kürzeste $v_i$-$v_j$-Weg im Kreis $G$ ein aufsteigender Weg ist. Analog heißt die Distanzkante $d$ **absteigend**, wenn der kürzeste $v_i$-$v_j$-Weg im Kreis $G$ ein absteigender Weg ist.

**Definition 2.10.** Sei $G(V, E, \omega)$ ein Kreis mit $G_d$-erfüllender Gewichtsfunktion $\omega$ und $G_d(V, D, \delta)$ ein Distanzgraph auf $G$. Die **aufsteigende Wegmatrix** $\mathbf{W}^+$ ist eine $(|D| \times |E|)$-Binärmatrix, welche definiert ist durch

$$\mathbf{W}^+_{ij} = \begin{cases} 1 & \text{falls } e_j \text{ im zu } d_i \text{ zugehörigen aufsteigenden Weg enthalten ist,} \\ 0 & \text{sonst.} \end{cases}$$

Analog wird die **absteigende Wegmatrix** $\mathbf{W}^-$ definiert durch

$$\mathbf{W}^-_{ij} = \begin{cases} 1 & \text{falls } e_j \text{ im zu } d_i \text{ zugehörigen absteigenden Weg enthalten ist,} \\ 0 & \text{sonst.} \end{cases}$$

Die Wegmatrizen $\mathbf{W}^+$ und $\mathbf{W}^-$ sind zueinander Komplementärmatrizen. Es gilt somit $\mathbf{W}^+ = \mathbf{1}_{|D|\times|E|} - \mathbf{W}^-$.

Anhand eines Beispiels, siehe Abbildung 2, veranschaulichen wir uns die soeben eingeführten Definitionen, die im Zusammenhang mit Distanzgraphen stehen.

Die Abbildung 2 zeigt einen Kreis $G$ mit fünf Knoten, fünf Kanten und einer $G_d$-erfüllender Gewichtsfunktion sowie einen zugehörigen Distanzgraphen $G_d$ mit drei Distanzkanten. Vergleichen wir das Kantengewicht der Distanzkante $d_1 = v_2 v_5$ im Distanzgraphen mit der Länge des kürzesten $v_2$-$v_5$-Weges im Kreis, so stellen wir fest, dass beide den Wert 15 haben. Da der kürzeste $v_2$-$v_5$-Weg in $G$ absteigend ist, ist die Distanzkante $d_1$ als absteigende Distanzkante klassifiziert.

Die Funktion $\omega$ ist eine $G_d$-erfüllende Gewichtsfunktion, da die Kantengewichte aller Distanzkanten mit den Längen der kürzesten Wege im Kreis übereinstimmen. Anhand der beiden $v_2$-$v_4$-Wegen im Kreis erkennen wir, dass auf- und absteigende Wege dieselbe Länge haben können, wie im vorliegenden Fall mit einer Länge von 20. Zudem zeigt sich, dass die Gewichtsfunktion $\omega$ nicht notwendigerweise eindeutig ist. Ändern wir beispielsweise die Kantengewichte von $e_1$ und $e_5$ zu $\omega(e_1) = 10$ und $\omega(e_5) = 5$, so bleibt $\omega$ weiterhin $G_d$-erfüllend, da die Länge des kürzesten $v_2$-$v_5$-Weges unverändert bleibt und nach wie vor dem Kantengewicht $\delta(d_1) = 15$ der Distanzkante $d_1$ entspricht.

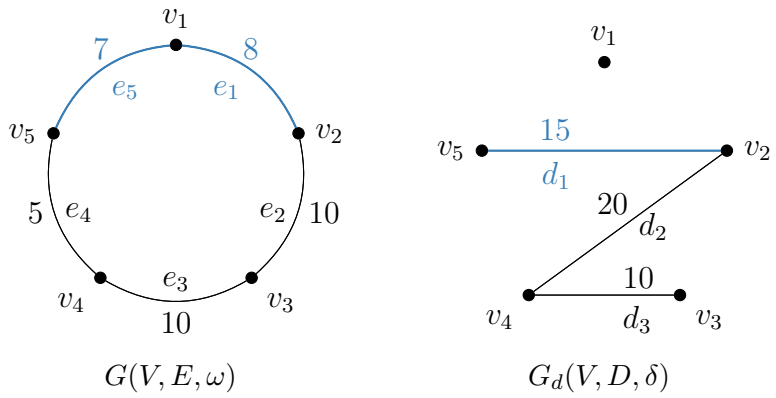



**Abbildung 2:** Beispiel einer $G_d$-erfüllenden Gewichtsfunktion $\omega$ auf einem Kreis $G(V, E, \omega)$ und des zugehörigen Distanzgraphen $G_d(V, D, \delta)$. Die blau markierten Kanten stellen die Distanzkante $d_1$ im Distanzgraphen $G_d$ und den dazugehörigen kürzesten $v_2$-$v_5$-Weg in Kreis $G$ dar.

Die aufsteigende und absteigende Wegmatrix für die in Abbildung 2 dargestellten Graphen sind wie folgt:

$$\mathbf{W}^+ = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \text{ und } \mathbf{W}^- = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Zum Beispiel zeigt das Matrixelement $\mathbf{W}^+_{23} = 1$ an, dass die Kante $e_3$ im aufsteigenden Weg zur Distanzkante $d_2$ enthalten ist. Dies entspricht dem $v_2$-$v_4$-Weg $P^+(\{v_2, v_3, v_4\}, \{e_2, e_3\})$ im Kreis.

## 2.2 Das inverse Kürzeste-Wege-Problem

Während im Kürzeste-Wege-Problem die Kantengewichte eines Graphen bekannt sind und dazu verwendet werden einen, mehrere oder alle kürzesten Wege im Graphen zu bestimmen, stellt das inverse Kürzeste-Wege-Problem (IKWP) die Umkehrung dieser Fragestellung dar. Im inversen Fall sind nicht die Kantengewichte, sondern die Längen der kürzesten Wege zwischen bestimmten Knotenpaaren vorgegeben. Ziel ist es, eine Gewichtsfunktion für die Kanten des Graphen zu finden, die diese kürzesten Wege realisiert.

Gemäß den im vorangegangen Abschnitt 2.1 eingeführten Definitionen wird die Länge der kürzesten Wege eines Graphen durch dessen Distanzgraph beschrieben. Wir definieren das IKWP in Anlehnung an das Entscheidungsproblem in [Mol95, S. 8] wie folgt:

**Definition 2.11 (IKWP).** Seien ein Graph $G(V, E)$ und ein Distanzgraph $G_d(V, D, \delta)$ mit nichtnegativer Gewichtsfunktion $\delta : D \to \mathbb{R}_0^+$ auf $G_d$ gegeben. Finde eine nichtnegative $G_d$-erfüllende Gewichtsfunktion $\omega : E \to \mathbb{R}_0^+$ auf $G(V, E, \omega)$ oder zeige, dass keine solche Funktion existiert.

Falls das IKWP lösbar ist, existiert eine $G_d$-erfüllende Gewichtsfunktion $\omega$ auf $G$, sodass die Kantengewichte $\delta(v_i v_j)$ der Distanzkanten $v_i v_j \in D$ den Längen der kürzesten $v_i$-$v_j$-Wege in $G$ entsprechen. Zu bestimmen, ob eine solche Gewichtsfunktion existiert oder nicht, ist ein $\mathcal{NP}$-vollständiges Problem, siehe [Mol95, S. 18]. Die Klasse der $\mathcal{NP}$-vollständigen Probleme umfasst Entscheidungsprobleme, die sowohl in $\mathcal{NP}$ liegen als auch $\mathcal{NP}$-schwer sind. Dabei gilt:

- Ein Problem liegt in $\mathcal{NP}$, wenn dessen Lösung in polynomieller Zeit verifiziert werden kann.
- Ein Problem ist $\mathcal{NP}$-schwer, wenn jedes andere Problem aus $\mathcal{NP}$ auf dieses in polynomieller Zeit reduziert werden kann.

Ob Algorithmen existieren, die Probleme aus $\mathcal{NP}$ in polynomieller Zeit lösen, ist weiterhin offen. Hierbei handelt es sich um das sogenannte $\mathcal{NP} \neq \mathcal{P}$-Problem, vgl. [Weg03].

## 2.3 Das inverse Kürzeste-Wege-Problem auf Kreisen

Ein Spezialfall des im Abschnitt 2.2 beschriebenen IKWP ist das inverse Kürzeste-Wege-Problem auf Kreisen (IKWPK). Dieses ist insofern eine spezielle Variante des IKWP, da der zugrunde liegende Graph, für den eine $G_d$-erfüllende Gewichtsfunktion bestimmt werden soll, ein Kreis ist.Um das IKWPK in diesem Abschnitt als ganzzahliges lineares Problem zu formulieren, fordern wir, dass die gesuchte $G_d$-erfüllende Gewichtsfunktion sowohl ganzzahlige Werte annimmt als auch das Gesamtgewicht des Kreises minimiert.

**Definition 2.12 (IKWPK).** Seien ein Kreis $G(V, E)$ und ein Distanzgraph $G_d(V, D, \delta)$ mit nichtnegativer Gewichtsfunktion $\delta : D \to \mathbb{Z}^+$ auf $G_d$ gegeben. Finde eine nichtnegative $G_d$-erfüllende Gewichtsfunktion $\omega : E \to \mathbb{Z}_0^+$ auf $G(V, E, \omega)$, die das Gesamtgewicht $\sum_{j=1}^{|E|} \omega(e_j)$ des Kreises minimiert, oder zeige, dass keine solche Funktion existiert.

Als ganzzahliges lineares Problem formuliert, erfolgt die Minimierung des Gesamtgewichts durch die folgende Zielfunktion wie in [Sau23, S. 8] beschrieben:

$$\min \sum_{j=1}^{|E|} \omega(e_j) \tag{1}$$

unter den Nebenbedingungen

$$\sum_{j \in \mathbf{W}_{i\cdot}^{+}} \omega(e_j) \geq \delta(d_i) \qquad \forall i \in \{1, \ldots, |D|\} \tag{2}$$

$$\sum_{j \in \mathbf{W}_{i\cdot}^{-}} \omega(e_j) \geq \delta(d_i) \qquad \forall i \in \{1, \ldots, |D|\} \tag{3}$$

$$\sum_{j \in \mathbf{W}_{i\cdot}^{+}} \omega(e_j) - b_i \cdot \mathrm{K} \leq \delta(d_i) \qquad \forall i \in \{1, \ldots, |D|\} \tag{4}$$

$$\sum_{j \in \mathbf{W}_{i\cdot}^{-}} \omega(e_j) - (1 - b_i) \cdot \mathrm{K} \leq \delta(d_i) \qquad \forall i \in \{1, \ldots, |D|\} \tag{5}$$

$$b_i \in \{0, 1\} \qquad \forall i \in \{1, \ldots, |D|\} \tag{6}$$

$$\omega(e_j) \geq 0 \qquad \forall j \in \{1, \ldots, |E|\}, \tag{7}$$

wobei $\mathrm{K} \in \mathbb{R}$ gilt.

Bei der ursprünglichen Formulierung des inversen Problems, siehe Definition 2.11, ist es nicht erforderlich, eine Lösung mit minimalem Gesamtgewicht zu suchen. Dennoch orientieren wir uns an den Ansätzen früherer Arbeiten in diesem Bereich und verwenden die Zielfunktion (1), die das Gesamtgewicht des Kreises minimiert. Hierdurch stellen wir sicher, dass unter allen zulässigen Lösungen eine mit möglichst geringen Kantengewichten gewählt wird.

Wie wir bereits im Abschnitt 2.1 festgestellt haben, werden zwei verschiedene Knoten $v_i$ und $v_j$ eines Kreises durch genau zwei Wege verbunden. Das heißt, wir wissen nicht im Voraus, ob der kürzeste Weg dem aufsteigenden oder dem absteigenden $v_i$-$v_j$-Weg entspricht. Daher sind zusätzliche Nebenbedingen notwendig, um die Längen beider Wege sowohl von oben als auch von unten zu beschränken. Die Bedingungen (2) und (3) stellen sicher, dass sowohl der aufsteigende als auch der absteigende Weg mindestens die Länge der zugehörigen Distanzkante hat. Damit der kürzeste Weg exakt die Länge der zugehörigen Distanzkante erreicht, fügen wir die Bedingungen (4) und (5) hinzu. Diese enthalten die Binärvariablen $b_i$ und die reelle Konstante K. Die Binärvariablen legen wir für jede Distanzkante so fest, dass $b_i = 0$ gilt, wenn der kürzeste Weg aufsteigend ist. Dadurch entfällt die Subtraktion der Konstante K in der Nebenbedingung (4). Die Kombination der Bedingungen (2) und (4) garantiert, dass die Länge des aufsteigenden Weges exakt der Länge der zugehörigen Distanzkante entspricht. Analog gilt $b_i = 1$, wenn der kürzeste Weg absteigend ist. Die Konstante K stellt sicher, dass die Bedingung (4) bzw. (5) für den jeweils längeren der beiden $v_i$-$v_j$-Wege im Kreis erfüllt wird. Die Nebenbedingungen ergänzen wir durch die Nichtnegativitätsbedingung (7), welche sicherstellt, dass alle Kantengewichte des Kreises nichtnegative Werte annehmen.

Aufgrund der exponentiellen Wachstumsrate der möglichen Kombinationen der Binärvariablen, beschrieben durch $2^{|D|}$, ist das Problem nicht in polynomieller Zeit lösbar. Daher gehört das IKWPK als ganzzahliges lineares Problem ebenso wie das IKWP, vgl. Abschnitt 2.2, zur Klasse der $\mathcal{NP}$-vollständigen Probleme, siehe [Sch98, S. 245].

## 2.4 Das lineare Komplementaritätsproblem

Das lineare Komplementaritätsproblem (LKP) ist ein Problem in der Optimierungstheorie mit zahlreichen Anwendungen, unter anderem in der quadratischen Programmierung, der Spieltheorie und der Modellierung von Marktgleichgewichten, siehe [CPS09, S. 4 ff.]. Im Gegensatz zu klassischen Optimierungsproblemen, die durch eine Zielfunktion charakterisiert sind, existiert beim LKP keine zu maximierende oder zu minimierende Funktion. Stattdessen werden zwei Vektoren bestimmt, die ein System von Gleichungen und Ungleichungen erfüllen.

Formal definieren wir das LKP, angelehnt an [CPS09, S. 2], wie folgt:

**Definition 2.13 (LKP).** Sei eine quadratische Matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ und ein Vektor $\mathbf{q} \in \mathbb{R}^n$ gegeben. Finde Vektoren $\mathbf{w} \in \mathbb{R}^n$ und $\mathbf{z} \in \mathbb{R}^n$, sodass die folgenden Bedingungen erfüllt sind:

$$\mathbf{w} = \mathbf{M}\mathbf{z} + \mathbf{q} \tag{8}$$

$$\mathbf{w}, \mathbf{z} \geq \mathbf{0} \tag{9}$$

$$\mathbf{w}^\mathsf{T}\mathbf{z} = 0, \tag{10}$$

oder zeige, dass keine solche Vektoren existieren. Das lineare Komplementaritätsproblem bezeichnen wir mit LKP$(\mathbf{M}, \mathbf{q})$.

Die Nichtnegativitätsbedingungen (9) stellen sicher, dass alle Komponenten der Vektoren $\mathbf{w}$ und $\mathbf{z}$ nichtnegativ sind. Unter Hinzunahme der Bedingung (10) gilt für jedes $i \in \{1, \ldots, n\}$ entweder

$$\mathbf{w}_i = 0 \text{ und } \mathbf{z}_i \geq 0 \tag{11}$$

oder

$$\mathbf{w}_i \geq 0 \text{ und } \mathbf{z}_i = 0. \tag{12}$$

Das bedeutet, dass für jedes Paar $(\mathbf{w}_i, \mathbf{z}_i)$ höchstens eine der beiden Komponenten einen positiven Wert bzw. mindestens eine der Komponenten den Wert null annimmt. Eine Bedingung mit dieser Eigenschaft nennen wir Komplementaritätsbedingung.

Die Lösbarkeit eines LKP hängt im Wesentlichen von den Eigenschaften der Matrix $\mathbf{M}$ ab. Ist $\mathbf{M}$ positiv definit, so existiert für jeden Vektor $\mathbf{q} \in \mathbb{R}^n$ eine eindeutige Lösung des LKP, siehe [CPS09, S. 141]. Eine Matrix ist genau dann positiv definit, wenn für alle nichttrivialen Vektoren $\mathbf{x} \in \mathbb{R}^n$ die Bedingung $\mathbf{x}^\mathsf{T}\mathbf{M}\mathbf{x} > 0$ erfüllt ist. Eine weitere Klasse von Matrizen, für die eine eindeutige Lösung des LKP existiert, sind die sogenannten $\mathbf{P}$-Matrizen. Eine Matrix $\mathbf{M}$ heißt $\mathbf{P}$-Matrix, wenn alle ihre Hauptminoren positiv sind, siehe [CPS09, S. 147]. Das LKP besitzt für jeden Vektor $\mathbf{q} \in \mathbb{R}^n$ genau dann eine eindeutige Lösung, wenn $\mathbf{M}$ eine $\mathbf{P}$-Matrix ist, siehe [CPS09, S. 148].

Wenn keine spezifischen Eigenschaften der Matrix $\mathbf{M}$ vorliegen, gehört das LKP zu den $\mathcal{NP}$-vollständigen Problemen. Für bestimmte Klassen von Matrizen existieren jedoch Algorithmen mit polynomieller Laufzeit. Beispielsweise kann das LKP für positiv semidefinite Matrizen $\mathbf{M}$ in $\mathcal{O}(n^3 L)$ gelöst werden, wobei $L$ von der Größe der Matrix $\mathbf{M}$ abhängt, vgl. [KMY89].

## 2.5 Das inverse Kürzeste-Wege-Problem auf Kreisen und Komplementarität

In diesem Abschnitt führen wir die Unterkapitel 2.3 und 2.4 zusammen und analysieren, ob sich das IKWPK von einem ganzzahligen linearen Problem in ein LKP transformieren lässt. Eine Möglichkeit ist, das IKWPK zunächst in die Standardform zu überführen und anschließend mithilfe eines dualen Vektors sowie zweier Schlupfvektoren in ein LKP umzuwandeln, vgl. [MY97, S. 9 ff.]. Dieser Ansatz hat jedoch den Nachteil, dass die Binärvariablen erhalten bleiben und das Problem weiterhin nicht in polynomieller Zeit lösbar ist. Daher entscheiden wir uns für eine andere Vorgehensweise, bei der wir aus den Nebenbedingungen des IKWPK eine Komplementaritätsbedingung ableiten. Dies ermöglicht uns, die Binärvariablen vollständig aus der Problemstellung zu eliminieren. Dies ist relevant, da deren Anzahl an möglichen Kombinationen mit der Anzahl der Distanzkanten exponentiell wächst.

Im ersten Schritt formulieren wir das IKWPK als ganzzahliges lineares Minimierungsproblem, siehe Unterkapitel 2.3, in Matrixform. Dazu führen wir die Vektoren $\mathbf{x} \in \mathbb{Z}_0^{+|E|}$ und $\mathbf{d} \in \mathbb{Z}^{+|D|}$ ein, die die gesuchten Kantengewichte des Kreises bzw. die der Distanzkanten des Distanzgraphen enthalten. Hieraus resultiert das IKWPK in der nachstehenden Form:

$$\min \mathbf{1}_{|E|\times 1}^{\mathsf{T}} \mathbf{x} \tag{13}$$

unter den Nebenbedingungen

$$\mathbf{W}^{+}\mathbf{x} \geq \mathbf{d} \tag{14}$$
$$(\mathbf{1}_{|D|\times|E|} - \mathbf{W}^{+})\,\mathbf{x} \geq \mathbf{d} \tag{15}$$
$$\mathbf{W}^{+}\mathbf{x} - \mathbf{b}\cdot \mathrm{K} \leq \mathbf{d} \tag{16}$$
$$(\mathbf{1}_{|D|\times|E|} - \mathbf{W}^{+})\,\mathbf{x} - (\mathbf{1}_{|D|\times 1} - \mathbf{b})\cdot \mathrm{K} \leq \mathbf{d} \tag{17}$$
$$\mathbf{x} \geq \mathbf{0} \tag{18}$$
$$\mathbf{b} \in \{0,1\}^{|D|}. \tag{19}$$

Im Folgenden betrachten wir ausschließlich die Nebenbedingungen. Diese stellen wir so um, dass die rechten Seiten der Ungleichungen Nullvektoren enthalten. Dies führt zu:

$$\mathbf{W}^{+}\mathbf{x} - \mathbf{d} \geq \mathbf{0} \tag{20}$$
$$(\mathbf{1}_{|D|\times|E|} - \mathbf{W}^{+})\,\mathbf{x} - \mathbf{d} \geq \mathbf{0} \tag{21}$$
$$\mathbf{W}^{+}\mathbf{x} - \mathbf{d} - \mathbf{b}\cdot \mathrm{K} \leq \mathbf{0} \tag{22}$$
$$(\mathbf{1}_{|D|\times|E|} - \mathbf{W}^{+})\,\mathbf{x} - \mathbf{d} - (\mathbf{1}_{|D|\times 1} - \mathbf{b})\cdot \mathrm{K} \leq \mathbf{0} \tag{23}$$
$$\mathbf{x} \geq \mathbf{0} \tag{24}$$
$$\mathbf{b} \in \{0,1\}^{|D|}. \tag{25}$$

Die Binärvariablen, die wir eliminieren möchten, sind in den Bedingungen (22) und (24) enthalten. Eine Binärvariable $b_i$ nimmt den Wert null an, wenn der kürzeste

Weg zur Distanzkante $d_i$ im Graphen $G$ aufsteigend ist, und den Wert eins, wenn dieser absteigend ist. Daher gelten für jede Distanzkante $d_i$ mit $i \in \{1, \ldots, |D|\}$ im IKWPK entweder die Bedingungen

$$\mathbf{W}^+_{i\cdot}\mathbf{x} - \mathbf{d}_i \leq 0 \text{ und } (\mathbf{1}_{i\cdot} - \mathbf{W}^+_{i\cdot})\,\mathbf{x} - \mathbf{d}_i - \mathrm{K} \leq 0, \tag{26}$$

wenn die Distanzkante aufsteigend ist, oder

$$\mathbf{W}^+_{i\cdot}\mathbf{x} - \mathbf{d}_i - \mathrm{K} \leq 0 \text{ und } (\mathbf{1}_{i\cdot} - \mathbf{W}^+_{i\cdot})\,\mathbf{x} - \mathbf{d}_i \leq 0, \tag{27}$$

wenn die Distanzkante absteigend ist. Unter Hinzunahme der Bedingungen (20) und (21) folgt, dass für jedes $i$ entweder $\mathbf{W}^+_{i\cdot}\mathbf{x} - \mathbf{d}_i = 0$ oder $(\mathbf{1}_{i\cdot} - \mathbf{W}^+_{i\cdot})\,\mathbf{x} - \mathbf{d}_i = 0$ gilt. Dies führt zur Komplementaritätsbedingung

$$(\underbrace{\mathbf{W}^+\mathbf{x} - \mathbf{d}}_{\geq \mathbf{0}})^\mathsf{T} \cdot (\underbrace{(\mathbf{1}_{|D|\times|E|} - \mathbf{W}^+)\,\mathbf{x} - \mathbf{d}}_{\geq \mathbf{0}}) = \mathbf{0}, \tag{28}$$

die es uns ermöglicht, das IKWPK als Minimierungsproblem ohne Binärvariablen zu formulieren. Die ursprünglichen Bedingungen (22) und (24) werden dabei durch die Komplementaritätsbedingung (28) ersetzt. Somit erhalten wir die folgende Darstellung des Minimierungsproblems:

$$\min \mathbf{1}^\mathsf{T}_{|E|\times 1}\mathbf{x} \tag{29}$$

unter den Nebenbedingungen

$$\mathbf{W}^+\mathbf{x} - \mathbf{d} \geq \mathbf{0} \tag{30}$$
$$(\mathbf{1}_{|D|\times|E|} - \mathbf{W}^+)\,\mathbf{x} - \mathbf{d} \geq \mathbf{0} \tag{31}$$
$$(\mathbf{W}^+\mathbf{x} - \mathbf{d})^\mathsf{T} \cdot ((\mathbf{1}_{|D|\times|E|} - \mathbf{W}^+)\,\mathbf{x} - \mathbf{d}) = \mathbf{0} \tag{32}$$
$$\mathbf{x} \geq \mathbf{0}. \tag{33}$$

Im nächsten Schritt überführen wir das Minimierungsproblem in eine LKP-ähnliche Form. Hierzu definieren wir die Vektoren

$$\mathbf{w} = \mathbf{W}^+\mathbf{x} - \mathbf{d} \text{ und } \mathbf{z} = (\mathbf{1} - \mathbf{W}^+)\,\mathbf{x} - \mathbf{d}. \tag{34}$$

Basierend auf den Nichtnegativitätsbedingungen (30) und (31) sowie der Komplementaritätsbedingung (32) aus dem IKWPK erhalten wir die Nebenbedingungen wie sie im LKP$(\mathbf{M}, \mathbf{q})$, siehe Definition 2.13, formuliert sind. Um alle Nebenbedingungen des ursprünglichen Minimierungsproblems zu berücksichtigen, ergänzen wir das Problem um die Nichtnegativitätsbedingung (33).

Setzen wir die Vektoren $\mathbf{w}$ und $\mathbf{z}$ in die Standardform des LKP, also die Gleichung $\mathbf{w} = \mathbf{M}\mathbf{z} + \mathbf{q}$ ein, so erhalten wir:

$$\mathbf{W}^+\mathbf{x} - \mathbf{d} = \mathbf{M} \cdot ((\mathbf{1}_{|D|\times|E|} - \mathbf{W}^+)\,\mathbf{x} - \mathbf{d}) + \mathbf{q}, \tag{35}$$

Jedoch zeigt sich, dass die Gleichung (35) nicht nach den gesuchten Kreiskantengewichten im Vektor $\mathbf{x}$ aufgelöst werden kann, da die Wegmatrix $\mathbf{W}^+$ keine Inverse besitzt. Dies hat zwei Ursachen:

1. Die Matrix $\mathbf{W}^+$ ist im Allgemeinen nicht quadratisch, da sie die Zuordnung von Kanten zu aufsteigenden Wegen pro Distanzkante beschreibt und daher die Dimension $|D| \times |E|$ besitzt.

2. Auch im quadratischen Fall ist $\mathbf{W}^+$ nicht invertierbar, da ihre Determinante stets null ist. Dies ist der Fall, da die Kreiskante mit dem höchsten Index $e_{\max}$ in keinem aufsteigenden Weg enthalten ist, sodass die letzte Spalte von $\mathbf{W}^+$ eine Nullspalte ist.

Daher können wir das IKWPK mithilfe der hergeleiteten Formulierung nicht lösen. Im Kapitel 3 betrachten wir deshalb eine Verallgemeinerung des LKP, das erweiterte lineare Komplementaritätsproblem, um eine Lösung für das IKWPK zu ermöglichen.

# 3 Das inverse Kürzeste-Wege-Problem auf Kreisen als erweitertes lineares Komplementaritätsproblem

Im vorangegangenen Unterkapitel 2.5 haben wir gezeigt, dass das IKWPK aufgrund der fehlenden Invertierbarkeit der Wegmatrix nicht in eine LKP-ähnliche Form gebracht werden kann. In diesem Kapitel erweitern wir daher unseren Ansatz, indem wir das erweiterte lineare Komplementaritätsproblem (ELKP) einführen und das IKWPK entsprechend zum ELKP sowie zu dessen homogener Variante transformieren. Hierbei orientieren wir uns an der Veröffentlichung von De Schutter und De Moor [DSDM95], die das ELKP als Verallgemeinerung des LKP eingeführt und Algorithmen zur Lösung des Problems entwickelt haben. Um die Effizienz der Algorithmen zu steigern, passen wir die einzelnen Module an unser spezifisches Problem an.

## 3.1 Das erweiterte lineare Komplementaritätsproblem

Das erweiterte lineare Komplementaritätsproblem (ELKP) ist eine Verallgemeinerung des LKP und ist nach [DSDM95, S. 292 f.] wie folgt definiert:

**Definition 3.1** (**ELKP**)**.** Seien die Matrizen $\mathbf{A} \in \mathbb{R}^{p \times n}$ und $\mathbf{B} \in \mathbb{R}^{q \times n}$, die Vektoren $\mathbf{c} \in \mathbb{R}^p$ und $\mathbf{e} \in \mathbb{R}^q$ sowie $m$ Teilmengen $\phi_j \subseteq \{1, 2, \ldots, p\}$ für $j \in \{1, \ldots, m\}$ gegeben. Finde einen Vektor $\mathbf{x} \in \mathbb{R}^n$, sodass

$$\sum_{j=1}^{m} \prod_{i \in \phi_j} (\mathbf{Ax} - \mathbf{c})_i = 0 \tag{36}$$

unter den Nebenbedingungen

$$\mathbf{Ax} \geq \mathbf{c} \tag{37}$$
$$\mathbf{Bx} = \mathbf{e} \tag{38}$$

gilt, oder zeige, dass kein solcher Vektor existiert.

Die Komplementaritätsbedingung des ELKP ist durch die Gleichung (36) gegeben. Da aus der Nebenbedingung (37) folgt, dass $\mathbf{Ax} - \mathbf{c} \geq \mathbf{0}$ gilt, muss für jede Teilmenge $\phi_j$ das Produkt der jeweiligen Komponenten in der Komplementaritätsbedingung null sein. Dies erlaubt uns, die Komplementaritätsbedingung nicht global, sondern partiell auf jeder Teilmenge $\phi_j$ durch

$$\prod_{i \in \phi_j} (\mathbf{Ax} - \mathbf{c})_i = 0 \text{ für } j \in \{1, \ldots, m\} \tag{39}$$

zu beschreiben. Hierbei umfasst jede Teilmenge $\phi_j$ eine Gruppe von Ungleichungen aus $\mathbf{Ax} \geq \mathbf{c}$, welche zusammen die partielle Komplementaritätsbedingung erfüllen müssen. Damit das Produkt über die Komponenten der Teilmenge $\phi_j$ null wird, muss mindestens ein Faktor gleich null sein, also mindestens eine der zugehörigen

Ungleichungen muss mit Gleichheit erfüllt sein.

Wir definieren zusätzlich das homogene erweiterte lineare Komplementaritätsproblem (HELKP), da die von De Schutter und De Moor in [DSDM95] beschriebenen Algorithmen speziell die homogene Variante des ELKP lösen. Zur Homogenisierung des ELKP führen wir einen nichtnegativen Skalar $\alpha \geq 0$ ein mit dem wir den gesuchten Vektor $\mathbf{x}$ aus dem ELKP erweitern. Dadurch erhalten wir den im HELKP gesuchten Vektor

$$\mathbf{u} = \begin{pmatrix} \mathbf{x} \\ \alpha \end{pmatrix}.$$

Um die Nebenbedingungen (37) und (38) des ELKP zu homogenisieren, definieren wir die erweiterten Matrizen

$$\mathbf{P} = \begin{pmatrix} \mathbf{A} & -\mathbf{c} \\ \mathbf{0}_{1\times n} & 1 \end{pmatrix} \text{ und } \mathbf{Q} = \begin{pmatrix} \mathbf{B} & -\mathbf{e} \end{pmatrix}.$$

Unter Verwendung der beiden Matrizen und des Vektors $\mathbf{u}$ formulieren wir das HELKP analog zur inhomogenen Variante nach [DSDM95, S. 293] wie folgt:

**Definition 3.2** (**HELKP**). Seien die Matrizen $\mathbf{P} \in \mathbb{R}^{p\times n}$ und $\mathbf{Q} \in \mathbb{R}^{q\times n}$ sowie $m$ Teilmengen $\phi_j \subseteq \{1, 2, \ldots, p\}$ für $j \in \{1, \ldots, m\}$ gegeben. Finde einen nichttrivialen Vektor $\mathbf{u} \in \mathbb{R}^n$, sodass

$$\sum_{j=1}^{m} \prod_{i\in\phi_j} (\mathbf{Pu})_i = 0 \tag{40}$$

unter den Nebenbedingungen

$$\mathbf{Pu} \geq \mathbf{0} \tag{41}$$
$$\mathbf{Qu} = \mathbf{0} \tag{42}$$

gilt, oder zeige, dass kein solcher Vektor existiert.

Analog zum inhomogenen Fall ist die Komplementaritätsbedingung des HELKP durch die Gleichung (47) gegeben. Aufgrund der Nebenbedingung (48) können wir auch für das HELKP die partielle Bedingung

$$\prod_{i\in\phi_j} (\mathbf{Pu})_i = 0 \text{ für } j \in \{1, \ldots, m\} \tag{43}$$

heranziehen. Die partielle Komplementaritätsbedingung (43) erlaubt es uns, die vollständige Bedingung nicht in jeder Iteration überprüfen zu müssen. Stattdessen prüfen wir für ausgewählte Teilmengen $\phi_j$, ob das Produkt der jeweiligen Komponenten null ergibt. Dieses Vorgehen nutzen wir im Kapitel 3.3 bei der algorithmischen Lösungssuche, um die Effizienz des Algorithmus zu verbessern.

## 3.2 Formale Problembeschreibung des inversen Kürzeste-Wege-Problems auf Kreisen als erweitertes lineares Komplementaritätsproblem

In diesem Abschnitt überführen wir die Problembeschreibung des IKWPK vom LKP, siehe Kapitel 2.5, zum im vorherigen Kapitel eingeführten ELKP bzw. HELKP.

Um das Ungleichungssystem $\mathbf{Ax} \geq \mathbf{c}$ des ELKP für das IKWPK zu formulieren, bestimmen wir zunächst die Matrix $\mathbf{A}$ und den Vektor $\mathbf{c}$. Ausgehend von den im LKP gesuchten Vektoren

$$\mathbf{w} = \mathbf{W}^+\mathbf{x} - \mathbf{d} \text{ und } \mathbf{z} = (\mathbf{1}_{|D|\times|E|} - \mathbf{W}^+)\,\mathbf{x} - \mathbf{d},$$

die beide die Nichtnegativitätsbedingung $\mathbf{w}, \mathbf{z} \geq \mathbf{0}$ des LKP erfüllen sollen, setzen wir

$$\mathbf{A} = \begin{pmatrix} \mathbf{W}^+ \\ \mathbf{1}_{|D|\times|E|} - \mathbf{W}^+ \\ \mathbf{I}_{|E|} \end{pmatrix} \text{ und } \mathbf{c} = \begin{pmatrix} \mathbf{d} \\ \mathbf{d} \\ \mathbf{0}_{|E|\times 1} \end{pmatrix}.$$

Dadurch berücksichtigt die Matrix $\mathbf{A}$ sowohl den aufsteigenden als auch den absteigenden Weg für jede Distanzkante. Der Vektor $\mathbf{c}$ enthält die zugehörigen Distanzkantenlängen. Das Hinzufügen der $|E|$-dimensionalen Einheitsmatrix in der Matrix $\mathbf{A}$ und des $|E|$-dimensionalen Nullvektors im Vektor $\mathbf{c}$ stellt sicher, dass die gesuchten Kreiskantengewichte $\mathbf{x}$ nichtnegativ sind.

Als Nächstes bestimmen wir die Teilmengen $\phi_j \subseteq \{1, 2, \ldots, p\}$. Diese nehmen für das IKWPK eine spezielle Form an, da die Komplementaritätsbedingung des ELKP auf der Komplementaritätsbedingung $\mathbf{w}^\mathsf{T}\mathbf{z} = 0$ des LKP basiert. Letztere fordert, dass für jede Komponente $i \in \{1, 2, \ldots, |D|\}$ entweder $\mathbf{w}_i \geq 0$ und $\mathbf{z}_i = 0$ oder $\mathbf{w}_i = 0$ und $\mathbf{z}_i \geq 0$ gilt. Dementsprechend konstruieren wir für jede Distanzkante eine Teilmenge $\phi_j$, die sowohl den aufsteigenden als auch den absteigenden Weg der Distanzkante berücksichtigt. Jede Teilmenge $\phi_j$ enthält somit genau zwei Indizes, die diese beiden Ungleichungen in $\mathbf{Ax} \geq \mathbf{c}$ repräsentieren. Dadurch erhalten wir für das IKWPK genau $|D|$ Teilmengen $\phi_j = \{i, i + |D|\}$ mit $i \in \{1, 2, \ldots, |D|\}$. Durch die Wahl der Teilmengen ist die Komplementaritätsbedingung des ELKP eindeutig bestimmt.

Falls der Distanzgraph keine parallelen Kanten enthält, bleibt das Gleichungssystem $\mathbf{Bx} = \mathbf{e}$ leer und alle weiteren Bestandteile des ELKP entsprechen der oben beschriebenen Struktur. Eine Distanzkante $d_{ij} \in D$ des Distanzgraphen $G_d$ nennen wir **parallel**, wenn sie dieselben Knoten $v_i, v_j \in V$ verbindet wie eine Kante $e_{ij} \in E$ des Kreises $G$.

Basierend auf dieser Definition formulieren wir nun eine zentrale Eigenschaft der Kantengewichte im IKWPK. Das folgende Theorem stellt sicher, dass das Gewicht einer Kreiskante in $G$ mit dem Gewicht der entsprechenden parallelen Distanzkante im Distanzgraphen $G_d$ übereinstimmt.

**Theorem 3.3.** *Sei $G(V, E, \omega)$ ein Kreis mit $G_d$-erfüllender Gewichtsfunktion $\omega$ und $G_d(V, D, \delta)$ der zugehöriger Distanzgraph. Dann gilt für jede Kante $e_i$ mit*

*$i \in \{1, \ldots, |E|\}$ zu der eine parallele Distanzkante $d_{\hat{i}}$ mit $\hat{i} \in \{1, \ldots, |D|\}$ existiert die Gleichheit*

$$\omega(e_i) = \delta(d_{\hat{i}}).$$

*Beweis.* Wir zeigen die Aussage per Widerspruch. Sei $\mathbf{x} = (\omega(e_1), \omega(e_2), \ldots, \omega(e_{|E|}))^\mathsf{T}$ eine optimale Lösung des IKWPK und $\mathbf{d} = (\delta(d_1), \ldots, \delta(d_{|D|}))^\mathsf{T}$ der Vektor mit den Längen der Distanzkanten in $G_d$. Wir nehmen an, es existiert eine Kante $e_k$ mit zugehöriger paralleler Distanzkante $d_{\hat{k}}$ für die die strikte Ungleichung $\omega(e_k) = \mathbf{x}_k > \mathbf{d}_{\hat{k}} = \delta(d_{\hat{k}})$ gelte. Das heißt, dass das optimale Kantengewicht der Kante $e_k$ strikt größer ist als die Länge ihrer parallelen Distanzkante. Wir konstruieren einen neuen Lösungskandidaten $\mathbf{x}^*$, indem wir $\mathbf{x}^*_k = \mathbf{d}_{\hat{k}}$ setzen und alle anderen Komponenten mit $i \neq k$ von $\mathbf{x}$ übernehmen. Wir zeigen im Folgenden, dass $\mathbf{x}^*$ eine zulässige Lösung des IKWPK ist und zudem ein geringeres Gesamtgewicht besitzt als $\mathbf{x}$.
Die Nichtnegativitätsbedingungen sind für $\mathbf{x}^*$ offensichtlich erfüllt, da per Definition $\delta(d_{\hat{k}}) = \mathbf{d}_{\hat{k}} \geq 0$ gilt und alle übrigen Komponenten übernommen wurden.
Bei den Ungleichheitsbedingungen muss für die Distanzkante $d_{\hat{k}}$ das Paar von komplementären Bedingungen $\mathbf{W}^+_{\hat{k}.}\mathbf{x}^* \geq \mathbf{d}_{\hat{k}}$ und $(\mathbf{1}_{\hat{k}.} - \mathbf{W}^+_{\hat{k}.})\,\mathbf{x}^* \geq \mathbf{d}_{\hat{k}}$ erfüllt sein. Die Ungleichungen stellen sicher, dass sowohl der aufsteigende als auch der absteigende Weg im Kreis mindestens die Länge der jeweiligen Distanzkante hat. Unter Einbeziehung der Komplementaritätsbedingung erhalten wir die stärkere Forderung, dass mindestens eine der Bedingungen mit Gleichheit erfüllt sein muss. Da in der initialen Lösung $\mathbf{x}_k > \mathbf{d}_{\hat{k}}$ gilt, kann der kürzeste Weg nicht ausschließlich der Kante $e_k$ entsprechen sondern muss entlang aller anderen Kanten des Kreises verlaufen. Beim Übergang von $\mathbf{x}$ zu $\mathbf{x}^*$ wird $\mathbf{x}_k$ auf den kleineren Wert $\mathbf{d}_{\hat{k}}$ gesetzt und der kürzeste Weg bleibt erhalten. Für alle anderen Distanzkanten ändert sich ebenfalls nichts, da aufgrund der Konstruktion

$$\mathbf{d}_{\hat{k}} = \mathbf{x}^*_k = \sum_{l \neq k} \mathbf{x}^*_l \geq \sum_{i \in S} \mathbf{x}^*_i \text{ für alle } S \subseteq \{\mathbf{x}^*_1, \mathbf{x}^*_2, \ldots, \mathbf{x}^*_{|E|}\} \backslash \{\mathbf{x}^*_k\}$$

gilt. Das heißt, die Distanzkante $\mathbf{d}_{\hat{k}}$ hat das größte Gewicht im Distanzgraphen und alle kürzesten Wege der anderen Distanzkanten $\mathbf{d}_{\hat{i} \neq \hat{k}}$ sind im kürzesten Weg von $\mathbf{d}_{\hat{k}}$ enthalten. Dementsprechend bleiben für $\mathbf{x}^*$ die Ungleichheitsbedingungen und die Komplementaritätsbedingung unberührt.
Da $\mathbf{x}^*$ die Nichtnegativitäts-, Ungleichheits- und Komplementaritätsbedingungen erfüllt, ist $\mathbf{x}^*$ eine zulässige Lösung des IKWPK. Jedoch hat $\mathbf{x}^*$ ein geringeres Gesamtgewicht als die optimale Lösung $\mathbf{x}$, da $\mathbf{x}^*_k = \mathbf{d}_{\hat{k}} < \mathbf{x}_k$ gilt. Dies widerspricht der Optimalität von $\mathbf{x}$. Daraus folgt, dass das Kantengewicht einer Kreiskante mit dem Gewicht ihrer parallelen Distanzkante übereinstimmen muss. □

Sind parallele Distanzkanten im Distanzgraphen vorhanden, ergänzen wir die Nebenbedingungen $\mathbf{Ax} \geq \mathbf{c}$ um das Gleichungssystem $\mathbf{Bx} = \mathbf{e}$ auf. Dabei repräsentieren die Zeilen der Matrix $\mathbf{B}$ die parallelen Kanten des Kreises, während der Vektor $\mathbf{e}$ die zugehörigen Längen der Distanzkanten enthält. Die Kantengewichte der parallelen Kanten sind durch dieses Gleichungssystem eindeutig bestimmt.

Da parallele Kanten bereits durch $\mathbf{Bx} = \mathbf{e}$ erfasst sind, berücksichtigen wir sie nicht erneut in den Ungleichheitsbedingungen $\mathbf{Ax} \geq \mathbf{c}$. Folglich enthält die Matrix $\mathbf{A}$ nur

die Zeilen, die nicht-parallelen Kanten entsprechen. Gleiches gilt für den Vektor $\mathbf{c}$ sowie die Teilmengen $\phi_j$.

Unter Berücksichtigung von parallelen Distanzkanten und den zugehörigen Kreiskanten, die wir beide mit *par* kennzeichnen, und allen nicht-parallelen Kanten, die wir mit *non-par* markieren, erhalten wir die Formulierung des IKWPK als ELKP wie folgt:

**Definition 3.4 (IKWPK als ELKP).** Seien ein Kreis $G(V, E)$ und ein Distanzgraph $G_d(V, D, \delta)$ mit nichtnegativer Gewichtsfunktion $\delta : D \to \mathbb{Z}^+$ auf $G_d$ sowie der Vektor $\mathbf{d} = (\delta(d_1), \delta(d_2), \ldots, \delta(d_{|D|}))^\mathsf{T}$ mit den Längen der Distanzkanten gegeben. Weiterhin definieren wir die Matrizen

$$\mathbf{A} = \begin{pmatrix} \mathbf{W}^+_{\text{non-par}} \\ \mathbf{1}_{|D_{\text{non-par}}|\times|E|} - \mathbf{W}^+_{\text{non-par}} \\ \mathbf{I}_{|E|} \end{pmatrix} \in {\mathbb{Z}_0^+}^{p\times|E|} \text{ und } \mathbf{B} = \left(\mathbf{I}_{|E_{\text{par}}|}\right) \in {\mathbb{Z}_0^+}^{q\times|E|},$$

sowie die Vektoren

$$\mathbf{c} = \begin{pmatrix} \mathbf{d}_{\text{non-par}} \\ \mathbf{d}_{\text{non-par}} \\ \mathbf{0}_{|E|\times 1} \end{pmatrix} \in {\mathbb{Z}_0^+}^{p} \text{ und } \mathbf{e} = \left(\mathbf{d}_{\text{par}}\right) \in {\mathbb{Z}^+}^{q}.$$

Zusätzlich definieren wir $|D_{\text{non-par}}|$ Teilmengen $\phi_j = \{i, i + |D_{\text{non-par}}|\}$ mit $i \in \{1, 2, \ldots, |D_{\text{non-par}}|\}$. Finde eine nichtnegative $G_d$-erfüllende Gewichtsfunktion $\omega : E \to \mathbb{Z}_0^+$ auf $G(V, E, \omega)$, sodass für den Vektor $\mathbf{x} = (\omega(e_1), \omega(e_2), \ldots, \omega(e_{|E|}))^\mathsf{T}$ die Komplementaritätsbedingung

$$\sum_{j=1}^{m} \prod_{i\in\phi_j} (\mathbf{Ax} - \mathbf{c})_i = 0 \tag{44}$$

unter den Nebenbedingungen

$$\mathbf{Ax} \geq \mathbf{c} \tag{45}$$
$$\mathbf{Bx} = \mathbf{e} \tag{46}$$

gilt und das Gesamtgewicht $\sum_{j=1}^{|E|} \mathbf{x}_j$ des Kreises minimiert wird, oder zeige, dass keine solche Funktion existiert.

Betrachten wir erneut das Beispiel des Kreises und den dazugehörigen Distanzgraphen aus Abbildung 2. Offensichtlich sind die Distanzkanten $d_2$ und $d_4$ parallele Kanten zu $e_2$ und $e_4$, da sie jeweils dieselben Knoten verbinden. Daraus ergibt sich das Gleichungssystem der Nebenbedingung $\mathbf{Bx} = \mathbf{e}$ des ELKP wie folgt:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 15 \\ 10 \end{pmatrix}.$$

Die Nebenbedingung $\mathbf{Ax} \geq \mathbf{c}$ des ELKP erfasst die auf- und absteigenden Wege der nicht-parallelen Kanten für das Beispiel wie folgt:

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \geq \begin{pmatrix} 20 \\ 20 \\ 20 \\ 20 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Für die Komplementaritätsbedingung erhalten wir die Teilmengen $\phi_j = \{\{1,3\},\{2,4\}\}$ mit $j \in \{1,2\}$.

Um das ELKP abschließend zu homogenisieren, führen wir einen nichtnegativen Skalar $\alpha \geq 0$ ein. Entsprechend definieren wir die Matrizen $\mathbf{P}$ und $\mathbf{Q}$ sowie den Vektor $\mathbf{u}$ gemäß der Transformation im Kapitel 3.1. Damit erhalten wir das HELKP des IKWPK, welches wir formal wie folgt beschreiben:

**Definition 3.5 (IKWPK als HELKP).** Seien ein Kreis $G(V,E)$ und ein Distanzgraph $G_d(V,D,\delta)$ mit nichtnegativer Gewichtsfunktion $\delta : D \to \mathbb{Z}^+$ auf $G_d$ sowie der Vektor $\mathbf{d} = (\delta(d_1), \delta(d_2), \ldots, \delta(d_{|D|}))^\mathsf{T}$ mit den Längen der Distanzkanten gegeben. Weiterhin definieren wir die Matrizen

$$\mathbf{P} = \begin{pmatrix} \mathbf{W}^+_{\text{non-par}} & -\mathbf{d}_{\text{non-par}} \\ \mathbf{1}_{|D_{\text{non-par}}| \times |E|} - \mathbf{W}^+_{\text{non-par}} & -\mathbf{d}_{\text{non-par}} \\ \mathbf{I}_{|E|} & \mathbf{0}_{|E| \times 1} \\ \mathbf{0}_{1 \times |E|} & 1 \end{pmatrix} \in \mathbb{Z}^{p \times |E|+1}$$

und

$$\mathbf{Q} = \begin{pmatrix} \mathbf{I}_{|E_{\text{par}}|} & -\mathbf{d}_{\text{par}} \end{pmatrix} \in \mathbb{Z}^{q \times |E|+1}.$$

Zusätzlich definieren wir $|D_{\text{non-par}}|$ Indexmengen $\phi_j = \{i, i + |D_{\text{non-par}}|\}$ mit $i \in \{1,2,\ldots,|D_{\text{non-par}}|\}$. Finde eine nichtnegative $G_d$-erfüllende Gewichtsfunktion $\omega : E \to \mathbb{Z}_0^+$ auf $G(V,E,\omega)$, sodass für den Vektor $\mathbf{u} = (\omega(e_1), \omega(e_2), \ldots, \omega(e_{|E|}), \alpha)^\mathsf{T}$ die Komplementaritätsbedingung

$$\sum_{j=1}^{m} \prod_{i \in \phi_j} (\mathbf{Pu})_i = 0 \tag{47}$$

unter den Nebenbedingungen

$$\mathbf{Pu} \geq \mathbf{0} \tag{48}$$

$$\mathbf{Qu} = \mathbf{0} \tag{49}$$

gilt, wobei $\alpha \geq 0$, und das Gesamtgewicht $\sum_{j=1}^{|E|} \mathbf{u}_j$ des Kreises minimiert wird, oder zeige, dass keine solche Funktion existiert.

Setzen wir das Beispiel aus Abbildung 2 fort, so erhalten wir für die Nebenbedingung $\mathbf{Pu} \geq \mathbf{0}$ im HELKP das folgende System:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & -20 \\ 0 & 1 & 1 & 0 & -20 \\ 0 & 0 & 1 & 1 & -20 \\ 1 & 0 & 0 & 1 & -20 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \alpha \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Die Teilmengen $\phi_j = \{\{1,3\},\{2,4\}\}$ mit $j \in \{1,2\}$ bleiben durch die Homogenisierung unverändert. Für das Gleichungssystem $\mathbf{Qu} = \mathbf{0}$ im HELKP ergibt sich für das Beispiel entsprechend:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & -15 \\ 0 & 0 & 0 & 1 & 10 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \alpha \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Wir haben das IKWPK sowohl als ELKP als auch in seiner homogenen Variante definiert. Im nächsten Unterkapitel setzen wir uns mit deren algorithmischen Lösungsverfahren auseinander.

## 3.3 Algorithmisches Lösungsverfahren für das homogene erweiterte lineare Komplementaritätsproblem

In diesem Unterkapitel beschäftigen wir uns mit der algorithmischen Lösung des IKWPK. Zunächst analysieren wir die strukturellen Eigenschaften der Lösungsmenge des HELKP. Nachfolgend erläutern wir die notwendigen Vorverarbeitungsschritte zur Reduktion der Probleminstanz. In den letzten drei Unterkapiteln erläutern wir den jeweiligen Algorithmus und unsere spezifischen algorithmischen Anpassungen, die wir basierend auf den drei Teilen des algorithmischen Lösungsverfahren von De Schutter und De Moor [DSDM95] entwickeln.

### 3.3.1 Strukturelle Eigenschaften der Lösungsmenge

Die Lösungsmenge des IKWPK als HELKP weist eine besondere Struktur auf, die wir im Folgenden herleiten. Dazu führen wir zunächst grundlegende Begriffe der Polyedertheorie ein. Diese Begriffe benötigen wir sowohl zur Beschreibung der Lösungsmenge als auch zur späteren Erläuterung der Algorithmen. Die ersten beiden Definitionen basieren auf [Sch98] und alle weiteren sind [DSDM95, S. 291] entnommen.

**Definition 3.6.** Eine Menge $H \subset \mathbb{R}^n$ heißt **linearer Halbraum**, wenn $\mathbf{a} \in \mathbb{R}^n \backslash \{\mathbf{0}\}$ existiert, sodass $H = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}^\mathsf{T}\mathbf{x} \geq 0\}$ gilt.

**Definition 3.7.** Eine Menge $H \subset \mathbb{R}^n$ heißt **lineare Hyperebene**, wenn $\mathbf{a} \in \mathbb{R}^n \backslash \{\mathbf{0}\}$ existiert, sodass $H = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{a}^\mathsf{T}\mathbf{x} = 0\}$ gilt.

**Definition 3.8.** Eine Menge $P \subseteq \mathbb{R}^n$ ist ein **Polyeder**, wenn $\mathbf{A} \in \mathbb{R}^{m \times n}$ und $\mathbf{b} \in \mathbb{R}^m$ existieren, sodass $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$. Das heißt, $P$ ist die Lösungsmenge eines endlichen Systems von linearen Ungleichungen.

**Definition 3.9.** Eine Menge $K \subseteq \mathbb{R}^n$ ist ein **polyedrischer Kegel**, wenn $\mathbf{A} \in \mathbb{R}^{m \times n}$ existiert, sodass $K = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{0}\}$. Das heißt, $K$ ist die Lösungsmenge eines endlichen Systems von homogenen linearen Ungleichungen.

**Definition 3.10.** Sei $P$ ein nichtleeres Polyeder. Der **Linearitätsraum** von $P$ ist $\mathrm{Lin}(P) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{0}\}$. Die Basisvektoren von $\mathrm{Lin}(P)$ nennen wir **Zentralstrahlen**. Ist $\mathrm{Lin}(P) = \{\mathbf{0}\}$, dann heißt das Polyeder $P$ **spitz**.

Ein Polyeder bzw. ein polyedrischer Kegel ist also spitz, wenn sein Linearitätsraum nur den Nullvektor enthält. In diesem Fall gilt $n = \mathrm{rg}(\mathbf{A})$, siehe [DSDM95, S. 291]. Visuell können wir uns einen spitzen polyedrischen Kegel im dreidimensionalen Raum als einen Kegel mit flachen Seitenflächen vorstellen, der seine Spitze im Ursprung des Koordinatensystems hat und sich entlang seiner Seitenkanten ins Unendliche ausdehnt. Für weitere geometrische Interpretationen siehe [Sch98] oder [Hoc17]. Die Kanten des spitzen polyedrischen Kegels entsprechen den sogenannten **Extremalstrahlen**, vgl. [DSDM95, S. 291].

Im Allgemeinen wird die Lösungsmenge eines HELKP durch die beiden Nebenbedingungssysteme mit den homogenen linearen Ungleichungen $\mathbf{P}\mathbf{u} \geq \mathbf{0}$ und den homogenen linearen Gleichungen $\mathbf{Q}\mathbf{u} = \mathbf{0}$ bestimmt. Die Lösungsmenge bildet folglich einen polyedrischen Kegel $K$. Mittels des folgenden Theorems zeigen wir, dass dieser polyedrische Kegel für das IKWPK zudem spitz ist.

**Theorem 3.11.** *Die Lösungsmenge des IKWPK enthält keine Zentralstrahlen.*

*Beweis.* Sei $K = \{\mathbf{u} \in \mathbb{Z}_0^{+|E|+1} \mid \mathbf{R}\mathbf{u} \geq \mathbf{0}\}$ der polyedrische Kegel, der die Lösungsmenge des IKWPK beschreibt, wobei $\mathbf{R} = \begin{pmatrix} \mathbf{P} \\ \mathbf{Q} \end{pmatrix}$. Aufgrund der Nichtnegativitätsbedingungen der Kantengewichte enthält die Matrix $\mathbf{P}$ die Einheitsmatrix $\mathbf{I}_{|E|+1}$ als Untermatrix. Somit hat die Matrix $\mathbf{P}$ vollen Spaltenrang, also $\mathrm{rg}(\mathbf{P}) = |E| + 1$. Da $\mathbf{P}$ eine Untermatrix von $\mathbf{R}$ ist, folgt, dass auch $\mathbf{R}$ vollen Spaltenrang besitzt. Betrachten wir nun den Linearitätsraum $L = \{\mathbf{u} \in \mathbb{Z}_0^{+|E|+1} \mid \mathbf{R}\mathbf{u} = \mathbf{0}\}$ des polyedrischen Kegels. Wegen des vollen Spaltenrangs von $\mathbf{R}$ besitzt die Gleichung $\mathbf{R}\mathbf{u} = \mathbf{0}$ ausschließlich die triviale Lösung $\mathbf{u} = \mathbf{0}$. Der Linearitätsraum enthält also nur den Nullvektor. Somit enthält die Lösungsmenge $K$ des IKWPK keine Zentralstrahlen und bildet einen spitzen polyedrischen Kegel. □

Daraus lässt sich ableiten, dass für das IKWPK jeder beliebige Punkt $\mathbf{u} \in K$ als nichtnegative Linearkombination der Extremalstrahlen dargestellt werden kann, vgl. [DSDM95, S. 298]. Sei $\mathcal{E}$ die Menge der Extremalstrahlen von $K$. Dann existieren

nichtnegative Skalare $\kappa_k \geq 0$, sodass $\mathbf{u} = \sum_{\mathbf{e}_k \in \mathcal{E}} \kappa_k \mathbf{e}_k$ eindeutig ist, siehe [DSDM95, S. 298].

Extremalstrahlen, die die Komplementaritätsbedingung nicht erfüllen, können nicht zur Lösung des ELKP beitragen. Dies ermöglicht uns Extremalstrahlen im Algorithmus frühzeitig zu verwerfen. Im Folgenden zeigen wir die Eigenschaft für das IKWPK basierend auf [DSDM95, S. 299 f.].

**Eigenschaft 3.12.** *Falls der Extremalstrahl $\mathbf{e}_l \in \mathcal{E}$ die Komplementaritätsbedingung nicht erfüllt, so gilt für jede Teilmenge $\mathcal{E}_s \subset \mathcal{E}$ mit $\mathbf{e}_l \in \mathcal{E}_s$ und für alle $\kappa_k \geq 0$ mit $\kappa_l > 0$, dass $\mathbf{u} = \sum_{\mathbf{e}_k \in \mathcal{E}_s} \kappa_k \mathbf{e}_k$ die Komplementaritätsbedingung ebenfalls nicht erfüllt.*

*Beweis.* Da der Extremalstrahl $\mathbf{e}_l \in \mathcal{E}$ die Komplementaritätsbedingung nicht erfüllt, gilt $\sum_{j=1}^{m} \prod_{i \in \phi_j} (\mathbf{P}\mathbf{e}_k)_i \neq 0$. Da zudem $\mathbf{P}\mathbf{e}_l \geq 0$ gilt, muss es mindestens eine Indexmenge $\phi_j$ geben, sodass $(\mathbf{P}\mathbf{e}_l)_i \neq 0$ für alle $i \in \phi_j$.
Wir nehmen an, dass $\mathbf{u} = \sum_{\mathbf{e}_l \in \mathcal{E}_s} \kappa_k \mathbf{e}_k$ die Komplementaritätsbedingung erfüllt. Dann folgt

$$\begin{aligned}\sum_{j=1}^{m} \prod_{i \in \phi_j} \Big( \mathbf{P} \Big( \sum_{\mathbf{e}_k \in \mathcal{E}} \kappa_k \mathbf{e}_k \Big) \Big)_i &= \sum_{j=1}^{m} \prod_{i \in \phi_j} \Big( \sum_{\mathbf{e}_k \in \mathcal{E}} \kappa_k (\mathbf{P}\mathbf{e}_k)_i \Big) \\ &= \sum_{j=1}^{m} \prod_{i \in \phi_j} \Big( \underbrace{\kappa_l}_{>0} \underbrace{(\mathbf{P}\mathbf{e}_l)_i}_{\geq 0} + \sum_{\mathbf{e}_k \in \mathcal{E}_s \setminus \{\mathbf{e}_l\}} \underbrace{\kappa_k}_{\geq 0} \underbrace{(\mathbf{P}\mathbf{e}_k)_i}_{\geq 0} \Big) = 0.\end{aligned}$$

Die Komplementaritätsbedingung ist demnach nur erfüllt, wenn für alle $j \in \{1, 2, \ldots, m\}$ mindestens ein $i \in \phi_j$ existiert, sodass

$$\kappa_k (\mathbf{P}\mathbf{e}_l)_i + \sum_{\mathbf{e}_k \in \mathcal{E}_s \setminus \{\mathbf{e}_l\}} \kappa_k (\mathbf{P}\mathbf{e}_k) = 0 \text{ und } (\mathbf{P}\mathbf{e}_l)_i = 0$$

gilt. Dies steht jedoch im Widerspruch zu unserer Annahme, dass $(\mathbf{P}\mathbf{e}_l)_i \neq 0$ für alle $i \in \phi_j$. Daher folgt, dass unsere Annahme falsch ist und $\mathbf{u}$ die Komplementaritätsbedingung ebenfalls nicht erfüllt. □

Im Folgenden betrachten wir eine weitere zentrale Eigenschaft der Lösungsmenge des HELKP. Wir zeigen, dass jedes positive skalare Vielfache einer Lösung wiederum eine Lösung ist, siehe [DSDM95, S. 299].

**Eigenschaft 3.13.** *Ist $\mathbf{u} \in \mathbb{Z}_0^{+n}$ eine Lösung des HELKP, dann ist $\kappa\,\mathbf{u}$ für alle $\kappa \geq 0$ ebenfalls eine Lösung des HELKP.*

*Beweis.* Da $\mathbf{u}$ eine Lösung des HELKP ist, gilt $\mathbf{P}\mathbf{u} \geq \mathbf{0}$ und $\mathbf{Q}\mathbf{u} = \mathbf{0}$. Wegen der Linearität und da $\kappa \geq 0$ gilt, folgt $\mathbf{P}(\kappa\,\mathbf{u}) = \kappa\,(\mathbf{P}\mathbf{u}) \geq \mathbf{0}$. Ebenso gilt $\mathbf{Q}(\kappa\,\mathbf{u}) = \kappa\,(\mathbf{Q}\mathbf{u}) = \kappa \cdot \mathbf{0} = \mathbf{0}$. Wir bezeichnen die Anzahl der Teilmengen $\phi_j$ in der Komplementaritätsbedingung mit $\#\phi_j$. Die folgende Gleichung zeigt, dass die Komplementaritätsbedingung ebenfalls weiterhin erfüllt ist:

$$\sum_{j=1}^{m} \prod_{i \in \phi_j} (\mathbf{P}(\kappa\,\mathbf{u}))_i = \sum_{j=1}^{m} \prod_{i \in \phi_j} \kappa\,(\mathbf{P}\mathbf{u})_i = \sum_{j=1}^{m} \kappa^{\#\phi_j} \prod_{i \in \phi_j} (\mathbf{P}\mathbf{u})_i = \sum_{j=1}^{m} \kappa^{\#\phi_j} \cdot 0 = 0.$$

Der letzte Zwischenausdruck ist gleich null, da $\mathbf{u}$ als Lösung des HELKP die partielle Komplementaritätsbedingung $\prod_{i\in\phi_j}(\mathbf{Pu})_i = 0$ für alle $j \in \{1,\dots,m\}$ erfüllt. Somit ist $\kappa\,\mathbf{u}$ eine Lösung des HELKP. □

Um eine Lösung des IKWPK als HELKP zu bestimmen, erzeugen wir schrittweise eine Menge von Extremalstrahlen und übernehmen folglich jeweils nur diejenigen, welche die (partielle) Komplementaritätsbedingung erfüllen. Eine detaillierte Beschreibung des Algorithmus folgt ab Kapitel 3.3.3. Zuvor erläutern wir die Vorverarbeitungsschritte der Probleminstanzen.

### 3.3.2 Vorverarbeitung der Probleminstanzen

Der Algorithmus zur Lösung des IKWPK ist in mehrere Schritte unterteilt, deren Grundidee wir in den folgenden Kapiteln skizzieren. Zunächst betrachten wir die zugrundeliegende Probleminstanz, die als Eingabe für den Algorithmus dient. Dabei folgen wir dem Ansatz aus [DSDM95, S. 300], wonach die Nebenbedingung $\mathbf{Pu} \geq \mathbf{0}$ in zwei Systeme aufgeteilt wird, das heißt

$$\mathbf{P}_1\mathbf{u} \geq \mathbf{0} \text{ und } \mathbf{P}_2\mathbf{u} \geq \mathbf{0}.$$

Die Matrix $\mathbf{P}$ wird in die beiden Matrizen $\mathbf{P}_1$ und $\mathbf{P}_2$ unterteilt. Dabei enthält $\mathbf{P}_1$ alle Ungleichungen, die Teil der Komplementaritätsbedingung sind, während $\mathbf{P}_2$ alle übrigen Bedingungen umfasst. Für das IKWPK als HELKP, siehe Definition 3.5, erhalten wir somit die folgende Aufteilung der Matrix $\mathbf{P}$:

$$\mathbf{P}_1 = \begin{pmatrix} \mathbf{W}^+_{\text{non-par}} & -\mathbf{d}_{\text{non-par}} \\ \mathbf{1}_{|D_{\text{non-par}}|\times|E|} - \mathbf{W}^+_{\text{non-par}} & -\mathbf{d}_{\text{non-par}} \end{pmatrix} \in \mathbb{Z}^{p_1\times|E|+1}$$

und

$$\mathbf{P}_2 = \begin{pmatrix} \mathbf{I}_{|E|} & \mathbf{0}_{|E|\times 1} \\ \mathbf{0}_{1\times|E|} & 1 \end{pmatrix} \in \mathbb{Z}_0^{+\,p_2\times|E|+1}.$$

Die Indexmengen wählen wir nun so, dass $\phi_j \subseteq \{1,\dots,p_1\}$ gilt. Die Nebenbedingung $\mathbf{Qu} = \mathbf{0}$ bleibt zunächst unverändert.

Wir nehmen im Folgenden mehrere Vorverarbeitungssehritte an der Probleminstanz vor, um die Komplexität dieser zu reduzieren und die Effizienz des Algorithmus zu verbessern. Hierfür gehen wir nicht auf die vollständige Implementierung des Algorithmus ein, sondern nur auf einzelne Anpassungen, die für die Vorverarbeitung relevant sind.

**Beschränkung der Eingabe auf Distanzgraphen ohne isolierte Knoten.** Ohne Beschränkung der Allgemeinheit grenzen wir die Eingabe auf Distanzgraphen ohne isolierte Knoten ein, sodass jeder Knoten zu mindestens einer Distanzkante inzident ist. Im Gegensatz zu Distanzgraphen mit isolierten Knoten ergibt sich das Kantengewicht in der Lösung somit aus einzelnen Kreiskante, anstatt als variable Summe der Gewichte mehrerer Kanten.

**Festlegen der Kreiskantengewichte von parallelen Distanzkanten.** Existieren parallele Distanzkanten, so sind die zugehörigen Kreiskantengewichte eindeutig bestimmt, siehe Theorem 3.3. Daher setzen wir die entsprechenden Gewichte in einem Vorverarbeitungsschritt direkt im Lösungsvektor $\mathbf{u}$ fest. Da wir Variablen vorab festlegen, reduziert sich deren Anzahl in der Probleminstanz und die zugehörigen Spalten in den Nebenbedingungen fallen weg. Ferner entfallen die entsprechenden Gleichungen in der Matrix $\mathbf{Q}$ und müssen im Gleichungssystem $\mathbf{Qu} = \mathbf{0}$ im Algorithmus 2 nicht weiter berücksichtigt werden.

**Klassifikation von Distanzkanten.** Durch einen paarweisen Vergleich von Distanzkanten können wir für einige davon bestimmen, ob sie aufsteigend oder absteigend sind. Dies basiert auf der Analyse der ihr zugehörigen aufsteigenden Wege im Kreis: Entweder ist einer der Wege vollständig im anderen enthalten oder sie überschneiden sich zum Teil oder sie verlaufen getrennt voneinander. Berücksichtigen wir zusätzlich die Längen der Distanzkanten, dann lässt sich die folgende Eigenschaft, vgl. [Sau23, S. 24], ableiten:

**Eigenschaft 3.14.** *Seien $d_1$ und $d_2$ Distanzkanten im Distanzgraphen $G_d$ mit $\delta(d_1) \geq \delta(d_2)$. Zudem seien $\mathbf{W}^+_{d_1\cdot}$ und $\mathbf{W}^+_{d_2\cdot}$ die zugehörigen aufsteigenden binären Wegevektoren. Wir bezeichnen mit $\circ$ das komponentenweise Hadamard-Produkt. Dann gilt:*

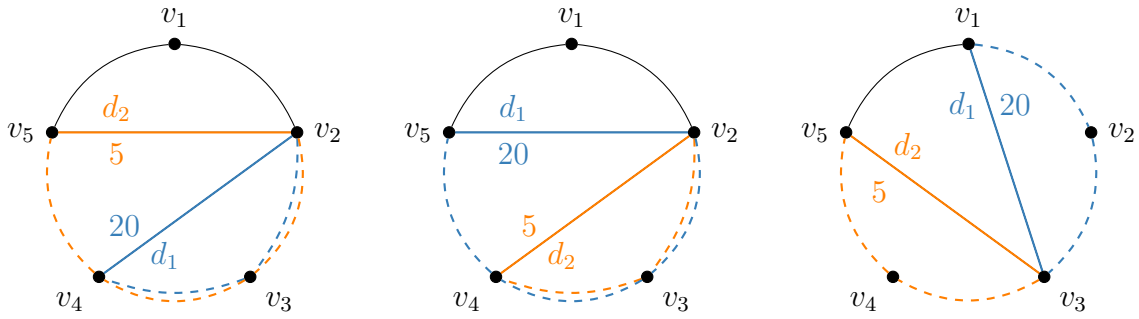
1. *Falls $||\mathbf{W}^+_{d_1\cdot} \circ \mathbf{W}^+_{d_2\cdot}||_1 = ||\mathbf{W}^+_{d_1\cdot}||_1$, dann ist der aufsteigende Weg von $d_1$ vollständig im aufsteigenden Weg von $d_2$ enthalten. Folglich ist die Distanzkante $d_2$ absteigend.*

2. *Falls $||\mathbf{W}^+_{d_1\cdot} \circ \mathbf{W}^+_{d_2\cdot}||_1 = ||\mathbf{W}^+_{d_2\cdot}||_1$, dann ist der aufsteigende Weg von $d_2$ vollständig im aufsteigenden Weg von $d_1$ enthalten. Folglich ist die Distanzkante $d_2$ aufsteigend.*

3. *Falls $||\mathbf{W}^+_{d_1\cdot} \circ \mathbf{W}^+_{d_2\cdot}||_1 = 0$, dann verlaufen die aufsteigende Wege der Distanzkanten $d_1$ und $d_2$ getrennt voneinander. Folglich ist die Distanzkante $d_2$ aufsteigend.*

In Abbildung 3 sind exemplarisch jeweils ein Beispiel für die drei beschriebenen Fälle der Eigenschaft 3.14 dargestellt.

In allen weiteren Fällen lässt sich keine Aussage über die Distanzkanten treffen. Für jede klassifizierbare Distanzkante ist der kürzeste Weg im Kreis $G$ eindeutig bestimmt. Unter der Annahme, dass wir eine aufsteigende Distanzkante identifiziert haben, fügen wir der Matrix $\mathbf{Q}$ die entsprechende Gleichung für den aufsteigenden Weg hinzu. Gleichzeitig entfernen wir beide zu der Distanzkante zugehörigen Ungleichungen aus der Matrix $\mathbf{P}_1$ und damit auch aus der Komplementaritätsbedingung. Die Ungleichung für den absteigenden Weg ist automatisch erfüllt, da dieser gemäß des Kantengewichtsvergleichs mindestens genauso lang ist wie der aufsteigende Weg. Bei absteigenden Distanzkanten gehen wir analog vor. Dadurch reduziert sich die Komplexität der Probleminstanz insbesondere in der Nebenbedingung $\mathbf{P}_1\mathbf{u} \geq \mathbf{0}$.




**(1)** $d_2$ ist absteigend. **(2)** $d_2$ ist aufsteigend. **(3)** $d_2$ ist aufsteigend.

**Abbildung 3:** Beispielhafte Klassifikation von Distanzkanten im Distanzgraphen $G_d$. (1) Der aufsteigende Weg der Distanzkante mit geringerem Gewicht ist vollständig im aufsteigenden Weg der Distanzkante mit größerem Gewicht enthalten, (2) der aufsteigende Weg der Distanzkante mit größerem Gewicht ist vollständig im aufsteigenden Weg der Distanzkante mit kleinerem Gewicht enthalten und (3) die aufsteigenden Wege verlaufen getrennt voneinander. Die Distanzkante $d_1$ mit dem größerem Gewicht von 20 ist blau und die Distanzkante $d_1$ mit dem kleinerem Gewicht von 5 orange markiert. Die aufsteigenden Wege der Distanzkanten sind gestrichelt dargestellt.

**Umstrukturierung von $\mathbf{P}_1$ und der Indexmengen $\phi_j$.** Die Menge der Extremalstrahlen wird im Algorithmus 1 in jeder Iteration um Extremalstrahlen, die die partielle Komplementaritätsbedingung erfüllen, erweitert. In der $k$-ten Iteration wird die partielle Komplementaritätsbedingung für alle $\phi_j$ überprüft, deren Indizes vollständig in der Menge $\{1, \ldots, k\}$ enthalten sind. In der aktuellen Konstruktion von $\mathbf{P}_1$ befinden sich alle Ungleichungen zu den aufsteigenden Wegen in einem Block und anschließend alle Ungleichungen zu den absteigenden Wege in einem weiteren Block. Dabei gilt für die Indexmengen $\phi_j = \{i, i + |D|_{\text{non-par}}\}$. Diese Struktur hat zur Folge, dass die erste partielle Komplementaritätsbedingung erst ab der $(|D_{\text{non-par}}|+1)$-ten Zeile überprüft wird und Extremalstrahlen erst dann als ungültig erkannt und verworfen werden. Um die Anzahl der gültigen Extremalstrahlen über den Algorithmus 1 hinweg möglichst klein zu halten, ordnen wir die Zeilen in $\mathbf{P}_1$ so um, dass jeweils zwei konsekutive Zeilen die Ungleichungen für den aufsteigenden und absteigenden Weg einer Distanzkante enthalten. Dadurch wird die erste partielle Komplementaritätsbedingung bereits ab der zweiten Iteration überprüft und somit werden ungültige Extremalstrahlen deutlich früher verworfen.

**Initialisierung der Extremalstrahlen eliminiert $\mathbf{P}_2$.** Da die Lösungsmenge des IKWPK keine Zentralstrahlen enthält, siehe Theorem 3.11, initialisieren wir die Menge der Extremalstrahlen $\mathcal{E}$ nicht als leere Menge. Stattdessen setzen wir $\mathcal{E} = \{\mathbf{e}_i = (I_{|E|+1})_{\cdot i} \text{ für } i = 1, \ldots, |E|+1\}$, vgl. [DSDM95, S. 317]. Im Algorithmus werden die initialisierten Extremalstrahlen auf die Komplementaritätsbedingung geprüft. Da dies äquivalent zu den durch die Matrix $\mathbf{P}_2$ gegebene Ungleichungen ist und wir $\mathbf{P}_2$ somit implizit berücksichtigen, schließen wir diese von der weiteren Verarbeitung aus. Dadurch reduzieren wir ebenfalls die Größe der Probleminstanz.

Durch die verschiedenen Schritte in der Vorverarbeitung reduzieren wir die Komplexität unserer Probleminstanz, bevor wir die Algorithmen zur Lösungsfindung anwenden. In den folgenden drei Unterkapiteln erläutern wir die einzelnen Module des Gesamtalgorithmus und zeigen, wie diese zur Lösung des IKWPK als HELKP eingesetzt werden.

### 3.3.3 Modul 1: Lösen der Nebenbedingung $\mathbf{P}_1\mathbf{u} \geq \mathbf{0}$ des IKWPK

Im Folgenden erläutern wir den ersten von drei Teilen des Lösungsalgorithmus für das IKWPK. Der Algorithmus 1 basiert auf der *Double Description Method*, siehe [FP96]. Ziel der Methode ist es, aus der H-Darstellung eines Polyeders – also der Darstellung als Schnittmenge von Halbräumen – schrittweise die zugehörige V-Darstellung zu bestimmen. Diese wird für das IKWPK als HELKP durch Extremalstrahlen beschrieben. Initial ist die Lösungsmenge jedoch als H-Darstellung über die Nebenbedingungen des IKWPK gegeben, aus der wir schrittweise die zugehörige V-Darstellung mittels des Lösungsalgorithmus ermitteln. Der Algorithmus 1 löst hierbei das Ungleichungssystem $\mathbf{P}_1\mathbf{u} \geq \mathbf{0}$. Da die Lösungsmenge des IKWPK keine Zentralstrahlen enthält, siehe Theorem 3.11, lassen wir im Vergleich zum Algorithmus 1 in [DSDM95, S. 301 f.] alle Schritte mit Zentralstrahlen aus und passen die Initialisierung wie im Folgenden beschrieben an.

**Initialisierung** (Z. 1-2)
Zu Beginn wird die Menge der Extremalstrahlen durch $\mathcal{E} = \{\mathbf{e}_i = (I_n)_{\cdot i} \text{ für } i = 1, \ldots, n\}$ initialisiert. Somit besteht $\mathcal{E}$ initial aus den Einheitsvektoren von $\mathbb{R}^n$. Außerdem wird die Matrix $\mathbf{P}_{\text{nec}}$ initial auf die Einheitsmatrix $\mathbf{I}_n$ gesetzt. Die Matrix $\mathbf{P}_{\text{nec}}$ enthält im $k$-ten Iterationsschritt alle Ungleichungen die das aktuelle Polyeder beschreiben. Wir nutzen $\mathbf{P}_{\text{nec}}$ im Algorithmus insbesondere zum Überprüfen, ob zwei Extremalstrahlen adjazent zueinander sind.

**Iterationsschritte** (Z. 3-22)
Nach der Initialisierung ziehen wir in jedem Iterationsschritt $k$ die $k$-te Ungleichung aus der Matrix $\mathbf{P}_1$ heran. Diese Ungleichung definiert einen weiteren Halbraum, welcher den aktuellen Lösungsraum weiter einschränkt. Zunächst klassifizieren wir die bereits vorhandenen Extremalstrahlen anhand derer Residuen. Für jeden Extremalstrahl $\mathbf{e}$ berechnen wir das Residuum $\text{res}(\mathbf{e}) = (\mathbf{P}_1)_k.\mathbf{e}$. Anhand des jeweiligen Werts fügen wir den Extremalstrahl einer der folgenden drei Mengen hinzu:

- $\mathcal{E}^+$: Diese Menge enthält die Extremalstrahlen mit $\text{res}(\mathbf{e}) > 0$. Die Extremalstrahlen liegen innerhalb des neuen Halbraums. Wir übernehmen sie in die Menge der Extremalstrahlen $\mathcal{E}$, wenn sie die partielle Komplementaritätsbedingung erfüllen.
- $\mathcal{E}^-$: Diese Menge enthält die Extremalstrahlen mit $\text{res}(\mathbf{e}) < 0$. Die Extremalstrahlen befinden sich außerhalb des neuen Halbraums und werden verworfen. Falls sie jedoch adjazent zu Extremalstrahlen aus $\mathcal{E}^+$ sind, nutzen wir diese zur Erzeugung neuer Extremalstrahlen.
- $\mathcal{E}^0$: Diese Menge enthält die Extremalstrahlen mit $\text{res}(\mathbf{e}) = 0$. Die Extremalstrahlen liegen auf der Grenze des Halbraums und erfüllen die Komplementa-

**Algorithmus 1:** Löse das Ungleichungssystem $\mathbf{P}_1\mathbf{u} \geq \mathbf{0}$ des IKWPK als HELKP, vgl. [DSDM95, Algorithmus 1]

**Eingabe:** $m$, $p_1$, $n$, $\{\phi_j\}_{j=1}^{m}$, $\mathbf{P}_1 \in \mathbb{R}^{p_1 \times n}$

$\mathcal{E} \leftarrow \{\mathbf{e}_i = (I_n)_{\cdot i} \text{ für } i = 1, \ldots, n\}$
$\mathbf{P}_{\text{nec}} \leftarrow I_n$
**für** $k = 1, 2, \ldots, p_1$ **tue**
  $\forall\, \mathbf{e} \in \mathcal{E}:\ \text{res}(\mathbf{e}) \leftarrow (\mathbf{P}_1)_{k\cdot}\mathbf{e}$
  $\mathcal{E}^+ \leftarrow \{\mathbf{e} \in \mathcal{E} \,|\, \text{res}(\mathbf{e}) > 0\}$
  $\mathcal{E}^- \leftarrow \{\mathbf{e} \in \mathcal{E} \,|\, \text{res}(\mathbf{e}) < 0\}$
  $\mathcal{E}^0 \leftarrow \{\mathbf{e} \in \mathcal{E} \,|\, \text{res}(\mathbf{e}) = 0\}$
  $\mathcal{E} \leftarrow \mathcal{E}^0 \cup \{\mathbf{e} \in \mathcal{E}^+ \,|\, \mathbf{e} \text{ erfüllt die partielle Komplementaritätsbedingung}\}$
  **wenn** $\mathcal{E}^- = \emptyset$ **dann**
    `/* Fall 1: Die k-te Ungleichung ist redundant. */`
  **Ende**
  **sonst**
    `/* Fall 2: Erzeuge neue Extremalstrahlen. */`
    **für** *alle Paare* $(\mathbf{e}^+, \mathbf{e}^-) \in \mathcal{E}^+ \times \mathcal{E}^-$ **tue**
      **wenn** $\mathbf{e}^+$ *und* $\mathbf{e}^-$ *adjazent sind* **dann**
        $\mathbf{e}^{\text{new}} \leftarrow \text{res}(\mathbf{e}^+)\,\mathbf{e}^- - \text{res}(\mathbf{e}^-)\,\mathbf{e}^+$
        **wenn** $\mathbf{e}^{new}$ *die partielle Komplementaritätsbedingung erfüllt* **dann**
          $\mathcal{E} \leftarrow \mathcal{E} \cup \{\mathbf{e}^{\text{new}}\}$
        **Ende**
      **Ende**
    **Ende**
    Füge die $k$-te Zeile von $\mathbf{P}_1$ zu $\mathbf{P}_{\text{nec}}$ hinzu.
  **Ende**
**Ende**

**Ausgabe:** $\mathcal{E}$, $\mathbf{P}_{\text{nec}}$

ritätsbedingung. Wir übernehmen sie in die Menge der Extremalstrahlen $\mathcal{E}$.

In der $k$-ten Iteration erfüllt die partielle Komplementaritätsbedingung die Gleichung

$$\prod_{i \in \phi_j} (\mathbf{P}_1 \mathbf{u})_i = 0 \text{ für alle } j \in \{1, 2, \ldots, m\}, \text{ sodass } \phi_j \subset \{1, 2, \ldots, k\}.$$

Existiert keine Indexmenge $\phi_j$, die in $\{1, 2, \ldots, k\}$ enthalten ist, so gilt die partielle Komplementaritätsbedingung per Definition als erfüllt, vgl. [DSDM95, S. 303]. Die folgende Eigenschaft aus [DSDM95, Eigenschaft 4.1.] zeigt, dass wir die partielle Komplementaritätsbedingung ausschließlich für Extremalstrahlen in $\mathcal{E}^+$ sowie neu erzeugte Extremalstrahlen prüfen müssen.

**Eigenschaft 3.15.** *Ist $\mathbf{e} \in \mathcal{E}^0$ in Iteration $k$ enthalten und erfüllt in der Iteration $k-1$ die partielle Komplementaritätsbedingung, dann erfüllt $\mathbf{e}$ die partielle Komplementaritätsbedingung auch in Iteration $k$.*

*Beweis.* Ist $\mathbf{e} \in \mathcal{E}^0$ in Iteration $k$ enthalten, dann gilt $\text{res}(\mathbf{e})(\mathbf{P}_1)_{k\cdot}\mathbf{e} = (\mathbf{P}_1\mathbf{e})_{k\cdot} = 0$. Daraus folgt für alle Indexmengen $\phi_j$ mit $k \in \phi_j$, dass $\prod_{i\in\phi_j}(\mathbf{P}_1\mathbf{u})_i = 0$ gilt. Andererseits erfüllt $\mathbf{e}$ die partielle Komplementaritätsbedingung in Iteration $k-1$. Daher gilt $\prod_{i\in\phi_j}(\mathbf{P}_1\mathbf{u})_i = 0$ für alle Indexmengen $\phi_j \subset \{1, 2, \ldots, k-1\}$. Hieraus schließen wir, dass die partielle Komplementaritätsbedingung $\prod_{i\in\phi_j}(\mathbf{P}_1\mathbf{u})_i = 0$ für alle Indexmengen $\phi_j$ mit $\phi_j \subset \{1, 2, \ldots, k\}$ erfüllt ist. Somit erfüllt $\mathbf{e}$ auch in Iteration $k$ die partielle Komplementaritätsbedingung. □

Folglich überprüfen wir alle Extremalstrahlen $\mathbf{e} \in \mathcal{E}^+$ darauf, ob sie die partielle Komplementaritätsbedingung erfüllen. Ist dies der Fall, fügen wir sie der Menge der Extremalstrahlen $\mathcal{E}$ hinzu. Anhand der zuvor definierten Extremalstrahlenmengen unterscheiden wir im Algorithmus anschließend zwischen zwei Fällen:

**Fall 1: Redundante Ungleichung** (Z. 9-10)
Falls $\mathcal{E}^-$ leer ist, dann ist die $k$-te Ungleichung redundant.

**Fall 2: Erzeuge neue Extremalstrahlen** (Z. 11-21)
Falls $\mathcal{E}^-$ nicht leer ist, dann existieren Extremalstrahlen, die die $k$-te Ungleichung nicht erfüllen. Wir nutzen die Extremalstrahlen aus $\mathcal{E}^-$ und $\mathcal{E}^+$, um neue Extremalstrahlen zu erzeugen. Um Redundanzen in der Lösungsmenge zu vermeiden, nutzen wir dafür adjazente Extremalstrahlen, die wir mithilfe der Matrix $\mathbf{P}_{\text{nec}}$ bestimmen.

Um die Adjazenz zwischen zwei Extremalstrahlen festzustellen, greifen wir auf zwei verschiedene Tests zurück, die auf den Mengen der Nullindizes $\mathcal{I}_0(\mathbf{e})$ der beiden Extremalstrahlen basieren. Für einen Extremalstrahl $\mathbf{e} \in \mathcal{E}$ definieren wir die Menge der Nullindizes als $\mathcal{I}_0(\mathbf{e}) = \{i \mid (\mathbf{P}_1\mathbf{e})_i = 0\}$ [DSDM95, S. 304]. Diese berechnen wir für die auf Adjazenz zu testenden Extremalstrahlen. Die beiden Tests beruhen jeweils auf den folgenden Eigenschaften, siehe [DSDM95, Eigenschaft 4.2] und [DSDM95, Eigenschaft 4.3]:

**Eigenschaft 3.16** (Adjazenztest 1)**.** *Seien* $\mathbf{e}_1, \mathbf{e}_2 \in \mathcal{E}$ *zwei Extremalstrahlen eines spitzen polyedrischen Kegels der Dimension* $n$*. Falls* $\mathbf{e}_1$ *und* $\mathbf{e}_2$ *adjazent sind, dann gilt:*

$$|\mathcal{I}_0(\mathbf{e}_1) \cap \mathcal{I}_0(\mathbf{e}_2)| \geq n - 2.$$

Diesen Adjazenztest setzen wir ein, da er insbesondere bei einer großen Menge an Extremalstrahlen $\mathcal{E}$ deutlich effizienter als der zweite Test ist. Dieser überprüft, dass neben den Extremalstrahlen $e_1$ und $e_2$ keine weiteren Extremalstrahlen in der zugehörigen minimalen Fläche enthalten sind.

**Eigenschaft 3.17** (Adjazenztest 2)**.** *Seien* $\mathbf{e}_1, \mathbf{e}_2 \in \mathcal{E}$ *zwei Extremalstrahlen. Falls* $\mathbf{e}_1$ *und* $\mathbf{e}_2$ *adjazent sind, dann existiert kein* $\mathbf{e} \in \mathcal{E}$*, sodass* $\mathcal{I}_0(\mathbf{e}) \subset \mathcal{I}_0(\mathbf{e}_1) \cap \mathcal{I}_0(\mathbf{e}_2)$ *gilt.*

Zwar ist der zweite Adjazenztest im Allgemeinen hinreichend, kann in unserem Fall jedoch auch von nicht adjazenten Extremalstrahlen positiv durchlaufen werden, wenn in vorherigen Iterationen entsprechende Extremalstrahlen verworfen wurden, die die partielle Komplementaritätsbedingung nicht erfüllen. Daher stellen wir sicher, dass alle neu erzeugten Extremalstrahlen die partielle Komplementaritätsbedingung erfüllen, bevor wir sie in die Menge der Extremalstrahlen $\mathcal{E}$ aufnehmen. Somit sind beide Tests im HELKP notwendige, aber keine hinreichenden Bedingungen für die Adjazenz zweier Extremalstrahlen.

Passieren zwei Extremalstrahlen $\mathbf{e}^-$ und $\mathbf{e}^+$ beide Adjazenztests, dann erzeugen wir einen neuen Extremalstrahl $\mathbf{e}^{\text{new}} = \operatorname{res}(\mathbf{e}^+)\,\mathbf{e}^- - \operatorname{res}(\mathbf{e}^-)\,\mathbf{e}^+$. Diese Konstruktion können wir anhand der folgenden Bemerkung aus [DSDM95, S. 302] nachvollziehen.

*Bemerkung* 3.18. Seien $\mathbf{e}_1$ und $\mathbf{e}_2$ zwei Extremalstrahlen im $k$-ten Iterationsschritt, für die $\operatorname{res}(\mathbf{e}_1)\operatorname{res}(\mathbf{e}_2) < 0$ gilt. Dann erfüllt der neu erzeugte Extremalstrahl $\mathbf{e}^{\text{new}} = |\operatorname{res}(\mathbf{e}_1)|\mathbf{e}_2 + |\operatorname{res}(\mathbf{e}_2)|\mathbf{e}_1$ die Gleichung $(\mathbf{P}_1)_{k\cdot}\,\mathbf{e}^{\text{new}} = 0$.

*Beweis.* O.B.d.A. seien $\mathbf{e}_1 \in \mathcal{E}^+$ und $\mathbf{e}_2 \in \mathcal{E}^-$, sodass $\operatorname{res}(\mathbf{e}_1) > 0$ und $\operatorname{res}(\mathbf{e}_2) < 0$ gelte. Dann erhalten wir

$$\begin{aligned}
(\mathbf{P}_1)_{k\cdot}\,\mathbf{e}^{\text{new}} &= (\mathbf{P}_1)_{k\cdot}\,(\,|\operatorname{res}(\mathbf{e}_1)|\,\mathbf{e}_2 + |\operatorname{res}(\mathbf{e}_2)|\,\mathbf{e}_1) \\
&= (\mathbf{P}_1)_{k\cdot}\,(\operatorname{res}(\mathbf{e}_1)\,\mathbf{e}_2 - \operatorname{res}(\mathbf{e}_2)\,\mathbf{e}_1\,) \\
&= \operatorname{res}(\mathbf{e}_1)\,(\mathbf{P}_1)_{k\cdot}\,\mathbf{e}_2 - \operatorname{res}(\mathbf{e}_1)\,(\mathbf{P}_1)_{k\cdot}\mathbf{e}_1 \\
&= \operatorname{res}(\mathbf{e}_1)\operatorname{res}(\mathbf{e}_2) - \operatorname{res}(\mathbf{e}_1)\operatorname{res}(\mathbf{e}_2) \\
&= 0.
\end{aligned}$$

□

Erfüllt der neu erzeugte Extremalstrahl die partielle Komplementaritätsbedingung, dann übernehmen wir diesen in die Menge der Extremalstrahlen $\mathcal{E}$. Wurde der neue Extremalstrahl aus zwei nicht adjazenten Extremalstrahlen gebildet, dann kann er die partielle Komplementaritätsbedingung nicht erfüllen und wird an dieser Stelle verworfen, siehe [DSDM95, S. 305].

Ist $\mathcal{E}$ am Ende des $k$-ten Iterationsschritt leer, dann hat das HELKP nur die triviale Lösung $\mathbf{u} = \mathbf{0}$.

Als finale Ausgabe des Algorithmus 1 erhalten wir die Menge der Extremalstrahlen $\mathcal{E}$, die die Nebenbedingung $\mathbf{P}_1\mathbf{u} \geq \mathbf{0}$ erfüllen, sowie die aktualisierte Matrix $\mathbf{P}_{\text{nec}}$. Im nächsten Schritt des Gesamtalgorithmus berücksichtigen wir zusätzlich die Nebenbedingung $\mathbf{Qu} = \mathbf{0}$, um die Lösungsmenge weiter einzuschränken.

### 3.3.4 Modul 2: Lösen der Nebenbedingung $\mathbf{Qu} = \mathbf{0}$ des IKWPK

Der Algorithmus 2 löst das Gleichungssystem $\mathbf{Qu} = \mathbf{0}$ und wird nur dann ausgeführt, wenn die Matrix $\mathbf{Q}$ nicht leer ist. Dies ist dann der Fall, wenn Distanzkanten in der Vorverarbeitung der Problemdistanz, wie im Kapitel 3.3.2 beschrieben, durch den Kantengewichtsvergleich als aufsteigend oder absteigend klassifiziert wurden. Dann führen wir den Algorithmus 2 aus.

**Algorithmus 2:** Löse das Gleichungssystem $\mathbf{Qu} = \mathbf{0}$ des IKWPK als HELKP, vgl. [DSDM95, Algorithmus 2]

**Eingabe:** $m$, $p_1$, $q$, $n$, $\{\phi_j\}_{j=1}^{m}$, $\mathbf{P}_1 \in \mathbb{R}^{p_1 \times n}$, $\mathbf{Q} \in \mathbb{R}^{q \times n}$, $\mathcal{E}$, $\mathbf{P}_{\text{nec}}$

**für** $k = 1, 2, \ldots, q$ **tue**
  $\forall\, \mathbf{e} \in \mathcal{E}:\ \text{res}(\mathbf{e}) \leftarrow (\mathbf{Q})_k.\mathbf{e}$
  $\mathcal{E}^+ \leftarrow \{\mathbf{e} \in \mathcal{E} \,|\, \text{res}(\mathbf{e}) > 0\}$
  $\mathcal{E}^- \leftarrow \{\mathbf{e} \in \mathcal{E} \,|\, \text{res}(\mathbf{e}) < 0\}$
  $\mathcal{E}^0 \leftarrow \{\mathbf{e} \in \mathcal{E} \,|\, \text{res}(\mathbf{e}) = 0\}$
  **wenn** $\mathcal{E}^+ = \emptyset$ **und** $\mathcal{E}^- = \emptyset$ **dann**
    `/* Fall 1: Die k-te Gleichung ist redundant. */`
  **Ende**
  **sonst**
    `/* Fall 2: Erzeuge neue Extremalstrahlen. */`
    $\mathcal{E} \leftarrow \mathcal{E}^0$
    **für** *alle Paare* $(\mathbf{e}^+, \mathbf{e}^-) \in \mathcal{E}^+ \times \mathcal{E}^-$ **tue**
      **wenn** $\mathbf{e}^+$ *und* $\mathbf{e}^-$ *adjazent sind* **dann**
        $\mathbf{e}^{\text{new}} \leftarrow \text{res}(\mathbf{e}^+)\,\mathbf{e}^- - \text{res}(\mathbf{e}^-)\,\mathbf{e}^+$
        **wenn** $\mathbf{e}^{new}$ *die Komplementaritätsbedingung erfüllt* **dann**
          $\mathcal{E} \leftarrow \mathcal{E} \cup \{\mathbf{e}^{\text{new}}\}$
        **Ende**
      **Ende**
    **Ende**
  **Ende**
**Ende**

**Ausgabe:** $\mathcal{E}$

Die Struktur des Algorithmus 2 basiert auf dem im vorherigen Abschnitt 3.3.3 beschriebenen Algorithmus 1, unterscheidet sich aber in mehreren Merkmalen. Während wir im Algorithmus 1 zunächst sowohl alle Extremalstrahlen mit positiven Residuum als auch die mit Residuum gleich null übernehmen, übernehmen wir im

Algorithmus 2 nur Letztere in die Menge der Extremalstrahlen. Diese Änderung ergibt sich daraus, dass in der nun zu lösenden Nebenbedingung Gleichungen statt Ungleichungen vorliegen. Zudem testen wir neu erzeugte Extremalstrahlen auf die Erfüllung der gesamten statt der partiellen Komplementaritätsbedingung. Den ersten Adjazenztest passen wir ebenfalls an. Zwei Extremalstrahlen sind nicht adjazent, wenn sie in der $k$-ten Iteration weniger als $n-(k-1)-2$ gemeinsame Indizes in ihren Mengen der Nullindizes aufweisen. Diese Anpassung ist notwendig, da wir die Matrix $\mathbf{P}_{\text{nec}}$ nicht mehr erweitern, da jeder Extremalstrahl $\mathbf{e}$ nach der $k$-ten Iteration die Gleichung $\mathbf{Qu} = \mathbf{0}$ bereits erfüllt, siehe [DSDM95, S. 306 ff.].

Im Gegensatz zur Implementierung von [DSDM95] wenden wir den Algorithmus 2 ausschließlich auf das Gleichungssystem $\mathbf{Qu} = \mathbf{0}$ und nicht auf die Nebenbedingung $\mathbf{P}_2\mathbf{u} \geq \mathbf{0}$ an. Diese berücksichtigen wir in unser Probleminstanz nicht, da keine Matrix $\mathbf{P}_2$ existiert. Diese Eigenschaft der Probleminstanz haben wir bereits im Unterkapitel 3.3.2 aufgezeigt.

Die Algorithmen 1 und 2 bestimmen einen spitzen polyedrischen Kegel, dessen V-Darstellung durch die Menge der Extremalstrahlen $\mathcal{E}$ gegeben ist. Eine Lösung des IKWPK als HELKP kann als nichtnegative Linearkombination der Extremalstrahlen dargestellt werden, jedoch führt nicht jede beliebige Teilmenge von $\mathcal{E}$ zu einer gültigen Lösung des HELKP. Um die gültigen Teilmengen zu bestimmen, führen wir als Nächstes das Konzept der Kreuz-Komplementarität ein.

### 3.3.5 Modul 3: Bestimmen der kreuz-komplementären Teilmengen

In diesem Abschnitt bestimmen wir aus der Menge der Extremalstrahlen $\mathcal{E}$ die maximal kreuz-komplementären Teilmengen. Dazu verwenden wir einen Algorithmus, der auf der Identifikation maximaler Cliquen in einem Graphen basiert. Eine Clique ist ein Teilgraph, in dem alle Knoten miteinander verbunden sind. Diese ist maximal, wenn sie in keiner anderen Clique enthalten ist. Wir übernehmen den Algorithmus vollständig aus der Arbeit von De Schutter und De Moor, siehe [DSDM95, S. 308 ff.], ohne Änderungen an der Methodik vorzunehmen. Im Folgenden erläutern wir das zugrundeliegende Konzept und die Funktionsweise des Algorithmus.

Wir führen zunächst die Definition des Begriffs Kreuz-Komplementarität aus [DSDM95, Definition 4.5.] ein:

**Definition 3.19.** Eine Lösungsmenge $\mathcal{S}$ eines HELKP heißt **kreuz-komplementär**, wenn Koeffizienten $\lambda_k \in \mathbb{R}$ und $\kappa_k \geq 0$ existieren, sodass für

$$u = \sum_{s_k \in \mathcal{S}^{\text{cen}}} \lambda_k s_k + \sum_{s_k \in \mathcal{S}^{\text{nc}}} \kappa_k s_k \tag{50}$$

die Komplementaritätsbedingung erfüllt ist. Die Mengen $\mathcal{S}^{\text{cen}} = \{\mathbf{s} \in \mathcal{S} \,|\, \mathbf{Ps} = 0\}$ und $\mathcal{S}^{\text{nc}} = \{\mathbf{s} \in \mathcal{S} \,|\, \mathbf{Ps} \neq 0\}$ bezeichnen wir als zentrale bzw. nicht-zentrale Lösungen.

Es ist ausreichend, dass wir genau eine strikt positive Linearkombination der Extremalstrahlen einer Teilmenge überprüfen, um zu entscheiden, ob die Extremalstrahlen

der Teilmenge kreuz-komplementär sind oder nicht [DSDM95, Eigenschaft 4.7.].

Um zu bestimmen, welche Extremalstrahlen eine kreuz-komplementäre Teilmenge bilden, konstruieren wir einen ungerichteten Hilfsgraphen $\mathcal{G}(V, \mathcal{E})$. Jeder Knoten in $\mathcal{G}$ entspricht einem Extremalstrahl $\mathbf{e} \in \mathcal{E}$ und eine Kante zwischen zwei Knoten gibt an, dass die zugehörigen Extremalstrahlen kreuz-komplementär sind. Die Suche nach maximalen kreuz-komplementären Teilmengen entspricht dann der Bestimmung der maximalen Cliquen in $\mathcal{G}$, wobei zusätzlich die Komplementaritätsbedingung für die gesamte in der Clique enthaltene Linearkombination erfüllt sein muss.

In einem ersten Schritt reduzieren wir die Menge der Extremalstrahlen, welche wir insgesamt betrachten müssen. Hierzu nutzen wir die folgenden beiden Eigenschaften:

**Eigenschaft 3.20.** [DSDM95, Eigenschaft 4.9.] *Falls* $\mathbf{e} \in \mathcal{E}$ *die Gleichung* $\mathbf{P}_1\mathbf{e} = \mathbf{0}$ *erfüllt, dann gehört* $\mathbf{e}$ *zu jeder kreuz-komplementären Teilmenge.*

Daher können wir diese zentralen Lösungen der Menge $\mathcal{E}^0$ hinzufügen und im Algorithmus aussparen.

**Eigenschaft 3.21.** [DSDM95, Eigenschaft 4.10.] *Sind* $e_1, e_2 \in \mathcal{E}$ *zwei Extremalstrahlen und gilt für alle* $i \in \{1, 2, \ldots, p_1\}$*, dass* $(\mathbf{P}_1 e_1)_i = 0$ *genau dann erfüllt ist, wenn* $(\mathbf{P}_1 e_2)_i = 0$*, dann gehört* $e_1$ *zur kreuz-komplementären Menge genau dann, wenn* $e_2$ *zu der Menge gehört.*

Extremalstrahlen, die die Gleichung $\mathbf{P}_1\mathbf{e} = \mathbf{0}$ für dieselben Zeilen erfüllen, gehören somit zur selben kreuz-komplementären Teilmenge. Die Eigenschaft 3.21 definiert somit eine Äquivalenzrelation $\sim$ auf der Menge $\mathcal{E}\backslash\mathcal{E}^0$ wie folgt:

$$\mathbf{e}_1 \sim \mathbf{e}_2, \text{ wenn für alle } i \in \{1, 2, \ldots, p_1\} \text{ gilt: } ((\mathbf{P}_1\mathbf{e}_1)_i = 0) \Leftrightarrow ((\mathbf{P}_1\mathbf{e}_2)_i = 0).$$

Pro Äquivalenzklasse übernehmen wir je einen Extremstrahl in $\mathcal{E}_{\text{red}}$. Die Reduktion auf $\mathcal{E}_{\text{red}}$ verringert die Anzahl an Iterationen des Algorithmus ohne Informationsverlust. Anschließend bilden wir aus $\mathcal{E}_{\text{red}}$ die korrespondierende Menge $\mathcal{S} = \{\mathbf{P}_1\mathbf{e} \,|\, \mathbf{e} \in \mathcal{E}_{\text{red}}\}$ und wenden auf dieser den folgenden Algorithmus an.

**Initialisierung**
Die Initialisierung des Algorithmus findet im Abschnitt Algorithmus 3 statt. Zunächst berechnen wir die binäre Darstellung aller $\mathbf{s}_k \in \mathcal{S}$, um die Komplementaritätsbedingung effizient überprüfen zu können. Für die Menge $\mathcal{S}$ bilden wir dann den Hilfsgraphen $\mathcal{G}$ in Form der oberen Dreiecksmatrix *cross*. Diese Matrix kodiert, ob zwei Extremalstrahlen kreuz-komplementär sind. Da der Algorithmus auf der Tiefensuche basiert, ist *depth* die zentrale Variable, welche die aktuelle Rekursionstiefe enthält. In der Matrix *vertices* speichern wir für jede Rekursionstiefe, welche Extremalstrahlen als Kandidaten für eine erweiterte kreuz-komplementäre Menge infrage kommen. Die Arrays *start* und *last* enthalten Spaltenindizes der Matrix *vertices*, sodass alle Kandidaten in der aktuellen Rekursionstiefe berücksichtigt werden.

**Algorithmus 3:** Bestimme die kreuz-komplementären Teilmengen (Teil 1), siehe [DSDM95, Algorithmus 3]

**Eingabe:** $m$, $\mathcal{S}$, $\{\phi_j\}_{j=1}^{m}$

$\Gamma \leftarrow \emptyset$

`/* Erstelle die kreuz-komplementäre Hilfsmatrix */`

$\mathcal{B} \leftarrow \{\text{binary}(s_k) \mid s_k \in \mathcal{S}\}$

**für** $k = 1, 2, \ldots, \#\mathcal{B} - 1$ **tue**

  **für** $l = k + 1, k + 2, \ldots, \#\mathcal{B}$ **tue**

    **wenn** $(b_k \vee b_l)$ *die Komplementaritätsbedingung erfüllt* **dann**

      $cross(k, l) \leftarrow 1$

    **Ende**

    **sonst**

      $cross(k, l) \leftarrow 0$

    **Ende**

  **Ende**

**Ende**

$depth \leftarrow 1$

$start(1) \leftarrow 0$

$last(1) \leftarrow \#\mathcal{B}$

**Iterationsschritte**

Der Algorithmus 4 bestimmt rekursiv alle maximalen kreuz-komplementären Teilmengen der Extremalstrahlen. Ausgehend von einer leeren Menge wird diese schrittweise erweitert. In jeder Iteration fügen wir einen Extremalstrahl zur bestehenden Menge hinzu, ohne die Komplementaritätsbedingung zu verletzen. Anschließend werden mögliche Kandidaten der nächsten Rekursionstiefe identifiziert, die kreuzkomplementär zur aktuellen Auswahl sind.

Die Tiefensuche wird solange fortgesetzt, bis kein weiterer Kandidat gefunden wird. In diesem Fall wird die Teilmenge als neue maximale kreuz-komplementäre Teilmenge in $\Gamma$ aufgenommen, sofern sie nicht bereits als echte Teilmenge einer zuvor gefundenen Teilmenge in $\Gamma$ enthaltenen ist. Falls noch nicht alle Extremalstrahlen berücksichtigt wurden, wird zum letzten Entscheidungspunkt zurückgekehrt, um alternative Erweiterungen der Teilmenge zu erstellen.

Das Verfahren endet, sobald alle Extremalstrahlen berücksichtigt wurden. Die resultierende Lösungsmenge $\Gamma$ enthält dann sämtliche maximalen kreuz-komplementären Teilmengen.

In allen resultierenden Teilmengen $\mathcal{S} \in \Gamma$ ersetzen wir dann jedes $\mathbf{s}_k$ durch seinen zugehörigen Extremalstrahl $\mathbf{e}_k$. Zusätzlich fügen wir der jeweiligen Teilmenge alle Extremalstrahlen hinzu, die in der Äquivalenzklasse von $\mathbf{e}_k$ enthalten sind. Abschließend fügen wir jeder Teilmenge alle in $\mathcal{E}_0$ enthaltenen Extremalstrahlen hinzu. Durch diese Schritte erhalten wir $\Gamma$, die Menge der Teilmengen $\mathcal{E}_s$ der kreuzkomplementären Extremalstrahlen. Somit ist $\mathbf{u}$ eine Lösung des HELKP, wenn eine Teilmenge $\mathcal{E}_s \in \Gamma$ existiert, sodass $\mathbf{u} = \sum_{\mathbf{e}_k \in \mathcal{E}_s} \kappa_k \mathbf{e}_k$ mit $\kappa_k \geq 0$ gilt. Im folgenden

**Algorithmus 4:** Bestimme die kreuz-komplementären Teilmengen (Teil 2), siehe [DSDM95, Algorithmus 3]

```
1  ∀k ∈ {1,2,...,#B} : vertices(1,k) ← k solange depth > 0 tue
2  |  start(depth) ← start(depth) + 1
3  |  b ← ⋁_{d=1}^{depth} b_{vertices(d,start(d))}
4  |  current_vertex ← vertices(depth, start(depth))
   |  /* Ermittle Kandidaten für die nächste Rekursionstiefe:      */
5  |  next_depth ← depth + 1
6  |  start(next_depth) ← 0
7  |  last(next_depth) ← 0
8  |  für k = start(depth) + 1,...,last(depth) tue
9  |  |  new_vertex ← vertices(depth, k)
10 |  |  wenn cross(current_vertex, new_vertex) = 1 dann
11 |  |  |  wenn (b ∨ b_{new_vertex}) die Komplementaritätsbedingung erfüllt dann
   |  |  |  |  /* Füge einen neuen Kandidaten zur Teilmenge hinzu:  */
12 |  |  |  |  last(next_depth) ← last(next_depth) + 1
13 |  |  |  |  vertices(next_depth, last(next_depth)) ← new_vertex
14 |  |  |  Ende
15 |  |  Ende
16 |  Ende
17 |  wenn last(next_depth) > 0 dann
18 |  |  depth ← next_depth
19 |  Ende
   |  /* Speichere die maximal kreuz-komplementäre Teilmenge:      */
20 |  sonst
21 |  |  S^new ← ⋃_{d=1}^{depth} {s_{vertices(d,start(d))}}
22 |  |  wenn ∀S_s ∈ Γ : S^new ⊄ S_s dann
23 |  |  |  Γ ← Γ ∪ {S^new}
24 |  |  Ende
   |  |  /* Fall 1: Alle Extremalstrahlen wurden berücksichtigt.   */
25 |  |  wenn start(1) + depth − 1 = #B dann
26 |  |  |  depth ← 0
27 |  |  Ende
   |  |  /* Fall 2: Gehe zurück bis zum letzten Extremalstrahl, wo
   |  |     eine Entscheidung getroffen wurde.                     */
28 |  |  sonst
29 |  |  |  solange start(depth) = last(depth) tue
30 |  |  |  |  depth = depth − 1
31 |  |  |  Ende
32 |  |  Ende
33 |  Ende
34 Ende
```

**Ausgabe:** $\Gamma = \{\mathcal{S}_1, \mathcal{S}_2, \dots\}$

Kapitel extrahieren wir aus $\Gamma$ die Lösungen für das inhomogene ELKP.

## 3.4 Extrahieren der Lösungen für das inhomogene erweiterte lineare Komplementaritätsproblem

Im vorherigen Kapitel haben wir die Lösungsmenge für das IKWPK als HELKP ermittelt. In diesem Abschnitt erläutern wir die Extraktion der Lösung für das ELKP. Ausgangspunkt hierfür ist die Menge von kreuz-komplementären Teilmengen $\Gamma$, welche wir im vorherigem Unterkapitel 3.3 bestimmt haben. Wir folgen dem in [DSDM95, S. 317] beschriebenen Vorgehen zur Extraktion der Lösung für das ELKP.

Die Lösungsmenge $\Gamma$ des IKWPK als HELKP besteht ausschließlich aus Extremalstrahlen. Diese haben aufgrund der Konstruktion des HELKP die Form

$$\mathbf{u} = \begin{pmatrix} \mathbf{x} \\ \alpha \end{pmatrix} \text{ mit } \alpha \geq 0.$$

Um aus der Lösungsmenge des HELKP die Lösungen für das inhomogene ELKP zu bestimmen, teilen wir die enthaltenen Lösungsvektoren in zwei Mengen ein – die unendlichen Strahlen $\mathcal{H}^{\text{inf}}$ und die endlichen Strahlen $\mathcal{H}^{\text{fin}}$.
Hierzu normalisieren wir zunächst diejenigen Vektoren aus $\Gamma$, die eine positive $\alpha$-Komponente besitzen. Die Normalisierung ist aufgrund der Eigenschaft 3.13 möglich, da jede Lösung des HELKP durch eine Skalierung mit einem nichtnegativen Faktor ebenfalls eine Lösung bleibt. Ist $\alpha > 0$ dividieren wir alle Komponenten des Vektors durch $\alpha$, sodass die $\alpha$-Komponente den Wert eins annimmt. Diese Vektoren fassen wir in der Menge

$$\mathcal{E}^{\text{inf}} = \{\mathbf{e}_k \in \mathcal{E} \,|\, \alpha_k = 1\}$$

zusammen. Diejenigen Vektoren aus $\Gamma$, deren $\alpha$-Komponente gleich null ist, erfassen wir analog in der Menge

$$\mathcal{E}^{\text{fin}} = \{\mathbf{e}_k \in \mathcal{E} \,|\, \alpha_k = 0\}.$$

Im nächsten Schritt extrahieren wir den $\mathbf{x}$-Anteil von jedem Vektor aus $\mathcal{E}^{\text{inf}}$ sowie $\mathcal{E}^{\text{fin}}$ und erhalten dadurch die Menge der unendlichen Strahlen $\mathcal{H}_s^{\text{inf}}$ sowie die Menge der endlichen Strahlen $\mathcal{H}_s^{\text{fin}}$.
Für jede Teilmenge $\mathcal{E}_s \in \Gamma$ bestimmen wir anschließend die entsprechenden Teilmengen $\mathcal{H}_s^{\text{inf}} \subset \mathcal{H}^{\text{inf}}$ und $\mathcal{H}_s^{\text{fin}} \subset \mathcal{H}^{\text{fin}}$. Dabei behalten wir jedoch ausschließlich die Paare

$$\{\mathcal{H}_s^{\text{inf}}, \mathcal{H}_s^{\text{fin}}\},$$

deren Teilmenge der endlichen Strahlen $\mathcal{H}_s^{\text{fin}}$ nichtleer ist. Durch dieses Vorgehen erhalten wir die Menge $\Lambda$ von Paaren kreuz-komplementärer unendlicher und endlicher Strahlen. Das folgende Theorem, siehe [DSDM95, Theorem 4.13.], gibt die Voraussetzungen an, unter denen diese Paare eine Lösung für das IKWPK als ELKP bilden.

**Theorem 3.22.** *Seien $\mathcal{H}^{inf}$, $\mathcal{H}^{fin}$ und $\Lambda$ gegeben. Dann ist $\mathbf{x}$ genau dann eine Lösung des inhomogenen ELKP, wenn ein Paar $\{\mathcal{H}_s^{inf}, \mathcal{H}_s^{fin}\} \in \Lambda$ existiert, sodass*

$$\mathbf{x} = \sum_{x_k \in H_s^{inf}} \kappa_k x_k + \sum_{x_k \in H_s^{fin}} \mu_k x_k$$

*mit* $\kappa_k \geq 0$, $\mu_k \geq 0$ *und* $\sum_k \mu_k = 1$ *gilt.*

Nachdem wir die Menge $\Lambda$ der Paare kreuz-komplementärer unendlicher und endlicher Strahlen bestimmt haben, wenden wir auf diese das Minimierungsproblem IKWPK, siehe Definition 3.4, unter Berücksichtigung des Theorems 3.22 an. Dadurch erhalten wir eine oder mehrere Lösungen des ELKP.

Im nächsten Kapitel simulieren wir die algorithmische Lösungsfindung anhand verschiedener Testinstanzen und analysieren die Effizienz des Verfahrens.

# 4 Simulation und Effizienzfaktoren

In diesem Kapitel beschäftigen wir uns zunächst mit der praktischen Analyse des von uns angepassten Algorithmus zur Lösung des IKWPK aus Kapitel 3.3 und 3.4. Hierzu vergleichen wir unsere modifizierte Implementierung mit dem Algorithmus von De Schutter und De Moor in [DSDM95]. Zunächst erläutern wir die Konstruktion der Testinstanzen, welche wir in der Simulation verwenden. Es folgt eine Beschreibung der technischen Implementierung und eine beispielhafte Klassifikation von Lösungsvarianten. Danach visualisieren und diskutieren wir die Simulationsergebnisse, insbesondere hinsichtlich der Laufzeit der verschiedenen Algorithmusvarianten. Abschließend fassen wir die Faktoren, welche die Effizienz des Lösungsalgorithmus besonders beeinflussen, zusammen.

## 4.1 Konstruktion der Testinstanzen

Die hier beschriebene Konstruktion von Testinstanzen verwenden wir ausschließlich zu Simulationszwecken. Alle weiteren Beispiele, die im Kapitel 4 enthalten sind, sind von dieser Konstruktion unabhängig ausgewählte Probleminstanzen.

**Festlegung der Kantendichte** Die Kantendichte $d(G)$ eines Graphen $G$ definieren wir als $d(G) = \frac{2|E|}{|V|(|V|-1)}$, siehe [Die06, S. 178]. Sie nimmt Werte zwischen 0 und 1 an, wobei $d(G) = 0$ einem kantenlosen Graphen und $d(G) = 1$ einem vollständigen Graphen entspricht. Für unsere Testinstanzen legen wir eine Kantendichte von $d = \frac{1}{2}$ fest. Damit berechnet sich die Anzahl der Distanzkanten durch $|D| = \frac{1}{4} \cdot |V|(|V|-1)$. Da die Anzahl der Distanzkanten ganzzahlig sein muss, runden wir diesen Wert auf. Die Anzahl der Distanzkanten wächst bei konstanter Kantendichte $\frac{1}{2}$ quadratisch in Abhängigkeit von der Knotenanzahl, wie in der Abbildung 4 dargestellt. Entsprechend erzeugen wir im Folgenden Testinstanzen mit einer Distanzkantenanzahl zwischen $3 \leq |D| \leq 95$.

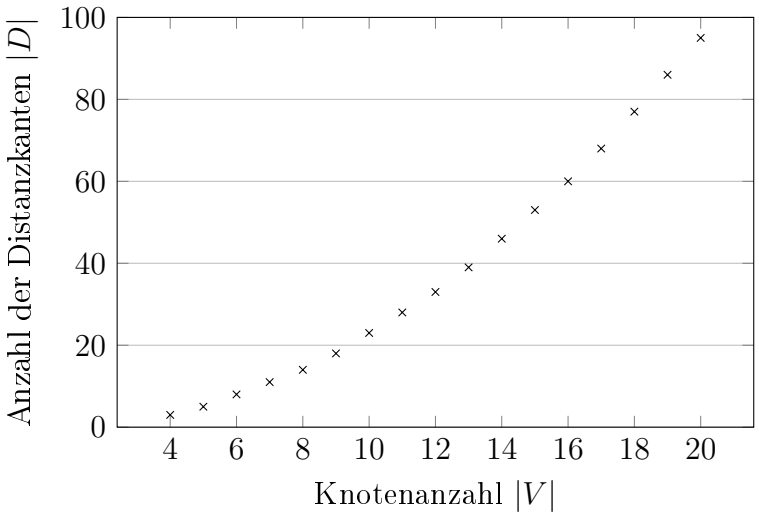



**Abbildung 4:** Zusammenhang zwischen Distanzkantenanzahl $|D|$ und Knotenanzahl $|V|$ bei konstanter Kantendichte $\frac{1}{2}$.

**Implementierung** Zur Erzeugung der Testinstanzen beginnen wir mit der Konstruktion eines gewichteten, ungerichteten Kreisgraphen $G(V, E)$ mit $n$ Knoten und ebenso vielen Kanten. Hierbei ist $n \geq 4$. Jeder Kante $e \in E$ weisen wir ein zufälliges ganzzahliges Gewicht aus dem Intervall $[1, 10]$ zu. Basierend auf diesem gewichteten Kreisgraphen berechnen wir die paarweisen kürzesten Wege zwischen allen Knotenpaaren. Da es für jedes Knotenpaar $(v_i, v_j)$ zwei Wege zwischen den Knoten gibt, den aufsteigenden und absteigenden Weg, bestimmen wir für jedes Knotenpaar die Längen beider Wege und speichern den minimalen Wert in einer Hilfsmatrix ab. Die Hilfsmatrix dient in den weiteren Schritten zur Gewichtszuteilung der zufällig erzeugten Distanzgraphen. Die eigentlichen Testinstanzen, also die Distanzgraphen $G_d(V, D)$, entstehen, indem wir aus dem vollständigen Graphen $K_n$ eine zufällige Auswahl an Kanten auswählen, sodass jeder Knoten zu mindestens einer Kante inzident ist. Letzteres stellt sicher, dass unsere Testinstanzen keine isolierten Knoten aufweisen. Den Anzahl der Kanten berechnen wir anhand der zuvor erläuterten Formel, sodass die Kantendichte des erzeugten Graphen bei $\frac{1}{2}$ liegt. Die Gewichtung der Distanzkanten erfolgt anhand der zuvor berechneten Hilfsmatrix. Das Gewicht einer Distanzkante $d_{ij} \in D$ entspricht dann dem Wert des Elements $(i, j)$ in der Hilfsmatrix. Die Matrixdarstellung der Distanzgraphen erfolgt in Form von Adjazenzmatrizen. In der Adjazenzmatrix ist für die Distanzkanten im Element $(i, j)$ das Gewicht des kürzesten Weges zwischen den Knoten $i$ und $j$ im Kreisgraphen enthalten. Um möglichst verlässliche Ergebnisse zu erhalten, erstellen wir für jede Knotenanzahl $n$ mehrere Testinstanzen, die für alle Versionen einen gültiges Ergebnis liefern.

## 4.2 Implementierung des Algorithmus

**Technische Umgebung** Die Implementierung des Algorithmus setzen wir in der Programmiersprache Julia (Version 1.10.2) um und verwenden dabei die folgenden Bibliotheken:

- **Dateiverarbeitung:** `JSON`, `Dates`, `FilePathsBase`
- **Lineare Algebra:** `LinearAlgebra`
- **Optimierung:** `JuMP`, `Clp`

Die Entwicklung und Berechnung der Testinstanzen erfolgt in der Entwicklungsumgebung Visual Studio Code (Version 1.97.2). Die Berechnungen führen wir auf einem System mit der folgenden Konfiguration durch:

- **Prozessor:** Intel Core i7, 1,9 GHz – 2,11 GHz
- **Betriebssystem:** Windows 11
- **Arbeitsspeicher (RAM):** 16 GB

**Ein- und Ausgabeparameter** Sowohl die Eingabe der Testinstanzen als auch die Ausgabe der Ergebnisse erfolgt über das JSON-Format. Zur Repräsentation der Distanzgraphen verwenden wir Adjazenzmatrizen und fügen jeder Testinstanz zusätzliche Metadaten wie die Anzahl der Knoten im Graphen, den Knotengrad und die Anzahl der Distanzkanten hinzu. Die Eingabe einer Testinstanz weist die folgende Struktur auf:

```
{
"id": 51,
"num_of_vertices": 5,
"distance_matrix": [
                   [0,8,11,0,0],
                   [8,0,8,0,13],
                   [11,8,0,2,0],
                   [0,0,2,0,0],
                   [0,13,0,0,0]
                   ],
"num_of_distance_edges": 5
}.
```

Nach der Verarbeitung der Testinstanz enthält die Ausgabe neben den ursprünglichen Metadaten unter anderem die berechneten Kreiskantengewichte, Laufzeitinformationen sowie die Gesamtanzahl der verarbeiteten Zentral- und Extremalstrahlen und die Anzahl der Iterationen aus dem Hauptteil des Algorithmus. Die Ausgabe erfolgt in der folgenden Struktur:

```
{
"result_edge_weights": [8,8,2,3,6],
"total_central_rays": 0,
"execution_time_s_preprocessing": 0.0,
"id": 51,
"execution_time_s_algorithm": 0.0009999275207519531,
"version": 3,
"total_extreme_rays": 7,
"total_iterations": 7,
"execution_time_s_total": 0.03399991989135742,
"is_valid_result": true,
"num_of_vertices": 5,
"total_edge_weights": 27,
"num_of_distance_edges": 5
}.
```

Diese Struktur ermöglicht uns eine detaillierte Analyse der Testergebnisse hinsichtlich der Laufzeiten. Die Gesamtlaufzeit der Verarbeitung einer Probleminstanz geben wir über den Parameter `execution_time_s_total` an. Diese enthält sowohl die Vorverarbeitungszeit der Instanz (`execution_time_s_preprocessing`) als auch die Laufzeit der algorithmischen Module (`execution_time_s_algorithm`). Alle Zeitangaben erfolgen in Sekunden.

Der Masterthesis liegen im Anhang alle verwendeten Implementierungsversionen

und Testinstanzen bei. Durch diese lassen sich die Simulationsergebnisse aus dem nachfolgenden Kapitel reproduzieren.

## 4.3 Klassifikation der Lösungsinstanzen anhand von Beispielen

Je nach Probleminstanz kann die Lösungsmenge entweder leer sein, mehrere Lösungen mit dem gleichen Gesamtgewicht oder mehrere Lösungen mit unterschiedlichem Gesamtgewicht ausgeben. Zur Veranschaulichung betrachten wir diese drei Fälle exemplarisch an jeweils einem Distanzgraphen.

**Keine Lösungen** Existiert keine Lösung, dann wurde keine $G_d$-erfüllende Gewichtsfunktion auf dem Kreis $G$ für den gegebenen Distanzgraphen gefunden. In diesem Fall ist der Lösungsparameter `result_edge_weights` leer. Ein solches Beispiel ist in der Abbildung 5 dargestellt.

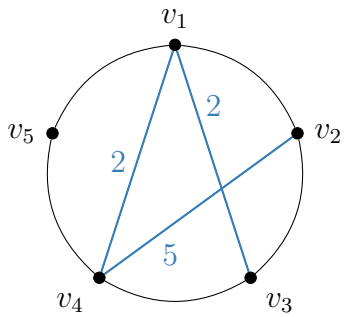



**Abbildung 5:** Beispiel eines in Blau dargestellten Distanzgraphen, für den keine $G_d$-erfüllende Gewichtsfunktion auf dem Kreis $G$ existiert.

**Mehrere Lösungen mit gleichem Gesamtgewicht** Werden mehrere Lösungen mit gleichem Gesamtgewicht ausgegeben, dann können wir alle Lösungen als äquivalent betrachten. Ein Beispiel mit zwei $G_d$-erfüllende Gewichtsfunktion auf dem Kreis $G$ mit gleichem Gesamtgewicht finden wir in der Abbildung 6.

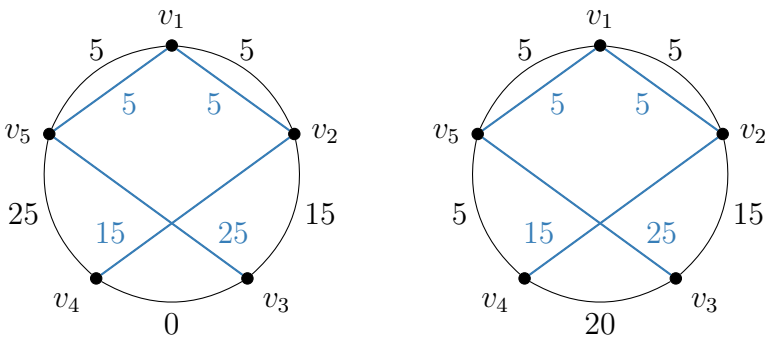



**Abbildung 6:** Beispiel eines in Blau dargestellten Distanzgraphen mit zwei verschiedenen $G_d$-erfüllenden Gewichtsfunktionen gleichen Gesamtgewichts auf dem Kreis $G$. Beide Kreise haben das Gesamtgewicht 50.

**Mehrere zulässige Lösungen mit unterschiedlichem Gesamtgewicht** Gibt der Algorithmus mehrere zulässige Lösungen mit unterschiedlichen Gesamtgewichten aus, dann ist nicht jede Lösung optimal, da wir im IKWPK das minimale Gesamtgewicht bestimmen. In diesem Fall verwerfen wir alle Lösungen mit größerem Gesamtgewicht. Die Abbildung 7 zeigt ein Beispiel mit zwei $G_d$-erfüllenden Gewichtsfunktionen, wobei wir nur die mit kleinerem Gesamtgewicht 34 als optimal betrachten.

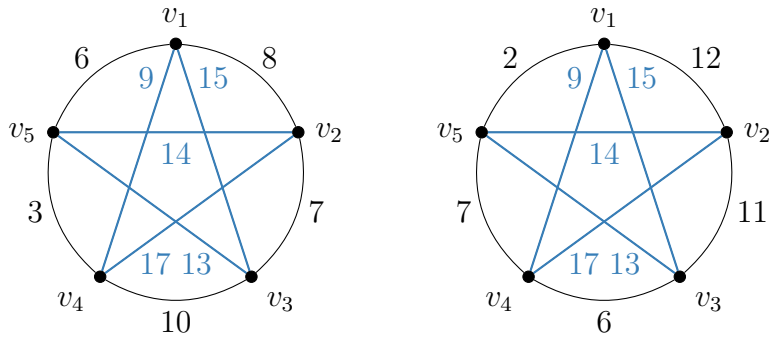



**Abbildung 7:** Beispiel eines in Blau dargestellten Distanzgraphen mit zwei verschiedenen $G_d$-erfüllenden Gewichtsfunktionen unterschiedlichen Gesamtgewichts auf dem Kreis $G$. Der linke Kreisgraph weist ein Gesamtgewicht von 34 auf, wohingegen der rechte Kreisgraph ein höheres Gesamtgewicht von 38 hat.

Standardmäßig geben wir die erste verbliebene Lösungsinstanz aus.

## 4.4 Darstellung und Analyse der Simulationsergebnisse

In der folgenden Simulation untersuchen wir die Effizienz von drei unterschiedlichen Versionen des Lösungsalgorithmus für das IKWPK. Unser Ziel ist es, die Unterschiede insbesondere hinsichtlich der Laufzeit und der Anzahl der betrachteten Strahlen zu analysieren.

Die unterschiedlichen Versionen des Lösungsalgorithmus haben folgende Eigenschaften:

- **Version 1:** Diese Implementierung entspricht vollständig dem von De Schutter und De Moor beschriebenen Algorithmus aus [DSDM95]. Der Algorithmus verwendet sowohl Zentral- als auch Extremalstrahlen. Parallele Distanzkanten werden in der Nebenbedingung $\mathbf{Qu} = \mathbf{0}$ verarbeitet.
- **Version 2:** Diese Version verwendet ausschließlich Extremalstrahlen. Der Algorithmus wird deshalb gemäß Kapitel 3.3.2 initialisiert und enthält die Algorithmenmodule 1 und 2, wie in Kapitel 3.3.3 und 3.3.4 beschrieben. Parallele Distanzkanten werden in der Nebenbedingung $\mathbf{Qu} = \mathbf{0}$ analog zur Version 1 verarbeitet.
- **Version 3:** Diese Implementierung entspricht der in Kapitel 3.3 entwickelten Variante. Der Algorithmus verwendet ausschließlich Extremalstrahlen. In der

Vorverarbeitung werden parallele Distanzkanten identifiziert und deren Werte in den Kantengewichten direkt festgelegt. Außerdem erfolgt ein Kantengewichtsvergleich. Dadurch identifizierte kürzeste auf- und absteigende Wege werden in der Nebenbedingung $\mathbf{Qu} = \mathbf{0}$ verarbeitet.

Die Versionen unterscheiden sich in ihrer Struktur deutlich. Wir erwarten, dass die Beschränkung auf Extremalstrahlen in den Versionen 2 und 3 sowie die zusätzlichen Vorverarbeitungsschritte in Version 3 einen positiven Einfluss auf die Laufzeit und die Gesamtanzahl der verarbeiteten Strahlen haben. In der nachfolgenden Analyse der Simulationsergebnisse überprüfen wir diese Vermutung und untersuchen, inwiefern die Modifikationen zu einer Effizienzsteigerung beitragen.

Da die Version 1 bei Instanzen mit mindestens 11 Knoten nach mehr als 30 Minuten Laufzeit keine Ergebnisse ausgibt, liegen für diese Version ausschließlich Testergebnisse bis $|V| = 10$ vor.

Die Simulationsergebnisse, welche in den Abbildungen 8 und 9 dargestellt sind, geben die Entwicklung der durchschnittlichen Gesamtlaufzeiten und verarbeitete Strahlen für Graphen mit 4 bis 10 Knoten wider.

In Abbildung 8 erkennen wir, dass die durchschnittliche Laufzeit bei der Version 1 mit zunehmender Knotenanzahl deutlich steigt. Die Laufzeit liegt bei $|V| = 4$ bei etwa 0,03 s und erhöht sich auf 1,73 s bei $|V| = 10$, was auf ein möglicherweise exponentielles Wachstum hindeutet. Dagegen sehen wir für die beiden Versionen 2 und 3 durchgehend geringere Laufzeiten. Für die Version 3 bleibt die Laufzeit bei kleinen Instanzen mit bis zu 10 Knoten unter 0,15 s. Die Laufzeit der Version 2 bleibt über dieselben Instanzen unter dem noch kleinerem Wert von 0,08 s. Dies liegt vermutlich daran, dass die Vorverarbeitungsschritte der Version 3 bei Instanzen mit geringer Knotenanzahl einen größeren Einfluss auf die Gesamtlaufzeit haben als die Algorithmen 1 und 2 selbst. Insbesondere handelt es sich hierbei um Unterschiede im Millisekundenbereich, weshalb wir für den Datenausschnitt schließen können, dass die Versionen 2 und 3 in etwa gleich schnell sind.

Für die durchschnittliche verarbeitete Strahlenanzahl zeigt sich in Abbildung 9, dass deren Anzahl für größere werden Testinstanzen bei allen Versionen zunimmt. Die Strahlenanzahl setzt sich aus Extremalstrahlen und Zentralstrahlen zusammen. Bei der Version 1 erkennen wir einen starken Anstieg der Anzahl. Dies ist ein zentraler Faktor, welcher das deutlich schlechtere Laufzeitverhalten der Version 1 erklärt. Ursache hierfür ist, dass in der Version 1 sowohl Zentral- als auch Extremalstrahlen verwendet werden. Dafür spricht auch, dass der Anstieg der durchschnittlichen Strahlenanzahl mit wachsender Knotenanzahl für die Versionen 2 und 3 deutlich geringer ist. Beide Versionen verwenden ausschließlich Extremalstrahlen. Wir beobachten, dass die Anzahl der Strahlen für die Version 2 durchgehend mehr als doppelt so hoch ist wie die der Version 3. Bei der Knotenanzahl 10 verarbeitet die Version 2 durchschnittlich 705 Extremalstrahlen und die Version 3 hingegen nur 197 Extremalstrahlen. Dies ist ein Hinweis darauf, dass die Vorverarbeitung in Version 3 die Komplexität der Probleminstanz deutlich reduziert.

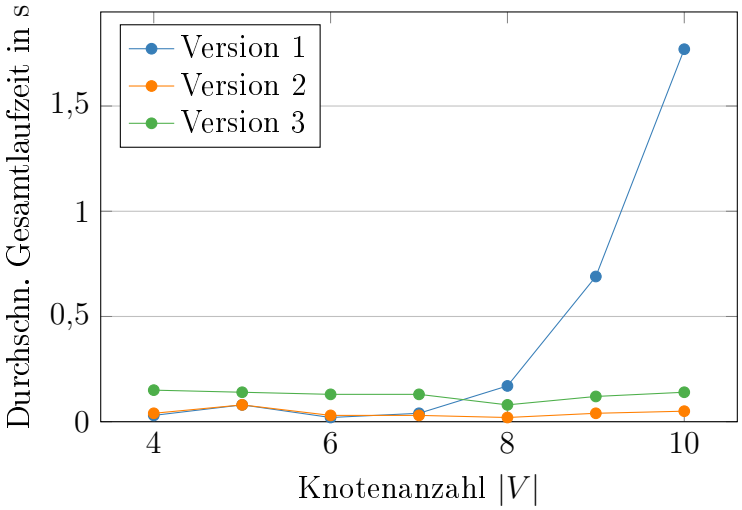



**Abbildung 8:** Abhängigkeit der durchschnittlichen Gesamtlaufzeit von der Knotenanzahl $|V|$ zwischen 4 und 10 je Version.

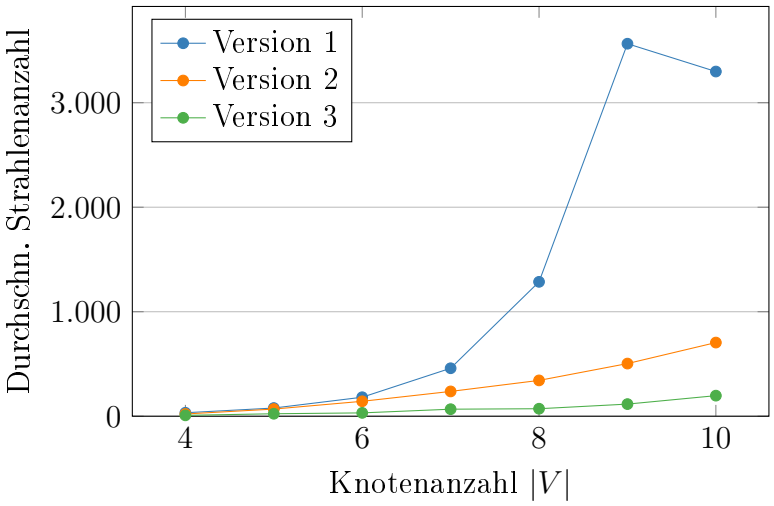



**Abbildung 9:** Abhängigkeit der durchschnittlichen Strahlenanzahl von der Knotenanzahl $|V|$ zwischen 4 und 10 je Version.

Die Abbildungen 10, 11 und 12 zeigen die Ergebnisse für Instanzen mit bis zu 20 Knoten.

Zunächst fällt in der Abbildung 10 auf, dass die durchschnittlichen Gesamtlaufzeiten der Version 3 bei Instanzen bis zu 20 Knoten durchgängig niedrig, unter dem Wert von 0,45 s, bleiben. Für die Version 2 zeichnet sich dagegen ab Instanzen mit 16 Knoten ein starker Anstieg der Laufzeit ab. Diese überschreitet bei 16 Knoten 1s und steigt auf bis zu 4,9 s bei der Knotenanzahl 19 an. Dies deutet daraufhin, dass die Vorverarbeitungsschritte der Version 3 insbesondere für größere Probleminstanzen einen positiven Einfluss haben. Die beobachteten Schwankungen im Verlauf könnten auf die Auswahl der Testinstanzen zurückzuführen sein.

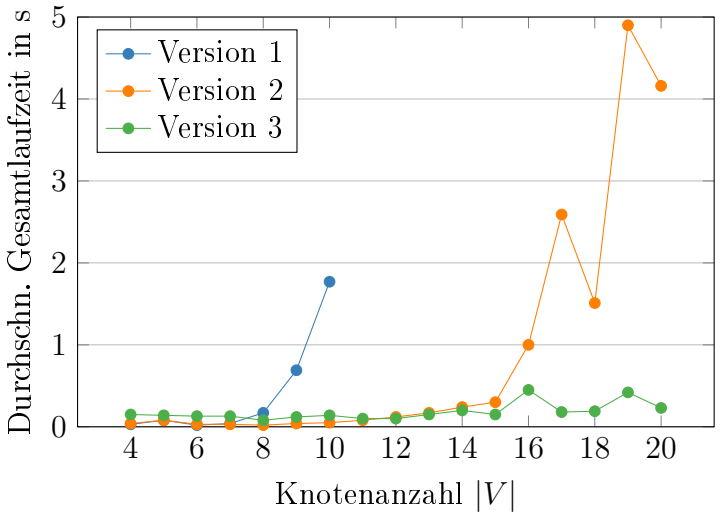



**Abbildung 10:** Abhängigkeit der durchschnittlichen Gesamtlaufzeit von der Knotenanzahl $|V|$ zwischen 4 und 20 je Version.

In der Abbildung 11 sehen wir, dass sich die durchschnittliche verarbeitete Strahlenanzahl für die Versionen 2 und 3 wie zuvor auch bei höherer Knotenanzahl weiter fortschreibt. Während Version 2 bei 20 Knoten circa 8616 Extremalstrahlen verarbeitet, sind dies bei Version 3 nur 1021 Extremalstrahlen.

Vergleichen wir die in Abbildung 12 dargestellte durchschnittliche verarbeitete Strahlenanzahl pro Iteration, so sehen wir, dass die durchschnittliche Strahlenanzahl pro Iteration mit höherer Knotenanzahl sehr stark ansteigt. In der Version 1 wird bei 9 Knoten der Wert 85 erreicht. Dies verdeutlicht den Mehraufwand, der pro Iteration durch die Verwendung von Zentral- und Extremalstrahlen entsteht. Dahingegen zeigt sich ein kontinuierlicher aber moderater Anstieg der Werte in Version 2. Insbesondere ist der Anstieg weniger ausgeprägt als wir dies für die Gesamtanzahl der Strahlen beobachten konnten. Dies deutet daraufhin, dass die Anzahl der Iterationen für größere Testinstanzen ebenfalls zunimmt. Die Version 3 zeigt durchgehend die niedrigsten Werte pro Iteration an. Diese steigen für die untersuchten Instanzen nicht über 20 Strahlen pro Iteration.

Aus den Ergebnissen können wir folgern, dass die Beschränkung auf Extremalstrah-

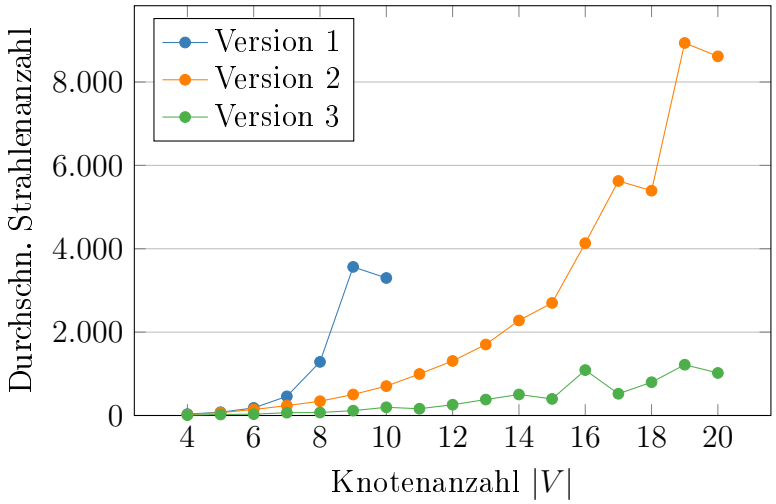



**Abbildung 11:** Abhängigkeit der durchschnittlichen Strahlenanzahl von der Knotenanzahl $|V|$ zwischen 4 und 20 je Version.

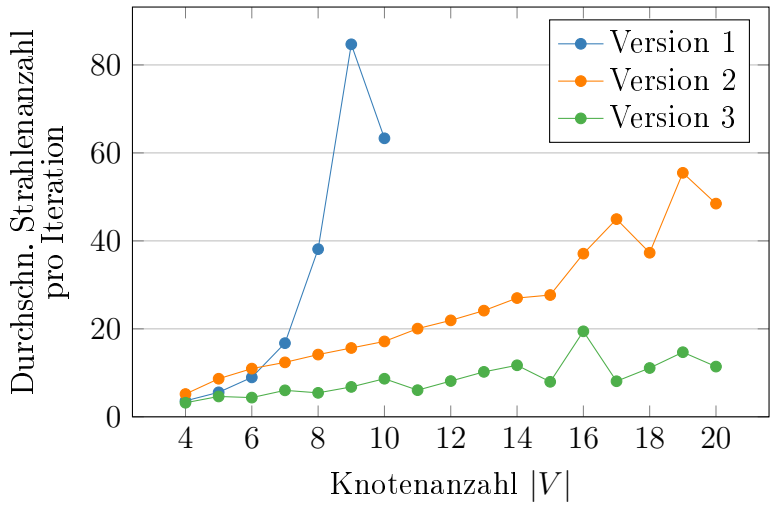



**Abbildung 12:** Abhängigkeit der durchschnittlichen Strahlenanzahl pro Iteration von der Knotenanzahl $|V|$ zwischen 4 und 20 je Version.

len sowohl zu einer geringeren Anzahl der verarbeiteten Strahlen und damit einhergehend zu einer besseren Laufzeit des Lösungsalgorithmus für das IKWPK führt. Die durchschnittliche Strahlenanzahl bzw. Gesamtlaufzeit wächst zwar auch bei den Versionen 2 und 3 mit größer werdender Knotenanzahl, jedoch ist der Anstieg deutlich moderater als in der Version 1. Wie vermutet haben die Vorverarbeitungsschritte in der Version 3 einen weiteren positiven Effekt auf die Effizienz des Lösungsalgorithmus. Da durch die Vorverarbeitungsschritte die Komplexität der Probleminstanz verringert wird, führt das zu einer im Durchschnitt deutlich geringeren Anzahl von verarbeiteten Strahlen. Dies wird insbesondere bei Instanzen mit höherer Knotenanzahl deutlich und zeigt sich in geringen Gesamtlaufzeiten der Version.

In der Abbildung 13 sind die Laufzeiten der Version 3 detailliert dargestellt. Wir sehen, dass sich die Vorverarbeitungslaufzeit über den Verlauf nicht stark verändert und zwischen einem Wert von 0,06 s und 0,17 s schwankt. Sollten wir Instanzen mit einer höheren Anzahl verarbeiten, könnte dieser Wert steigen, da wir im Kantengewichtsvergleich paarweise Kanten vergleichen. Für die Laufzeit der Algorithmen sehen wir eine größere Varianz und eine Laufzeitsteigerung von 0 s bis zu 0,28 s. Ersteres kommt vor, wenn die Lösung allein durch die Vorverarbeitung bestimmt wird. Die Daten bestätigen unsere Vermutung, dass die Vorverarbeitung bei Instanzen mit geringer Knotenanzahl einen größeren Einfluss auf die Gesamtlaufzeit hat als die Algorithmen.

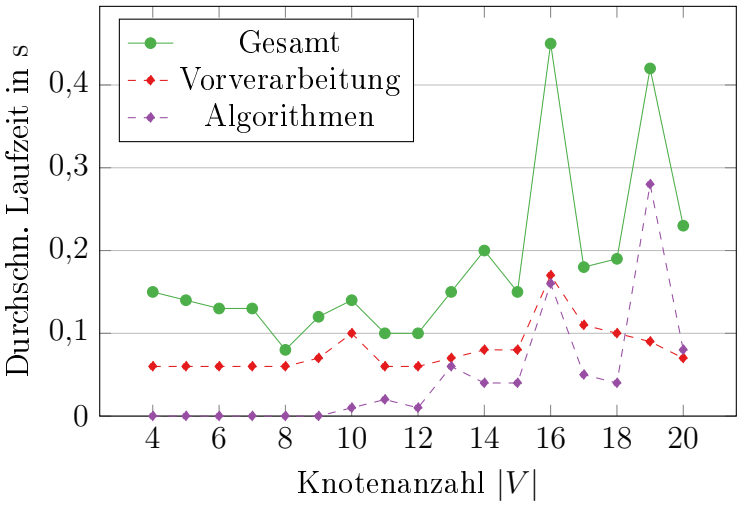



**Abbildung 13:** Abhängigkeit der durchschnittlichen Laufzeiten von der Knotenanzahl $|V|$ zwischen 4 und 20 in der Version 3.

Unsere Simulation erfolgt ausschließlich auf Testinstanzen mit einer Kantendichte von $\frac{1}{2}$. Die beobachteten Laufzeiten stehen in starker Abhängigkeit der jeweilig verwendeten Testinstanzen. So könnten bei Instanzen mit einer höheren Anzahl von parallelen Distanzkanten die Vorteile der Version 3 gegenüber der Version 2 noch stärker sichtbar werden. Um dies weitergehend zu untersuchen, analysieren wir eine Reihe von Distanzgraphen. Ausgangspunkt bildet ein Distanzgraph mit 8 Knoten ohne parallele Distanzkanten, der in Abbildung 14 dargestellt ist. Indem wir je Iteration eine weitere parallele Distanzkante zum Distanzgraphen hinzufügen, entstehen

neue Instanzen mit wachsender Anzahl paralleler Distanzkanten. Diese untersuchen wir hinsichtlich ihrer Gesamtlaufzeit und algorithmischen Laufzeit.

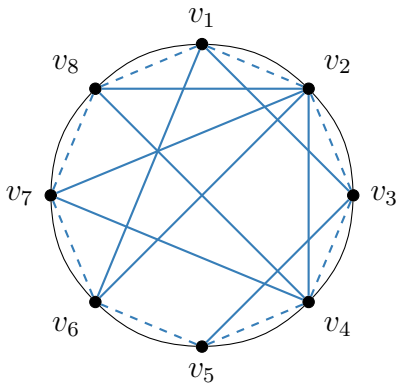



**Abbildung 14:** Das Beispiel eines Distanzgraphen mit 8 Knoten und ohne parallele Distanzkanten ist durch durchgängige Linien dargestellt. Zur Erzeugung weiterer Instanzen fügen wir pro Iteration eine weitere parallele Distanzkante hinzu, die gestrichelt markiert sind.

In Abbildung 15 sehen wir, dass sich die Gesamtlaufzeiten der Version 2 und 3 für Distanzgraphen ohne, mit einer und mit allen parallelen Distanzkanten deutlich voneinander unterscheiden. Die Gesamtlaufzeiten der Version 3 weisen für diese Instanzen eine um 0,1 s bis 0,44 s langsamere Laufzeit als die der Version 2 auf.

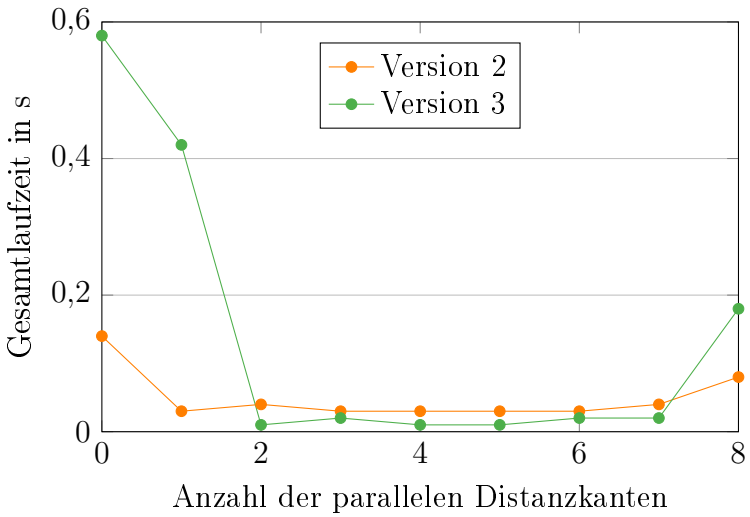



**Abbildung 15:** Gesamtlaufzeiten der Instanzen zum Beispiel in Abbildung 14 für die Version 2 und 3 über der Anzahl paralleler Distanzkanten. Die Angabe der Messdaten erfolgt in Sekunden.

In Abbildung 16 sind die algorithmischen Laufzeiten der Instanzen dargestellt. Auffällig ist, dass die Laufzeiten der Version 3 unabhängig von der Anzahl paralleler Distanzkanten durchgehend unter denen der Version 2 liegen. Zudem verringert sich die algorithmische Laufzeit der Version 3 mit zunehmender Anzahl paralleler

Distanzkanten. Diese Beobachtung erklären wir dadurch, dass in der Vorverarbeitung mit wachsender Anzahl paralleler Distanzkanten die zu lösende Probleminstanz zunehmend kleiner wird. Der Grund dafür ist, dass die Kantengewichte paralleler Distanzkanten in der Version 3 bereits in der Vorverarbeitung explizit festgelegt werden. Sind alle parallelen Distanzkanten im Distanzgraphen vorhanden, ist das IKWPK sogar vollständig gelöst, ohne die Algorithmen zu durchlaufen. Darüber hinaus ist auffällig, dass die Version 3 auch für Distanzgraphen ohne oder mit nur einer parallelen Distanzkante etwas bessere algorithmische Laufzeiten aufweist als Version 2. Dies führen wir auf den Kantengewichtsvergleich in der Vorverarbeitung zurück, der auf- und absteigende Wege identifiziert. Dieser Schritt wird ebenfalls in der Vorverarbeitung vorgenommen. Jedoch schlägt sich der dadurch entstandene zusätzliche Aufwand zwar nicht in der algorithmischen Laufzeit aber in der Gesamtlaufzeit nieder. Dies erklärt die höheren Gesamtlaufzeiten für die Version 3 im Vergleich zur Version 2, welche wir zuvor für bestimmte Instanzen festgestellt haben.

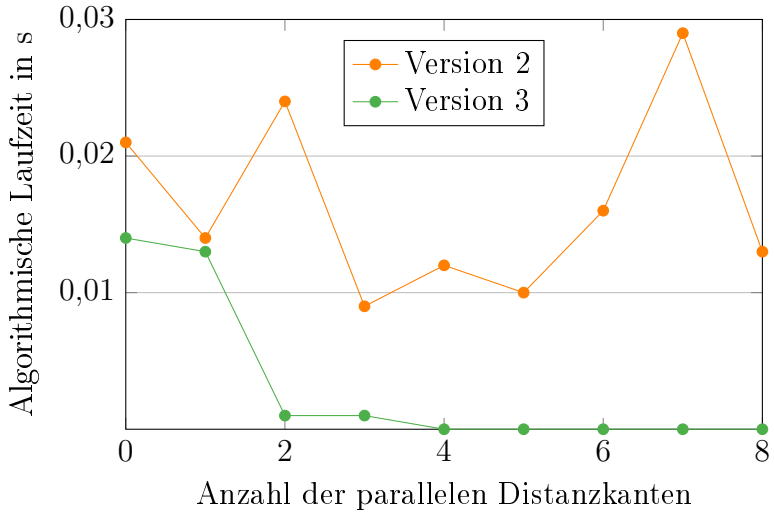



**Abbildung 16:** Vergleich der algorithmischen Laufzeiten zwischen der Version 2 und 3 für die Instanzen des Beispiels aus Abbildung 14 über der Anzahl paralleler Distanzkanten.

Die Ergebnisse zeigen, dass die Vorverarbeitung in Version 3 zwar die algorithmische Laufzeit verbessert, jedoch nicht zwingend zu einer geringeren Gesamtlaufzeit führt als Version 2. Andererseits haben wir in Abbildung 13 gesehen, dass die Vorverarbeitungszeit bei Graphen mit bis zu 20 Knoten mit zunehmender Knotenanzahl weniger Einfluss auf die Gesamtlaufzeit hat.

## 4.5 Effizienzfaktoren

In diesem Kapitel identifizieren wir die Faktoren, welche die Effizienz des Algorithmus zur Lösung des ELKP, den wir spezifisch zur Lösung des IKWPK angepasst haben, beeinflussen.

Die Effizienz der Algorithmen 1 bis 3 wird insbesondere durch die Anzahl der erzeugten Zentral- und Extremalstrahlen beeinflusst, vgl. [DSDM95, S. 319]. Die Si-

mulation hat gezeigt, dass die Einschränkung auf Extremalstrahlen zu einer starken Verminderung der zu verarbeitenden Strahlen sowohl insgesamt als auch pro Iteration führt. Weiterhin konnten wir die Komplexität der Probleminstanz in Version 3 durch mehrere Verarbeitungsschritte verringern. Hierzu gehören das Festlegen der Kreiskantengewichte von parallelen Distanzkanten und die Identifikation von auf- und absteigenden Wegen. Zusätzlich hat die Umstrukturierung von $\mathbf{P}_1$ dazu beigetragen, dass Strahlen, die die partielle Komplementaritätsbedingung nicht erfüllen, in allen Versionen frühzeitig verworfen wurden. Die detaillierte Untersuchung der Version 3 hat zwar gezeigt, dass die Vorverarbeitung einen zusätzlichen Zeitaufwand verursacht, jedoch vor allem bei Probleminstanzen mit vielen parallelen Distanzkanten zu deutlich kleineren algorithmischen Laufzeiten führt.

Weitere entscheidende Einflussfaktoren auf die Effizienz sind neben den Testinstanzen die verwendete Hardware sowie die Implementierung in Julia. Letzteres beeinflusst die Laufzeit der Algorithmen maßgeblich durch die dort verwendeten Operationen. Da alle Versionen von uns implementiert wurden, sind die Laufzeiten über die Versionen hinweg im Wesentlichen vergleichbar. Dennoch besteht Potential, die Implementierung durch eine effizientere Programmierung zu optimieren, um sie für Probleminstanzen mit einer höheren Knotenanzahl robust zu machen sowie die beobachteten Laufzeiten weiter zu verringern. Dies gilt insbesondere für die Vorverarbeitung in der Version 3.

Da das allgemeine ELKP zu den $\mathcal{NP}$-schweren Problemen gehört, siehe [DSDM95, Theorem 4.16.], ist es nicht effizient lösbar. Ob das IKWPK für unsere entwickelte Version 3 effizient lösbar ist, lässt sich aus der Simulation nicht ableiten.

# 5 Zusammenfassung

Ziel dieser Arbeit war die Entwicklung eines geeigneten Lösungsverfahrens für das IKWPK. Bei der Rekonstruktion der Kreiskantengewichte war dabei zu beachten, dass jeweils genau zwei Wege zwischen zwei beliebigen Knoten als kürzester Weg in Frage kommen – der aufsteigende und der absteigende Weg.

Zunächst haben wir festgestellt, dass das IKWPK als ganzzahliges lineares Programm nicht effizient lösbar ist. Das Problem enthält für die ab- und aufsteigenden Wege Nebenbedingungen, die Binärvariablen erfordern. Deren Anzahl an möglichen Kombinationen wächst exponentiell. Ebenso erwies sich die Formulierung als LKP als ungeeignet, da die Wegmatrix $\mathbf{W}^+$ nicht invertierbar ist. Deshalb haben wir das IKWPK in eine allgemeinere Form des LKP transformiert – in das ELKP und dessen homogene Variante. Das ELKP ist als $\mathcal{NP}$-schweres Problemen zwar ebenfalls nicht effizient lösbar, jedoch handelt es sich bei dem IKWPK aufgrund des zugrundeliegenden Kreisgraphen um ein viel spezifischeres Problem. Die von De Schutter und De Moor entwickelten Methodiken zur Lösung des ELKP haben wir auf das IKWPK angepasst. Eine zentrale Eigenschaft des IKWPK als HELKP ist, dass dessen Lösungsmenge ausschließlich aus Extremalstrahlen besteht. Deshalb konnten wir alle Berechnungsschritte mit Zentralstrahlen aus den ursprünglichen Algorithmen von De Schutter und De Moor entfernen. Zudem haben wir Vorverarbeitungsschritte der Probleminstanzen entwickelt, die die Komplexität dieser verkleinern. Insbesondere haben das Festlegen von Kantengewichten paralleler Distanzkanten und die Identifikation von auf- und absteigenden Wegen durch den Kantengewichtsvergleich dazu beigetragen.

Die Simulation auf Testinstanzen mit 4 bis 20 Knoten haben gezeigt, dass unsere Anpassungen sowohl die Anzahl der verarbeiteten Strahlen als auch die Gesamtlaufzeiten des Lösungsalgorithmus reduzieren konnten. Während sich in der Gesamtlaufzeit des Algorithmus von De Schutter und De Moor ein exponentielles Wachstum andeutete, zeigten die angepassten Versionen einen deutlich moderateren Anstieg der Gesamtlaufzeit. Insbesondere ließ sich beobachten, dass die Vorverarbeitung bei größeren Instanzen zu noch geringeren Gesamtlaufzeiten führt. Dies ist insbesondere darauf zurückzuführen, dass durch die Vorverarbeitung vor allem die Komplexität der Probleminstanz reduziert wird, was zu einer geringeren Anzahl an zu verarbeitenden Extremalstrahlen pro Iteration führt.

Wir haben somit ein Lösungsverfahren für das IKWPK gefunden. Ob dieses effizient im Sinne von polynomieller Laufzeit ist, bedarf einer weitergehenden Untersuchung mittels einer detaillierten Komplexitätsanalyse sowie weiteren Simulationen auf größeren Testinstanzen.

Im Laufe der vorliegenden Arbeit haben sich mehrere Fragestellungen ergeben, die über den Rahmen dieser Arbeit hinausgehen und sich für zukünftige Untersuchungen anbieten:

- Die in dieser Arbeit in Kapitel 3.3.2 erläuterten Vorverarbeitungsschritte, bei denen das Kantengewicht durch parallele Distanzkanten explizit bestimmt und

Ungleichungsbedingungen mittels des Kantengewichtsvergleichs durch Gleichungsbedingungen ersetzt werden, haben gezeigt, dass manche Probleminstanzen deutlich effizienter zu lösen sind als andere. Um Rückschlüsse auf die Rechenkomplexität ziehen zu können, wäre daher zu klären, welche Instanzen zum Worst-Case-Verhalten unseres Algorithmus führen.

- Die Vorverarbeitung der Probleminstanzen führt, wie in Kapitel 4.5 aufgezeigt, zu besseren Laufzeiten des Lösungsalgorithmus. Dies ist insbesondere der Fall, wenn im Distanzgraphen viele parallele Distanzkanten enthalten sind. Erste Versuche haben jedoch auch gezeigt, dass der Algorithmus ebenfalls bei einer kleineren Anzahl paralleler Kanten effizienter ist, wenn im Distanzgraphen durch den Kantengewichtsvergleich absteigende und aufsteigende Wege identifiziert werden konnten. Daher wäre zu quantifizieren, in welchem Maße der Kantengewichtsvergleich zu einer Verbesserung der Laufzeit beiträgt.

- In einigen aktuellen Arbeiten werden mögliche Erweiterungen der *Double Description Method*, auf der auch unsere Algorithmen 1 und 2 basieren, diskutiert. Ein Thema ist dabei der Einfluss der Reihenfolge, in der die Ungleichungen bearbeitet werden. Dies haben wir für die Matrix $\mathbf{P}_1$ im Kapitel 3.3.2 ebenfalls thematisiert. Auch der Einsatz alternativer Adjazenztests wird diskutiert. Es wäre daher zu untersuchen, inwieweit diese Ansätze im Kontext des IKWPK zu verbesserten Laufzeiten führen.

Hinsichtlich der letzten Fragestellung verweisen wir auf die folgenden Arbeiten:

- Zolotykh, Nikolai Yu: *New modification of the double description method for constructing the skeleton of a polyhedral cone*, 2012.

- Genov, Blagoy: *The Convex Hull Problem in Practice : Improving the Running Time of the Double Description Method*, 2015.

- Semenov, Sergey O.; Zolotykh, Nikolai Yu : *An Experimental Analysis of Dynamic Double Description Method Variations*, 2022.

# Anhang

Der übermittelte Anhang enthält die für die Simulation in Kapitel 4 verwendeten Algorithmen, Testinstanzen und Ergebnisse. Der Anhang umfasst die folgenden Dateien.

## Dateiverzeichnis

`Ergebnisformatierer.jl` Konvertiert JSON-Ergebnisdateien in eine CSV-Datei.

`Simulationsbeispiel_input.json` Enthält die Instanzen des Beispiels 14.

`Simulationsergebnisse.csv` Enthält die Simulationsergebnisse für die Analyse.

`Testinstanzengenerator.jl` Algorithmus zum Erstellen der Testinstanzen.

`V1_IKWPK.jl` Version 1 des Lösungsalgorithmus für das IKWPK.

`V2_IKWPK.jl` Version 2 des Lösungsalgorithmus für das IKWPK.

`V2_Simulationsbeispiel_output.json` Enthält die Ergebnisse des Beispiels aus Abbildung 14 für die Version 2.

`V3_IKWPK.jl` Version 3 des Lösungsalgorithmus für das IKWPK.

`V3_Simulationsbeispiel_output.json` Enthält die Ergebnisse des Beispiels aus Abbildung 14 für die Version 3.

Zusätzlich sind für die in der Simulation untersuchten Kombinationen aus Version $x$ und Knotenanzahl $n$ die folgenden Dateien vorhanden:

`Instanzen_`$n$`.json` Enthält die verwendeten Testinstanzen für eine gegebene Knotenanzahl $n$.

`V`$x$`_Instanzen_`$n$`_output.json` Enthält die Ergebnisse der Berechnungen für die entsprechenden Testinstanzen mit Knotenanzahl $n$ für Version $x$.

# Literatur


[BGW21] Beineke, Lowell W. ; Golumbic, Martin C. ; Wilson, Robin J.: *Topics in Algorithmic Graph Theory.* Cambridge University Press, 2021

[CPS09] Cottle, Richard W. ; Pang, Jong-Shi ; Stone, Richard E.: *The Linear Complementarity Problem.* Society for Industrial and Applied Mathematics, 2009

[Die06] Diestel, Reinhard: *Graphentheorie.* Springer, 2006

[DSDM95] De Schutter, Bart ; De Moor, Bart: The extended linear complementarity problem. In: *Mathematical Programming* 71 (1995), Dezember, Nr. 3, S. 289–325

[FP96] Fukuda, Komei ; Prodon, Alain: Double description method revisited. In: Deza, Michel (Hrsg.) ; Euler, Reinhardt (Hrsg.) ; Manoussakis, Ioannis (Hrsg.): *Combinatorics and Computer Science*, Springer Berlin Heidelberg, 1996, S. 91–111

[Gen15] Genov, Blagoy: *The Convex Hull Problem in Practice : Improving the Running Time of the Double Description Method.* 2015

[Hoc17] Hochstättler, Winfried: *Lineare Optimierung.* Springer Spektrum, 2017

[KMY89] Kojima, Masakazu ; Mizuno, Shinji ; Yoshise, Akiko: A polynomial-time algorithm for a class of linear complementarity problems. In: *Mathematical Programming* 44 (1989), S. 1–26

[Mol95] Moll, Christoph: *Das inverse Kürzeste-Wege-Problem*, Universität zu Köln, Diss., 1995

[MY97] Murty, Katta G. ; Yu, Feng-Tien: *Linear Complementarity, Linear and Nonlinear Programming.* 1997

[Sau23] Sauer, Adrian: *Ein Rekonstrutionsproblem von Teilstreckenlängen aus Tourendaten bei Rundreisen.* 2023. – Bachelorarbeit

[Sch98] Schrijver, Alexander: *Theory of Linear and Integer Programming.* Wiley, 1998

[SZ22] Semenov, Sergey ; Zolotykh, Nikolai Y.: An Experimental Analysis of Dynamic Double Description Method Variations, 2022, S. 178–188

[Weg03] Wegener, Ingo: *Komplexitätstheorie.* Springer, 2003

[Zol12] Zolotykh, Nikolai Y.: New modification of the double description method for constructing the skeleton of a polyhedral cone. In: *Computational Mathematics and Mathematical Physics* 52 (2012)