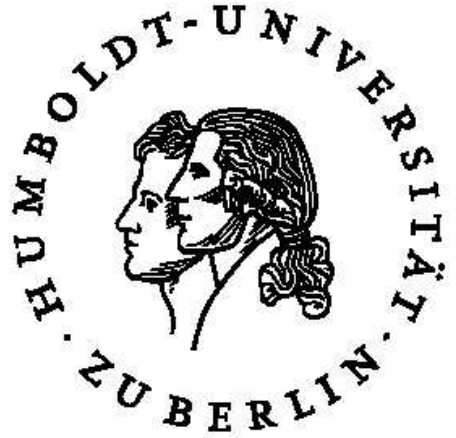
Humboldt-Universität zu Berlin



Mathematisch-Naturwissenschaftliche Fakultät II
Institut für Informatik
Lehrstuhl für Algorithmen und Komplexität

Diplomarbeit

Ein Approximationsalgorithmus zur Berechnung eines 2-Spanners in planaren Triangulationen

Mariano Zelke

Berlin, den 28. November 2005

Betreuer: PD Dr. Stefan Hougardy

Inhaltsverzeichnis

1 Einleitung

Von Peleg und Ullman [12] im Jahre 1987 eingeführt, ist das Konzept eines k-Spanners seitdem vielfach untersucht worden. Für eine konstante rationale Zahl $k \geq 1$ ist ein *k-Spanner* eines Graphen $G = (V, E)$ ein spannender Subgraph $S = (V, E')$ von G, für den für alle $u, v \in V$ gilt:

$$dist_S(u, v) \leq k \cdot dist_G(u, v)$$

Die Distanz zwischen je zwei Knoten ist damit im k-Spanner höchstens um den Faktor k größer als die Distanz im ursprünglichen Graphen G. Die Zahl k wird *stretch factor* genannt.

Spanner spielen eine Rolle beispielsweise beim Design von Routing-Schemata für Netzwerke, in denen kleine Routing-Tabellen wünschenswert sind [13]. Verwendung finden sie auch bei der Approximation von vollständigen euklidischen Graphen [4] und in der Biologie bei der Rekonstruktion von phylogenetischen Bäumen [1].

Natürlich kann das Modell eines k-Spanners auf einen gewichteten Graphen angewendet werden, dessen Kanten beliebige Längen haben können. Allerdings werden im Folgenden nur ungewichtete Graphen behandelt. In diesem Falle bildet ein Lemma von Peleg und Schäffer [11] von 1989 eine nützliche Charakterisierung eines k-Spanners: S ist genau dann ein k-Spanner von G, wenn für alle Kanten $\{u, v\} \in E \setminus E'$ gilt, dass in S ein Pfad p zwischen u und v existiert, der aus höchstens k Kanten besteht. Die in S fehlende Kante $\{u, v\}$ wird als von p *gespannt* bezeichnet.

Vor diesem Hintergrund ergibt in ungewichteten Graphen nur die Betrachtung von ganzzahligen k einen Sinn. Da alle Distanzen in G ganzzahlig sind, ist ein k-Spanner von G auch ein $\lfloor k \rfloor$-Spanner von G: Offenbar wird eine in S fehlende Kante $\{u, v\}$, wenn sie in S durch einen Pfad der Länge k gespannt wird, auch durch einen Pfad der Länge $\lfloor k \rfloor$ in S gespannt.

Der einzige 1-Spanner, den ein Graph G besitzt, ist G selbst. Betrachtenswert ist daher nur die Suche nach einem k-Spanner mit $k \geq 2$.

Jeder k-Spanner von G ist natürlich auch ein k'-Spanner von G für alle $k' \geq k$. Da jeder Graph G ein 1-Spanner für sich selbst ist, bildet G auch einen k-Spanner für sich selbst mit allen $k \geq 1$. Insofern ist die Suche nach einem beliebigen k-Spanner in einem Graphen G trivial. Interessant und damit auch schwieriger ist vielmehr die Bestimmung von k-Spannern, die aus möglichst wenigen Kanten bestehen. Ein *sparsest-k-spanner* von G ist ein k-Spanner von G mit der minimalen Anzahl von Kanten.

Peleg und Schäffer [11] zeigen, dass die Bestimmung eines sparsest-2-spanners für einen Graphen $\mathcal{NP}$-schwer ist. Cai [3] weitet dies aus und kennzeichnet die Berechnung eines sparsest-k-spanners als $\mathcal{NP}$-vollständig für alle $k \geq 2$.

Ein möglicher Weg, sich $\mathcal{NP}$-vollständigen Problemen zu nähern, stellen polynomielle Approximationsalgorithmen dar, die, wenn schon nicht die optimale Lösung, so doch wenigstens eine hinreichend gute Lösung berechnen. Doch dieser Weg ist hier wenig Erfolg versprechend: Nach Kortsarz [9] ist das Problem

der Suche eines sparsest-k-spanners für allgemeine ungewichtete Graphen für alle $k \geq 2$ so schwer zu approximieren wie das SetCoverProblem. Damit existiert ein konstantes $c > 0$, so dass ein sparsest-k-spanner in polynomieller Laufzeit mit einer Güte von $c \cdot \log |V|$ nicht approximierbar ist, falls $\mathcal{P} \neq \mathcal{NP}$. Bis auf einen konstanten Faktor wird diese Schranke von Kortsarz und Peleg [10] erreicht, die einen polynomiellen Algorithmus angeben, der für allgemeine ungewichtete und ungerichtete Graphen einen sparsest-2-spanner mit einer Güte von $\Theta\left(\log \frac{|E|}{|V|}\right)$ approximiert.

Die Hartnäckigkeit des Problems lenkt das Interesse auf eingeschränkte Graphenklassen. Beispielsweise kennzeichnet Venkatesan et al. [15] das Problem der Berechnung eines sparsest-k-spanners mit $k \geq 2$ für chordale Graphen ebenso als $\mathcal{NP}$-vollständig wie für Graphen mit begrenztem Knotengrad. Auch für bipartite Graphen wird abseits des trivialen Falles $k = 2$ für alle $k \geq 3$ $\mathcal{NP}$-Vollständigkeit gezeigt. Auf der anderen Seite wird die Bestimmung eines sparsest-k-spanners für $k \geq 3$ auf Intervall- und Split-Graphen als polynomiell lösbar dargestellt.

Eine natürliche Unterklasse von Graphen bilden die planaren Graphen. k-Spanner auf planaren Graphen sind für die Routenplanung von Interesse: Straßenkarten lassen sich im wesentlichen durch planare Graphen darstellen. Ein Spanner einer Straßenkarte hat tendenziell weniger Kanten als die Straßenkarte selbst und erlaubt damit die schnellere Beantwortung von Routing-Anfragen. Gleichzeitig kann ein Spanner die Distanzen der Straßenkarte genügend gut widerspiegeln um die berechneten Antworten nicht zu weit von einer optimalen Lösung abweichen zu lassen. Brandes und Handke [2] zeigen, dass die Berechnung eines sparsest-k-spanners für $k \geq 5$ auf planaren Graphen $\mathcal{NP}$-vollständig ist. Dagegen ist die Komplexität des Problems für $k \in \{2, 3, 4\}$ auf planaren Graphen bis heute unbekannt.

Der schon erwähnte Algorithmus von Kortsarz und Peleg zur Approximation eines sparsest-2-spanners in allgemeinen Graphen liefert auf planaren Graphen eine konstante Approximationsgüte. Allerdings ist eine konstante Approximationsgüte auf planaren Graphen auch trivial erreichbar: Ein sparsest-2-spanner eines zusammenhängenden Graphen G muss selbst zusammenhängend und darum mindestens ein Baum mit $n - 1$ Kanten sein. Da G als planarer Graph höchstens $3n - 6$ Kanten enthält, bildet G an sich schon eine Approximation eines sparsest-2-spanners der Güte höchstens $3 - \varepsilon$. Ein Algorithmus zur Berechnung eines 2-Spanners in einem planaren Graphen kann nur dann von Interesse sein, wenn er eine Approximationsgüte von deutlich unter 3 liefert. Dies trifft auf den Algorithmus von Kortsarz und Peleg nicht zu.

Duckworth et al. [6] beschäftigen sich mit 4-fach zusammenhängenden planaren Triangulationen, also planare Triangulationen ohne trennende Dreiecke. Für die Bestimmung eines 2-Spanners auf dieser Graphenklasse wird ein PTAS angegeben und damit die Möglichkeit, einen 2-Spanner in polynomieller Laufzeit beliebig gut zu approximieren. Zu diesem Zweck wird das Problem der Bestimmung eines sparsest-2-spanners von G in das Problem der Bestimmung eines größten induzierten Matchings im face-edge-incidence Graph G' von G überführt. Die-

ses wiederum kann durch die Berechnung eines MAXIMUM2-INDEPENDENTSET im Linegraphen von $G^{'}$ gelöst werden. Ist G eine 4-fach zusammenhängende planare Triangulation, so ist der Linegraph von $G^{'}$ planar und erlaubt damit ein PTAS für die Bestimmung eines MAXIMUM2-INDEPENDENTSET.

Die vorliegende Arbeit entwickelt einen Algorithmus zur Berechnung eines 2-Spanners in allgemeinen planaren Triangulationen mit polynomieller Laufzeit. Damit wird über den Algorithmus von Duckworth et al. hinausgegangen, auch trennende Dreiecke sind in der zu bearbeitenden planaren Triangulation erlaubt. Der Algorithmus erreicht dabei eine Approximationsgüte von $\frac{5}{3} - \varepsilon$.

2 Definitionen und Notationen

Es wird davon ausgegangen, dass der Leser mit den grundlegenden Notationen und Definitionen der Graphentheorie vertraut ist. Um Unklarheiten zu vermeiden, wird dennoch eine kurze Einführung der benötigten Begriffe gegeben. Für eine vertiefende Beschäftigung sei an die einschlägige Literatur zur Graphentheorie verwiesen, beispielsweise [5].

Ein *Graph* $G = (V, E)$ ist ein Paar bestehend aus einer endlichen Menge V und $E \subseteq \binom{V}{2}$. Die Elemente von V heißen *Knoten*, die Elemente von E werden *Kanten* genannt. Alle Graphen in der vorliegenden Arbeit sind ungewichtet und ungerichtet, haben keine Mehrfachkanten oder Schleifen. Die Anzahl der Knoten $|V|$ wird auch als n bezeichnet. Ein Graph heißt *vollständig*, falls $E = \binom{V}{2}$, ein solcher Graph wird mit K_n bezeichnet.

Zwei Knoten $x, y \in V$ heißen *adjazent*, wenn $\{x, y\} \in E$, die Knoten sind dann durch ein Kante *verbunden*. Abkürzend kann die Kante auch xy genannt werden. Die *Nachbarschaft* eines Knotens x bezeichnet die Menge $N(x) := \{y \in V | \{x, y\} \in E\}$, die Elemente von $N(x)$ werden die *Nachbarn* von x genannt. Die Anzahl der Knoten, mit denen x benachbart ist, also $|N(x)|$, wird als *Grad* des Knotens x bezeichnet und mit $d(x)$ abgekürzt.

Ein Graph $G' = (V', E')$ heißt *Subgraph* eines Graphen $G = (V, E)$, falls $V' \subseteq V$ und $E' \subseteq \binom{V'}{2} \cap E$. Gilt dabei sogar $E' = \binom{V'}{2} \cap E$, so wird G' *induzierter* Subgraph genannt. Ist G' Subgraph von G und ist $V' = V$, so ist G' *spannender* Subgraph von G.

Ein *Pfad* in einem Graphen ist ein Folge von Knoten $x_0, x_1, \ldots, x_l$, wobei alle Knoten paarweise verschieden und aufeinander folgende Knoten adjazent sind. Die Anzahl der Kanten eines Pfades, beim Pfad $x_0, x_1, \ldots, x_l$ sind dies l Stück, wird als *Länge* des Pfades bezeichnet. Die Länge eines kürzesten Pfades zwischen zwei Knoten x und y beschreibt den *Abstand* oder die *Distanz* $dist(x, y)$ der Knoten x und y. Ist $x_0, \ldots, x_{l-1}$ $l \geq 3$ ein Pfad, so bildet der Graph bestehend aus dem Pfad zuzüglich der Kante $x_{l-1}x_0$ einen Kreis der Länge l, die der Anzahl der Kanten (oder Knoten) im Kreis entspricht. Ein *Dreieck* ist ein Kreis der Länge drei.

Gibt es für je zwei Knoten x, y in G einen Pfad zwischen x und y, so ist G *zusammenhängend*. Ein inklusionsmaximaler zusammenhängender Subgraph von G heißt *Zusammenhangskomponente* von G. Alle im Folgenden betrachteten Graphen werden als zusammenhängend angesehen. Ist dies für einen Graphen G nicht der Fall, können die Zusammenhangskomponenten von G einzeln betrachtet werden.

Ein Graph G wird *k-fach zusammenhängend* genannt, falls G mindestens $k + 1$ Knoten enthält und nach dem Entfernen einer Menge von beliebigen k Knoten zusammenhängend ist.

Ein Graph G kann in einer Ebene dargestellt werden, so dass Knoten von G als Punkte und Kanten von G als Linien abgebildet werden. Dabei verläuft jede

Linie für eine Kante xy genau zwischen den beiden Punkten, die die Knoten x und y in der Ebene repräsentieren. Kann ein Graph G in dieser Form so in der Ebene dargestellt werden, dass sich je zwei Linien höchstens in einem gemeinsamen Knoten schneiden, so ist G ein *planarer* Graph. Ein *eingebetteter* planarer Graph G besteht aus dem Graphen selbst und der *Einbettung* von G, also der Zuweisung von Knoten und Kanten von G zu Koordinaten in der Ebene.

Die Kanten eines eingebetteten planaren Graphen zerlegen die Ebene in zusammenhängende Regionen, die *Gebiete* genannt werden. Alle Gebiete bis auf eines sind beschränkt, das eine unbeschränkte Gebiet heißt *äußeres Gebiet*. Ein Graph bildet eine *planare Triangulation*, falls alle Gebiete, auch das äußere Gebiet, durch ein Dreieck begrenzt werden. Entsprechend kann ein planarer Graph G *trianguliert* werden: Es werden so lange Kanten zwischen den Knoten von G eingefügt, bis keine weitere Kante mehr eingefügt werden kann, ohne die Planarität von G zu verletzten. In diesem Sinne ist eine planare Triangulation ein planarer Graph mit der maximalen Anzahl von Kanten.

Neben Begriffen aus der Graphentheorie findet der Leser vorliegender Arbeit auch in der Komplexitätstheorie beheimatetes Vokabular. Davon wird an dieser Stelle lediglich die Approximationsgüte eines Algorithmus definiert. Für eingehendere Betrachtungen zum Thema der Komplexitätstheorie sei [7] empfohlen.

Bezeichne $OPT(I)$ die optimale Lösung eines Minimierungsproblems zu einer gegebenen Instanz I und sei $A(I)$ die Lösung, die ein Algorithmus A für die Instanz I berechnet. Dann heißt der Wert

$$\rho(A) := \sup_I \left\{ \frac{A(I)}{OPT(I)} \right\}$$

die *Approximationsgüte* des Algorithmus A. Demnach bildet die Approximationsgüte eines Algorithmus A eine obere Schranke für den Faktor, um den eine Lösung von A von der optimalen Lösung abweichen kann.

3 Planare Triangulationen und trennende Dreiecke

Der Ansatz von Duckworth et al. [6] erlaubt die Approximation eines 2-Spanners auf einer eingebetteten planaren Triangulation. Die gegebene Triangulation muss dazu allerdings 4-fach zusammenhängend sein, darf also insbesondere keine trennenden Dreiecke enthalten. Diese Forderung kann für den in dieser Arbeit vorgestellten Algorithmus fallen gelassen werden. Die Existenz von trennenden Dreiecken in einer planaren Triangulation macht nicht nur keine Schwierigkeiten. Vielmehr ist die Behandlung von trennenden Dreiecken im Sinne des zu berechnenden 2-Spanners zentraler Bestandteil des Algorithmus.

Der Algorithmus nutzt dabei Charakteristika von planaren Triangulationen und trennenden Dreiecken aus. Diese Eigenschaften werden im vorliegenden Abschnitt vorgestellt.

Zuerst seien einige bekannte Kennzeichen einer planaren Triangulation ohne Beweis aufgeführt, siehe dazu jeweils bspw. [5]: Eine planare Triangulation G mit n Knoten besteht aus genau $3n-6$ Kanten und ist 3-fach zusammenhängend. Jede Kante liegt in mindestens zwei Dreiecken und gehört zu genau zwei Gebieten. Insgesamt gibt es $2n-4$ Gebiete, die alle von genau einem Dreieck umrandet werden.

Ein *trennendes Dreieck* D in G ist ein Dreieck nach dessen Entfernung G nicht mehr zusammenhängend ist. In einer eingebetteten planaren Triangulation liegen Knoten und Kanten von G sowohl außerhalb als auch innerhalb von D. Eine Entfernung von D ergibt zwei Zusammenhangskomponenten von G, bestehend aus Knoten und Kanten außerhalb und innerhalb von D.

Unter Verwendung einer Proposition aus [5] lässt sich angeben, wie viele der Dreiecke, in denen eine Kante liegt, trennende Dreiecke sind.

Proposition 3.1 (Diestel [5]) *Die Grenzen eines Gebietes in einem 3-fach zusammenhängenden planaren Graphen sind genau die induzierten Kreise, die keine trennenden Kreise sind.*

Korollar 3.2 *Von den d Dreiecken, in denen eine Kante in einer planaren Triangulation liegt, sind genau $d-2$ Dreiecke trennend.*

Beweis
Da jede Kante e zur Umrandung von genau zwei Gebieten gehört und jedes Gebiet von einem Dreieck umrandet wird, bilden auch nur genau zwei der d Dreiecke, in denen e liegt die Grenze eines Gebietes. Die restlichen $d-2$ Dreiecke, die gleichzeitig induzierte Kreise sind, in denen e liegt, umranden kein Gebiet und sind damit nach Proposition 3.1 trennende Dreiecke.

□

Ist eine planare Triangulation 4-fach zusammenhängend, so enthält sie gar keine trennenden Dreiecke. Die untere Schranke für die Anzahl von trennenden

Dreiecken in einer planaren Triangulation ist damit gleich Null. Das folgende Lemma beantwortet die naheliegende Frage nach einer oberen Schranke für diese Anzahl.

Lemma 3.3 *In einer planaren Triangulation befinden sich höchstens $n-4$ trennende Dreiecke.*

Beweis
Zuerst wird eine obere Schranke für die Anzahl aller Dreiecke in einer planaren Triangulation G berechnet. Dazu wird bestimmt, in wie vielen Dreiecken ein Knoten v von G mit Knotengrad $d(v)$ liegen kann: Betrachte den Stern S, der sich als Subgraph von G aus v verbunden mit seinen Nachbarn ergibt. S hat $d(v)+1$ Knoten und $d(v)$ Kanten. Wird S trianguliert ohne einen Knoten einzufügen, so ergibt sich ein Graph $S^{'}$ mit $3(d(v)+1)-6$ Kanten, es können also höchstens $3(d(v)+1)-6-d(v) = 2d(v)-3$ Kanten zu S hinzugefügt werden, ohne die Planaritätsbedingung zu verletzen. Jede im Zuge dieser Triangulierung zu S hinzugefügte Kante erhöht die Anzahl von Dreiecken, in denen v liegt um eins. Wenn $S^{'}$ entstanden ist, liegt v in $2d(v)-3$ Dreiecken. Offenbar ist dies die Höchstanzahl von Dreiecken, in denen ein Knoten v vom Grad $d(v)$ in einer planaren Triangulation liegen kann.

Die Addition dieses Wertes über alle Knoten von G ergibt eine obere Schranke für die Anzahl aller Dreiecke in G. Allerdings ist zu beachten, dass jedes Dreieck in G auf diese Weise genau dreimal gezählt wird, die Summe also entsprechend zu dritteln ist. Außerdem wird zur Abschätzung verwendet, dass die Summe aller Knotengrade der doppelten Kantenanzahl entspricht.

$$\begin{aligned}
\text{Anzahl Dreiecke in } G &\leq \frac{1}{3}\sum_{v\in V}\Big(2d(v)-3\Big) \\
&= \frac{2}{3}\sum_{v\in V}\Big(d(v)\Big) - \frac{3n}{3} \\
&= \frac{2}{3}(6n-12) - n \\
&= 3n-8
\end{aligned}$$

Jedes Gebiet von G wird von genau einem Dreieck umrandet. Bis auf die hier vernachlässigte Ausnahme, dass G nur aus einem Dreieck besteht, werden keine zwei Gebiete von G vom gleichen Dreieck umrandet. Damit gibt es in G genau $2n-4$ Dreiecke, die ein Gebiet umranden und nach Proposition 3.1 keine trennenden Dreiecke sind. Eine Anzahl von höchstens $3n-8-(2n-4) = n-4$ Dreiecken in G sind damit trennende Dreiecke.

□

Es bleibt anzumerken, dass diese obere Schranke scharf ist, also auch von einer planaren Triangulation erreicht werden kann. Interessanterweise ist dies dann der Fall, wenn der sparsest-2-spanner von G ein Baum ist, siehe dazu Lemma 3.7.

Trennende Dreiecke können ineinander geschachtelt sein, innerhalb eines trennenden Dreieckes können ein oder mehrere andere trennende Dreiecke liegen. In diesem Sinne bauen die trennenden Dreiecke eine Baumstruktur auf: Ein trennendes Dreieck B, das in der planaren Triangulation innerhalb eines trennenden Dreieckes A liegt, befindet sich in der Baumstruktur im Unterbaum mit der Wurzel A. B liegt *unter* oder *unterhalb von* A, analog befindet sich A *über* oder *oberhalb* von B. B ist Sohn von A in der Baumstruktur, wenn B innerhalb von A in der planaren Triangulation liegt, es aber dort kein trennendes Dreieck C gibt, so dass C innerhalb von A und B innerhalb von C liegt. In diesem Falle liegt B *direkt unter* oder *direkt unterhalb* von A, entsprechend liegt A *direkt über* oder *direkt oberhalb* von B.

Auf der untersten Ebene des Baumes, den Blättern, liegen die trennenden Dreiecke, die selber keine solchen mehr enthalten, auf der obersten Ebene des Baumes, der Wurzel, befindet sich ein trennendes Dreieck, das in keinem anderen trennenden Dreieck mehr liegt. Gibt es mehrere trennende Dreiecke, die in keinem anderen trennenden Dreieck mehr liegen, so besteht die Baumstruktur aus mehreren Bäumen. Die obersten trennenden Dreiecke, die jeweils die Wurzel eines Baumes darstellen, liegen nur noch innerhalb des Dreieckes, das den Rand des äußeren Gebietes der planaren Triangulation bildet.

Die drei Kanten eines trennenden Dreieckes D werden als *Außenkanten* oder *äußere Kanten* von D bezeichnet. Die *Innenkanten* oder *innere Kanten* von D werden mithilfe der Baumstruktur der trennenden Dreiecke rekursiv definiert: Ist D ein Blatt in der Baumstruktur, so sind alle Kanten, die in der planaren Triangulation innerhalb von D liegen, Innenkanten von D. Ist D kein Blatt in der Baumstruktur der trennenden Dreiecke, dann sind die Innenkanten von D die Kanten, die in der planaren Triangulation innerhalb von D liegen aber nicht schon Innenkanten von unterhalb von D liegenden trennenden Dreiecken sind.

Für ein trennendes Dreieck D wird D_I definiert als Subgraph von G bestehend aus Innen- und Außenkanten von D. Ist im Folgenden abkürzend von einem trennenden Dreieck D_I die Rede, so ist damit immer der Subgraph aus Außenkanten und Innenkanten des trennenden Dreieckes D gemeint. Wann immer im Folgenden der Index I bei dieser Bezeichnung auftaucht, zeigt er damit an, dass neben den Kanten von D selbst (Außenkanten von D) immer auch die Innenkanten von D gemeint sind.

Zur Illustration der Baumstruktur, Außen- und Innenkanten von trennenden Dreiecken ist Abbildung 1 angegeben. Dargestellt ist ein Ausschnitt aus einer planaren Triangulation, der drei trennende Dreiecke D, E und F enthält. Die trennenden Dreiecke liegen in einer Baumstruktur, die rechts oben in der Abbildung 1 gezeigt ist: Oberhalb von D befindet sich kein trennendes Dreieck mehr, damit bildet D eine Wurzel in der Baumstruktur der trennenden Dreiecke. E liegt unterhalb der Wurzel D, F unterhalb von E. Da unterhalb von F kein trennendes Dreieck mehr liegt, bildet F ein Blatt in der Baumstruktur.

So ergeben sich die Innenkanten der trennenden Dreiecke nach Definition. F

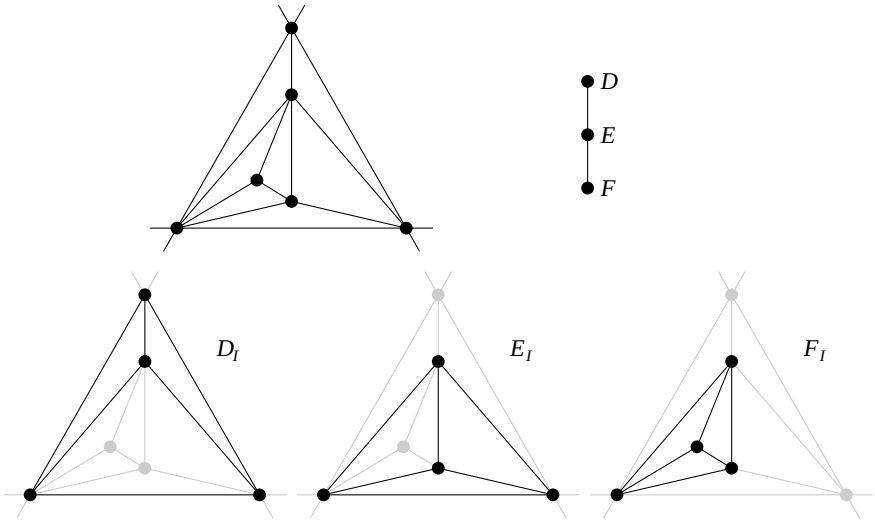


Abbildung 1: Ausschnitt einer planaren Triangulation, der aus drei trennenden Dreiecken und ihren Innenkanten besteht. Die drei trennenden Dreiecke liegen in der abgebildeten Baumstruktur zueinander.

als Blatt zählt alle Kanten in seinem Inneren als Innenkanten. Die Innenkanten von E dagegen ergeben sich aus den Kanten, die innerhalb des trennenden Dreieckes E liegen, aber keine Innenkanten von F sind. Ebenso sind weder Innenkanten von F noch von E Innenkanten von D. Auf diese Weise ergeben sich für die trennenden Dreiecke D, E und F die Graphen D_I, E_I und F_I wie dargestellt, jeweils bestehend aus drei Außen- und drei Innenkanten.

Durch die Charakteristika einer planaren Triangulation und die Definition von D_I ergeben sich zwei später benötigte Konsequenzen, die in den beiden folgenden Lemmata aufgeführt werden.

Lemma 3.4

(1) Nur höchstens eine der drei Außenkanten eines trennenden Dreieckes D_I ist auch Außenkante oberhalb von D_I liegender trennender Dreiecke.

(2) Die drei Außenkanten eines trennenden Dreieckes D_I bilden Innenkanten in höchstens zwei verschiedenen trennenden Dreiecken oberhalb von D_I.

Beweis(1)
Gibt es kein trennendes Dreieck oberhalb von D_I, so ist nichts zu zeigen, ansonsten bezeichne $\widehat{D_I}$ das trennende Dreieck direkt oberhalb von D_I. Es kann höchstens eine Außenkante von D_I auch Außenkante von $\widehat{D_I}$ sein, da zwei Dreiecke nur höchstens eine Kante gemeinsam haben können, so auch die Dreiecke

jeweils bestehend aus den Außenkanten von $\widehat{D_I}$ und D_I. Mindestens zwei Außenkanten von D_I sind also Innenkanten von $\widehat{D_I}$. Kein oberhalb von $\widehat{D_I}$ liegendes trennendes Dreieck hat als Außenkante eine Innenkante von $\widehat{D_I}$. Damit ist höchstens eine Außenkante von D_I auch Außenkante von oberhalb von D_I liegenden trennenden Dreiecken.

Beweis(2)
Es ist klar, dass jede Außenkante a von D_I in höchstens einem trennenden Dreieck $\widetilde{D_{I,a}}$ oberhalb von D_I Innenkante ist, da in allen oberhalb von $\widetilde{D_{I,a}}$ liegenden trennenden Dreiecken a nicht mehr enthalten ist. Da wie im Beweis zu (1) gezeigt mindestens zwei Außenkanten von D_I Innenkanten des selben trennenden Dreieckes direkt oberhalb von D_I sind, kann lediglich die dritte Außenkante von D_I in einem zweiten trennenden Dreieck als Innenkante auftreten.

□

Lemma 3.5 *Für jedes trennende Dreieck D in einer planaren Triangulation G bildet der Subgraph D_I eine planare Triangulation, in der jede Kante in genau zwei Dreiecken liegt.*

Beweis
Betrachte D_I als aus G entstanden, indem die Knoten und Kanten aus $G - D_I$ aus G entfernt wurden. In G als planarer Triangulation sind alle Gebiete durch ein Dreieck umrandet. Durch das Entfernen der Knoten und Kanten aus $G - D_I$ bleibt D_I natürlich planar. Weiterhin kann durch diese Entfernung kein Gebiet in D_I entstehen, das durch einen Kreis größer als ein Dreieck begrenzt ist: Das Entfernen von Kanten, die außerhalb von D_I liegen, können die Gebiete von D_I nicht beeinflussen. Kanten aus $G - D_I$, die innerhalb von D_I liegen, müssen Innenkanten von trennenden Dreiecken unterhalb von D_I sein. Ihr Entfernen kann lediglich Gebiete in D_I zurücklassen, die durch ein Dreieck begrenzt sind.

Jede Kante von G liegt in mindestens zwei Dreiecken. Eine Kante e in D_I wird durch die Entfernung $G - D_I$ aus G nur genau der Dreiecke beraubt, die dazu führen, dass e in mehr als zwei Dreiecken liegt. Damit bildet D_I eine planare Triangulation, in der jede Kante in genau zwei Dreiecken liegt.

□

Da alle Kanten von D_I als planarer Triangulation nach Lemma 3.5 in genau zwei Dreiecken liegen, heißt dies, dass D_I neben drei Außenkanten mindestens drei Innenkanten besitzen muss. In diesem minimalen Falle ist D_I eindeutig, bildet nämlich einen K_4. Aber auch die nächstgrößere Gestalt, die D_I annehmen kann ist eindeutig und wird durch das folgende Lemma beschrieben.

Lemma 3.6 *Bildet D_I keinen K_4, so besitzt D_I mindestens neun Innenkanten. Hat D_I genau neun Innenkanten, so ist D_I von der in Abbildung 2 gezeigten Gestalt.*

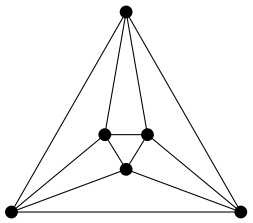


Abbildung 2: Oktaeder

Beweis
Betrache dazu Abbildung 3. Sie zeigt die drei Außenkanten a, b und c und einen Teil der Innenkanten von D_I. Abgebildet sind genau die Innenkanten von D_I, die mit einer Außenkante von D_I ein Dreieck bilden.

Betrachte die Innenkanten e und h. Wären diese beiden Innenkanten identisch, so würden die Kanten b, d und i ein Dreieck bilden und D_I wäre ein K_4: Jede weitere Innenkante von D_I außer $e = h$, d und i läge in einem der trennenden Dreiecke *ade*, *cei* oder *bdi*. Damit wäre diese Kante keine Kante von D_I, sondern Innenkante eines unterhalb von D_I liegenden trennenden Dreieckes. Ist D_I also kein K_4, so muss $e \neq h$, genauso wie $d \neq f$ und $g \neq i$ sein. Ist D_I kein K_4, so enthält D_I also mindestens die abgebildeten Innenkanten. Da nach Lemma 3.5 jede Kante von D_I, Innen- wie Außenkante, in genau zwei Dreiecken liegt, bildet D_I selbst eine planare Triangulation. Der Knoten x kann nicht mit dem Knoten y adjazent sein, sonst wäre D_I ein K_4. Die einzige Möglichkeit, den Graphen in der Abbildung 3 zu triangulieren ohne einen Knoten hinzu zufügen, führt dann auf die im Lemma angesprochene Gestalt.

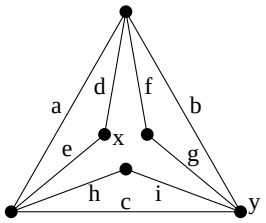


Abbildung 3: Außenkanten und ein Teil der Innenkanten von D_I

□

Das folgende Lemma zeigt, dass aus der Existenz eines hinreichend kleinen 2-Spanners in einer planaren Triangulation G auf eine Mindestanzahl von trennenden Dreiecken in G geschlossen werden kann. Für den späteren Beweis der Approximationsgüte wird das Lemma mit dem kleinstmöglichen 2-Spanner, dem sparsest-2-spanner, von G angewendet und daraus eine untere Schranke für die Anzahl trennender Dreiecke von G gefolgert.

Lemma 3.7 *Sei G eine planare Triangulation mit n Knoten und $3n-6$ Kanten, S sei ein 2-Spanner von G. Enthält S $\frac{3n-6}{2}-k$ Kanten, so besitzt G mindestens $2k$ trennende Dreiecke.*

Beweis
Als Beweis wird ein Prozess beobachtet, der sukzessive in S alle Kanten aus $G-S$ einfügt. Die bei diesem Prozess entstehende Anzahl trennender Dreiecke wird abgeschätzt und dies entspricht am Ende des Prozesses einer unteren Schranke für die Anzahl trennender Dreiecke in G. Wenn $k \leq 0$ ist, so ist offenbar nichts zu zeigen, es gelte in der Folge also $k > 0$.

Betrachte am Beginn des Prozesses alle Kanten von S als rot gefärbt, alle Kanten aus $G - S$, die nacheinander eingefügt werden, als blau gefärbt. Ein Dreieck aus zwei roten und einer blauen Kante in G wird als *brr-Dreieck* bezeichnet. Zusätzlich sei jede rote Kante mit einem Zähler ausgestattet, blaue Kanten haben keinen Zähler. Anfangs ist jeder Zähler mit null initialisiert.

Wird eine blaue Kante b aus $G-S$ eingefügt, so muss sich dabei mindestens ein brr-Dreieck bilden. Es besteht aus zwei roten Kanten r_1 und r_2 und eben der blauen Kante b: b ist über zwei Kanten in S gespannt worden, die beiden roten Kanten r_1 und r_2 aus S, die mit b in einem Dreieck liegen, müssen also existieren und das brr-Dreieck r_1r_2b muss entstehen. Bilden sich mehrere brr-Dreiecke beim Einfügen der Kante b, so wähle genau eines von diesen und seien dessen rote Kanten r_1 und r_2. Die Zähler von r_1 und r_2 werden in diesem Schritt um jeweils eins erhöht. Eventuell bilden sich beim Einfügen von b noch weitere Dreiecke, möglicherweise auch weitere brr-Dreiecke aus zwei anderen roten Kanten und b. Dies wäre dann der Fall, wenn in S mehrere 2-Pfade existieren, die b spannen. Wichtig ist hierbei, dass nur die Zähler von r_1 und r_2 inkrementiert werden, die Zähler anderer roter Kanten, auch wenn diese mit b ein brr-Dreieck bilden, bleiben unverändert.

Da S aus $\frac{3n-6}{2}-k$ Kanten besteht, werden im Verlaufe des Prozesses $\frac{3n-6}{2}+k$ Kanten eingefügt. Jede eingefügte Kante erhöht genau zwei Zähler um eins. Wenn am Ende des Prozesses G aus roten und blauen Kanten aufgebaut wurde, muss die Summe der Zähler über alle roten Kanten deshalb $3n-6+2k$ betragen.

Offensichtlich gibt jeder Zähler einer roten Kante r eine untere Schranke dafür an, in wie vielen brr-Dreiecken r liegt. Eine Kante einer planaren Triangulation, die in d Dreiecken liegt, muss nach Korollar 3.2 in $d-2$ trennenden Dreiecken liegen. Wird nun vom Zähler einer jeden roten Kante ein Betrag von zwei subtrahiert (Zähler, die echt kleiner als zwei sind, werden auf null gesetzt), zeigt der Zähler von r nach dieser Subtraktion an, in wie vielen trennenden brr-Dreiecken r liegt. Dies ist sogar eine untere Schranke: brr-Dreiecke, die während des Prozesses des Kanteneinfügens an dieser Kante gebildet wurden, ohne den Zähler der Kante zu erhöhen, drücken sich im Zähler gar nicht aus. Die Summe der so subtrahierten Beträge über alle Zähler ist höchstens $3n-6-2k$, kann sogar noch kleiner sein, da von Zählern kleiner als zwei ein kleinerer Betrag abgezogen wurde. Die Summe der nach diesen Subtraktionen entstandenen Zählerwerte über alle Zähler ist damit mindestens so groß:

$$3n-6+2k-(3n-6-2k)=4k$$

Der Wert des Zählers einer jeden roten Kante r gibt nach der Subtraktion eine untere Schranke dafür an, wie viele von den brr-Dreiecken, in denen r liegt, trennende Dreiecke sind. Die Summe der Zähler ist mindestens $4k$ und jedes brr-Dreieck führte höchstens dazu, dass die Zähler seiner beiden roten Kanten um eins erhöht wurden (eventuell gibt es brr-Dreiecke, die gar keinen Zähler erhöht haben). Jedes brr-Dreieck trägt also höchstens einen Betrag von zwei zu dieser Summe der Zähler bei. Beim Prozess des Kanteneinfügens müssen also mindestens $4k/2 = 2k$ trennende brr-Dreiecke entstanden sein. Damit enthält G mindestens $2k$ trennende Dreiecke.

□

Wenn S kleinstmöglich und damit ein Baum ist, so ist die eben bewiesene Schranke scharf: Hat S $n-1$ Kanten, so ist $k = \frac{n}{2} - 2$ und G enthält nach Aussage mindestens $2k = n - 4$ trennende Dreiecke. Nach Lemma 3.3 ist diese Anzahl größtmöglich. Damit enthält G im Falle, dass S ein Baum ist, genau $n - 4$ trennende Dreiecke und so das Maximum an Dreiecken, das eine planare Triangulation überhaupt enthalten kann.

4 Ein einfacher 2-Spanner Algorithmus

In diesem Abschnitt wird ein simpler Algorithmus `A_einfach` vorgestellt, der einen 2-Spanner auf einer planaren Triangulation berechnet. Mit seiner Hilfe soll verdeutlicht werden, wie sich ein nicht trivialer 2-Spanner grundsätzlich bestimmen lässt. Am Beispiel von `A_einfach` werden Begriffe eingeführt und erläutert, die auch später noch Verwendung finden. Am Ende des Abschnittes findet sich eine Aussage, die es erlaubt, die Approximationsgüte von Algorithmen abzuschätzen, die im Grunde nach dem Muster von `A_einfach` vorgehen und dabei eventuell zusätzliche Verbesserungen durchführen.

4.1 Der Algorithmus `A_einfach`

`A_einfach` entscheidet für jede Kante einer gegebenen planaren Triangulation G, entweder sie zu entfernen oder sie zum gerade berechneten 2-Spanner hinzuzufügen. Jede Kante von G, für die diese Entscheidung noch nicht getroffen wurde, wird als *freie Kante* bezeichnet. Für ein Dreieck efg kann *e über f und g gespannt werden*: e wird entfernt und f und g jeweils als Spanner-Kante gekennzeichnet, so sie es denn nicht bereits sind. Zwei Kanten e und f, die in einem Dreieck liegen, können nur in genau einem Dreieck zusammen liegen. Insofern ist die dritte Kante, mit der e und f in einem Dreieck liegen, eindeutig. Deshalb kann auch abkürzend davon gesprochen werden, dass *e über f gespannt wird*: e wird entfernt, f und die eindeutige dritte Kante, mit der e und f ein Dreieck bilden, werden als Spanner-Kanten markiert, wenn sie es nicht bereits sind. Sind die Kanten f und g eines Dreieckes efg Spanner-Kanten und ist e entfernt, so ist die Kante e *gespannt*.

Durch das Kantenentfernen von `A_einfach` werden Dreiecke zerstört. Insbesondere können so freie Kanten entstehen, die in weniger als zwei Dreiecken liegen, wie dies anfangs in G für alle Kanten noch der Fall ist. Eine freie Kante, die in genau einem Dreieck liegt, wird *kritische Kante* genannt.

```
A_einfach(G = (V, E))

  while Es existieren noch freie Kanten in G

i)      if   Es existiert eine kritische Kante k in G
        then Sei klm das Dreieck, in dem k liegt
             Entferne k
             Markiere(l), Markiere(m)
             continue

ii)     Wähle eine beliebige freie Kante f in G,
        wähle b und c als Kanten, die mit f ein Dreieck bilden
        Entferne f
        Markiere(b), Markiere(c)

  endwhile
```

`Markiere`(e)

if e ist freie Kante
then mache e zur Spanner-Kante
else mache beliebige freie Kante zur Spanner-Kante

Offenbar entfernt `A_einfach` in jedem Schritt genau eine Kante aus G. Dafür werden pro Schritt genau zwei freie Kanten von G zu Spanner-Kanten gemacht. Zwei Kanten, die mit der entfernten Kante in einem Dreieck liegen werden ausgewählt und als Spanner-Kanten markiert wenn sie es nicht bereits sind. Für jede der zwei Kanten, die bereits eine Spanner-Kante ist, wird eine beliebige andere freie Kante zur Spanner-Kante gemacht.

Die Funktionsweise eines Schrittes vom Typ ii) des Algorithmus ist in Abbildung 4 gezeigt. Sichtbar ist ein Ausschnitt einer planaren Triangulation, alle

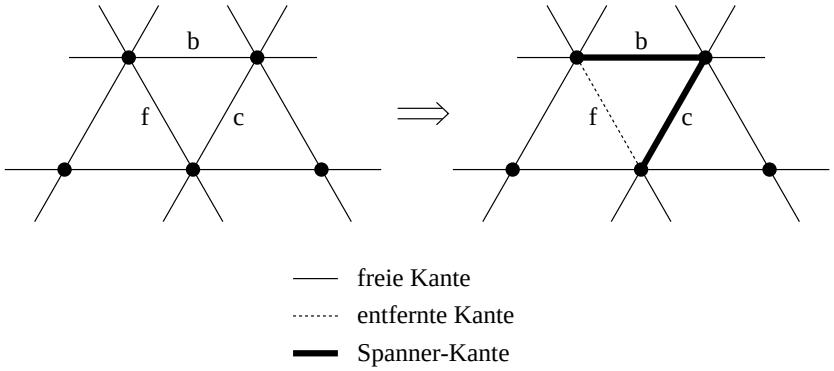


Abbildung 4: Schritt des Algorithmus `A_einfach`

im linken Teil der Abbildung befindlichen Kanten sind freie Kanten, gezeichnet mit normaler Strichstärke. Offensichtlich befinden sich alle Kanten in mindestens zwei Dreiecken, damit existiert keine kritische Kante. In einem Schritt vom Typ ii) kann `A_einfach` die freie Kante f entfernen. Im Bild ist dieses Entfernen von f durch die gestrichelte Linie angedeutet. Dabei muss der Algorithmus allerdings sicherstellen, dass f über zwei andere Kanten gespannt wird. Deshalb werden die beiden freien Kanten b und c zu Spanner-Kanten, illustriert durch die breitere Strichstärke für b und c. Die in der Abbildung benutzte Kennzeichnung von freien, entfernten und Spanner-Kanten werden wie eben eingeführt in allen folgenden Darstellungen verwendet.

4.2 Gültigkeit des berechneten Spanners

Der Algorithmus `A_einfach` entfernt in jedem Schritt genau eine Kante e aus G. Im Sinne eines zu berechnenden 2-Spanners ist dies natürlich nur dann zulässig,

wenn e gespannt ist. Dazu müssen zwei Kanten f und g, die mit e das Dreieck efg bilden, Spanner-Kanten sein. Da der Algorithmus für jede entfernte Kante genau zwei andere freie Kanten als Spanner-Kanten markiert, kann er sicherstellen, dass f und g markiert sind, wenn e entfernt wird.

Allerdings kann dies nur dann funktionieren, wenn der Algorithmus nie eine Kante entfernen will, die in keinem Dreieck liegt. Eine solche Kante könnte nicht gespannt und damit auch nicht entfernt werden.

An dieser Stelle ist die bevorzugte Behandlung von kritischen Kanten durch `A_einfach` von Bedeutung. Sie stellt sicher, dass während der Abarbeitung von G durch den Algorithmus nie eine freie Kante entsteht, die in keinem Dreieck mehr liegt. Das folgende Lemma sichert sogar jedem Algorithmus, der wie `A_einfach` kritischen Kanten einen Vorrang einräumt, diese Eigenschaft zu.

Lemma 4.1 *Sei A ein Algorithmus,*

(1) der pro Schritt eine freie Kante e entfernt und sicherstellt, dass e gespannt ist und

(2) kritische Kanten bevorzugt entfernt.

Wird A auf einer planaren Triangulation G gestartet, so entsteht nie eine freie Kante, die in keinem Dreieck mehr liegt

Beweis
Vor dem Start des Algorithmus A liegen alle Kanten einer planaren Triangulation G in mindestens zwei Dreiecken. Daher reicht es aus, für jeden Schritt von A zu zeigen, dass, wenn vor dem Schritt von A jede freie Kante von G in mindestens einem Dreieck liegt, dies auch nach dem Schritt der Fall ist.

Bei einem nicht bevorzugten Schritt von A, der eine freie nicht-kritische Kante von G entfernt, gibt es in G keine kritische Kante und alle Kanten von G liegen damit in mindestens zwei Dreiecken. Pro Schritt von A wird nur eine Kante von G entfernt, keine freie Kante e, die in mindestens zwei Dreiecken liegt, kann also in einem Schritt zwei Dreiecke verlieren, in denen e mindestens liegt. Damit gibt es auch nach einem solchen nicht bevorzugten Schritt keine freie Kante von G, die in keinem Dreieck liegt.

Bei einem bevorzugten Schritt von A wird eine kritische Kante k über das einzige Dreieck klm gespannt, in dem k liegt. Es wird dabei ausschließlich ein Dreieck zerstört, gerade das Dreieck klm. Darum können auch nur die drei Kanten k, l und m jeweils eines ihrer Dreiecke beraubt werden. Keine der drei Kanten ist nach dem Schritt aber noch freie Kante, k wird entfernt, l und m als Spanner-Kante markiert, wenn sie es nicht schon sind.

□

Korollar 4.2 *Der Algorithmus `A_einfach` berechnet für eine planare Triangulation G einen gültigen 2-Spanner.*

Beweis
Die Bedingungen des Lemmas 4.1 treffen offenbar auf `A_einfach` zu. Zu jeder Kante, die entfernt wird, kann `A_einfach` so zusichern, dass diese gespannt ist.

□

4.3 Größe des berechneten Spanners

Jeder Graph ist selber ein trivialer 2-Spanner für sich selbst. Folglich ist die Größe eines 2-Spanners als Minimierungsproblem interessant. Die Größe des von `A_einfach` berechneten 2-Spanners ist leicht anzugeben, ein Drittel aller Kanten von G werden entfernt, der berechnete 2-Spanner besteht aus genau zwei Drittel aller Kanten. Gleichwohl wird bei einem Blick auf `A_einfach` deutlich, dass Möglichkeiten zur Optimierung des berechneten 2-Spanners vernachlässigt werden. So ist es möglicherweise nicht nötig, für jede entfernte Kante zwei Kanten als Spanner-Kanten zu markieren. Ganz im Gegenteil kann ein Kante e, die mit einer bereits als Spanner-Kante ausgewählten Kante in einem Dreieck liegt, entfernt werden, ohne zwei andere Kanten markieren zu müssen, die e spannen. Die bereits bestehende Spanner-Kante kann mit genutzt werden, um e zu spannen.

Eine freie Kante, die mit einer Spanner-Kante in einem Dreieck liegt, wird *günstige* Kante genannt. Eine freie Kante kann gleichzeitig günstig und kritisch sein: Dies ist dann der Fall, wenn das einzige Dreieck, in dem die Kante liegt, eine bereits als Spanner-Kante markierte Kante enthält.

Es lässt sich nun ein Schritt eines Algorithmus vorstellen, der eine freie Kante e entfernt aber dabei, wenn e günstige Kante ist, weniger als zwei andere freie Kanten zu Spanner-Kanten macht. Wird eine Kante e von G über genau eine schon als Spanner-Kante markierte Kante gespannt, so wird nur genau eine weitere Kante dabei zu einer Spanner-Kante. Solch ein Schritt wird ein *Gewinnschritt* genannt. In Abgrenzung dazu, wird ein Schritt, der eine Kante über zwei freie Kanten spannt und damit zwei freie Kanten zu Spanner-Kanten macht als *Nichtgewinnschritt* bezeichnet. Im Sinne des Zieles, einen möglichst kleinen Spanner zu berechnen, ist ein Gewinnschritt einem Nichtgewinnschritt überlegen: Eine Kante weniger wird zu einer Spanner-Kante. Im noch günstigeren Fall, dass eine Kante über zwei bereits als Spanner-Kanten festgelegte Kanten gespannt wird, ist von einem *Zweifachgewinnschritt* die Rede. In diesem Falle wird eine Kante entfernt und keine Kante als Spanner-Kante markiert, zwei Kanten weniger, als bei einem Nichtgewinnschritt markiert würden.

Wie der Name nahelegt, entspricht ein Zweifachgewinnschritt zwei Gewinnschritten: Eine Anzahl von x Zweifachgewinnschritten machen $2x$ Kanten weniger zu Spanner-Kanten als x Nichtgewinnschritte dies tun. Um den gleichen Gewinn von $2x$ weniger markierten Spanner-Kanten zu erhalten, ist die doppelte Anzahl von $2x$ Gewinnschritten nötig. Insofern können Gewinnschritte und Zweifachgewinnschritte ineinander umgerechnet werden und treten, unter Berücksichtigung des Umrechenfaktors, als Synonym füreinander auf.

Offenbar sind sämtliche Schritte, die `A_einfach` ausführt Nichtgewinnschritte. Das Potenzial, einen kleineren 2-Spanner zu berechnen, kann so natürlich nicht ausgeschöpft werden. Erst der Algorithmus, der im nächsten Kapitel vorgestellt wird, versucht, Gewinnschritte einzusetzen, um einen tendenziell kleineren 2-Spanner zu berechnen.

Ist die Anzahl von Gewinn- und Nichtgewinnschritten bekannt, die ein Algorithmus A ausführt, kann die Größe des berechneten Spanners angegeben werden. Lemma 3.7 folgert aus der Größe eines sparsest-2-spanners in einer planaren Triangulation G eine Anzahl von trennenden Dreiecken in G. Ist ein Zusammenhang zwischen der Anzahl trennender Dreiecke und der Anzahl Gewinnschritte von A bekannt, so kann aus der Größe eines sparsest-2-spanners die Größe eines von A berechneten Spanners abgeschätzt werden. Wie sich in dieser Weise die Approximationsgüte von A folgern lässt, zeigt das folgende Lemma.

Lemma 4.3 *Sei G eine planare Triangulation mit $|D|$ trennenden Dreiecken. Sei A ein Algorithmus, der auf G einen 2-Spanner durch Nichtgewinn- und $\alpha \cdot |D|$ Gewinnschritten berechnet, $\alpha < 2$. Dann erreicht A auf G eine Approximationsgüte von $2 - \frac{\alpha}{3} - \varepsilon$.*

Beweis
Zum Abschätzen der Güte wird A mit dem Algorithmus `A_einfach` verglichen, dessen berechneter Spanner eine bekannte Größe hat. Da G aus $3n - 6$ Kanten besteht, berechnet `A_einfach` einen 2-Spanner der Größe $2n-4$, 2/3 aller Kanten werden ausschließlich durch Nichtgewinnschritte zu Spanner-Kanten gemacht. Gibt es in G eine Anzahl von $|D|$ trennenden Dreiecken, so sind demgegenüber $\alpha \cdot |D|$ Schritte von A Gewinnschritte.

`A_einfach` legt durch x Schritte $3x$ Kanten von G fest, $2x$ davon werden zu Spanner-Kanten, x Kanten entfernt. Wird die gleiche Anzahl von $3x$ Kanten von A festgelegt und gibt es dabei $3g$ Gewinnschritte, so sind nur $2x - g$ der $3x$ festgelegten Kanten als Spanner-Kanten markiert worden, $x + g$ Kanten wurden entfernt. Eine Anzahl von $3g$ Gewinnschritten von A führt also dazu, dass der letztendlich von A berechnete 2-Spanner g Kanten weniger besitzt, als ein von `A_einfach` gefundener Spanner. Die Approximationsgüte von A lässt sich damit nach oben beschränken:

$$
\begin{aligned}
\text{Approx.-güte von } A &= \frac{|\text{Spanner von } A|}{|\text{sparsest-2-spanner } S2S|} \\
&= \frac{|\text{Spanner von } \texttt{A\_einfach}|}{|\text{S2S}|} - \frac{\frac{1}{3} \cdot |\text{Gewinnschritte von } A|}{|S2S|} \\
&= \frac{\frac{2}{3} \cdot (3n-6) - \frac{1}{3} \cdot |\text{Gewinnschritte von } A|}{|S2S|} \\
&= \frac{\frac{2}{3} \cdot (3n-6) - \frac{1}{3} \cdot \alpha \cdot |\text{trennende Dreiecke}|}{|S2S|}
\end{aligned}
$$

$$\begin{aligned} & \overset{\text{Lemma 3.7}}{\leq} & \frac{\frac{2}{3} \cdot (3n-6) - \frac{1}{3} \cdot \alpha \cdot 2k}{\frac{3n-6}{2} - k} \\ & = & \frac{2n - 4 - \frac{2}{3} \cdot \alpha \cdot k}{\frac{3n-6}{2} - k} \end{aligned}$$

Der Wert des Ausdrucks steigt mit steigendem k, da $\alpha < 2$. Der Maximalwert für k ergibt sich daraus, dass ein sparsest-2-spanner mit $\frac{3n-6}{2} - k$ Kanten mindestens ein Baum sein muss und damit $n-1$ Kanten hat. Damit muss gelten, dass $\frac{3n-6}{2} - k \geq n-1$ und k kann so höchstens den Wert $\frac{n-4}{2}$ haben. Wird dieser Wert eingesetzt, ergibt sich:

$$\begin{aligned} \text{Approximationsgüte von } A & \leq & \frac{2n - 4 - \frac{2}{3} \cdot \alpha \cdot \left(\frac{n-4}{2}\right)}{\frac{3n-6}{2} - \frac{n-4}{2}} \\ & = & \frac{2n - 2 - \frac{\alpha}{3}(n-1) + \alpha - 2}{n-1} \\ & = & 2 - \frac{\alpha}{3} + \frac{\alpha - 2}{n-1} \\ & = & 2 - \frac{\alpha}{3} - \varepsilon \end{aligned}$$

□

Selbstverständlich kann das eben bewiesene Lemma auch auf den Algorithmus `A_einfach` angewendet werden. Da keine Gewinnschritte stattfinden ist $\alpha = 0$. Die vom Lemma versprochene Approximationsgüte von `A_einfach` liegt dann bei $2 - \varepsilon$. Es ist direkt ersichtlich, dass diese Güte tatsächlich auf `A_einfach` zutrifft und der Beweis des vorigen Lemmas in diesem Falle scharf ist. Der von `A_einfach` berechnete 2-Spanner hat auf jeder planaren Triangulation G die Größe $2n-4$. Der sparsest-2-spanner von G hat mindestens $n-1$ Kanten, was ein Approximationsgüte von $2 - \varepsilon$ für `A_einfach` ergibt.

5 Approximationsalgorithmus mit Güte $\frac{5}{3} - \varepsilon$

Thema dieses Abschnittes ist der Algorithmus `Berechne2Spanner`, der in polynomieller Laufzeit auf einer planaren Triangulation einen 2-Spanner bestimmt und dabei eine Approximationsgüte von $\frac{5}{3} - \varepsilon$ erreicht. Eine Vielzahl von Definitionen und Zusammenhängen, die in vorigen Abschnitten eingeführt und erläutert wurden, finden hier Verwendung. Auf dieser Grundlage wird die Arbeitsweise des Algorithmus analysiert und gezeigt, dass die versprochene Approximationsgüte erreicht wird.

Allerdings sei an eine Definition an dieser Stelle ausdrücklich erinnert: Die Bezeichnung D_I steht, auch wenn nur abkürzend vom trennenden Dreieck D_I die Rede ist, immer für die Außen- und Innenkanten eines trennenden Dreieckes D.

5.1 Der Algorithmus `Berechne2Spanner`

`Berechne2Spanner`$(G = (V, E))$

i) Ermittle die Baumstruktur der trennenden Dreiecke in G
Stelle dabei fest, welche trennenden Dreiecke D_I einen K_4 bilden und welche trennenden Dreiecke gemeinsame Außenkanten haben

ii) Starte die Routine `BearbeiteTrennendesDreieck` nacheinander auf allen trennenden Dreiecken D_I, die Wurzeln eines Baumes in der Baumstruktur der trennenden Dreiecke bilden

iii) Starte den Algorithmus `A_einfach` auf den Kanten, die nicht Innenkanten eines trennenden Dreieckes sind

Als ersten Schritt stellt der Algorithmus die Baumstruktur fest, in der die trennenden Dreiecke in der gegebenen planaren Triangulation liegen. Diese Baumstruktur bildet sich zu jedem trennenden Dreieck aus den Informationen, welche trennenden Dreiecke direkt oberhalb und direkt unterhalb davon liegen. Damit ist dann auch bekannt, welche trennenden Dreiecke insgesamt oberhalb und unterhalb eines jeden trennenden Dreieckes liegen. Weiterhin bestimmt der Algorithmus für jedes trennende Dreieck D_I, ob D_I einen K_4 bildet und mit welchen anderen trennenden Dreiecke D_I gemeinsame Außenkanten besitzt. Auf diese Informationen kann der Algorithmus während seiner gesamten Laufzeit zugreifen, ebenso, wie die Routine `BearbeiteTrennendesDreieck`.

Ist die Baumstruktur der trennenden Dreiecke bekannt, so ist klar, welche trennenden Dreiecke jeweils eine Wurzel von den möglicherweise mehreren Bäumen der Baumstruktur bilden. Dies sind genau jene trennenden Dreiecke, oberhalb derer keine trennenden Dreiecke liegen. Auf den trennenden Dreiecken

D_I, die in dieser Weise Wurzeln darstellen, startet `Berechne2Spanner` nun in beliebiger Reihenfolge jeweils das Herzstück des Algorithmus in Form der Routine `BearbeiteTrennendesDreieck`.

Hat die Routine die Wurzeln der trennenden Dreiecke abgearbeitet, sind eventuell noch freie Kanten in G übrig. Dann wird der Algorithmus `A_einfach` gestartet, um die restlichen freien Kanten festzulegen, also zu entfernen oder zu Spanner-Kanten zu machen. Zu diesem Zweck wird `A_einfach` auf den Kanten gestartet, die nicht Innenkanten eines trennenden Dreieckes sind, denn nur unter diesen können sich die restlichen freien Kanten noch befinden.

5.2 Die Routine `BearbeiteTrennendesDreieck`

Die Hauptarbeit des Algorithmus `Berechne2Spanner` wird in der Routine `BearbeiteTrennendesDreieck` ausgeführt. Sie entscheidet für alle Kanten von G, die Innenkanten von trennenden Dreiecken sind, welche zu entfernen und welche als Spanner-Kanten zu behalten sind.

Die Routine `BearbeiteTrennendesDreieck` aufgerufen auf einem trennenden Dreieck D_I mit Innenkanten prüft zuerst, ob direkt unterhalb von D_I noch trennende Dreiecke existieren. Dies ist genau dann der Fall, wenn D_I kein Blatt in der Baumstruktur der trennenden Dreiecke von G ist. Befinden sich direkt unterhalb von D_I noch trennende Dreiecke, so ruft sich die Routine rekursiv auf diesen auf.

Die Reihenfolge, in der sich die Routine auf den direkt unterhalb von D_I liegenden trennenden Dreiecken aufruft, ist nicht völlig freigestellt: Zuerst startet sich die Routine auf allen direkt unterhalb von D_I liegenden trennenden Dreiecken $D_I^{'}$, die, sofern vorhanden, größer als ein K_4 sind. Dann folgen, wenn sie existieren, die trennenden Dreiecke $D_I^{'}$, die einen K_4 bilden, der eine Außenkante hat, die auch Außenkante von D_I ist. Zuletzt kommen, wenn vorhanden, die restlichen K_4 an die Reihe. Sind alle direkt unterhalb von D_I liegenden trennenden Dreiecke abgearbeitet, so sind nach der Arbeitsweise einer Rekursion alle trennenden Dreiecke überhaupt abgearbeitet, die unterhalb von D_I liegen. Dies ist der *Zeitpunkt, an dem die Routine auf D_I zu arbeiten beginnt* bzw. markiert den *Beginn der Abarbeitung* oder *Start* der Routine auf D_I. Das *Ende der Abarbeitung von D_I* bezeichnet den Moment, an dem die Routine auf D_I beendet wird. Die *Abarbeitung* oder *Bearbeitung* von D_I dauert naturgemäß vom Beginn der Abarbeitung bis zum Ende der Abarbeitung von D_I.

Möglicherweise sind durch die vorherige Abarbeitung von anderen trennenden Dreiecken zu diesem Zeitpunkt bereits Innen- und Außenkanten von D_I als Spanner-Kanten markiert. Sind dagegen alle Kanten, sowohl Innen- als auch Außenkanten von D_I zum Zeitpunkt, an dem die Routine auf D_I zu arbeiten beginnt, freie Kanten und bildet D_I einen K_4, so wird D_I ein *freier* K_4 genannt. Beachte, dass für die Eigenschaft, freier K_4 zu sein entscheidend ist, dass ein K_4 zum Zeitpunkt des Abarbeitungsbeginns nur aus freien Kanten besteht. Auch wenn zu einem späteren Zeitpunkt nicht mehr alle Kanten des K_4 freie Kanten sind, kann er dennoch als freier K_4 bezeich-

Routine `BearbeiteTrennendesDreieck`(D_I)

if Es existieren trennende Dreiecke direkt unterhalb von D_I
then Starte `BearbeiteTrennendesDreieck` auf allen trennenden Dreiecken $D_I^{'}$, die direkt unterhalb von D_I liegen und
i) größer als ein K_4 sind
ii) K_4 sind und eine Außenkante mit D_I gemeinsam haben
iii) K_4 sind und keine Außenkante mit D_I gemeinsam haben

/* *Hier sind alle trennenden Dreiecke unterhalb von* D_I *abgearbeitet* */
/* *Die Routine beginnt auf* D_I *zu arbeiten bzw. Abarbeitung von* D_I *beginnt* */

if D_I ist ein freier K_4
then `BearbeiteFreienK`$_4$(D_I)
return

if D_I ist ein änderungsbedürftiger K_4
then `Modifiziere`(D_I)

while Es existieren freie Innenkanten von D_I,
1. **if** Es existiert eine kritische Innenkante k von D_I
then spanne k über die Kanten, mit denen k im Dreieck liegt
continue

2. **if** Es existiert eine günstige Innenkante g von D_I
then if Es existiert eine günstige Innenkante g_A von D_I,
die Zweifachgewinnschritt erlaubt und
mit einer Außenkante von D_I in einem Dreieck liegt
then spanne g_A mittels Zweifachgewinnschritt
continue

if Es existiert eine günstige Innenkante g_B von D_I,
die Zweifachgewinnschritt erlaubt
then spanne g_B mittels Zweifachgewinnschritt
continue

spanne g über die als Spanner-Kante gekennzeichnete Kante
continue

3. **if** Es existiert eine freie Innenkante e von D_I,
die über eine Außenkante von D_I gespannt werden kann
then spanne e über eine Außenkante
continue

4. spanne eine beliebige freie Innenkante e von D_I
über zwei andere Kanten, die mit e ein Dreieck bilden

endwhile

net werden. Ist D_I ein freier K_4, so startet die Routine auf D_I die Funktion `BearbeiteFreienK`$_4$ und lässt D_I so eine Sonderbehandlung zukommen. Nach der Beendigung von `BearbeiteFreienK`$_4(D_I)$ beendet sich auch die Routine `BearbeiteTrennendesDreieck` und die Abarbeitung von D_I endet.

Besitzt ein trennendes Dreieck D_I bei Beginn der Abarbeitung auf D_I die Gestalt wie in Abbildung 5 gezeigt und sind die Innenkanten d und e bei der Abarbeitung des freien K_4 *ade* (Innenkanten von *ade* sind nicht abgebildet) als Spanner-Kanten markiert worden, f dagegen ist freie Kante, so ist D_I ein *änderungsbedürftiger* K_4. Ob die Außenkanten von D_I bei Beginn der Abarbeitung

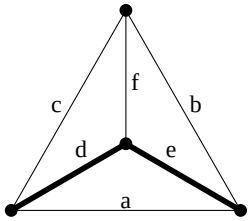


Abbildung 5: Änderungsbedürftiger K_4

von D_I Spanner-Kanten sind oder nicht, spielt dabei keine Rolle. Die Routine `BearbeiteTrennendesDreieck` fängt einen solchen änderungsbedürftigen K_4 ab und modifiziert ihn wie später gezeigt wird. Danach fährt die Routine auf D_I fort.

Nun beginnt die Routine alle freien Innenkanten von D_I abzuarbeiten, das heißt, für jede freie Innenkante zu entscheiden, ob diese zu entfernen oder zur Spanner-Kante zu machen ist.

Als erstes wird geprüft, ob es eine kritische Innenkante in D_I gibt. Gibt es eine solche Kante k, so wird k über die einzigen beiden Kanten, mit denen k in einem Dreieck liegt in Schritt 1 der Routine gespannt.

Gibt es keine kritische Innenkante in D_I, so wird nach einer günstigen Innenkante g gesucht. Findet sich eine solche, so wird Schritt 2 ausgeführt. Unter mehreren günstigen Kanten gibt die Routine der den Vorzug, die einen Zweifachgewinnschritt erlaubt. Eine freie Innenkante g von D_I *erlaubt* einen Zweifachgewinnschritt, wenn sie in einem Dreieck *ghi* liegt, in dem h und i Spanner-Kanten sind. g kann dann in einem Zweifachgewinnschritt gespannt, also entfernt werden, ohne dass eine weitere freie Kante als Spanner-Kante markiert wird. Gibt es mehrere Innenkanten in D_I, die einen Zweifachgewinnschritt erlauben, so wird jener der Vorzug gegeben, die mit einer Außenkante von D_I in einem Dreieck liegt. Die Routine `BearbeiteTrennendesDreieck` gibt also einem Zweifachgewinnschritt, der eine Kante spannt, die mit einer Außenkante von D_I ein Dreieck bildet, den Vorzug vor einem Zweifachgewinnschritt, der keine solche Kante spannt, und diesem wiederum den Vorzug vor einem einfachen Gewinnschritt, der kein Zweifachgewinnschritt ist.

Existiert in D_I weder eine kritische noch eine günstige Innenkante, so kann möglicherweise eine freie Innenkante e von D_I über eine Außenkante von D_I gespannt werden. Ist dies der Fall, so wird Schritt 3 ausgeführt und e über eine Außenkante gespannt. Ansonsten wird in Schritt 4 einfach eine freie Innenkante e über zwei Kanten gespannt, mit denen e in ein Dreieck bildet.

Sowohl Schritt 3 als auch Schritt 4 sind Nichtgewinnschritte, für jede entfernte Kante werden genau zwei andere Kanten als Spanner-Kanten markiert. In Schritt 2 werden pro entfernter Kante echt weniger als zwei andere Kanten zu Spanner-Kanten gemacht. Bei einem einfachen Gewinnschritt, der kein Zweifachgewinnschritt ist, wächst die Zahl der Spanner-Kanten nur um eins. Wird in Schritt 2 ein Zweifachgewinnschritt ausgeführt, so ist wird gar keine Kante als Spanner-Kante gekennzeichnet. Schritt 1 kann ein Nichtgewinnschritt sein oder ein Gewinnschritt, dann nämlich, wenn die zu spannende kritische Kante gleichzeitig eine günstige Kante ist.

Offenbar ist nach dem Ende der Abarbeitung von D_I durch die Routine `BearbeiteTrennendesDreieck` keine Innenkante von D_I mehr freie Kante, entweder sie wurde entfernt oder als zum Spanner zugehörig markiert. Es ist aber wichtig, dass die Routine keine Außenkanten von D_I entfernt hat, sie sind entweder als Spanner-Kanten gekennzeichnet oder freie Kanten geblieben, keine von ihnen wurde bei der Abarbeitung von D_I entfernt.

5.3 Die Funktion BearbeiteFreienK$_4$

Bildet D_I zum Zeitpunkt des Beginns der Abarbeitung einen freien K_4, so wird die Funktion BearbeiteFreienK$_4$ auf D_I aufgerufen, die D_I gesondert behandelt.

Funktion BearbeiteFreienK$_4$(D_I)

i) **if** D_I hat Außenkante mit oberhalb von D_I liegenden trennenden Dreiecken gemeinsam
then Sei b diese Außenkante von D_I
else Sei b beliebige Außenkante von D_I

ii) Sei f eine Innenkante von D_I, die mit b benachbart ist
Spanne f, allerdings nicht über b

iii) Spanne die entstandene kritische Innenkante d

Der Ablauf der Funktion BearbeiteFreienK$_4$ ist in Abbildung 6 dargestellt. D_I als K_4 besteht aus den Innenkanten d, e und f, die Außenkanten heißen a, b, c. Alle Kanten sind nach der Definition eines freien K_4 freie Kanten. Die Funktion stellt zuerst fest, ob D_I eine Außenkante besitzt, die auch Außenkante von oberhalb von D_I liegenden trennenden Dreiecken ist. Nach Lemma 3.4 ist dies

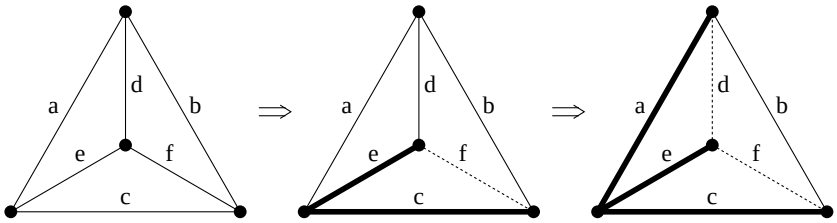


Abbildung 6: Ablauf der Funktion `BearbeiteFreienK`$_4$

höchstens eine Außenkante von D_I. Hat D_I eine solche, wird diese als b bezeichnet, wenn nicht, b als beliebige Außenkante von D_I ausgewählt. Dann wird eine Innenkante, die mit b benachbart ist, im Bild f, gespannt, also entfernt, und die Innenkante, die nicht mit b benachbart ist, und eine Außenkante ungleich b werden zu Spanner-Kanten (ebenso hätte d über e und a gespannt werden können). Dabei entsteht eine kritische Kante, im Bild d, die zugleich günstige Kante ist. Sie wird über das einzige Dreieck, in dem sie noch in D_I liegt in einem Gewinnschritt gespannt. Nach Beendigung der Funktion gibt es keine freien Innenkanten von D_I mehr, die Routine `BearbeiteTrennendesDreieck` würde keine Schritte mehr ausführen. Deshalb kann sich die Routine `BearbeiteTrennendesDreieck` auch auf D_I beenden, sobald `BearbeiteFreienK`$_4$ auf D_I zum Ende gekommen ist.

Der Sinn der Funktion `BearbeiteFreienK`$_4$ liegt darin, zu verhindern, dass die Außenkante, die ein freier K_4 eventuell mit oberhalb von ihm liegenden trennenden Dreiecken gemeinsam hat, bei der Abarbeitung des freien K_4 zur Spanner-Kante gemacht wird. Hat ein freier K_4 keine Außenkante mit oberhalb liegenden trennenden Dreiecken gemeinsam, so ist es unerheblich, welche die beiden bei Abarbeitung markierten Außenkanten sind.

5.4 Die Funktion `Modifiziere`

Funktion `Modifiziere`(D_I)

i) Bezeichne als d und e die beiden markierten Innenkanten von D_I

ii) Bezeichne als x die markierte Innenkante des direkt unterhalb von D_I liegenden trennenden Dreieckes ade, bei dessen Abarbeitung d und e markiert wurden

iii) Bezeichne als y und z die entfernten Innenkanten von ade

vi) Entferne x, mache e zur freien Kante, mache y zur Spanner-Kante und mache a zur Spanner-Kante, falls a dies noch nicht ist

Die Arbeitsweise von `Modifiziere` ist in Abbildung 7 dargestellt. Sichtbar sind die Außenkanten a, b und c und die Innenkanten d, e und f von D_I. Zusätzlich sind die Innenkanten x, y und z des freien K_4 *ade* direkt unterhalb von D_I abgebildet. Wenn `Modifiziere` auf D_I gestartet wird, ist D_I ein änderungs-

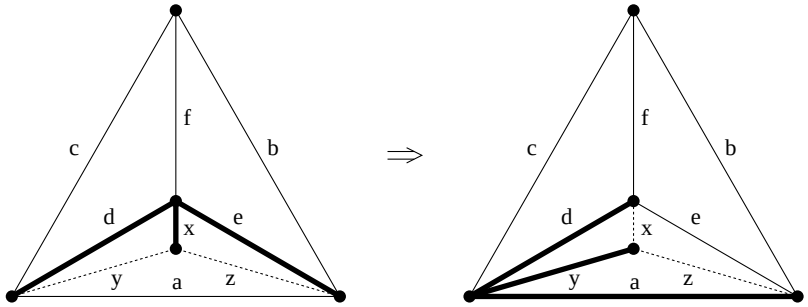


Abbildung 7: Ablauf der Funktion `Modifiziere`

bedürftiger K_4. Nach Definition gibt es dann einen freien K_4, im Bild *ade*, bei dessen Abarbeitung zwei Innenkanten von D_I, im Bild d und e markiert wurden, die dritte Innenkante, in der Abbildung f, ist eine freie Kante. Die Außenkanten von D_I können freie Kanten oder Spanner-Kanten sein. Da *ade* freier K_4 ist, ist bekannt, welche Kanten bei der Abarbeitung von *ade* entfernt und welche markiert wurden. Dazu genügt ein Blick auf die Funktion `BearbeiteFreienK`$_4$, die *ade* als freien K_4 bearbeitet haben muss. Dabei wurden gerade die Außenkanten von *ade*, die nicht Außenkanten eines oberhalb von *ade* liegenden trennenden Dreieckes, also nicht Außenkanten von D_I sind, markiert, das sind d und e. Ebenso klar ist die Gestalt der Innenkanten von *ade*, die Funktion `BearbeiteFreienK`$_4$ hat y und z entfernt und x zur Spanner-Kante gemacht.

Die Funktion `Modifiziere` verändert nun die wohlbekannte Gestalt von *ade*, indem x entfernt, y zur Spanner-Kante und e zur freien Kante gemacht wird. Wenn a noch keine Spanner-Kante ist, so wird auch a als solche markiert. Natürlich wird die Eigenschaft eines 2-Spanners durch die Funktion `Modifiziere` nicht verletzt: x ist jetzt über d und y gespannt, z über y und a. Auf andere als die abgebildeten Kanten braucht dabei keine Rücksicht genommen zu werden, die Aktion kann lokal ablaufen: Da die Kanten d, e und x erst bei der Bearbeitung des freien K_4 *ade* als Spanner-Kanten markiert wurden, gibt es keine anderen Kanten außer y und z, die über diese Kanten gespannt werden, die also darauf angewiesen sind, dass d, e und x Spanner-Kanten sind. Verlieren also die Kanten e und x ihren Status als Spanner-Kanten wieder, so muss nur auf die Kanten y und z Rücksicht genommen werden, dies tut `Modifiziere` wie beschrieben. Die Anzahl der Spanner-Kanten erhöht sich durch `Modifiziere` nicht, wenn a schon Spanner-Kante war, ist diese Anzahl sogar gesunken. Der Gewinnschritt, der auf dem freien K_4 *ade* stattgefunden hat, ist also nicht verloren gegangen.

Die Routine `BearbeiteTrennendesDreieck` fährt nach den Veränderungen von `Modifiziere` auf D_I fort, schließlich sind f und e als freie Innenkanten von D_I noch zu bearbeiten.

5.5 Gültigkeit des berechneten Spanners

Es ist nun zu zeigen, dass der Algorithmus `Berechne2Spanner` einen gültigen 2-Spanner berechnet. Offenbar ist es dazu nötig, nachzuweisen, dass jede Kante, die im Verlauf des Algorithmus entfernt wird, gespannt wird, es also zwei Spanner-Kanten gibt, die die Endpunkte der entfernten Kante verbinden.

Natürlich muss dabei auf der Routine `BearbeiteTrennendesDreieck` das Hauptaugenmerk liegen. Folgendes Lemma sagt aus, dass zum Zeitpunkt des Beginns der Abarbeitung der Routine auf jedem D_I die Gestalt von D_I unversehrt ist.

Lemma 5.1 *Bei Beginn der Abarbeitung eines jeden trennenden Dreieckes D_I durch die Routine `BearbeiteTrennendesDreieck` ist keine Kante aus D_I entfernt.*

Beweis
Zwar hat die Routine zum Zeitpunkt des Bearbeitungsbeginns auf D_I schon andere trennende Dreiecke bearbeitet und dabei Kanten entfernt. Dabei aber kann keine Kante aus D_I entfernt worden sein: Die Routine entfernt auf dem jeweils aktuell bearbeiteten trennenden Dreieck nur Innenkanten, nie aber eine Außenkante. Keine Innenkante von D_I ist nach Definition auch Innenkante eines anderen trennenden Dreieckes. Folglich kann auch keine Innenkante von D_I im Zuge der Abarbeitung von anderen trennenden Dreiecken entfernt worden sein. Die Außenkanten von D_I können nur Innenkanten in oberhalb von D_I liegenden trennenden Dreiecken sein. Diese wurden noch nicht bearbeitet, das stellen die rekursiven Aufrufe der Routine sicher. Genauso wenig wie die Innenkanten können also auch die Außenkanten von D_I noch nicht entfernt worden sein.

□

Unter Verwendung bereits bewiesener Lemmata lässt sich nun zeigen, dass jede freie Kante, die von der Routine entfernt wird, im Sinne eines 2-Spanners gültig gespannt ist.

Lemma 5.2 *Jede freie Kante, die die Routine `BearbeiteTrennendesDreieck` in den Schritten 1-4 entfernt, wird über zwei Spanner-Kanten gespannt.*

Beweis
Die Routine ist so angelegt, dass jede freie Kante, die in den Schritten 1-4 entfernt wird und noch in einem Dreieck liegt, auch über zwei Kanten gespannt wird. Die Frage ist also, ob jede freie Kante, auf die die Routine in den Schritten 1-4 stoßen kann auch in mindestens einem Dreieck liegt.

Nach Lemma 5.1 ist D_I zum Zeitpunkt des Abarbeitungsstarts auf D_I unversehrt. Damit bildet D_I in diesem Moment nach Lemma 3.5 eine planare Triangulation. Kommt die Funktion `BearbeiteFreienK`$_4$ zum Einsatz, beendet sich die Routine danach und keiner der Schritte 1-4 wird ausgeführt. Wird `Modifiziere` durchgeführt, so hinterlässt `Modifiziere` der Routine die Kanten von D_I als planare Triangulation. In jedem Falle bildet vor der erstmaligen Ausführung einer der Schritte 1-4 D_I eine planare Triangulation.

Für die Schritte 1-4 der Routine kann nun versucht werden, Lemma 4.1 anzuwenden. Bis auf eine Ausnahme erfüllen die Schritte 1-4 der Routine die Forderungen an den Algorithmus A aus diesem Lemma: Ist eine Außenkante von D_I kritisch, so wird sie, im Gegensatz zu einer kritischen Innenkante von D_I, nicht bevorzugt gespannt. Vielmehr wird eine Außenkante von D_I während der Abarbeitung von D_I überhaupt nicht gespannt.

Dies allerdings stellt kein Problem dar. Eine Außenkante von D_I wird während der Abarbeitung von D_I nicht gespannt, ist also nicht darauf angewiesen, dass sie in D_I in einem Dreieck liegt. Von allen freien Innenkanten werden jene bevorzugt von der Routine gespannt, die kritisch sind. Aus diesem Grunde wird wie im Beweis von Lemma 4.1 ausgeführt eine kritische Innenkante von D_I nie des letzten Dreieckes beraubt werden, in dem diese noch liegt. So kann keine freie Innenkante von D_I entstehen, die in keinem Dreieck mehr liegt und damit von den Schritten 1-4 der Routine nicht gespannt werden kann.

□

Die Eigenschaft des Algorithmus, einen gültigen 2-Spanner zu berechnen, lässt sich nun einfach folgern.

Korollar 5.3 *Der Algorithmus `Berechne2Spanner` aufgerufen auf einer planaren Triangulation G liefert einen gültigen 2-Spanner für G.*

Beweis
Jede Kante von G, die Innenkante irgend eines trennenden Dreieckes ist, wird durch die Routine `BearbeiteTrennendesDreieck` festgelegt. Geschieht dies für eine Kante e in einem der Schritte 1-4 der Routine, so gibt es nach Lemma 5.2 für e zwei Spanner-Kanten, die mit e in einem Dreieck liegen und e spannen. Wird e in der Funktion `BearbeiteFreienK`$_4$ entfernt, so werden auch dort zwei Spanner-Kanten markiert, die e spannen, siehe dazu die Erläuterungen zur Funktion `BearbeiteFreienK`$_4$. Die Funktion `Modifiziere` ändert zwar den Status von einigen Innenkanten, dies passiert aber auf eine Weise, die sicherstellt, dass jede entfernte Kante im Sinne eines 2-Spanners gespannt ist. Das ist bei der Besprechung der Funktion `Modifiziere` schon erörtert worden.

Alle Kanten, die nach der Bearbeitung sämtlicher trennender Dreiecke noch freie Kanten darstellen, sind entweder Außenkanten von trennenden Dreiecken, die eine Wurzel in der Baumstruktur der trennenden Dreiecke bilden oder Kanten, die in gar keinen trennenden Dreiecken liegen. Der Subgraph von G bestehend aus diesen Kanten sei F genannt. Der Algorithmus `Berechne2Spanner`

ruft auf diesen Kanten von F abschließend den Algorithmus `A_einfach` auf. F besteht aus bereits markierten Kanten und freien Kanten: Die Außenkanten von trennenden Dreiecken, die eine Wurzel in der Baumstruktur bilden, sind entweder markierte oder freie Kanten, keinesfalls aber entfernt. Die Kanten, die in keinem trennenden Dreieck liegen sind in jedem Falle freie Kanten und keine von ihnen wurde bereits entfernt. F kann als ein trennendes Dreieck mit Innenkanten betrachtet werden: Die Kanten des Dreieckes von G, welches das äußere Gebiet von G umrandet sind die Außenkanten von F. Kanten, die Außenkanten von trennenden Dreiecken sind, die eine Wurzel bilden und Kanten, die in gar keinem trennenden Dreieck liegen sind die Innenkanten von F. Die gleichen Strukturaussagen, die für trennende Dreiecke D_I gelten, deren unterhalb liegende trennenden Dreiecke abgearbeitet sind, treffen auch auf F zu: F ist eine planare Triangulation, in der jede Kante in genau zwei Dreiecken liegt. Nach Korollar 4.2 berechnet `A_einfach` auf F einen gültigen 2-Spanner.

□

5.6 Approximationsgüte

Nachdem klar ist, dass der Algorithmus einen gültigen 2-Spanner für G bestimmt, wird der Fokus auf die Größe des berechneten Spanners gerichtet. Diese Größe entspricht offenbar der Anzahl der Kanten, die der Algorithmus im Verlaufe der Berechnung als Spanner-Kanten markiert. Schon im vorigen Kapitel wurde deutlich, dass sich die Größe eines berechneten Spanners aus der Anzahl der Gewinn- und Nichtgewinnschritte ergibt. Offenbar können Gewinnschritte lediglich in der Routine `BearbeiteTrennendesDreieck`, inklusive der Funktion `BearbeiteFreienK`$_4$, stattfinden. Die folgenden Lemmata beschäftigen sich damit, wann ein Gewinnschritt von der Routine und deren Unterfunktionen ausgeführt wird. Das erste Lemma behandelt den Fall, dass das trennende Dreieck kleinstmöglich und damit ein K_4 ist, die darauf folgenden beiden Lemmata decken alle größeren trennenden Dreiecke ab.

Lemma 5.4 *Ist D_I ein K_4 und enthält D_I mindestens eine freie Innenkante beim Beginn der Abarbeitung der Routine `BearbeiteTrennendesDreieck` auf D_I, so wird mindestens ein Gewinnschritt auf D_I ausgeführt.*

Beweis
Fallunterscheidung über die Zahl der freien Kanten von D_I beim Beginn der Abarbeitung der Routine auf D_I:
Fall 1: Alle Kanten von D_I sind freie Kanten
D_I bildet einen freien K_4. Es wird die Funktion `BearbeiteFreienK`$_4$ aufgerufen, die D_I abarbeitet. Wie bei der Besprechung der Funktion erläutert findet dabei genau ein Gewinnschritt statt.

Fall 2: Mindestens eine Kante von D_I ist keine freie Kante
Die Kanten von D_I, die nach Fallunterscheidung keine freien Kanten sind,

können nur schon als Spanner-Kanten gekennzeichnete sein. Da nach Voraussetzung mindestens eine freie Innenkante in D_I existiert, muss in D_I eine freie Innenkante mit einer schon als Spanner-Kante markierten Kante in einem Dreieck liegen, also günstige Kante sein. Schritt 1 braucht nicht ausgeführt zu werden, da nach Lemma 5.1 keine kritische Kante in D_I existiert. Die Routine führt also zu Beginn Schritt 2 und damit einen Gewinnschritt durch.

□

Lemma 5.5 *Ist D_I ein trennendes Dreieck und kein K_4 und enthält D_I beim Beginn der Abarbeitung der Routine* `BearbeiteTrennendesDreieck` *auf D_I nur freie Kanten, so sind mindestens zwei Schritte der Routine auf D_I Gewinnschritte.*

Beweis
Die Aussage wird mit Hilfe der Abbildung 8 bewiesen. Nach Lemma 5.1 liegt

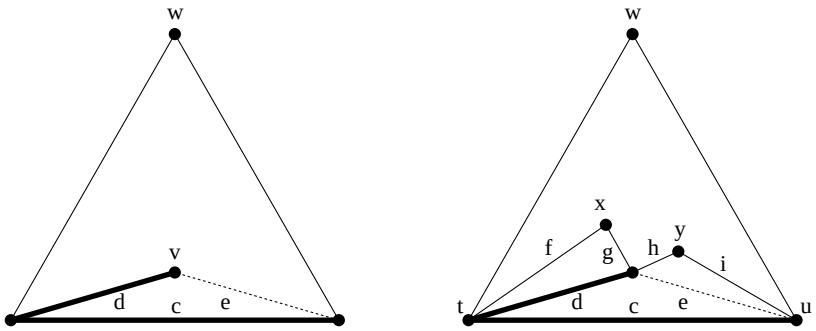


Abbildung 8: Situation nach erstem Schritt der Routine auf D_I

zu Beginn der Abarbeitung jede Kante von D_I in genau zwei Dreiecken. Spiegelt Abbildung 8 diesen Fakt für einige Kanten nicht wider, so geschieht dies nur aus beweistechnischen Gründen, denn die Abbildung dient lediglich dazu, die bisherigen Argumente zum Beweis der Aussage zusammenzufassen und zu verdeutlichen.

Da die Routine zu Beginn weder eine günstige noch eine kritische Kante findet, wird Schritt 3 ausgeführt. Es entsteht ein Situation wie in Abbildung 8 auf der linken Seite dargestellt: Die Kante e wird über d und die Außenkante c gespannt. Die Knoten v und w können nicht adjazent sein, denn D_I besitzt mindestens einen fünften Knoten. Dieser müsste in einem der drei Teildreiecke des so entstandenen K_4 liegen und ein trennendes Dreieck innerhalb von D_I entstände. Dieses läge in der Baumstruktur der trennenden Dreiecke unterhalb von D_I und wäre von der Routine schon behandelt. Die Innenkanten und damit der fünfte Knoten von D_I wären jetzt ausgeblendet und die Routine fände lediglich einen K_4 vor, was einen Widerspruch zur Voraussetzung darstellt.

Da die Kante d aber in zwei Dreiecken liegt, muss der Knoten x existieren und damit die Kanten f und g. Auch die Kante e liegt in zwei Dreiecken. Da

die Routine bei Behandlung von D_I wie besprochen kein trennendes Dreieck innerhalb von D_I vorfinden kann, ist x auch nicht mit u verbunden. Damit muss der Knoten y existieren und mit ihm die Kanten h und i. Soweit ist die Situation in Abbildung 8 auf der rechten Seite dargestellt.

Offenbar gibt es nach dem ersten Schritt genau zwei kritische Kanten h und i, die von der Routine nun bevorzugt behandelt werden müssen. Es findet eine Fallunterscheidung danach statt, ob x und y adjazent sind.

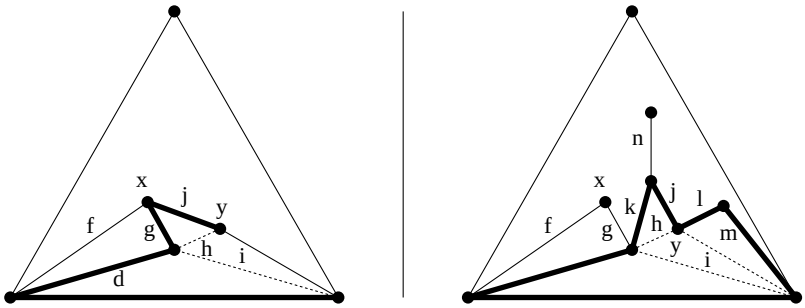


Abbildung 9: Fallunterscheidung nach Adjazenz von x und y

Fall 1: x und y sind adjazent, siehe Abbildung 9 linke Seite
Die Kanten j und g werden durch das Spannen der kritischen Kante h zu Spanner-Kanten, unabhängig davon, ob i als kritische Kante davor oder danach und egal über welche Kanten gespannt wird. Nachdem beide kritischen Kanten h und i gespannt wurden, gibt es keine kritische Kante in D_I mehr. f ist jetzt eine günstige Kante, die mit zwei Spanner-Kanten in einem Dreieck liegt. Es ist an dieser Stelle ein Zweifachgewinnschritt mit dem Spannen von f über d und g möglich. Ein solcher findet nun statt, wenn nicht durch das Spannen von f über d und g, so an anderer Stelle in D_I.

Fall 2: x und y sind nicht adjazent, siehe Abbildung 9 rechte Seite
Beim Entfernen der kritischen Kanten h und i werden die Kanten k und j zu Spanner-Kanten, ebenso die Kanten l und m, egal in welcher Reihenfolge h und i gespannt werden. Möglicherweise ist m eine Außenkante von D_I. Es kann nicht der Fall sein, dass j und l identisch sind, es würde andernfalls ein trennendes Dreieck innerhalb von D_I entstehen. Als günstige Kanten werden f oder g mit einem ersten Gewinnschritt gespannt. Eine weitere Kante n, die mit j in einem Dreieck liegt (und darum nicht mit x verbunden sein kann), bleibt von diesem ersten Gewinnschritt und vom Forträumen der entstehenden kritischen Kanten unbeeinflusst, da sie weder mit f noch mit g in einem Dreieck liegt. Diese Kante n liegt mit j, einer Spanner-Kante in einem Dreieck und kann dann mit Schritt 2 und dem damit zweiten Gewinnschritt gespannt werden. Die gleiche Argumentation funktioniert auch andersherum: Wird n im ersten Gewinnschritt gespannt, kann dies und das Abräumen einer eventuell entstehenden kritischen

Kante den zweiten Gewinnschritt bei f oder g nicht verhindern.

□

Lemma 5.6 *Ist D_I ein trennendes Dreieck ungleich K_4, das beim Beginn der Abarbeitung der Routine* `BearbeiteTrennendesDreieck` *auf D_I mindestens eine Spanner-Kante und mindestens eine freie Innenkante enthält. Dann führt die Routine auf D_I mindestens einen Gewinnschritt aus.*

Beweis
Nach Lemma 5.1 findet die Routine anfangs keine kritische Kante in D_I vor. Da es sowohl mindestens eine innere freie Kante als auch mindestens eine Kante gibt, die schon als Spanner-Kante ausgewählt wurde, muss es eine freie innere Kante geben, die mit einer schon als Spanner-Kante gewählten in einem Dreieck liegt. Damit ist diese freie innere Kante eine günstige Kante und wird von der Routine am Anfang mit Schritt 2, einem Gewinnschritt, gespannt.

□

Die bisherigen Betrachtungen decken fast alle Fälle ab, denen sich die Routine beim Beginn der Abarbeitung auf einem trennenden Dreieck D_I gegenüber sehen kann: D_I kann ein K_4, aber auch größer sein, D_I kann schon Spanner-Kanten enthalten oder nicht, solange mindestens eine innere Kante von D_I noch frei ist. Die einzige Möglichkeit, die bisher noch nicht betrachtet wurde, ist jene, bei der die Routine beim Abarbeitungsbeginn auf D_I alle inneren Kanten schon als Spanner-Kanten markiert vorfindet. Dann findet natürlich überhaupt kein Schritt statt, denn D_I beinhaltet in diesem Falle keine freie Innenkante. Es gibt also trennende Dreiecke, die keinen Gewinnschritt ermöglichen. Wie dennoch aus der Anzahl der trennenden Dreiecke eine Mindestanzahl an durchgeführten Gewinnschritten gefolgert werden kann, zeigt das folgende zentrale Lemma.

Lemma 5.7 *Sei G eine planare Triangulation. Die Anzahl der Gewinnschritte, die die Routine* `BearbeiteTrennendesDreieck` *auf den trennenden Dreiecken von G ausführt, ist mindestens so groß wie die Anzahl der trennenden Dreiecke in G.*

Beweis
Jedem trennenden Dreieck wird ein Gewinnschritt zugeteilt.
Offenbar ist diese Zuteilung kein Problem, wenn die Routine beim Abarbeitungsstart auf D_I ein trennendes Dreieck antrifft, das mindestens eine freie innere Kante enthält. Nach den vorigen Lemmata führt die Routine mindestens einen Gewinnschritt auf D_I aus, D_I kann dieser Gewinnschritt zugeteilt werden.

Ein Problem ergibt sich erst dann, wenn D_I beim Beginn der Abarbeitung der Routine bereits keine freien Innenkanten mehr besitzt. Das bedeutet, dass alle Innenkanten von D_I schon als Spanner-Kanten markiert sind und die Routine auf D_I überhaupt keine Schritte und damit auch keinen Gewinnschritt ausführt. (Eine derartige Situation kann durchaus auftreten, der Graph, an dem

abschließend die Schärfe der Approximationsgüte gezeigt wird, lässt eine solche Konstellation entstehen.)

Sind beim Abarbeitungsbeginn der Routine auf D_I bereits alle Innenkanten von D_I als Spanner-Kanten markiert, so können diese Spanner-Kanten nur bei der Bearbeitung von unterhalb von D_I liegenden trennenden Dreiecken entstanden sein. Bei deren Bearbeitung wurden Außenkanten zu Spanner-Kanten gemacht, die nun als Innenkanten von D_I auftreten. Ziel des Beweises ist es, zu zeigen, dass in den trennenden Dreiecken, die unterhalb von D_I liegen und bei deren Bearbeitung durch die Routine sämtliche Innenkanten von D_I markiert wurden, mindestens ein Gewinnschritt stattgefunden hat, der noch keinem anderen trennenden Dreieck zugeteilt wurde. Dieser Gewinnschritt kann dann D_I zugeteilt werden.

Sei $D_I^{'}$ ein unterhalb von D_I liegendes trennendes Dreieck, bei dessen Bearbeitung eine Außenkante als Spanner-Kante markiert wurde, die jetzt als Innenkante von D_I beim Beginn der Abarbeitung der Routine auf D_I bereits markiert ist.

Es reicht aus, zu zeigen, dass die Anzahl der Gewinnschritte bei der Bearbeitung von $D_I^{'}$ so groß ist, dass allen oberhalb von $D_I^{'}$ liegenden trennenden Dreiecken, die eine Innenkante enthalten, die bei der Bearbeitung von $D_I^{'}$ markiert wurde, ein Gewinnschritt zur Verrechnung angeboten werden kann.

Genau dies wird für alle Gestalten von $D_I^{'}$ bis auf eine Ausnahme per vollständiger Induktion gezeigt und im Folgenden **Induktionssausage** genannt. Die Ausnahme wird gesondert behandelt.

Erfüllt $D_I^{'}$ die Induktionsaussage, so haben in $D_I^{'}$ mehr Gewinnschritte stattgefunden, als der eine, der $D_I^{'}$ zum Beweis der Aussage des Lemmas zugeteilt werden muss. Die in dieser Weise überschüssigen Gewinnschritte, die in $D_I^{'}$ stattgefunden haben, können den trennenden Dreiecken, die oberhalb von $D_I^{'}$ liegen und eine Innenkante enthalten, die während der Abarbeitung von $D_I^{'}$ als Außenkante von $D_I^{'}$ markiert wurde, zugeteilt werden, wenn auf ihnen nicht selbst ein Gewinnschritt stattfindet. Dies ist gemeint, wenn die Rede davon ist, dass Gewinnschritte von $D_I^{'}$ an oberhalb von $D_I^{'}$ liegende trennende Dreiecke zur Verrechnung angeboten werden.

Betrachte als Beispiel die Abarbeitung von $D_I^{'}$, während der drei Gewinnschritte durchgeführt und dabei zwei Außenkanten a und b von $D_I^{'}$ zu Spanner-Kanten gemacht werden. Jede Außenkante von $D_I^{'}$ kann in höchstens einem oberhalb von $D_I^{'}$ liegenden trennenden Dreieck als Innenkante auftauchen. Die beiden Außenkanten a und b treten also höchstens in zwei trennenden Dreiecken A und B als Innenkanten auf. Von den drei Gewinnschritten während der Abarbeitung von $D_I^{'}$ können A und B jeweils einer zur Zuteilung angeboten werden, der dritte Gewinnschritt wird $D_I^{'}$ zugeteilt.

Der Induktionsanfang zeigt, dass die Induktionsaussage für ein trennendes Dreieck $D_I^{'}$ gilt, das ein Blatt der Baumstruktur der trennenden Dreiecke bildet.

Unterhalb von $D_I^{'}$ befindet sich also kein trennendes Dreieck mehr, was bedeutet, dass alle Innenkanten von $D_I^{'}$ beim Beginn der Abarbeitung der Routine freie Kanten sind.

Induktionsanfang: $D_I^{'}$ ist ein Blatt in der Baumstruktur der trennenden Dreiecke

Induktionsanfang Fall 1: $D_I^{'}$ ist ein K_4
Betrachte folgende Tabelle. Die Spalten geben die Anzahl der als Spanner-Kanten markierten Außenkanten von $D_I^{'}$ zu dem Zeitpunkt an, an dem die Routine auf $D_I^{'}$ zu arbeiten beginnt. Die Zeilen enthalten die Anzahl der markierten Außenkanten am Ende der Abarbeitung von $D_I^{'}$ durch die Routine. Die Zellen der Tabelle enthalten die Anzahl der Gewinnschritte, die die Routine auf $D_I^{'}$ unter der jeweiligen Kombination ausführt. Alle mit „-“ gefüllten Zellen stehen für Kombinationen, die aufgrund der Arbeitsweise der Routine nicht auftreten können.

		mark. Außenkanten vorher			
		0	1	2	3
mark.	0	-	-	-	-
Außen-	1	-	-	-	-
kanten	2	1	2	3	-
nachher	3	-	-	2	3

Der Tabelleneintrag „1“ in der ersten Spalte entsteht durch die Abarbeitung der Funktion `BearbeiteFreienK`$_4$ genau wie in Lemma 5.4 Fall 1 beschrieben. Der Tabelleneintrag „2“in der vorletzten Zeile entsteht auch ähnlich diesem Fall, lediglich die Außenkante, über die im ersten Schritt gespannt wird, ist bereits eine Spanner-Kante. Damit ist der erste Schritt schon ein Gewinnschritt und so ergeben sich insgesamt zwei Gewinnschritte in dieser Gestalt von $D_I^{'}$. Sind beim Start der Routine auf $D_I^{'}$ bereits genau zwei Außenkanten von $D_I^{'}$ Spanner-Kanten (vorletzte Spalte der Tabelle), so hat die Routine zwei Möglichkeiten: Entweder wird im ersten Schritt, in jedem Falle ein Gewinnschritt, die Innenkante als Spanner-Kante markiert, die mit beiden bereits markierten Außenkanten jeweils in einem Dreieck liegt. Dann ist der zweite Schritt ein Zweifachgewinnschritt und es haben insgesamt drei Gewinnschritte stattgefunden, ohne dass eine weitere Außenkante zu einer Spanner-Kante wird. Andernfalls wird die dritte Außenkante von $D_I^{'}$ zur Spanner-Kante und es finden zwei Gewinnschritte statt. Die drei Gewinnschritte zum Tabelleneintrag „3 “in der letzten Spalte entstehen unabhängig von der Reihenfolge, in der die Routine die Innenkanten bearbeitet.

Mit Ausnahme der Kombination in der ersten Spalte gilt die Induktionsaussage: Die Anzahl der Gewinnschritte, die die Routine auf $D_I^{'}$ durchführt ist um

mindestens eins größer als die Anzahl der von der Routine während der Abarbeitung von $D_I^{'}$ markierten Außenkanten von $D_I^{'}$ (Differenz von Außenkanten nachher und Außenkanten vorher). Damit kann $D_I^{'}$ einer dieser Gewinnschritte zugeteilt werden und jedem oberhalb von $D_I^{'}$ liegenden trennenden Dreieck, in dem eine während der Bearbeitung von $D_I^{'}$ markierte Außenkante von $D_I^{'}$ als Innenkante auftritt, auch ein Gewinnschritt zur Verrechnung angeboten werden.

Der Fall in der ersten Spalte bildet die vor dem Induktionsanfang angesprochene Ausnahme. Die Routine erzeugt hier nur einen Gewinnschritt, der $D_I^{'}$ zugeteilt werden muss, und macht zwei Außenkanten von $D_I^{'}$ zu Spanner-Kanten ohne oberhalb von $D_I^{'}$ liegenden trennenden Dreiecken einen weiteren Gewinnschritt anbieten zu können.

Ein trennendes Dreieck, das unter diese Ausnahme fällt, bildet einen freien K_4. Von der Routine wird es gesondert behandelt, durch die Abarbeitung von `BearbeiteFreienK`$_4$. Diese Sonderbehandlung eines freien K_4 hat zur Folge, dass die beiden bei seiner Abarbeitung markierten Außenkanten nur noch als Innenkanten des direkt oberhalb des freien K_4 liegenden trennenden Dreieckes D_I auftauchen. In allen anderen oberhalb des freien K_4 liegenden trennenden Dreiecken $\widehat{D_I}$, die dann auch oberhalb von D_I liegen, sind diese Kanten nicht mehr sichtbar, da sie als Innenkanten von D_I keine Kanten von $\widehat{D_I}$ sind.

Zusammenfassend lässt sich zum Fall 1 des Induktionsanfanges sagen, dass bei der Bearbeitung eines K_4, der kein freier K_4 ist, für jede während der Abarbeitung markierte Außenkante des K_4 ein Gewinnschritt zur Verrechnung an oberhalb liegende trennende Dreiecke angeboten wird. Ein freier K_4 bietet keinen Gewinnschritt zur Verrechnung an, unterliegt aber einer Sonderbehandlung durch die Routine, so dass die beiden bei der Abarbeitung des freien K_4 markierten Außenkanten Innenkanten des direkt oberhalb des freien K_4 liegenden trennenden Dreieckes sind.

Induktionsanfang Fall 2: $D_I^{'}$ ist größer als ein K_4
Auch in diesem Falle betrachte eine Tabelle, die alle möglichen Kombinationen von markierten Außenkantenanzahlen vor und nach der Bearbeitung von $D_I^{'}$ aufführt. In der ersten Zeile einer Tabellenzelle ist die Anzahl der Kanten modulo 3 angegeben, die von der Routine auf $D_I^{'}$ festgelegt werden: Alle Innenkanten von $D_I^{'}$ ($\equiv 0 \ mod \ 3$) plus Außenkanten nachher minus Außenkanten vorher. Ein Nichtgewinnschritt der Routine legt $\equiv 0 \ mod \ 3$ Kanten fest, ein Gewinnschritt legt $\equiv 2 \ mod \ 3$ Kanten fest. Ein Zweifachgewinnschritt legt genauso wie zwei einfache Gewinnschritte $\equiv 1 \ mod \ 3$ Kanten fest. Aus der Anzahl $mod \ 3$ der festzulegenden Kanten in der ersten Zeile einer Tabellenzelle folgt die Mindestanzahl der Gewinnschritte, die die Routine auf $D_I^{'}$ durchführt in der zweiten Zeile einer Zelle. Beachte, dass in $D_I^{'}$ nach Lemma 5.5 mindestens zwei Gewinnschritte stattfinden. Dies gilt auch, wenn beim Start der Routine auf $D_I^{'}$ schon Außenkanten von $D_I^{'}$ Spanner-Kanten sind, es findet lediglich im Beweis von Lemma 5.5 schon im ersten Schritt ein Gewinnschritt statt.

		markierte Außenkanten vorher			
		0	1	2	3
mark.	0	-	-	-	-
Außen-	1	$\equiv 1$ **2**	$\equiv 0$ **3**	-	-
Kanten	2	$\equiv 2$ **4**	$\equiv 1$ **2**	$\equiv 0$ **3**	-
nachher	3	$\equiv 0$ **3**	$\equiv 2$ **4**	$\equiv 1$ **2**	$\equiv 0$ **3**

Alle möglichen Kombinationen mit einer Ausnahme erfüllen auch hier folgenden Zusammenhang: Die Anzahl der Gewinnschritte ist um mindestens eins größer als die Anzahl der während der Abarbeitung markierten Außenkanten. Lediglich der linke untere Tabelleneintrag folgt dem nicht. Es werden drei Außenkanten von $D_I^{'}$ bei dessen Abarbeitung markiert aber möglicherweise finden nur drei Gewinnschritte statt. Dies stellt aber kein Problem dar: Die drei Außenkanten eines trennenden Dreieckes $D_I^{'}$ können nach Lemma 3.4 nur in höchstens zwei oberhalb von $D_I^{'}$ liegenden trennenden Dreiecken als Innenkanten auftauchen. Diesen höchstens zwei trennenden Dreiecken kann jeweils ein Gewinnschritt zur Verrechnung angeboten werden, ein Gewinnschritt bleibt für $D_I^{'}$ selbst.

Damit gilt für Fall 2 des Induktionsanfanges, dass jedes oberhalb von $D_I^{'}$ liegende trennende Dreieck, das eine markierte Innenkante besitzt, die als Außenkante von $D_I^{'}$ markiert wurde, einen Gewinnschritt von $D_I^{'}$ zur Verrechnung angeboten bekommt.

An dieser Stelle ist der Induktionsanfang abgeschlossen und für jedes trennende Dreieck $D_I^{'}$, das ein Blatt in der Baumstruktur der trennenden Dreiecke und kein freier K_4 ist, gilt die Induktionsaussage.

Es ist deutlich geworden, dass eine Außenkante, die bei der Abarbeitung eines freien K_4 markiert wurde, keinen Gewinnschritt an oberhalb liegende trennende Dreiecke anbieten kann. Um solche Kanten bezeichnen zu können, sei an dieser Stelle eine Benennung eingeführt: Eine Kante *stammt* von einem freien K_4, wenn sie als Außenkante eines freien K_4 bei dessen Abarbeitung markiert wurde.

Der Induktionsschritt beschäftigt sich nun mit einem trennenden Dreieck $D_I^{'}$, das kein Blatt in der Baumstruktur der trennenden Dreiecke ist und damit selbst trennende Dreiecke enthält. Es wird angenommen, dass jedes unterhalb von $D_I^{'}$ liegende trennende Dreieck, sofern es kein freier K_4 ist, die Induktionsaussage erfüllt.

Induktionsschritt: $D_I^{'}$ ist kein Blatt in der Baumstruktur der trennenden Dreiecke
Falls $D_I^{'}$ beim Start der Routine auf $D_I^{'}$ nur freie Innenkanten enthält, so hat $D_I^{'}$ die gleiche Gestalt wie die beim Induktionsanfang behandelten trennenden Dreiecke. Die dort gezeigten Zusammenhänge gelten in diesem Falle auch hier. Deshalb gilt dann die Induktionsaussage auch für $D_I^{'}$ mit der Ausnahme, dass $D_I^{'}$ ein freier K_4 ist. Falls $D_I^{'}$ ein freier K_4 ist, kann die Induktionsaussage nicht gezeigt werden und $D_I^{'}$ unterliegt der Sonderbehandlung wie oben beschrieben durch die Funktion `BearbeiteFreienK`$_4$.

Im Folgenden sind im Induktionsschritt deshalb nur noch trennende Dreiecke zu behandeln, die beim Start der Routine bereits als Spanner-Kanten markierte Innenkanten enthalten.

Induktionsschritt Fall 1: $D_I^{'}$ ist ein K_4
Falls $D_I^{'}$ beim Start der Routine genau eine markierte Innenkante besitzt, so betrachte Abbildung 10, $D_I^{'}$ ist das Dreieck *abc*, die Kante e ist die einzige bereits beim Start der Routine auf $D_I^{'}$ markierte Innenkante. Die Kante e kann

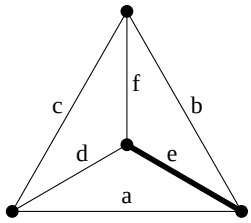


Abbildung 10: $D_I^{'}$ besitzt beim Start der Abarbeitung genau eine bereits markierte Innenkante

nicht von einem freien K_4 stammen: Da eine bei der Abarbeitung eines freien K_4 markierte Außenkante wegen der Sonderbehandlung wie besprochen nur im direkt darüber liegenden trennenden Dreieck als Innenkante sichtbar ist, müsste bef oder ade ein freier K_4 sein. Dann aber müsste die zweite Außenkante, die bei der Abarbeitung eines freien K_4 markiert wird, nach der Sonderbehandlung auch Innenkante des trennenden Dreieckes *abc* sein. Dies ist nicht der Fall, da *abc* nur genau eine markierte Innenkanten enthält. Damit besitzt *abc* eine markierte Innenkante, die nicht von einem freien K_4 stammt und erhält damit nach Induktionsaussage einen Gewinnschritt zur Verrechnung angeboten. Dieser Gewinnschritt wird $D_I^{'}$ zugeteilt. Bei der Abarbeitung der Routine auf $D_I^{'}$ wird d über a und e gespannt und f über e und b. Jeder dieser Schritte ist ein Gewinnschritt und die Außenkanten a und b können, wenn sie dabei als Spanner-Kanten markiert werden, es also nicht bereits vorher waren, jeweils einen Gewinnschritt an die trennenden Dreiecke anbieten, in denen sie als Innenkanten auftreten.

In dem Falle, dass $D_I^{'}$ beim Start der Routine genau zwei markierte Innenkanten enthält, betrachte Abbildung 11, $D_I^{'}$ ist das Dreieck *abc*, die Kanten d und e

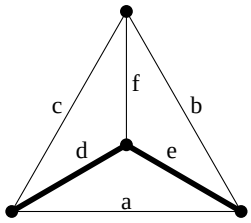


Abbildung 11: $D_I^{'}$ besitzt beim Start der Abarbeitung genau zwei bereits markierte Innenkanten

sind bereits als Spanner-Kanten markiert. Wurde eine der Kanten d und e bei der Abarbeitung eines trennenden Dreieckes markiert, das kein freier K_4 ist, so bekommt $D_I^{'}$ von diesem Dreieck nach Induktionsaussage einen Gewinnschritt zugeteilt. Von der Routine wird f über c und d oder über b und e gespannt, in jedem Falle durch einen Gewinnschritt. Wird dabei eine der Außenkanten c oder b als Spanner-Kante markiert, so kann dem trennenden Dreieck, in dem diese Außenkante als Innenkante auftritt, ein Gewinnschritt zur Verrechnung angeboten werden.

Im anderen Falle, dass $D_I^{'}$ beim Start der Routine genau zwei markierte Innenkanten besitzt, die beide bei der Abarbeitung eines freien K_4 entstanden sind, ist $D_I^{'}$ gerade ein änderungsbedürftiger K_4. Da durch die Sonderbehandlung eines freien K_4 seine beiden markierten Außenkanten nur im direkt darüber liegenden trennenden Dreieck als Innenkanten sichtbar sind, muss, falls d und e bei der Abarbeitung eines freien K_4 markierte Außenkanten sind, *ade* dieser freie K_4 sein. Die Routine startet `Modifiziere` auf $D_I^{'}$, da $D_I^{'}$ ein änderungsbedürftiger K_4 ist. Dadurch wird der Status einiger Innen- und Außenkanten des freien K_4 *ade* verändert wie bei der Besprechung der Funktion `Modifiziere` dargestellt wurde. Wichtig in diesem Zusammenhang ist, dass der Gewinnschritt, der bei der Bearbeitung des freien K_4 *ade* stattgefunden hat, und *ade* zugeteilt ist, erhalten bleibt. Das Verhältnis von Spanner-Kanten zu entfernten Kanten in *ade* verschiebt sich nicht in Richtung der Spanner-Kanten. Die Zuteilung von Gewinnschritten, die Ziel des ganzen Beweises ist, wird nicht beeinträchtigt. Nach Beendigung der Funktion `Modifiziere` hat $D_I^{'}$ die Gestalt wie in Abbildung 12 gezeigt: a ist eine Spanner-Kante, e ist eine freie Kante, der Rest der Kanten von $D_I^{'}$ wurde durch die Funktion nicht beeinflusst.

Die Routine fährt nach `Modifiziere`($D_I^{'}$) mit der Abarbeitung von $D_I^{'}$ fort. Da die Routine bei einem Schritt vom Typ 2 einem Zweifachgewinnschritt den Vorzug vor einem einfachen Gewinnschritt gibt, finden bei der Abarbeitung von *abc* noch mindestens drei Gewinnschritte statt: e wird über a und d in einem

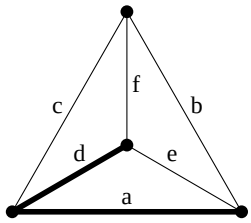


Abbildung 12: $D_I^{'}$ nach der Funktion `Modifiziere`

Zweifachgewinnschritt gespannt, f über c und d in einem einfachen Gewinnschritt, falls c schon Spanner-Kante ist, sogar in einem Zweifachgewinnschritt. Von diesen drei Gewinnschritten kann ein Gewinnschritt abc zugeteilt werden. Ein zweiter Gewinnschritt kann dem trennenden Dreieck angeboten werden, das a als Innenkante enthält, ein dritter Gewinnschritt jenem Dreieck, das c als Innenkante enthält. Die Kante b wird bei der Abarbeitung von abc in diesem Falle nicht als Spanner-Kante markiert. Damit erfüllt abc die Induktionsaussage.

Im Falle, dass $D_I^{'}$ genau drei markierte Innenkanten enthält, findet auch kein Schritt der Routine auf $D_I^{'}$ statt, es kann bei der Bearbeitung von $D_I^{'}$ auch kein Außenkante zur Spanner-Kante werden. Im Sinne der Induktionsaussage ist nichts zu zeigen.

Der Fall 1 des Induktionsschrittes ist an dieser Stelle beendet, $D_I^{'}$ als K_4 und als trennendes Dreieck, das in der Baumstruktur kein Blatt ist, erfüllt die Induktionsaussage, wenn $D_I^{'}$ kein freier K_4 ist.

Induktionsschritt Fall 2: $D_I^{'}$ ist größer als ein K_4
Jede als Spanner-Kante markierte Innenkante, die $D_I^{'}$ bereits beim Abarbeitungsbeginn der Routine auf $D_I^{'}$ enthält, wurde entweder bei der Abarbeitung eines freien K_4 oder bei der Abarbeitung eines trennenden Dreieckes von anderer Gestalt als Spanner-Kante markiert.

Induktionsschritt Fall 2A: $D_I^{'}$ enthält mindestens eine markierte Innenkante, die nicht von einem freien K_4 stammt
Die markierte Innenkante, die nicht von einem freien K_4 stammt, bietet $D_I^{'}$ nach Induktionsaussage einen Gewinnschritt zur Verrechnung an. Dieser Gewinnschritt wird $D_I^{'}$ zugeteilt. Werden bei der Abarbeitung der Routine auf $D_I^{'}$ keine Außenkanten von $D_I^{'}$ markiert, so ist für den Beweis, dass $D_I^{'}$ die Induktionsaussage erfüllt, nichts zu zeigen. Deshalb ist auch nichts zu zeigen, wenn $D_I^{'}$ beim Start der Routine keine freien Innenkanten mehr enthält, dann führt die Routine gar keinen Schritt auf $D_I^{'}$ aus und damit wird auch keine Außenkante markiert.

Enthalte $D_I^{'}$ also sowohl eine freie Innenkante als auch eine bereits markiert Innenkante. Dann gibt es in $D_I^{'}$ auch eine freie Innenkante, die mit einer bereits markierten Innenkante in einem Dreieck liegt, also eine günstige Kante ist. Da $D_I^{'}$ beim Beginn der Routine keine kritischen Kanten nach Lemma 5.1 enthält, ist der erste Schritt der Routine auf $D_I^{'}$ ein Gewinnschritt.

Wird bei der Abarbeitung von $D_I^{'}$ insgesamt nur genau eine Außenkante von $D_I^{'}$ markiert, so erfüllt $D_I^{'}$ offensichtlich die Induktionsaussage: Der Gewinnschritt, der der erste Schritt der Routine auf $D_I^{'}$ ist, kann dem trennenden Dreieck, in dem diese Außenkante als Innenkante auftritt, zur Verrechnung angeboten werden.

Auch wenn mindestens zwei Gewinnschritte auf $D_I^{'}$ ausgeführt werden, erfüllt $D_I^{'}$ die Induktionsaussage: Egal wie viele Außenkanten von $D_I^{'}$ bei der Abarbeitung markiert werden, die Außenkanten treten nach Lemma 3.4 in höchstens zwei verschiedenen oberhalb von $D_I^{'}$ liegenden trennenden Dreiecken als Innenkanten auf. Damit kann jedem dieser trennenden Dreiecke ein Gewinnschritt zur Verrechnung angeboten werden.

Die einzig verbleibende Möglichkeit, dass während der Abarbeitung von $D_I^{'}$ nur genau ein Gewinnschritt stattfindet, dafür aber zwei oder drei Außenkanten von $D_I^{'}$ markiert werden, kann nicht auftreten, wie jetzt gezeigt wird. Es wird angenommen, das Gegenteil gilt, es findet nur ein Gewinnschritt auf $D_I^{'}$ statt, insgesamt werden aber zwei oder drei Außenkanten von $D_I^{'}$ als Spanner-Kanten markiert.

Zuerst wird zusätzlich angenommen, dass der erste Schritt der Routine auf $D_I^{'}$, der wie besprochen ein Gewinnschritt sein muss, keine Außenkante von $D_I^{'}$ markiert. Betrachte dazu Abbildung 13, die einen Teil von $D_I^{'}$ in Form einiger Innen- und Außenkanten von $D_I^{'}$ darstellt. Im ersten Schritt wird a als günstige Kante

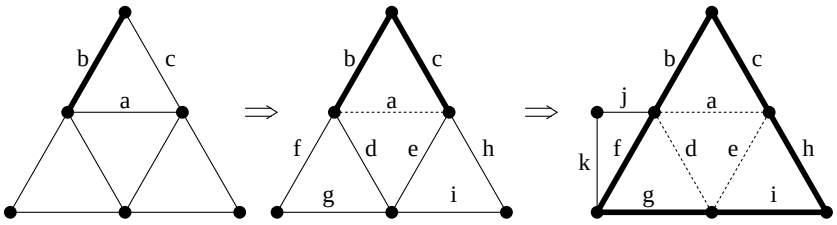


Abbildung 13: Als günstige Kante wird a zuerst in einem Gewinnschritt über b und c gespannt. Die dabei kritisch werdenden Kanten d und e werden danach gespannt.

über b und c gespannt. Sind d und e freie Kanten, so sind es nun kritische Kanten, die die Routine in den nächsten zwei Schritten vom Typ 1 entfernt: d wird über f und g gespannt, e über h und i.

In diesen beiden Schritten vom Typ 1 müssen zwei Außenkanten von $D_I^{'}$ als Spanner-Kanten markiert worden sein: Da nach Annahme nur ein Gewinn-

schritt auf $D_I^{'}$ stattfindet, kann überhaupt kein weiterer Schritt auf $D_I^{'}$ stattfinden. Ein weiterer Schritt kann nicht vom Typ 1 sein, es gibt keine weiteren kritischen Kanten in $D_I^{'}$. Das Stattfinden eines weiteren Schrittes heißt, dass es mindestens eine freie Innenkante in $D_I^{'}$ gibt, da es aber auch als Spanner-Kanten markierte Kanten in $D_I^{'}$ gibt, gäbe es dann auch eine freie Innenkante, die mit einer markierten in einem Dreieck läge, also günstige Innenkante wäre. Ein weiterer Schritt auf $D_I^{'}$ wäre damit ein Gewinnschritt, der nach Annahme nicht stattfindet.

Da weder h und i gleichzeitig noch f und g gleichzeitig Außenkanten von $D_I^{'}$ sein können, ist unter der Annahme, dass mindestens zwei Außenkanten von $D_I^{'}$ markiert wurden und keine davon im ersten Schritt, eine Kante eines jeden Paares hi und fg eine Außenkante von $D_I^{'}$. Da zwei Außenkanten einen gemeinsamen Knoten haben müssen, können nur g und i diese beiden Außenkanten von $D_I^{'}$ sein. Es ist nun auch klar, dass sowohl d als auch e jeweils eine freie Kante gewesen sein müssen. Andernfalls wären nicht zwei kritische Kanten beim Entfernen von a entstanden und es hätten keine zwei Schritte vom Typ 1 stattgefunden, um zwei kritische Kanten zu spannen. Zwei Außenkanten von $D_I^{'}$ hätten so nicht markiert werden können. Genauso müssen die Kanten f, g, h und i freie Kanten gewesen sein, andernfalls, wäre einer der beiden Schritte vom Typ 1 ein Gewinnschritt gewesen, ein Widerspruch zur Annahme.

Betrachte nun die Kante f und die beiden Kanten j und k, mit denen f innerhalb von $D_I^{'}$ im Dreieck liegt. Es kann nicht sein, dass k eine Außenkante ist, andernfalls müsste k mit i einen gemeinsamen Knoten haben. Diesen Knoten hätte i dann aber auch mit j gemeinsam und es entstände das Dreieck idj, welches ein trennendes Dreieck innerhalb von $D_I^{'}$ wäre.

Ist eine der Kanten k oder j eine freie Kante, so findet auf $D_I^{'}$ ein weiterer Schritt statt, der wie besprochen ein Gewinnschritt ist und der Annahme widerspricht. Sind beide Kanten schon Spanner-Kanten, so wäre anstelle des einfachen Gewinnschrittes beim Spannen von a über b und c ein Zweifachgewinnschritt möglich gewesen. Da die Routine einem Zweifachgewinnschritt den Vorrang vor einem einfachen Gewinnschritt gibt, so hätte ein Zweifachgewinnschritt stattgefunden und der Annahme widersprochen.

Als zweites wird angenommen, dass im ersten Schritt der Routine auf $D_I^{'}$ eine Außenkante von $D_I^{'}$ als Spanner-Kante markiert wird. Betrachte dazu Abbildung 14, die Argumente sind insgesamt denen im vorigen Absatz zu Abbildung 13 sehr ähnlich. Sei g die Außenkante, über die im ersten Schritt, der ein Gewinnschritt sein muss, gespannt wird, und die so zur Spanner-Kante wird: d wird als günstige Kante über f und g gespannt. Da g Außenkante von $D_I^{'}$ ist, kann von den auf der Abbildung gezeigten Kanten nur noch i auch eine Außenkante von $D_I^{'}$ sein. Ist a eine freie Kante, so wird a als kritische Kante in einem folgenden Schritt der Routine vom Typ 1 über b und c gespannt, eventuell auch erst nach dem Spannen der kritischen Kante e. Ist b bereits Spanner-Kante, so ist dies ein Widerspruch, das Spannen von a über b und c wäre der zweite Gewinnschritt der Routine auf $D_I^{'}$. War also b freie Kante und ist nach dem Spannen von a

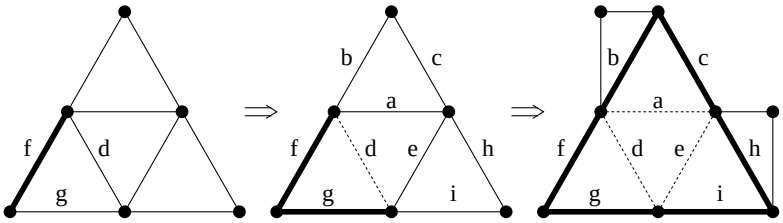


Abbildung 14: Zuerst wird d in einem Gewinnschritt über f und g gespannt, e und a als kritische Kanten werden danach gespannt

über b und c als Spanner-Kante markiert, so betrachte die beiden Kanten , mit denen b außer a und c noch im Dreieck liegt. Mit den gleichen Argumenten wie bei j und k in Abbildung 13 kann auch hier ein Widerspruch gezeigt werden.

Ist a schon Spanner-Kante, so wurde a nicht durch den ersten Schritt zur kritischen Kante und es fand kein Spannen von a über b und c statt. Nach Annahme wurden aber mindestens zwei Außenkanten von $D_I^{'}$ zu Spanner-Kanten, e muss also als kritische Kante über h und i gespannt werden. War h schon vorher Spanner-Kante, so ist dies ein zweiter Gewinnschritt im Widerspruch zur Annahme. Ansonsten kann bei den beiden Kanten, die außer e und i noch mit h im Dreieck liegen, ein Widerspruch wie bei Kante b gezeigt werden.

Induktionsschritt Fall 2B: Alle markierten Innenkanten, die $D_I^{'}$ beim Start der Routine enthält, stammen von freien K_4
Die Kanten, die beim Start der Routine auf $D_I^{'}$ schon Spanner-Kanten sind und von freien K_4 stammen, müssen von solchen freien K_4 stammen, die direkt unterhalb von $D_I^{'}$ liegen. Dies stellt die Sonderbehandlung der freien K_4 durch die Funktion `BearbeiteFreienK4` sicher. Für einen freien K_4, der direkt unterhalb von $D_I^{'}$ liegt, gibt es zwei Möglichkeiten. Entweder hat ein freier K_4 direkt unterhalb von $D_I^{'}$ eine Außenkante, die auch Außenkante von $D_I^{'}$ ist oder er hat nur Außenkanten, die alle als Innenkanten von $D_I^{'}$ auftreten.

Induktionsschritt Fall 2Bi: Alle markierten Innenkanten, die $D_I^{'}$ beim Start der Routine enthält, stammen von freien K_4, die eine Außenkante mit $D_I^{'}$ gemeinsam haben
Von dieser Sorte freier K_4 kann es maximal drei geben. Jeder dieser freien K_4 markiert bei seiner Abarbeitung durch die Routine genau zwei Außenkanten, die jetzt beide als Innenkanten von $D_I^{'}$ auftauchen und zwar als die Innenkanten von $D_I^{'}$, die mit der Außenkante, die der freie K_4 mit $D_I^{'}$ gemeinsam hat, ein Dreieck in $D_I^{'}$ bilden. In Abbildung 15 ist ein solches Beispiel gezeigt. Hier ist abc ein freier K_4, der eine Außenkante, nämlich c, mit $D_I^{'}$ gemeinsam hat. Die beiden Innenkanten, die mit c ein Dreieck bilden, im Bild a und b, wurden durch die Sonderbehandlung des freien K_4 markiert. Analoges gilt für den zweiten freien K_4 def, der die Außenkante f mit $D_I^{'}$ gemeinsam hat.

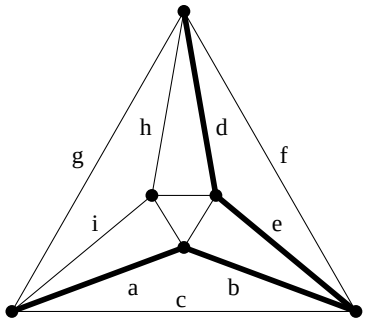


Abbildung 15: $D_I^{'}$ mit bereits vier markierten Innenkanten, die von freien K_4 direkt unterhalb von $D_I^{'}$ stammen

Betrachte nun die folgende Tabelle, die in den Zeilen die möglichen Anzahlen der freien K_4 aufführt, die Innenkanten von $D_I^{'}$ markieren, also direkt unterhalb von $D_I^{'}$ liegen und mit $D_I^{'}$ eine Außenkante gemeinsam haben. Die Spalten zeigen die möglichen Anzahlen der von der Routine bei der Bearbeitung von $D_I^{'}$ markierten Außenkanten von $D_I^{'}$.

		bei Abarbeitung von $D_I^{'}$ markierte Außenkanten			
		0	1	2	3
freie K_4	0				
direkt unterhalb von $D_I^{'}$, die Außenkante mit $D_I^{'}$ gemeinsam haben	1		$\equiv 2$ **1**	$\equiv 0$ **3**	-
	2		$\equiv 0$ **3**	-	-
	3		-	-	-

Die erste Zeile steht für den Fall, dass kein freier K_4 direkt unterhalb von $D_I^{'}$ existiert, der eine Außenkante mit $D_I^{'}$ gemeinsam hat. Dann existieren gar keine markierten Innenkanten beim Start der Routine auf $D_I^{'}$ und es sei auf den Induktionsanfang verwiesen. Die erste Zeile ist deshalb leer gelassen worden. Ebenso uninteressant ist an dieser Stelle die erste Spalte, die dafür steht, dass keine Außenkante von $D_I^{'}$ von der Routine markiert wird. Zum Beweis der Induktionsaussage ist dabei nichts zu zeigen.

Die mit einem „-“markierten Zellen der Tabelle stehen für Kombinationen, die nicht auftreten können: Die Außenkante, die ein freier K_4 mit $D_I^{'}$ gemeinsam hat, liegt in $D_I^{'}$ mit zwei Innenkanten im Dreieck, die schon als Spanner-Kanten

markiert sind, siehe die Kanten c und f der Abbildung 15. Eine solche Außenkante wie c oder f kann von der Routine bei der Abarbeitung von $D_I^{'}$ nicht markiert werden, sie ist von markierten Innenkanten eingeschlossen. Die Anzahl der Außenkanten, die die Routine auf $D_I^{'}$ markieren kann, entspricht also der Anzahl der Außenkanten von $D_I^{'}$, die gerade nicht gleichzeitig Außenkante eines direkt unterhalb von $D_I^{'}$ liegenden freien K_4 sind, so kann in der Abbildung 15 die Routine nur die Außenkante g bei der Abarbeitung von $D_I^{'}$ markieren.

Ansonsten enthält jede Zelle der Tabelle in der ersten Zeile die Anzahl der Kanten modulo 3, die durch die Routine auf $D_I^{'}$ festgelegt werden. Das sind alle Innenkanten von $D_I^{'}$ ($\equiv 0 \ mod \ 3$) abzüglich zwei schon markierte Innenkanten von $D_I^{'}$ durch jeden freien K_4 direkt unterhalb von $D_I^{'}$ (Anzahl siehe Tabellenzeile) zuzüglich der Anzahl bei der Abarbeitung von $D_I^{'}$ markierter Außenkanten (Anzahl siehe Tabellenspalte). Die zweite Zeile einer Tabellenzelle enthält die kleinste Anzahl von Gewinnschritten, die die Routine auf $D_I^{'}$ ausführt. Zum Zusammenhang von zu bearbeitenden Kanten in der ersten Zeile einer Zelle und den Gewinnschritten in der zweiten Zeile siehe Induktionsanfang Fall 2.

Betrachte die beiden Einträge der Tabelle, die für $\equiv 0 \ mod \ 3$ abzuarbeitende Kanten von $D_I^{'}$ stehen. Da in beiden Kombinationen Außenkanten von $D_I^{'}$ markiert werden, finden auch Schritte der Routine auf $D_I^{'}$ statt. In beiden Kombinationen existieren schon vor dem Start der Routine auf $D_I^{'}$ markierte Innenkanten. Das heißt, dass der erste Schritt der Routine in beiden Kombinationen ein Gewinnschritt ist. Eine Anzahl von $\equiv 0 \ mod \ 3$ Kanten lässt sich, wenn mindestens ein Gewinnschritt stattfindet nur durch mindestens drei Gewinnschritte der Routine auf $D_I^{'}$ abarbeiten. In den beiden betrachteten Tabellenzellen wird die Induktionsaussage erfüllt: Ein Gewinnschritt wird $D_I^{'}$ zugeteilt, weiterhin kann ein Gewinnschritt jedem trennenden Dreieck angeboten werden, das eine bei der Abarbeitung von $D_I^{'}$ markierte Außenkante als Innenkante enthält.

Unklar ist an dieser Stelle, wie die Zelle, die für $\equiv 2 \ mod \ 3$ abzuarbeitende Kanten steht, die Induktionsaussage erfüllen soll: Allein aus der Anzahl der abzuarbeitenden Kanten von $\equiv 2 \ mod \ 3$ kann nicht mehr als ein Gewinnschritt gefolgert werden. Dieser muss $D_I^{'}$ zugeteilt werden, ein zweiter Gewinnschritt, der laut Induktionsaussage für die bei der Abarbeitung von $D_I^{'}$ markierte Außenkante stattfinden müsste, kann daraus nicht abgeleitet werden. Damit auch diese Kombination die Induktionsaussage erfüllt, wird im Folgenden gezeigt, dass unter der Kombination der betrachteten Tabellenzelle mindestens zwei Gewinnschritte von der Routine auf $D_I^{'}$ ausgeführt werden und damit auch hier die Induktionsaussage gilt.

Betrachte dazu zuerst den Fall, dass $D_I^{'}$, beim Start der Routine unversehrt nach Lemma 5.1, die Gestalt des nach Lemma 3.6 einzigen trennenden Dreieckes größer als ein K_4 mit neun Innenkanten hat, auf dem die Routine gestartet werden kann. $D_I^{'}$ ist in Abbildung 16 links oben dargestellt. Der eine freie K_4 direkt unterhalb von $D_I^{'}$ sei abc, der die Innenkanten a und b von $D_I^{'}$ markiert hat. Offensichtlich gibt es vier günstige Kanten, damit vier Möglichkeiten für

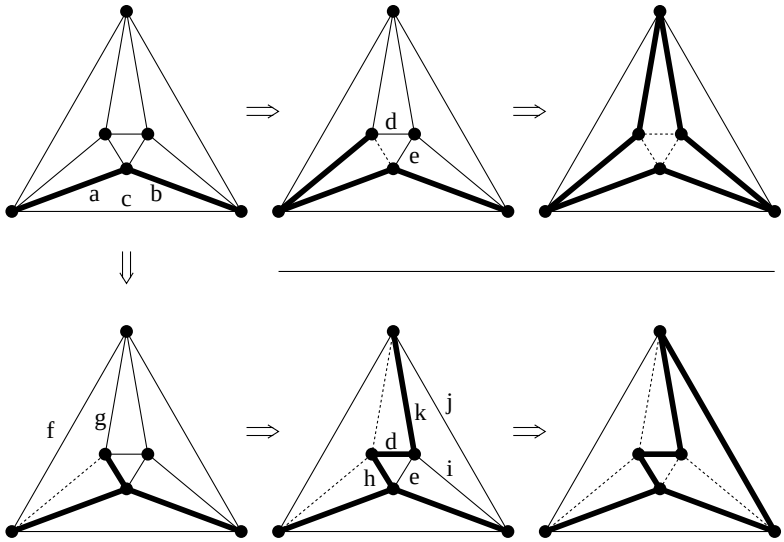


Abbildung 16: Zwei Möglichkeiten der Abarbeitung von $D_I^{'}$

die Routine in einem Schritt vom Typ 2 eine günstige Kante zu entfernen. Werden die symmetrischen Möglichkeiten zusammengefasst, so ergeben sich zwei Varianten, die beide in der Abbildung 16 aufgeführt sind.

In der oberen Variante müssen nach dem ersten Gewinnschritt die beiden kritischen Kanten d und e gespannt werden. Danach existiert keine freie Innenkante mehr und die Routine führt keine weiteren Schritte mehr aus. Es wird offensichtlich keine Außenkante von $D_I^{'}$ markiert und damit ist für die Induktionsaussage nichts zu zeigen.

In der unteren Variante braucht von den kritischen Kanten nur g gespannt werden, f ist eine Außenkante und wird als solche nicht entfernt. Da die Routine einem Zweifachgewinnschritt den Vorrang vor einem einfachen Gewinnschritt gibt, wird nun e in einem solchen Zweifachgewinnschritt über d und h gespannt. Die Kante i, die dabei kritisch wird, wird in einem letzten Schritt, nochmals ein Gewinnschritt, über j und k gespannt. Insgesamt wurde eine Außenkante von $D_I^{'}$, nämlich j, zur Spanner-Kante und es konnten vier Gewinnschritte, zwei einfache und ein Zweifachgewinnschritt, gezählt werden. Damit kann $D_I^{'}$ ein Gewinnschritt zugeteilt werden und auch dem trennenden Dreieck, in dem j als Innenkante auftritt.

Ist das trennende Dreieck $D_I^{'}$ andernfalls eines, das größer als ein K_4 ist und nicht die Gestalt in Abbildung 16 hat, so hat $D_I^{'}$ laut Lemma 3.6 mehr als neun Innenkanten, mindestens also zwölf. Zwei dieser Innenkanten sind schon vom direkt unterhalb von $D_I^{'}$ liegenden freien K_4 laut Tabellenkombination markiert. Da es sowohl markierte als auch freie Innenkanten in $D_I^{'}$ gibt, ist der erste Schritt

ein Gewinnschritt und damit verringert sich die Anzahl der freien Innenkanten von zehn auf acht. Bei diesem ersten Schritt entstehen höchstens zwei kritische Kanten, die in höchstens zwei Schritten vom Typ 1 der Routine gespannt werden. Damit werden höchstens sechs weitere freie Innenkanten festgelegt, es existieren danach mindestens zwei freie Innenkanten in $D_I^{'}$. Es muss dann ein zweiter Gewinnschritt stattfinden.

Im Falle, dass $D_I^{'}$ beim Start der Routine nur markierte Innenkanten enthält, die von freien K_4 stammen, die eine gemeinsame Außenkante mit $D_I^{'}$ haben, erfüllt $D_I^{'}$ die Induktionsaussage.

Induktionsschritt Fall 2Bii: $D_I^{'}$ enthält eine markierte Innenkante, die von einem freien K_4 stammt, der keine Außenkante mit $D_I^{'}$ gemeinsam hat
Ein freier K_4 direkt unterhalb von $D_I^{'}$, der mit $D_I^{'}$ keine gemeinsame Außenkante hat, wird der rekursiven Abarbeitungsreihefolge der Routine entsprechend nach den trennenden Dreiecken abgearbeitet, die direkt unterhalb von $D_I^{'}$ liegen und keine K_4 sind und auch nach den direkt unterhalb von $D_I^{'}$ liegenden K_4, die mit $D_I^{'}$ eine gemeinsame Außenkante haben.

In der Menge der direkt unterhalb von $D_I^{'}$ liegenden freien K_4, die keine gemeinsame Außenkante mit $D_I^{'}$ haben, und von denen nach Voraussetzung mindestens einer existiert, muss es also einen geben, der als letztes von allen unterhalb von $D_I^{'}$ liegenden trennenden Dreiecken abgearbeitet wird. Betrachte diesen freien K_4. Vor seiner Abarbeitung sind alle seine Außenkanten freie Kanten nach der Definition eines freien K_4. Während seiner Abarbeitung werden genau zwei seiner Außenkanten als Spanner-Kanten markiert, die dritte nicht. Alle seine Außenkanten sind Innenkanten von $D_I^{'}$ nach Voraussetzung. Sie bilden in $D_I^{'}$ ein Dreieck, von dem zwei Kanten markiert sind, die dritte nicht. Daran ändert sich bis zu Abarbeitungsbeginn der Routine auf $D_I^{'}$ auch nichts, da kein anderes unterhalb von $D_I^{'}$ liegendes trennendes Dreieck mehr nach diesem freien K_4 abgearbeitet wird. Die Routine kann im ersten Schritt auf $D_I^{'}$ auf den drei Außenkanten des freien K_4 einen Zweifachgewinnschritt ausführen.

Wird insgesamt bei der Abarbeitung von $D_I^{'}$ nur höchstens eine Außenkante von $D_I^{'}$ als Spanner-Kante markiert, so ist für die Induktionsaussage nichts mehr zu zeigen: Der Zweifachgewinnschritt zählt wie zwei einfache Gewinnschritte, einer kann $D_I^{'}$ zugeteilt werden, ein zweiter dem trennenden Dreieck angeboten werden, in dem die höchstens eine bei der Abarbeitung von $D_I^{'}$ markierte Außenkante als Innenkante auftritt.

Ebenso wenig ist für die Induktionsaussage zu zeigen, falls ein weiterer, dann der dritte, Gewinnschritt auf $D_I^{'}$ stattfindet: Ein Gewinnschritt wird $D_I^{'}$ zugeteilt, die Außenkanten von $D_I^{'}$ treten nach Lemma 3.4 in höchstens zwei verschiedenen trennenden Dreiecken oberhalb von $D_I^{'}$ auf, jedem kann ein Gewinnschritt zu Verrechnung angeboten werden.

Der verbleibende Fall, dass bei der Abarbeitung von $D_I^{'}$ außer dem anfänglichen Zweifachgewinnschritt kein weiterer Gewinnschritt stattfindet und minde-

stens zwei Außenkanten von $D_I^{'}$ markiert werden, wird angenommen und zum Widerspruch geführt.

Betrachte dazu Abbildung 17, die einen Teil der Innen- und Außenkanten von $D_I^{'}$ zeigt. Im ersten Schritt der Routine auf $D_I^{'}$ wird c im Zweifachgewinnschritt

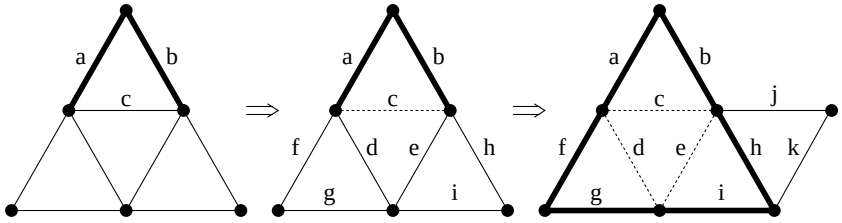


Abbildung 17: Zunächst wird c mit einem Zweifachgewinnschritt über a und b gespannt. Die dabei entstehenden kritischen Kanten d und e werden danach gespannt.

über a und b gespannt. Nach Voraussetzung ist keine der drei Kanten eine Außenkante von $D_I^{'}$. Bei den jetzt folgenden maximal zwei Schritten vom Typ 1 müssen mindestens zwei Außenkanten von $D_I^{'}$ markiert werden. Andernfalls müsste nach diesen maximal zwei Schritten vom Typ 1 noch ein weiterer Schritt stattfinden, dieser wäre aber ein Gewinnschritt, da markierte Innenkanten von $D_I^{'}$ aber keine kritischen Innenkanten von $D_I^{'}$ mehr existieren. Weder f und g gleichzeitig, noch h und i gleichzeitig können diese beiden Außenkanten von $D_I^{'}$ sein. Von den abgebildeten Kanten können nur g und i zwei Außenkanten von $D_I^{'}$ sein. Da diese also wie besprochen in den maximal zwei Schritten vom Typ 1 der Routine markiert werden, die auf den anfänglichen Zweifachgewinnschritt folgen, müssen d und e freie, damit nach dem ersten Schritt auch kritische Kanten sein, die auch keine Außenkanten sind. Sie werden also wie im Bild gespannt. h kann nicht vor dem Spannen von e schon Spanner-Kante gewesen sein, sonst wäre dieses Spannen von e ein Gewinnschritt im Widerspruch zur Annahme. Auch kann h keine Außenkante von $D_I^{'}$ sein, da i eine ist. h muss dann mit zwei weiteren Kanten von $D_I^{'}$ in einem Dreieck liegen, seien dies j und k. Beides können keine Außenkanten von $D_I^{'}$ sein, j nicht, weil sie keinen gemeinsamen Knoten mit i hat, k nicht, weil sonst g als Außenkante von $D_I^{'}$ mit j und e ein trennendes Dreieck innerhalb von $D_I^{'}$ bilden würde.

Ist eine der beiden Kanten j und k eine freie Kante, kann hier ein weiterer Gewinnschritt im Widerspruch zur Annahme stattfinden. Sind sowohl j als auch k schon markierte Kanten, so hätte statt des ersten Zweifachgewinnschrittes an dieser Stelle ein Zweifachgewinnschritt stattfinden können, h wäre über j und k gespannt worden. Da von der Routine unter mehreren möglichen Zweifachgewinnschritten der vorgezogen wird, der eine Kante entfernt, die mit einer Außenkante von $D_I^{'}$ ein Dreieck bildet, so wäre h als Kante, die mit einer Außenkante von $D_I^{'}$ ein Dreieck bildet, nämlich i, vor der Kante c entfernt worden.

c selbst kann eine solche Kante, die mit einer Außenkante von $D_I^{'}$ ein Dreieck bildet, nicht sein. Sonst wären d oder e Außenkanten von $D_I^{'}$, was wie schon besprochen zum Widerspruch führt.

An dieser Stelle endet der Induktionsschritt Fall 2Bii. Damit erfüllt $D_I^{'}$ die Induktionsaussage, wenn mindestens eine markierte Innenkante von $D_I^{'}$ vor dem Start der Routine auf $D_I^{'}$ von einem freien K_4 stammt, der keine Außenkanten mit $D_I^{'}$ gemeinsam hat.

Damit endet auch der Induktionsbeweis. Wenn $D_I^{'}$ kein freier K_4 ist, gilt also, dass allen trennenden Dreiecken, die oberhalb von $D_I^{'}$ liegen und eine Innenkante enthalten, die bei der Abarbeitung von $D_I^{'}$ markiert wurde, ein Gewinnschritt zur Verrechnung angeboten wird.

Der Beweis des Satzes kann nun beendet werden. Es sei daran erinnert, dass der Aufwand des Induktionsbeweises nur getrieben wurde, um den Fall abzudecken, dass D_I bereits beim Start der Routine auf D_I keine freien Innenkanten enthält. In diesem Falle führt die Routine auf D_I keinen Schritt und damit auch keinen Gewinnschritt aus, der D_I zugeteilt werden kann. Enthält D_I in dieser Situation eine Innenkante, die bei der Abarbeitung eines trennenden Dreieckes $D_I^{'}$ entstanden ist, der kein freier K_4 ist, so kann der Gewinnschritt, den D_I laut Induktionsaussage von $D_I^{'}$ angeboten bekommt, D_I zugeteilt werden.

Diese Argumentation könnte offenbar nicht angewendet werden, wenn alle Innenkanten von D_I durch die Abarbeitung von freien K_4 markiert wären. Dies kann aber nicht der Fall sein, wie im Folgenden klar wird. Die Annahme, D_I enthalte beim Beginn der Abarbeitung der Routine auf D_I nur durch die Abarbeitung von freien K_4 markierte Innenkanten wird zum Widerspruch geführt.

Ist D_I selbst ein K_4 und enthält damit genau drei Innenkanten, so müssten diese von mindestens zwei freien K_4 unterhalb von D_I stammen, denn von jedem freien K_4 werden genau zwei Außenkanten markiert, die als Innenkanten im direkt oberhalb liegenden trennenden Dreiecke auftauchen. Bei der Abarbeitung dieser mindestens zwei freien K_4 wären aber durch die Sonderbehandlung jeweils genau die beiden Außenkanten der beiden freien K_4 als Spanner-Kanten markiert worden, die Innenkanten von D_I sind. Damit müsste D_I mindestens vier markierte Innenkanten besitzen, was einen Widerspruch darstellt.

Ist D_I im anderen Falle größer als ein K_4, dann enthält D_I nach Lemma 3.6 mindestens neun Innenkanten. Nur genau sechs Innenkanten von D_I bilden mit einer Außenkante von D_I ein Dreieck, es gibt also eine Innenkante e, die mit keiner Außenkante von D_I ein Dreieck bildet. Sind alle Innenkanten von D_I vor dem Start der Routine auf D_I durch die Abarbeitung von freien K_4 markiert worden, so muss es einen freien K_4 unterhalb von D_I geben, bei dessen Abarbeitung e markiert wurde. Dieser freie K_4 hat keine Außenkante mit D_I gemeinsam. Unter allen diesen freien K_4, die keine Außenkante mit D_I gemein-

sam haben, gibt es wegen der Abarbeitungsreihenfolge der Routine einen, der zuletzt von allen unterhalb von D_I liegenden trennenden Dreiecken abgearbeitet wird. Betrachte diesen freien K_4. Nach Definition eines freien K_4 sind vor der Abarbeitung dieses freien K_4 alle seine Außenkanten freie Kanten. Nach seiner Abarbeitung sind genau zwei seiner Außenkanten markiert, eine Außenkante bleibt freie Kante. Diese freie Außenkante ist eine Innenkante von D_I und kann nicht während einer Abarbeitung eines anderen trennenden Dreieckes unterhalb von D_I als Spanner-Kante markiert werden. Dann aber ist diese Innenkante auch beim Start der Routine auf D_I noch eine freie Kante. Damit können nicht sämtliche Innenkanten von D_I beim Start der Routine auf D_I schon durch die Abarbeitung von freien K_4 markiert sein.

□

Lemma 5.7 hat einen direkten Zusammenhang zwischen der Anzahl der trennenden Dreiecke und der Ausführungshäufigkeit eines Gewinnschrittes hergestellt. Die Approximationsgüte des Algorithmus `Berechne2Spanner` ist damit eine einfache Folgerung aus Lemma 4.3.

Korollar 5.8 *Der Algorithmus `Berechne2Spanner`, gestartet auf einer planaren Triangulation G, erreicht in polynomieller Laufzeit eine Approximationsgüte von $\frac{5}{3} - \varepsilon$.*

Beweis
Der Algorithmus `Berechne2Spanner` aufgerufen auf einer planaren Triangulation G benutzt die Routine `BearbeiteTrennendesDreieck`. Diese führt nach Lemma 5.7 eine Anzahl von Gewinnschritten durch, die mindestens so groß ist wie die Anzahl trennender Dreiecke in G. Damit kann auf `Berechne2Spanner` Lemma 4.3 mit $\alpha = 1$ angewendet werden und es ergibt sich eine Güte von $\frac{5}{3} - \varepsilon$.

Es bleibt die polynomielle Laufzeit des Algorithmus zu zeigen. Gleichzeitig zu einer gegebenen planaren Triangulation G benötigt `Berechne2Spanner` eine Einbettung von G in die Ebene. Nach Schnyder [14] ist jeder planare Graph in ein $(n-2) \times (n-2)$-Gitter einbettbar. Dabei liegen die Knoten auf Gitterpunkten, haben damit ganzzahlige Koordinaten, die Kanten verlaufen geradlinig und die Einbettung ist in $\mathcal{O}(n)$ berechenbar. Liegt eine Einbettung dieser Gestalt nicht vor, kann sie für G vorab in polynomieller Zeit bestimmt werden.

`Berechne2Spanner` kann dann über alle Dreiecke in G gehen und mittels der allen Knoten durch die Einbettung zugewiesenen Koordinaten bestimmen, innerhalb und außerhalb welcher Dreiecke sich Knoten befinden und so die Dreiecke kennzeichnen, die trennende Dreiecke in G sind. Damit ist auch schon offenkundig, welche trennenden Dreiecke gemeinsame Außenkanten haben. Durch die Informationen, welche trennenden Dreiecke sich innerhalb von welchen anderen trennenden Dreiecken befinden, baut `Berechne2Spanner` die Baumstruktur der trennenden Dreiecke auf. Ist diese Baumstruktur bekannt, so kann aufsteigend beginnend bei den Blättern in der Baumstruktur für jedes trennende Dreieck D

die Menge seiner Innenkanten und damit der Subgraph D_I bestimmt werden. Hierdurch ist dann auch klar, welche D_I einen K_4 bilden. Da die Anzahl der trennenden Dreiecke genauso wie die Anzahl aller Dreiecke in G höchstens linear ist, ist das Sammeln dieser Informationen über G sicherlich in polynomieller Zeit möglich.

Die trennenden Dreiecke in G werden nun nacheinander von der Routine `BearbeiteTrennendesDreieck` behandelt. Sind alle unterhalb eines trennenden Dreieckes D_I liegenden trennenden Dreiecke abgearbeitet, so wird bestimmt, welche Kanten von D_I freie, günstige und Spanner-Kanten sind. Die Funktion `BearbeiteFreienK`$_4$ und `Modifiziere` laufen auf D_I sicherlich in polynomieller Zeit ab. Die Schritte 1-4 der Routine `BearbeiteTrennendesDreieck` finden insgesamt höchstens so oft statt, wie es Kanten in D_I gibt. Da Kanten von D_I entfernt und andere als Spanner-Kanten markiert werden, finden Statusveränderungen statt, Kanten in D_I werden zu kritischen oder günstigen Kanten. Allerdings können nur die Kanten in D_I einer Statusveränderung unterliegen, die mit einer Kante, die entfernt oder als Spanner-Kante gekennzeichnet wird, in einem Dreieck liegen. Folglich ist ein Update der Listen der kritischen und günstigen Kanten nach jedem der Schritte 1-4 der Routine leicht in polynomieller Zeit möglich. Insgesamt ergibt sich eine polynomielle Laufzeit der Routine auf einem trennenden Dreieck und durch die höchstens lineare Anzahl trennender Dreiecke in G auch insgesamt eine polynomielle Laufzeit für alle Durchgänge der Routine `BearbeiteTrennendesDreieck` in G.

Seit der anfänglichen Untersuchung von G kennt `Berechne2Spanner` die Kanten in G, die Innenkanten von trennenden Dreiecken sind. Diese sind entweder bereits entfernt oder als Spanner-Kanten festgelegt. Letztere werden nun ausgeblendet, als Kanten verbleiben nur solche, die keine Innenkanten eines trennenden Dreieckes sind. Leicht lassen sich diese restlichen Kanten in Spanner-Kanten, freie und günstige Kanten einteilen. Auf den restlichen Kanten arbeitet nun `A_einfach`, die Anzahl der Schritte entspricht dabei genau der Anzahl der freien Kanten. Ein Update der für `A_einfach` nötigen Liste der kritischen Kanten kann wie in der Routine `BearbeiteTrennendesDreieck` leicht nach jedem Schritt stattfinden. So ist insgesamt auch die abschließende Phase des Algorithmus `Berechne2Spanner` in polynomieller Zeit durchführbar.

□

Die Analyse der Approximationsgüte ist scharf. Betrachte dazu den Graphen G in Abbildung 18. Der triangulierte Stern besteht aus 13 Knoten und 33 Kanten. Der Stern bildet einen 2-Spanner und gleichzeitig einen Baum, für einen sparsest-2-spanner $S2S$ gilt damit $|S2S| = 12$. k aus dem Beweis der Güte nimmt damit seinen maximalen Wert von $\frac{n-4}{2}$ an. Lemma 3.7 verspricht dann $2k = n-4 = 9$ trennende Dreiecke in G, genauso viele, wie sich in G tatsächlich finden. G erreicht so die maximale Anzahl von trennenden Dreiecken, die in einer planaren Triangulation überhaupt liegen können nach Lemma 3.3.

Es gibt in G drei trennende Dreiecke, die nicht mehr unterhalb eines anderen trennenden Dreieckes liegen. Sie bilden also eine Wurzel für jeweils einen Baum

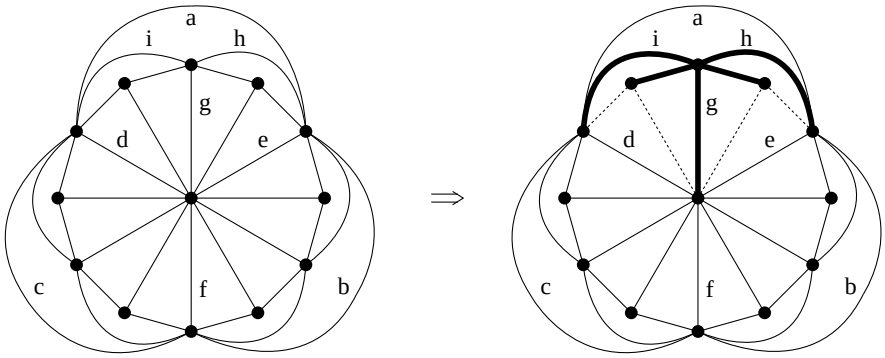


Abbildung 18: `Berechne2Spanner` auf trianguliertem Stern

der trennenden Dreiecke, die unterhalb von ihnen liegen. Diese drei trennenden Dreiecke sind ade, bef und cdf. Direkt unterhalb eines jeden von ihnen befinden sich noch jeweils zwei weitere trennende Dreiecke. Nachdem der Algorithmus `Berechne2Spanner` auf G die Baumstruktur der trennenden Dreiecke bestimmt hat, wählt er sich eines dieser drei obersten trennenden Dreiecke, o.B.d.A. sei dies ade. Der Algorithmus startet die Routine `BearbeiteTrennendesDreieck` auf ade, allerdings beginnt diese erst auf ade zu arbeiten, wenn alle trennenden Dreiecke unterhalb von ade schon abgearbeitet sind. Dies ist nicht der Fall, unterhalb von ade befinden sich die trennenden Dreiecke hge und igd, beide wurden noch nicht von der Routine abgearbeitet. Die Routine startet sich selbst auf beiden trennenden Dreiecken, da unterhalb von ihnen keine unbearbeiteten, da überhaupt keine, trennenden Dreiecke mehr liegen, findet als nächstes auch die Abarbeitung dieser trennenden Dreiecke statt. O.B.d.A. starte der Algorithmus zuerst auf hge. Als freier K_4 ist hge der Sonderbehandlung unterworfen und die Routine startet `BearbeiteFreien`K_4 auf hge. Die Funktion markiert die beiden Außenkanten h und g als Spanner-Kanten, die jeweils nicht auch gleichzeitig Außenkante vom direkt oberhalb von hge liegenden trennenden Dreieck ade sind. Bei der Abarbeitung von hge findet nach Lemma 5.4 genau ein Gewinnschritt statt.

Dann wird igd durch die Routine abgearbeitet. igd ist kein freier K_4, da die Außenkante g bereits beim Start der Routine markiert ist. Nach Lemma 5.7 Induktionsanfang Fall 1 wird durch die Routine eine beliebige zweite Außenkante von igd markiert, dies kann i sein, und es finden genau zwei Gewinnschritte statt. Da alle unterhalb von ade liegenden trennenden Dreiecke abgearbeitet sind, beginnt nun die Abarbeitung von ade durch die Routine. Dort kann sie aber keinen Schritt ausführen, die einzigen Innenkanten i, h und g sind bereits markiert. Die Abarbeitung von ade ist damit bereits beendet. Auf den drei bisher bearbeiteten trennenden Dreiecken haben genau drei Gewinnschritte stattgefunden. Soweit zeigt Abbildung 18 rechts die Situation.

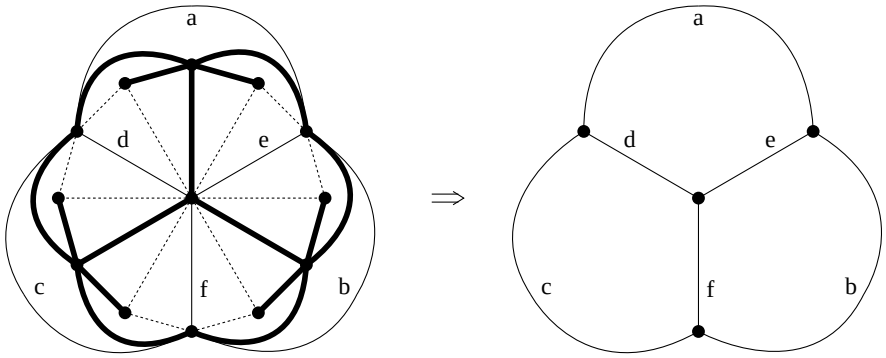


Abbildung 19: Nach Bearbeitung aller trennenden Dreiecke werden alle Innenkanten ausgeblendet

In der gleichen Weise wie auf *ade* startet der Algorithmus die Routine auf *bef* und *cdf* und es entsteht die in der Abbildung 19 links gezeigte Situation. Insgesamt haben auf den neun trennenden Dreiecken genau neun Gewinnschritte stattgefunden. Alle trennenden Dreiecke sind abgearbeitet, ihre Innenkanten werden jetzt ausgeblendet und übrig sind nur die Außenkanten der obersten trennenden Dreiecke, siehe Abbildung 19 rechts. Auf diesem Subgraphen von G kommt der Algorithmus `A_einfach` zum Einsatz, siehe dazu Abbildung 20. Der Algorithmus kann jetzt f über c und d spannen. Als kritische Kante wird

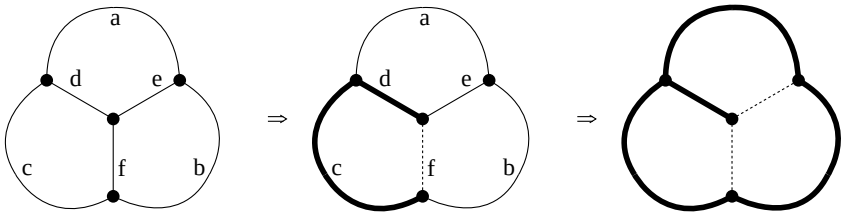


Abbildung 20: `A_einfach` auf den verbleibenden Kanten

e als nächstes entfernt und a zur Spanner-Kante gemacht, damit sichergestellt ist, dass e über zwei Spanner-Kanten gespannt ist, in diesem Falle a und d. Da `A_einfach` in jedem Schritt genau zwei freie Kanten zu Spanner-Kanten macht, wird b als letzte freie Kante auch als Spanner-Kante markiert.

Der so berechnete Spanner besteht offenbar aus 19 Kanten, die Güte der Approximation ist damit $\frac{19}{12}$. Lemma 4.3 verspricht als obere Schranke der Approximationsgüte $2 - \frac{\alpha}{3} + \frac{\alpha-2}{n-1}$, bevor der hintere Teil in einem „$-\ \varepsilon$“ verschwindet (da $\alpha < 2$). Für `Berechne2Spanner` kann wie im Beweis von Satz 5.8 ausgeführt $\alpha = 1$ gesetzt werden. Folglich ergibt sich für die Güte von `Berechne2Spanner` $2 - \frac{1}{3} + \frac{1-2}{n-1}$, mit $n = 13$ wie für den betrachteten Graphen G ist dies gleich $\frac{19}{12}$. Damit erreicht `Berechne2Spanner` auf G genau die obere Schranke der bewie-

senen Approximationsgüte.

Ausgehend vom Graphen G kann leicht eine Familie von Graphen angegeben werden, auf denen der Algorithmus wie auf G nur die bewiesene Güte erreicht. Dazu kann G in sich selbst eingefügt werden und zwar jeweils an die Stelle der sechs trennenden Dreiecke, unterhalb derer selbst keine trennenden Dreiecke mehr liegen. $S2S$ ist nach wie vor ein Baum, die Anzahl der trennenden Dreiecke gleich $n - 4$ und für jedes trennende Dreieck wird genau ein Gewinnschritt ausgeführt.

6 Schluss

In der vorliegenden Arbeit wurde der Algorithmus `Berechne2Spanner` entwickelt, der auf einer planaren Triangulation einen 2-Spanner in polynomieller Zeit bestimmt. Die dabei gefundene Lösung weicht um höchstens den Faktor $\frac{5}{3} - \varepsilon$ von der optimalen Lösung ab. Damit ist `Berechne2Spanner` der beste bekannte Algorithmus für die Bestimmung eines 2-Spanners in allgemeinen planaren Triangulationen. Der Ansatz von Duckworth et al. [6] erreicht zwar eine bessere Güte, ist aber nur auf 4-fach zusammenhängenden planaren Triangulationen anwendbar.

Allerdings ist die Komplexität des Problems, für das `Berechne2Spanner` eine Lösung approximiert bis heute ungeklärt. Es ist offen, ob die Berechnung eines sparsest-k-spanners in planaren Graphen für $k \in \{2, 3, 4\}$ ein $\mathcal{NP}$-schweres Problem darstellt. Könnte die Bestimmung eines sparsest-2-spanners in planaren Graphen als $\mathcal{NP}$-schweres Problem gekennzeichnet werden, so wäre es sehr unwahrscheinlich, dass ein polynomieller Algorithmus existiert, der einen sparsest-2-spanner eines planaren Graphen exakt berechnet. Damit wären polynomielle Approximationsalgorithmen wie `Berechne2Spanner` erste Wahl zur Bestimmung eines 2-Spanners.

Natürlich lässt der vorliegende Algorithmus und die Beschäftigung mit diesem einigen Raum für Verbesserungen. Der Beweis des zentralen Lemmas 5.7, das die Mindestanzahl der Gewinnschritte mit der Anzahl der trennenden Dreiecke gleichsetzt, ist lang und unübersichtlich. Wünschenswert ist sicherlich, die Gültigkeit dieser Aussage kürzer und eleganter zu zeigen. Dass dies nicht gelungen ist, lässt den Autor allerdings vermuten, dass einem Algorithmus, der in einer Arbeitsweise wie `Berechne2Spanner` vorgeht, keine bessere Approximationsgüte zugesichert werden kann. `Berechne2Spanner` betrachtet sukzessive und relativ isoliert voneinander die trennenden Dreiecke eines Graphen. Dabei wird versucht, einen möglichst guten 2-Spanner für das aktuelle betrachtete trennende Dreieck zu bestimmen. Die Wechselwirkungen und Beziehungen gerade benachbarter trennender Dreiecke werden kaum beachtet. Ein Algorithmus, der darauf mehr Rücksicht nimmt, könnte eine bessere Güte als `Berechne2Spanner` erreichen.

Besondere Sorgfalt widmet `Berechne2Spanner` den trennenden Dreiecken, deren Innen- und Außenkanten einen K_4 bilden. Gleichzeitig stellen diese in Form von freien K_4 auch besondere Hürden für den Beweis der Approximationsgüte dar. Naheliegend ist daher die Frage, welche Güte `Berechne2Spanner` auf einer planaren Triangulation erreicht, in der kein trennendes Dreieck D_I einen K_4 bildet. Dafür wäre dann auch von Interesse, welche Größe ein sparsest-2-spanner auf einem solchen Graphen haben kann.

Ein lohnendes Ziel in diesem Zusammenhang stellt sicherlich die Erweiterung von `Berechne2Spanner` auf allgemeine planare Graphen G dar. Insbesondere wäre dann mit Kanten umzugehen, die eventuell in weniger als zwei Dreiecken liegen. Dabei bilden Kanten von G, die in überhaupt keinen Dreiecken liegen gar kein Problem: Sie können nicht gespannt werden und müssen auf jeden Fall Teil

des 2-Spanners sein. Schwierigkeiten bereitet das Gemisch von Kanten die in einer und Kanten die in mindestens zwei Dreiecken liegen. Welche Kante dann zu entfernen und welche als Spanner-Kante zu markieren ist, scheint bei einem rein lokalen Blick auf einige Kanten kaum sinnvoll entscheidbar zu sein.

Literatur

[1] H. Bandelt, A. Dress, *Reconstructing the shape of a tree from observed dissimilarity data*, Adv. Appl. Math. 7 (1986), 309-343.

[2] U. Brandes, D. Handke, *NP-completeness results for minimum planar spanners*, Proc. 23rd International Workshop on Graph-Theoretic Concepts in Computer Science, WG'97, Lecture Notes in Computer Science, 1335 (1997), New York/Berlin, Springer-Verlag, 85-99.

[3] L. Cai, *NP-completeness of minimum spanner problems*, Discr. Appl. Math., 48 (1994), 187-194.

[4] L.P. Chew, *There is a planar graph almost as good as the complete graph*, Proceedings 2nd ACM Symposium on Computational Geometry, Yorktown Heights, NY (1986), 169-177.

[5] R. Diestel, *Graph Theory*, Heidelberg/Berlin, Springer-Verlag (2005).

[6] W. Duckworth, N.C. Wormald, M. Zito, *A PTAS for the Sparsest 2-Spanner Problem in 4-Connected Planar Triangulations*, Journal of Discrete Algorithms 1 (2003), 67-76.

[7] M.R. Garey, D.S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness*, W.H.Freeman and Company, 1979.

[8] D. Handke, *Graphs with Distance Guarantees*, Dissertation, Universität Konstanz, Fakultät für Mathematik und Informatik, 1999.

[9] G. Kortsarz, *On the Hardness of Approximating Spanners*, Proceedings of APPROX'98, Lecture Notes in Computer Science, 1444 (1998), New York/Berlin, Springer-Verlag, 135-146.

[10] G. Kortsarz, D. Peleg, *Generating sparse 2-spanners*, Proc. 3rd Scandinavian Workshop on Algorithm Theory, SWAT'92 (1992), 301-309.

[11] D. Peleg, A.A. Schäffer, *Graph Spanners*, J. Graph Theory, 13 (1989), 99-116.

[12] D. Peleg, J.D. Ullman, *An optimal synchronizer for the hypercube*, Proc. 6th ACM Symposium on Principles of Distributed Computing, Vancouver (1987), 77-85.

[13] D. Peleg, E. Upfal, *A tradeoff between space and efficiency for routing tables*, Proceedings of the 20th ACM Symposium on Theory of Computing, Chicago (1988), 43-52.

[14] W. Schnyder, *Embedding Planar Graphs on the Grid*, Proc. 1st Annual ACM-SIAM Symp. on Discrete Algorithms, San Francisco (1990), 138-147.

[15] G. Venkatesan, U. Rotics, M. Madanlal, J.A. Makowsky, C. Pandu Rangan, *Restrictions of minimum spanner problems*, Information and Computation, 136(2) (1997), 143-164.

Danksagung

An erster Stelle sei meinen Eltern Monika und Winfried Zelke gedankt, ohne deren finanzielle Unterstützung ein Hochschulbesuch für mich unmöglich gewesen wäre. An nötigem Rat und Bestärkung haben sie es nie fehlen lassen und so die besten Voraussetzungen geschaffen für meine Konzentration auf das Studium.

Den Spaß und das Interesse am theoretischen Zweig der Informatik verdanke ich PD Dr. Stefan Hougardy. Seine Vorlesungsreihe „Graphen und Algorithmen“ hat mich zur Beschäftigung mit theoretischen Fragestellungen angetrieben und führte so zur vorliegenden Arbeit. Seine Betreuung während meiner Zeit als studentische Hilfskraft am Lehrstuhl für Algorithmen und Komplexität gaben mir Einblicke in unterschiedlichste Themengebiete und Denkanstöße in verschiedene Richtungen. Nicht zuletzt verdanke ich ihm die Begleitung meiner Diplomarbeit mit hilfreichen Kommentaren und Kritik.

Ein herzlicher Dank gebührt Dr. Manuel Bodirsky für seine prompte Bereitschaft, meine Diplomarbeit überprüfend zu lesen. Weiterhin sei Stefan Kirchner gedankt, der immer ein offenes Ohr und Zeit erübrigen konnte, um auch die seltsamsten Fragen zu beantworten.

Schließlich sei meine Freundin Friederike Klose erwähnt. In einigen Bereichen mit Ermutigung, in anderen mit der Forderung nach mehr Gelassenheit fand sie immer die richtigen Worte. Jene fehlen mir nun, um meiner Dankbarkeit Ausdruck zu verleihen.

Selbstständigkeits-/Einverständniserklärung

Hiermit erkläre ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die aufgeführten Quellen und Hilfsmittel verwendet zu haben.
Ich bin damit einverstanden, dass die vorliegende Arbeit in der Bibliothek des Institutes für Informatik der Humboldt-Universität zu Berlin öffentlich ausgelegt wird.

Berlin, den 28. November 2005 Mariano Zelke