Erklärung zum modified Yao-Step angewandt auf den Delaunay Graphen

Tim Budweg

***Zusammenfassung*— Diese Arbeit erläutert die Spannerbildung mithilfe der Delaunay-Triangulation und dem modified Yao-Step aus [2]. Der *stretch-factor* liegt bei 3.54 in Bezug zum Euklidischen Graphen und beschränkt den Ausgangsgrad der Knoten auf 14.**

***Index Terms*— Spanner, ad-hoc Netze, Delaunay Graph, Modified Yao-Step**

I. Einleitung

Es gibt viele Anwendungsbereiche für drahtlose Sensornetze. Zum Beispiel können Waldbrände oder Tsunamis durch ein Sensornetz-Frühwarnsystem schneller erkannt werden. Sowohl in diesem Beispiel als auch in anderen Anwendungsgebieten ist es gewünscht, dass alle gesendeten Nachrichten ankommen. Garantierte Nachrichtenauslieferung wird von mehreren Algorithmen bereits erreicht (z.B.: Face-Routing: siehe [1]). Jedoch müssen dafür bestimmte Voraussetzungen an das Sensornetz erfüllt sein. Die wichtigste Voraussetzung ist die Planarität des Netzes, das heißt, dass keine *Kantenschnitte* (siehe II-B) existieren dürfen. Diese Planarität zu erreichen ist ein wichtiges Ziel von *Topologiekontrollen*.

Ein weiterer Punkt ist, dass diese Netze willkürlich groß werden können. Deshalb führt ein *zentralisierter* Algorithmus zu einer sehr langen Laufzeit mit hohem Energieverbrauch, weil die Menge der versendeten Nachrichten in Abhängigkeit zu der globalen Netzgröße steht.

Damit die Nachrichten nicht über beliebig lange Pfade geroutet werden müssen, wird eine Beschränkung der Pfadverlängerung gewünscht. Pfadverlängerungen entstehen, wenn (spezielle) kantenlöschende Algorithmen auf Graphen angewendet werden. Der hier behandelte Algorithmus ist streng lokal, was die Anzahl der gesendeten Nachrichten minimiert. Außerdem produziert dieser einen *t-Spanner* des euklidischen Graphen. Diese Thematik wird im Folgenden anhand des Artikel von [2] erläutert. Dort wurde ein streng lokaler Algorithmus aufbauend auf den *Delaunay Graphen* angewandt. Das Ergebnis ist ein Graph, welcher im Bezug zum euklidischen Graphen einen stretch-factor von $t = 3.54$ und zusätzlich eine Beschränkung des Ausgangsgrad eines Knoten von $k = 14$ hat.

II. Begriffe und Notationen

Im Folgenden werden die zum Verständnis der Arbeit nötigen Grundbegriffe und abkürzende Notationen erklärt.

A. Graphen

1) Euklidischer Graph: Ein euklidischer Graph ist ein spezieller Graph. Alle Knoten sind mit allen anderen Knoten verbunden (Clique) und die Kantengewichte entsprechen der euklidischen Distanz beider Eckpunkte. Ein Beispiel finden Sie in Abbildung 1.

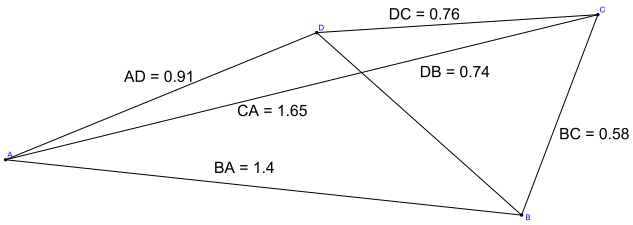

Abbildung 1. Ein euklidischer Graph

2) Unit Disk Graph: Der Unit Disk Graph ist ein euklidischer Graph ohne alle Kanten, die länger als ein konstantes $c \in \mathbb{R}$ sind. In Abbildung 2 sehen Sie den euklidischen Graph aus Abbildung 1 als Unit Disk Graph mit $c = 1$.

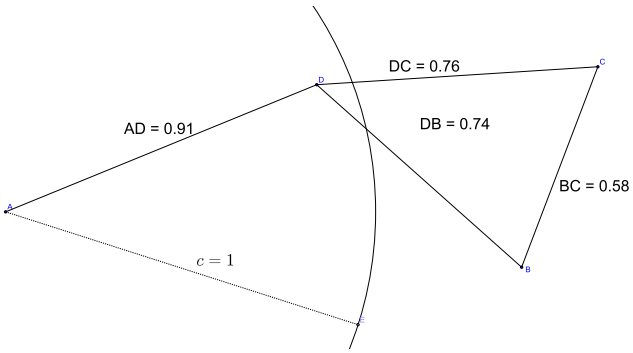

Abbildung 2. Der Unit-Disk-Graph zu Abbildung 1 mit $c = 1$

B. Kantenschnitte

Ein Kantenschnitt ist ein Punkt, in dem sich zwei oder mehr Kanten desselben Graphen schneiden.

C. Spanner

Gegeben ist ein Graph G, welcher ein Subgraph vom euklidischen Graphen E ist. G enthält alle Knoten von E, aber nicht alle Kanten. Die Umwege, die durch das Löschen von Kanten entstehen, dürfen nur um einen konstanten Faktor ansteigen.

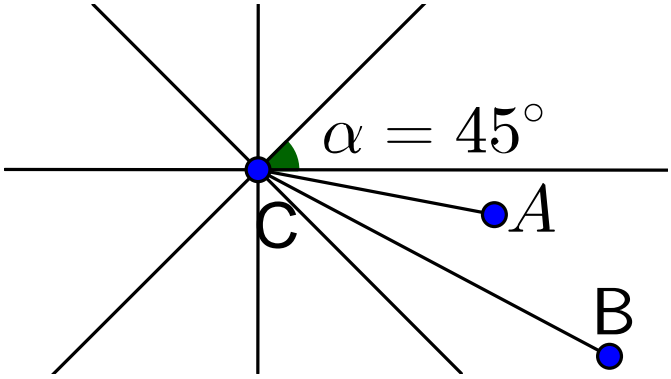

Abbildung 3. Punkt C mit acht $45°$ großen Kegeln. Kürzeste Kante AC wird ausgewählt

1) Euklidischer Spanner: H ist genau dann ein euklidischer Spanner von G, wenn die kürzesten Pfade zwischen allen Knoten maximal um einen konstanten Faktor t vergrößert werden. Die Notation $c_G(A, B)$ bedeutet: der kürzeste Pfad zwischen A und B im Graphen G. Formal bedeutet das:

$$c_H(A, B) \leq t \cdot c_G(A, B) \tag{1}$$

(Formel entnommen aus [2] und angepasst)

D. Yao Step

Gegeben ist ein Graph G. Für alle Knoten $A \in G$ wird folgender Algorithmus ausgeführt:

- Erzeuge k gleich große Kegel um A. $k \in \mathbb{N} > 6$
- Bestimme die kürzeste Kante in jedem Kegel ausgehend von C.
- Lösche alle Kanten, die nicht von beiden Endpunkten ausgewählt wurden.

E. Delaunay Triangulation

Die Delaunay Triangulation erzeugt aus einem beliebigen zusammenhängenden Graphen einen geometrischen (= euklidischen) Spanner mit dem Streckungsfaktor $c_{del} \approx 2.42$ und einem beliebig hohen Ausgangsgrad eines Knotens. Dazu werden alle Dreiecke betrachtet. Wenn der Kreis durch alle Eckpunkte des Dreiecks keine weiteren Punkte des Graphen enthält, sind diese drei Kanten auch im Delaunay Graphen. Um Fallunterscheidungen zu vermeiden wird angenommen, dass keine vier Punkte auf einem Kreis liegen.

F. Synchronizität

Bei einem synchronen Algorithmus ist die Arbeitsweise der Knoten in Runden und diese in Phasen eingeteilt. Eine Runde sieht beispielsweise wie folgt aus:

- Zuerst erhalten alle Knoten ihre Nachrichten, falls es welche gibt.
- Danach verarbeiten sie die Nachrichten und stellen ggf. weitere Berechnungen an.
- Am Ende der Runde werden Nachrichten an andere Knoten verschickt.

Hier ist zu beachten, dass alle gesendeten Nachrichten zu Beginn der Folgerunde angekommen sind.

G. Lokale Algorithmen

Ein Algorithmus ist genau dann lokal, wenn jeder Knoten ausschließlich mit seiner *k-Hop* Nachbarschaft kommuniziert. ($k \in \mathbb{N}$) Formal reicht dies aus, um einen Algorithmus lokal zu nennen. Das Problem, dass der Algorithmus eine lange Laufzeit hat (siehe Einleitung), ist dadurch jedoch nicht gelöst. Anhand des *Maximal Independent Set (Fortan: MIS)* Algorithmus lässt sich ein Beispiel konstruieren, welches dieses Problem veranschaulicht. Betrachten Sie Abbildung 4.

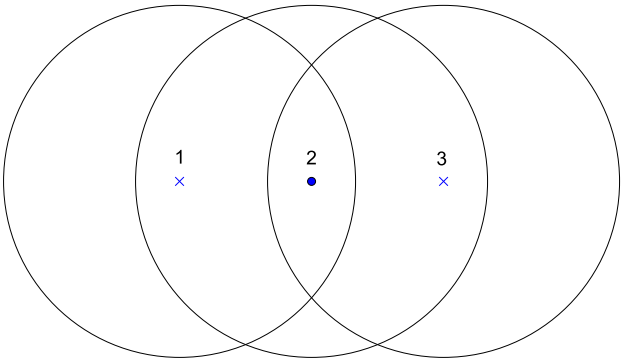

Abbildung 4. $\times$ sind Knoten im Maximal Independent Set, $\bullet$ sind keine Knoten des MIS, Die Kreise um die Knoten sind die jeweiligen Sendebereiche.

- **Runde 1**: Knoten 1 wird MIS-Knoten, da er von allen seinen Nachbarn, welche noch keine Rolle haben, derjenige mit der kleinsten ID ist. Knoten 2 und 3 warten, da es Knoten gibt, welche noch keine Rolle und eine kleinere ID haben.
- **Runde 2**: Knoten 2 entscheidet, dass er kein MIS-Knoten ist, da er in seiner Nachbarschaft Knoten 1 kennt, der schon MIS-Knoten ist. Knoten 3 wartet.
- **Runde 3**: Knoten 3 wird MIS-Knoten, da Knoten 2 kein MIS-Knoten ist und somit 3 die kleinste ID hat.

Diese Schritte lassen sich durch den gesamten Graphen sukzessiv durchführen. Der Algorithmus hat demnach eine Laufzeit in der Größenordnung von

$$\theta(dia(G))$$

wobei $dia(G)$ für den Durchmesser (längster kürzester Pfad) des Graphen steht.

Die Definition des lokalen Algorithmus trifft hier zu, weil jeder Knoten nur mit seinen unmittelbaren Nachbarn, der 1-Hop-Nachbarschaft, kommuniziert.

H. Strenge Lokalität

Wie im Abschnitt II-G beschrieben, sind lokale Algorithmen noch nicht ausreichend, um den Nachrichtenaufwand zu verringern. Strenge Lokalität ist gegeben, wenn ein *synchroner* Algorithmus unabhängig von der Netzgröße in konstanter Zeit terminiert (siehe [3]).

III. Modified Yao Step

Im Folgenden wird der Algorithmus anhand eines Beispiels verfolgt. Der Graph E, welchen wir für dieses Beispiel verwenden, ist in Abbildung 5 zu sehen.

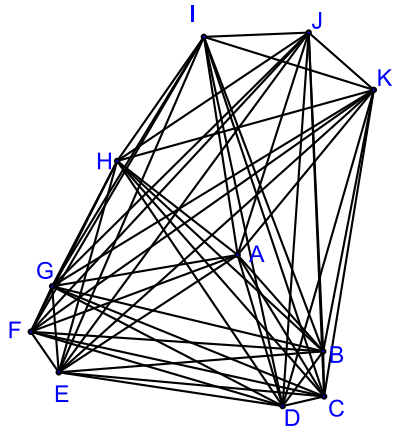

Abbildung 5. Ein euklidischer Graph, welchen wir als Beispiel nutzen.

Daraus wird zuerst der Delaunay Graph gebildet, indem durch die Eckpunkte aller Dreiecke ein Kreis gebildet wird. Genau dann, wenn in einem Umkreis $\bigcirc KLM$ kein Punkt des Graphen enthalten ist, sind die drei Kanten, welche KLM verbinden, im Delaunay Graphen vorhanden. Das Ergebnis ist in Abbildung 6 zu sehen.

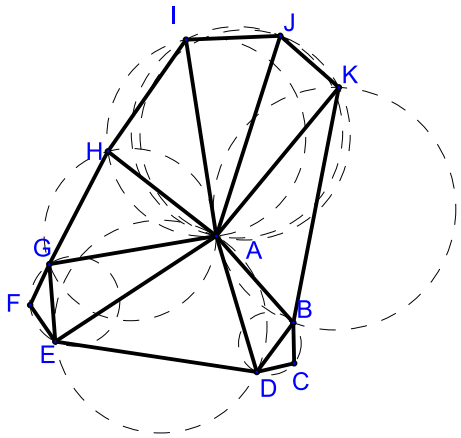

Abbildung 6. Ein Delaunaygraph mit den Umkreisen, welche keinen weiteren Punkt beinhalten.

Der Umkreis durch $\triangle ABC$ beinhaltet beispielsweise den Punkt D und deshalb werden die drei Kanten AB, BC und AC vorerst nicht in den Delaunay-Graphen übernommen. Für die Kanten AB und BC gibt es jedoch einen anderen Umkreis, der keine weiteren Punkte enthält und deshalb sind diese Kanten im Delaunay-Graphen enthalten (Umkreis durch die drei Punkte BCD für die Kante BC und $\bigcirc ABD$ für die Kante AB.)

Des Weiteren legen wir einen Integer Parameter $k = 8$ fest. Um die Spannereigenschaften zu gewährleisten, muss $k \geq 14$ gelten, aber wir ignorieren diese Bedingung, da sie für das Verständnis des Algorithmus keine Rolle spielt. Der Vorteil, den wir durch diese Vereinfachung erhalten, ist, dass die Übersichtlichkeit der Abbildungen verbessert wird.

Wir betrachten Punkt A. Um diesen Punkt müssen k Strahlen gebildet werden, die von A ausgehen. Dabei spielt es keine Rolle, wie der erste Strahl angeordnet wird. (Im Beispiel verläuft er horizontal nach rechts.) Die Winkel zwischen zwei aufeinander folgenden Strahlen müssen alle gleich groß sein. Durch diesen Vorgang bilden sich k gleich große Kegel, deren Spitzen in A liegen. Abbildung 7 zeigt diesen Vorgang.

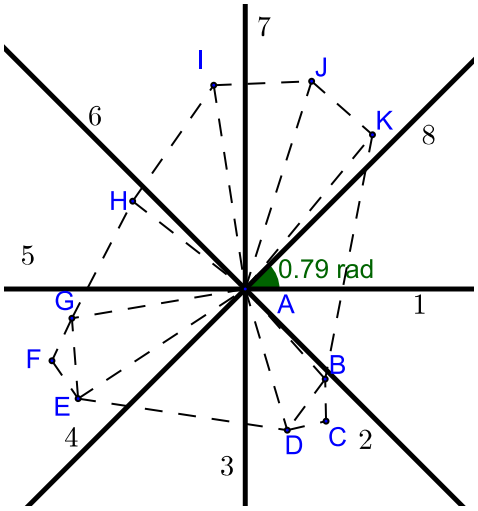

Abbildung 7. k gleich große Kegel um A. Jeder Kegel hat einen Öffnungswinkel von 0.79 rad.

Als Nächstes wählen wir in jedem - nicht leeren - Kegel die von A ausgehende, kürzeste Kante aus. Das sind in diesem Beispiel die Kanten AB, AG, AH, AI und AK (vgl. Abbildung 8). Die Nummerierung in der Abbildung bezieht sich sowohl auf die am nächsten stehende Kante als auch auf den Kegel, in welchem die Zahl enthalten ist. Kante 1 und Kegel 1 sind somit eindeutig durch die 1 definiert.

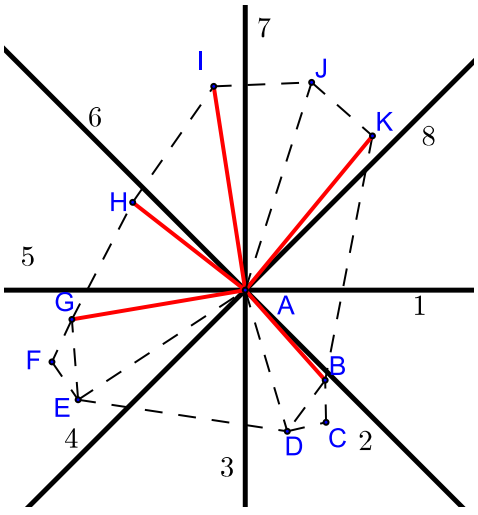

Abbildung 8. Auswahl der Kanten AB, AG, AH, AI und AK und Nummerierung der Kanten und Kegel.

Im nächsten Schritt iteriert der Algorithmus über alle Kegel, welche keinen Knoten enthalten. Zwei oder mehr aufeinander

folgende leere Kegel, werden nur einmal behandelt. Kegel 1 und 8 sind leer. Da beide Kegel aneinander liegen, werden diese zusammengefasst und als ein Iterationsschritt interpretiert. Für die Anzahl der aufeinander folgenden leeren Kegel werden zwei unterschiedliche Fälle unterschieden: Fall 1 wird behandelt, wenn $l > 1$ und sonst Fall 2 ($l = 1$). Da momentan $l = 2$ ist (Kegel 1 und 8), wird Fall 1 ausgeführt. Jetzt werden diejenigen Kanten betrachtet, welche von A ausgehen und noch nicht ausgewählt wurden. Das sind die Kanten AD, AE und AJ (, denn die anderen von A ausgehenden Kanten, wurde bereits im Schritt zuvor ausgewählt.) Von dem durch Kegel 1 und Kegel 2 bestimmten Segment wählt der Algorithmus die im Uhrzeigersinn ersten $\frac{l}{2}$, noch nicht selektierten Kanten ausgehend von A aus. In diesem Fall wird *eine* nächste Kante ausgewählt (, da $\frac{l}{2} = 1$ ist). Dies ist die Kante AD (,da AB bereits ausgewählt wurde).

Dieser Vorgang wird ebenfalls gegen den Uhrzeigersinn ausgeführt. Dort wird die Kante AJ ausgewählt. (vgl.: Abbildung 9)

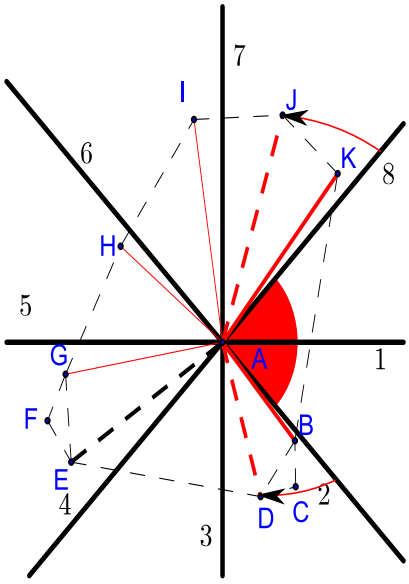

Abbildung 9. Auswahl der Kanten AJ und AD

IV. Geometrische Spanner

Es gilt folgendes Theorem zu beweisen:

Theorem IV.1. *Für jedes $k \geq 14$ existiert ein Subgraph G' vom Delaunay-Graph G, sodass G' einen maximalen Ausgangsgrad von k und einen stretch-factor von $1 + 2\pi(k \cos \frac{\pi}{k})^{-1}$ hat.*

Gegeben sind zwei Kanten CA und CB im Delaunay-Graph G vom Euklidischen Graph E. Ohne Beschränkung der Allgemeinheit sei CA die kürzeste Kante. Die Autoren von [2] beweisen dieses Theorem mithilfe des *outward* und des *inward* Path. Der outward Path kann gebildet werden, wenn kein Knoten von G innerhalb von $\triangle ABC$ liegt. Der inward Path verbindet die Knoten A und B, wenn Knoten innerhalb von $\triangle ABC$ liegen.

A. Der outward Path

Die Autoren von [2] benutzen eine rekursive Definition um den Pfad von A nach B zu definieren. Es gilt zu beweisen, dass dieser Pfad in jedem Fall existiert, weil anderenfalls der Graph nicht zwingend verbunden ist. Dieser Fall darf nicht eintreten. Außerdem muss nachgewiesen werden, dass die Länge dieses Pfades maximal $1+2\pi(k \cos \frac{\pi}{k})^{-1}$ länger ist als der vorherige Pfad von A nach B.

- **Basisfall**: Die Rekursion endet, wenn $AB \in G$. (vgl. Abbildung 3: AB müssen verbunden sein, weil B der einzige Knoten in einem Kegel von A ist.)
- **Rekursionsfall**: $AB \notin G$

Aufgrund der Delaunay-Eigenschaft, dass eine Kante nur dann nicht in G enthalten ist, wenn im Kreis $\bigcirc ABC$ ein weiterer Punkt liegt, muss deshalb ein weiterer Punkt in $\bigcirc ABC$ liegen. Diesen Punkt nennen die Autoren von [2] einen *intermediate Point*. Um den Pfad von A nach B zu erhalten, wird ein Pfad von A zum intermediate Point T und von diesem zu B erstellt und konkateniert (siehe Abbildung 10). Dieser Rekursionsfall kann öfter eintreten, da es bezüglich A und T oder B und T einen weiteren intermediate Point T' geben kann und dazwischen rekursiv fortführend bis der Basisfall eintritt. Der Rekursionsfall bewahrt immer folgende Eigenschaften:

- (O_1) und (O_2) sind vollständig innerhalb von (O) (vgl. Abbildung 11)
- α und β sind beide kleiner als der Winkel $\angle AOB$ (vgl. Abbildung 12)
- Alle drei Winkel sind kleiner als $\frac{4\pi}{k}$

Abbildung 13 zeigt ein Beispiel eines kompletten outward Path.

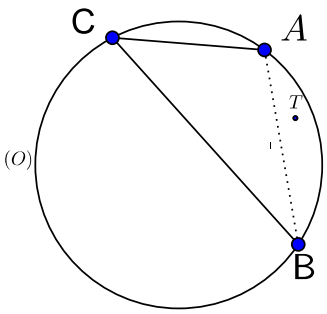

Abbildung 10. T $\in G$ ist ein Knoten. Dieser Knoten wird intermediate Point mit Bezug zu A und B genannt.

B. Beweis

Es wird behauptet, dass von A nach B ein Pfad p existiert, sodass:

$$|CA| + |p| \leq (1 + 2\pi(\cos \frac{\pi}{k})^{-1})|CB| \qquad (2)$$

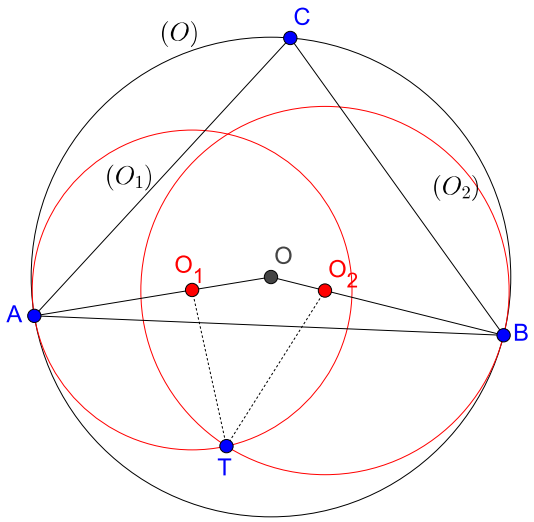

Abbildung 11. (O_1) und (O_2) sind vollständig innerhalb von (O), (O_1) ist der Kreis durch A und T mit Mittelpunkt auf dem Segment AO, (O_2) verläuft durch B und T und der Mittelpunkt liegt auf dem Segment BO

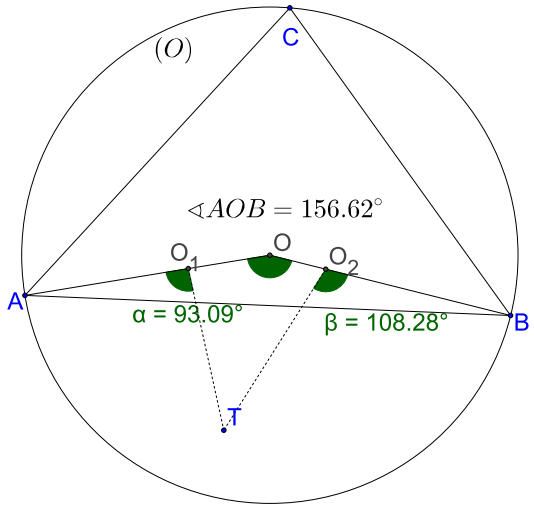

Abbildung 12. α und β sind beide kleiner als AOB, gleiche Konstruktion wie in Abbildung 11, A und B liegen im selben Kegel ausgehend von C.

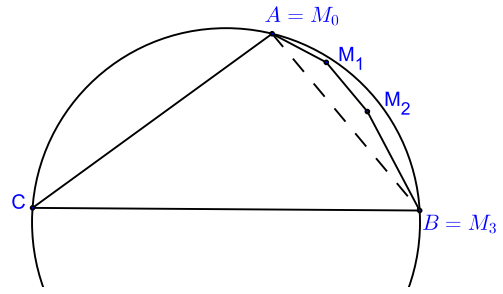

Abbildung 13. Zwischen A und B ist ein fertiger Outward Path mit den Intermediate Points M_1 und M_2

In Worten: Die Länge des Pfades p plus den Wert $|CA|$ darf nicht größer sein als eine Konstante ($t \approx 1.45$ für $k = 14$) multipliziert mit der Länge der Kante CB. Es gelten folgende Bedingungen und Formeln:

- $|AB| = 2 \cdot \angle BCA \cdot |OA|$ mit O als Mittelpunkt des Kreises $\bigcirc ABC$ (siehe Abbildung 14 für ein Beispiel)
- $\sin\theta = \frac{|AB|}{2|OA|}$
- $|CA| = |CB|$ Denn dadurch ist $|CA| + |AB|$ maximal

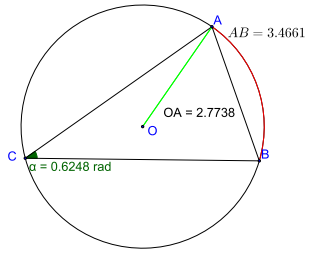

Abbildung 14. Beispiel zu Formel IV-B: $|AB| = 2 \cdot 0.6248 \cdot 2.7738 \approx 3.4661$

C. Der inward Path

Die Menge S besteht aus allen Punkten innerhalb von $\triangle ABC$, Punkt A und Punkt B. In Abbildung 15 ist ein Beispiel eines Inward Path. $CH(S)$ sind alle Punkte der Konvexen Hülle von S. Folgende drei Voraussetzungen gelten:

Proposition 1.
- $CN_i \in G$
- $|CN_i| \leq |CN_{i+1}|$
- $\angle N_{i-1}N_iN_{i+1} \geq \pi$ *(Winkel Richtung C)*

Abbildung 15 zeigt, dass alle Winkel $\angle N_{i-1}N_iN_{i+1}$ größer als π sind. Außerdem ist jede Kante CN_i kürzer als CN_{i+1}. Da CA ($= CN_0$) die kürzeste Kante in diesem Kegel ist, muss CN_1 länger sein. Für CN_1 gilt dieselbe Bedingung. Folglich muss CN_2 länger sein, etc.

Hinzu kommt die Voraussetzung $\angle N_iCN_{i+1} \leq \angle BCA$. Dies sind genau die Bedingungen, welche im Voraus bewiesen wurden, um einen outward Path zu konstruieren. Zwischen jedem Paar N_iN_{i+1} gibt es einen outward Path, der dieses Paar verbindet. Diese gesamte Konstruktion, der Pfad von A nach B, wird inward Path genannt.

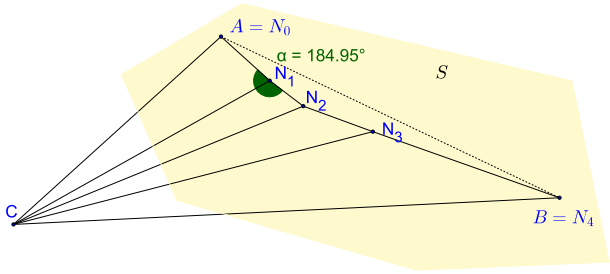

Abbildung 15. Voraussetzungen für den inward Path und Beispiel eines inward Path $N_0N_1N_2N_3N_4$

1) Beweis:

LITERATUR

[1] H. Frey and I. Stojmenovic, "On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks," in *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '06. New York, NY, USA: ACM, 2006, pp. 390–401. [Online]. Available: http://doi.acm.org/10.1145/1161089.1161133

[2] I. A. Kanj and L. Perkovic, "On geometric spanners of euclidean and unit disk graphs," *CoRR*, vol. abs/0802.2864, 2008. [Online]. Available: http://arxiv.org/abs/0802.2864

[3] J. Suomela, "Survey of local algorithms," *ACM Comput. Surv.*, vol. 45, no. 2, pp. 24:1–24:40, Mar. 2013. [Online]. Available: http://doi.acm.org/10.1145/2431211.2431223