Lösungsvorschlag: Triangulierung konzyklischer Punkte

Algorithmus. Gegeben ist ein Polygon P bestehend aus n Punkten, die alle auf dem selben Kreis liegen. Wir wissen, dass der schwache Dualgraph der optimalen Triangulierung von P ein Pfad ist für den die dualen Sehnen den lexikographisch größten Längenvektor haben. Sei π dieser Pfad. Unser Plan ist es, die zu den Kanten in π dualen Sehnen ihrer Größe nach sortiert (aufsteigend) aufzuzählen. Betrachte dazu zunächst die Kante e in π, die dual zur längsten Sehne ist. Löscht man e, so wird π in zwei Teilpfade π_1 und π_2 zerlegt. Beachte, dass wenn man in π_i (für $i \in \{1, 2\}$) von dem Blatt in Richtung e läuft, so werden die zugehörigen Sehnen immer größer. Wir müssen also im Prinzip nur einen Merge-Sort-Schritt auf diesen beiden Pfaden ausführen, um die sortierte Liste der Sehnen zu bekommen.

Zunächst müssen wir die ersten Knoten von π_1 und π_2 (also die Blätter von π) finden. Dazu wählen wir die zwei maximalen Ohren. Überlappen sich diese, so wählen wir das größte und drittgrößte Ohr. Überlappen diese sich ebenfalls, so wählen wir das zwei- und drittgrößte Ohr (diese können sich dann nicht auch noch überlappen). Diese Ohren können in $\mathcal{O}(n)$ Zeit gefunden werden. Seien s_1 und s_2 die zu diesen Ohren gehörenden Sehnen und seien v_1 und v_2 die zugehörigen Knoten im Dualgraph. Wir initialisieren π_1 mit v_1 und π_2 mit v_2. Außerdem speichern wir uns für jeden der beiden Pfade eine Menge von Kandidaten-Sehnen, die als nächstes hinzugefügt werden. Diese Mengen nennen wir kand_1 und kand_2 und initialisieren $\text{kand}_i = \{s_i\}$ für $i \in \{1, 2\}$.

In jedem Schritt machen wir nun das Folgende. Wir wählen den Pfad aus, für den das Maximum der Kandidaten kleiner ist, erweitern ihn mit dem maximalen Kandidaten und erzeugen neue Kandidaten. Die Idee dahinter ist die, dass wir jeden Pfad möglichst mit der größten Kandidaten-Sehne fortsetztten. Da wir die Sehnen sortiert aufzählen wollen müssen wir den Pfad wählen, für den das Maximum kleiner ist. Um das etwas formaler zu machen, setzte $i = 1$, falls die längste Sehne in kand_1 kürzer ist als die längste Sehne in kand_2. Sonst, setze $i = 2$. Nun erweitern wir π_i, indem wir die Kante dual zur längsten Sehne in kand_i anhängen. Sei s_i diese Sehne mit den Endpunkten p und q. Seien außerdem s'_i und s''_i die *Nachbarsehnen* von s_i in Laufrichtung, also die Sehnen, die p mit einem Nachbarn von q, bzw. q mit einem Nachbarn von p verbinden, siehe Abbildung 1. Schließlich wird $\text{kand}_i = \{s'_i, s''_i\}$ gesetzt.

Diesen Prozess iterieren wir, bis die letzten eingefügten Sehnen für π_1 und π_2 sich einen Endpunkt teilen. Dann verbinden wir diesen Endpunkt mit allen noch übrigen gegenüberliegenden Punkten, sodass die Sehnenlängen aufsteigend sind. Da wir so für jede eingefügte Sehne nur $\mathcal{O}(1)$ Zeit benötigen, läuft dieser Prozess insgesamt in $\mathcal{O}(n)$ Zeit ab.

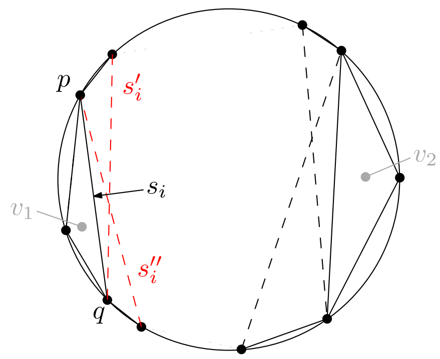

Abbildung 1: Erweiterung von v_1 aus mit der Kante dual zu s_i. In rot sind die die beiden neuen Kandidaten s'_i und s''_i dargestellt.

Korrektheit. Um die Korrektheit zu zeigen sei zunächst angemerkt, dass die hinzugefügten Sehnen immer länger werden. Wir zählen die Sehnen also tatsächlich entsprechend des Längenvektors auf. Außerdem können wir zeigen, dass wir in jedem Schritt die längste Sehne wählen, aus einer Menge von Sehnen, aus der wir mindestens eine wählen müssen. Jede andere Wahl hätte unweigerlich zu einem lexikographisch kleineren Längenvektor geführt, womit wir das Optimum finden. Um zu zeigen, dass wir tatsächlich in jedem Schritt aus einer Menge von Sehnen wählen, aus der wir eine wählen müssen, betrachte drei Fälle.

1. Für die Wahl der ersten Sehne wissen wir, dass wir wenigstens zwei der Ohren-Sehnen wählen müssen. Wir müssen also neben der größten Ohren-Sehne noch eine weitere wählen. Schneiden sich die größte und die zweitgrößte Ohren-Sehne, so müssen wir aus der Menge aller Ohren-Sehnen ohne die größten zwei eine Sehne wählen. In beiden Situationen wählen wir im ersten Schritt das Maximum dieser Menge.

2. Nimm ohne Beschränkung der Allgemeinheit an, dass im ersten Schritt π_1 erweitert wurde. Nun wird π_1 weiter erweitert, bis die längere der beiden Sehnen in kand_1 länger ist als die Ohrensehne in kand_2. Beachte, dass wir zu diesem Zeitpunkt noch eine Ohrensehne wählen müssen. Dabei dürfen wir keine wählen, die eine schon für π_1 gewählte Sehnen schneidet. Glücklicherweise enthält kand_2 genau die längste unter all diesen Ohren-Sehnen. Wir wählen also wieder die längste Sehne aus einer Menge von Sehnen, aus der wir eine wählen müssen.

3. Der dritte Fall ist quasi der Normalfall. Sei π_i der Pfad der erweitert werden soll. Da die letzten Sehnen der beiden Pfade sich noch keinen Endpunkt teilen hat π_i die Kante e (dual zur maximalen Sehne) noch nicht erreicht und muss erweitert werden. Es gibt aber nur zwei mögliche Sehnen, die π_i erweitern können und diese sind in kand_i gespeichert. Aus dieser Menge wählen wir die größere der beiden Sehnen.

In dem Moment, in dem die letzten Sehnen von π_1 und π_2 sich einen Endpunkt teilen haben wir π_1 und π_2 genau genommen noch nicht komplett fertig berechnet (es fehlen ggf. noch mehr Kanten als nur e). Für die restlichen Sehnen bleibt aber auch keine Wahl mehr, daher hören wir hier schon auf und wählen für den Rest die einzige mögliche Triangulierung.