Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Zuzana Beerliova

Rekonstruktion von Graphen

Diploma Thesis DA-2004-02
Winter Term 2003/2004

Tutor: Prof. Dr. Thomas Erlebach

Supervisor:
Prof. Dr. Thomas Erlebach

10.3.2004

Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

Zusammenfassung

Das Graphrekonstruktions-Optimierungsproblem ist im Offline-Fall[1] wie folgt definiert: Zu einem gegebenen Graphen $G = (V, E)$ finde eine minimale Menge $M \subseteq V$, so dass durch Messung der Punkte in M der Graph G rekonstruiert wird. Durch die notwendige Präzisierung der Begriffe "Messung" und "Rekonstruktion" ergeben sich verschiedene Varianten des Problems. Die Messung eines Punktes $v \in V$ kann alle kürzesten Wege von v zu allen übrigen Knoten (LG-Modell) oder nur einen Kürzeste-Wege-Baum (Shortest Path Tree) mit Wurzel v (SPT-Modell[2]) liefern. Rekonstruktion kann bedeuten, alle Kanten von G (ALL-E-Modell, nur im Offline-Fall sinnvoll) oder alle Kanten und Nichtkanten von G (ALL-Modell) in Erfahrung zu bringen.

In [2] und [1] wurde die NP-Vollständigkeit von LG-ALL-E und die $O(\log n)$-Approximierbarkeit ($n = |V|$) von LG-ALL und LG-ALL-E bewiesen. Hier zeigen wir die NP-Vollständigkeit von LG-ALL und des hier betrachteten SPT-ALL-E und SPT-ALL Modells, sowie die $o(\log n)$-Nichtapproximierbarkeit von LG-ALL-E und LG-ALL. Weiter zeigen wir die optimale Lösbarkeit von LG-ALL-E (in polynomieller Zeit) für eine spezielle Klasse ausserplanarer Graphen.

[1] Im Online-Fall ist zu Beginn nur die Knotenmenge V von G bekannt

[2] Hier muss man weiter präzisieren, auf welche Art und Weise die Messprozedur aus mehreren kürzesten Wegen von v zu v' einen auswählt, sowie ob diese Information bei der Rekonstruktion benützt werden darf. Wir betrachten lediglich den Fall, wo die Eindeutigkeit der kürzesten Wege mittels einer zu G gehörenden Gewichtsfunktion, die auch bei der Rekonstruktion bekannt ist, erreicht wird.

Inhaltsverzeichnis

Aufgabenstellung für die Diplomarbeit

„Rekonstruktion von Graphen“

Institut TIK, D-ITET, ETH Zürich
Wintersemester 2003/2004

Studentin:	Zuzana Beerliova
Aufgabensteller:	Prof. Dr. Thomas Erlebach
Pate D-MATH:	Prof. Dr. Hans-Jakob Lüthi
Betreuung:	Prof. Dr. Thomas Erlebach
Beginn:	30.10.2003
Ende:	05.03.2004

Thematischer Hintergrund

In Kommunikationsnetzen wie dem Internet gibt es keine zentrale Stelle, die vollständige Informationen über alle Netzwerkelemente hat. Deshalb wird nach Verfahren gesucht, die es ermöglichen, die Struktur eines unbekannten Netzes durch Messungen herauszufinden. Eine Messung könnte etwa darin bestehen, dass man von einem Knoten aus die Wege im Netz von diesem Knoten zu allen anderen Knoten in Erfahrung bringt. Das Ziel ist es dann, mit möglichst wenig Messungen die Netzstruktur möglichst genau zu bestimmen.

Diese Problemstellung lässt sich mathematisch wie folgt modellieren. Das Netzwerk ist ein zusammenhängender ungerichteter Graph $G = (V, E)$. Die Kantenmenge des Komplementärgraphen von G wird mit $\bar{E}$ bezeichnet. Wir nennen $\bar{E}$ auch die Menge der *Nichtkanten*. Bei jedem Knoten des Graphen kann eine Messung durchgeführt werden. Eine Messung bei Knoten $v \in V$ liefert eine Menge $E_v \subseteq E$ von Kanten, die sicher im Graph enthalten sind, und evtl. auch eine Menge $\bar{E}_v \subseteq \bar{E}$ von Kanten, die sicher nicht im Graph enthalten sind. Wie die Mengen E_v und $\bar{E}_v$ aussehen, hängt vom Messmodell ab. Wenn man an jedem Knoten einer Menge $M \subseteq V$ gemessen hat, weiss man, dass die Kanten der Menge $E_M := \bigcup_{v \in M} E_v$ sicher in G enthalten sind und die Kanten der Menge $\bar{E}_M := \bigcup_{v \in M} \bar{E}_v$ sicher nicht im Graphen enthalten sind. Ziel ist es nun, mit möglichst wenig Messungen M Informationen über den Graphen zu erhalten (in Form der Mengen E_M und $\bar{E}_M$), die eine gewisse Zielvorgabe erfüllen (z.B. dass $E_M = E$ und $\bar{E}_M = \bar{E}$ sein soll).

Aus algorithmischer Sicht kann diese Problemstellung sowohl im Online-Fall (wo der Algorithmus den Graph nicht kennt und den nächsten Messknoten in Abhängigkeit der bisherigen Messergebnisse bestimmt) als auch im Offline-Fall (wo der Algorithmus den Graph kennt) studiert werden.

Ziel der Diplomarbeit ist es, diese Problemstellung für verschiedene Messmodelle

und Zielvorgaben theoretisch zu untersuchen. Die wichtigsten Messmodelle sind hierbei:

- Alle kürzesten Wege (Abkürzung: LG, von *layered graph*): E_v ist die Menge aller Kanten, die auf einem kürzesten Weg (bezüglich Anzahl der Kanten) von v zu irgendeinem anderen Knoten liegen. $\bar{E}_v$ ist die Menge aller Nichtkanten, deren Endknoten verschiedene Abstände von v haben.
- Kürzeste-Wege-Baum (Abkürzung: SPT, von *shortest-path tree*): E_v ist die Kantenmenge eines Kürzeste-Wege-Baums mit Startknoten v. Um den Baum eindeutig zu machen, nehmen wir an, dass alle Kantenlängen zufällig in $[1-\varepsilon, 1+\varepsilon]$ für $\varepsilon \ll 1$ gewählt sind. Die *Schichtnummer* eines Knotens im Baum ist als sein Abstand (Anzahl Kanten) von der Wurzel definiert. $\bar{E}_v$ ist dann die Menge aller Nichtkanten, deren Endknoten im Kürzeste-Wege-Baum Schichtnummern haben, die sich um mindestens 2 unterscheiden.

Die wichtigsten Varianten der Zielvorgaben sind:

- Exakte Rekonstruktion (Abkürzung: ALL): $E_M = E$ und $\bar{E}_M = \bar{E}$
- Exakte Rekonstruktion der Kanten (Abkürzung: ALL-E): $E_M = E$
- α-Rekonstruktion: $|E_M| \geq \alpha \cdot |E|$ und $|\bar{E}_M| \geq \alpha \cdot |\bar{E}|$
- Parameter-Rekonstruktion: Ziel ist, einen gewissen Graphparameter (z.B. Durchmesser, Cliquenzahl) bestimmen zu können.

Für die einzelnen Modellvarianten sind die folgenden Punkte von Interesse:

- Algorithmen, die online oder offline mit möglichst wenig Messungen die Zielvorgabe erfüllen können, und ihre Analyse (Approximationsrate oder kompetitive Rate).
- Komplexitätsbestimmung im Offline-Fall (welche Varianten sind NP-schwer, welche sind polynomiell lösbar?)
- untere Schranken für die Güte von Online-Algorithmen
- obere und untere Schranken für die optimale Anzahl Messungen in speziellen Graphklassen und im allgemeinen

Teilaufgaben

Da es sich um eine Diplomarbeit mit theoretischer Ausrichtung handelt, können die erzielbaren Ergebnisse nicht vorhergesehen werden. Die zu bearbeitenden Teilaufgaben sind:

1. Arbeiten Sie sich in die Thematik ein und machen Sie sich mit den Ergebnissen der bisher durchgeführten Untersuchungen zu dem Thema in [2] und [1] vertraut.

2. (Hauptteil) Untersuchen Sie verschiedene Problemvarianten und versuchen Sie, neue Ergebnisse zu erarbeiten. Versuchen Sie, unter anderem die folgenden Fragestellungen zu beantworten:

   - Ist das Offline-Problem im Fall von LG-Messungen und Zielvorgabe ALL NP-schwer?
   - Ist das Offline-Problem im Fall von SPT-Messungen und Zielvorgabe ALL-E NP-schwer?
   - Gibt es für die Offline-Varianten des Problems bessere Algorithmen als von SetCover übertragene $O(\log |V|)$-Approximationsalgorithmen?
   - Wie viele Messungen benötigt man im Fall von SPT-Messungen bei einfachen Graphklassen wie z.B. Radgraphen?
   - Gibt es für das Online-Problem im Fall von LG-Messungen und Zielvorgabe ALL einen Algorithmus mit guter kompetitiver Rate? Welche unteren Schranken kann man zeigen?

3. Fassen Sie die Ergebnisse Ihrer Arbeit in einem schriftlichen, gut strukturierten Schlussbericht zusammen.

4. Präsentieren Sie ihre Untersuchungen und die erzielten Ergebnisse in einem Vortrag, der voraussichtlich am Institut für Operations Research bei Prof. Lüthi stattfinden wird.

Viel Erfolg!

Literaturverzeichnis

[1] F. Eberhard. Grundlagen für das Mapping des Internet-Graphen, Juli 2003. Semesterarbeit SA-2003.30, Computer Engineering and Networks Laboratory (TIK), ETH Zürich.

[2] L. S. Ram. Tree-based graph reconstruction, 2002. CGC Pre-Doc Project, ETH Zürich.

1 Definitionen, Bekannte Ergebnisse

Wir betrachten ein Netzwerk als einen ungerichteten, ungewichteten und zusammenhängenden Graphen $G = (V, E)$. Die Kantenmenge des Komplementärgraphen von G wird mit $\bar{E}$ bezeichnet. Wir nennen $\bar{E}$ auch die Menge der Nichtkanten. Eine Messung beim Knoten $v \in V$ liefert eine Menge $E_v \subseteq E$ von Kanten, die sicher im Graph enthalten sind, und eine Menge von Nichtkanten $\bar{E}_v \subseteq \bar{E}$, die sicher nicht im Graphen enthalten sind. Wie die Mengen E_v und $\bar{E}_v$ aussehen hängt vom Messmodell ab.

Definition 1. Wir sagen, dass eine Kante $e \in E$ (Nichtkante $e' \in \bar{E}$) durch eine Menge von Messpunkten $S \subseteq V$ *entdeckt wird*, wenn $\exists v \in S : e \in E_v$ ($e' \in \bar{E}_v$). Alle Kanten des Graphen G werden *rekonstruiert (entdeckt)* durch S, wenn gilt: $E = E_S := \cup_{v \in S} E_v$. Der ganze Graph G wird *rekonstruiert (entdeckt)* durch S, wenn zusätzlich gilt: $\bar{E} = \bar{E}_S := \cup_{v \in S} \bar{E}_v$.

Definition 2.

(i) Das Problem, zu einem gegebenen Graphen $G = (V, E)$ eine Menge $S \subseteq V$ mit minimaler Kardinalität zu finden, die den ganzen Graphen rekonstruiert, heisst das offline *ALL-Graphrekonstruktion Optimierungsproblem*.

Das zugehörige Entscheidungsproblem lautet: Sei $G = (V, E)$ und eine ganze Zahl k gegeben. Ist es möglich den Graphen durch eine Menge $S \subseteq V$ zu rekonstruieren, die höchstens k Punkte enthält?

Bei der online Variante dieses Problems ist zu Beginn nur die Knotenmenge von G bekannt, die Messpunktmenge wird immer aufgrund der bisherigen Messergebnisse bestimmt.

(ii) Das *ALL-E-Graphrekonstruktionsproblem*, bei dem nur alle Kanten von G rekonstruiert werden sollen, ist analog definiert.[3]

Bei einem online Problem werden wir sagen, dass eine Kante von einer Messung entdeckt (nicht entdeckt) wird auch wenn noch nicht klar ist, ob es sich um eine Kante oder eine Nichtkante handelt. Bei einem offline Problem werden wir hingegen (da möglich) immer zwischen Kanten und Nichtkanten unterscheiden.

Jetzt folgt noch eine allgemeine Definition.

Definition 3. Seien $\Pi_1 = (L_1, \Sigma_1)$ und $\Pi_2 = (L_2, \Sigma_2)$ zwei Entscheidungsprobleme. Wir sagen, dass Π_1 *auf* Π_2 *reduzierbar ist*, wenn eine Abbildung $\rho : \Sigma_1^* \to \Sigma_2^*$ existiert, so dass $\forall x \in \Sigma_1^*$: $x \in L_1 \iff \rho(x) \in L_2$.
Ist ρ zusätzlich in polynomieller Zeit berechenbar, so ist Π_1 *in polynomieller Zeit reduzierbar auf* Π_2.

Bei Optimierungsproblemen wird mit Reduzierbarkeit (wenn nicht explizit anders gesagt) auch nur die Reduzierbarkeit der entsprechenden Entscheidungsprobleme gemeint, die keineswegs Aussagen über die Beziehung zwischen den approximativen Lösungen der Optimierungsprobleme macht.

[3] Dieses Problem ist nur im Offline-Fall sinnvoll.

1.0.1 Bekannte Ergebnisse

In [2] und [1] wurden folgende Aussagen über das LG-Modell [4] des Graphrekonstruktionsproblems bewiesen.

Satz 4.

(i) Um alle Kanten eines vollständigen Graphen der Grösse n zu entdecken sind $n-1$ Messungen notwendig und hinreichend.

(ii) Um alle Kanten und Nichtkanten eines Kreises zu entdecken sind 2 Messungen notwendig und hinreichend.

(iii) Für einen Graphen mit einer Clique der Grösse k sind mindestens $\log_2 k$ Messungen notwendig, um alle Kanten von G zu entdecken.

Weiter wurde in [2] und [1] die optimale Anzahl Messungen für Radgraphen und für Bäume bestimmt. Hier folgt noch eine allgemeine untere Schranke aus [1]:

Satz 5. *Gegeben sei ein Graph $G = (V, E)$. Für einen Subgraphen $H = (V', E')$ mit $V' \subseteq V$, $E' \subseteq E$ und Durchmesser $diam_H$ gilt: Es werden mindestens $\log_{(diam_H+1)}(|V'|)$ Messknoten aus V benötigt, um alle Kanten und Nichtkanten von G zu entdecken.*

Und zuletzt noch zwei wichtige Aussagen über die Komplexität des Graphrekonstruktionsproblems:

Satz 6. *Das LG-ALL-E Entscheidungsproblem ist NP-vollständig.*

Satz 7. *Es gibt eine $O(\log n)$-Approximation für den Offline-Fall des LG-ALL-E und des LG-ALL Optimierungsproblems.*

[4] Das LG-Modell wird im nächsten Kapitel genau definiert.

2 Das LG-Modell

In diesem Kapitel werden wir ein Messmodell des Graphrekonsruktionsproblems betrachten, in dem die Messung eines Knotens v alle Kanten liefert, die auf einem kürzesten Weg von v zu irgendeinem anderen Knoten liegen.

Definition 8. Sei v ein Knoten des Graphen $G = (V, E)$. *Ein Schichtengraph (layered graph) mit Wurzel* v, abgekürzt $LG(v)$, bezeichnet den Kürzeste-Wege-Teilgraphen von G, der alle kürzesten Wege von v zu allen anderen Knoten enthält.

Bemerkung 9. Ein Schichtengraph ist i.A. kein Baum. Eine Kante $e = (t, u)$ ist genau dann im $LG(v)$ enthalten, wenn t und u verschiedene Abstände von v haben. Wir sagen auch, dass sich t und u nicht auf dem selben Level befinden.

Definition 10. Das *LG-Modell* ist ein Messmodell des Graphrekonstruktionsproblems, in dem die Messung eines Knotens v den Schichtengraph $LG(v)$ liefert.

Bemerkung 11. In diesem Modell ist also $E_v = LG(v)$ und $\bar{E}_v$ ist die Menge aller Nichtkanten, deren Endpunkte verschiedene Abstände von v haben.

Beispiel. Abbildung 1 zeigt einen Graphen $G = (V, E)$ mit der Knotenmenge $V = \{a, b, c, d, e\}$ und den Schichtengraph $LG(a)$. Man sieht leicht, dass die Kante (b, c) und die Nichtkante (c, d) von a nicht entdeckt werden, hingegen wird zum Beispiel die Nichtkante (c, e) von a entdeckt.

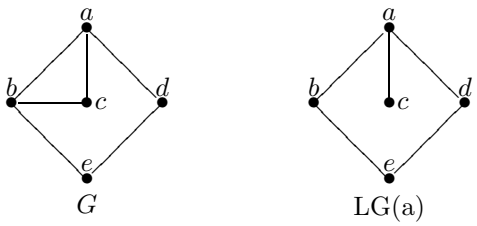

Abbildung 1:

Beispiel. Für die Entdeckung aller Kanten und Nichtkanten eines Kreises sind 2 Messungen notwendig und hinreichend.

Definition 12. Das Problem, zu einem gegebenen Graphen $G = (V, E)$ eine Menge $S \subseteq V$ mit minimaler Kardinalität zu finden, die den ganzen Graphen rekonstruiert, heisst das *LG-ALL-Graphrekonstruktion Optimierungsproblem*. Das zugehörige Entscheidungsproblem lautet: Sei $G = (V, E)$ und eine ganze Zahl k gegeben. Ist es möglich den Graphen durch eine Menge $S \subseteq V$ zu rekonstruieren, die höchstens k Punkte enthält? Das *LG-ALL-E Graphrekonstruktionsproblem*, bei dem nur alle Kanten von G rekonstruiert werden sollen, ist analog definiert.

Beide Entscheidungsprobleme sind NP-vollständig. Hier soll die NP-Vollständigkeit von LG-ALL gezeigt werden. Die NP-Vollständigkeit von LG-ALL-E wurde in [2] durch eine Reduktion von 3-SAT bewiesen. Beide Beweise sind technisch ähnlich. Wir wollen uns zuerst an Hand des übersichtlicheren Beweises von LG-ALL-E mit den wesentlichen Elementen vertraut machen.

2.1 Beweisidee der NP-Vollständigkeit von LG-ALL-E

Zu jeder Instanz ϕ von 3-SAT konstruieren wir eine Instanz (G, k) von LG-ALL-E, so dass gilt: Die boolsche Formel ϕ ist genau dann erfüllbar, wenn eine Menge $S \subset V$ von k Punkten existiert, die alle Kanten des Graphen G rekonstruiert.

Gegeben eine boolsche Formel ϕ mit den n Variablen $x_1, x_2, \ldots, x_n$ und den m Klauseln $C_1, C_2, \ldots, C_m$ gehen wir dazu wie folgt vor:

1. Für jede Variable x_i konstruieren wir einen Variablenbaustein (VB_i) mit zwei Literalknoten x_i und $\bar{x}_i$ und zwei Hilfsknoten a_i und b_i gemäss Abb. 2 a).

2. Für jede Klausel C_j konstruieren wir einen Klauselbaustein (KB_j) mit einem Klauselknoten C_j und einem Hilfsknoten C'_j gemäss Abb. 2 b).

3. Wir führen einen Hilfsknoten v ein, der mit allen Knoten des Graphen verbunden ist. Dies garantiert uns, dass alle kürzesten Wege in G höchstens Länge 2 haben.

4. Wir verbinden jeden Klauselknoten C_j mit allen in der Klausel C_j enthaltenen Literalen.

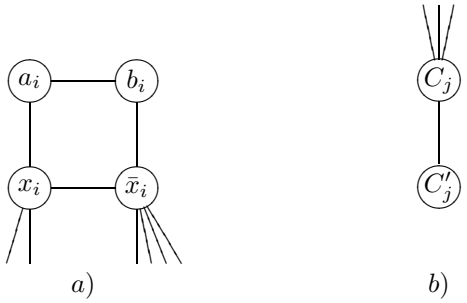

Abbildung 2: Variablenbaustein für LG-ALL-E

Bemerkung 13. Die Variablenbausteine sind so konstruiert, dass durch beliebig viele Messungen ausserhalb des Variablenbausteines VB_i mindestens eine Kante unentdeckt bleibt (die Kante (a_i, b_i)). Somit ist in jedem Variablenbaustein eine Messung (o.B.d.A. an einem Literal) notwendig, um alle Kanten der Variablenbausteine zu entdecken. Es ist leicht zu sehen, dass dies auch hinreichend ist.

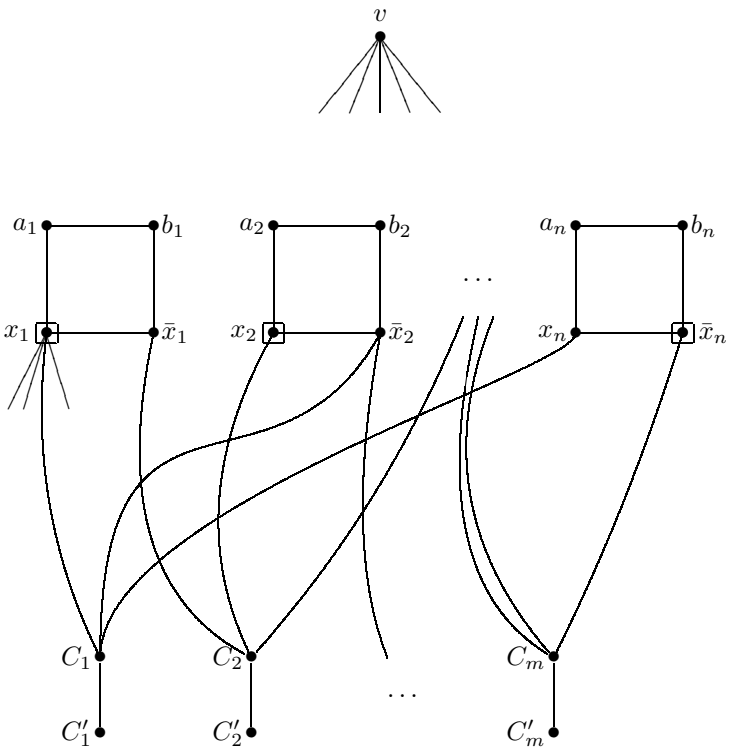

Abbildung 3: Graph zur Reduktion von 3-SAT auf LG-ALL-E.

Bemerkung 14. Für die Entdeckung der einzigen Kante K_j innerhalb eines Klauselbausteins KB_j gilt: Falls in jedem VB_i ein Knoten gemessen wird, wird K_j genau dann (ohne zusätzliche Messungen) entdeckt, wenn der Klauselknoten C_j mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden ist.

Behauptung 15. *Die boolsche Formel ϕ mit n Variablen ist genau dann erfüllbar, wenn eine Menge $S \subset V$ von n Punkten existiert, die alle Kanten von G entdeckt.*

Beweis.

$\Rightarrow$: Gegeben eine ϕ erfüllende Wahrheitsbelegung, definieren wir die Messpunktmenge $S \subset V$ als die Menge der wahren Literale. Wir zeigen hier, dass diese die Kanten der Klauselbausteine entdeckt. (Es sei dem Leser überlassen, nachzuprüfen, dass auch alle übrigen Kanten entdeckt werden, sobald in jedem VB ein Literal gemessen wird.[5]) Da ϕ erfüllt ist, enthält jede Klausel mindestens ein wahres Literal, also ist jeder Klauselknoten (nach Konstruktion von G und Definition von S) mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden. Mit Bemerkung 14 folgt die Behauptung.

[5] Unabhängig davon, ob die zugehörige Wahrheitsbelegung ϕ erfüllt.

$\Leftarrow$: Sei $S \subset V$ eine n-punktige Menge, die alle Kanten des Graphen G entdeckt. Nach Bemerkung 13 enthält S aus jedem der n Variablenbausteine genau einen Knoten. Nach Bemerkung 14 ist somit jeder Klauselknoten mit mindestens einem gemessenen Literalknoten verbunden. Setzen wir nun alle gemessenen Literale wahr, erhalten wir eine Wahrheitsbelegung, die ϕ erfüllt. [6]

2.2 Die NP-Vollständigkeit von LG-ALL

Die NP-Vollständigkeit von LG-ALL beweisen wir ebenfalls durch eine Reduktion von 3-SAT. Zu jeder Instanz ϕ von 3-SAT (mit n Variablen und m Klauseln), konstruieren wir eine Instanz (G, k) von LG-ALL, so dass gilt: Die boolsche Formel ϕ ist genau dann erfüllbar, wenn der Graph $G = (V, E)$ durch eine Menge $S \subset V$ von k Punkten rekonstruierbar ist.

Der Grap G wird wieder aus n Variablenbausteinen, m Klauselbausteinen und einem zusätzlichen Hilfspunkt v bestehen, der mit allen anderen Knoten von G verbunden ist. Die Variablenbausteine (VB) bestehen aus den beiden Literalknoten und einer Menge von Hilfsknoten. Die Klauselbausteine (KB) bestehen aus einem Klauselknoten und aus Hilfsknoten.

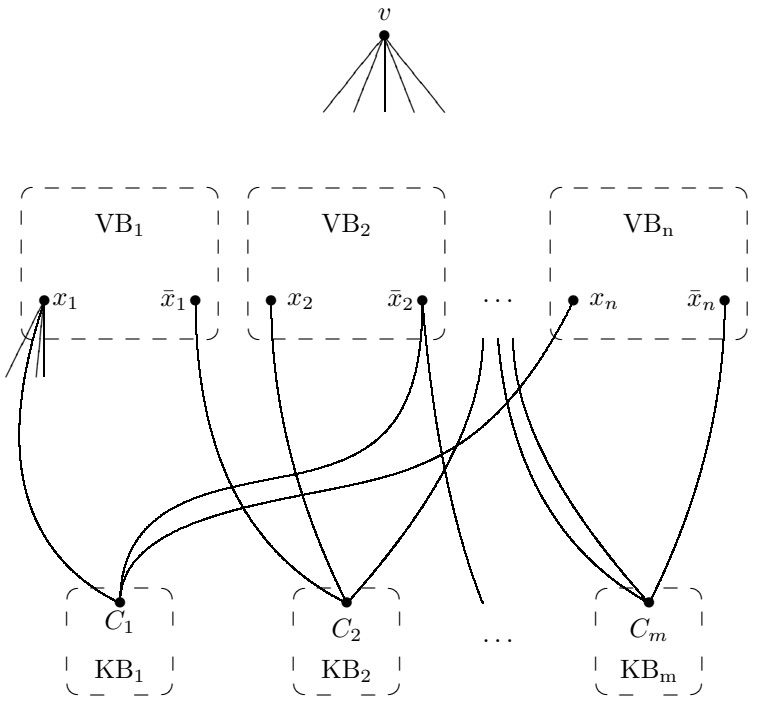

Abbildung 4: Schematische Darstellung des zu konstruierenden Graphen G

[6] Wird keiner der Punkte x_i, $\bar{x}_i$ gemessen, so kann x_i nach Belieben wahr oder falsch gesetzt werden.

Ausser den Kanten zu v und den Kanten, die jeden Klauselknoten mit seinen 3 Literalknoten verbinden, gibt es keine weiteren Kanten zwischen den Bausteinen. Da jeder VB und KB nur via den Literalknoten resp. Klauselknoten mit den anderen Punkten des Graphen (ausser v) verbunden ist, haben alle VB- und KB-Hilfspunkte den Abstand 2 von allen Knoten ausserhalb ihres Bausteins (ausser von v).

(Daraus folgt die später wichtige Eigenschaft, dass die internen Beziehungen zwischen den Hilfspunkten eines Bausteins nur durch eine direkte Messung an dessen Knoten entdeckt werden.)

Soweit ist die Situation gleich wie im vorherigen Beweis. Da wir aber ausser den Kanten auch alle Nichtkanten entdecken müssen, sind die Bausteine etwas komplizierter und die Menge von Messpunkten, die den ganzen Graphen G entdeckt, grösser.

Bemerkung 16. Vorher mussten wir nur für je 2 Punkte, die durch eine Kante verbunden sind, einen Messpunkt finden, von dem sie verschiedene Abstände haben, jetzt müssen wir das für beliebige 2 Punkte tun. Insbesondere darf, um alle Kanten und Nichtkanten von G durch eine Messpunktmenge zu entdecken, höchstens ein Punkt existieren, der von allen Messpunkten Abstand 2 hat.

2.2.1 Konstruktion der Klauselbausteine

Für jede Klausel C_j konstruieren wir aus dem Klauselknoten C_j und zwei Hilfsknoten C'_j und C''_j ein Klauseldreieck. Dieses wird nur via den Klauselknoten C_j mit anderen Knoten (ausser v) des Graphen verbunden.

Lemma 17. *Eine Menge von Messpunkten, die alle Kanten der Klauselbausteine entdeckt, enthält für jedes j einen der Hilfspunkte C'_j, C''_j und entdeckt auch alle Nichtkanten zwischen den Klauselbausteinen, alle Kanten zwischen den Klauselbausteinen und v, sowie alle Kanten und Nichtkanten von den KB-Punkten zu den VB-Punkten.*

Beweis. Für jedes j gilt: Die Hilfspunkte C'_j und C''_j haben beide Abstand 2 zu allen Punkten ausserhalb ihres Klauselbausteins (ausser zu v) und Abstand 1 zu dem Klauselknoten C_j und zu v. Die Kante (C'_j, C''_j) wird also nur durch eine Messung an C'_j oder C''_j entdeckt. Werde nun o.B.d.A C'_j gemessen. Die Kanten (C'_j, C''_j), (C'_j, v) und alle Nichtkanten ausgehend von C'_j werden also trivialerweise entdeckt. Da C_j und C''_j die einzigen Punkte (ausser v) sind, die zu C'_j Abstand 1 haben, werden die Kanten und Nichtkanten von diesen beiden Punkten zu allen anderen Punkten (ausser zu v) entdeckt. Die Kanten von C_j und C''_j zu v werden durch eine Messung in einem anderen (beliebigen) Klauselbaustein entdeckt, da dann v den Abstand 1 und C_j und C''_j den Abstand 2 haben.

Bemerkung 18. Durch die Messung an C'_j wird die Kante (C_j, C''_j) nicht entdeckt, es ist also entweder eine zusätzliche Messung in KB_j notwendig oder diese Kante muss durch eine Messung an einem der 3 Klauselliterale entdeckt werden.

(Es gibt keinen anderen Messknoten in G, der diese Kante entdeckt.) Die Klauselbausteine haben daher wieder die wichtige Eigenschaft, dass ein gemessenes (wahres) Literal einer Klausel eine Messung im KB spart.

2.2.2 Konstruktion der Variablenbausteine

Wir brauchen einen Variablenbaustein, für den folgendes gilt:

1. Es gibt eine gewisse Anzahl Messungen innerhalb des Variablenbausteins, die notwendig und gleichzeitig hinreichend für die Entdeckung des Variablenbausteins ist (unabhängig davon, wieviele Messungen ausserhalb des Variablenbausteins stattfinden).
2. Eine minimale Menge von Messungen, die einen Variablenbaustein entdeckt, enthält höchstens eines seiner beiden Literale.
3. Für jeden der beiden Literalknoten eines Variablenbausteins gibt es weitere Messpunkte, die mit dem Literalknoten zusammen eine minimale, den ganzen Variablenbaustein entdeckende Menge bilden.
4. Eine Menge von Messpunkten, die alle Variablenbausteine entdeckt, entdeckt auch alle Nichtkanten zwischen den Variablenbausteinen.

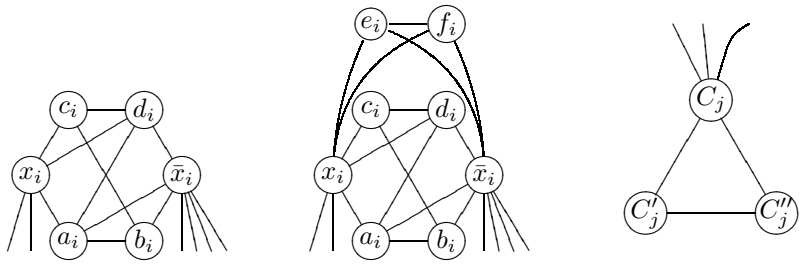

Abbildung 5: Bausteine zur Reduktion von 3-SAT auf LG-ALL: a) Eine Vorstufe des Variablenbausteins. b) Der komplette Variablenbaustein. c) Der Klauselbaustein.

Wir fangen mit einem VB an, der 2 Literalknoten und 4 Hilfsknoten enthält[7], die miteinander wie in Abb 5 verbunden sind. Weiter soll gelten, dass die Hilfsknoten mit keinen anderen Punkten (ausser mit v) durch eine Kante verbunden sind. Für einen so eingebetteten 6-Eck-Variablenbaustein VB_i gilt:

Lemma 19.

(i) Um den ganzen Variablenbaustein zu rekonstruieren sind genau 2 Messungen innerhalb des Variablenbausteines notwendig und hinreichend. (unabhängig davon wieviele Messungen ausserhalb stattfinden)

[7] Unsere Versuche mit 4-,5-punktigen Variablenbausteinen waren erfolglos.

(ii) Die beiden Literalknoten x_i und $\bar{x}_i$ reichen nicht um den ganzen Variablenbaustein zu entdecken.

(iii) Für jeden der beiden Literalknoten x_i und $\bar{x}_i$ existiert im VB_i ein Hilfsknoten, so dass diese beiden Knoten (der Literalknoten und der Hilfsknoten) den ganzen VB_i entdecken.

Beweis.

(i) Notwendig: Die 4 Hilfspunkte a_i, b_i, c_i und d_i haben von dem Hilfspunkt v alle den Abstand 1, von allen anderen Punkten ausserhalb des VB_i alle den Abstand 2, ihre internen Beziehungen können also durch Messungen ausserhalb des Variablenbausteins VB_i nicht entdeckt werden. Um alle Kanten und Nichtkanten des Kreises $a_i b_i c_i d_i$ zu entdecken sind also mindestens 2 Messungen innerhalb des Variablenbausteins nötig.

Hinreichend: Man kann leicht nachprüfen, dass die Messungen an den Knoten x_i und c_i oder $\bar{x}_i$ und b_i den ganzen VB_i entdecken. Dies beweist auch (iii).

(ii) Die Messung an den beiden Literalknoten x_i und $\bar{x}_i$ ist nicht hinreichend, da die Kante (a_i, d_i) unentdeckt bleibt.

Die 6-Eck-Variablenbausteine erfüllen also die ersten 3 Anforderungen. Das Problem ist die vierte Forderung. Bei den Messungen an x_i und c_i oder $\bar{x}_i$ und b_i hat der nicht gemessene Literalknoten von beiden Messpunkten den Abstand 2, die Nichtkanten zwischen den nichtgemessenen Literalen aller Variablenbausteine werden somit nicht entdeckt. Dieses Problem lösen wir, indem wir jedem Variablenbaustein zwei weitere Hilfspunkte hinzufügen, die wir gemäss Abb 5 b) mit den beiden Literalknoten des Variablenbausteins verbinden. (Dies macht eine dritte Messung im VB notwendig.) Nun sind alle Voraussetzungen, die wir an die Variablenbausteine gesetzt haben erfüllt, denn es gilt:

Lemma 20.

(i) Um den ganzen Variablenbaustein zu rekonstruieren, sind genau 3 Messungen innerhalb des Variablenbausteins notwendig und hinreichend. (unabhängig davon wieviele Messungen ausserhalb stattfinden)

(ii) Die beiden Literalknoten x_i und $\bar{x}_i$ und ein beliebiger dritter Messknoten im VB_i reichen nicht, um den ganzen Variablenbaustein zu entdecken.

(iii) Für jeden der beiden Literalknoten x_i und $\bar{x}_i$ eines Variablenbausteins VB_i existieren im VB_i zwei weitere Messpunkte, so dass diese drei Punkte den ganzen Variablenbaustein VB_i entdecken.

(iv) Die Messpunktmengen, die einen Variablenbaustein VB_i entdecken, entdecken auch alle Nichtkanten zwischen VB_i und den anderen Variablenbausteinen.

Beweis.

(i) Da die neuen Hilfspunkte e_i, f_i eines Variablenbausteins VB_i von den alten Hilfspunkten a_i, b_i, c_i, d_i den Abstand 2 haben, ändert sich nichts an der Tatsache, dass der Kreis $a_i b_i c_i d_i$ nur durch mindestens 2 Messungen innerhalb des 6-Ecks entdeckt werden kann. Weiter haben e_i und f_i von allen anderen Punkten von G den gleichen Abstand, ihre Kante kann also nur durch einen dieser beiden Punkte entdeckt werden. Es sind also 3 Messungen notwendig um VB_i zu entdecken. Es ist leicht nachzuprüfen, dass die Messungen an x_i, c_i und e_i oder $\bar{x}_i$, b_i und e_i den ganzen Variablenbaustein VB_i entdecken. Dies beweist auch (iii)

(ii) Die beiden Literalknoten x_i und $\bar{x}_i$ entdecken die Kanten (a_i, d_i) und (e_i, f_i) nicht. Es gibt keinen Knoten, der beide diese Kanten entdeckt.

(iii) Wurde in (i) gezeigt

(iv) Man kann nachprüfen, dass gilt: Entdecken 3 VB_i-Punkte den ganzen VB_i, so gibt es keinen Punkt im VB_i, der von allen 3 dieser Punkte den Abstand 2 hat. Da 2 Punkte aus verschiedenen Variablenbausteinen immer den Abstand 2 haben, folgt die Behauptung.

2.2.3 Konstruktion des Graphen und NP-Vollst. von LG-ALL

Abbildung 6 zeigt die endgültige Form des Graphen G_ϕ (weiter G), der zu einer boolschen Formel ϕ (mit n Variablen $x_1, \ldots, x_n$ und m Klauseln $C_1, \ldots, C_m$) aus den oben beschriebenen Bausteinen durch das Verbinden jedes Klauselknotens mit seinen drei Literalknoten konstruiert wird.

Lemma 21.

(i) Zur Entdeckung des Graphen G sind mindestens $3n + m$ Punkte notwendig und zwar 3 Messpunkte (davon höchstens ein Literalknoten) in jedem Variablenbaustein und ein Hilfspunkt (o.B.d.A. C'_j) in jedem Klauselbaustein.

(ii) Sei S eine Menge von $3n + m$ Messpunkten, die alle Variablenbausteine und alle KB-Hilfskanten (C'_j, C''_j), $j = 1, \ldots, m$ entdeckt. Dann wird G genau dann vollständig rekonstruiert durch S, wenn jeder Klauselknoten mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden ist.

Beweis.

(i) Folgt direkt aus Lemma 17 und Lemma 20

(ii) Sei S nach Voraussetzung. Dann enthält S nach (i) 3 Messpunkte von jedem Variablenbaustein und einen Hilfsknoten (o.B.d.A. C'_j) von jedem Klauselbaustein. Dies macht bereits $3n + m$ Messpunkte, es gibt also keine weiteren Punkte in S.

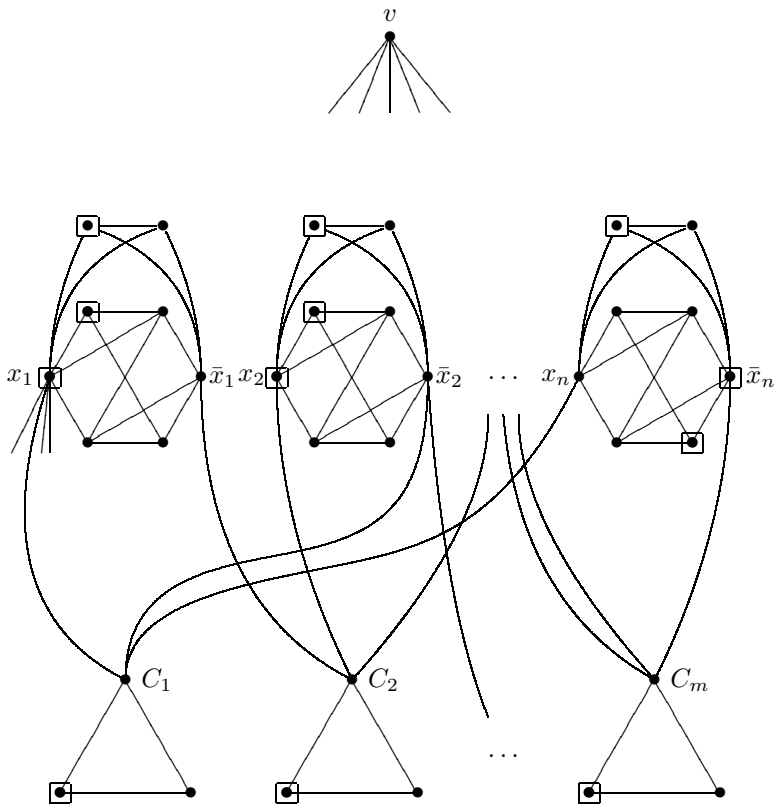

Abbildung 6: Graph zur Reduktion von 3-SAT auf LG-ALL.

$\Rightarrow$: Sei G vollständig rekonstruierbar durch S. Für jedes KB_j gilt: Die Kante (C_j, C_j'') kann nur durch einen ihrer Endpunkte oder einen Literalknoten mit Abstand 1 von C_j entdeckt werden. Da S weder C_j noch C_j'' enthält, wird die Kante (C_j, C_j'') durch einen mit C_j verbundenem Literalknoten entdeckt.

$\Leftarrow$: Sei jeder Klauselknoten mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden. Dann werden alle Kanten (C_j, C_j''), $j = 1, .., m$ und somit alle Klauselbausteine KB_j, $j = 1, \ldots, m$ durch S entdeckt und mit Lemma 17 auch alle von den KB-Punkten ausgehenden Kanten und Nichtkanten. Nach Voraussetzung werden alle Variablenbausteine entdeckt und nach Lemma 20 (iv) auch alle Nichtkanten zwischen den Variablenbausteinen. Die Kanten von den VB-Punkten zu v werden z.B. durch die Messung an C_1' entdeckt. Somit wird der ganze Graph G durch S entdeckt.

Satz 22. *Sei ϕ eine boolsche Formel und $G_\phi = (V, E)$ der zu ihr konstruierte Graph. ϕ ist genau dann erfüllbar, wenn eine Menge $S \subset V$ von $3n + m$*

Messpunkten existiert, die den ganzen Graphen G_ϕ entdeckt.

Beweis.

$\Rightarrow$: Sei eine ϕ erfüllende Wahrheitsbelegung gegeben. Wir definieren die Menge der gemessenen Literalknoten $S' \subset V$ als die Menge der wahren Literale. Diese erweitern wir gemäss Lemma 20 (iii) zu einer $3n$-punktigen Menge, die alle Variablenbausteine entdeckt. Dann fügen wir noch gemäss Lemma 17 die m Klauselhilfspunkte $C'_1, \ldots, C'_m$ hinzu, die alle Klauselhilfskanten $(C'_1, C''_1), \ldots, (C'_m, C''_m)$ entdecken. Nach Lemma 21 (ii) entdeckt die so erhaltene Menge S den ganzen Graphen G, falls alle Klauselknoten mit mindestens einem gemessenen Literalknoten verbunden sind. Da jede Klausel nach Voraussetzung mindestens ein wahres Literal enthält, ist dies nach Konstruktion von G und Definition von S erfüllt.

$\Leftarrow$: Sei $S \subset V$ eine $3n + m$ punktige Messpunktmenge, die G vollständig rekonstruiert. Nach Lemma 21 (ii) ist jeder Klauselknoten mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden. Definieren wir nun eine Wahrheitsbelegung, indem wir alle gemessenen Literale wahr setzen, wird nach Konstruktion von G jede Klausel von ϕ mindestens ein wahres Literal enthalten.[8]

Satz 23. *Das LG-ALL Entscheidungsproblem ist NP-vollständig.*

Beweis. Gegeben ein Graph $G = (V, E)$ und eine Menge von Messpunkten $S \subseteq V$ kann man in polynomieller Zeit (in $|V|$) nachprüfen ob S G entdeckt. Das Entscheidungsproblem LG-ALL ist somit in NP.

Nach Satz 22 kann man jedes 3-SAT Problem auf ein LG-ALL Problem reduzieren. Da der dafür zu konstruierende Graph G $8n + 3m + 1$ Punkte und $((8n + 3m + 1)^2 - (8n + 3m + 1))/2$ Kanten und Nichtkanten enthält kann diese Reduktion in polynomieller Zeit (in n) ausgeführt werden. LG-ALL ist also auch NP-schwer.

[8] Wird in einem VB_i keiner seiner Literalknoten x_i, $\bar{x}_i$ gemessen, so kann x_i beliebig wahr oder falsch gesetzt werden.

2.3 Nicht-Approximierbarkeit von LG-ALL-E und LG-ALL

Durch Reduktion auf das Set Cover Optimierungsproblem (SC) wurde in [2] und [1] die $O(\log n)$-Approximierbarkeit der Offline-Variante von LG-ALL-E und LG-ALL gezeigt. Hier zeigen wir, dass dieses Resultat nicht wesentlich verbessert werden kann, denn es gibt keine $o(\log n)$-Approximation von LG-ALL-E und LG-ALL, sofern P $\neq$ NP.

Zuerst präsentieren wir zwei kleinere Ergebnisse für den Online-Fall:

Behauptung 24. *Es gibt keine 2-Approximation für die Online-Varianten sowohl von LG-ALL-E als auch von LG-ALL.*

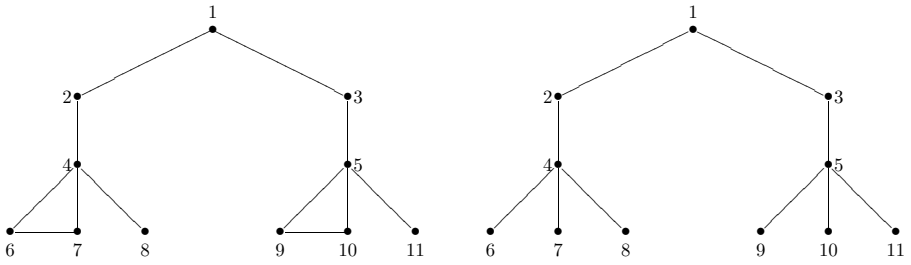

Abbildung 7:

Beweis. Betrachte den Graphen G in Abb. 7 links. Die Kanten (6,7) und (9,10) können nur durch einen ihrer Endknoten entdeckt werden. Kein Knoten von G entdeckt daher alle Kanten. Die Knoten 6 und 9 entdecken zusammen alle Kanten und Nichtkanten von G. Somit enthält die optimale Lösung von sowohl LG-ALL-E als auch von LG-ALL genau zwei Knoten.

Für einen Algorithmus sind zu Beginn alle Knoten äquivalent, sei also o.B.d.A 1 der erste Messknoten.

Abb. 7 rechts zeigt LG(1). Da die Punkte 6,7 und 8 für den Algorithmus äquivalent sind, könnte eine zweite Messung an 8 stattfinden und die dritte mit demselben Argument bei 11. Um alle Kanten von G zu entdecken sind dann immer noch mindestens zwei weitere Messungen notwendig, also braucht der Algorithmus in diesem Fall mindestens $2 \cdot \text{Opt} + 1$ Messungen.

Behauptung 25. *Es gibt keinen Approximationsalgorithmus für die Online-Varianten von LG-ALL-E und LG-ALL, der in jedem Schritt mindestens die Hälfte oder einen anderen beliebigen festen Anteil der Kanten (Kanten und Nichtkanten) entdeckt, die die beste Messung in diesem Schritt entdecken würde.*

Beweis. Betrachte den Graphen G in Abb. 8 links, in dem der Knoten $n+1$ mit allen Knoten $n/2, \ldots, n$ (n sei o.B.d.A. gerade) durch eine Kante verbunden ist. Nach der ersten Messung bei $n+2$ sieht LG($n+2$) wie in Abb. 8 rechts aus. Die Knoten $1, \ldots, n+1$ sind äquivalent. Eine Messung bei einem Knoten $v \in V$ entdeckt nun zusätzlich zu allen von v ausgehenden Kanten und Nichtkanten auch

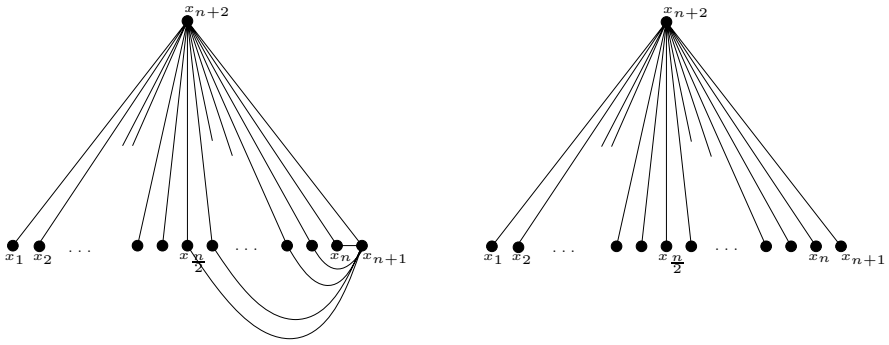

Abbildung 8:

alle Kanten und Nichkanten zwischen den Nachbarn von v und allen anderen Knoten. Der Knoten $n+1$ hat ausser dem Knoten $n+2$ weitere $n/2$ Nachbarn. Er entdeckt also $n/2$ neue Kanten und $\frac{n}{2} + \frac{n}{2} \cdot \frac{n}{2}$ neue Nichtkanten. Eine Messung bei einem Knoten mit dem einzigen Nachbar $n+2$ entdeckt hingegen nur n neue Nichtkanten und keine neuen Kanten. Da diese beiden Knoten für einen Online-Algorithmus ununterscheidbar sind, folgt die Behauptung.

2.3.1 Die $o(\log n)$-Nichtapproximierbarkeit von LG-ALL-E

Nun zeigen wir, wie versprochen, dass unter der Voraussetzung P $\neq$ NP kein $o(\log n)$-Approximationsalgorithmus für die Optimierungsprobleme LG-ALL-E und LG-ALL existiert. Dazu benützen wir die in [3] bewiesene Nichtexistenz einer $o(\log n)$-Approximation für das Test Collection Optimierungsproblem (TCP).

Definition 26. Das *Test Collection Optimierungsproblem* $(S, \mathcal{C})$ ist wie folgt definiert: Sei S eine gegebene Grundmenge und $\mathcal{C} \subseteq \mathcal{P}(S)$ eine gegebene Menge von Teilmengen von S. Finde eine minimale Teilmenge $\mathcal{C}' \subseteq \mathcal{C}$, so dass gilt: Für jedes Paar $x, y \in S$ existiert ein $C \in \mathcal{C}'$, das entweder x oder y enthält. Mit anderen Worten: Für jedes Paar von Punkten $x, y \in S$ gibt es eine Menge in $\mathcal{C}'$, die x und y unterscheidet (trennt).

Bemerkung 27. Wir können die Elemente von S als Krankheiten und die Elemente von $\mathcal{C}$ als Tests interpretieren, wobei: Ein Test $C \in \mathcal{C}$ reagiert positiv auf eine Krankheit $x \in S \Leftrightarrow x \in C$. Wir suchen eine minimale Menge von Tests, die jede Krankheit in S eindeutig identifizieren kann[9].

Satz 28. *Sei n die Anzahl Elemente von S. Das TCP $(S, \mathcal{C})$ ist nicht $o(\log n)$-approximierbar (sofern P $\neq$ NP).*

Beweis. Siehe [3].

[9] Es kann eine Krankheit geben, die dadurch identifiziert wird, dass alle Tests negativ ausfallen.

Bemerkung 29. Der obige Satz wird in [3] durch eine Reduktion von Set Cover (SC) auf TCP bewiesen. Zu einer Instanz $(S_{\mathrm{SC}}, \mathcal{C}_{\mathrm{SC}})$ von Set Cover mit n_{SC} Knoten und m_{SC} Sets wird in polynomieller Zeit eine Instanz $(S_{\mathrm{TCP}}, \mathcal{C}_{\mathrm{TCP}})$ von TCP konstruiert, mit $n_{\mathrm{TCP}} := 2 \cdot n_{\mathrm{SC}} \cdot \lceil \log_2 n_{\mathrm{SC}} \rceil$ Krankheiten und $m_{\mathrm{TCP}} := m_{\mathrm{SC}} \cdot \lceil \log_2 n_{\mathrm{SC}} \rceil + \lceil \log_2(n_{\mathrm{SC}} \cdot \lceil \log_2 n_{\mathrm{SC}} \rceil) \rceil$ Tests, für die gilt:

(i) $\mathrm{Opt}_{\mathrm{TCP}} = \lceil \log_2 n_{\mathrm{SC}} \rceil \cdot \mathrm{Opt}_{\mathrm{SC}} + \lceil \log_2(n_{\mathrm{SC}} \cdot \lceil \log_2 n_{\mathrm{SC}} \rceil) \rceil$

(ii) Aus jeder Lösung von TCP kann eine Lösung von SC konstruiert werden, so dass

$$|\mathrm{Lösung(SC)}| \leq \frac{|\mathrm{Lösung(TCP)}| - \lceil \log_2(n_{\mathrm{SC}} \cdot \lceil \log_2 n_{\mathrm{SC}} \rceil) \rceil}{\log_2 n_{\mathrm{SC}}}. \tag{1}$$

Ein $o(\log n)$-Approximationsalgorithmus $\mathcal{A}_{\mathrm{TCP}}$ für TCP könnte nun dazu benützt werden, SC mit Approximationsquotient $o(\log n)$ zu approximieren, denn aus der oberen Schranke für den Output $A_{\mathrm{TCP}} := |\mathcal{A}_{\mathrm{TCP}}((S_{\mathrm{TCP}}, \mathcal{C}_{\mathrm{TCP}}))|$

$$A_{\mathrm{TCP}} \leq o(\log n_{\mathrm{TCP}}) \cdot \mathrm{Opt}_{\mathrm{TCP}} \tag{2}$$

kann eine obere Schranke für den Output A_{SC} berechnet werden:

$$A_{\mathrm{SC}} \leq o(\log n_{\mathrm{SC}}) \cdot \mathrm{Opt}_{\mathrm{SC}} \tag{3}$$

Da SC nach [4] nicht $o(\log n)$-approximierbar ist, würde daraus P = NP folgen.

Die folgende Bemerkung werden wir später brauchen.

Bemerkung 30. In [4] wird eine Klasse von SC-Problemen mit n Knoten und weniger als n^s (für ein geeignetes $s \in \mathbb{N}$) Sets konstruiert, die nicht $o(\log n)$-approximierbar ist.

Satz 31. *Das Test Collection Problem ist auf das LG-ALL-E Problem polynomiell reduzierbar.*

Beweis. Folgt später.

Da TCP NP-vollständig ist, beweist die Aussage von Satz 31 zwar erneut die NP-Vollständigkeit von LG-ALL-E, liefert aber a priori keine Garantie, dass aus einem Approximationsalgorithmus für LG-ALL-E ein brauchbarer Approximationsalgorithmus für TCP konstruiert werden kann. (Nicht jede Reduktion ist approximationserhaltend.) Hier benötigen wir die folgende Version mit einfacherem Beweis:

Satz 32. *Zu einer Instanz $(S, \mathcal{C})$ von TCP mit n_{TCP} Krankheiten und m_{TCP} Tests kann in polynomieller Zeit eine Instanz $G = (V, E)$ von LG-ALL-E mit $n_{\mathrm{TCP}} + m_{\mathrm{TCP}}$ Knoten konstruiert werden, so dass gilt:*

(i) Aus jeder gültigen Lösung $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ von TCP kann (in polynomieller Zeit) eine gültige Lösung $L_{\mathrm{G}} \subset V$ von LG-ALL-E berechnet werden wobei $|L_{\mathrm{G}}| \leq |L_{(S,\mathcal{C})}| + 1$

(ii) Aus jeder gültigen Lösung $L_G \subseteq V$ von LG-ALL-E kann (in polynomieller Zeit) eine Lösung $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ von TCP berechnet werden wobei $|L_{(S,\mathcal{C})}| \leq |L_G|$.

(iii) Es gilt $\mathrm{Opt}_{LG} - 1 \leq \mathrm{Opt}_{TCP} \leq \mathrm{Opt}_{LG} \leq \mathrm{Opt}_{TCP} + 1$

Beweis. Sei ein Test Collection Problem mit Grundmenge S und Testmenge $\mathcal{C}$ gegeben. Wir konstruieren den Graphen $G = (V, E)$ wie folgt: Jede Krankheit in S und jeder Test in $\mathcal{C}$ bildet einen Knoten von G, das heisst $V = S \cup \mathcal{C}$. Die Krankheitsknoten bilden eine Clique, d.h. sie sind alle durch Kanten miteinander verbunden. Zusätzlich ist noch jeder Test mit seinen Krankheiten verbunden.

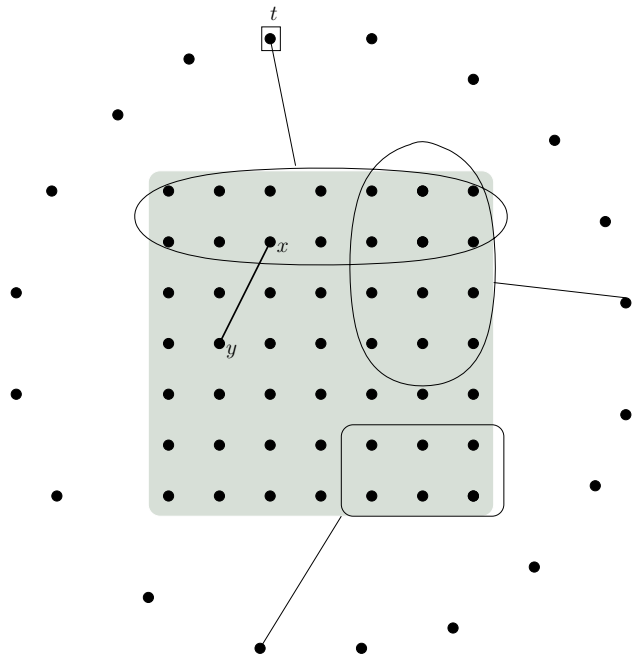

Abbildung 9: Zu einer Instanz von TCP konstruierter Graph. Die im Zentrum liegenden Krankheitsknoten bilden eine Clique, was der Übersichtlichkeit halber nur durch den grauen Hintergrund symbolisiert wird. Im Kreis um die Krankheitsclique sind die Testknoten angeordnet. Die "Lassos" deuten an, mit welchen Krankheitsknoten ein Testknoten durch eine Kante verbunden ist. Die explizit eingezeichnete Kante (x, y) der Krankheitsclique wird z.B. durch Messung des Testknotens t entdeckt.

Behauptung 33. *Eine Menge $\mathcal{C}' \subseteq \mathcal{C}$ von Tests entdeckt genau dann alle Kanten zwischen den Krankheiten, wenn sie eine gültige Lösung des TCP $(S, \mathcal{C})$ bildet, das heisst wenn die Tests in $\mathcal{C}'$ alle Krankheiten aus S unterscheiden können.*

Beweis. Eine Kante zwischen zwei Krankheiten x und y wird genau dann von einer Testknotenmenge $\mathcal{C}' \subseteq \mathcal{C}$ entdeckt, wenn es in $\mathcal{C}'$ einen Testknoten t gibt, der von den Knoten x und y unterschiedliche Abstände hat, das heisst in diesem Fall von einem der Knoten x, y Abstand 1 und von dem anderen Abstand 2. Das ist nach Konstruktion genau dann der Fall, wenn der Test t im Test Collection Problem $(S, \mathcal{C})$ die beiden Krankheiten trennt.

(i) Sei eine gültige Lösung $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ von TCP $(S, \mathcal{C})$ gegeben. Nach Behauptung 33 entdecken die Tests in $L_{(S,\mathcal{C})}$ alle Kanten zwischen den Krankheitsknoten. Da $L_{(S,\mathcal{C})}$ alle Krankheiten unterscheidet, gibt es höchtstens eine Krankheit, bei der alle Tests negativ sind. Nach Konstruktion gibt es daher in G höchstens einen Krankheitsknoten $k \in V$, der von allen gemessenen Testknoten Abstand 2 hat. Da alle Testknoten voneinander mindestens Abstand 2 haben, werden alle Kanten von den Testknoten zu den Krankheitsknoten (ausser zu k) entdeckt. Messen wir zusätzlich noch den Knoten k, so werden alle Kanten von G entdeckt.

(ii) Gegeben sei eine Menge $L_{\mathrm{G}} \subseteq V$, die alle Kanten von G entdeckt. Besteht L_{G} nur aus den Testknoten, so bilden diese Tests nach Behauptung 33 die Lösung von TCP. Sonst gilt:

Behauptung 34. *Sei $L_{\mathrm{G}} = L_{\mathrm{S}} \cup L_{\mathcal{C}}$ wobei $L_{\mathrm{S}} \subseteq S$ und $L_{\mathcal{C}} \subseteq \mathcal{C}$. Die Menge der Krankheitsmessknoten L_{S} kann durch eine Menge von Tests $L'_{\mathcal{C}} \subseteq \mathcal{C}$ ersetzt werden, wobei $|L'_{\mathcal{C}}| \leq |L_{\mathrm{S}}|$ ist und $L_{\mathcal{C}} \cup L'_{\mathcal{C}} =: L_{(S,\mathcal{C})}$ eine gültige Lösung von TCP bildet.*

Beweis. Werden alle Testknoten in $L_{\mathcal{C}}$ gemessen und alle entdeckten Kanten entfernt, so zerfällt die Krankheitsclique in eine Menge von isolierten kleineren Cliquen. Denn bleibt eine Kante (x, y) unentdeckt, so hat y von jedem Messknoten den gleichen Abstand wie x. Bleibt auch die Kante (x, z) unentdeckt, so muss auch noch (y, z) unentdeckt bleiben.

Um eine Clique zu entdecken müssen alle ihren Knoten bis auf einen gemessen werden. Seien nun $K_1, \ldots, K_p$ die unentdeckten Cliquen und $k_1, \ldots, k_p$ ihre Grössen. Dann gilt $|L_{\mathrm{S}}| \geq (k_1 - 1) + \ldots + (k_p - 1)$. Betrachte nun eine Clique K_i. Für jedes Paar von Krankheiten x, y in K_i gibt es einen (noch nicht gemessenen) Test, der diese beiden Krankheiten trennt. Da jeder Test, der mindestens ein Paar von Krankheiten in K_i trennt, die ganze Clique K_i in zwei Teile teilt (und alle Krankheiten im einen Teil von allen Krankheiten im anderen Teil trennt), und die Clique K_i in höchstens k_i Teile geteilt werden kann, sind höchstens $k_i - 1$ solche Tests nötig, um alle Krankheiten in K_i voneinander zu trennen. Wir nennen diese Menge von Tests T_i. Es ist nun klar, dass die Menge $L'_{\mathcal{C}} = T_1 \cup \ldots \cup T_p$ zusammen mit $L_{\mathcal{C}}$ alle Krankheiten des TCP $(S, \mathcal{C})$ trennt.

(iii) Folgt aus (i) und (ii).

Satz 35. *Es existiert keine $o(\log n)$-Approximation des LG-ALL-E Optimierungsproblems (sofern $P \neq NP$).*

Beweis. Wir werden zeigen: Aus einem $o(\log n)$-Approximationsalgorithmus $\mathcal{A}_{\mathrm{LG}}$ von LG-ALL-E kann ein $o(\log n)$-Approximationsalgorithmus $\mathcal{A}_{\mathrm{TCP}}$ für eine Klasse von TC-Problemen mit $m \leq n^s$ konstruiert werden. (Dabei ist n die Anzahl Krankheiten, m die Anzahl Tests und $s \in \mathbb{N}$ fest aber beliebig.)

Da nach Bemerkung 29 und 30 eine $o(\log n)$-nichtapproximierbare Klasse von TCP Instanzen existiert, die die Bedingung $m \leq n^s$ für ein geignetes s erfüllt, folgt daraus die $o(\log n)$-Nichtapproximierbarkeit von LG-ALL-E.

Sei $\mathcal{A}_{\mathrm{LG}}$ ein $o(\log n)$-Approximationsalgorithmus für LG-ALL-E. Für die Grösse des Outputs $A_{\mathrm{LG}}(G) := |\mathcal{A}_{\mathrm{LG}}(G)|$ gilt also die obere Schranke

$$A_{\mathrm{LG}}(G) \leq o(\log n_{\mathrm{LG}}) \cdot \mathrm{Opt}_{\mathrm{LG}}(G). \tag{4}$$

Wir definieren den Algorithmus $\mathcal{A}_{\mathrm{TCP}}$ zur Approximation des TCP $(S, \mathcal{C})$ mit n_{TCP} Krankheiten und m_{TCP} Tests wie folgt:

1. Zu $(S, \mathcal{C})$ konstruiere gemäss Satz 32 den Graphen G mit $n_{\mathrm{LG}} = n_{\mathrm{TCP}} + m_{\mathrm{TCP}}$ Knoten.
2. Wende $\mathcal{A}_{\mathrm{LG}}$ zur Approximation des LG-ALL-E Optimierungsproblems G an. $\mathcal{A}_{\mathrm{LG}}$ liefert eine Lösung $\mathcal{A}_{\mathrm{LG}}(G) =: L_{\mathrm{G}} \subseteq V$ mit $A_{\mathrm{LG}}(G) \leq o(\log n_{\mathrm{LG}}) \cdot \mathrm{Opt}_{\mathrm{LG}}(G)$.
3. Aus der Lösung L_{G} berechne gemäss Satz 32 (ii) eine gültige Lösung $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ des Test Collection Problems $(S, \mathcal{C})$, mit $|L_{(S,\mathcal{C})}| \leq |L_{\mathrm{G}}|$.

Mit Satz 32 (iii) folgt für $A_{\mathrm{TCP}}((S, \mathcal{C})) = |L_{(S,\mathcal{C})}|$ die obere Schranke

$$\begin{aligned} &A_{\mathrm{TCP}}((S, \mathcal{C})) \leq A_{\mathrm{LG}}(G) \leq o(\log n_{\mathrm{LG}}) \cdot \mathrm{Opt}_{\mathrm{LG}}(G) \\ &\leq o(\log(n_{\mathrm{TCP}} + n_{\mathrm{TCP}}^s))[\mathrm{Opt}_{\mathrm{TCP}}((S, \mathcal{C})) + 1] = o(\log n_{\mathrm{TCP}}) \cdot \mathrm{Opt}_{\mathrm{TCP}}((S, \mathcal{C})). \end{aligned} \tag{5}$$

$\mathcal{A}_{\mathrm{TCP}}$ ist also ein $o(\log n)$-Approximationsalgorithmus für TCP.

2.3.2 Die $o(\log n)$-Nichtapproximierbarkeit von LG-ALL

Die $o(\log n)$-Nichtapproximierbarkeit von LG-ALL zeigen wir ebenfalls durch eine Reduktion von TCP. Zu jeder Instanz $(S, \mathcal{C})$ von TCP mit $m_{\mathrm{TCP}} \leq n_{\mathrm{TCP}}^s$ (wobei n_{TCP} die Anzahl Krankheiten und m_{TCP} die Anzahl Tests bezeichne und $s \in \mathbb{N}$ eine beliebige Konstante sei), konstruieren wir (in polynomieller Zeit) eine Instanz $G = (V, E)$ des LG-ALL Optimierungsproblems, so dass gilt: Eine $o(\log n)$-Approximation von LG-ALL angewandt auf G könnte dazu benützt werden, das Test Collection Problem $(S, \mathcal{C})$ (in polynomieller Zeit) mit Approximationsquotient $o(\log n)$ zu approximieren.

Der zu konstruierende Graph G besteht wieder aus einer Clique von n_{TCP} Krankheitsknoten und aus m_{TCP} Testknoten, die mit ihren Krankheitsknoten verbunden sind. Neu kommen noch $2(\lceil \log_2 m_{\text{TCP}} \rceil + 2)$ zu $\lceil \log_2 m_{\text{TCP}} \rceil + 2$ Paaren verbundene Hilfsknoten dazu. Diese Paare werden so mit den Testknoten verbunden, dass gilt:

1. Beide Punkte eines Paares sind mit den gleichen Testknoten verbunden.
2. Die ersten $\lceil \log_2 m_{\text{TCP}} \rceil$ Paare sind so mit den Testknoten verbunden, dass jeder Test eine eindeutige (will heissen ihn eindeutig identifizierende) Kombination von Hilfspunktnachbarn hat. Es gibt also keine zwei Testknoten, die mit der gleichen Kombination von Hilfsknoten verbunden sind.[10]
3. Die letzten zwei Paare von Hilfsknoten werden mit allen Testknoten verbunden.

Weiter gibt es noch einen zusätzlichen Hilfsknoten v, der mit allen Knoten von G verbunden ist.

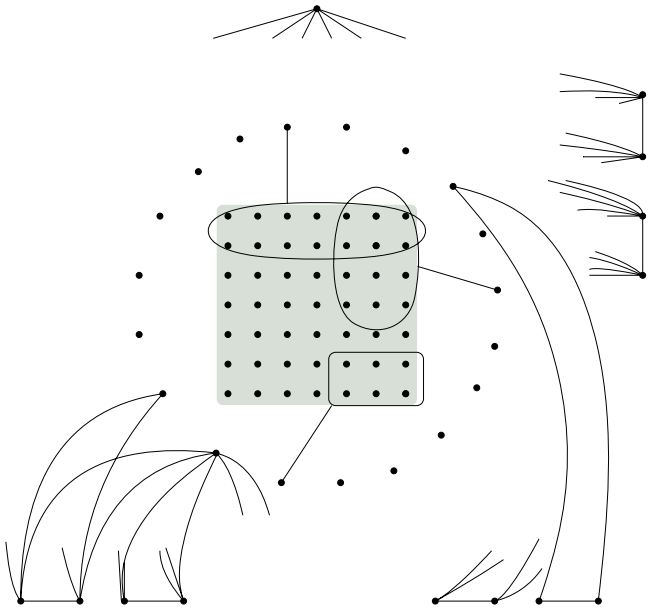

Abbildung 10: Graph zur Reduktion von TCP auf LG-ALL. Krankheitsclique und Testknoten sind gleich wie im LG-ALL-E Fall angeordnet. Zusätzlich sind unten die Hilfsknotenpaare zur Adressierung der Testknoten, rechts oben zwei mit allen Testknoten verbundene Hilfsknotenpaare und oben der mit allen Knoten verbundene Hilfknoten v eingezeichnet.

[10] Dazu sind klarerweise $\lceil \log_2 m_{\text{TCP}} \rceil$ Hilfsknotenpaare notwendig und hinreichend.

Bemerkung 36. Aus 1 folgt, dass aus jedem Paar von Hilfspunkten ein Knoten gemessen werden muss, um alle Kanten zwischen den Hilfspunkten zu entdecken[11]. Dies ist auch hinreichend, um alle Kanten und Nichtkanten zwischen den Hilfspunkten (inklusive v), so wie alle Kanten und Nichtkanten von den Hilfspunkten (inklusive v) zu den Krankheitsknoten zu entdecken.

Bemerkung 37. Aus 2 folgt, dass je eine Messung in jedem Hilfsknotenpaar alle Nichtkanten zwischen den Testknoten entdeckt, denn für zwei beliebige Testknoten gibt es immer einen gemessenen Hilfsknoten, der genau zu einem von ihnen benachbart ist.

Bemerkung 38. Aus 3 folgt, dass eine Messung im letzten Hilfsknotenpaar alle Kanten und Nichtkanten von den Testknoten zu den Krankheitsknoten, sowie alle Kanten und Nichtkanten von den Testknoten zu den restlichen Hilfsknotenpaaren entdeckt. Alle Kanten von dem nichtgemessenen Hilfspunkt des letzten Paares zu den Testknoten werden durch eine Messung im vorletzten Hilfspunktpaar entdeckt.

Lemma 39.

(i) Eine Messung in jedem Hilfspunktpaar ist notwendig und hinreichend, um alle Kanten und Nichtkanten von G, ausser den Kanten zwischen den Krankheitsknoten, zu entdecken. Letztere bleiben bei diesen Messungen alle unentdeckt.

(ii) Die Kanten zwischen den Krankheitsknoten können nur durch Messungen an den Testknoten oder an den Krankheitsknoten entdeckt werden.

(iii) Eine Menge $\mathcal{C}'$ von Tests entdeckt genau dann alle Kanten zwischen den Krankheiten, wenn sie eine gültige Lösung des zugehörigen TCP bildet.

Beweis.

(i) Folgt aus den Bemerkungen 36, 37 und 38.

(ii) Klar.

(iii) Die Aussage ist dieselbe wie in Behauptung 33. Der Beweis bleibt gültig, denn die Beziehungen zwischen den Krankheitsknoten und den Testknoten sind die gleichen.

Satz 40. *Das Test Collection Problem ist in polynomieller Zeit auf das LG-ALL-Problem reduzierbar. Konkret gilt: Sei $(S, \mathcal{C})$ eine Instanz von TCP und sei G der zu $(S, \mathcal{C})$ konstruierte Graph. Dann gilt:*

(i) Aus jeder gültigen Lösung $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ von TCP kann (in polynomieller Zeit) eine gültige Lösung $L_G \subset V$ von LG-ALL berechnet werden, wobei $|L_G| = |L_{(S,\mathcal{C})}| + \lceil \log_2 m_{TCP} \rceil + 2$.

[11] Mit Kanten sind hier nur die wirklich vorhandenen Kanten gemeint, konkret sind dies hier die Kanten, die die Hilfpunkte zu Paaren verbinden.

(ii) Aus jeder gültigen Lösung $L_G \subseteq V$ von LG-ALL kann (in polynomieller Zeit) eine gültige Lösung $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ von TCP berechnet werden, wobei $|L_{(S,\mathcal{C})}| \leq |L_G| - \lceil \log_2 m_{TCP} \rceil - 2$.

(iii) $\text{Opt}_{LG} = \text{Opt}_{TCP} + \lceil \log_2 m_{TCP} \rceil + 2$.

Beweis.

(i) Sei $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$ eine gültige Lösung von TCP. $L_{(S,\mathcal{C})}$ entdeckt nach Lemma 39 (iii) alle Kanten zwischen den Krankheitsknoten. Erweitern wir $L_{(S,\mathcal{C})}$ gemäss Lemma 39 (i) um $\lceil \log_2 m_{TCP} \rceil + 2$ Hilfspunkte, erhalten wir eine Messknotenmenge, die alle Kanten und Nichtkanten von G entdeckt.

(ii) Sei $L_G \subseteq V$ eine gültige Lösung von LG-ALL. Nach Lemma 39 (i) enthält sie $\lceil \log_2 m_{TCP} \rceil + 2$ Hilfspunkte, die alle Kanten und Nichtkanten von G ausser den Kanten innerhalb der Krankheitsclique entdecken, und die Krankheitsclique gänzlich unentdeckt lassen.

Die Kanten innerhalb der Krankheitsclique werden nach Lemma 39 (ii) durch eine Teilmenge $L'_G \subset L_G$ entdeckt, die nur Testknoten und Krankheitsknoten enthält. Dabei gilt $|L'_G| \leq |L_G| - \lceil \log_2 m_{TCP} \rceil - 2$.

Auf die Menge L'_G können wir nun Behauptung 34 anwenden. Somit erhalten wir die Menge $L_{(S,\mathcal{C})} \subseteq \mathcal{C}$, die nur noch Tests enthält und eine gültige Lösung des TCP $(S, \mathcal{C})$ bildet, wobei $|L_{(S,\mathcal{C})}| \leq |L'_G| \leq |L_G| - \lceil \log_2 m_{TCP} \rceil - 2$.

(iii) Folgt aus (i) und (ii).

Bemerkung 41. Es ist leicht einzusehen, dass die obige Reduktion auch eine Reduktion von TCP auf LG-ALL-E ist. Dies liefert den noch ausstehenden Beweis von Satz 31.

Satz 40 liefert einen weiteren Beweis der NP-Vollständigkeit von LG-ALL. Wegen des additiven Terms $\lceil \log_2 m_{TCP} \rceil$ ist eine solche Reduktion im allgemeinen nicht approximationserhaltend. Da es aber für unsere Zwecke reicht, die $o(\log n)$-nichtapproximierbare Klasse von TC-Problemen mit $m_{TCP} \leq n^s_{TCP}$ (für ein geeignetes $s \in \mathbb{N}$) zu betrachten, rettet uns die folgende Bemerkung:

Bemerkung 42. Für eine Instanz von TCP mit n_{TCP} Krankheiten und m_{TCP} Tests gilt: $\text{Opt}(S, \mathcal{C}) \geq \lceil \log_2 n_{TCP} \rceil$.

Lemma 43. *Betrachte eine Klasse von TCP Instanzen mit $m_{TCP} \leq n^s_{TCP}$ und die dazu konstruierten LG-ALL Instanzen. Sei $G = (V, E)$ der zu einer TCP Instanz $(S, \mathcal{C})$ konstruierte Graph. Dann gilt:*

(i) $o(\log_2 n_{LG}) = o(\log_2 n_{TCP})$

(ii) $\text{Opt}_{LG}(G) \leq (s + 3) \cdot \text{Opt}_{TCP}((S, \mathcal{C}))$

Beweis. Nach Konstruktion von G gilt:

$$n_{\mathrm{LG}} = n_{\mathrm{TCP}} + m_{\mathrm{TCP}} + 2(\lceil \log_2 m_{\mathrm{TCP}} \rceil + 2) + 1 \leq 4n_{\mathrm{TCP}}^s \tag{6}$$

und

$$\begin{aligned} \mathrm{Opt}_{\mathrm{LG}}(G) &= \mathrm{Opt}_{\mathrm{TCP}}((S,\mathcal{C})) + \lceil \log_2 m_{\mathrm{TCP}} \rceil + 2 \\ &\leq \mathrm{Opt}_{\mathrm{TCP}}((S,\mathcal{C})) + s\lceil \log_2 n_{\mathrm{TCP}} \rceil + 2 \leq (s+3) \cdot \mathrm{Opt}_{\mathrm{TCP}}((S,\mathcal{C})) \end{aligned} \tag{7}$$

wobei der letzte Schritt aus der obigen Bemerkung folgt.

Satz 44. *Das LG-ALL Optimierungsproblem ist nicht $o(\log n)$-approximierbar (sofern $P \neq NP$).*

Beweis. Wir zeigen, dass aus einem $o(\log n)$-Approximationsalgorithmus $\mathcal{A}_{\mathrm{LG}}$ für LG-ALL ein $o(\log n)$-Approximationsalgorithmus $\mathcal{A}_{\mathrm{TCP}}$ für eine Klasse von TC-Problemen mit $m_{\mathrm{TCP}} \leq n_{\mathrm{TCP}}^s$ ($s \in \mathbb{N}$ beliebig) konstruiert werden kann, was P = NP implizieren würde.

$\mathcal{A}_{\mathrm{TCP}}$ arbeitet wie folgt: Zu der Instanz $(S,\mathcal{C})$ von TCP konstruiert er eine Instanz G von LG-ALL, berechnet $\mathcal{A}_{\mathrm{LG}}(G)$ und daraus eine gültige Lösung von TCP. Aus der oberen Schranke $A_{LG}(G) \leq o(\log n_{\mathrm{LG}}) \cdot \mathrm{Opt}_{\mathrm{LG}}(G)$ folgt wegen obigem Lemma die obere Schranke

$$\begin{aligned} A_{\mathrm{TCP}}((S,\mathcal{C})) &\leq A_{\mathrm{LG}}(G) - \lceil \log_2 m_{\mathrm{TCP}} \rceil - 2 \\ &\leq o(\log n_{\mathrm{LG}}) \cdot \mathrm{Opt}_{\mathrm{LG}}(G) - \lceil \log_2 m_{\mathrm{TCP}} \rceil - 2 \\ &\leq o(\log n_{\mathrm{TCP}}) \cdot (s+3) \cdot \mathrm{Opt}_{\mathrm{TCP}}((S,\mathcal{C})) \\ &= o(\log n_{\mathrm{TCP}}) \cdot \mathrm{Opt}_{\mathrm{TCP}}((S,\mathcal{C})). \end{aligned} \tag{8}$$

Da dies alles in polynomieller Zeit (in n_{TCP}) ausgeführt werden kann, folgt die Behauptung.

2.4 Die optimale Lösbarkeit von LG-ALL-E für einen Non-Branching Planar Chordal Ring

In diesem Kapitel stellen wir eine spezielle Klasse von ausserplanaren Graphen vor, für die das LG-ALL-E Offline-Optimierungsproblem mittels dynamischen Programmierens effizient lösbar ist. Wir geben einen offline Algorithmus an, der für diese Graphen eine optimale Lösung des LG-ALL-E Problems in polynomieller Zeit findet.

Seien G_1 und G_2 zwei Teilgraphen mit genau einer gemeinsamen Kante (x, y). Wir wollen untersuchen, wie sich Messungen in G_1 auf die Entdeckung von Kanten in G_2 auswirken.

Lemma 45. *Für eine Kante (a, b) in G_2 gilt:*

(i) Sei v_1 ein Punkt in G_1 mit $\mathrm{d}(v_1, x) < \mathrm{d}(v_1, y)$. Dann gilt: (a, b) wird durch v_1 entdeckt $\Leftrightarrow$ (a, b) wird durch x entdeckt.

(ii) Seien u_1 und v_1 zwei Punkte von G_1 mit $\mathrm{d}(u_1, x) = \mathrm{d}(u_1, y)$ und $\mathrm{d}(v_1, x) = \mathrm{d}(v_1, y)$. Dann gilt: (a, b) wird durch u_1 entdeckt $\Leftrightarrow$ (a, b) wird durch v_1 entdeckt.

Beweis.

(i) Da (x, y) eine Kante ist, folgt aus $\mathrm{d}(v_1, x) < \mathrm{d}(v_1, y)$ für y $\mathrm{d}(v_1, y) = \mathrm{d}(v_1, x) + 1$ und für jeden beliebigen Punkt v_2 in G_2 $\mathrm{d}(v_1, v_2) = \mathrm{d}(v_1, x) + \mathrm{d}(x, v_2)$. Nun wird (a, b) genau dann von x entdeckt, wenn $\mathrm{d}(x, a) \neq \mathrm{d}(x, b)$ gilt. Wegen $\mathrm{d}(v_1, a) = \mathrm{d}(v_1, x) + \mathrm{d}(x, a)$ und $\mathrm{d}(v_1, b) = \mathrm{d}(v_1, x) + \mathrm{d}(x, b)$ folgt: $\mathrm{d}(v_1, a) \neq \mathrm{d}(v_1, b) \Leftrightarrow \mathrm{d}(x, a) \neq \mathrm{d}(x, b)$. Daraus folgt die Behauptung.

(ii) Aus $\mathrm{d}(u_1, x) = \mathrm{d}(u_1, y)$ und $\mathrm{d}(v_1, x) = \mathrm{d}(v_1, y)$ folgt für einen beliebigen Knoten v_2 in G_2:

$$\begin{aligned}\mathrm{d}(u_1, v_2) &= \min\{\mathrm{d}(u_1, x) + \mathrm{d}(x, v_2), \mathrm{d}(u_1, y) + \mathrm{d}(y, v_2)\} \\ &= \mathrm{d}(u_1, x) + \min\{\mathrm{d}(x, v_2), \mathrm{d}(y, v_2)\}\end{aligned}$$

$$\mathrm{d}(v_1, v_2) = \mathrm{d}(v_1, x) + \min\{\mathrm{d}(x, v_2), \mathrm{d}(y, v_2)\}$$

Für (a, b) gilt also:

$$\begin{aligned}\mathrm{d}(u_1, a) \neq \mathrm{d}(u_1, b) \Leftrightarrow \min\{\mathrm{d}(x, a), \mathrm{d}(y, a)\} &\neq \min\{\mathrm{d}(x, b), \mathrm{d}(y, b)\} \\ &\Leftrightarrow \mathrm{d}(v_1, a) \neq \mathrm{d}(v_1, b)\end{aligned}$$

In Bezug auf die Entdeckung der Kanten von G_2 kann man also die Punkte von G_1 in drei Äquivalenzklassen unterteilen: Die Klasse der Punkte näher bei x bezeichnen wir mit $[x]^{(1)}$, die Klasse der Punkte näher bei y mit $[y]^{(1)}$ und die Klasse der Punkte mit gleichem Abstand zu beiden mit $[xy]^{(1)}$.[12] Werden

[12] Dasselbe gilt auch umgekehrt für die Punkte von G_2 bezüglich der Kantenentdeckung in G_1.

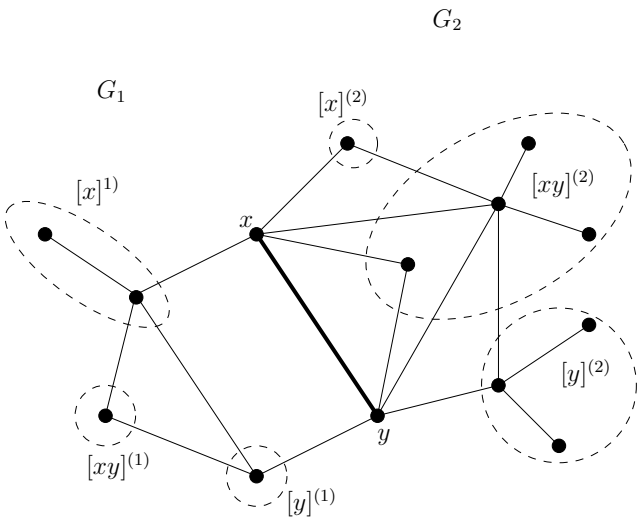

Abbildung 11: Zwei Teilgraphen G_1 und G_2, mit genau einer gemeinsamen Kante.

in G_1 einige Punkte gemessen, ist für die Kantenentdeckung in G_2 einzig die Information wichtig, welche der drei Äquivalenzklassen $[x]^{(1)}$, $[y]^{(1)}$, $[xy]^{(1)}$ in der Messpunktmenge vertreten sind.
Der charakteristische Vektor $I_{G_1G_2} = (\delta_{[x]^{(1)}}, \delta_{[y]^{(1)}}, \delta_{[xy]^{(1)}})$ enthält die ganze Information, die für die Bestimmung der durch die Messungen in G_1 entdeckten Kanten von G_2 erforderlich ist. ($\delta_{[x]^{(1)}} = 1$ falls die Klasse $[x]^{(1)}$ vertreten ist, 0 sonst, etc.) Analog ist $I_{G_2G_1} = (\delta_{[x]^{(2)}}, \delta_{[y]^{(2)}}, \delta_{[xy]^{(2)}})$.

Umgekehrt kann man zwei gegebene Vektoren $I_{G_1G_2}$ und $I_{G_2G_1}$ als eine Art Randbedingung betrachten, und nach einer minimalen Menge von Punkten in G_1 suchen, die $I_{G_1G_2}$ erfüllen und zusammen mit $I_{G_2G_1}$ alle Kanten von G_1 entdecken. Dabei interpretiert man $I_{G_1G_2}$ als den geforderten Einfluss von Messungen in G_1 auf die Kantenentdeckung in G_2 und $I_{G_2G_1}$ als den gegebenen Einfluss einer nicht weiter spezifizierten Messknotenmenge in G_2 auf die Kantenentdeckung in G_1.

Da die Punkte x und y eine spezielle Position haben, betrachten wir sie von jetzt an separat. Neu bezeichnet $[x]$ die Äquivalenzklasse von Punkten näher bei x (als bei y), jedoch ohne x selbst. Analog entält $[y]$ den Knoten y nicht mehr, $[xy]$ bleibt unverändert. Der Randbedingungsvektor hat neu die Form $R_{xy} = (\delta_x, \delta_y, I_{G_1G_2}, I_{G_2G_1}) \in \{0,1\}^8$.

Nun betrachten wir eine spezielle Klasse von ausserplanaren Graphen.

Definition 46. Ein Non-Branching Planar Chordal Ring NBPCR ist ein ausserplanarer Graph, der wie in Abbildung 12 gezeichnet werden kann, d.h. als zwei links und rechts zu einem Zyklus verbundene Ketten mit weiteren Kanten (Sehnen), die nur von einer Kette zur anderen führen dürfen. Anders beschrie-

ben besteht ein NBPCR aus einer zusammenhängenden Kette von Zyklen, in der jeder Zyklus (ausser dem ersten und dem letzten) genau einen Vorgänger und einen Nachfolger hat, und zwei benachbarte Zyklen immer genau eine gemeinsame Kante haben.

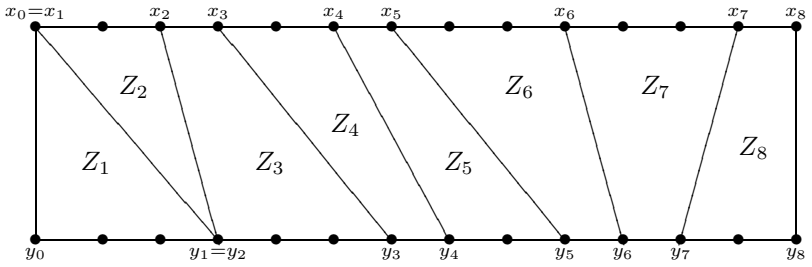

Abbildung 12: Ein NBPCR als zwei verbundene Ketten mit Sehnen.

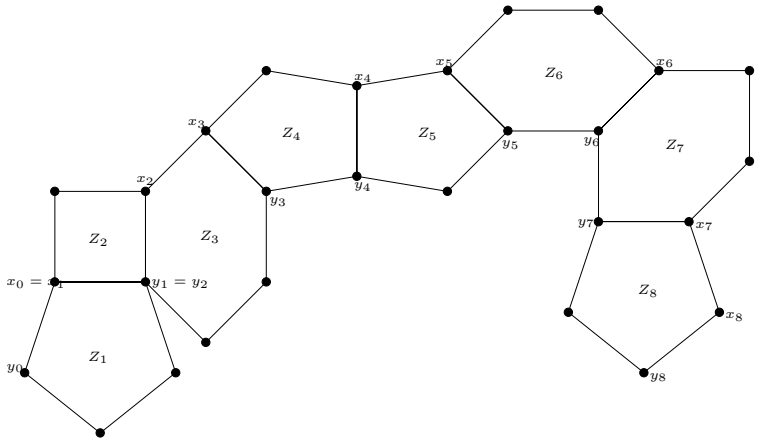

Abbildung 13: Der gleiche NBPCR als Folge von Zyklen.

Nun zeigen wir, dass das LG-ALL-E Offline-Optimierungsproblem für die Klasse der NBPCR Graphen effizient lösbar ist. Dazu konstruieren wir einen Algorithmus $\mathcal{A}$, der beim ersten Zyklus des NBPCR Graphen startet und sich von links nach rechts eine Sehne nach der anderen betrachtend zum letzten Zyklus durcharbeitet.

Um $\mathcal{A}$ beschreiben zu können führen wir folgende Notation ein:

- Wir numerieren die Zyklen von links nach rechts mit $Z_1, Z_2, \ldots, Z_m$ und die Sehnen mit (x_0, y_0), $(x_1, y_1), \ldots, (x_m, y_m)$. Dabei kann es vorkommen, dass x_i und x_{i+1} (oder y_i und y_{i+1}) zusammenfallen. Die Knotenmenge eines Zyklus Z_i bezeichnen wir mit $V(Z_i)$.

- Für jede Sehne (x_i, y_i) bezeichnet G_i den gesamten Graphen links von (x_i, y_i) inklusive die Kante (x_i, y_i) und H_i den Graphen rechts von (x_i, y_i) ohne die Kante (x_i, y_i)[13]. Die Knotenmenge von G_i und H_i bezeichnen wir mit $V(G_i)$ und $V(H_i)$.

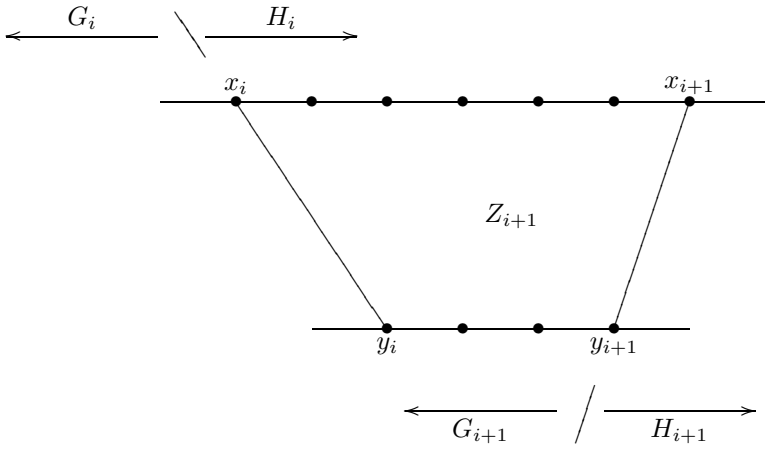

Abbildung 14:

- Für eine Sehne (x_i, y_i) und eine Menge von Knoten M bezeichne $I_{G_iH_i}(M)$ bzw. $I_{H_iG_i}(M)$ den charakteristischen Vektor von M bezüglich (x_i, y_i). Umgekehrt bezeichne für einen gegebenen Vektor $I_{G_iH_i} \in \{0,1\}^3$ $M(I_{G_iH_i})$ eine beliebige Realisierung dieses Vektors, das heisst M enthält höchstens einen Vertreter jeder Äquivalenzklasse und $I_{G_iH_i}(M(I_{G_iH_i})) = I_{G_iH_i}$.

- Für eine Sehne (x_i, y_i) gibt ein Vektor $R_{x_i,y_i} = (\delta_{x_i}, \delta_{y_i}, I_{G_iH_i}, I_{H_iG_i})$ die Randbedingungen wie oben beschrieben an. Ein solcher Randbedingungsvektor heisst gültig, falls sowohl $I_{G_iH_i}$ als auch $I_{H_iG_i}$ eine Realisierung haben. Von nun an seien stillschweigend alle betrachteten Randbedingungen gültig.

  Wir sagen, dass eine Menge $M \subseteq V$ die Forderung $\delta_{x_i}, \delta_{y_i}, I_{G_iH_i}$ erfüllt, wenn $\delta_{x_i}(M) = \delta_{x_i}$, $\delta_{y_i}(M) = \delta_{y_i}$ und $I_{G_iH_i}(M) = I_{G_iH_i}$ gilt.

- Für jede Sehne und für jede gültige Randbedingung R_{x_i,y_i} bezeichnet $\mathrm{Opt}(R_{x_i,y_i})$ eine minimale Menge von Punkten in G_i, die die Forderung δ_x, δ_y, $I_{G_iH_i}$ erfüllt und zusammen mit $I_{H_iG_i}$ alle Kanten von G_i entdeckt. Falls keine solche Menge existiert nennen wir die Randbedingung R_{x_i,y_i} nicht erfüllbar.

- Oben haben wir Punkte links (oder rechts) einer Sehne in Bezug auf diese Sehne in drei Äquivalenzklassen geteilt. Nun teilen wir Punkte zwischen

[13] aber mit den Knoten x_i und y_i

zwei Sehnen (x_i, y_i) und (x_{i+1}, y_{i+1}) auf die gleiche Art aber in Bezug auf beide Sehnen in neun Äquivalenzklassen ein. Zum Beispiel bezeichne $[x_i, x_{i+1}]$ die Klasse der Punkte, die näher bei x_i als bei y_i und gleichzeitig näher bei x_{i+1} als bei y_{i+1} liegen (wiederum ohne die Sehnenendpunkte selbst) etc.

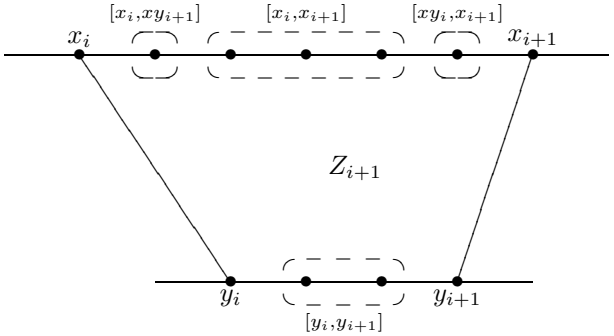

Abbildung 15: Äquivalenzklassen zwischen zwei Sehnen.

Für eine Menge M von Messpunkten zwischen diesen beiden Sehnen enthält der charakteristische Vektor
$J_{i,i+1}(M)=(\delta_{[x_i,x_{i+1}]}, \delta_{[x_i,y_{i+1}]}, \delta_{[x_i,xy_{i+1}]}, \delta_{[y_i,x_{i+1}]}, \ldots, \delta_{[xy_i,xy_{i+1}]}) \in \{0,1\}^9$
die ganze Information, die für die Entdeckung von Kanten links von (x_i, y_i) und rechts von (x_{i+1}, y_{i+1}) relevant ist. Analog wie oben bezeichne für einen gegebenen Vektor $J_{i,i+1}$ $M(J_{i,i+1})$ eine beliebige Realisierung dieses Vektors. $J_{i,i+1}$ heisst gültig, falls er eine Realisierung besitzt.

Der Algorithmus $\mathcal{A}$ startet bei (x_0, y_0) und berechnet für jede gültige Randbedingung $R_{x_0,y_0} = (\delta_{x_0}, \delta_{y_0}, 0, 0, 0, I_{H_0G_0})$ die Menge $\mathrm{Opt}(R_{x_0,y_0})$.[14]

Dann berechnet er sukzessive für jede Sehne (x_i, y_i) $i = 1, \ldots, m$ und jede gültige und erfüllbare Randbedingung $R_{x_i,y_i} \in \{0,1\}^8$ die Menge $\mathrm{Opt}(R_{x_i,y_i})$. (Genauere Beschreibung eines Schrittes folgt später.) Es ist klar, dass für jede erfüllbare Randbedingung $R_{x_m,y_m} = (\delta_{x_m}, \delta_{y_m}, I_{G_mH_m}, 0, 0, 0)$ die Menge $\mathrm{Opt}(R_{x_m,y_m})$ alle Kanten des Graphen entdeckt. Wählt man unter diesen Mengen eine minimale aus (sie muss nicht eindeutig sein), so hat man eine optimale Lösung von LG-ALL-E gefunden.

2.4.1 Ein Schritt des Algorithmus $\mathcal{A}$

Angenommen für eine Sehne (x_i, y_i) sind für alle erfüllbaren Randbedingungen $R_{x_i,y_i} = (\delta_{x_i}, \delta_{y_i}, I_{G_iH_i}, I_{H_iG_i}) \in \{0,1\}^8$ die Mengen $\mathrm{Opt}(R_{x_i,y_i})$ bekannt. Sei $R_{x_{i+1},y_{i+1}} = (\delta_{x_{i+1}}, \delta_{y_{i+1}}, I_{G_{i+1}H_{i+1}}, I_{H_{i+1}G_{i+1}}) \in \{0,1\}^8$ eine gültige Randbedingung für (x_{i+1}, y_{i+1}), für die nun der Algorithmus $\mathcal{A}$ in diesem Schritt $\mathrm{Opt}(R_{x_{i+1},y_{i+1}})$ berechnen soll. Dazu geht $\mathcal{A}$ wie folgt vor:

[14] $I_{G_0H_0} = (0,0,0)$, da G_0 nur aus der Kante (x_0, y_0) besteht und die Äquivalenzklassen $[x_0]^{(0)}$, $[y_0]^{(0)}$, $[xy_0]^{(0)}$ folglich leer sind.

Für jede gültige Randbedingung $R_{x_i,y_i} = (\delta_{x_i}, \delta_{y_i}, I_{G_iH_i}, I_{H_iG_i}) \in \{0,1\}^8$ und jeden gültigen Vektor $J_{i,i+1} \in \{0,1\}^9$ berechnet er gemäss Flussdiagramm in Abbildung 16 (falls sie existiert) eine Menge $M \subseteq V(G_{i+1})$, die die Forderung $\delta_{x_{i+1}}, \delta_{y_{i+1}}, I_{G_{i+1}H_{i+1}}$ erfüllt und zusammen mit $I_{H_{i+1}G_{i+1}}$ alle Kanten von G_{i+1} entdeckt. Unter allen solchen Mengen wählt er dann eine minimale aus. (Falls es keine solche Menge gibt ist $R_{x_{i+1},y_{i+1}}$ nicht erfüllbar.) Bezeichnen wir die ausgewählte Menge mit $\mathcal{A}(R_{x_{i+1},y_{i+1}})$.

Behauptung 47. *Falls die Randbedingung $R_{x_{i+1},y_{i+1}} = (\delta_{x_{i+1}}, \delta_{y_{i+1}}, I_{G_{i+1}H_{i+1}}, I_{H_{i+1}G_{i+1}}) \in \{0,1\}^8$ erfüllbar ist berechnet $\mathcal{A}$ in diesem Schritt $\mathrm{Opt}(R_{x_{i+1},y_{i+1}})$, das heisst es gilt $\mathcal{A}(R_{x_{i+1},y_{i+1}}) = \mathrm{Opt}(R_{x_{i+1},y_{i+1}})$. Denn für jede Menge $S \subseteq V(G_{i+1})$, die die Forderung $\delta_{x_{i+1}}$, $\delta_{y_{i+1}}$, $I_{G_{i+1}H_{i+1}}$ erfüllt und zusammen mit $I_{H_{i+1}G_{i+1}}$ alle Kanten von G_{i+1} entdeckt gilt: $|S| \geq |\mathcal{A}(R_{x_{i+1},y_{i+1}})|$.*

Beweis. Sei $S = S_i \cup S_{i+1}$ wobei $S_i \subseteq V(G_i)$ und $S_{i+1} \subseteq V(G_{i+1}) \setminus V(G_i) = V(Z_{i+1}) \setminus \{x_i, y_i\}$. Seien $I_{G_iH_i}(S_i)$, $\delta_{x_i}(S_i)$, $\delta_{y_i}(S_i)$, $J_{i,i+1}(S_{i+1})$, $\delta_{x_{i+1}}(S_{i+1})$ und $\delta_{y_{i+1}}(S_{i+1})$ die charakteristischen Vektoren von S_i und S_{i+1} und sei weiter $I_{H_iG_i}(S_{i+1}, I_{H_{i+1}G_{i+1}}) := I_{H_iG_i}(S_{i+1} \cup M(I_{H_{i+1}G_{i+1}}))$.

Dann erfüllt S_i trivialerweise die Forderung $\delta_{x_i}(S_i)$, $\delta_{y_i}(S_i)$, $I_{G_iH_i}(S_i)$ und zusammen mit $I_{H_iG_i}(S_{i+1}, I_{H_{i+1}G_{i+1}})$ entdeckt S_i alle Kanten von G_i. Der Randbedingungsvektor $R_{x_i,y_i}(S, I_{H_{i+1}G_{i+1}}) = (\delta_{x_i}(S_i), \delta_{y_i}(S_i), I_{G_iH_i}(S_i), I_{H_iG_i}(S_{i+1}, I_{H_{i+1}G_{i+1}}))$ ist also gültig und erfüllbar und es gilt $|S_i| \geq \mathrm{Opt}(R_{x_i,y_i}(S, I_{H_{i+1}G_{i+1}}))$. Der Vektor $J_{i,i+1}(S_{i+1})$ ist auch gültig.

Diese beiden Vektoren wurden also bei der Berechnung von $\mathcal{A}(R_{x_{i+1},y_{i+1}})$ vom Algorithmus betrachtet und haben (nach ihrer Definition) die ersten beiden Tests des Flussdiagrams bestanden.

Nun kommen wir zur Entdeckung von Z_{i+1}:

- Falls S_{i+1} leer ist, einen Punkt aus einer Äquivalenzklasse mit nur einem Vertreter enthält oder mindestens 2 Punkte enthält, folgt sofort, dass der Output des Flussdiagrams eine Menge M mit $|M| \leq |S|$ ist. In den ersten beiden Fällen ist $M = \mathrm{Opt}(R_{x_i,y_i}(S, I_{H_{i+1}G_{i+1}})) \cup S_{i+1}$, im dritten Fall ist spätestens der vierte Test "YES".

- Es bleibt noch den Fall zu behandeln, wenn S_{i+1} einen einzigen Punkt enthält, aber aus einer Äquivalenzklasse mit mehreren Vertetern.

  Zu zeigen ist, dass dann jeder Vertreter dieser Äquivalenzklasse zusammen mit $I_{G_iH_i}(S_i), \delta_{x_i}(S_i), \delta_{y_i}(S_i)$ und $I_{H_{i+1}G_{i+1}}$ alle Kanten von Z_{i+1} entdeckt.

  Dies ist trivialerweise der Fall, wenn bereits $I_{G_iH_i}(S_i), \delta_{x_i}(S_i), \delta_{y_i}(S_i)$ und $I_{H_{i+1}G_{i+1}}$ alle Kanten von Z_{i+1} entdecken, oder wenn Z_{i+1} ein gerader Zyklus ist.

  Ist der Zyklus Z_{i+1} ungerade und entdecken $I_{G_iH_i}(S_i), \delta_{x_i}(S_i), \delta_{y_i}(S_i)$ und $I_{H_{i+1}G_{i+1}}$ nicht alle seinen Kanten, so bleibt genau eine Kante unentdeckt.

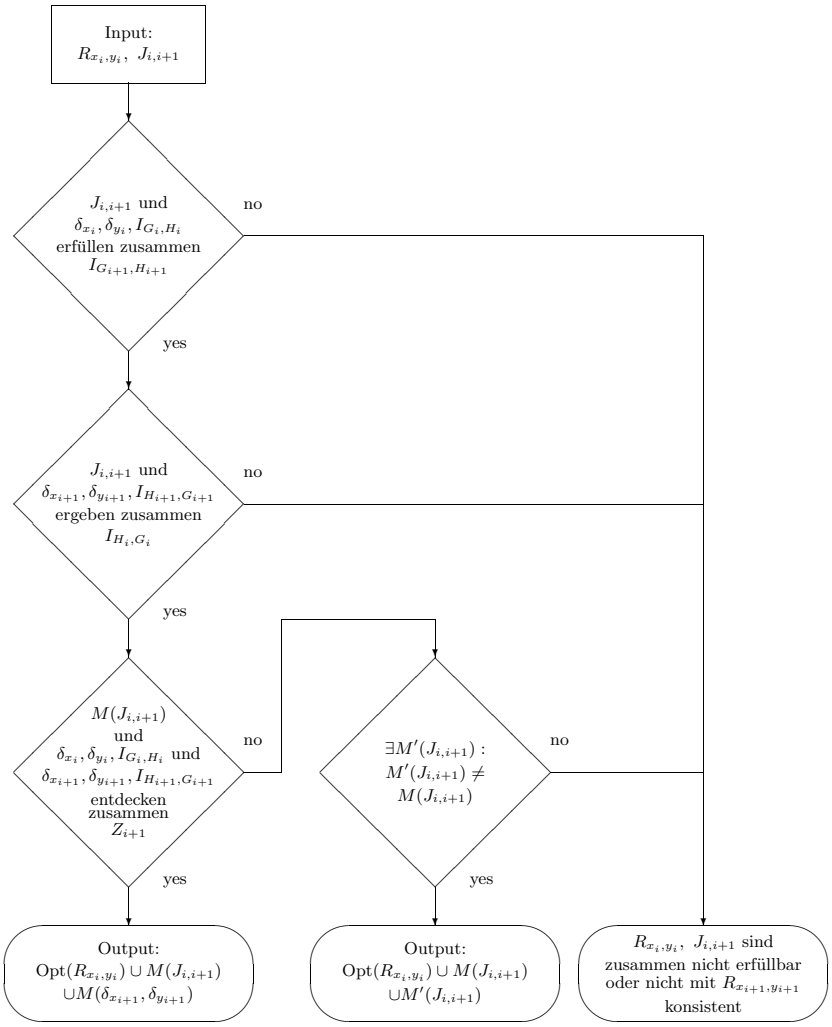

Abbildung 16: Flussdiagramm, das beschreibt, wie innerhalb des Algorithmus ausgehend von R_{x_i,y_i} und $J_{i,i+1}$ eine Menge $M \subseteq V(G_{i+1})$ berechnet wird, die die Forderung $\delta_{x_{i+1}}, \delta_{y_{i+1}}, I_{G_{i+1}H_{i+1}}$ erfüllt und zusammen mit $I_{H_{i+1}G_{i+1}}$ alle Kanten von G_{i+1} entdeckt. Bemerkung: Da 2 beliebige Knoten eines Kreises alle seine Kanten entdecken, folgt aus dem Erreichen des vierten IF-statements, dass $M(J_{i,i+1})$ nur einen einzigen Punkt enthält. Gibt es einen weiteren Vertreter seiner Äquivalenzklasse, entdecken diese gemeinsam Z_{i+1}.

– Ist es eine der Sehnen (x_i, y_i), (x_{i+1}, y_{i+1}), so kann es einen Knoten in $V(Z_{i+1})$ geben, der sie nicht entdeckt (der Knoten mit dem gleichen Abstand zu beiden Endpunkten der unentdeckten Sehne), dieser ist aber der einzige Vertreter seiner Äquivalenzklasse.
– Ist es keine der Sehnen, so ist es eine Kante deren Endpunkte von einem der Knoten $x_i, y_i, x_{i+1}, y_{i+1}$ den gleichen Abstand haben. Dann gibt es aber keinen anderen Knoten in Z_{i+1}, von dem die Enpunkte der unentdeckten Sehne den gleichen Abstand haben, und somit wird sie von jedem Knoten zwischen den beiden Sehnen entdeckt.

Damit ist bewiesen, dass im Flussdiagramm bereits der dritte Test mit dem Output $M = \mathrm{Opt}(R_{x_i,y_i}(S, I_{H_{i+1}G_{i+1}})) \cup M(J_{i,i+1})$ bestanden wird, wobei klarerweise $|M| \leq |S|$ gilt.

Da die Anzahl der Äquivalenzklassen und damit auch die Anzahl der Randbedingungsvektoren für jede Sehne durch eine Konstante beschränkt ist und die Anzahl Sehnen sowie die Grösse der optimalen Messpunktmenge linear in $n = |V|$ ist, folgt, dass der Algorithmus, so wie er beschrieben ist in polynomieller Zeit in n läuft. Ist der Output des Flussdiagrams nicht die Menge selbst sondern nur ihre Grösse (mit der Referenz auf R_{x_i,y_i}, $J_{i,i+1}$ und der Angabe ob der 3. Test bestanden wurde) so ist die Laufzeit sogar linear in n.

2.5 Der MinMaxEdges Algorithums für die online Approximation von LG-ALL

In [1] wurde der MinMaxEdges Algorithmus für die online Approximation von LG-ALL vorgeschlagen, der in einem Schritt den nächsten Messpunkt wie folgt bestimmt:

Sei S die Menge der bis jetzt gemessenen Knoten und $E_S \cup \bar{E}_S$ die Menge der bis jetzt entdeckten Kanten und Nichtkanten. Sei $G_S = (V, E_S)$ und $G'_S = (V, (E \cup \bar{E}) \backslash \bar{E}_S)$. Die Kantenmenge von G_S enthält also nur die bis jetzt entdeckten Kanten, die Kantenmenge von G'_S enthält ausser den bis jetzt entdeckten Kanten auch alle potentiellen Kanten.

Für die Abstände in G, G_S und G'_S gilt $\mathrm{d}_{G'_S}(a,b) \leq \mathrm{d}_G(a,b) \leq \mathrm{d}_{G_S}(a,b)$, das heisst $\mathrm{d}_G(a,b) \in [\mathrm{d}_{G'_S}(a,b), \mathrm{d}_{G_S}(a,b)]$. Betrachten wir nun einen potentiellen Messknoten v. Für jeden Knoten wird so die untere und obere Schranke für seinen Abstand von v in G berechnet. Eine Kante (a,b) wird sicher durch v entdeckt wenn gilt: $[\mathrm{d}_{G'_S}(v,a), \mathrm{d}_{G_S}(v,a)] \cap [\mathrm{d}_{G'_S}(v,b), \mathrm{d}_{G_S}(v,b)] = \emptyset$, sie kann muss aber nicht entdeckt werden in allen anderen Fällen ausser $\mathrm{d}_{G'_S}(v,a) = \mathrm{d}_{G_S}(v,a) = \mathrm{d}_{G'_S}(v,b) = \mathrm{d}_{G_S}(v,b)$. So können Schranken für die minimale und maximale Anzahl Kanten (min_v und max_v) bestimmt werden, die eine Messung von v entdecken kann. Aufgrund dieser Information wird (nach einem noch zu bestimmenden Kriterium) der nächste Messpunkt gewählt. In der MinMaxEdges(min) Variante des Algorithmuss wird zuerst min_v und dann max_v maximiert, in der MinMaxEdges(max) Variante umgekehrt. Es ist auch sinnvoll ein Kriterium zu betrachten, das beide Werte gleichzeitig berücksichtigt.

In [1] wurden MinMaxEdges(min) und MinMaxEdges(max) getestet. Die dort erhaltenen Resultate schlossen die Hoffnung nicht aus, es könnte sich bei MinMaxEdges um einen Algorithmus mit einer konstanten Approximationsrate handeln.

Inzwischen wissen wir, dass kein $o(\log n)$-Approximationsalgorithmus für LG-ALL existiert. Ein konkretes Beispiel eines Graphen, auf dem MinMaxEdges(min) nur eine Approximationsrate $\Omega(\log n)$ erreicht, scheint der Gittergraph zu sein.

Satz 48. *Zwei Messungen sind hinreichend, um alle Kanten und Nichtkannten eines Gittergraphs zu entdecken.*

Beweis. Es ist leicht zu sehen, dass zwei beliebige, nicht gegenüberliegende Eckpunkte des Gitters alle Kanten und Nichtkanten entdecken.

Wir haben den MinMaxEdges(min) Algorithmus für Gittergraphen getestet, wobei wir die erste Messung immer in der Mitte des Gitters bestimmt haben. In der folgenden Tabelle ist die Anzahl Messknoten m aufgeführt, die der Algorithmus gebraucht hat, um $n \times n$ Gittergraphen zu rekonstruieren. Interessant ist neben der Anzahl auch die Verteilung der Messknoten. Bei den getesteten Gittern mit $n = 13, \ldots, 139$ lag die eine Koordinate (o.B.d.A y-Koordinate) der ersten $m-2$ Messknoten zwischen $\lceil \frac{n}{2} - 1 \rceil$ und $\lceil \frac{n}{2} + 1 \rceil$, während die x-Koordinate ab dem vierten Messknoten der Reihe nach immer ungefähr in der

n:	3	5	9	11	13	17	19	31	33	59	79	99	119	139
m:	3	5	5	6	7	7	9	9	11	11	13	13	13	15

Tabelle 1:

Mitte zwischen dem am weitesten links liegenden Messknoten und dem linken Rand oder dem am weitesten rechts liegenden Messknoten und dem rechten Rand lag, bis links 1 oder 2 und rechts $n-2$ oder $n-1$ erreicht wurde. Dann folgten die letzten 2 Messknoten irgendwo auf dem Gitterrand.

Sollte diese Verteilung für Gitter beliebiger Grösse gelten, würde daraus für MinMaxEdges(min) die Approximationsrate $\Omega(\log n)$ folgen.[15]

Bei den in [1] getesteten Graphen haben MinMaxEdges(min) und MinMaxEdges(max) immer sehr ähnliche Resultate erzielt. Bei dem Gittergraphen ist es nicht mehr der Fall. In unseren Tests mit Gittergraphen der Grösse 25×25, 33×33, 67×67 und 131×131 hat MinMaxEdges(max) immer nur 4 Messungen gebraucht um alle Kanten und Nichtkanten des Gitters zu entdecken.

[15] Da beim Gittergraph bereits die erste Messung alle Kanten liefert, wäre in diesem Fall ein Algorithmus erfolgreicher, der in jedem Schritt den nächsten Messknoten unter der Annahme wählt, dass bereits alle Kanten entdeckt wurden.

3 Das SPT-MODELL

Definition 49. Das *SPT-Modell* ist ein Messmodell des Graphrekonstruktionsproblems, in dem die Messung eines Knotens v einen Kürzeste-Wege-Baum (shortest-path tree) $SPT(v)$ mit Wurzel v liefert.

Hier muss präzisiert werden, auf welche Art und Weise die Messprozedur aus mehreren kürzesten Wegen von v zu v' einen auswählt, sowie ob diese Information bei der Auswahl der Messknotenmenge bekannt ist und ob sie bei der Rekonstruktion des Graphen benützt werden darf. Daraus ergeben sich wieder verschiedene Varianten des Problems.

Wir betrachten hier lediglich denn Fall, wo die Eindeutigkeit der kürzesten Wege mittels einer zu G gehörenden Gewichtsfunktion, mit den Werten nahe bei 1, die auch bei der Rekonstruktion bekannt ist, erreicht wird.

Definition 50. Das offline *SPT-Graphrekonstruktionsproblem* (G, w, k) lautet: Gegeben ein Graph $G=(V, E)$, eine Gewichtsfunktion w: $(V, V) \longrightarrow (1-\epsilon, 1+\epsilon)$, $\epsilon << 1$ und eine ganze Zahl k. Gibt es eine Menge von k Messpunkten in V, die alle Kanten (und Nichtkanten) von G, mit Kenntniss von w, entdeckt?

Definition 51. Bezeichne $SP(u, v)$ den kürzesten Weg von u zu v, so heisst die Anzahl Kanten von $SP(u, v)$ der *Schichtenabstand* von u zu v. Die Summe der Gewichte der Kanten von $SP(u, v)$ bezeichnen wir mit $w(SP(u, v))$.

Bemerkung 52. Bei diesem Modell ist also E_v die Kantenmenge von $SPT(v)$ und für $\bar{E}_v$ gilt: Eine Nichtkante $(t, u) \in \bar{E}$ wird genau dann durch v entdeckt, wenn im $SPT(v)$ gilt: t und u sind entweder mindestens 2 Schichten[16] voneinander entfernt, oder sie liegen in benachbarten Schichten (o.B.d.A. t näher zu v als u) und es gilt $w(SP(v, t)) + w(t, u) < w(SP(v, u))$.

3.1 Die NP-Vollständigkeit von SPT-ALL-E

Gegeben ein gewichteter Graph $G = (V, E, w)$ und eine Menge $S \subseteq V$ kann man in polynomieller Zeit nachprüfen, ob S alle Kanten von G entdeckt. SPT-ALL-E ist also in NP.

Um zu zeigen, dass SPT-ALL-E NP-schwer ist, geben wir eine Reduktion von 3-SAT auf SPT-ALL-E an, die in polynomieller Zeit ausgeführt werden kann. Zu jeder Instanz ϕ von 3-SAT konstruieren wir (in polynomieller Zeit) eine Instanz (G, w, k) von SPT-ALL-E so dass gilt: Die boolsche Formel ϕ ist genau dann erfüllbar, wenn es eine Menge $S \subset V$ von k Messpunkten gibt, die mit Kenntnis von w alle Kanten des Graphen G entdeckt.

Sei also eine boolsche Formel ϕ mit n Variablen $x_1, \ldots, x_n$ und m Klauseln $C_1, \ldots, C_m$ gegeben.

- Die Punktmenge V des zu konstruierenden Graphen G besteht aus den Literalknoten $x_1, \bar{x_1}, \ldots, x_n, \bar{x_n}$, den Klauselknoten $C_1, \ldots, C_m$ und einem einzigen Hilfsknoten v.

[16] Nicht mit dem Schichtenabstand von t zu u verwechseln!

- Die Kantenmenge E von G besteht aus den Kanten $(x_1, \bar{x_1}), \ldots, (x_n, \bar{x_n})$ zwischen den beiden Literalknoten jeder Variablen, aus den Kanten die jede Klausel mit ihren drei Literalen verbinden und aus den Kanten zwischen dem Hilfsknoten v und allen anderen Punkten von G.
- Die Gewichtsfunktion w ist wie folgt definiert: Alle von v ausgehenden Kanten, haben das Gewicht 1, alle Kanten zwischen den beiden Literalen einer Variablen haben das Gewicht $1+3\epsilon$, und alle Kanten zwischen einem Literal und einer Klausel haben das Gewicht $1-\epsilon$. (Die Gewichte der Nichtkanten sind bei der Entdeckung der Kanten irrelevant, können also beliebig sein.)

Abb. 17 zeigt einen solchen gewichteten Graphen (G, w).

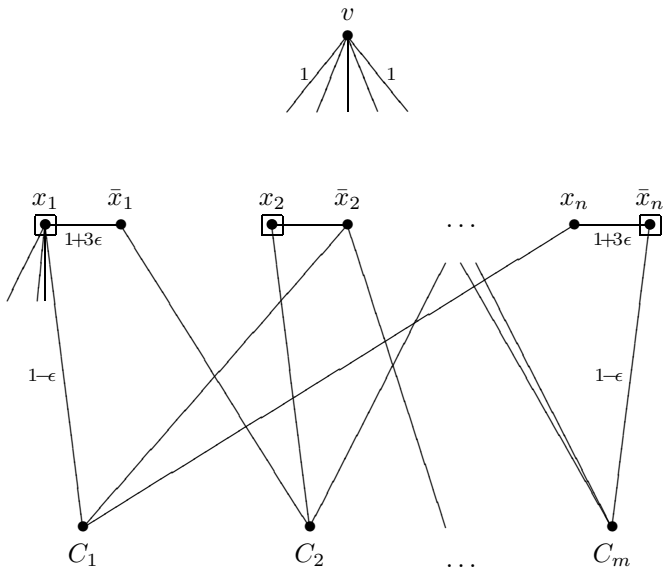

Abbildung 17: Graph zur Reduktion von 3-SAT auf SPT-ALL-E.

Bemerkung 53. Die Gewichtsfunktion w ist so konstruiert, dass gilt:

(i) Der kürzeste Weg von einem Klauselknoten zu einem nicht benachbarten Literalknoten führt immer via den Hilfsknoten v (und nicht via einen zu beiden benachbarten Literalknoten).

(ii) Sind zwei Literalknoten von verschiedenen Variablen mit dem gleichen Klauselknoten verbunden, so führt der kürzeste Weg zwischen diesen Literalknoten via den gemeinsamen Klauselknoten (und nicht via v).

Lemma 54. *Eine Messung an je einem der beiden Literale jeder Variablen ist hinreichend und notwendig um alle Kanten zwischen den Literalen zu entdecken.*

Beweis.

- Hinreichend: Für jede Kante $(x_i, \bar{x}_i)$ wird einer ihrer Endknoten gemessen, sie wird also trivialerweise entdeckt.

- Notwendig: Sei $e = (x_i, \bar{x}_i)$. Um e zu entdecken muss ein Messpunkt notwendigerweise verschiedene Schichtenabstände zu x_i und $\bar{x}_i$ haben. Dies wird (abgesehen von den Endpunkten von e) höchstens von einigen Klauselknoten erfüllt. Der kürzeste Weg von einem Klauselknoten zu einem Literalknoten ist aber entweder der direkte (falls diese beiden Knoten durch eine Kante verbunden sind) oder er führt via den Hilfsknoten v. In keinem Fall enthält er also die Kante e. Diese wird somit von keinem Klauselknoten entdeckt.

Lemma 55. *Eine Messpunktmenge $S \subseteq V$, die alle Kanten $(x_i, \bar{x}_i)$, $i = 1, \ldots, n$ entdeckt, entdeckt auch alle von v ausgehenden Kanten.*

Beweis. Sei S gegeben. Nach Lemma 54 enthält S für jede Variable einen ihrer beiden Literalknoten. Die Kanten von den gemessen Literalknoten zu v werden trivialerweise entdeckt.

Sei x_i nun ein nicht gemessener Literalknoten. Um die Kante (v, x_i) zu entdecken, brauchen wir ein gemessenes Literal (ungleich $\bar{x}_i$), das in keiner Klausel zusammen mit x_i vorkommt. (Da sonst wegen den Kantengewichten der Weg via die gemeinsame Klausel kürzer ist.) Existiert kein solches, transformieren wir ϕ, durch Hinzufügen einer neuen Klausel $C_{m+1} = (x_{n+1} \vee x_{n+2} \vee x_{n+3})$, in eine neue boolsche Formel ψ. Die Äquivalenz zwischen ϕ und ψ ist offensichtlich. Die Menge S enthält dann (o.B.d.A.) zusätzlich auch noch die Knoten x_{n+1}, x_{n+2} und x_{n+3}. Der Messknoten x_{n+1} entdeckt nun alle Kanten von v zu den Punkten $x_1, \bar{x_1}, \ldots, x_n, \bar{x_n}$. Die Kanten von v zu den neuen Literalen werden durch einen der ursprünglichen Messknoten entdeckt.

Es bleibt zu zeigen, wie die Kanten von v zu den Klauselknoten entdeckt werden. Eine Kante $e = (v, C_j)$ wird wegen der Kantengewichte (Bemerkung 53) von jedem in der Klausel C_j nicht vorkommenden Literal entdeckt. Existiert kein solches Literal, so benutzen wir wieder die oben definierte zu ϕ äquivalente boolsche Formel ψ.

Lemma 56. *Eine Menge $S \subset V$ von n Messpunkten, die alle Kanten zwischen den Literalen entdeckt, entdeckt genau dann alle Kanten von G, wenn jeder Klauselknoten mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden ist.*

Beweis. Sei $S \subset V$ gegeben. Nach Voraussetzung entdeckt S alle Kanten zwischen den Literalknoten, nach Lemma 55 auch alle von v ausgehenden Kanten. Nach Lemma 54 enthält S für jede Variable x_i eines ihrer beiden Literale. Dies sind bereits n Punkte, die Menge S enthält also keine weiteren Messpunkte. Die Kanten von den Klauselknoten zu den gemessenen Literalknoten werden trivialerweise entdeckt.

Es bleibt zu zeigen, dass S genau dann alle Kanten von den nichtgemessenen Literalknoten zu den Klauselknoten entdeckt, wenn jeder Klauselknoten mit mindestens einem gemessenen Literalknoten verbunden ist.

$\Rightarrow$: Sei $e = (C_j, x_i)$ eine von S entdeckte Kante, wobei $x_i \notin S$. Wegen der Kantengewichte kann e nicht durch $\bar{x}_i$ entdeckt werden (Bemerkung 53). Sei (o.B.d.A) $x_{i'} \in S$ der Messknoten der e entdeckt. C_j und x_i haben folglich verschiedene Schichtenabstände von $x_{i'}$. Da x_i von $x_{i'}$ nach Konstruktion den Schichtenabstand 2 hat und da es im G keine grösseren Schichtenabstände gibt, hat C_j den Schichtenabstand 1 von $x_{i'}$.

$\Leftarrow$: Sei der Klauselknoten C_j mit dem gemessenen Literalknoten x_i durch eine Kante verbunden. Die Kante (C_j, x_i) wird somit trivialerweise entdeckt. Wegen der Kantengewichte (Bemerkung 53 (ii)) entdeckt x_i auch die Kanten vom Klauselknoten C_j zu dessen beiden anderen Literalknoten.

Satz 57. *Sei ϕ eine boolsche Formel und $G_\phi = (V, E, w)$ der zu ihr konstruierte gewichtete Graph. ϕ ist genau dann erfüllbar, wenn eine Menge $S \subset V$ von n Messpunkten existiert, die alle Kanten des Graphen G_ϕ entdeckt.*

Beweis.

$\Rightarrow$: Sei eine ϕ-erfüllende Wahrheitsbelegung gegeben. Wir definieren die Messpunktmenge S als die Menge der wahren Literale. Diese n-punktige Menge entdeckt nach Lemma 54 alle Kanten zwischen den Literalen. Nach Lemma 56 entdeckt S genau dann alle Kanten des Graphen G, wenn jede Klausel mit mindestens einem gemessenen Literalknoten verbunden ist. Da nach Voraussetzung jede Klausel mindestens ein wahres Literal enthält, ist dies nach Konstruktion von G erfüllt.

$\Leftarrow$: Sei $S \subset V$ eine Menge von n Messpunkten, die alle Kanten von G entdeckt. S entdeckt insbesondere alle Kanten zwischen den Literalknoten und enthält somit nach Lemma 54 genau je eines der beiden Literale jeder der n Variablen (und keine weiteren Punkte). Nach Lemma 56 ist jeder Klauselknoten mit mindestens einem gemessenen Literalknoten verbunden. Setzen wir nun jedes gemessene Literal wahr, erhalten wir eine gültige und nach Konstruktion von G auch ϕ-erfüllende Wahrheitsbelegung.

Somit haben wir gezeigt:

Satz 58. *Das SPT-ALL-E Entscheidungsproblem ist NP-vollständig.*

3.2 Die NP-Vollständigkeit von SPT-ALL

Gegeben ein Graph $G = (V, E)$, eine Gewichtsfunktion $w : (V, V) \to (1-\epsilon, 1+\epsilon)$ und eine Menge von Messknoten $S \subseteq V$, kann man in polynomieller Zeit (in $|V|$) nachprüfen, ob S mit Kenntnis von w alle Kanten und Nichtkanten von G entdeckt. Das Entscheidungsproblem SPT-ALL ist also in NP.

Um zu zeigen, dass SPT-ALL auch NP-schwer ist, geben wir wieder eine polynomielle Reduktion von 3-SAT auf SPT-ALL an.

Zu einer gegebenen boolschen Formel ϕ (mit n Variablen und m Klauseln) konstruieren wir gemäss Abb.18 einen gewichteten Graphen $G = (V, E, w)$.

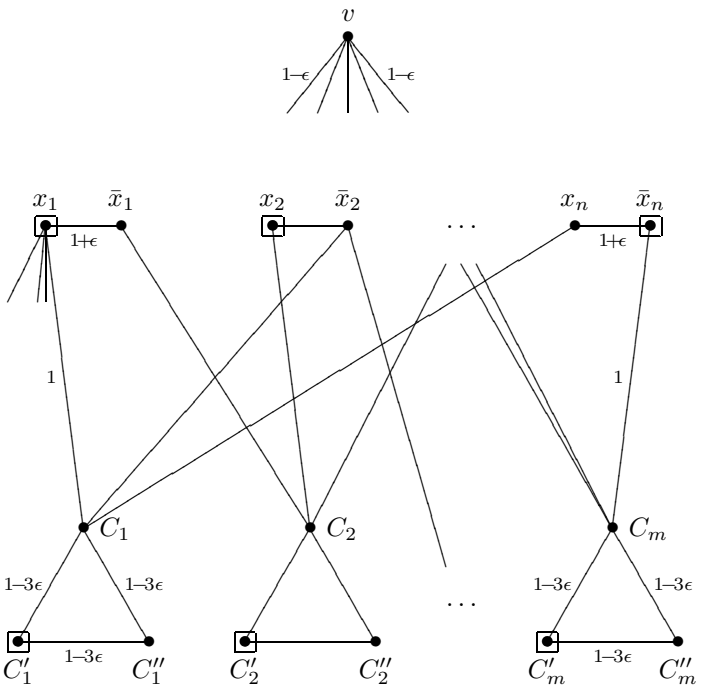

Abbildung 18: Graph zur Reduktion von 3-SAT auf SPT-ALL.

- Die Knotenmenge V von G besteht aus $2n$ Literalknoten $x_1, \bar{x_1}, \ldots, x_n, \bar{x_n}$, m Klauselknoten $C_1, \ldots, C_m$, $2m$ Klauselhilfsknoten $C'_1, C''_1, \ldots, C'_m, C''_m$ und aus dem Hilfsknoten v (der wieder mit allen anderen Knoten von G durch eine Kante verbunden ist).
- Die Kantenmenge E von G besteht aus den Kanten zwischen den beiden Literalknoten einer Variable, aus den Kanten der Klauseldreiecke, aus den Kanten, die jeden Klauselknoten mit seinen 3 Literalknoten verbinden, und aus den Kanten von v zu allen anderen Knoten in V.
- Die Gewichtsfunktion $w : (V, V) \to (1 - 6\epsilon, 1 + 6\epsilon)$ ist wie folgt definiert:
  - Alle von v ausgehenden Kanten haben das Gewicht $1 - \epsilon$
  - Die Kanten $(x_1, \bar{x_1}), \ldots, (x_n, \bar{x_n})$ zwischen den beiden Literalknoten einer Variable haben das Gewicht $1 + \epsilon$.
  - Alle Kanten der Klauseldreiecke haben das Gewicht $1 - 3\epsilon$
  - Alle Kanten und Nichtkanten von den Klauselknoten $C_1, \ldots, C_m$ zu den Literalknoten $x_1, \bar{x_1}, \ldots, x_n, \bar{x_n}$ haben das Gewicht 1.
  - Alle anderen Nichtkanten haben das Gewicht $1 - 5\epsilon$

.

Diese Konstruktion kann in polynomieller Zeit (in n) ausgeführt werden.

Bemerkung 59. Die Gewichtsfunktion w ist so konstruiert, dass gilt:

(i) Der kürzeste Weg von einem Klauselknoten zu einem nicht benachbarten Literalknoten führt immer via den Hilfsknoten v (und nicht via einen zu beiden benachbarten Literalknoten).

(ii) Der kürzeste Wege zwischen zwei Literalknoten von verschiedenen Variablen führt immer via den Hilfsknoten v (und nicht via einen zu beiden benachbarten Klauselknoten)

(iii) Der kürzeste Weg von einem Klauselhilfsknoten zu einem Literalknoten führt, falls vorhanden, via einen zu beiden benachbarten Klauselknoten, sonst via den Hilfsknoten v.

Lemma 60. *Eine Messung an je einem der beiden Literale jeder Variablen ist hinreichend und notwendig um alle Kanten*[17] *zwischen den Literalen zu entdecken.*

Beweis.

- Hinreichend: Für jede Kante $(x_i, \bar{x}_i)$ wird einer ihrer Endknoten gemessen, sie wird also trivialerweise entdeckt.
- Notwendig: Sei $e = (x_i, \bar{x}_i)$. Um e zu entdecken muss ein Messpunkt notwendigerweise verschiedene Schichtenabstände zu x_i und $\bar{x}_i$ haben. Dies wird (abgesehen von den Endpunkten von e) höchstens durch einige Klauselknoten erfüllt. Der kürzeste Weg von einem Klauselknoten zu einem Literalknoten ist aber entweder der direkte (falls diese beiden Knoten durch eine Kante verbunden sind) oder er führt via den Hilfsknoten v. In keinem Fall enthält er also die Kante e. Diese wird somit von keinem Klauselknoten entdeckt.

Lemma 61. *Eine Messpunktmenge $S \subseteq V$, die alle Kanten zwischen den Literalknoten entdeckt, entdeckt auch alle von den Literalknoten zu v führende Kanten, sowie alle Nichtkanten zwischen den nichtbenachbarten Literalen.*

Beweis. Sei S gegeben. Nach Lemma 60 enthält S für jede Variable einen ihrer beiden Literalknoten. Die Kanten und Nichtkanten von den gemessen Literalknoten werden trivialerweise entdeckt.
Sei nun x_i ein nicht gemessener Literalknoten. Wegen der Kantengewichte (Bemerkung 59) führt der kürzeste Weg zwischen zwei nichtbenachbarten Literalknoten immer via den Hilfspunkt v. Die Kante (v, x_i) wird also durch einen beliebigen gemessenen Literalknoten ausser $\bar{x}_i$ entdeckt. Die Nichtkante von x_i zu einem anderen Literalknoten $x_{i'}$ wird von $\bar{x}_i$ entdeckt, denn x_i und $x_{i'}$ haben verschiedene Schichtenabstände von $\bar{x}_i$ und $w(SP(\bar{x}_i, x_{i'})) = 2 - 2\epsilon > 2 - 4\epsilon = w(\bar{x}_i, x_i) + w(x_i, x_{i'})$.

[17] Mit Kanten sind wirklich nur die existierenden Kanten gemeint.

Lemma 62. *Eine Menge von Messpunkten, die alle Kanten der Klauseldreiecke entdeckt, enthält für jedes j einen der Hilfsknoten C'_j, C''_j. Eine Messpunktmenge, die für jedes j einen der Hilfsknoten C'_j, C''_j enthält, entdeckt alle Nichtkanten zwischen den Klauseldreiecken, alle Kanten zwischen den Klauseldreiecken und v, sowie alle Kanten und Nichtkanten von den Knoten der Klauseldreiecke zu den Literalknoten.*

Beweis. Für jedes j gilt: Die Hilfspunkte C'_j und C''_j haben beide Schichtenabstand 2 zu allen Punkten ausserhalb ihres Klauseldreiecks (ausser zu v) und Schichtenabstand 1 zu dem Klauselknoten C_j und zu v. Die Kante (C'_j, C''_j) wird also nur durch eine Messung an C'_j oder C''_j entdeckt.

Werde nun o.B.d.A C'_j gemessen. Die Kanten (C'_j, C''_j), (C'_j, v) und alle Nichtkanten ausgehend von C'_j werden also trivialerweise entdeckt.

Die Knoten C_j und C''_j sind die einzigen Knoten (ausser v), die zu C'_j Schichtenabstand 1 haben. Wegen Bemerkung 59(iii) entdeckt C'_j alle vom Klauselknoten C_j zu allen Literalknoten führende Kanten und Nichtkanten. Die Nichtkanten zwischen dem Klauselhilfsknoten C''_j und den Literalknoten werden ebenfalls von C_j entdeckt, denn für jeden Literalknoten x_i gilt: $w(SP(C_j, x_i)) \geq 2 - 3\epsilon > 2 - 8\epsilon = w(C_j, C''_j) + w(C''_j, x_i)$.

C'_j entdeckt auch die Nichtkanten zwischen C_j und anderen Klauselknoten, denn für einen Klauselknoten $C_{j'}$ gilt: $w(SP(C'_j, C_{j'})) = 2 - \epsilon > 2 - 8\epsilon = w(C'_j, C_j) + w(C_j, C_{j'})$. Analog zeigt man, dass C'_j auch die Nichtkanten zwischen C_j und allen Klauselhilfsknoten sowie die Nichtkanten zwischen C''_j und allen Knoten der anderen Klauseldreiecke entdeckt.

Die Kanten von C_j und C''_j zu v werden durch eine Messung an einem anderen (beliebigen) Klauselhilfspunkt $C'_{j'}$ entdeckt, da dann v den Schichtenabstand 1 und C_j und C''_j den Schichtenabstand 2 von $C'_{j'}$ haben und der kürzeste Weg von $C'_{j'}$ zu C_j und C''_j via v führt.

Bemerkung 63. Durch die Messung an C'_j wird die Kante (C_j, C''_j) nicht entdeckt, es ist also entweder eine zusätzliche Messung an ihren Endknoten notwendig oder diese Kante muss durch eine Messung an einem der 3 Klauselliterale entdeckt werden. (Es gibt keinen anderen Messknoten in G, der diese Kante entdeckt.) Die Klauseldreiecke haben daher wieder die wichtige Eigenschaft, dass ein gemessenes (wahres) Literal einer Klausel eine Messung im Klauseldreieck spart.

Lemma 64.

(i) Zur Entdeckung des ganzen Graphen G sind mindestens $n+m$ Messknoten notwendig und zwar je einer der beiden Literalknoten jeder der n Variablen und je ein Hilfsknoten (o.B.d.A C'_j) in jedem der m Klauseldreiecke.

(ii) Eine Menge $S \subset V$ von $n+m$ Messknoten, die alle Kanten $(x_1, \bar{x}_1), \ldots, (x_n, \bar{x}_n)$ und $(C'_1, C''_1), \ldots, (C'_m, C''_m)$ entdeckt, entdeckt genau dann den ganzen Graphen G, wenn jeder Klauselknoten mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden ist.

Beweis.

(i) Folgt direkt aus Lemma 60 und Lemma 62.

(ii) Sei S nach Voraussetzung. Dann enthält S nach (i) je einen der beiden Literalknoten jeder der n Variablen und die m Hilfsknoten $C'_1, \ldots, C'_m$ und keine weiteren Knoten. S entdeckt trivialerweise die Kanten $(C'_1, C_1), \ldots,$ (C'_m, C_m). Nach Lemma 61 und Lemma 62 entdeckt S auch alle von v ausgehenden Kanten, alle Kanten und Nichtkanten zwischen den Literalknoten und den Klauselknoten, alle Nichtkanten zwischen den nichtbenachbarten Literalknoten und alle Nichtkanten zwischen den Klauseldreiecken. Es bleibt also zu zeigen, dass S genau dann auch die Kanten $(C_1, C''_1), \ldots, (C_m, C''_m)$ (und somit den ganzen Graphen G) entdeckt, wenn jeder Klauselknoten mit mindestens einem gemessenem Literalknoten verbunden ist. Dies ist der Fall, denn die Kante (C_j, C''_j) wird durch keinen der Hilfsknoten $C'_1, \ldots, C'_m$ entdeckt und durch einen Literalknoten wird sie genau dann entdeckt, wenn dieser zum Klauselknoten C_j benachbart ist.

Satz 65. *Sei ϕ eine boolsche Formel und $G_\phi = (V, E, w)$ der zu ihr konstruierte gewichtete Graph. ϕ ist genau dann erfüllbar, wenn eine Menge $S \subset V$ von $n+m$ Messpunkten existiert, die alle Kanten und Nichtkanten des Graphen G_ϕ entdeckt.*

Beweis.

$\Rightarrow$: Sei eine ϕ erfüllende Wahrheitsbelegung gegeben. Wir definieren die Messpunktmenge S' als die Menge der wahren Literale und erweitern sie nach Lemma 64 (i) um die m Klauselhilfspunkte $C'_1, \ldots, C'_m$ zu S. Diese $n+m$-punktige Menge S entdeckt nun nach Lemma 64 (ii) genau dann den ganzen Graphen G wenn jeder Klauselknoten mit mindestens einem gemessenen Literalknoten durch eine Kante verbunden ist. Da jede Klausel mindestens ein wahres Literal enthält, ist dies nach Konstruktion von G und Definition von S erfüllt.

$\Leftarrow$: Sei $S \subset V$ eine Menge von $n+m$ Messpunkten, die alle Kanten von G entdeckt. Nach Lemma 64 (i) enthält S je eines der beiden Literale jeder der n Variablen und die m Klauselhilfsknoten $C'_1, \ldots, C'_m$ (und keine weiteren Punkte). Nach Lemma 64 (ii) ist jeder Klauselknoten mit mindestens einem gemessenen Literalknoten verbunden. Setzen wir nun jedes gemessene Literal wahr, erhalten wir eine gültige und nach Konstruktion von G auch ϕ-erfüllende Wahrheitsbelegung.

Somit haben wir gezeigt:

Satz 66. *Das SPT-ALL Entscheidungsproblem ist NP-vollständig.*

4 Rekapitulation und Ausblick

Unsere Untersuchungen haben gezeigt, dass es nicht nur, wie erwartet, NP-schwer ist das Graphrekonstruktionsproblem optimal zu lösen, sondern auch, dass es wider Erwarten auch NP-schwer ist dieses Problem mit Approximationsrate $o(\log n)$ zu approximieren.

Während es für den Offline-Fall einen von Set Cover übertragenen $O(\log n)$-Approximationsalgorithmus gibt, sind weiterhin keine online Approximationsalgorithmen mit bewiesenen Schranken bekannt.

Aufgrund von Experimenten mit Gittergraphen scheint der, in [1] vorgeschlagene MinMaxEdges(min) online Algorithmus, die untere Schranke $\Omega(\log n)$ zu haben, während für den MinMaxEdges(max) noch kein Graph bekannt ist, auf dem MinMaxEdges(max) schlecht ist.

Für das vermutlich schlechte Resultat, das MinMaxEdges(min) beim Gitter erzielt, gibt es in diesem speziellen Fall die einfache Abhilfe, den MinMaxEdges(min) mit einem Algorithmus zu kombinieren, der in jedem Schritt den nächsten Messknoten unter der Annahme wählt, dass bereits alle Kanten des Graphen entdeckt wurden.

Dazu wäre interessant zu untersuchen, wie stark sich das Optimum ändern kann, wenn man einem Graphen eine Kante zufügt oder entfernt.

Offen bleibt auch die Frage, wie das Problem aussieht, wenn statt einer exakten Rekonstruktion nur eine teilweise Rekonstruktion verlangt wird, zum Beispiel wenn nur 95% aller Kanten und Nichtkanten entdeckt werden sollen.

Weiter wurde hier gar nicht auf die anderen Varianten des SPT-Messmodells eingegangen, die aber nicht weniger relevant oder interesant sind, als der hier betrachtete Fall.

5 Schlussbemerkung

An dieser Stelle möchte ich mich bei Prof. Thomas Erlebach für die hervorragende Betreuung meiner Arbeit bedanken. Dank der richtigen Balance zwischen Unterstüzung und Selbstständigkeit hat diese Arbeit Spass gemacht und hat zu den beschriebenen Resultaten geführt.

Hiermit bestätige ich, dass ich diese Arbeit selbstständig und nur unter der Benützung der hier aufgeführten Quellen gemacht habe.

Zuzana Beerliova

Literatur

[1] F. Eberhard. Grundlagen für das Mapping des Internet-Graphen, Juli 2003. Semesterarbeit SA-2003.30, Computer Engineering and Networks Laboratory (TIK), ETH Zürich.

[2] L. S. Ram. Tree-based graph reconstruction, 2002. CGC Pre-Doc-Project, ETH Zürich.

[3] Bjarni V. Halldorsson, Magnus M. Halldorsson, R. Ravi. On the Approximability of the Minimum Test Collection Problem, ESA 2001: 158-169

[4] S. Arora and M. Sudan. Improved low degree testing and its applications. In Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, pages 485-495, El Paso, Texas, 4-6 May 1997.