Universität Paderborn
Prof. Dr. Johannes Blömer

Gruppe 27:
Eli Kogan-Wang (7251030)
David Noah Stamm (7249709)
Bogdan Rerich (7248483)
Jan Schreiber(7253698)

Berechenbarkeit und Komplexität – WS 2022/2023

Heimübung 9
Abgabe: 19. Dezember 2022 – 13:00 Uhr
(Dieser Übungszettel besteht aus 4 Aufgaben mit insgesamt 24 Punkten)

Aufgabe 1 (NP-Vollständigkeit) (3 Punkte)
In Heimübung 8 Aufgabe 2 haben wir die disjunktive Normaleform (DNF) einer aussagenlogischen Formel ϕ definiert und gezeigt, dass die Sprache aller erfüllbaren DNF–Formeln DisSat in P liegt. Eine solche Formel ist widerlegbar, wenn es (mindestens) eine Belegung gibt, sodass die Formel nicht wahr ist. Betrachten sie folgende Sprache

$$\textsc{DisUnsat} = \{\langle\phi\rangle \mid \phi \text{ ist eine widerlegbare Formel in disjunktiver Normalform}\} .$$

Zeigen Sie, dass DisUnsat NP-vollständig ist.

Lösung: Z.z.
1. DisUnsat $\in NP$
2. 3SAT $\leq_p$ DisUnsat

1. Konstruiere eine DTM M als Verifizierer:

1. Prüfe ob w von der Form $\langle\phi, c\rangle$, sodass ϕ eine aussagenlogische Formel in DNF und c eine Belegung der Variablen in ϕ. Wenn nicht, dann lehne ab.
2. Wenn die Formel ϕ mit der Belegung c falsch ist, dann akzeptiere.

Jede Variable kommt o.B.d.A. mindestens einmal in ϕ vor. In c kommt sie maximal einmal vor. Also liegt die Länge von c polynomiell unter der von ϕ Korrektheit:
w $\in$ DisUnsat:
Dann gibt es eine Belegung, sodass ϕ falsch ist. w $\notin$ DisUnsat:
Wenn w falsch kodiert ist lehnt M ab. Wenn ϕ nicht widerlegbar ist, dann lehnt M jedes w ab.

Schritt 1 und Schritt 2 lassen sich beide in linearer Zeit, also unter polynomieller Zeit ausführen.

2. Konstruiere eine polynomielle Reduktion f von 3SAT auf DisUnsat:
Sei w ein Wort. Wenn w nicht von der Form $\langle\phi\rangle$ mit ϕ eine aussagenlogische Formel in 3KNF ist, dann gebe $f(w) = \epsilon$ zurück.

w ist nun von der Form $\langle \phi \rangle$ mit ϕ eine aussagenlogische Formel in 3KNF.

Konstruiere eine aussagenlogische Formel ϕ' in DNF aus ϕ indem man alle $\wedge$ durch $\vee$ ersetzt und alle $\vee$ durch $\wedge$ ersetzt und alle Literale durch ihre negierten Versionen ersetzt.

Nun ist $\phi' \iff \neg\phi$ und ϕ' ist eine aussagenlogische Formel in DNF.

$f(w) = \langle \phi' \rangle$

Korrektheit:

$w \in$ 3SAT:

ϕ ist eine aussagenlogische Formel in 3KNF.

ϕ' ist eine aussagenlogische Formel in DNF.

Ein Erfüllung von ϕ ist eine Widerlegung von ϕ'.

$w \notin$ 3SAT:

Entweder ist w falsch kodiert oder $w = \langle \phi \rangle$ ist nicht erfüllbar.

Wenn w falsch kodiert ist, dann ist $f(w) = \epsilon$ und $\epsilon \notin$ DisUnsat.

Wenn $w = \langle \phi \rangle$ ist nicht erfüllbar, dann ist ϕ' nicht widerlegbar und $f(w) = \langle \phi' \rangle$ ist nicht in DisUnsat.

Polynomielle Laufzeit:

Die Länge von $f(w)$ ist polynomiell unter der von w.
Zusätzlich werden nur substitutionen ausgeführt, die polynomiell viele Schritte brauchen.

Damit ist f polynomiell. Und f ist eine Reduktion von 3SAT auf DisUnsat: 3SAT $\leq_p$ DisUnsat.

Aufgabe 2 (NP-Vollständigkeit) (7 Punkte)
Betrachten Sie die folgende Sprache

$$\textsc{HittingSet} = \left\{ \langle X, S, k \rangle \;\middle|\; \begin{array}{l} X \text{ ist eine endliche Menge, } S \subseteq \mathcal{P}(X) \text{, und es gibt} \\ H \subseteq X \text{ mit } |H| \leq k \text{, sodass für alle } T \in S : H \cap T \neq \emptyset. \end{array} \right\}.$$

Zeigen Sie, dass HittingSet NP-vollständig ist. Nutzen Sie in Ihrer Reduktion die *NP-vollständige* Sprache VertexCover, welche in Heimübung 7 Aufgabe 1 definiert wurde.

Lösung: Wir zeigen;

1. HittingSet $\in NP$
2. VertexCover $\leq_p$ HittingSet

Beweis von 1:
Wir zeigen existenz eines Verifizierers M für HittingSet.
Wir beschreiben M bei Eingabe w wie folgt:
Wenn w nicht von der Form $\langle\langle X, S, k\rangle, c\rangle$, wobei X eine endliche Menge, S eine endliche Menge von Mengen ist, k eine natürliche Zahl und c eine endliche Menge ist, dann lehne w ab. (Form-Check)
Nun ist w von der Form $\langle\langle X, S, k\rangle, c\rangle$, wobei X eine endliche Menge, S eine endliche Menge von Mengen ist, k eine natürliche Zahl und c eine endliche Menge ist.
Nun wird wie folgt verfahren:

1. Prüfe, ob $S \subseteq \mathcal{P}(X)$ indem man für jedes $T \in S$ prüft ob $T \subseteq X$. Wenn nicht, lehne ab.
2. Prüfe, ob $c \subseteq X$ indem man für jedes $x \in c$ prüft ob $x \in X$. Wenn nicht, lehne ab.
3. Prüfe, ob $|c| \leq k$. Wenn nicht, lehne ab.
4. Prüfe, ob für alle $T \in S$ gilt, dass $c \cap T \neq \emptyset$. Wenn nicht, lehne ab.
5. Akzeptiere.

Korrektheit:
Das Zertifikat c ist ein Hitting-Set für S mit $|c| \leq k$.
Wenn $\langle X, S, k\rangle \in$ HittingSet ist, dann gibt es ein Hitting-Set c mit $|c| \leq k$ für S.
Wenn $w \notin$ HittingSet, dann gibt es kein Hitting-Set für S mit $|c| \leq k$, oder w ist falsch kodiert.
Laufzeit:
Schritt 1 und 2 iterieren polynomiell über die Eingabe und sind damit polynomiell.
Schritt 3 ist trivial polynomiell in der Eingabe.
Schritt 4 iteriert polynomiell über die Mengen $T \in S$, als auch über die Elemente $x \in c$ und ist damit polynomiell, sofern $|c|$ polynomieller GröSSe ist.
Polynomielle GröSSe von c:
$|c|$ ist polynomieller GröSSe, da $x \subseteq X$ geprüft wird und X endlich ist.
Beweis von 2 (VertexCover $\leq_p$ HittingSet):
Wir zeigen existenz einer polynomiellen Reduktion f von VertexCover auf HittingSet.
Also: $w \in$ VertexCover $\iff f(w) \in$ HittingSet.
Wir beschreiben f bei Eingabe w wie folgt:
Wenn w nicht von der Form $\langle G, k\rangle$, wobei $G = (V, E)$ ein ungerichteter Graph ist und k eine natürliche Zahl ist, dann konstruiere eine Invalide Eingabe für HittingSet (bspw. ϵ).

Nun ist w von der Form $\langle G, k\rangle$, wobei $G = (V, E)$ ein ungerichteter Graph ist und k eine natürliche Zahl ist.
$f(\langle G = (V, E), k\rangle) = \langle V, E, k\rangle$.
f ist polynomiell:
f ist trivial polynomiell, da f nur die Eingabe prüft und diese dann direkt weitergibt (linear in der Eingabe).
Korrektheit:
Richtung $\Rightarrow$:
Wenn $w \in \textsc{VertexCover}$, dann existiert ein $C \subseteq V$ mit $|C| \leq k$ und für alle $e \in E$ gilt $e \cap C \neq \emptyset$.
Damit ist $H = C$ ein Hitting-Set für E auf V mit $|H| \leq k$.
Damit ist $f(w) = \langle V, E, k\rangle \in \textsc{HittingSet}$.
Richtung $\Leftarrow$:
Wenn $w \notin \textsc{VertexCover}$, dann ist w falsch kodiert, oder es existiert kein $C \subseteq V$ mit $|C| \leq k$ und für alle $e \in E$ gilt $e \cap C \neq \emptyset$.
Wenn w falsch kodiert ist, dann ist $f(w)$ falsch kodiert.
Wenn es kein $C \subseteq V$ mit $|C| \leq k$ und für alle $e \in E$ gilt $e \cap C \neq \emptyset$ gibt, dann gibt es auch kein $H \subseteq V$ mit $|H| \leq k$ und für alle $T \in E$ gilt $T \cap H \neq \emptyset$.
Damit ist $f(w) \notin \textsc{HittingSet}$.

Aufgabe 3 (NP-Vollständigkeit) (7 Punkte)

Betrachten Sie die folgende Sprache

$$\textsc{mTSP} = \left\{ (X, d, L) \;\middle|\; \begin{array}{c} X \text{ ist eine endliche Menge, } d \text{ eine Metrik auf } X\text{, und } L \in \mathbb{N}, \\ \text{sodass es eine Rundreise durch alle } x \in X \text{ der Länge} \leq L \text{ gibt.} \end{array} \right\}.$$

Die Definition von *Metrik auf X* finden sie auf Präsenzübung 9 Aufgabe 2. Zeigen Sie, dass $\textsc{mTSP}$ NP-vollständig ist. Verwenden Sie dafür eine Reduktion von TSP_{ent}, welche NP-vollständig ist, auf $\textsc{mTSP}$.

Lösung: Es ist folgendes zu zeigen:

1) mTSP $\in$ NP

2) $TSP_{ent} \leq_p mTSP$

Zu 1):
Sei V eine DTM.
Betrachte folgenden Verifizierer:

V bei Eingabe x $\in \{0,1\}^*$

1. (Formcheck) Prüfe, ob x = (X,d,L,C) mit X endliche Menge (Sei $|X|$= n), d Metrik auf X, L $\in \mathbb{N}$ und C = $(c(x_1), ..., c(x_n))$ eine Permutation auf X ist.

2. Falls $d(x_n, x_1) + \sum\limits_{k=1}^{n} d(x_k, x_{k+1}) \leq L$ akzeptiere

3. lehne ab

Laufzeit von V:

1. Dies ist in polynomieller Zeit möglich, da X und L trivialerweise in polynomieller Zeit überprüft werden können. Für C und d ist dieses auch möglich, da X endlich ist.
2. Da diese Summe endlich ist und ein Vergleich dieser mit L in polynomieller Zeit möglich ist, kann dieser Schritt in polynomieller Zeit umgesetzt werden.
3. Trivialerweise in polynomieller Zeit

Polynomilität von C:

Da C eine Permutation auf einer Endlichen Menge ist, kann dieses triviale in konstanter Zeit bezüglich $|X|$ berechnet werden.

Behautung: V ist ein Verifzierer für $mTSP$

Angenommen x $\in$ mTSP
$\implies$ Es existiert eine Rundreise, dessen Länge kleiner als L ist. $\implies$ Wähle diese Rundreise als Permutation in C $\implies$ (X,d,L,C) $\in$ L(V).
Angenommen x $\notin$ mTSP

$\implies$ Es existiert keine Rundreise, dessen Länge kleiner gleich L ist.

$\implies$ Es existiert keine Permutation, welche für C gewählt werden könnte, s.d. (X,d,L,C) $\notin$ L(V).

$\Rightarrow$ V ist ein polynomieller Verifzierer von mTSP

$\Rightarrow$ mTSP $\in$ NP $\quad\square$

Zu 2)

Sei $\emptyset\neq$X= $\{1,...,n\}$die Menge der Städte der Rundreise sind

Betrachte nun folgende Reduktionsfunktion:

$$f(w) = \begin{cases} (X, d_\Delta, L) & \text{falls w} = (\Delta, L) \\ (X, d_L, L) & \text{sonst} \end{cases}$$

Wobei Δ =(d_ij), wie auf Seite 68 im Skript definiert ist.

Behauptung: d_Δ ist eine Metrik auf X, welche durch Δ wie folgt induziert wird:

$$d_\Delta(x,y) = \begin{cases} d_{xy} & \text{mit } 1 \ \leq \text{x,y} \leq \text{n} \end{cases}$$

Beweis:

Die d_Δ ist offensichtlicherweise wohldefiniert. Nach den Eigenschaften für Δ auf Seite 68 im Skript, folgen die positiv Definitheit (die Distanz zwischen zwei Unterschiedlichen Städten kann niemals Null sein) und Symmetrie direkt. Folglich ist nur noch die Dreicksungleichung zu zeigen.
Seien x,y,z $\in$ X beliebig, aber fest. o.B.d.A x $\neq$ y, da sonst trivial

Betrachte nun d_Δ(x,z) + d_Δ(y,z).

Fallunterscheidung:

Angenommen z $\neq$ x $\vee$ y

$\implies$ d_Δ(x,z) + d_Δ(z,y) = $d_{x,z}$ + $d_{z,y}$ > $d_{x,y}$ = d_Δ(x,y), da im Raum trivialerweise die Dreicksungleichung gilt.

Angenommen z = x $\vee$ y o.B.d.A z = x

$\implies$ d_Δ(x,z) + d_Δ(y,z) = d_Δ(x,x) + d_Δ(x,y) = d_Δ(x,y)

$\implies$ d_Δ(x,y) $\leq$ d_Δ(x,z) + d_Δ(y,z) $\forall x, y, z \in X$

$\Rightarrow$ d_Δ ist Metrik.

Behauptung: d_L ist eine Metrik auf X mit

$$d_L(x,y) = \begin{cases} L & \text{falls x} \neq \text{y} \\ 0 & \text{sonst} \end{cases}$$

Beweis:

d_L ist trivialerweise wohldefiniert, positiv Definitheit und symmetrisch. Folglich ist nur noch zu zeigen, dass d_L die Dreicksungleichung erfüllt.

Seien x,y,z $\in$ X beliebig, aber fest. o.B.d.A x $\neq$ y, da sonst trivial

$\Rightarrow$ d_L(x,y) = L.

Betrachte nun d_L(x,z) + d_L(y,z).

Fallunterscheidung:

Angenommen z $\neq$ x $\vee$ y

$\implies d_L$(x,z) + d_L(y,z) = 2L > d_L(x,y)

Angenommen z = x $\vee$ y o.B.d.A z = x

$\implies d_L$(x,z) + d_L(y,z) = L = d_L(x,y)

$\implies d_L$(x,y) $\leq d_L$(x,z) + d_L(y,z) $\forall x, y, z \in X$

$\Rightarrow d_L$ ist Metrik.

Behauptung: f ist eine polynomiel berechenbare Funktion.

Es kann trivialerweise in polynomieller Zeit geprüft werden, ob eine Eingabe von der Form (Δ,L) ist.

Falls x=(Δ,L) so wird x auf (X,d_Δ,L) abgebildet. Die d_Δ kann in konstanter polynomiellerzeit bezüglich $\mathcal{O}(n^2)$ berechnet werden, da $|X|$= n und damit endlich ist.

Falls x $\neq$ (Δ,L) so wird auf (X,d_L,L) die Metrik in konstanter polynomiellerzeit bezüglich $\mathcal{O}(n^2)$ berechnet werden, da $|X|$= n und damit endlich ist.

Folglich ist f polynomiel berechenbar.

Behauptung: w $\in TSP_{ent} \iff$ f(w) $\in mTSP$

Angenommen w $\in TSP_{ent}$
d.h. w = (Δ, L) und $\exists$ Permutation auf X s.d. $d_{x_n,x_1} + \sum\limits_{k=1}^{n} d_{x_k,x_{k+1}} \leq L$.
$\implies$ f(w)= (X,d_Δ,L).

Nach Definition von d_Δ gilt:

$d_{x_n,x_1} + \sum\limits_{k=1}^{n} d_{x_k,x_{k+1}}$ =$d_\Delta(x_n, x_1) + \sum\limits_{k=1}^{n} d_\Delta(x_k, x_{k+1}) \leq L$

$\Rightarrow$ f(w) $\in mTSP$

Angenommen w $\notin TSP_{ent}$

Fallunterscheidung:

i) w $\neq$ (Δ,L)

$\Rightarrow f(w) = (X, d_L, L)$. Da X $\neq \emptyset$ hat jede Rundreise eine Distanz von $|X|$ L, weswegen f(w) $\notin mTSp$

ii) w = (Δ,L), aber w $\notin TSP_{ent}$

$\implies \forall$ Permutationen C auf x gilt $d_{x_n,x_1} + \sum_{k=1}^{n} d_{x_k,x_{k+1}} > L$

$\implies$ f(w)= (X,d_Δ,L).

Nach Definition von d_Δ gilt:

$d_{x_n,x_1} + \sum_{k=1}^{n} d_{x_k,x_{k+1}} = d_\Delta(x_n, x_1) + \sum_{k=1}^{n} d_\Delta(x_k, x_{k+1}) > L$ $\forall$ Permutationen

$\Rightarrow$ f(w) $\notin mTSp$

$\Rightarrow$ w $\in TSP_{ent} \iff$ f(w) $\in mTSP$

$\Rightarrow TSP_{ent} \leq_p mTSP \quad \square$

Da TSP_{ent} NP-Vollständig folgt nach Satz 3.6.9 mit 1 und 2):

mTsp ist NP vollständig $\quad \square$

Aufgabe 4 (Varianten von Sprachen in P und NP) (7 Punkte)
Betrachten Sie die folgende Sprache.

$$\text{Graph-}k\text{-Coloring} = \left\{ \langle G \rangle \;\middle|\; \begin{array}{c} G = (V, E) \text{ ist ein ungerichteter Graph} \\ \text{und es gibt eine Funktion } c : V \to \{1, \ldots, k\}, \\ \text{sodass } \forall \{u, v\} \in E : c(u) \neq c(v). \end{array} \right\}$$

Zeigen Sie

1. Graph-2-Coloring $\in$ P.
2. Für alle $k \geq 3$ ist Graph-3-Coloring $\leq_p$ Graph-k-Coloring.

Lösung:
1) **Z.z.: Graph-2-Coloring $\in$ P**

Fun Fact: Aus Mod/DuA ist bekannt, dass ein Graph genau dann zwei färbbar ist, wenn dieser Bipatit ist.

Sei M im folgenden eine DTM.

M bei Eingabe x $\in \{0, 1\}^*$

1. (Formcheck) Prüfe, ob x = $\langle G \rangle$

2. Führe eine Breitensuche mit folgender Modifikation aus:
   Färbe Knoten abwechselnd in einer der zwei Farben. Falls Knoten entdeckt wird, der bereits gefärbt. Überprüfe, ob die Farbe der Knoten gleich der Farbe im aktuellen Interationsschritt ist. Falls dies für einen der Knoten nicht der Fall ist lehne ab.

3. Akzeptiere

Laufzeit von M

1. Kann trivialerweise in polynomieller Zeit durchgeführt werden

2. Die Breitensuche hat eine Laufzeit von $\mathcal{O}(|V| + |E|)$ da die Modifikationen nur eine Konstanten Laufzeit bezüglich $\mathcal{O}(|V|+|E|)$ verursachen (Jeder Knoten wird nur einmal bei seiner ersten Entdeckung gefärbt. Es gibt maximal $|E|$ vergleiche von Knoten).

3. Trivial in polynomieller Zeit

Behauptung L(M) = Graph-2-Coloring

i) Graph-2-Coloring ⊂ L(M)

Sie x ∈ Graph-2-Coloring beliebig, aber fest.

d.h. x= $\langle G \rangle$ und G ist 2 färbbar.

Folglich wird x in (1) im Formcheck nicht abgelehnt.

In (2) wird G auch nicht abgelehnt, da der Graph zwei färbbar ist und durch die abwechselnde Färbung bei der Breitensuche genau eine Solche Färbung generiert wird. Also wird x nach 3 akzeptiert und es gilt: x ∈ L(M)
ii) L(M) ⊂ Graph-2-Coloring

Sei x ∉ Graph-2-Coloring, also ist x entweder von der Falschen Form wird also in (1) abgelehnt, oder ist nicht zwei färbbar. Also existiert ein Knoten, welchem durch die Breitensuche 2 unterschiedliche Farben zugeordnet werden. Folglich wird x abgelehnt.

Also gilt in allen Fällen x ∉ L(M)

$\Rightarrow L(M) = \textsc{Graph-2-Coloring} \Rightarrow \textsc{Graph-2-Coloring} \in P \quad \square$

2) Sei $k \in \mathbb{N}$ und $k \geq 3$ beliebig, aber fest.

Z.z **Graph-3-Coloring** $\leq_p$ **Graph-k-Coloring** $\forall$ k

Beweis durch vollständige Induktion über n mit n $\geq$ 3.

IA: n = 3 klar.

IV: Es gelte Graph-3-Coloring $\leq_p$ Graph-k-Coloring $\forall k \leq n$

IS: n $\implies$ n+1.

Aufgrund der IV ist nur noch: Graph-k-Coloring $\leq_p$ Graph-$k+1$-Coloring

Wir definieren die Reduktionsfunktion $f : \{0,1,\}^* \rightarrow \{0,1,\}^*$:

$$f(w) = \begin{cases} \langle G_w \rangle & \text{wenn w} = \langle G \rangle\text{, wobei } \langle G \rangle \text{ die Kodierung eines Graphen ist} \\ \langle G_0 \rangle & \text{wenn w} \neq \langle G \rangle \text{ sonst} \end{cases}$$

mit G = (V,E) und x $\notin$ V

Sei $G_w = (V_w, E_w)$ mit $V_w = V \cup \{x\}$ und $E_w = E \cup \{\{x,v\} | \forall v \in V\}$

und sei $G_0 = (V_0, E_0)$ mit $V_0 = V$ und $E_0 = E \cup \{\{u,v\} | \forall u, v \in V\}$

Laufzeit von f(W):

Zu Prüfen, ob eine Eingabe der Kodierung eines Graphen entspricht ist trivialerweise in polynomieller Zeit berechbar möglich.

Falls w = $\langle G \rangle$, so wird ein Knoten und $|V|$ Kanten hinzugefügt. Dies ist offensichtlicher weise in polynomieller Zeit möglich und berechbar.

Falls w $\neq$ $\langle G \rangle$, so werden maximal $|V|$ neue Kanten hinzugefügt. Dies ist auch in polynomieller Zeit möglich und berechbar.

$\Rightarrow$ f ist in polynomieller Zeit berechenbar.

Behauptung: w $\in$ Graph-k-Coloring $\iff$ f(w) $\in$ Graph-$k+1$-Coloring

Angenommen w $\in$ Graph-k-Coloring. o.B.d.A x $\notin$ V

$\implies$ w = $\langle G \rangle$. Also wird w auf $\langle G_w \rangle$ abgebildet. Da w $\in$ Graph-k-Coloring existiert eine Abbildung $c_k : V \rightarrow \{0, .., k\}$ s.d $\forall \{u,v\} \in E : c_k(u) \neq c_k(v)$.

Betrachte folgende Fortsetzung von c_k:

$c_{k+1} : V \to \{0, .., k, k+1\}$

$$c_{k+1}(v) = \begin{cases} c_k(v) & \text{wenn v} \in V. \\ k+1 & \text{falls v=x} \end{cases}$$

Nach Konstruktion gilt $\forall \{u, v\} \in E_w : c_k(u) \neq c_k(v)$.

$\Rightarrow$ f(w) $\in$ GRAPH-$k+1$-COLORING.

Angenommen w $\notin$ GRAPH-k-COLORING.

Fallunterscheidung
i) w $\neq \langle G \rangle$, dann wird w unter f auf $\langle G_0 \rangle$, welches trivialerweise nur 1 färbbar ist. Folglich w $\notin$ GRAPH-$k+1$-COLORING, da k $\geq$ 3

ii) w $= \langle G \rangle$, aber $\nexists c_k : V \to \{0, .., k\}$ s.d $\forall \{u, v\} \in E : c_k(u) \neq c_k(v)$.

Folglich wird w auf $\langle G_w \rangle$ abgebildet.

Angenommen w $\in$ GRAPH-$k+1$-COLORING, also muss eine Abbildung $c_k : V \to \{0, .., k\}$ s.d $\forall \{u, v\} \in E : c_k(u) \neq c_k(v)$ existieren. (*)

Da x eine Kante zu jedem Knoten in V hat muss gelten, dass x die Farbe k+1 zu geordnet werden muss. Also ist jede Abbildung c_{k+1} von der folgenden Form sein:

$$c_{k+1}(v) = \begin{cases} c_k(v) & \text{wenn v} \in V. \\ k+1 & \text{falls v=x} \end{cases}$$

wobei $c_k : V \to \{0, .., k\}$ eine Färbung von V ist. $\lightning$ (*)

$\Rightarrow$ w $\notin$ GRAPH-$k+1$-COLORING.

$\Rightarrow$ w $\in$ GRAPH-k-COLORING $\iff$ f(w) $\in$ GRAPH-$k+1$-COLORING

$\Rightarrow$ GRAPH-k-COLORING $\leq_p$ GRAPH-$k+1$-COLORING

Nachdem Induktionsprinzip folgt also:
GRAPH-3-COLORING $\leq_p$ GRAPH-k-COLORING $\forall$ k
$\square$