Graph Coloring Algorithms

Einführung

INHALT

Vorwort

Definition von Graph

Farben von dev Knoten des Graphs

Farben der Kanten des Graphs mit verschiedenen Verfahren

I Typ – Kantefärbung – edge-coloring

II Typ – f-Farben – f-coloring

III Typ – (g,f) Farben – (g,f)-coloring oder factorization

IV Typ – allgemeines Farben – Total-coloring

Grundsetzliche Begrife

Verschiedene Typen von Graphen

1. bipartite

2. planar

3. s-degenerate

4. k-tree

5. series-parallel

Kantenfärbung

Nachwort

Literatur / Quellen

Vorwort

Graph Coloring Algorithm ist ein grundlegendes Problem, das oft in verschiedene Zeitplanen brauchende Situationen kommt wie Dateiabtauschen und Computernetze. Durch Graph Coloring ist es möglich diese Prozesse genauer anzusehen und eine Lösung zu finden. Graphen sind die in der Informatik am häufigsten verwendete Abstraktion. Jedes System, welches aus diskreten Zuständen oder Objekten und Beziehungen zwischen diesen besteht, kann als Graph modelliert werden. Viele Anwendungen erfordern effiziente Algorithmen zur Verarbeitung von Graphen.

In praktischen und theoretischen Anwendungen treten Situationen auf, die durch ein Objektsystem und Beziehungen zwischen diesen Objekten charakterisiert werden können. Die Graphentheorie stellt zur Beschreibung ein Modell zur Verfügung: einen Graphen. Die problemunabhängige Beschreibung mittels eines Graphen lässt die Gemeinsamkeit von Problemen aus den verschiedensten Anwendungsgebieten erkennen. Die Graphentheorie ermöglicht somit die Lösung vieler Aufgaben, welche aus dem Blickwinkel der Anwendung keine Gemeinsamkeiten haben. Die algorithmische Graphentheorie stellt zu diesem Zweck Verfahren zur Verfügung, die problemunabhängig formuliert werden können. Ferner erlauben Graphen eine anschauliche Darstellung, welche die Lösung von Problemen häufig leicht zugänglich macht.

Wir werden einen Überblick über die folgenden Themen werfen:

1. f-coloring

2. (g,f)-coloring

3. total coloring für verschiedene Klasse von Graphs

– *bipartite Graphs*

– *series-parallel Graphs*

– *planar Graphs*

– *Graphs mit fixed degeneracy, tree widht, genus, arboricity, unicycluc index or thickness*

Definition von Graph

Graph ist ein geordnetes Paar von Knoten - Satz V und Kanten - Satz E. Bildlich kann man G darstellen, indem man seine Ecken als Knoten zeichnet und zwei dieser Knoten immer dann durch eine Linie verbindet, wenn die entsprechenden Ecken eine Kante sind. G = (V,E) sei ein Graph auf V. Für V schreiben wir auch V (G), für E auch E(G). Je nach Zusammenhang identifiziert man gelegentlich G mit V oder mit E; so schreibt man statt v V (G) oder e E(G) auch kurz v G oder e G. Zwei Kanten x, y von G sind (adjazent oder) benachbart in G und heißen Nachbarn voneinander, wenn xy E(G) ist. Zwei Kanten sind benachbart, falls sie eine gemeinsame Endknote haben.

Der Grad (oder die Valenz) dG(v) = d(v) einer Knote v ist die Anzahl |E(v)| der mit v inzidenten Kanten; dies ist gerade die Anzahl der Nachbarn von v.

Beim Graph Coloring gibt es verschiedene Möglichkeiten, wie man einen Graph färben kann: Farben von der Knoten, Farben von der Kanten und Farben den beiden.

* Farben von dev Knoten des Graphs

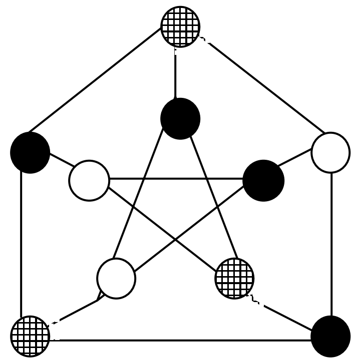
Das Ziel ist Farben von den Knoten des Graphs mit minimale Anzahl von Farben (man nennt das chromatik Zahl $\chi(G)$ von G) sodass keine benachbarte Kante mit derselben Farbe gefärbt wird.

Ein Beispiel dazu ist unten zu sehen.

Farbe 1

Farbe 2

Farbe 3



Es gibt 3 verschiedene Farben und das ist genau das Minimum Farben, mit der Graph gefärbt werden kann. In diesem Fall ist das chromatik Zahl $\chi(G)$ von G=3. Es ist öffensichtlich zu sehen, dass es keine benachbarte Kanten mit derseleber Farbe gibt und das trifft die Vorausetzung genau zu.

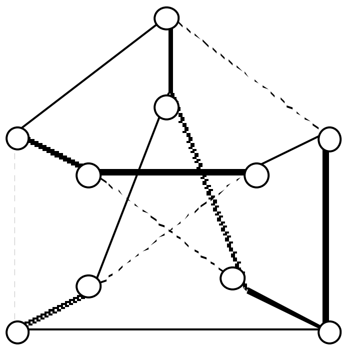
* Farben der Kanten des Graphs mit verschiedenen Verfahren

In diesem Fall gibt es mehrere Subtypen: I. edge-cobring, II. f-coloring, III. (g,f)-coloring, IV. total coloring

I Typ – Kantefärbung – edge-coloring

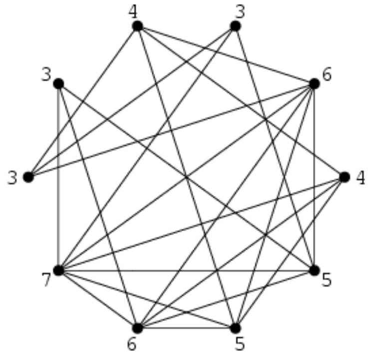
Das Ziel ist das Farben der Kanten eines Graphs mit minimale Anzahl von Farben (man nennt das chromatik Index $\chi'(G)$ von G) sodass keine benachbarte Kante mit derselber Farbe gefärbt wird.
Das unten stehende Bild zeigt ein Beispiel dafür.

Farbe 1
Farbe 2
Farbe 3
Farbe 4



Es werden 4 verschiedene Farben benutzt, deshalb ist das chromatik Index $\chi'(G)$ von G=4.

Die **maximal** Grad von Graph ist $\Delta(G)$ oder nur Δ - das ist die **maximal** Zahl von Kanten, die eine Knote berühren.



Weil alle Kanten von derselbe Knote verschiedene Farben haben sollen, deshalb ist $\chi'(G) \geq \Delta(G)$ oder4 (Farben) > 3 (Kanten)

Die min Grad von Graph ist $\delta(G)$. Für diesen Graph ist 3.

Die Graph Coloring ist NP-hard Problem, das bedeutet dass es Algorithm welches Lösung ist relativ zu anderer Lösung von NP (nondeterministic polinomial)-Problem gibt.

Es gibt aber Algorithmen, die das Problem in polynomial time lösen:

z. B. Shanon beweist, dass die Kanten jedes Graphs mit $3\Delta(G)/2$ Farben gefärbt werden können (oder $\chi'(G) \geq 3\Delta(G)/2$)

Nach Vizing ist das für den schlicht (simple) Graph richtig: $\chi'(G) \geq \Delta(G)+1$

Oder König schreibt für bipartite Graph: $\chi'(G) = \Delta(G)$

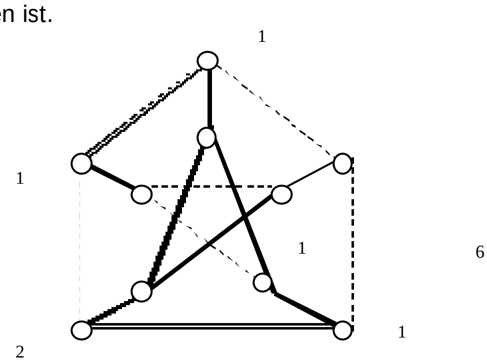
II Typ – f-Farben – f-coloring

Hier färbt man die Kanten des Graphs mit minimale Anzahl von Farben (es ist f-chromatik Index $\chi'_f(G)$ von G genannt) und gleichzeitig soll jede Farbe so viel Mal bei der Knote „v“ vorhanden sein wie f(v), wo f(v) positive Integerzahl mit „v“ gebunden ist.

Farbe 1

Farbe 2

Farbe 3



Graph Coloring Algorithms
EINFÜHRUNG

Es ist deutlich zu sehen dass die Anzahl der Farben das Resultat der Funktion f(n) entspricht. Es gibt 2 gleiche Farben, wo f(n)=2 ist und keine gleiche Farben, wo f(n)=1 ist.

** Edge-coloring- ist spezieler Typ von f-coloring, bei dem f(v)=1 für jede Knote „v“*

* Beispiel für Anwendung von diesem Typ Graph Coloring

Gebraucht als Dateiabtauschmodell an Computernetze, wo:
- die Knote des Graphs ist ein Computer
- die Kante ist eine Datei, die zwischen die beiden mit ihr gebundenen (Computer) abgetauscht werden soll
- Integer-Zahl f(v) ist die Zahl von den vorhandenen Kommunikations-Ports an der Knote (Computer)
- die gleich gefärbten Kanten sind Dateien, die gleichzeitig durch die Netz getauscht werden können.

Auf diese Weise entsprechen f-coloring und $\chi'_f(G)$ zum Planen von Datei-Abtausch mit minimale Zeit dafür.
Das Problem ist auch NP-hard.

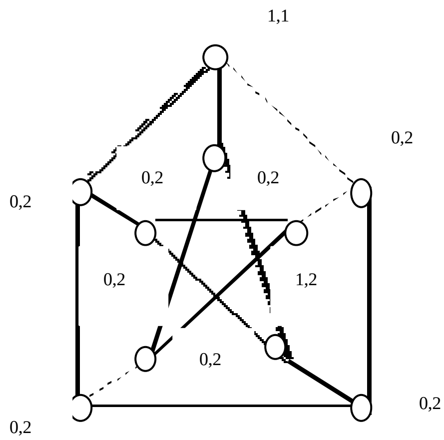
III Typ – (g,f) Farben – (g,f)-coloring oder factorization

Man färbt jede Kante des Graphs mit minimale Anzahl von Farben (wo diese minimale Anzahl ist (g,f)-chromatik Index $\chi_{gf}(G)$ von G genannt) und jede Farbe ist so viel Mal bei der Knote „v“ vorhanden wie mindestens g(v) und am meisten f(v), wo g(v), f(v) positive Integer-Zahl mit „v“ gebunden sind.

Farbe 1

Farbe 2

Farbe 3



** Edge-coloring - ist spezieler Typ von (g,f)-coloring, bei dem g(v)=0 und f(v)=1 für jede Knote „v“;*

** f-coloring ist (g,f)-coloring, wenn g(v)=0 für jede Knote „v“;*

* **Beispiel für Anwendung von diesem Typ Graph Coloring**

Gebraucht bei Bilanz-Schaffung vom Laden (engl. load) Dateiabtausch-Prozess an den Computer, wo:

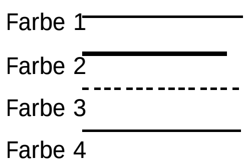
- die Knote (v von V) ist ein Computer, der f(v) Kommunikations-Ports hat und von den mindestens g(v) für Datei-Abtausch in jeder Zeit slot gebraucht werden.

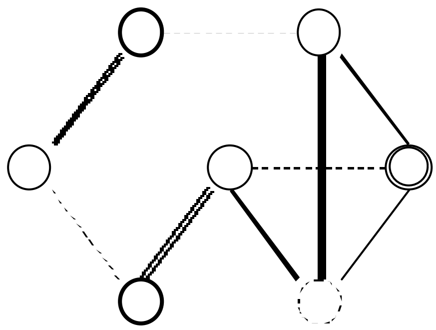
Das ist auch so ein NP-hard Problem wie edge-coloring Problem. Deshalb wird nicht erwartet, dass dieses Problem für jeden Graph gelöst werden kann. Anderseits gibt es solche Lösungen für partial k-trees, an hohe Grenze von $\chi_{gf}(G)$ und linear Algorithmen. Außerdem ist es möglich, verschiedene hohe Grenze von $\chi'_{gf}(G)$ und effiziente konsequente und parallele Algorithmen festzustellen, die (g,f)-coloring Problem für verschiedene Typen von Graphen lösen.

IV Typ – allgemeines Farben – Total-coloring

(eine Mischung zwischen Farben von Kanten und Farben von Knoten)

Das Farben ist mit minimale Anzahl von Farben realisiert, die man total chromatik Zahl $\chi_t(G)$ von G nennt, gleichzeitig soll keine benachbarte Knote mit derselber Farbe, keine benachbarten Kante mit derselber Farbe, und keine Kante mit derselber Farbe wie ihr Ende in den beiden Richtungen gefärbt werden.





Das Total-coloring Problem liegt darauf, dass vorhanden Graph mit $\chi_t(G)$ Farben gefärbt werden soll.

Es ist offensichitlich, dass $\Delta(G)+1\leq\chi_t(G)$, weil jede Kante von einer Knote verschiedene Farben haben sollen und jede Kante Knoten mit verschiedenen Farben verbinden soll.

Zum Beispiel wenn die Grad ist 3 , das bedeutet dass diese 3 Kante verschiedene Farben haben sollen, und die Knote, die sie verbindet, auch verschiedene Farbe habe soll. Das macht genau 4.

Die Idee von Total-coloring des Graphs war von Behzad und Vizing 1965 präsentiert.

Grundsetzliche Begrife

Für besser Verständnis des folgenden Materials ist es notwendig einige grundliegende Begriffe erläutert zu werden.

1. G=(V,E) V(G) und E(G) Sätze (sets)

Wie schon vorher gesagt, ist G das Graph und V(E) ist der Satz (Set) von Kanten, mit den der Satz von Knoten V(G) gebunden ist.

2. n(G) n – Zahl der Knoten im Graph

3. m(G) m – Zahl der Kanten im Graph

4. wenn m(G)=0 dann ist der Graph trivial

5. ohne multiple Kanten und ohne Self-loops schlicht Graph

6. Multigraph

7. Die Grad von Graph d(v,G) oder nur d(v)

Die Grad einer Knote v ist die Anzahl |E(v)| der mit v inzidenten Kanten; dies ist gerade die Anzahl der Nachbarn von v.

8. Die maximale Grad ist $\Delta(G)$ oder nur Δ;

Die minimale Grad ist $\delta(G)$ oder nur δ

9. Entfernen den Knoten in $V' \subseteq V(G)$ G-V'

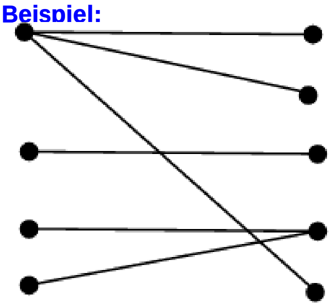
10. Entwerfen den Kanten in $E' \subseteq E(G)$ G-E'

Verschiedene Typen von Graphen

1. bipartite $G=(V_1,V_2,E)$; $e \in V_1 x V_2$ für jede $e \in E$

Die Erläuterung der oben stehende Formula ist: Ein bipartite Graph ist so in 2 Sets geteilt, so dass keine 2 Knoten von einem Set benachbart sind.

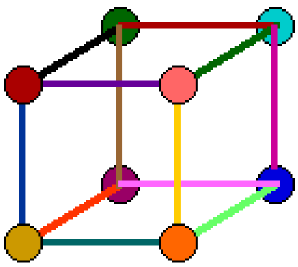
Beispiel:

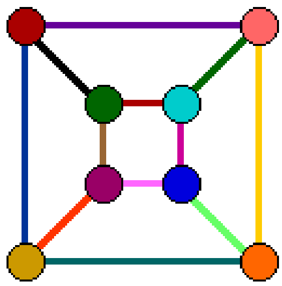


2. planar

Bei diesem Graph intersect keine 2 Kanten einander außer an dem gemeinsamen Knote

Unten ist ein interessantes Beispiel zu sehen. Das ist ein Cube, der in 3 Dimensionen steht und offensichtlich kreuzen sich die Kanten einander.
Das zweite Bild ist eine Transformation und präsentiert ein planar Grap h,





wo die Verbindungen zwischen die Knoten die gleiche sind, aber sie kreuzen sich nicht mehr.

•3. **s-degenerate** wo „s“ ist eine Integer-Zahl; $d(v_i, G_i) \leq s$ für jede i, $1 \leq i \leq n$, wo $G_i = G - \{v_1, v_2, \ldots, v_{i-1}\}$

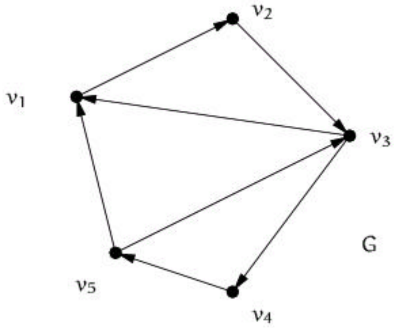
•Ein Graph ist s-degenerate nur wenn jeder von seinen Subgraphen hat eine Knote mit maximale Grad s

•degeneracy s(G) ist minimale Integer-Zahl s, für die G s-degenerate ist

•Jeder planar Graph hat $\delta(G) \leq 5$ deshalb $s(G) \leq 5$

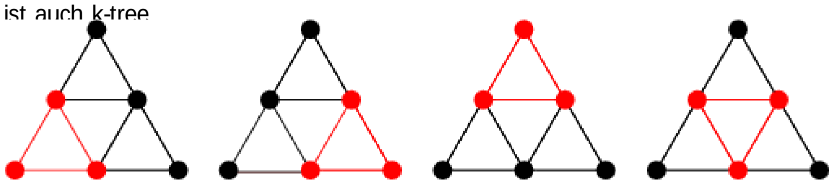
•Jeder Graph kann mit s(G)+1 Farben gefärbt werden

•trotzdem beweißt Vizing wenn $\Delta(G) \geq 2s$ dann $\chi'(G) = \Delta(G)$



4. **k-tree**

Das ist ein kompleter Graph an k Knoten oder er hat eine Knote $v \in V$ dessen Nachbarn induce Clique (kompleter Subgraph) mit Größe k und G-{v} ist auch k-tree



cliques

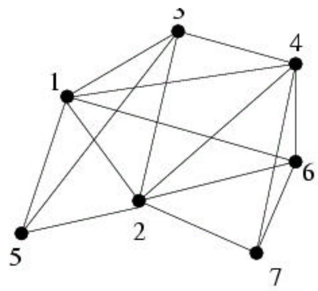


Figure 6: A k-tree with $k = 3$.

Partial k -tree ist ein Subgraph von k-tree

Tree-width k(G) von G ist die minimale Integer-Zahl k, so G ist a partial k-tree, deshalb $s(G) \leq k(G)$

Bei dem kompleten Graph sind alle Knoten mit Kanten gebunden.

5. **series-parallel**

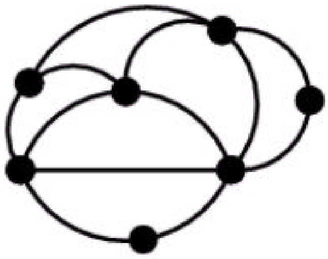
* Der Graph von einfacher Kante ist series-parallel Graph

Die Ende v_s und v_t der Kante sind terminals $v_s(G)$ $v_t(G)$

* Der Graph von 2 series-parallel Graphen G_1 und G_2 mit $v_t(G_1)$ $v_s(G_2)$ ist $v_s(G)=v_s(G_1)$; $v_t(G)= v_t(G_2)$

* Der Graph von 2 series-parallel Graphen G_1 und G_2 bestimmt $v_s(G_1)$ mit $v_s(G_2)$ und $v_t(G_1)$ mit $v_t(G_2)$ ist series-parallel multigraph dessen terminals $v_s(G)=v_s(G_1)=v_s(G_2)$ und $v_t(G)=v_t(G_1)=v_t(G_2)$ sind

$k(G)=2$ für jede series-parallel schlicht Graph



In diese Einführung werden einpaar Basisbegriffe mitgeteilt, damit die folgenden vertiefenden Beschreibungen von Graph Coloring klar sein.

Arboricity a(G)

Das ist die minimale Zahl von Subsets, in denen G geteilt werden kann

Nash-Williams beweisen $a(G)=\max_{H\subseteq G}[m(H)/(N(H)-1]$, wo H an nontrivial Subraphen läuft

$a(G) \leq s(G)$, weil Subgraph H von G ist s-degenerate und deshlab

$m(H) \leq s(G)(n(H)-1)$ und $m(H)/(n(H)-1) \leq s(G)$

wenn G planar ist, dann $a(G) \leq 3$, weil $m(H) \leq 3n(H)-3$ für jeden nontrivial Subgraph H von G

Inicyclic Index

a'(G) – ist die minimale Zahl in der G in einen Graph mit einem Cycle geteilt werden kann

Forest ist unicyclic Graph und kann in 1 oder 2 forest geteilt werden

deshalb $a'(G) \leq a(G) \leq 2a'(G)$

Thickness q(G)

$\theta(G)$ ist die minimale Zahl von planar Subgraph, in der G geteilt werden kann

$\theta(G) \leq a'(G) \leq a(G) \leq 3\theta(G)$, weil jeder unicyclic Graph planar ist und jeder planar Graph in maximal 3 forest geteilt werden kann

Genus g(G)

g(G) ist die minimale Zahl von Handle, die zu der Sphäre gefügt werden muss, so dass G auf der Resultaten Oberflache gesetzt werden kann.

Wenn G planar ist, ist $g(G)=0$

Wenn $g(G) \geq 1$, dann $\delta(G) \leq \lfloor (5+\sqrt{48g(G)+1})/2 \rfloor$

Außerdem gilt für jeden Subgraph H von G: $g(H) \leq g(G)$ wenn $g(G) \geq 1$,

dann $s(G) \leq \lfloor (5+\sqrt{48g(G)+1})/2 \rfloor$

Lemma 2.1.

für jeden nontrivial Graph:

a) $\delta(G) \leq 2a(G)-1$

b) $\delta(G) \leq 2a'(G)$

von dieser Lemma und von den oberen Beweise

$a(H) \leq a(G)$, $a'(H) \leq a'(G)$, $\theta(H) \leq \theta(G)$, $g(H) \leq g(G)$

Lemma 2.2

a) $s(G) \leq k(G)$;

b) $s(G) \leq 2a(G)-1$;

c) $s(G) \leq 2a'(G)$;

d) $s(G) \leq 6\theta(G)-1$;

e) $s(G) \leq \lfloor (5+\sqrt{48g(G)+1})/2 \rfloor$ wenn $g(G) \geq 1$;

f) $s(G) \leq 5$ wenn G planar ist.

Minor von Graph

das ist ein neur Grap h, der durch wiederholtes Entfernen von Kanten geschaft ist

Class ζ von Graphen ist minor closed wenn jeder minor von G zu ζ gehört (G∈ ζ)

Das klassische Ergebnis von Mader zeigt dass jeder Graph G in minor closed class ζ, hat a degeneracy gebunden durch Konstant h(ζ)

s(G)≤h(ζ), wo h(ζ) Konstant mit ζ gebunden ist.

Beispiel h(ζ)=5 für Class ζ von planar Graphen

Kantenfärbungen

Für jeden Graphen *G* ist offenbar ' (*G*) ≥ Δ (*G*). Für bipartite Graphen gilt hier sogar Gleichheit:

Proposition (König 1916)

Für jeden bipartiten Graphen G gilt ' (*G*) = Δ(*G*).

Beweis. Wir verwenden Induktion nach ||*G*||. Für ||*G*|| = 0 ist die Behauptung wahr. Es sei nun *G* mit ||*G*||≥ 1 gegeben, Δ := Δ (*G*), und . die Behauptung sei wahr für Graphen mit weniger Kanten. Wir wählen eine Kante *xy* *G* und eine Kantenfärbung *E*(*G* - *xy*) {1, . . . , Δ}.

Kanten der Farbe bezeichnen wir als *-Kanten*.

In G-xy haben x und y jeweils höchstens Δ-1 inzid ente Kanten.

Wir können daher , â { 1, . . . , Δ} finden, so dass x mit keiner -

Kante inzidiert und y mit keiner -Kante. Können wir überdies =

wählen, so färben wir xy mit dieser Farbe und erhalten so unsere

gewünschte Δ-Kantenfärbung von G.

Wir nehmen daher an, dass ¹ ist und x mit einer -Kante

inzidiert. Wir setzen diese Kante zu einem maximalen Kantenzug W

fort, dessen Kanten abwechselnd mit und gefärbt sind. Da kein

solcher Kantenzug eine Ecke zweimal enthalten kann, existiert W und ist

ein Weg. Weiter ist y / W: sonst würde W in y mit

einer -Kante enden (nach Wahl von) und somit gerade Länge haben,

d.h. W +xy wäre ein Kreis ungerader Länge in G .

Wir färben nun alle Kanten in W um, indem wir die Farben und auf

W vertauschen. Nach Wahl von und aufgrund der Maximalität von

W sind auch nach der Umfärbung in ganz G - xy keine gleichfarbigen

Kanten benachbart. Wir haben somit eine Δ-Kantenfärbung von G-xy,

in der weder x noch y mit einer -Kante inzident ist. Indem wir xy mit

färben, können wir diese Färbung zu einer Δ-Kantenfärbung von G

fortsetzen.

(Vizing 1964)

Für jeden Graphen G gilt

$\Delta(G) \leq \quad '(G) \leq \Delta(G)+1$.

Beweis. Wir zeigen die zweite Ungleichung mit Induktion nach $||G||$.

Für $||G|| = 0$ ist sie trivial. Zum Induktionssch ritt sei $G = (V,E)$ gegeben,

$\Delta := \Delta(G) > 0$, und die Behauptung sei wahr für Graphen mit weniger

Kanten. Statt "Kantenfärbung mit (Δ+1) Farben" sagen wir

im folgenden kurz "Färbung". Eine mit einer Farbe gefärbte Kante

nennen wir wieder eine *-Kante*.

Zu jeder Kante e G existiert nach Induktionsannahme eine

Färbung von $G - e$. Zu jeder Ecke v gibt es wegen $d(v)$ Δ dabei

eine Farbe $\{1, \ldots, \Delta+1\}$, die von keiner mit v inzidenten Kante

getragen wird; wir sagen, diese Farbe *fehlt an v*. Ist eine weitere

Farbe, so existiert dann ein eindeutig bestimmter maximaler in v

beginnender Kantenzug (möglicherweise trivial), dessen Kanten

abwechselnd mit und gefärbt sind. Dieser Kantenzug ist ein Weg,

und wir nennen ihn den */â -Weg aus v*.

Wir nehmen an, G habe keine Färbung. Dann gilt:

Ist xy E und eine Färbung von G-xy gegeben, in der die Farbe an x und die Farbe an y fehlt, so endet der /â -Weg aus y in x. *(1)*

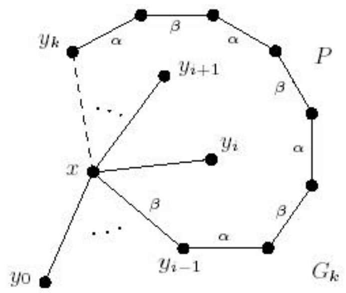
Anderenfalls könnten wir die Kanten des Weges umfärben (durch Vertauschung der Farben und) und xy mit färben; wir hätten dann eine Färbung von G, entgegen unserer Annahme.

Es sei xy_0 G eine Kante; nach Induktionsannahme hat $G_0 := G$-x, y_0 eine Färbung c_0. Es sei eine darin an x fehlende Farbe. Weiter sei y_0, y_1, . . . , y_k eine maximale mit y_0 beginnende Folge verschiedener Nachbarn von x in G, so dass in c_0 jeweils $c_0(xy_i)$ an y_{i-1} fehlt (i =1, . . . , k). Auf jedem der Graphen $G_i := G - xy_i$ definieren wir eine Färbung c_i durch

$c_i(e) := c_0(xy_j+1)$ für $e = xy_j$ mit j { 0, . . . , i-1 } *oder* $c_0(e)$ sonst;

beachte, dass in jeder dieser Färbungen die gleichen Farben an x fehlen wie in c_0.

Es sei nun eine in $c0$ an y_k fehlende Farbe. Natürlich fehlt dann auch in c_k an y_k. Fehlte auch an x, so könnten wir c_k zu einer Färbung von ganz G ergänzen, indem wir xy_k mit färbten. Die Ecke x ist also (in jedem c_i) mit einer -Kante inzident. Wegen der Maximalität von k gibt es daher ein i { 1, . . . , k -1 } mit i $c_0(xy_i)$ = . *(2)*



Der /â -Weg P in G_k

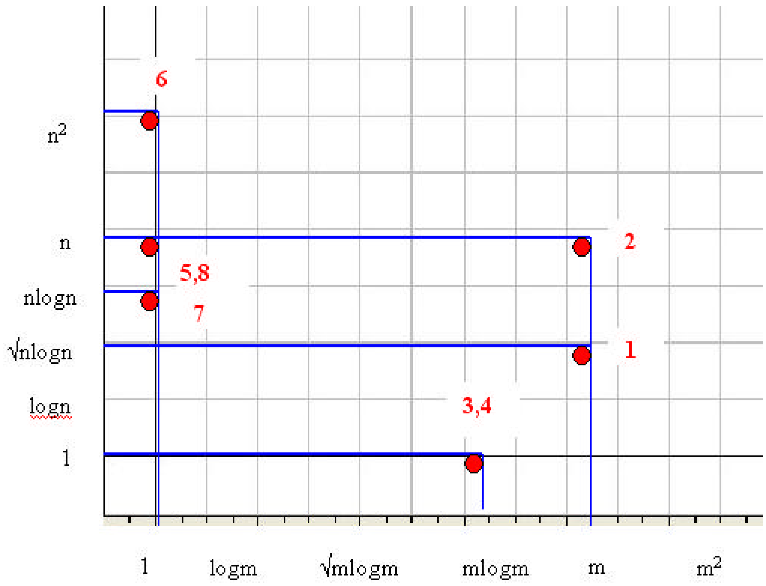
Es sei P der /â -Weg aus y_k in G_k (bezüglich c_k). Nach (1) endet P in x – und zwar mit einer -Kante, da an x fehlt.

Wegen $= c_0(xy_i) = c_k(xy_{i-1})$ ist dies die Kante xy_{i-1}. In c_0, und somit auch in c_{i-1}, fehlt jedoch an y_{i-1}; es sei P' der /â -Weg aus P' y_{i-1}in Gi_{i-1} (bezüglich c_{i-1}). Da P' eindeutig bestimmt ist, durchläuft P' zuerst $y_{i-1}Py_k$; beachte, dass die Kanten von $P°x$ in c_{i-1} genauso gefärbt sind wie in c_k. In c_0, und daher auch in c_{i-1}, ist y_k jedoch mit keiner -Kante inzident (nach Wahl von). Somit endet P' in y_k, im Widerspruch zu (1).

Der Satz von Vizing teilt die Graphen hinsichtlich ihres chromatischen Indexes in zwei Klassen ein; Graphen G mit $'(G) = \Delta(G)$ bezeichnet man häufig als Graphen der *Klasse 1* , Graphen G mit $'(G) = \Delta(G)+1$ als Graphen der *Klasse 2* .

SEQUENTIAL ALGORITHMS FOR EDGE-COLORING

1. SIMPLE GRAPH O(m nlogn)
2. MULTIGRAPH O(m (D+n))
3. BIPARTITE MG. O(mlogm)
4. SERIES-PARALLEL MG O(mlogm)
5. PARTIAL K-TREE O(n)
6. PLANAR GRAPH Δ≥8 O(n^2)
7. PLANAR GRAPH Δ≥9 O(nlogn)
8. PLANAR GRAPH Δ≥19 O(n)



Nachwort

Die Vorteile von Graphen bei der Beschreibung und Lösung von Topologie- und Optimierungsproblemen, die durch die schnellen Fortschritte in der Graphentheorie seit den fünfziger Jahren entstanden, konnten erfolgreich für praktische Anwendungen genutzt werden. Aus diesem Grund werden immer mehr grundlegende Aufgabenstellungen im Ingenieurwesen auf die abstrakten Problemstellungen der Graphentheorie zurückgeführt.

Ein typisches Beispiel für die Anwendung von Graphen im Ingenieurbereich ist die Suche nach dem kürzesten Weg zwischen zwei Elementen innerhalb eines Graphen. Hiermit lassen sich etwa Bestwegrouten in Verkehrsnetzen, der minimale Verbrauch von Kabeln zur Versorgung eines Hochhauses oder die effizienteste Nutzung von Industrierobotern zum Löten von Leiterplatinen bestimmen. Mit Graphen lässt sich das optimale Beladen eines Güterfahrzeugs ebenso gut ermitteln, wie der günstigste Ablaufplan für den Bau eines großen Gebäudekomplexes oder die Konfliktflächen aufeinandertreffender Fahrzeugströme an Verkehrsknotenpunkten.

Literatur / Quellen

1. Graph Coloring Algorithms Xiao Zhou, Takao Nishizeki
2. http://www.cs.sunysb.edu/~algorith/files/edge-coloring.shtml
3. http://www.utm.edu/departments/math/graph/glossary.html
4. http://www.utc.edu/~cpmawata/petersen/index.htm
5. http://mathworld.wolfram.com/NP-HardProblem.html
6. Reinhard Diestel. Graphentheorie. 2. Auflage. Springer-Verlag
7. www.math.tu-clausthal.de/~mawh/AlgoGraph/
8. **Graphentheorie** H. Albrecher C. Heuberger Institut für Mathematik