I n h a l t:

Gedanken zum Vierfarbenproblem

Das Vierfarbenproblem wurde im Jahre 1852 von einem englischen Studenten als schwierige Aufgabe entdeckt. Er wollte eine Landkarte der englischen Grafschaften mit vier Farben so einfärben, dass aneinander grenzende Gebiete nie dieselbe Farbe bekommen. 1878 wurde es in der Londoner Mathematischen Gesellschaft vorgestellt und verallgemeinert. Die Vierfarbenvermutung besagt, dass für jede beliebige Landkarte vier Farben ausreichend sind. Sie wurde 1976 von Haken und Appel durch Computerberechnungen bestätigt. Ein bisher wenig bekannter Beweis wurde von dem englischen Mathematiker George Spencer Brown in seinem Buch "Laws of Form" 1969 (dt. 1997) erbracht.
Hier sind meine Überlegungen zu diesem Thema.

Das Modell

Eine Kante eines Graphen begrenzt zwei benachbarte Flächen. Ein Knoten begrenzt mindestens zwei Kanten. Bei 4 Farben A,B,C,D wird jede Kante mit einem 2-Tupel der Farben bezeichnet; z.B. AB. Es gibt sodann 16 Tupel bzw. 10 symmetrisch gleichwertige Tupel (AB = BA). Beim Vierfarbenproblem sind die 4 Tupel AA, BB, CC, DD unzulässig. Somit verbleiben 6 verschiedene Kanten AB/BA, AC/CA, AD/DA, BC/CB, BD/DB, CD/DC.

Zum Beweis des Vierfarbenproblems ist zu zeigen, dass es bei Einfärbung einer beliebigen Fläche für jede Nachbarfläche mindestens eine Lösung gibt.

Fallunterscheidung nach dem Grad eines Knotens:

1. Knoten mit 2 Kanten: Sonderfall für beliebig geformte Flächen.

   Bei Vorgabe des Farbtupels einer Kante ist die andere Kante eindeutig bestimmt, d.h. identisch. Es gibt nur eine Nachbarfläche.

2. Knoten mit 3 Kanten: häufiger Fall für geographische Muster.

   Bei Vorgabe des Farbtupels einer Kante gibt es für die nächste Kante 2 Lösungen, für die dritte Kante 1 Lösung.

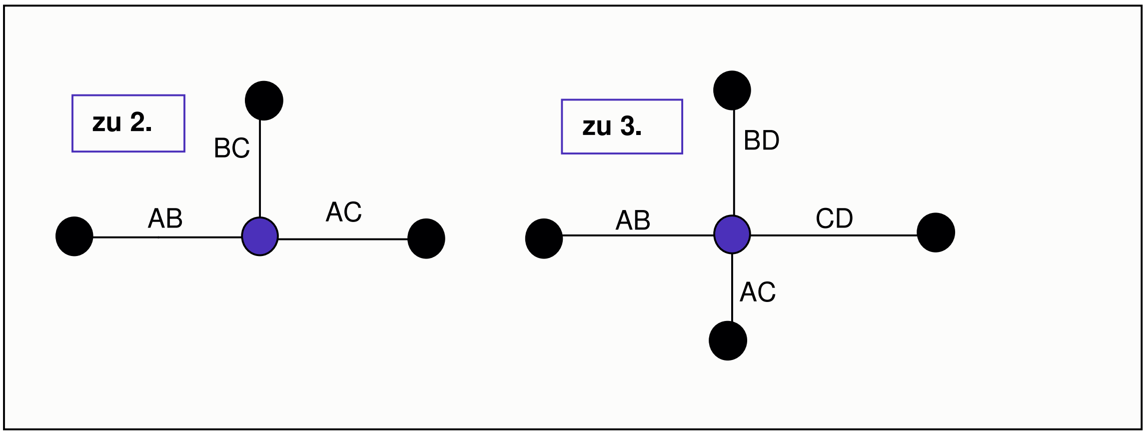
3. Knoten mit 4 Kanten: häufiger Fall für geometrische Muster.

   Bei Vorgabe des Farbtupels einer Kante gibt es für die gegenüberliegende Kante 1 Lösung bzw. ihr Spiegelbild. Die anderen Kanten sind dann eindeutig bestimmt.

4. Knoten mit mehr als 4 Kanten:

   <u>Es gibt keine Lösung mit 4 Farben!!</u>

Folgerung: Die Vierfarbenvermutung ist also nur zutreffend, wenn für benachbarte Flächen nicht nur ein gemeinsamer Punkt, sondern gemeinsame <u>Kanten</u> gefordert sind. Die maximale Anzahl der Kanten (Grad) eines Knotens der Landkarte bestimmt demnach die Anzahl der notwendigen Farben. Mit n Farben können $(n^2\text{-}n)/2$ verschiedene, zulässige Kanten gebildet werden.



Für 3 Farben A,B,C - das sind 3 verschiedene Kanten AB/BA, AC/CA, BC/CB - gilt:

1. Jeder Knoten darf max. 3 Kanten haben. Bei Vorgabe der Farbtupel einer Kante sind die anderen Kanten bis auf Symmetrie eindeutig bestimmt.
2. Jede innen liegende Fläche muss eine gerade Anzahl von einzufärbenden Nachbarflächen haben. Das ist die Summe der Kanten minus die Summe der Knoten einer Fläche.

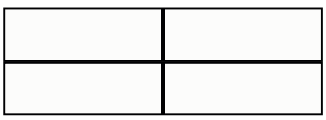
Es gibt in England einen Punkt im äußersten Südosten, an dem sich vier Grafschaften sehr nahe kommen, evtl. treffen (Westsussex, Eastsussex, Kent, Surrey)!
Es gibt mindestens eine Grafschaft, die von einer ungeraden Anzahl von Grafschaften umrahmt ist.

Zur Einfärbung der englischen Grafschaften sind also mindestens 4 Farben notwendig.

post scriptum

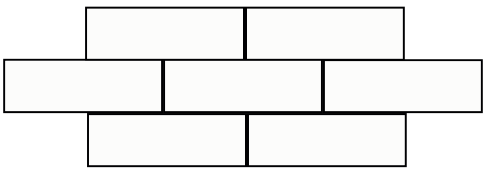
Die entscheidende Voraussetzung scheint mir zu sein, ob Flächen durch einen Punkt - also z.B. "4-Länderecke", wie in meiner Überlegung angenommen - oder durch mehrere Punkte, d.h. Kanten - also maximal 3-Länderecke - getrennt sind oder sein dürfen. Bei trennenden, bzw. gemeinsamen <u>Kanten</u> ist jede Kante eines Knotens zu den beiden anderen Kanten benachbart, um jeweils eine Fläche zu bilden. Trennung durch einen einzigen Punkt, wodurch beliebig viele Flächen in einem Punkt aufeinander treffen können, ist besonders bei geometrischen Mustern wie rechteckigen oder polygonen Kacheln die Regel.

Muster mit 4-Ländereck



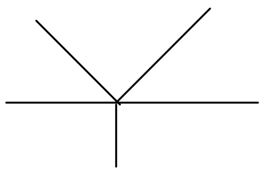
Dieses Muster erfordert 4 Farben

Muster mit 3-Ländereck



hier genügen 3 Farben

mehr als 4 Kanten



mehr als 4 Farben

Das (3n+1)-Problem, Ulam-Folge

Man nehme eine beliebige Zahl $n_0>1$, multipliziere sie mit 3 und addiere 1. Die Summe wird sooft wie möglich durch 2 dividiert. Mit diesem Ergebnis n_1 wird von vorne begonnen. Die bisher unbewiesene Hypothese ist nun, dass man nach endlich vielen Schritten immer 1 als Endergebnis erhalten wird. Der Algorithmus stammt von dem amerikanischen Mathematiker Stanislaw Ulam (*1909 Lemberg +1984). Hier sind einige unvollendete Überlegungen von mir zur Beweisführung.

Der Algorithmus als Formel: $3n_i + 1 = 2^{k_i} n_{i+1}$ oder $n_{i+1} = 2^{-k_i} (3n_i + 1)$

1. Teilbeweis

es wird bewiesen, dass
$3n - 1 = 2^k$ für ungerade k
$3n + 1 = 2^k$ für gerade k

In einem Intervall $[\, 2^k - 1, 2^k, 2^k +1 \,]$ gilt immer:
$2^k \neq 3n$ und entweder $2^k + 1 = 3n$ oder $2^k - 1 = 3n$.
Daraus folgt $(2^k - 1)(2^k +1) = 3n$ oder $2^{2k} - 1 = 3n$ oder $2^{2k} = 3n + 1$ für alle $k>0$.

Es gibt also zu jedem k ein n, so dass $3n + 1 = 2^{2k}$. Das ist der letzte Teilschritt im Algorithmus.

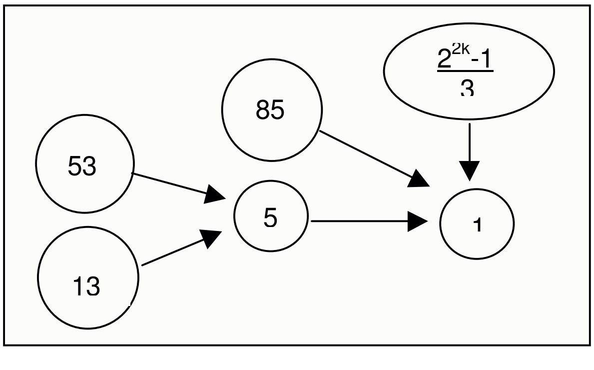
2. Teilbeweis

Aus der Formel $3n_i +1 = 2^{k_i} n_{i+1}$ folgt:

a) zu jedem n_i gibt es genau ein n_{i+1}
b) zu jedem n_{i+1} gibt es mindestens ein k_i mit zugehörigem n_i; folgt aus 1. Teilbeweis.
c) $n_i \neq n_{i+1}$ für $n_i > 1$
d) jedes n_{i+1} ist weder durch 2 noch durch 3 teilbar, d.h. $n_{i+1} = 6m +1$ oder $n_{i+1} = 6m - 1$.
e) $k_i > 0$

Daraus ergibt sich eine invers-hierarchische Struktur.

Beispiel:



Damit ist intuitiv plausibel, dass die Hypothese richtig ist. Im 3. Teilbeweis bleibt noch zu zeigen, dass $n_i \neq n_{i+m}$ ist, um Zyklen auszuschließen. Schon ein einziger Zyklus würde die Hypothese falsifizieren.

3. Teilbeweis

Fortführung des Algorithmus führt zu folgenden Formeln:

aus $(3n_i+1)\cdot 2^{-k} = n_{i+1}$ folgt für n_{i+2}: $(3n_i+1)\cdot 2^{-k}\cdot 3+1 = 2^m n_{i+2}$

und allgemein: $n_i\cdot 3^m\cdot 2^{-(k+l\ \ldots)}+3^{m-1}\cdot 2^{-(l+\ \ldots)}\ \ldots + 2^{-z} = n_{i+m}$

Alle Exponenten von 2 sind negativ. Daher ist die Gleichung für $n_i = n_{i+m}$ für ganze Zahlen nicht lösbar und $n_i \neq n_{i+m}$, was zu zeigen war.

Dieser Teilbeweis ist zugegebenermaßen etwas waghalsig und bedarf noch näherer (fachkundigerer) Überprüfung!

Anwendung und Bedeutung:

Eine praktische Anwendung ist mir bisher nicht bekannt geworden, außer als Programmierübung rekursiver Funktionen für Schüler und Studenten.
Der Kybernetiker Heinz v. Foerster hat das Konzept der nichttrivialen Maschinen entwickelt. Dies sind Systeme, wie biologische Organismen oder abstrakte Automaten, deren Output nicht nur vom Input, sondern zusätzlich von ihrem inneren Zustand abhängig ist. In seinem Konzept „order from noise“ beschreibt er Systeme, deren Output immer gleich, also vom Input unabhängig ist, während der innere Zustand in Abhängigkeit vom Input bis zu einem stabilen Endzustand transformiert wird. In der Sprache der Chaostheorie ein punktförmiger Attraktor oder in der Thermodynamik der Zustand der höchsten Entropie. Genau dafür kann die Ulamfolge ein Modell sein. Als konkrete Beispiele aus der Biologie sind die Zellteilung und das Immunsystem zu nennen.

Eine Ausführung des Algorithmus mit dem Computer hat für beispielhafte n_0 folgende interessante Resultate für die **Anzahl der Schritte** ergeben. Bemerkenswert ist die Clusterung gleicher Schrittzahlen. Ursache dafür sind vermutlich die ähnlichen Muster der Binärdarstellungen von n_0: die Interpretation $3n+1 = 2n +1 + n$ ergibt einen interessanten Ansatz für eine binäre Lösung.

Für $1 < n_0 <= 1$ Million ist die maximale Anzahl der Schritte = 178 bei $n_0 = 808984$.

10 Mill. $<= n_0 <=$ 10 Mill. + 100 und
100 Mill. $<= n_0 <=$ 100 Mill. + 100

10000000: 35	10000013: 59	10000026: 100
10000001: 35	10000014: 124	10000027: 54
10000002: 35	10000015: 124	10000028: 59
10000003: 44	10000016: 54	10000029: 54
10000004: 44	10000017: 54	10000030: 54
10000005: 54	10000018: 54	10000031: 100
10000006: 54	10000019: 54	10000032: 100
10000007: 42	10000020: 124	10000033: 124
10000008: 42	10000021: 59	10000034: 54
10000009: 76	10000022: 54	10000035: 54
10000010: 124	10000023: 59	10000036: 21
10000011: 54	10000024: 54	10000037: 54
10000012: 42	10000025: 100	10000038: 54

10000039: 54
10000040: 124
10000041: 100
10000042: 10
10000043: 54
10000044: 35
10000045: 35
10000046: 35
10000047: 76
10000048: 35
10000049: 59
10000050: 35
10000051: 59
10000052: 54
10000053: 59
10000054: 59
10000055: 35
10000056: 54
10000057: 59
10000058: 124
10000059: 59
10000060: 59
10000061: 59
10000062: 35
10000063: 35
10000064: 59
10000065: 59
10000066: 124
10000067: 54
10000068: 54
10000069: 44
10000070: 35
10000071: 59
10000072: 54
10000073: 44
10000074: 35
10000075: 44
10000076: 44
10000077: 44
10000078: 54
10000079: 100
10000080: 44
10000081: 44
10000082: 59
10000083: 44
10000084: 100
10000085: 59
10000086: 59
10000087: 59
10000088: 76
10000089: 100
10000090: 44
10000091: 44
10000092: 100
10000093: 100
10000094: 76
10000095: 100
10000096: 100
10000097: 100
10000098: 54
10000099: 59
10000100: 100

100000000: 43
100000001: 43
100000002: 55
100000003: 43
100000004: 43
100000005: 31
100000006: 72
100000007: 20
100000008: 20
100000009: 62
100000010: 5
100000011: 31
100000012: 31
100000013: 31
100000014: 62
100000015: 43
100000016: 43
100000017: 31
100000018: 43
100000019: 72
100000020: 72
100000021: 31
100000022: 13
100000023: 43
100000024: 31
100000025: 43
100000026: 5
100000027: 8
100000028: 31
100000029: 43
100000030: 6
100000031: 43
100000032: 43
100000033: 31
100000034: 6
100000035: 43
100000036: 23
100000037: 43
100000038: 31
100000039: 43
100000040: 8
100000041: 43
100000042: 5
100000043: 43
100000044: 11
100000045: 31
100000046: 10
100000047: 96
100000048: 72
100000049: 72
100000050: 72
100000051: 72
100000052: 96
100000053: 55
100000054: 72
100000055: 55
100000056: 6
100000057: 55
100000058: 5
100000059: 31
100000060: 31
100000061: 31
100000062: 43
100000063: 6
100000064: 62
100000065: 55
100000066: 55
100000067: 55
100000068: 55
100000069: 55
100000070: 72
100000071: 62
100000072: 55
100000073: 120
100000074: 5
100000075: 55
100000076: 120
100000077: 31
100000078: 8
100000079: 62
100000080: 31
100000081: 31
100000082: 55
100000083: 62
100000084: 8
100000085: 31
100000086: 55
100000087: 55
100000088: 55
100000089: 55
100000090: 5
100000091: 55
100000092: 31
100000093: 31
100000094: 6
100000095: 55
100000096: 13
100000097: 55
100000098: 6
100000099: 31
100000100: 55

Nebenbemerkung: für 3n-1 ist die Hypothese nicht zutreffend, weil hierbei die im 1. Teilbeweis angeführte Eigenschaft zu einem Widerspruch führt.
Beispiel: n_i = 5 führt zu einem Zyklus 5-7-5.

Der kleine Fermatsche Satz

Pierre de Fermat (1601 - 1665) hat im Jahr 1640 in einem Brief eine Behauptung aufgestellt, die später als kleiner Fermatscher Satz weltberühmt geworden ist. Sie lautet

$a^{p-1} -1$ ist teilbar durch p, wenn p Primzahl und nicht Teiler von a ist.

Anders formuliert lautet die Aussage **$a^{p-1} \equiv 1 \pmod p$**.
Eine äquivalente Form ist $a^p \equiv a \pmod p$.

Der Satz hat praktische Bedeutung für die Prüfung von Primzahlen und in der Informatik für Verschlüsselungssysteme.

Hier soll die Form $a^p \equiv a$ unter die Lupe genommen werden. a^p ist in der Kombinatorik die Anzahl von Variationen mit Wiederholung, also die Bildung von p-Tupeln aus einer Menge mit a Elementen. Es ist beispielsweise die Anzahl der Zahlen mit p Ziffern zur Basis a oder die Anzahl von Wörtern der Länge p aus einem Alphabet mit a Buchstaben. Diese Anzahl kann allerdings auch anders berechnet werden. Ein Beispiel für a^3:

a^3 = (a über 3)3! + 2(a über 2)3!/2!+ a.

Für a=10 ist die Summe logischerweise gleich 1000.
Diese Formel entstammt der Interpretation einer Variation als Verknüpfung von Kombination und Permutation. Das erste Summenglied ist die Anzahl von Variationen ohne Wiederholung, das zweite Glied ist die Anzahl der Wiederholungen von jeweils zwei Elementen und das dritte Glied ist die Anzahl aus der Wiederholung aller drei Elemente; eigentlich (a über 1)3!/3!=a. Man kann sich auch ein p-dimensionales, endliches Koordinatensystem der Länge a vorstellen, in dem man die Koordinaten aller möglichen Punkte berechnet. Das letzte Glied ist dann die Diagonale durch alle Dimensionen. Die Formel ist erweiterbar und verallgemeinerbar für höhere Potenzen, die dann weitere Zwischenglieder für die möglichen Wiederholungen enthält, während das letzte Glied immer gleich a ist.

Für a^4 lautet die Formel:

a^4 = (a über 4)4! + 3(a über 3)4!/2! + 2(a über 2)4!/3! + (a über 2)4!/2!2! + a

Entscheidend ist, dass alle Glieder ausser dem letzten den Faktor p! haben. Das heißt, die Summe ist kongruent zu a mod p, was nichts anderes als die Behauptung von Fermat ist. Als Nebensatz gilt, dass in dem Ausdruck (a über b) der Zähler immer durch b teilbar ist.

Noch eine Eigenschaft von a^n: $a^n-1 = (a-1)(a^{n-1} + a^{n-2} + ... a^0)$ oder $a^n \equiv 1 \pmod{a-1}$ für beliebige n.
Beispiel: $10^4 - 1 = 9999 = 9*1111$.

Eine weitere Interpretation von a^n ist die Zählung aller Punkte in einem n-dimensionalen Gitternetz mit Kantenlänge a. Für den 2-dimensionalen Fall, also a^2, kommt man so auf die Formel

a^2 = (a über 2)2! + a = a + 2∑i für i=1 bis i=a-1.

Auch hier gilt also der kleine Fermatsche Satz. In der Form a = a^2 - 2∑i für i=1 bis i=a-1 lässt sich aus dieser Formel unschwer ein Algorithmus zum Wurzelziehen bzw. zur Prüfung von Quadratzahlen ableiten.
in Pseudocode: i = 1; q = a^2; repeat q = q - 2i; ++i until q < 0; a = i;
Eine Zahl ist Quadratzahl, falls q+i = 0 nach dem letzten Schritt.

Das Problem des Handlungsreisenden oder "travelling salesman problem".

Ein Handlungsreisender sucht die kürzeste Route, um alle vorgesehenen Orte möglichst nur einmal zu besuchen. Dieses Problem gehört zur Klasse der sogenannten NP-vollständigen Probleme der Komplexitätstheorie; d.h. die Anzahl der Möglichkeiten steigt exponentiell mit der Anzahl der Orte und es ist nicht mit polynomialem Aufwand lösbar. Ausserdem kann es mehrere Lösungen geben, da jede Summe aus verschiedenen Summanden zusammengesetzt sein kann. Hier wird ein Näherungsalgorithmus vorgestellt mit Aufwand $O(n^3)$*, der das Problem lösen kann. Der Algorithmus ist so einfach, dass er leicht zu einer Blamage für den Autor werden könnte angesichts der Mindpower, die bereits in dieses Problem investiert worden ist. Die Bedeutung des Problems und seiner Lösung für Mathematik und Kybernetik ist dieses Risiko allerdings wert, zumal erste einfache Experimente erfolgreich waren. Der Beitrag ist daher selbst als Experiment zu betrachten.

____

*$O(n^3)$ ist eine in der Komplexitätstheorie übliche Bezeichnung für den Aufwand von Algorithmen. In diesem Fall steigt der Aufwand kubisch mit der Anzahl der Orte.

gegeben: eine Anzahl von Orten mit den Entfernungen zwischen den Orten. Bei n Orten gibt es $(n^2\text{-}n)/2$ Kanten und (n-1)!/2 richtungsunabhängige Routen.
gesucht: eine der kürzesten Routen von einem Ort zurück zum Ausgangsort, wobei jeder Ort nur einmal besucht wird.

Modell: ein vollständiger, ungerichteter, nach der Entfernung gewichteter Graph.

Algorithmus:

1. man ordnet alle Kanten als Folge an, aufsteigend sortiert nach ihrem Gewicht (Entfernung). Es gibt $(n^2\text{-}n)/2$ Kanten. Der Aufwand ist $O(n^2)$.

2. man streicht aus der Folge nacheinander die längste Kante, unter der Bedingung, dass jeder Knoten erhalten bleibt und dass für jeden Knoten zwei Kanten erhalten bleiben. Da es im Falle eines vollständigen Graphen für jeden Knoten n-1 zulässige Kanten gibt, darf jeder Knoten genau (n-3)-mal gestrichen werden. Der Aufwand für diesen Schritt ist $O(n^2)$.

3. Im günstigen Fall ist der Algorithmus hier beendet. Bei ungünstiger Verteilung der Gewichte dagegen können ein oder mehr Knoten mit mehr als zwei Kanten übrigbleiben. In diesem Fall wird die längste, ungelöschte Kante zu diesem Knoten an das Ende der Folge mit den größten Gewichten gesetzt und der Algorithmus der Streichungen wiederholt. Der Aufwand für diesen Schritt ist $O(n^3)$, weil die Kanten bis zu n-mal durchsucht werden müssen. Da es möglich ist, dass anstatt einer einzigen zusammenhängenden Route separate Teilrouten gebildet werden, muss auch darauf überprüft werden und der Algorithmus nach erneuter Umsortierung der Kanten wiederholt werden. Dazu wird in diesem Fall die längste, ungelöschte Kante wie oben an das Ende der Folge gesetzt.

Begründung:

Die verbleibenden Kanten bilden eine Teilmenge, so dass für jeden Knoten zwei Kanten enthalten sind und die Summe der Gewichte minimal ist. Die kürzeste Länge der Route ergibt sich definitiv aus der sortierten Anordnung der Folge. Da jeder Knoten in der

Teilmenge vertreten und an zwei Kanten gebunden ist, ergibt sich ein zusammenhängender Graph als Kreis und als gesuchte Route.

Fazit:

Der Algorithmus wurde als Java-Programm realisiert und mit Testdaten von bis zu 22 Städten erfolgreich ausgeführt. Dabei kamen plausible und auch überraschende Ergebnisse zustande, so z.B. dass die kürzeste Kante nicht in der Route vertreten sein muss. Da jede Vereinfachung gegenüber der "brute-force"-Methode ihren Preis fordert, sind auch hier Abstriche hinzunehmen. So kann bisher nicht garantiert werden, dass der Algorithmus in jedem Fall terminiert. Im Experiment hat sich gezeigt, dass bei kurzen Entfernungen nicht immer das Optimum gefunden wird, weil die großen Entfernungen viel stärker in die Summe eingehen als die kurzen. Aus diesem Grund sind subtraktive Algorithmen effektiver als additive, d.h. dass lange Kanten gelöscht werden anstatt kurze Kanten hinzuzufügen, besonders dann, wenn suboptimale Lösungen ausreichend sind.

Probleme können sich ergeben, wenn mehrere Kanten gleicher Länge vorhanden sind. In diesem Fall werden die Knoten in die Sortierung einbezogen. Im Experiment hat sich gezeigt, dass einzelne, weitab liegende Knoten zu Endlosschleifen bzw. zur erfolglosen Beendigung des Programms führen können, weil ihre langen Kanten frühzeitig gelöscht werden und dadurch voreilige Entscheidungen für andere Knoten getroffen werden.

Das Routenproblem und die Geografie.

Geografische Orte und ihre Entfernungen dazwischen unterliegen den Gesetzen der Geometrie. Zur Vereinfachung kann man sich auf die 2-dimensionale Ebene beschränken. Drei Knoten bilden mit ihren Kanten ein eindeutig bestimmtes Dreieck. Jeder hinzukommende Knoten ist mit drei Kanten ebenfalls eindeutig bestimmt. Alle seine weiteren Kanten sind Redundanz. Daraus läßt sich ein spezieller Algorithmus ableiten.

1. man bildet aus drei beliebigen Knoten das Dreieck als erste Route.

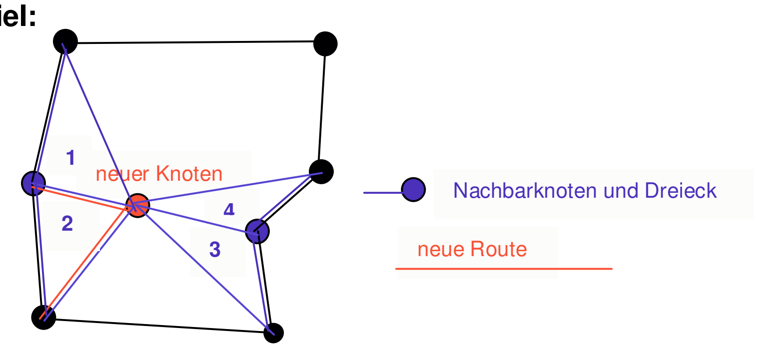
2. man nimmt einen weiteren Knoten hinzu und bestimmt seine zwei kürzesten Kanten zu Knoten in der bisherigen Route und die beiden begrenzenden Knoten als Nachbarknoten.

3. man bestimmt die drei bzw. vier Dreiecke, die von dem neuen Knoten, seinen beiden Nachbarknoten und deren Nachbarknoten in der bisher festgelegten Route gebildet werden. Die Anzahl der Dreiecke wird dadurch bestimmt, ob die beiden Nachbarknoten eine gemeinsame Kante haben oder nicht.

4. man bestimmt nun die günstigste Routenvariante, die sich aus den Dreiecken durch Hinzunahme des neuen Knotens bilden läßt. Eine Kante der bisherigen Route wird durch zwei Kanten des neuen Knotens ersetzt.

5. man fährt mit 2. fort bis alle Knoten integriert sind.

Beispiel:



Offenkundig wächst dieser Algorithmus nicht mehr exponentiell, da für jeden neuen Knoten nur noch maximal sechs Kanten aus n^2 Kanten bzw. vier Knoten aus n Knoten zu prüfen sind. Verbesserungen ergeben sich, wenn man statt zwei Dreiecken alle Dreiecke prüft. Weitere Optimierungen in Form von Glättungen sind möglich und notwendig. Ein Vorteil dieses Algorithmus ist, dass auch Routen mit Teilmengen der Knoten berechnet werden können.

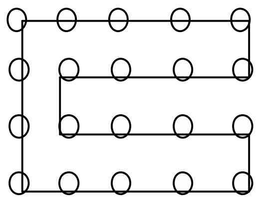
Der Algorithmus ist in Java realisiert und bringt gute und plausible Ergebnisse, allerdings mit vereinzelten Brüchen, die auf die Reihenfolge der abzuarbeitenden Knoten zurückzuführen sind.

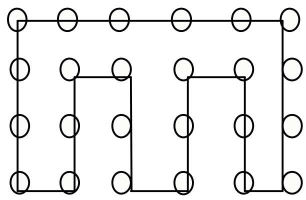
Grundsatzüberlegung 1:

Wählt man ein beliebiges Dreieck als Bezugsdreieck, dann ist jeder der übrigen Knoten des Graphen durch seine Kanten mit den drei Ecken des Bezugsdreiecks eindeutig bestimmt. Alle weiteren Kanten sind Redundanz, sofern sie den Bedingungen euklidischer Ebenen, insbesondere der Dreiecksungleichung, genügen. Insgesamt gibt es bei n Knoten n(n-1)(n-2)/6 - d.h. n über 3 - Dreiecke, weshalb die Vermutung nahe liegt, dass das Problem mit $O(n^3)$ vollständig lösbar ist und folglich nicht mehr zu NP gehört. Beispielsweise kann man die längste Kante als gemeinsame Hypotenuse c für alle Dreiecke wählen, die von den beiden Knoten dieser Kante und jeweils einem der übrigen Knoten gebildet werden. Dann kann die Höhe des Dreiecks und der Hypotenusenabschnitt p über die Formel $p = c/2 - (a^2 - b^2)/2c$ berechnet werden. Das sind quasi Koordinaten der Knoten.

Grundsatzüberlegung 2:

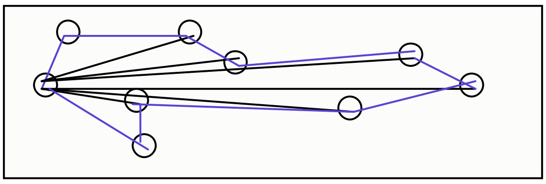
Gegeben sei ein rechteckiges Gitternetz, in dem alle benachbarten Punkte gleichen Abstand voneinander haben. Dann gibt es Lösungen für die kürzesten Routen, in denen weder Kreuzungen noch Diagonalen vorkommen:





Ein Graph ist als unvollständiges Gitternetz aufzufassen. Daraus ergeben sich heuristische Strategien zur Findung kürzester Routen in Graphen. Charakteristisch sind die drei zusammenhängenden Aussenseiten und die Mäanderformen. Zu beachten ist, ob die Anzahl der Knoten einer Seite gerade oder ungerade ist.

So kann der Winkel aller Knoten zu einem der beiden Punkte der längsten Kante berechnet werden und der Graph danach sortiert werden. Daraus ergibt sich ein kreuzungsfreier, geschlossener Weg.



Fazit:

Aus den beiden Überlegungen ergibt sich die Möglichkeit, die Knoten nach Hypotenusen-abschnitt p und Höhe zu sortieren und die Gitternetz-Strategie anzuwenden. Dadurch wird die Information über den Graphen angereichert, die notwendige Vorausschau verkürzt und die Berechnung aller Möglichkeiten überflüssig, so dass der Rechenaufwand deutlich reduziert wird. Ob ein Problem polynomial lösbar ist, hängt entscheidend davon ab, wieviel Information verfügbar ist. Besonders kommt es darauf an, ob versteckte und externe Information auf Grund von Naturgesetzen oder logischen Gesetzen nutzbar gemacht werden kann.
