22. Bundeswettbewerb Informatik, 1. Runde

Lösungshinweise und Bewertungskriterien

Allgemeines

Zuerst soll an dieser Stelle gesagt sein, dass wir uns sehr darüber gefreut haben, dass einmal mehr so viele Leute sich die Mühe gemacht und die Zeit zur Bearbeitung der Aufgaben genommen haben. Dass das Zeitnehmen nicht immer optimal klappt und es deshalb kurz vor Einsendeschluss etwas brenzlig wird, ist nachvollziehbar und völlig verständlich. Leider dürfen wir darauf keine Rücksicht nehmen. Insbesondere sind vollständige Einsendungen ein Muss. Im Einzelnen:

- Beinahe das Wichtigste ist, die Teilnahmeformulare vollständig und leserlich auszufüllen und für jedes Gruppenmitglied auch wirklich ein eigenes, vollständig ausgefülltes Formular abzugeben. Zu unnötig langen Suchprozessen führt es, wenn die Formulare zwischen den Aufgaben versteckt sind oder Teilnehmer ihre Namen nicht zumindest auf die erste Seite der Lösung schreiben – und zwar bei jeder Aufgabe!
- Online-Anmelder wurden gebeten, ihre Nummer außen auf den Umschlag zu schreiben. Die meisten haben das gemacht, aber leider nicht alle. Das führt zu Komplikationen und verzögert insbesondere die versprochene Rückmeldung per E-mail über den Eingang der Einsendung.
- Was uns auch hilft: Seien Sie mit Ihrem Namen nicht so geizig! Schreiben Sie Ihn ruhig häufiger, z. B. auf das erste Blatt jeder Aufgabe und auf Ihre CD oder Diskette(n).
- Beispiele werden als Teile des Programm-Ablaufprotokolls immer erwartet. In diesem Jahr war bei zwei Aufgaben ein Beispiel vorgegeben. Zu wenig Beispiele und erst recht die Nichtbearbeitung des vorgegebenen Beispiels führten zu Punktabzug.
- Beispiele, aber auch Programmdokumentation und Programmtext müssen ausgedruckt sein. Wir können aus Zeit- und Kostengründen keine Ausdrucke machen, so dass es prinzipiell nichts nützt, Beispiele nur auf CD/Diskette abzugeben oder gar nur ins Programm einzubauen. In solchen Fällen gab es Punktabzug.

- Zu einer Einsendung gehören auch lauffähige Programme. Kompilierung von Quellcode ist während der Bewertung nicht möglich. Für die gängigsten Skript-Sprachen stehen Interpreter zur Verfügung.
- Noch schlechter als Einsendungen nur auf Datenträgern wären für uns übrigens Einsendungen via E-mail oder anderen Internet-Wegen, auch wenn das für die Teilnehmer noch so praktisch wäre. Papiereinsendungen sind (zumindest zur Zeit und sicher auch noch in den nächsten Jahren) einfach unumgänglich.
- Eine Gruppeneinsendung schicken Sie uns bitte in einem gemeinsamen Umschlag, wir haben sonst größte Mühe, die Einsendungen richtig zuzuordnen. Wenn mehrere Einsendungen in einen Umschlag gesteckt werden, ist es besonders wichtig, bei der Online-Anmeldung bzw. auf den Anmeldebögen die Zusammensetzung der Gruppe anzugeben. Außerdem: Eine Gruppe muss sich auf eine Lösung pro Aufgabe einigen, und Gruppenmitglieder können nicht gleichzeitig auch eine eigene Einsendung schicken.

So, vielleicht nehmen Sie sich diese Anmerkungen ja zu Herzen, wenn Sie (hoffentlich) im nächsten Jahr wieder mitmachen.

Auch die folgenden eher inhaltlichen Dinge sollten Sie beachten:

- Lösungsideen sollten Lösungsideen sein und keine Bedienungsanleitungen oder Wiederholungen der Aufgabenstellung. Es soll beschrieben werden, welches Problem hinter der Aufgabe steckt und wie dieses Problem grundsätzlich angegangen wird. Eine einfache Mindestbedingung: Bezeichner von Programmelementen wie Variablen, Prozeduren etc. dürfen nicht verwendet werden – eine Lösungsidee ist nämlich unabhängig von solchen Realisierungsdetails. In diesem Jahr waren die meisten Lösungsideen in dieser Hinsicht recht gut, oft aber ein wenig kurz.
- Auch ein Programmablauf-Protokoll soll keine Bedienungsanleitung sein. Es beschreibt nicht, wie das Programm ablaufen sollte, auch nicht die zum Ablauf nötigen Interaktionen mit dem Programm, sondern protokolliert den tatsächlichen, inneren Ablauf eines Programms. Sprich: am besten protokolliert ein Programm seinen Ablauf selbst, z. B. durch Herausschreiben von Eingaben, Zwischenschritten oder -resultaten und Ausgaben.
- Oben wurde schon gesagt, dass Beispiele immer dabei sein sollten, zumindest eines davon in einem Programm-Ablaufprotokoll. Das hat seinen Grund: An den Beispielen ist oft direkt zu sehen, ob bestimmte Punkte korrekt beachtet wurden. Viele meinen nun, wir könnten die Programme ja laufen lassen und selbst auf Beispieldaten ansetzen, und liefern keine Beispiele oder nur Beispieldaten in elektronischer Form. Das können wir aber aus Zeitmangel in der Regel nicht. Außerdem ist nicht immer sicher, dass Programme, die auf dem eigenen PC laufen, auch auf einem anderen Computer ausführbar sind. Generell muss man sich darauf einstellen, dass nur das Papiermaterial angesehen wird!
- Mit den verschiedenen Beispielen sollten Sie wichtige Varianten des Programmablaufs zeigen, also auch Sonderfälle, die vom Programm berücksichtigt werden.

Einige Anmerkungen noch zur Bewertung:

- Pro Aufgabe werden maximal fünf Punkte vergeben, bei Mängeln gibt es entsprechend weniger Punkte. Für die Gesamtbewertung sind die drei am besten bewerteten Aufgabenlösungen maßgeblich, es lassen sich also maximal 15 Punkte erreichen. Einen 1. Preis erreichen Sie mit 14 oder 15 Punkten, einen 2. Preis mit 12 oder 13 Punkten und eine Anerkennung mit 10 oder 11 Punkten. Die Preisträger sind für die zweite Runde qualifiziert.
- Auf den Bewertungsbögen bedeutet ein Kreuz in einer Zeile, dass die (negative) Aussage in dieser Zeile auf Ihre Einsendung zutrifft. Damit verbunden war dann in der Regel der Abzug eines oder mehrerer Punkte. Eine Wellenlinie bedeutet „na ja, hätte besser sein können“, führte aber meist nicht zu Punktabzug. Mehrere Wellenlinien konnten sich aber zu einem Punktabzug addieren.
- Wellenlinien wurden übrigens häufig für die Dokumentation (also Lösungsidee, Programm-Dokumentation, Programmablauf-Protokoll und kommentierter Programm-Text) verteilt, obwohl Punktabzug auch gerechtfertigt gewesen wäre.
- Aber auch so ließ sich nicht verhindern, dass etliche Teilnehmer nicht weitergekommen sind, die nur drei Aufgaben abgegeben haben in der Hoffnung, dass schon alle richtig sein würden. Das ist ziemlich riskant, da Fehler sich leicht einschleichen.

Zum Schluss:

- Sollte Ihr Name auf der Urkunde falsch geschrieben sein, können Sie gerne eine neue anfordern. Uns passieren durchaus schon mal Tippfehler, und gelegentlich scheitern wir an der ein oder anderen Handschrift.
- Es ist verständlich, wenn jemand, der nicht weitergekommen ist, über eine Reklamation nachdenkt. Gehen Sie aber bitte davon aus, dass wir in allen kritischen Fällen, insbesondere denen mit 11 Punkten, genau nachgeprüft haben, ob nicht doch irgendwoher die Punkte zu holen gewesen wären, die das Weiterkommen ermöglicht hätten.

Wir wollen auch in diesem Jahr zusätzlich zu diesen Lösungsideen ausführlichere Beispiellösungen erstellen, die auf den Webseiten des Bundeswettbewerbs Informatik (`www.bwinf.de`) veröffentlicht werden – allerdings wohl nicht vor Ende Januar. Bis dahin kann man sich mit den Einsendungen befassen, die unter `www.tobias-thierer.de/bwifiles.html` von einigen Teilnehmern veröffentlicht wurden.

Aufgabe 1: Raff

Lösungsidee

Da das Programm laut Aufgabenstellung einen Tauschzyklus mit maximalem Gewinn pro Tausch ermitteln soll, entfällt die Möglichkeit, mittels einer Heuristik das optimale Ergebnis zu approximieren. Deswegen muss man aus der Menge der Tauschfolgen zuallererst einmal alle möglichen Tauschzyklen ermitteln; dies ist am besten mit einem Suchverfahren wie Tiefensuche möglich. Aus diesen Zyklen wird dann anschließend einer mit maximalem Gewinn pro Tausch ermittelt und als mögliche Lösung ausgegeben.

Teilzyklen? Keine Teilzyklen!

Dass dies auch funktioniert, ist bei näherer Betrachtung zunächst gar nicht offensichtlich: Da von der Aufgabenstellung aus ein Zyklus nur dadurch definiert ist, dass Ausgangs- und Endwährung identisch sind, ist z.B. sowohl Entenpesete $\rightarrow$ Krötendollar $\rightarrow$ Entenpesete, als auch Entenpesete $\rightarrow$ Krötendollar $\rightarrow$ Entenpesete $\rightarrow$ Krötendollar $\rightarrow$ Entenpesete sowie auch Entenpesete $\rightarrow$ Krötendollar $\rightarrow$ Wolfspfund $\rightarrow$ Krötendollar $\rightarrow$ Entenpesete ein gültiger Tauschzyklus.

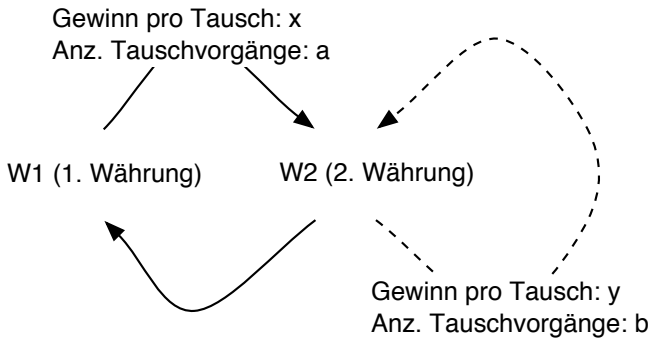


Abbildung 1: Tauschzyklus mit Teilzyklus

Verallgemeinert man diesen Gedanken, lässt sich Folgendes sagen: Gibt es für die Währung W_2 einen Tauschzyklus und für die Währung W_1 einen Tauschzykls der Form, $W_1 \ldots \rightarrow W_2 \ldots \rightarrow W_1$, dann gibt es unendlich viele weitere Zyklen der Form $W_1 \ldots \rightarrow W_2(\rightarrow \ldots \rightarrow W_2)* \ldots \rightarrow W_1$, wobei der $*$ besagt, dass der Teil in den Klammern beliebig oft wiederholt werden kann. Wir erhalten also Zyklen mit einem Teilzyklus, der bei W_2 beginnt und endet und ggf. mehrfach durchlaufen wird. Abbildung 1 illustriert diesen Sachverhalt.

Sind nun solche Teilzyklen sinnvoll? In der Aufgabenstellung ist folgendes Maß zur Mittelwertbildung gegeben: $(($Produkt der Kurse $- 1) \times 100)/($Anzahl der Tauschvorgänge$)$. Seien z.B. alle Tauschkurse gleich 2, dann erhält man nach n Tauschvorgängen einen Gewinn pro Tausch von $((2^n - 1) \times 100)/n$. Das Problem hierbei ist, dass dieser Wert für wachsendes n

ebenfalls monoton wächst, d.h. Dagobert würde einen maximalen Gewinn pro Tausch erzielen, wenn er möglichst oft tauscht. Je mehr Teilzyklen durchlaufen werden, desto größer wird der Gewinn. Damit wäre unendliches Tauschen ideal. Doch die Lösungsidee, die Menge von Tauschfolgen nach Tauschzyklen zu durchsuchen, muss scheitern (und wohl auch jeder andere Ansatz), wenn Teilzyklen unendlich viele Tauschzyklen möglich machen.

Man muss also entweder die Einschränkung vornehmen, dass jede Währung (abgesehen von der Startwährung, die natürlich auch als Endwährung auftritt) nur einmal in einem Zyklus auftreten darf oder aber, dass jeder Kurs aus der Kurstabelle nur einmal in einem Zyklus auftauchen darf. Beide Einschränkungen führen jeweils dazu, dass man mit erschöpfendem Suchen alle Möglichkeiten durchprobieren kann und damit wieder einen Zyklus mit maximalem Gewinn pro Tausch bestimmen kann.

Noch klarer gegen Teilzyklen kann argumentiert werden, wenn statt der in der Aufgabenstellung gegebenen Mittelung das geometrische Mittel[1] verwendet wird, was nach der reinen mathematischen Lehre für Mittelungen solchger multiplikativen Gewinne das korrekte Maß ist. Es bringt in diesem Fall keinerlei Vorteil, Tauschzyklen mit Teilzyklen zum Tauschen zu verwenden, wie die folgende Analyse zeigt (vgl. auch Abbildung 1): Wenn man ohne den Teilzyklus von W_1 über W_2 wieder nach W_1 tauscht, sei der geometrisch gemittelte Gewinn x. Wenn man nur den Teilzyklus von W_2 wieder nach W_2 tauscht, sei der Durchschnittsgewinn y, und wenn man von W_1 über W_2, n-mal über den Teilzyklus wieder nach W_2 und dann nach W_1 tauscht, beträgt der Gewinn pro Tausch $\sqrt[1+n]{xy^n}$.

Es lassen sich nun 3 Fälle unterscheiden:

1. $x > y \Rightarrow \sqrt[1+n]{xy^n} < \sqrt[1+n]{x^{1+n}} = x$

   In diesem Fall ist es also günstiger nicht den Teilzyklus zu verwenden, sondern nur von W_1 über W_2 wieder nach W_1 zu tauschen.

2. $x < y \Rightarrow \sqrt[1+n]{xy^n} < \sqrt[1+n]{y^{1+n}} = y$

   In diesem Fall ist es günstiger, nur den Teilzyklus zum Tauschen zu verwenden, also von W_2 wieder zurück nach W_2.

3. $x = y \Rightarrow \sqrt[1+n]{xy^n} = x = y$

   In diesem Fall ist es egal, welchen der drei möglichen Varianten man zum Tauschen verwendet. Es ist aber somit auch nicht von Nachteil, wenn man die Variante inklusive Teilzyklus aus der Liste der Möglichkeiten streicht.

Es folgt, dass man (bei Verwendung des geometrischen Mittels) ohne Bedenken Tauschzyklen, die Teilzyklen enthalten, von den Betrachtungen ausschließen kann, da sie bestenfalls den gleichen Gewinn pro Tausch erzielen wie ein kleinerer Tauschzyklus.

[1] Das geometrische Mittel von n Zahlen $a_1, \ldots, a_n$ ist definiert als $\sqrt[n]{a_1 \times \ldots \times a_n}$.

Beispiele

Auf der Grundlage der beschriebenen Lösungsidee wurde ein Programm entwickelt. Für einige Beispieltabellen werden dessen (korrekte) Ergebnisse angegeben. Das Programm arbeitet mit geometrischer Mittelung.

1. Beispiel (Pflichtbeispiel aus der Aufgabenstellung)

```
Beispiel für einen Tauschzyklus mit maximalem Gewinn
pro Tausch:

Entenpeseta --> Wolfspfund (Wechselkurs: 1.8)
Wolfspfund --> Entenpeseta (Wechselkurs: 0.7)

Gewinn pro Tausch: 12,25%
```

Diese Lösung ist auch bei Verwendung des Maßes aus der Aufgabenstellung optimal, mit einem Gewinn pro Tausch von 13% ($1,8 \times 0,7 = 1,26$). Natürlich ist der Zyklus Wolfspfund $\rightarrow$ Entenpeseta $\rightarrow$ Wolfspfund gleichwertig.

2. Beispiel

Bei diesem Beispiel wird eine Kurstabelle verwendet, die ausschließlich verlustbehaftetes Tauschen ermöglicht, nämlich:

```
A-->B:0.8
C-->D:0.95
E-->A:0.999
A-->D:0.953
E-->B:0.27
B-->A:0.6
A-->B:0.83
B-->C:0.9
D-->A:0.994
```

Das Ergebnis lautet:

```
Beispiel für einen Tauschzyklus mit minimalem Verlust
pro Tausch:

A --> D (Wechselkurs: 0.953)
```

```
D --> A (Wechselkurs: 0.994)

Gewinn pro Tausch: -2,67%
```

In so einem Fall reichte uns auch die Ausgabe, dass die vorliegende Tabelle keinen gewinnbringenden Tausch erlaubt.

3. Beispiel

Im 3. Beispiel wird folgende Kurstabelle verwendet, die überhaupt keinen geschlossenen Zyklus ermöglicht:

```
W-->Z:1.5
X-->W:0.7
X-->Z:1.2
X-->Y:0.3
Y-->Z:2.0
```

Ergebnis:

```
Es wurden keine Tauschzyklen gefunden.
```

Bewertungskriterien und häufige Fehler

Folgende Kriterien wurden bei der Bewertung berücksichtigt:

- Das Programm muss in der Lage sein, optimale Tauschzyklen zu finden. Eine Annäherung der Lösung, z.B. mittels Heuristiken, ist nicht ausreichend. Natürlich muss die Berechnung korrekt sein, insbesondere darf die Mittelung des Gewinns nicht vergessen werden.
- Es muss beschrieben sein, wie der Zyklusbegriff aufgefasst wird und welche Mittelwertberechnung gewählt wird. Wenn mit dem vorgegebenen Maß gearbeitet wurde, reicht die Aussage, dass Teilzyklen z.B. durch das Verbot von Währungswiederholungen vermieden werden.
- Das Programm muss für ein Währungspaar A,B mehrere Kurse akzeptieren (wie in der Beispieltabelle). Das kann aber auch so geschehen, dass beim Einlesen der Kurse für ein Währungspaar die schlechteren ignoriert werden.

- Es sollen weder die Kurse noch die Währungsbezeichnungen noch die Anzahl der Währungen oder Kurse im Programm fest verdrahtet sein. Es wurde aber akzeptiert, wenn die Namen der in der Aufgabenstellung genannten Währungen als Konstanten im Progamm definiert waren.
- Außer dem Gewinn pro Tausch muss auch die Währungsabfolge ausgegeben werden.

Wie bei allen anderen Aufgaben müssen genügend Beispiele angegeben werden. In der Aufgabenstellung werden drei Beispiele gefordert, eines davon muss die in der Aufgabenstellung gegebene Beispieltabelle sein.

Leider wurde in vielen Lösungsprogrammen die Mittelung des Gewinns vergessen. Es reicht nicht aus, die Mittelung erst nach dem Vergleich der möglichen Tauschzyklen vorzunehmen, denn auch dann werden längere Zyklen bevorzugt. Es sollten schon die besten Zyklen im Bezug auf den Gewinn pro Tausch gefunden werden, und das auch unabhängig von der Startwährung. In einigen Fällen war die Ausgangswährung von Hand vorzugeben, was nicht Sinn der Sache ist.

Häufig wurde einfach ignoriert, dass ein Lösungsprogramm die Währungstabelle einlesen sollte. Das haben wir in der Regel hingenommen. Es war bei dieser Aufgabe (wie so oft) am besten, die Eingabedaten aus einer Datei auszulesen. Damit erspart man sich die Programmierung aufwändiger Schnittstellen für die benutzergesteuerte Eingabe. Außerdem wird es dann leichter, das Programm flexibel zu halten. Ein vernünftiger Algorithmus für dieses Problem funktioniert natürlich völlig unabhängig davon, wie die Währungen heißen, um wieviele Währungen und um wieviele Kurse es sich handelt. Es wurde deshalb auch negativ bewertet, wenn die Struktur des Programms durch die Anzahl der Währungen bestimmt war; viel zu häufig waren im Quellcode sechs ineinander verschachtelte Schleifen zu entdecken, mit denen man für die gegebenen Währungen bei Ausschluss von Wiederholungen natürlich auch alle Möglichkeiten ausprobieren konnte.

Aufgabe 2: Keuch

Liste der Sprachformulierungen

Bei der Umsetzung der Alarmstrategie müssen für die markierten Sprachformeln numerische Werte eingesetzt werden:

- Wird bei einem Handy der Grenzwert für einen Schadstoff überschritten, alarmiert es akustisch und optisch seinen Besitzer.
- Gleichzeitig sendet das Handy an den Server seiner Funkzelle eine Gelb-Nachricht für diesen Schadstoff, die dieser an **einige** zufällig ausgewählte andere Handys in der gleichen Funkzelle weitersendet.
- Empfängt ein Handy **gelegentlich** einzelne Gelb-Nachrichten, tut es nichts.
- Empfängt ein Handy **innerhalb kurzer Zeit** **etliche** Gelb-Nachrichten bezüglich desselben Schadstoffes, alarmiert es seinen Besitzer, auch wenn sein entsprechender Grenzwert nicht überschritten ist.
- Empfängt der Zellenserver **innerhalb kurzer Zeit** **eine größere Anzahl** Gelb-Nachrichten bezüglich desselben Schadstoffes, sendet er eine entsprechende Rot-Nachricht an alle Handys in seiner Funkzelle und an alle Server der benachbarten Funkzellen.
- Empfängt ein Handy eine Rot-Nachricht, alarmiert es seinen Besitzer, auch wenn der entsprechende Grenzwert bei ihm nicht überschritten ist.

Somit gibt es sechs entsprechende Sprachformulierungen, die fünf Strategievariablen bilden bzw. nur vier, da „innerhalb kurzer Zeit“ (in Kombination mit „etliche“) zu „gelegentlich“ invers ist. Für die Grenzwerte der Handys sind in der Imsetzung zwar Variablen vorzusehen, diese sind aber keine Variablen oder Parameter der Strategie.

Alarmstrategie eines Handys

Es gibt drei Möglichkeiten einer Alarmauslösung:

1. Ein Grenzwert wird lokal überschritten.
   Dazu reicht es aus, die lokalen Messwerte auszulesen und mit den im Handy gespeicherten Grenzwerten zu vergleichen. Sind diese Werte über-/unterschritten, wird der Alarm aktiviert und eine Nachricht an den Server geschickt.
2. Eine Rot-Nachricht wird empfangen.
   Auch in diesem Fall reicht es unmittelbar einen Alarm auszulösen.

3. Etliche Gelb-Nachrichten werden empfangen.
   Für diese Alarmauslösung muss das Handy die letzten Meldungen im Speicher halten. Wieviele Nachrichten gespeichert werden müssen, hängt von der Anzahl der Schadstoffe und vom Wert der Strategievariablen „etliche“ ab. Insgesamt muss je Schadstoff Speicherplatz für „etliche“ Nachrichten vorhanden sein. [2]

Alarmstrategie des Zellenservers

Es gibt zwei Ereignisse, auf die der Zellenserver reagieren muss:

1. Gelb-Meldungen von Handys seiner Zelle
   Wie bei der Alarmstrategie der einzelnen Handys ist es auch hier erforderlich, dass der Zellenserver ausreichend Speicherplatz für eine „größere“ Anzahl an Gelb-Nachrichten je Schadstoff zur Verfügung hat. Es reicht wiederum nicht aus, nur eine Anzahl zu speichern. Handelt es sich um „wenige“ Gelb-Meldungen, muss der Server die Meldung an einige zufällige Handys weiterleiten, handelt es sich um viele, wird Rot-Alarm ausgelöst.
2. Rot-Meldungen aus den Nachbarzellen
   In der Aufgabenstellung ist nicht definiert, wie der Server hier zu reagieren hat. Eine geeignete Strategie wäre zum Beispiel eine Rot-Nachricht an alle Handys seiner Funkzelle zu senden, jedoch nicht an seine Nachbarserver.[3]

Obwohl nicht explizit gefordert, muss das Gesamtsystem mit Hilfe der in den Teilen 2 und 3 simuliert werden, um Ablaufbeispiele zu generieren. Dabei reicht es völlig aus, beide Teile in einem Gesamtprogramm zu integrieren; Interprozesskommunikation oder gar Netzwerkfähigkeit sind Kanonen, mit denen man diesen Spatz wirklich nicht beschießen muss. Wichtig: Die Simulation des Servers und der Handys erfolgt abwechselnd in jedem Zeittakt.

Provozierter Fehlalarm

Es gibt mehrere Möglichkeiten, Fehlalarme zu provozieren, sowohl für Einzelpersonen, als auch für Gruppen von Personen.

Im einfachsten Fall kann eine Einzelperson die in ihrem Handy gespeicherten Grenzwerte mehrfach unter den aktuellen Messwert ändern. Das Handy würde jeweils eine Nachricht abschicken. Gleiches Vorgehen kann natürlich auch von einer Gruppe von Personen durchgeführt werden, so dass die Nachrichten schneller aufeinander folgen.

[2] Es reicht hier nicht aus, eine Uhrzeit und eine Anzahl an Meldungen zu speichern. Es muss zwingend die Uhrzeit einer jeden Meldung mitgehalten werden! Wenn man beispielsweise nur Nachrichten zählen würde und den Zähler nach einem festen Intervall löschen würde, sind Fälle denkbar, in denen binnen kurzer Zeit etliche Meldungen eintreffen, ohne dass das Handy Alarm schlägt.

[3] Wichtig ist nur, dass der Server nicht wiederum alle seine Nachbarserver informiert, sonst klingeln alle Handys im Netz!

Als Abwehrmaßnahme gegen dieses Vorgehen empfiehlt es sich, den Server dahingehend zu erweitern, dass er zum einen mithält, von welchem Handy die Nachrichten kamen[4] und zum anderen die Messwerte einer Plausibilitätskontrolle unterzieht.

Außerdem ist denkbar, dass Personen ihre Handys bewusst Schadstoffen aussetzen und damit die gemessenen Werte bewusst verfälschen. Dieser Systemangriff wäre vom Server nicht erkennbar, da nicht feststellbar ist, ob die Werte zum Beispiel auf Grund eines (lokalen) Ereignisses (Unfall, Brand o.ä.) entstehen oder zum Beispiel von einer Kerze stammen, die unter das Handy gehalten wird.

Erweiterung der Alarmstrategie des Zellenservers

Über die oben genannten Erweiterungen hinaus sind noch weitere Punkte denkbar. So könnte der Zellenserver z.B. die Position der einzelnen alarm-gebenden Handys mithalten und so zum einen zwischen lokalem und größerem Ereignis unterscheiden und auch seine Alarmstrategie entsprechend anpassen. Zum Beispiel müsste bei einem lokal auftretenden Ereignis nicht der Server der Nachbarzelle informiert werden. Weiterhin kann in Abhängigkeit des Schadstoffes verschieden schnell (nach verschieden vielen Gelb-Nachrichten) eine Rot-Nachricht verschickt werden.

Bewertungskriterien und häufige Fehler

Teil 1: Gesucht sind hier nur die sechs Sprachformulierungen, die zu Strategievariablen werden. Dabei konnte erkannt werden, dass „gelegentlich“ sich als Negation aus „etliche“ und „innerhalb kurzer Zeit“ ergibt. Es ist akzeptabel, wenn die Formulierungen „etliche“ und „eine größere Anzahl“ gleichgesetzt werden. Ebenfalls ist akzeptabel, wenn „innerhalb kurzer Zeit“ in den beiden Fällen seines Auftretens unterschiedlich umgesetzt wird. Wer als Strategievariablen auch die einstellbaren Grenzwerte auf dem Handy und die Server-Reaktion auf Rot-Nachrichten nennt, antwortet an der Frage vorbei; die Nennung der Grenzwerte wurde aber toleriert. Sind die Strategievariablen im Programm fest verdrahtet bzw. nicht an einer Stelle editierbar, führt dies zu Punktabzug.

Teil 2: Die Alarmstrategie muss natürlich korrekt implementiert sein. Wie schon erläutert, ist es wichtig, dass die einzelnen Nachrichten mit Uhrzeit gespeichert werden, damit sie richtig gezählt werden können.

Teil 3: Auch der Anteil des Servers an der Alarmstrategie muss korrekt realisiert sein. Auch beim Server können Nachrichten falsch gezählt werden, aber doppelt wurde nicht abgezogen. Sinnvollerweise kann in diesem Teil die Reaktion auf die Rotnachricht aus einer Nachbarzelle behandelt werden. Wenn nicht hier, dann aber bei Teilaufgabe 5.

[4] Die Speicherung der Absender ist auf jeden Fall sinnvoll. Man denke nur an ein Handy mit einem Wackelkontakt an einem Sender, so dass dieser abwechselnd korrekte und verfälschte Werte liefert!

Teil 4: Im Prinzip ist es egal, welches Szenario hier diskutiert wird. Wichtig ist nur, dass es stimmig ist und mit den festgelegten Regeln für Handy und Server funktioniert. Fehlen darf dieser Teil aber nicht.

Teil 5: Es ist gar nicht mal so einfach, hier vernünftige Erweiterungen hinzuschreiben, die nicht schon in Teil 4 genannt wurden. Insofern nur Abzug, wenn der Aufgabenteil nicht bearbeitet wurde oder unbrauchbar ist.

Allgemein: Wichtig ist, dass Server und Handys immer wechselweise simuliert werden. Ansonsten kann es zu Synchronisierungsfehlern kommen. Natürlich werden auch bei dieser Aufgabe drei Beispiele erwartet, also drei verschiedene Simulationsläufe des Handy/Server-Gesamtsystems.

Da bei dieser Aufgabe nicht explizit Beispiele gefordert wurden, wurden in zahlreichen Einsendungen erst gar keine produziert. Die allgemeinen Regeln verlangen aber Ablaufprotokolle. Diese Aufgabe war letztlich nur interessant, wenn mit den entwickelten Programmen auch Simulationen betrieben wurden, und zwar solche eines Handy/Server-Gesamtsystems. Von einer erfolgreichen Bearbeitung wurde erwartet, dass sie dies erkennt. Alternativ ist es möglich, Handy und Server isoliert zu simulieren. Dann müssen aber Generatoren für Nachrichten realisiert werden, während für die gekoppelte Simulation ausreichend ist, Schadstoffwerte die Handys mit Schadstoffwerten zu versorgen.

Aufgabe 3: Grab

Lösungsidee

Bei dieser Aufgabe handelt es sich um ein Optimierungsproblem. Ziel ist es, eine optimale, gültige Lösung zu bestimmen. Eine *gültige Lösung* ist hierbei ein Grabungsplan, der die geforderten Bedingungen erfüllt, nämlich dass alle Diamantenvorkommen mit der Außenwelt verbunden sind, dass die Stollen nicht verzweigen und dass die Stollen nicht schräg verlaufen. Eine *optimale Lösung* hat die Eigenschaft, dass der Grabungsaufwand, nämlich die Summe der Längen aller Stollen (inkl. der Fundorte), minimal ist.

Grundsätzlich kann man solch ein Optimierungsproblem lösen, indem man *alle* gültigen Lösungen betrachtet und die beste auswählt. Es ist offensichtlich, dass diese Vorgehensweise immer korrekt ist.

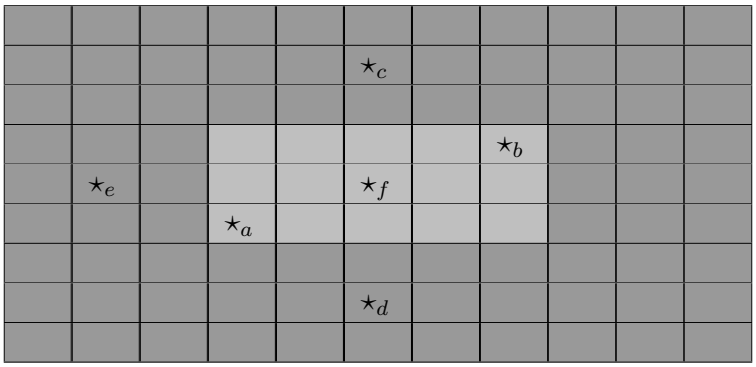
Die Hauptaufgabe ist also, die gültigen Lösungen systematisch zu erzeugen. Da die Größe des Bergs endlich ist, gibt es auch nur endlich viele verschiedene Lösungen. Gesucht ist also ein Verfahren, das in endlicher Zeit alle Lösungen erzeugt.

Eine mögliche Umsetzung

Im Folgenden sei eine Möglichkeit aufgezeigt, wie eine solche systematische Betrachtung ablaufen kann. Jedes Diamantenvorkommen kann entweder mit der Außenwelt oder jedem anderen Diamantenvorkommen verbunden werden. Dabei sind jeweils verschiedene Wege möglich. Das systematische Erzeugen aller möglichen Wege zwischen zwei Punkten ist aber zum einen nicht ganz einfach und erfordert zum anderen sehr großen Rechenaufwand, so dass an dieser Stelle schon eine Verbesserung vorweggenommen wird, um das Verfahren zu vereinfachen. Hierzu können wir folgende Beobachtungen verwenden.

- Zwischen zwei Diamantenvorkommen kann eine beliebige kürzeste Verbindung gewählt werden. Es ist also insbesondere nicht nötig, alle möglichen Wege zu betrachten. Abbildung 2 soll diesen Sachverhalt verdeutlichen.
- Die Felder, die außerhalb des Bergs liegen, aber direkt an ihn angrenzen, wollen wir „Ausgänge“ nennen. Die Felder, die innerhalb des Bergs liegen und direkt (nicht schräg) an einen Ausgang grenzen, heißen „Randfelder“. Es ist ausreichend, wenn man von einem Diamantenvorkommen nur den nächsten Ausgang betrachtet (bzw. einen beliebigen nächsten, falls es mehrere Ausgänge gibt, die die gleiche Entfernung haben). Falls es zwei kürzeste Wege gibt, die über verschiedene Randfelder zu dem gleichen Ausgang gelangen, müssen beide in Erwägung gezogen werden; ansonsten nur der eine. Abbildung 3 soll diesen Sachverhalt verdeutlichen.

Nun können wir mit folgendem rekursiven Backtracking-Algorithmus arbeiten, der mit dem ersten Diamantenvorkommen beginnt:



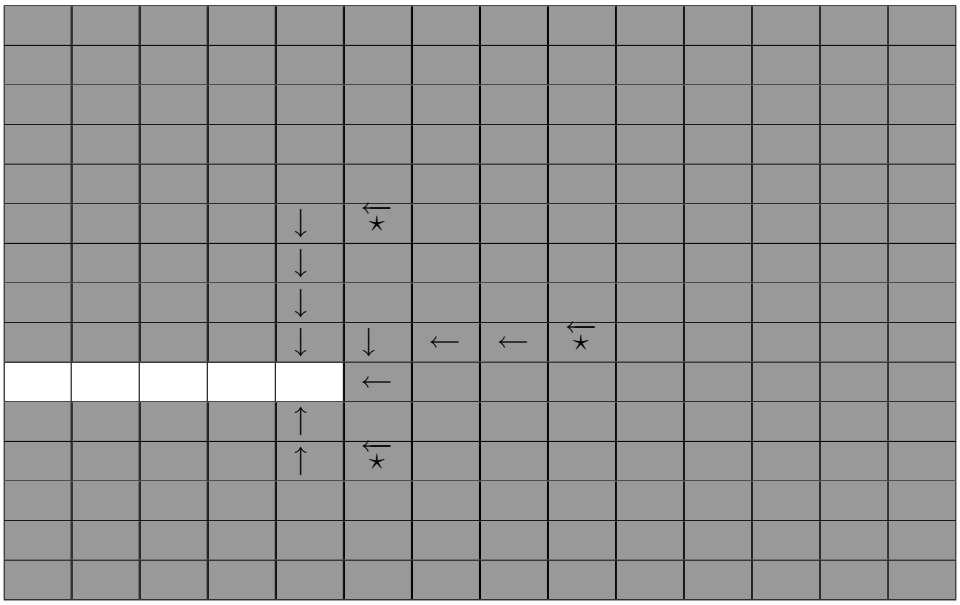
Alle kürzesten Wege zwischen den Fundorten a und b liegen in dem hervorgehobenen Rechteck:

1. Zwei Diamantenvorkommen c und d an gegenüberliegenden Seiten können unabhängig von der Wahl des Weges zwischen a und b *nicht* auf einem direkten Weg durch das Rechteck verbunden werden.
2. Zwei Fundorte c und e an benachbarten Seiten können auch außerhalb des Rechtecks optimal verbunden werden; es spielt also auch in diesem Fall keine Rolle, welche kürzeste Verbindung zwischen a und b gewählt wird.

Ein Diamantenvorkommen f, das innerhalb des Rechtecks liegt, kann „umsonst“ in den Weg zwischen a und b integriert werden.

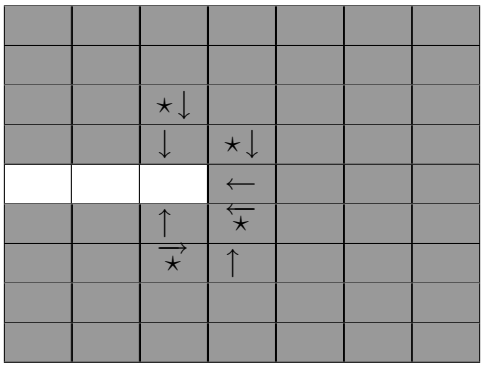
- Eine Verbindung von f mit genau einem Fundort außerhalb des Rechtecks kann also nicht zu einem optimalen Grabungsplan führen (da die Kosten dieser Verbindung überflüssig sind).
- Eine Verbindung von f mit zwei Diamantenvorkommen c und d oder c und e außerhalb des Rechtecks lässt sich auf den Fall 1 oder 2 reduzieren, wenn man davon ausgeht, dass die beiden Fundorte außerhalb des Rechtecks direkt verbunden werden und f in den Weg zwischen a und b integriert wird.

Abbildung 2: Kürzeste Verbindungen zwischen Diamantenvorkommen



(a) Drei Diamantenvorkommen teilen sich einen Ausgang

Für jeden Fundort gibt es zwei kürzeste Wege, die über verschiedene Randfelder zu dem Ausgang gelangen. Es reicht aus, für jeden Fundort beide Möglichkeiten auszuprobieren, um eine optimale Lösung zu finden.



(b) Vier Diamantenvorkommen teilen sich einen Ausgang

Es ist nicht möglich, vier (oder mehr) Diamantenvorkommen so zu platzieren, dass ein Konflikt an einem gemeinsamen Ausgang entsteht und es nicht mehr ausreicht, nur die zwei kürzesten Wege für jeden Fundort zu betrachten. Bei vier Fundorten ist nämlich immer mindestens einer nicht weiter von einem anderen Fundort als vom Ausgang entfernt, so dass eine Verbindung über den anderen Fundort möglich ist.

Abbildung 3: Gemeinsamer Ausgang

1. Bestimme mit Hilfe von Breitensuche einen kürzesten Weg von dem i-ten Diamantenvorkommen zu allen anderen Fundorten und einen oder zwei kürzeste Wege zu dem nächstgelegenen Ausgang.
2. Für alle gefundenen Wege:
   a) Grabe den Weg.
   b) Falls es sich bei dem i-ten Fundort um den letzten handelt, prüfe, ob es sich um eine gültige Lösung handelt und, im positiven Fall, ob sie besser ist als die bisher beste Lösung.
   c) Falls es sich bei dem aktuellen Fundort nicht um den letzten handelt, rufe den Algorithmus rekursiv für $i + 1$ auf.
   d) Mache die Grabung rückgängig.

Indem man bei der Breitensuche (Punkt 1) bei jedem Schritt nur in nördlicher, östlicher, südlicher oder westlicher Richtung vorgeht und Stollen nicht kreuzt, kann man schon einmal verhindern, dass die Lösung durch schräge Stollen oder Verzweigungen von Stollen zwischen zwei Fundorten ungültig wird. Weiterhin müssen Verzeigungen an Fundorten verhindert werden. Dazu muss man beim Graben der gefundenen Wege (Punkt 2) die Möglichkeiten überspringen, die zu einem Fundort führen, der bereits von einem anderen Diamantenvorkommen angesteuert wurde. Trotzdem ist es noch möglich, dass bei Punkt 2b immer noch ungültige Lösungen ankommen, nämlich Grabungspläne, bei denen nicht alle Fundorte mit der Außenwelt verbunden sind; dieser Fall tritt auf, wenn mehrere Diamantenvorkommen zyklisch miteinander verbunden sind. Dieser Fall muss deshalb gesondert geprüft werden.

Verbesserungen

Der große Nachteil der gewählten Vorgehensweise ist der zeitliche Aufwand. Wenn man wirklich alle Lösungen betrachten möchte, hängt die Laufzeit exponenziell von der Größe des Berges ab und übersteigt sehr schnell die Rechenleistung jedes aktuellen Computers. Durch die schon oben eingeführten Verbesserungen wird zumindest schon einmal erreicht, dass die Laufzeit nur noch exponenziell von der Anzahl der Diamantenvorkommen abhängt, die in den meisten Fällen deutlich kleiner als die Größe des Bergs ist. Wenn die Anzahl der Fundorte (wie in der Aufgabenstellung vorgegeben) 10 nicht übersteigt, ist das angegebene Verfahren geeignet, um in kurzer Zeit eine optimale Lösung zu finden. Für größere Eingaben wird es aber schnell sehr langsam. Wenn man das Programm beschleunigen möchte, kann man bspw. zusätzlich folgende Verbesserungen einbauen.

- Vor einem rekursiven Aufruf wird geprüft, ob die aktuelle Teillösung zu einer gültigen Lösung erweitert werden kann, die besser als die bisher beste ist. Ist dies nicht der Fall, kann man sich den rekursiven Aufruf sparen und mit dem Ausprobieren der nächsten Möglichkeit fortfahren (bzw. auf die nächsthöhere Ebene zurückkehren, wenn es sich um die letzte Möglichkeit auf der aktuellen Ebene handelt). Die aktuelle Teillösung

kann auf keinen Fall zu einer optimalen Lösung führen, wenn ihre Stollenlänge plus die Anzahl der noch zu betrachtenden Fundorte größer als die bisher beste Stollenlänge ist. Für jeden noch zu betrachtenden Fundort muss nämlich mindestens noch ein Stollen der Länge 1 gegraben werden. Dieses Abbruchkriterium wird weiter verbessert, wnn man für jeden Fundort nicht 1, sondern jeweils die kürzeste Verbindung zu einem anderen Fundort oder zu einem Ausgang hinzuaddiert. Diese Daten könnten am Anfang einmal berechnet werden.

- Zusätzlich könnte man bei jedem Fundort die Verbindungsmöglichkeiten nach wachsender Länge sortieren und in dieser sortieren Reihenfolge ausprobieren. Dies erhöht die Wahrscheinlichkeit, dass relativ früh eine gute Lösung gefunden wird. Und je früher gute Lösungen gefunden werden, desto öfter greift die o.g. Abbruchbedingung.

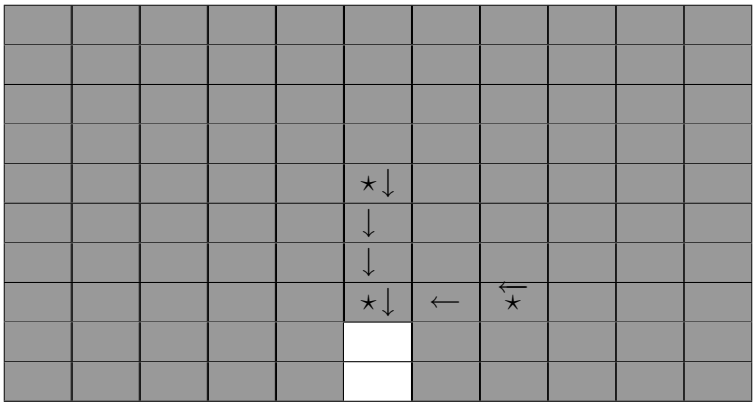
Minimal aufspannender Baum

Auf den ersten (und vielleicht auch zweiten) Blick könnte man meinen, das Problem sehr effizient und einfach lösen zu können, und zwar mit einem Algorithmus zur Bestimmung eines minimalen aufspannenden Baums (MST = Minimum Spanning Tree)[5] in einem gewichteten Graphen. In der Tat gibt es große Ähnlichkeiten. Es gibt aber auch wesentliche Unterschiede. Ein Unterschied ist, dass sich die Auswahl von Kanten gegenseitig ausschließen können (wenn durch die Auswahl von zwei Kanten bspw. eine Kreuzung entstehen würde) – bei der Bestimmung eines MST kann prinzipiell jede beliebige Kombination von Kanten gewählt werden. Außerdem kann es vorkommen, dass durch die Auswahl einer Kante eine andere verlängert wird (wenn ein Umweg nötig wird, um eine Kreuzung zu vermeiden) – auch dies ist bei einem Graphen normalerweise nicht der Fall. Der wichtigste Unterschied ist allerdings, dass auch keine Verzweigungen an den Fundorten erlaubt sind. Dadurch gibt es (ein paar wenige exotische) Fälle, bei denen jede gültige Lösung länger ist als das Gesamtgewicht eines MST. Abbildung 4 zeigt ein solches Beispiel.

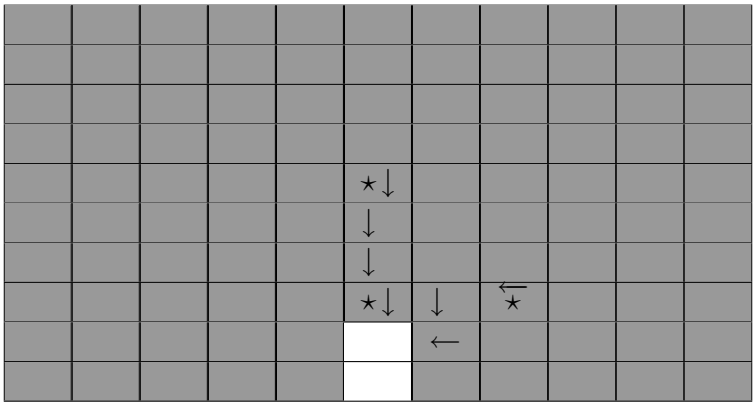
Trotzdem kann ein MST-Algorithmus auf verschiedene Weisen nützlich sein. Man kann bspw. das Gesamtgewicht eines MST berechnen, indem man einen vollständigen Graphen betrachtet, wobei jeder Fundort ein Knoten ist und die Außenwelt ebenfalls durch einen Knoten repräsentiert wird. Die Kantengewichte entsprechen den Manhattan-Abständen (d.h. $|x_2 - x_1| + |y_2 - y_1|$) zwischen den Fundorten untereinander bzw. den Fundorten und den nächstgelegenen Ausgängen. Kreuzungen von Stollen werden nicht beachtet. Der berechnete Wert kann als untere Schranke verwendet werden. Sobald eine gültige Lösung gefunden wurde, deren Stollenlänge dem Gesamtgewicht des MST entspricht, kann die Suche vorzeitig beendet werden, da eine bessere Lösung nicht mehr gefunden werden kann.

Alternativ kann man versuchen, einen MST-Algorithmus zu verwenden, um eine optimale gültige Lösung zu bestimmen. Damit der Algorithmus bei jeder Probleminstanz funktioniert,

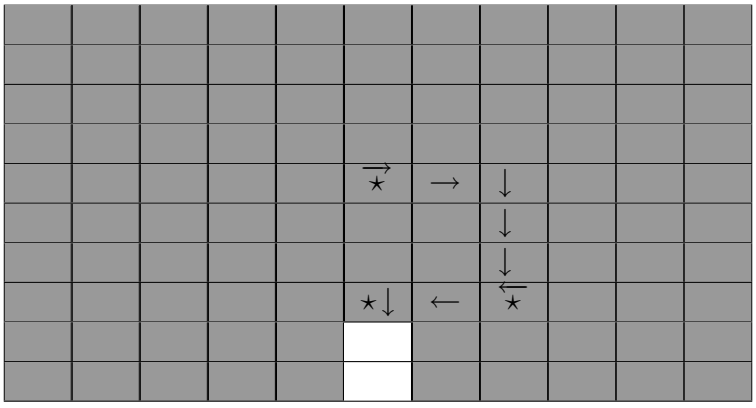
[5] Mehr erfahren zu diesem Begriff – und anderen in diesen Hinweisen genannten wie z.B. Breitensuche und Tiefensuche – kann man im Buch *Algorithmik* von Uwe Schöning, erschienen 2001 im Spektrum Akademischer Verlag.



(a) eine ungültige Lösung, die einem MST entspricht (Stollenlänge 6)



(b) eine optimale, gültige Lösung (Stollenlänge 7)



(c) eine nicht-optimale, gültige Lösung, die ein Verfahren, das auf einem MST-Algorithmus basiert, finden würde (Stollenlänge 8)

Abbildung 4: Gegenbeispiel MST

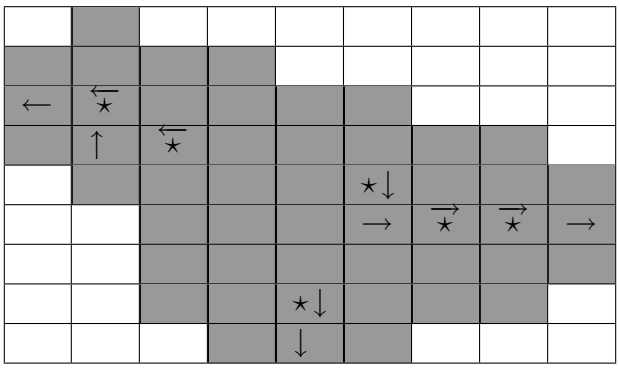


Abbildung 5: gefordertes Beispiel

muss man dabei auf die oben genannten Unterschiede achten, was ziemlich schwierig sein dürfte. Da aber in den meisten Fällen ein MST-Algorithmus schnell zu einer optimalen Lösung führt, lohnt sich eine Implementierung, bei der man zunächst prüft, ob man problemlos eine optimale Lösung finden kann, und nur notfalls auf das eingangs beschriebene, immer funktionierende Verfahren zurückgreift, um auch die exotischen Fälle optimal zu lösen.

Greedy

Eine weiteres recht effizientes Verfahren, das aber auch nicht immer optimale Lösungen garantieren kann, ist eine so genannte „gierige“ Vorgehensweise, die eine Reihe von Teilnehmern so oder in ähnlicher Form gewählt haben. Die Idee ist, sehr grob formuliert, etwa so: Für jeden Diamanten wird der kürzeste Weg zu einem Ausgang festgestellt. Dann, bis es für alle Diamanten einen Weg nach draußen gibt: 1. Für jeden Diamanten sehen, ob es nach draußen oder zu einem anderen Diamanten am kürzesten ist. 2. Im zweiten Fall beide Diamanten als Gruppe verbinden, die im weiteren bei Schritt 1 wie ein Diamant betrachtet wird.

Natürlich ist auch hier die Korrektheit der Grabungen sicherzustellen; man darf also einen Diamanten nicht mit seinem nächsten Nachbarn verbinden, wenn dieser schon an zwei anderen Verbindungen beteiligt ist. Zum anderen wird man – in Abhängigkeit von der Reihenfolge, in der die Diamanten betrachtet werden – nicht immer die optimale Lösung finden. Wenn man etwa das Beispiel aus Abbildung 4 behandelt, dort mit den beiden unteren Diamanten anfängt und diese lokal optimal verbindet, wird das globale Optimum verfehlt.

Beispiele

Abbildung 5 zeigt eine mögliche Lösung des in der Aufgabenstellung geforderten Beispiels. Die minimale Stollenlänge beträgt 11.

Abbildung 6 enthält eine Probleminstanz mit 10 Diamantenvorkommen und eine optimale Lösung. Eine Implementierung des im zweiten Abschnitt beschriebenen Verfahrens ohne die

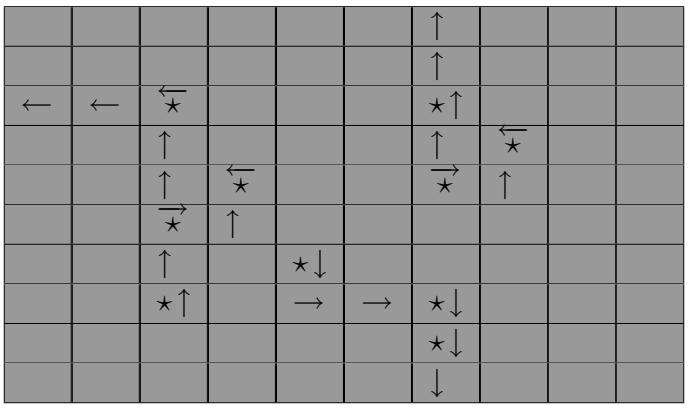


Abbildung 6: Beispiel mit 10 Diamantenvorkommen

später erwähnten Verbesserungen betrachtet 2.710.647 gültige Lösungen, um einen optimalen Grabungsplan zu bestimmen. Wenn man die Rekursion abbricht, wenn die aktuelle Teillösung nicht zu einer besseren Lösung als die bisher beste führen kann, werden nur noch 347 gültige Lösungen erzeugt. Das Sortieren der Möglichkeiten nach wachsender Länge reduziert die Anzahl der betrachteten Lösungen auf 198. Wenn man am Anfang auch noch das Gesamtgewicht eines MST berechnet und abbricht, wenn man diese untere Schranke erreicht hat, braucht man nur noch 139 Lösungen zu betrachten. Insgesamt werden bei diesem Beispiel durch die verschiedenen Verbesserungen 99,995% der gültigen Lösungen *nicht* erzeugt.

Bewertungskriterien

- Zunächst einmal muss man feststellen, dass diese Aufgabe schwerer ist als sie aussieht. Deshalb wird in der 1. Runde nicht erwartet, dass eine Einsendung eine effiziente *und* korrekte Lösung finden.
- In allen Fällen wird erwartet, dass nur gültige Lösungen ermittelt werden, d.h. die drei Regeln (alle Diamantenvorkommen erschlossen, keine Verzweigungen der Stollen, keine schrägen Stollen) müssen unbedingt eingehalten werden.
- Von einem ineffizienten Verfahren wird eine systematische Vorgehensweise (z.B. Backtracking) und Korrektheit erwartet; ein Beispiel dafür ist im zweiten Abschnitt beschrieben. Das Verfahren soll also immer (auch in exotischen Fällen) eine richtige Lösung finden. Wenn dies offensichtlich nicht der Fall ist, führt dies zu Punktabzug.
- Auch ein ineffizientes Verfahren muss mindestens so schnell sein, dass das geforderte Beispiel und generell Beispiele mit 10 Diamantenvorkommen gelöst werden können. Ein Verfahren, das bei n Planquadraten alle 2^n Grabungsmöglichkeiten durchgeht und prüft, ob es sich um eine Lösung handelt, ist z.B. nicht akzeptabel – und wird schon an den geforderten Beispielen scheitern. Jegliche Verbesserungen (Optimierungen) sind begrüßenswert, sie sollten aber zumindest kurz begründet werden und dürfen natürlich nicht dazu führen, dass die optimalen Lösungen übersehen werden.

- Von einem effizienten Verfahren (z.B. eines, das auf einem MST-Algorithmus basiert, oder auch ein Greedy-Ansatz) wird erwartet, dass es zumindest für die meisten Fälle eine korrekte Lösung findet. Wenn etwa das Gegenbeispiel aus Abbildung 4 nicht richtig gelöst wird, ist kein Punkt abgezogen worden. Allerdings wird erwartet, dass zumindest die Problematik erkannt wird, dass man die Stollen nicht unabhängig voneinander betrachten kann, sondern dass das Graben eines Stollens evtl. das Graben eines anderen Stollens verhindert.
- Die Aufgabenstellung fordert die grafische Darstellung von gefundenen Lösungen, aber nicht das automatische Erzeugen dieser Darstellung. Reine ASCII-Darstellungen sind zwar als Ausgabe des Programms akzeptabel, für den menschlichen Betrachter zeigen sie aber nicht immer eindeutig die Lösung an. Ein paar handgemalte Linien können da sehr hilfreich sein.
- Neben dem geforderten Beispiel werden mindestens zwei weitere Beispiele erwartet, die nicht zu einfach sein sollten (es sollten nicht beide weiteren Beispiele weniger als 6 Diamantenvorkommen haben).

Eine Bearbeitung dieser Aufgabe im Umfang dieses Abschnitts wurde natürlich nicht erwartet. Die große Mehrheit der erfolgreichen Einsendungen zu dieser Aufgabe wählten einen Greedy-Ansatz oder eine Lösung mit (verbessertem) Backtracking.

Aufgabe 4: Flitz

Lösungsidee

Das hier vorgestellte Modell geht von einem kreisförmigen Teich aus. Dies ist einerseits ziemlich nah an der Wirklichkeit, andererseits leicht zu implementieren. Außerdem ist ein Kreis völlig gleichförmig und in jeder Richtung symmetrisch, so dass es keine Ecken gibt, in denen Fische „stecken bleiben“ könnten. Die Goldfische werden ebenfalls, wie durch den Aufgabentext für die Visualisierung gefordert, durch Kreise modelliert, und zwar mit Durchmesser d_f. Es spielt sich also alles in zwei Dimensionen ab, was hier sicherlich eine sinnvolle Vereinfachung der Natur ist.

Das Sozialverhalten wird in drei Abstufungen modelliert: Zuneigung, Abneigung und neutrale Beziehung. Ist ein Goldfisch einem anderen zugeneigt, so bewegt er sich auf ihn zu. Hegt ein Fisch eine Abneigung gegen einen anderen, so bewegt er sich von ihm weg. Dabei sind die Geschwindigkeiten umso höher, je näher sich die betreffenden Fische sind, denn das erhöht den Grad der gegenseitigen Wahrnehmung. Diese Wünsche wirken also analog zu Kräften in der Physik in verschiedene Richtungen. Die vektorielle Addition dieser ergibt die Richtung und den Betrag, um den sich der Fisch in einem Zeitschritt fortbewegt. Um zu verhindern, dass die Goldfische den Teich verlassen, hat dessen Rand ebenfalls eine abstoßende Wirkung. Außerdem ergibt sich eine Abstoßung, wenn zwei Fische sich *zu* nahe kommen, denn schließlich wollen sie ja nicht zusammen stoßen, egal, wie lieb sie sich haben ;-). Die neutrale Beziehung bewirkt keine Bewegung, solange die Fische sich nicht zu nahe kommen, denn sie sind einander völlig egal.

Um nun das „muntere Treiben“ zu simulieren, wird Schritt für Schritt zunächst für alle Fische ihre neue Position berechnet und danach alle Fische an die entsprechende neue Position gesetzt. Dies resultiert in einer Animation, die beliebig lange laufen kann. Unter Umständen erreicht sie einen stabilen Zustand oder entwickelt sich periodisch. Um die Ausgabe zu beschleunigen, wird der aktuelle Zustand nur alle paar Zeitschritte (z. B. 20) wirklich ausgegeben.

Entscheidend für das Verhalten der Fische ist nun die Wahl der Funktionen für die Geschwindigkeiten. Dazu wird eine neue Größe eingeführt, der Sicherheitsabstand d_s. Das ist genau der Gleichgewichts-Abstand für Fische, die sich gern haben. Der Sicherheitsabstand d_s wird relativ zum Fischdurchmesser d_f konfiguriert (standardmäßig $\frac{3}{2}$). Für die Bewegung der Fische können nun z.B. folgende Festlegungen getroffen werden, r steht hierbei für den Abstand in Vielfachen von d_s:

$v_{Zuneigung} = \frac{1}{r^2} - \frac{1}{r^6}$ bewirkt, dass die anziehende Wirkung mit der Entfernung abnimmt. Ist der Abstand aber sehr klein, so wirkt eine sehr starke Abstoßung, die verhindert, dass die Fische zusammen stoßen. Die Funktion hat einen monoton steigenden Nulldurchgang bei 1, d. h. bei Abstand d_s heben sich Anziehung und Abstoßung auf, dies ist der Gleichgewichts-Abstand.

$v_{Abneigung} = -\frac{1}{r^2}$ hat den Effekt, dass die abstoßende Wirkung mit der Entfernung abnimmt.

$v_{Neutral} = \begin{cases} -\frac{1}{r^6} & \text{für} \quad r \leq d_s \\ 0 & \text{sonst} \end{cases}$ sorgt dafür, dass die Fische auch dann nicht zusammen stoßen, wenn sie sich eigentlich völlig egal sind. Es ist nicht garantiert, dass der Sicherheitsabstand d_s unter allen Umständen eingehalten wird. Drängeln sich viele liebestolle Fische auf engem Raum, kann er unterschritten werden. Es hat sich aber gezeigt, dass mit den gewählten Formeln und der gewählten Länge eines Zeitschritts (0.02) die Fische nie wirklich zusammenstoßen. Bei einem längerem Zeitschritt könnte es aber zu „Überreaktionen“ kommen.

Die Abstoßung vom Ufer wird analog zur Abstoßung der Fische mit neutraler Beziehung durch die folgende Formel erzeugt, wobei die Geschwindigkeit immer zu Mittelpunkt des Teichs gerichtet ist. r ist hier der Abstand des Fischs vom Rand.

$$v_{Rand} = \begin{cases} -\frac{1}{r^6} & \text{für} \quad r \leq d_s \\ 0 & \text{sonst} \end{cases}$$

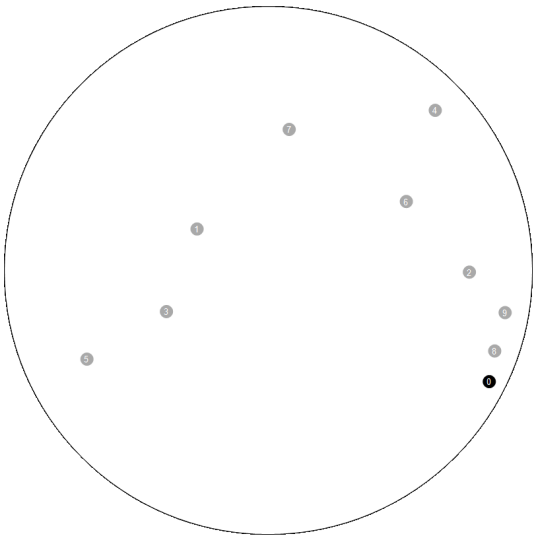
Zu Beginn der Simulation werden die Fische an zufälligen Positionen in den Teich gesetzt. Damit nicht gleich das Verhalten zu stark vorbeeinflusst wird, wird dabei ausnahmsweise der doppelte Sicherheitsabstand zueinander und zum Ufer eingehalten.

Beispiele

Auf der Grundlage der beschriebenen Lösungsidee wurde auch ein Beispielprogramm entwickelt, dass das Modell des Teiches, der Fische und ihrer Bewegungen realisiert und Simulationen ermöglicht. Im folgenden wird geschildert, was die Simulationen mit diesem Programm ergeben.

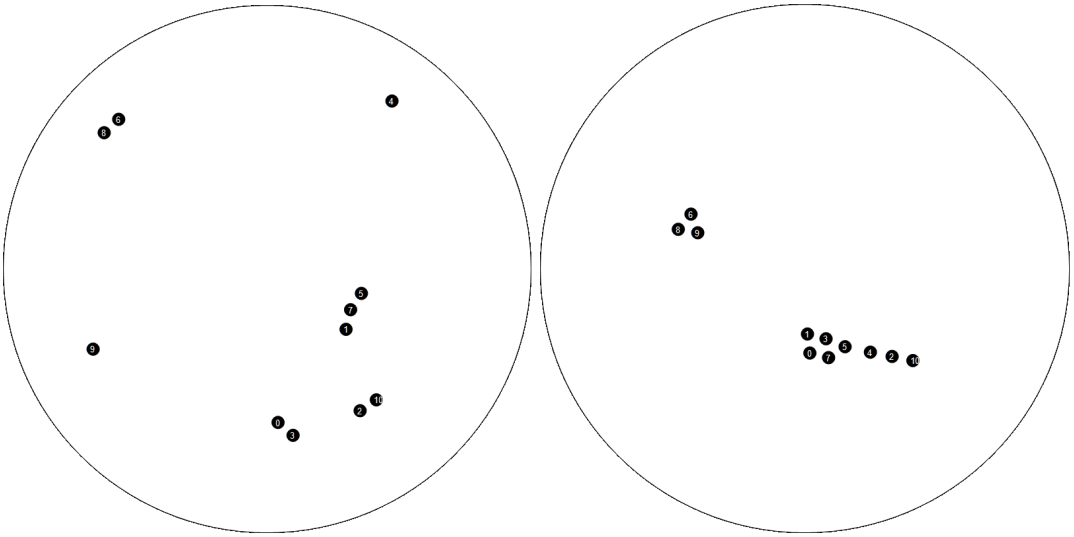
Erste Gegebenheit: Einzelgänger

Es werden insgesamt zehn Fische simuliert, der „Einzelgänger“ ist schwarz eingefärbt. Da dieser sich von allen anderen entfernt, nähert er sich zwangsläufig bald dem Rand. Falls er sich relativ weit in der Mitte befindet und von vielen anderen Fischen umgeben ist, kann das wahrlich zu einem slalomförmigen Spießruten-Laufen ausarten. Da sich die Verehrer trotzdem noch weiterhin nähern, versucht er immer zu fliehen. Das resultiert darin, dass er am Rand entlang im Kreis schwimmt und dabei von einem oder mehreren anderen Fischen verfolgt wird.



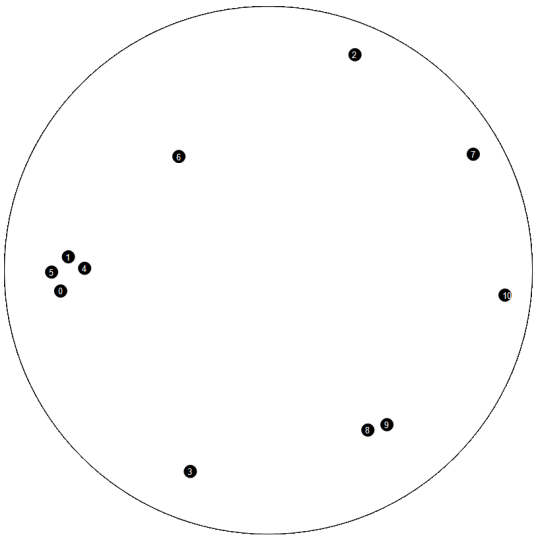
Zweite Gegebenheit: 70% Zuneigungen

Zuerst finden sich kleinere Gruppen zusammen (links), die sich später zu größeren (rechts) zusammenschließen. Da sich nicht unbedingt alle Mitglieder einer Gruppe grün sind, gibt es eine hohe Gruppendynamik, die dazu führt, dass sich innerhalb der Gruppen Fluktuationen zeigen und sich die Gruppen manchmal als Ganzes bewegen.



Dritte Gegebenheit: 70% Abneigungen

Im Fall von 70% Abneigungen bleibt es bei der Bildung von Gruppen der Größenordnung 2 bis 4. Dieser Endzustand ist schnell erreicht und ziemlich stabil, d. h. die Fische bewegen sich kaum noch.



Bewertungskriterien

Die Aufgabe ist eigentlich sehr frei gestellt. Es gibt daher viele Möglichkeiten, diese korrekt zu lösen. Insbesondere ist es möglich, den (rechteckigen) Teich in Rasterfelder einzuteilen und die Fische entlang des Rasters zu bewegen. Jedoch sollten die getroffenen Annahmen vernünftig sein oder zumindest vernünftig begründet werden. Folgende Punkte sollten bei den einzelnen Teilaufgaben berücksichtigt werden.

Teil 1: Modellierung

- Der Teich kann eine beliebige Form haben, allerdings sollte er nicht unbegrenzt groß sein oder als Torus (in einigen Einsendungen auch als Doughnut bezeichnetßmodelliert sein, denn dann ist es schlichtweg kein Goldfischteich mehr, sondern höchstens ein Ozean. Die damit verbundene Vereinfachung des Modells ist negativ zu bewerten. Die Forderung nach der Darstellung der Fische als Kreise legt eine rein zweidimensionale Simulation nahe, und mehr wird auch nicht erwartet. Drei Dimensionen schaden nicht, auch wenn es viel mehr Aufwand ist.
- Die Fische sollten weder gegenseitig zusammenstoßen noch den Teich verlassen können. Diese Restriktionen können entweder implizit aus der Berechnung der Bewegung folgen oder aber explizit überprüft werden. Erlaubt ist aber, wenn die Fische sich in einer Art zweieinhalbdimensionalen Realisierung über- oder untereinanderher bewegen können.
- Was für verschiedene Beziehungen können Fische untereinander haben? Sind diese Möglichkeiten kontinuierlich oder diskret? Das Minimum ist Zuneigung und Abneigung. Eine neutrale Beziehung kann, muss aber nicht modelliert werden. Wichtig ist, dass das gewählte Modell für Zu- und Abneigungen klar dokumentiert ist.

- Fische, die einander zugeneigt sind, sollten Nähe bevorzugen. Fische, die Abneigung verspüren, sollten sich tendenziell voneinander entfernen.

Teil 2: Visualisierung

- Mit der geforderten Visualisierung war eine softwaregesteuerte und am Bildschirm sichtbare Umsetzung des Modells aus Teil 1 gemeint. Bilder malen reicht also nicht, es ist aber auch nicht vorgegeben, dass ein eigenes Programm geschrieben werden musste. Auch spezielle Simulationswerkzeuge könnten hier eingesetzt werden.
- Um das Verhalten der Fische auch wirklich verfolgen und analysieren zu können, sollte irgendeine Form grafischer Darstellung implementiert sein, die sich mit der Zeit verändert. Es ist aber keine flüssige Animation gefordert. Die Fische müssen unterschiedliche Farben haben können.
- Es gibt hier keinen Algorithmus, der irgendwann terminiert. Die Simulation kann prinzipiell beliebig lange laufen. Der Benutzer sollte sie daher irgendwie steuern oder zumindest abbrechen können, und zwar ohne Zuhilfenahme des Betriebssystems.
- Der Teich, die Eigenschaften der Fische (Farbe!) und die Beziehungen sollten „weich“ konfigurierbar sein, z. B. durch eine Eingabedatei. Die Beziehungen können jedoch darin durchaus fest vorgegeben sein, es muss kein Zufallsprozess für die Beziehungsverteilung integriert sein.

Teil 3: Simulation

- Es muss nicht unbedingt eine Masse von Fischen simuliert werden. Es hat sich gezeigt, dass schon Anzahlen in der Größenordnung von 10 interessante Effekte zeigen. Deutlich weniger sollen es aber nicht sein.
- Zu jedem Beispiel sollte zumindest ein Zustand des Teichs dargestellt sein. Das typische dynamische Verhalten sollte in wenigen Sätzen beschrieben werden. Hier kann sich je nach Modellierung Unterschiedliches ergeben, es sollte jedoch nicht völlig der Vernunft und der natürlichen Erfahrung widersprechen. Beispielergebnisse siehe oben.

Aufgabe 5: Babbel

Lösungsidee

Aufgabenteil 1

Mit den Wörtern des Lexikon-Beispiels lassen sich einige Sätze konstruieren, in denen grundsätzliche Probleme der Wort-für-Wort-Übersetzung deutlich werden. Zur weiteren Veranschaulichung wollen wir jeweils noch ein anderes Beispiel angeben, das ebenfalls simpel ist, aber auch andere Wörter verwendet.

- *The mouse eats the cat. – Das Maus frisst der Katze.* Wörter mit mehreren Bedeutungen werden falsch übersetzt. Ein automatischer Wort-für-Wort-Übersetzer kann nicht wissen, welche Bedeutung eines Wortes er auswählen soll. Deutlicher als bei Artikeln wird dieses Problem bei Sätzen, in denen die ursprüngliche Bedeutung verloren geht: *The cat is a football fan. – Die Katze ist ein Fußball Ventilator.*
- *Eat the cat! – Fressen die Katze!* Wörter sind falsch konjugiert oder dekliniert. Manche Verbformen bzw. Fälle unterscheiden sich im Deutschen, sind aber im Englischen gleich (oder andersherum), daher entstehen grammatisch falsche Sätze. *The mice eat the cheese. – Die Mäuse essen der Käse.*
- *The cat runs into the mouse. – Die Katze rennt in die Maus.* Zusammengehörige Wörter werden separat übersetzt. Ein Wort-für-Wort-Übersetzer weiß nicht, welche Wörter eine zusammengehörende Phrase bilden und übersetzt sie daher separat (und falsch). *The cat takes a walk. – Die Katze nimmt ein Spaziergang.*
- *The cat runs after the mouse. – Die Katze rennt hinterher der Maus.* Die Wortstellung im Satz ist falsch, weil sie bei der automatischen Übersetzung nicht verändert wird, obwohl sie sich in unterschiedlichen Sprachen unterscheidet.

Aufgabenteil 2

Getestet werden www.google.com, FreeTranslation.com und www.translate.ru.

„I run to the mice and the mice run to the cat.“ wird von Google tadellos zu „*Ich laufe zu den Mäusen und die Mäuse laufen zur Katze*“ übersetzt, allerdings interessanterweise nur, wenn man den Satz ohne den Punkt am Ende eingibt. www.translate.ru liefert hingegen die Übersetzung „*Ich laufe zu den Mäusen und den zur Katze geführten Mäusen*“ und macht es sich damit irgendwie komplizierter als nötig. In dieser Übersetzungsfunktion wird *and* nicht als Konjunktion zwischen zwei Sätzen, sondern als Verbindung zwischen zwei Objekten angesehen. Generell sind zwar beide Alternativen gültig, im vorliegenden Fall sollte „run“ jedoch nicht als Partizipialkonstruktion übersetzt werden.

FreeTranslation.com übersetzt den Satz zu „*Ich laufe zu den Mäusen und dem Mäusen Lauf zur Katze*“ und offenbart dadurch auch Schwierigkeiten mit dem *and* als Konjunktion zweier Sätze. Das zweite Objekt „*dem Mäusen Lauf zur Katze*“ gibt leider gar keinen Sinn. Hier wurde run mit der falschen Bedeutung übersetzt, außerdem ist der falsche Fall gewählt worden. Separat wird „*The mice run to the cat*“ von FreeTranslation.com allerdings richtig zu „*Die Mäuse laufen zur Katze*“ übersetzt.

„The cat runs into the mouse.“ übersetzt www.google.com zu „*Die Katze läuft in die Maus*“, was der (richtigen) Wort-für-Wort-Übersetzung entspricht – google.com ist *run into* offensichtlich nicht als Redewendung / zusammengesetztes Verb bekannt. www.translate.ru bringt den Leser mit „*Die Katze gerät in die Maus*“ zum Schmunzeln. Hier wurde *run into* mit einer anderen Bedeutung gespeichert. Für „*The cat runs into trouble*“ wäre diese Übersetzung angebracht, der Unterschied in der Bedeutung bleibt diesem automatischen Übersetzer verborgen. FreeTranslation.com hat keine Probleme mit diesem Satz und liefert „*Die Katze begegnet der Maus*“.

„The cat is a Manchester United fan.“ wird von google zu „*Die Katze ist ein Manchester vereinigter Ventilator*“ verarbeitet. Google wählt hier nicht nur die falsche Bedeutung des Wortes *fan*, sondern erkennt *Manchester United* auch nicht als Eigenname. „Wort-für-Wort“ ist diese Übersetzung durchaus korrekt. www.translate.ru beweist mit „*Die Katze ist ein Anhänger von Manchester United*“, dass man den Satz auch korrekt übersetzen kann, wohingegen FreeTranslation.com „*Die Katze ist ein Manchester Anhänger Hat Vereint*“ liefert. *Fan* wurde hier korrekt übersetzt, aber *united* nicht als Teil des Namens, sondern als Partizipform gewertet und als Perfektform übersetzt.

Aufgabenteil 3

In Teil 1 kann man bereits fundamentale Schwierigkeiten erkennen. Eine maschinelle Übersetzung muss daher

- … über grammatikalische Grundkenntnisse zu Konjugationen, Deklinationen (und Ausnahmen!) und Wortstellungen verfügen und grammatische Extra-Konstruktionen wie z.B. „*to want s.o. to do s.th*“ erkennen. Sie muss die grammatische Struktur (Syntax) auf Satzebene durchschauen.
- … Informationen über den Kontext eines Wortes innerhalb eines Satzes oder innerhalb eines semantisch zusammenhängenden Abschnitts besitzen. Woher weiß das Programm, dass es sich bei „*I want you to..*“ um die erwähnte grammatische Konstruktion und nicht um den üblichen Gebrauch von „*want*“ handelt? Fast alle der aufgeführten Fehler lassen sich auf mangelndes Kontextverständnis zurückführen. Die Analyse der Bedeutung (Semantik) eines Textes ist also eine zentrale Aufgabe einer automatischen Übersetzung.
- … Redewendungen und Sprichwörter beherrschen. Hier ist die Übersetzung manchmal sehr schwierig, häufig nur durch eine Redewendung in der anderen Sprache ungefähr möglich: „*It’s raining cats and dogs. – Es regnet in Strömen.*“

- . . . hin und wieder angepasst werden, da sich die menschliche Sprache laufend weiterentwickelt (Rechtschreibreform, Neuentstehung von Wörtern aufgrund von Neuentwicklungen, . . .).

Bewertungskriterien

Teil 1: Es sollten wenigstens drei Beispiele genannt werden, die mindestens drei Probleme aufzeigen wie z.B.:

- Bei mehreren Bedeutungen eines Wortes ist nicht klar, welche Bedeutung zu wählen ist; im Englischen kann u.a. ein Wort sowohl als Verb als auch als Substantiv verwendet werden.
- Wörter können nicht richtig konjugiert bzw. dekliniert werden.
- Begriffe, die aus mehreren Worten bestehen, werden falsch übersetzt.
- Unterschiede in der Wortstellung werden nicht berücksichtigt.

Teil 2: Es ist gefordert, dass drei Übersetzungsfunktionen getestet werden. Es muss darauf geachtet werden, dass die verschiedenen Funktionen auch wirklich unterschiedlich arbeiten und nicht auf den gleichen Mechanismen basieren. Viele Übersetzungsfunktionen im Internet verwenden letztlich Babelfish/Systran! Auch hier sollten mindestens drei Beispiele für problematische Übersetzungen angegeben werden. Die Vermutungen über die Ursachen der Übersetzungsfehler sollten sinnvoll und nachvollziehbar begründet sein.

Teil 3: Auch hier lautet die magische Zahl drei: Drei grundlegende Probleme der Sprachübersetzung sollten angegeben sein. Die Probleme einer reinen Wort-zu-Wort-Übersetzung aus Teil 1 dürfen hier nicht einfach wiederholt werden; es muss sich zumindest um Verallgemeinerungen handeln. Mit insgesamt mindestens drei Beispielen sollten alle drei Probleme veranschaulicht werden.

Diese Aufgabe scheint zunächst einfach zu bearbeiten, erfordert aber doch eine gewisse Mühe. Dies ist häufig nicht erkannt worden, so dass erstaunlich viele Einsendungen zu dieser Aufgabe relativ schlecht bewertet wurden.

Aus den Einsendungen: Perlen der Informatik

Allgemeines

Worte des Wettbewerbs: Lehrtaste, Programm-Ablaufsprodukoll, namifizierte Liste, FIFO-Stack

Dann folgt nur noch eine Endlos-Schleife, die alle wichtigen Funktionen des Überprüfens durchführt.

Um das Programm übersichtlich zu halten, habe ich auf eine Rekursion verzichtet.

Wobei zu beachten ist, dass das Programm nach ASCII-Standard entwickelt ist.

. . . , was sich aber durch Hardcoding im Quelltext jederzeit ändern läßt. *Nein, so haben wir uns flexible Programme nicht vorgestellt.*

Nur Windows, der endlich große Arbeitsspeicher und die Prozessorarchitektur beschränken die Leistungsfähigkeit meines Programms.

Die folgende while-Schleife müsste immer terminieren, obwohl der Beweis sehr schwer oder gar nicht möglich ist.

Diesen Array habe ich a getauft, das hat sich bei mir so eingebürgert, dass mein erster Array a heißt.

Raff

Ursprünglich wollte ich die Schüler ja mit Drogen und Waffen dealen lassen, aber ich hatte befürchtet, das könnte den Humor des Korrektors überstrapazieren.

. . . das Herzstück des Programms, die rekursive Funktion GoToWallStreet, . . .

Keuch

Keusch *mehrfach verwendeter Alternativtitel*

Natürlich können auch ein paar Handybesitzer sich absprechen und gleichzeitig ihre Handys vergessen.

Die Luft wird durch die Klasse Luft dargestellt, sie enthält ein Array Schadstoffe.

Nach dem Starten des Programms mit den Standardgrenzwerten läuft das Programm munter vor sich hin, piepsend und Nachrichten ausgebend.

Kommunikation mit den Schadstoffen

„Roter Alarm wegen des Schadstoffes Ozon!!! Ihr Leben steht auf dem Spiel!! Wollen Sie es vorzeitig beenden?“

Ein gutes Handy sollte seinem Besitzer möglichst viele Freiheiten lassen.

Gegenstrategie: Einstellen der Grenzwerte nur von zertifiziertem Personal. Begründung: Außerdem schafft sie Arbeitsplätze, und unsere Wirtschaft kann jeden noch so kleinen Aufschwung wärmstens gebrauchen.

Eine erste Variante, generellen Alarm auszulösen, die sehr gut funktionieren sollte, ist ein Giftgasanschlag auf die Funkzelle.

Da es nicht Ziel der Aufgabe ist, ein funktionstüchtiges Handy zu entwickeln, …

Grab

Mein Programm gräbt nicht nach Diamanten, sondern die Diamanten versuchen, sozusagen sich selbst auf kürzestem Weg auszugraben.

… bwinf als globaler Speicherplatz für den besten Weg. *Das wäre ein prächtiges Motto für den BWINF – leider steht bwinf hier wohl eher für „best way information“ oder etwas Ähnliches.*

Flitz

Als „Fähigkeiten“ bekommt der Fisch Zeichnen, sprich Bewegung.

An den Wänden dieses Aquariums werden die Fische wie Billard-Kugeln reflektiert, weil die Scheiben so gut geputzt sind, dass die Fische sie nicht sehen können, und da Fische sich nur 5 Sekunden etwas merken können, werden sie ziemlich oft gegen die Scheibe prallen.

Ein Fisch könnte eine „unendliche“ Geschwindigkeit erreichen (falls er genügend Freunde/Feinde hat), die ja aus Prinzip nicht gehen kann.

Menschen halten z.B. ebenfalls eine Mindestentfernung von ca. 0,5 m ein. *Mag sein, aber keine Regel ohne Ausnahmen.*

Die Goldfische besitzen 3 verschiedene Geschwindigkeitsstufen.

Aus dem vorher gewählten Verhältnis wird nun durch die aus der Mathematik bekannte Formel die Anzahl möglicher geliebter Fische erstellt.

Der Fisch darf nicht durch’s Glas schwimmen. Quanteneffekte werden vernachlässigt.

Bezüglich der Gemochten muss Folgendes gelten: Je größer die Distanz, desto größer die Anziehungskraft … diese Eigenschaft entspricht in etwa dem menschlichen Wunsch nach dem, was man nicht hat.

Ein gewisses Maß an Selbstverliebtheit wird dabei nicht ausgeschlossen. Der Fisch hält also auch ab und zu mal inne und erfreut sich seiner selbst.

Folgende Aspekte werde ich im Modell berücksichtigen: Kollision Fisch - Teich, Kollision Fisch - Fisch, Wasserwiederstand

Babbel

Dennoch gibt es auch viele Sätze, welche sich durch das Verfahren korrekt übersetzen lassen:
… the cat runs the mouse after → Die Katze rennt der Maus hinterher.

Und falls dies ein Katzenliebhaber liest – auch bei Schweinen wird „essen“ übersetzt

Vermutlich liegt hier eine lexikalische Schwachstelle vor, in der die Übersetzung „Tisch“ für das Wort „table“ nicht gut genug involviert ist. Es liegt also eine Art von referentieller Ambiguität vor, die wie quasi alle Übersetzungsfehler nur durch eine bessere Erschließung der Bedeutung eines Satzes umgangen werden kann.

A postcard was sent by Karl → Ein Postkarte war sandtet beim Karl
[…] schießt hier den Vogel ab: Artikel nicht dekliniert, Verben nicht richtig konjugiert und falsche Präposition, mehr tut nicht gehen tun.

3. Satzzeichen: Ich bin versucht zu schreiben „wenn ich es schon nicht korrekt schaffe, wie soll es dann der Computer schaffen“.

„The mouse eats mice“ wird von der Wort-für-Wort-Übersetzung fälschlicherweise zu „Die Maus frisst Mäuse“ übersetzt. Richtig übersetzt müsste es „Die Maus frisst Mais“ heißen.