Institut für Informatik — TU München
Lehrstuhl für Informatik mit Schwerpkt. Wiss. Rech.
Prof. Dr. H.-J. Bungartz, Dipl.-Inf. M. Moltenbrey,
Dipl.-Inf. D. Pflüger

WS 2008/09

PSE Verkehrssimulation
Aufgabenblatt 5: Aktivitätspläne

Ausgegeben 08.01.2009 Abgabe 21.01.2009 12 Uhr

Aktivitätspläne

Ihr bisheriger Simulator ist mittlerweile in der Lage im Rahmen seiner Möglichkeiten realistisch den Straßenverkehr zu simulieren. Momentan werden allerdings die Fahrzeuge nach wie vor rein zufällig generiert und auf dem Verkehrsnetz platziert. Dies führt zu netten Simulationen, die aber nicht unbedingt die Realität widerspiegeln. Im realen Leben haben Fahrer bestimmte Ziele, die sie erreichen wollen. Dabei gehen sie von bestimmten Ausgangspunkten aus. So verlässt bspw. ein Arbeitnehmer morgens sein Haus, um zu seinem Arbeitsplatz zu fahren. Dieses Verhalten soll nun im Rahmen dieses Aufgabenblattes in Ihren Simulator integriert werden.

Aufgabe 10 „Parser für OD-Matrizen“

Bei einem Quelle-Ziel-Paar spricht man auch von einer OD-Beziehung (= *origin-destination*). Eine solche Beziehung ist immer nach dem gleichen Prinzip aufgebaut:

$$\text{Quellknoten} \rightarrow \text{Zielknoten} = \#\text{Fahrzeuge}$$

Sie gibt an, dass sich innerhalb des betrachteten Simulationszeitraums ausgehend von dem Quellknoten hin zu dem gegebenen Zielknoten die genannte Anzahl von Fahrzeugen bewegen.

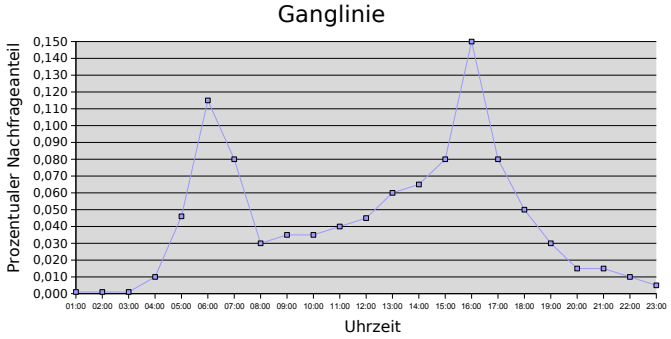
Um diesen Fall in unserem Simulator zu behandeln, müssen Sie ein neues XML-Format einführen und einen Parser dafür entwickeln.

```
<Activities>
<simulationTime> </simulationTime>
<od>
   <sourceNode> </sourceNode>
   <destNode> </destNode>
   <vehicles> </vehicles>
</od>
...
<od>
   <sourceNode> </sourceNode>
   <destNode> </destNode>
   <vehicles> </vehicles>
</od>
</Activities>
```

Durch das Einfügen des Parameters `destNode` sind wir in der Lage beliebige Quelle-Ziel-Beziehungen zu modellieren. Die Fahrzeuge werden nun für die gegebenen Quelle-Ziel-Paare zufällig über den Simulationszeitraum erzeugt.

Die Daten der ODs geben folglich die Anzahl der Fahrzeugfahrten im Simulationszeitraum an. In unserem Fall betrachten wir einen Simulationszeitraum von 24h. Natürlich ist die Anzahl der Fahrzeuge über den gesamtem Simulationszeitraum nicht gleichverteilt. Als Stichwort sei hier beispielsweise die Rush-Hour genannt.

Von	Bis	Faktor
00:00	01:00	0,001
01:00	02:00	0,001
02:00	03:00	0,001
03:00	04:00	0,001
04:00	05:00	0,010
05:00	06:00	0,046
06:00	07:00	0,115
07:00	08:00	0,080
08:00	09:00	0,030
09:00	10:00	0,035
10:00	11:00	0,035
11:00	12:00	0,040
12:00	13:00	0,045
13:00	14:00	0,060
14:00	15:00	0,065
15:00	16:00	0,080
16:00	17:00	0,150
17:00	18:00	0,080
18:00	19:00	0,050
19:00	20:00	0,030
20:00	21:00	0,015
21:00	22:00	0,015
22:00	23:00	0,010
23:00	00:00	0,005



Eine entsprechende Steuerung des Farhrzeugaufkommens kann mt einer sogenannten Ganglinie erreicht werden. Diese gibt für bestimmte Zeiträume den prozentualen Anteil am Gesamtaufkommen an.

a) Entwerfen und Implementieren Sie einen Parser für OD-Matrizen. Überlegen Sie sich geeignete Datenstrukturen, die einen schnellen Zugriff auf die einzelnen Komponenten erlauben und den Speicherverbrauch möglichst minimal halten.

b) Erweitern Sie Ihren Simualator dahingehend, dass bei der Generierung von Fahrzeugen an den Quellknoten Ganglinien berücksichtigt werden können.

Aufgabe 11 „Ermittlung des Weges“

Als nächsten Schritt müssen wir uns damit beschäftigen wie die Fahrzeuge von dem Quellknoten zu dem Zielknoten gelangen. Dem Verkehrsnetz liegt ein Graph zugrunde. Wir betrachten lediglich einen vereinfachten Fall: Fahrzeuge reagieren nicht auf Staus oder andere Probleme auf ihrem Weg. Man könnte es folgendermaßen auffassen: Jedes Fahrzeug berechnet den zu seinem OD-Paar passenden kürzesten Weg und fährt diesen anschließend stur ab – unabhängig davon was auf dem Weg passieren mag. Allerdings ist diese zugrunde liegende Idee zu ineffizient. Es wird Ihre Aufgabe sein, sich eine effizientere Strategie zu überlegen und zu implementieren.

Die Kurzwegsuche soll über den bekannten Algorithmus von Dijkstra erfolgen, der im Folgenden im Pseudocode aufgeführt ist:

```
for all v aus V do
    Dist(v) = unendlich;
    Pred(v) = empty;
end for

Dist(s) = 0;
Pred(s) = s;
U = {s};
while (!empty(U)) do
    u = extract_min(U);
    U = U - {u};
    for all (u,v) aus E do
        if (Dist(u)+c(u,v) < Dist(v)) then
          Dist(v) = Dist(u) + c(u,v);
          Pred(v) = u;
          if v nicht aus U then U = U + {v};
        end if
    end for;
end while
```

Listing 1: Dijkistras Algorithmus zur Ermittlung kürzester Wege

Der berechnete kürzeste Weg ist der abzufahrende Plan für ein Fahrzeug. Dieser Plan kann bspw. als eine Folge von Kanten oder Knoten gespeichert werden. Nachdem ein Fahrzeug generiert und sein Plan bekannt ist, so beginnt es diesen Plan abzuarbeiten. Hat es sein Ziel erreicht, wird das Fahrzeug aus der Simulation genommen. Als Kantengewicht betrachten wir lediglich die Länge der Strecke.

a) Entwickeln Sie eine Strategie wie sich die kürzesten Wege für die Fahrzeuge effizient mittels Dijkstra berechnen lassen. Berechnen Sie anschließend die kürzesten Wege und generieren Sie dadurch die Aktivitätspläne für die einzelnen Fahrzeuge. Überlegen Sie sich dabei wie sich die Pläne möglichst effizient speichern lassen, sowohl was ihren Speicherbedarf als auch ihre Zugriffszeiten angeht. (Hinweis: Es ist wünschenswert, wenn nicht mehrmals der gleiche Weg berechnet werden müsste.)

b) Generieren Sie nun im Laufe Ihrer Simulation die Fahrzeuge zufällig verteilt über den gesamten Simulationszeitraum an den entsprechenden Quellknoten. Lassen Sie nun die Fahrzeuge ihre jeweiligen Pläne abfahren und nehmen Sie sie bei Erreichen des Zielknotens aus der Simulation heraus.

c) Wiederholen Sie nun die Simulationsläufe aus Teilaufgabe b) unter Verwendung einer Ganglinie. Was fällt Ihnen auf? Geben Sie Ihre Beobachtungen schriftlich ab.

d) In der Realität wird ein Fahrer nicht immer den kürzesten Weg suchen und verwenden. Dies kann verschiedene Gründe haben. Erweitern Sie daher Ihren Simulator um die Möglichkeit den schnellsten Weg zu berechnen. Überlegen Sie sich dafür einen geeigneten Algorithmus.

e) Verwenden Sie nun für die Berechnung des kürzesten Weges anstelle des Algorithmus von Dijkstra den A*-Algorithmus.

f) Welche Beobachtung machen Sie bei der Verwendung der unterschiedlichen Suchstrategien in bezug auf das Simulationsverhalten?

Aufgabe 12 „Erweiterung der Visualisierung“

In dieser Aufgabe sollen die grafische Benutzeroberfläche erweitert und die Visualsierung um eine weitere Komponente ergänzt werden.

a) Erweitern Sie die grafische Benutzeroberfläche um Menüstrukturen zum Laden von Netz- und Nachfragedateien.

b) Ergänzen Sie die Visualisierung um eine Möglichkeit ein Hintergrundbild für das Stadtnetz von Karlsruhe darzustellen. Achten Sie dabei auf die korrekte Orientierung und Skalierung des Hintergrundbildes in Bezug auf das Verkehrsnetz.

Viel Spaß!