Universität Augsburg
Institut für Angewandte Informatik
Lehrstuhl für Organic Computing
Prof. Dr. rer. nat. Jörg Hähner

Dipl. Math. Stefan Rudolph
Eichleitnerstr. 30, Raum 502
stefan.rudolph@informatik.uni-augsburg.de
Tel.: +49 821 598 - 4630

Organic Computing (WS 2013/14)

Aufgabenblatt 1

Abgabe bis 28.10.2012, 23:59 Uhr

Aufgabe 1: Ant Colony Optimization

In der Vorlesung haben Sie das Verhalten von Ameisenkolonien kennengelernt. Die Ernteameisen sind für das Explorieren der Umgebung, das Auffinden von Futterquellen und dem Einbringen der Nahrung zum Nest verantwortlich. Die Chance eine Quelle zu finden ist dabei von der Größe der Quelle und der Komplexität des Weges dorthin und zurück abhängig. Um den Weg zur Futterquelle und zurück zu finden, bedienen sich die Ameisen dabei des Prinzips der Selbstorganisation durch das Absetzen von Pheromonen. Diese Pheromone können von den Ameisen aufgenommen werden und dienen so der Wegfindung. Durch die Akkumulation der Pheromone werden häufig besuchte Pfade stärker und bilden somit den kürzesten Pfad zu den entsprechenden Zielen aus.

Aufgabe 1a: Algorithmus

Im Folgenden soll dieser Ameisenalgorithmus auf das *Traveling Salesman Problem* (TSP) angewandt werden. Dieses besteht aus der Aufgabe, dass ein Reisender n Städte nacheinander genau 1 Mal besuchen und am Ende wieder bei seiner Startstadt ankommen möchte. Die Länge seiner Route soll dabei minimal sein. Da diese Aufgabe ein NP-schweres Problem darstellt, existiert dafür kein Algorithmus, der dieses exakt in polynomialer Zeit lösen könnte. Die naive Lösung, bei der alle zyklischen Permutationen berechnet werden, deren Gesamtlängen ermittelt und daraus die günstigste Route gewählt wird, ist nicht praktikabel. Da es $n!$ viele Routen gibt, liegt die Laufzeit dieses Verfahrens in $O(n!)$ (schlimmer als NP).

Der Ant-Colony-Optimization-Algorithmus (ACO) zur Anwendung auf das TSP ist wie folgt aufgebaut:

```
Initialisiere Parameter und Pheromone
while (Abbruchkriterium noch nicht erfüllt) do
    Erzeuge Zwischenlösungen
    Berechne Kosten der Lösungen
    if (Neue beste Lösung gefunden)
        Aktualisiere beste Lösung
    Aktualisiere Pheromonspuren
end
Gib beste Lösung aus
```

Aufgabe 1b: GUI

Fügen Sie dem Interface die Buttons `setup` und `go` hinzu. Verwenden Sie diesmal einen weiteren `go`-Button, der es erlaubt die Simulation schrittweise ausführen zu lassen. Des Weiteren werden drei Slider zur Anpassung verschiedener Attribute des Modells benötigt. Fügen Sie einen Slider hinzu, der es erlaubt, die Anzahl der maximalen Iterationen des Optimierungsverfahrens einzustellen. Der Algorithmus bricht ab, wenn er die definierte Anzahl an maximalen Iterationen erreicht hat. Über einen weiteren Slider wird der Grad der Verdunstung ρ der Pheromone in Prozent bestimmt. Die Anzahl der initial erstellten Städte soll ebenso frei wählbar sein. Diese sollen zufällig in der Welt erzeugt werden.

Aufgabe 1c: Initialisierung

Für die Kodierung der Entfernung zwischen den Städten und der Pheromonintensität der, diese Städte verbindenden Straßen, sollen Matrizen verwendet werden. Greifen Sie dazu auf die NetLogo-Erweiterung `matrix` zurück. Eine Möglichkeit, die Städte und Ameisen zu kodieren, wäre es, dafür Rassen anzulegen und die Straßen als Links zwischen Städten zu erzeugen. Implementieren Sie nun die `setup`-Funktion, in der die Welt samt der Städte, der Straßen und Ameisen erzeugt wird. Initial soll auf jeder Stadt genau eine Ameise erzeugt werden. Jede Stadt soll eine direkte Verbindung zu allen anderen Städten besitzen. Initialisieren Sie auch die Matrizen. Die Einträge der Pheromonmatrix sollen auf 1 gesetzt werden, die Einträge der Entfernungsmatrix auf die entsprechenden Distanzen zwischen den Städten.

Aufgabe 1d: ACO

Nach dem Sie die Simulation initialisiert haben, soll nun das Suchverhalten der Ameisen umgesetzt werden. Jede Ameise soll alle Städte genau einmal besuchen. Merken Sie sich dazu die besuchten Städte in einer Tabuliste. Eine Route endet mit der Ankunft in der Startstadt. Benutzen Sie für dieses Aufgabenblatt kein probabilistisches Auswahlverfahren um zu entscheiden, wohin sich die Ameise bewegt, sondern wählen Sie immer die Stadt mit der höchsten Wahrscheinlichkeit.

Die Auswahl der für Ameise k nächsten Stadt j mit Wahrscheinlichkeit p zum Zeitpunkt t wird durch folgende Formel bestimmt:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} * [\eta_{ij}]^{\beta}}{\sum_{l \notin tabu_k} [\tau_{il}(t)]^{\alpha} * [\eta_{il}]^{\beta}}, j \notin tabu_k, p \in [0,1]$$

mit

- $\tau_{ij}t$ Pheromonintensität von Stadt i nach j
- α Pheromonrelevanz $\in [0,1]$
- η_{ij} Kehrwehrt der Distanz zwischen Stadt i und j, also η_{ij} = $\frac{1}{d_{ij}}$ mit d_{ij} Distanz zwischen Stadt i und j
- β Distanzrelevanz $\in [0,1]$

Nachdem jede Ameise eine Tour beendet hat (wieder an ihrem Startpunkt angekommen ist), werden die Pheromone aller Straßen aktualisiert:

$$\tau_{ij}(t+1) = (1-\rho) * \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t), \rho \in]0,1]$$

wobei

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{1}{L^k(t)}, \text{falls } i,j \text{ von Ameise } k \text{ benutzt wurden} \\ 0, \text{sonst} \end{cases}$$

Das bedeutet, dass für jede Straße der in dieser Iteration besuchten Route die Pheromonintensität auf Basis der alten Intensität und des Verdunstungsfaktors neu berechnet wird. $L^k(t)$ ist hier einfach die Gesamtlänge der abgelaufenen Route.

Aufgabe 1e: Situation erstellen

Implementieren Sie abschließend eine Funktion, die 5 Städte auf den Punkten $(0,5)$, $(0,-15)$, $(21,3)$, $(6,22)$ und $(10,5)$ erzeugt und diese jeweils mit Straßen untereinander verbindet, so dass jede Stadt mit jeder anderen verbunden ist. Fügen Sie ebenfalls einen Switch hin, mit dem bestimmt werden kann, ob beim Setup diese Situation oder eine zufällig Verteilung der Städte aus Aufgabe 1a erzeugt werden soll.

Aufgabe 1f: Evaluation

Lassen sie sich in jedem Tick die beste Route jeder Ameise samt ihrer Distanz ausgeben. Was ist die schnellste Route, die Sie für die Szenerie aus Aufgabe d finden können?

Überlegen Sie, wie viele Routen in der naiven Variante ausprobiert werden müssten, um sicher die kürzeste Route zwischen 30 Städten zu finden. Wie lange würde das Verfahren dafür benötigen, wenn 1 Million Routen pro Sekunde berechnet werden können? Was für eine Laufzeit hat der ACO-Algorithmus?

Was bedeutet $\alpha = 0$? Welchem Suchverfahren entspricht dieses Verhalten? Wie ist das Verhalten, wenn $\beta = 0$ gesetzt wird?

Aufgabe 1g: Abgabe

Ihre Abgabe umfasst den Programmcode nach Aufgabenteil e und die schriftliche Beantwortung der Fragen in Aufgabenteil f. Verfassen Sie den schriftlichen Teil bitte auf Folien, so dass Sie eine Grundlage für eine mögliche Präsentation in der Übungsstunde haben. Senden Sie Ihre Abgabe bitte bis spätestens Montag, den 28.10.2012, 23:59 Uhr an johannesjungbluth@googlemail.com.