Daniel Radeloff
Matr.# 1627725
radeloff@gmx.de

Ameisenalgorithmen

-Seminararbeit-

Proseminar ‚Adaptive Systeme'
PD Dr. R. Brause

INHALT

Übersicht

Übersicht

In den folgenden Ausführungen soll das Thema ‚Ameisenalgorithmen' dargestellt werden. Zunächst möchte ich in Abschnitt eins die Herkunft dieses Optimierungsverfahrens klären, das seine Inspiration in der Natur, genauer im Verhalten von Insektenschwärmen, findet. Hierzu wird das gemeinschaftliche Optimierungsverhalten der Ameisen beleuchtet und in einem mathematischen Modell beschrieben. Ein zentraler Punkt ist hierbei die Kommunikation ‚sozialer Insekten', die über einen lokalen Botenstoff funktioniert.

Daran anschließend wird in Abschnitt zwei dieser natürliche Optimierungsprozess in ein System virtueller Ameisen übersetzt. Dabei werden den künstlichen Ameisen, um ein besseres Ergebnis zu erzielen, einige neue Eigenschaften hinzugefügt, die in der Natur nicht zu finden sind (Abschnitt 2.1).

Exemplarisch wird in Abschnitt 2.2 AntSystem (AS) als ein grundlegender und übersichtlicher Vertreter der ‚ant colony optimization'-Algorithmen eingeführt, später soll AntColonySystem (ACS) als weiterentwickeltes, effizientes Modell vorgestellt werden (Abschnitt 2.3). Beide Algorithmen werden besprochen, indem sie beispielhaft zur Lösung des Traveling Salesman Problems (TSP) angewendet werden.

Abschließend wird die Anwendung von Ameisenalgorithmen auf dynamische, verteilte Systeme skizziert (Abschnitt 2.4).

1. Soziale Insekten als Prototypen von Optimierungsalgorithmen

1.1 Kennzeichen Sozialer Insekten

Soziale Insekten nennt man jene Insekten, die sich zu meist gegenseitigem Nutzen staatenbildend verhalten. Ein weiteres Charakteristikum dieser Gemeinschaften ist die Arbeitsteilung, die sich in Fortpflanzung, Verteidigung und Nahrungsbeschaffung gliedert. Typische Vertreter sind Ameisen, Termiten, Bienen und Wespen.

Insekten sind im gemeinschaftlichen Verbund zu beeindruckenden komplexen Leistungen fähig, obwohl die Individuen des Staates nur über geringe kognitive Ressourcen verfügen. So sind beispielsweise die australischen Kompasstermiten in der Lage, einen bis zu vier Meter hohen wie tiefen Bau zu errichten, der in Nord-Süd-Richtung angelegt wird. Die Mittagssonne erreicht durch diese Bauweise nur die Schmalseite, die Morgen- und Abendsonne die Breitseite des Gebäudes. Hierdurch wird eine konstante Innentemperatur erreicht.

Ameisenalgorithmen wurden durch die Beobachtung von Ameisen inspiriert, welche ebenfalls die Fähigkeit besitzen, komplexe Probleme gemeinschaftlich zu lösen: Im

Verbund finden Ameisen bei der Nahrungssuche stets den kürzesten Weg zwischen Futterquelle und Nest. Der Verlauf der ‚Ameisenstraße' ist also das Produkt einer Kostenminimierung, die im folgenden genauer betrachtet werden soll.

1.2 Indirekte Kommunikation: Wegweiser auf Pheromonbasis

Ist eine Futterquelle vom Nest aus auf mehreren Wegen zu erreichen, so finden Ameisen schnell die kürzeste Verbindung. Wird eine Nahrungsquelle neu erschlossen, benutzen Ameisen in einer Initialphase vorübergehend verschiedene Wege, die schon nach einigen Minuten zugunsten des kürzesten Weges aufgegeben werden. Interessant an diesem Optimierungsprozess ist, dass die Information über Wegstrecken nicht an einem Punkt zusammengetragen und von einer zentralen Instanz ausgewertet wird, sondern vor Ort in Form eines Botenstoffes gespeichert wird. Ameisen markieren den Weg, den sie benutzen, mit Pheromonen. Diese werden nicht gezielt ausgeschieden, vielmehr werden sie permanent als kleine Depots auf dem Pfad hinterlegt, so dass ein Pheromon-Pfad entsteht, der von anderen Ameisen, die über diese Depots laufen, wahrgenommen wird. Wichtig ist zum einen, dass Pheromone den einzigen für die hier dargestellte Streckenoptimierung wichtigen Sinneseindruck liefern, zum anderen, dass Pheromone nur lokal abgelesen werden können, d.h. die Ameise erhält lediglich Informationen aus ihrer unmittelbaren Umgebung. Angenommen wird stets, dass jede Ameise die gleiche Pheromonmenge hinterlässt und dass Verdampfung vernachlässigbar ist. Diese Kommunikationsform, die im folgenden als *stigmergy* bezeichnet werden soll, zeichnet sich also durch folgende Punkte aus: (i) indirekte Kommunikation über (ii) ein physikalisches lokales Informationskorrelat, dass (iii) ausschließlich lokal ablesbar ist. *Stigmergy* ist ein konstituierendes Merkmal aller ‚ant colony optimization (ACO)'-Algorithmen, die in späteren Abschnitten behandelt werden.
Wie die hinterlegte Information zur Entscheidungsfindung der einzelnen Ameise führt, soll im sich anschließenden Kapitel erläutert werden.

1.3 Probabilitische Entscheidungsfindung

Welchen Weg eine Ameise einschlägt, hängt von der Pheromonkonzentration der einzelnen Wege ab: Ein Weg mit höherer Konzentration des Botenstoffs wird mit größerer Wahrscheinlichkeit gewählt als einer mit niedriger Konzentration. Dieser probabilistische Prozess wird im folgenden anhand einer Ameise, die an einer Weggabel eine Entscheidung treffen muss, beschrieben. Gegeben sei ein Pfad, der durch einen soliden

Gegenstand verlegt wurde. Folglich bestehen die zwei Möglichkeiten, den Gegenstand an seiner rechten oder linken Außenseite zu umwandern. Bereits m Ameisen sollen die andere Seite des Hindernisses erreicht und dabei Pheromone hinterlassen haben. Hierbei haben sich L_m für den linken, R_m für den rechten Pfad entschieden. Die Entscheidungsmechanismen der Ameisen werden gut abbildet, wenn man annimmt, dass sich die Wahrscheinlichkeit $P_r(m)$, mit der sich die Ameise (m+1) für den linken Pfad entscheidet,

$$P_L(m) = \frac{(L_m + k)^h}{(L_m + k)^h + (R_m + k)^h} \qquad (1)$$

beträgt, während die Ameise m+1 mit der Wahrscheinlichkeit $P_R(m) = 1 - P_R(m)$ den linken Pfad wählt. Weist man den Parameter h und k, anhand derer das Modell den experimentell gewonnene Daten angeglichen wird, die Werte 2 bzw. 20 zu, so werden die empirischen Beobachtungen am besten beschrieben.

Zwei wichtige Prinzipien sind für den Erfolg der pheromonbasierten Optimierung zuständig. Zum einen der oben beschriebene *autokatalytische Mechanismus*, welcher eine positive Rückkopplung beschreibt, die darin besteht, dass Pfade, die häufig benutzt werden, durch eine hohe Pheromonkonzentration mehr Ameisen anziehen als wenig benutzte Wege. Zum anderen ein Prinzip, das als *implicit solution evaluation* bezeichnet wird. Dieses bezeichnet das Phänomen, dass ein kürzerer Pfad attraktiver ist, da er von den Ameisen, die ihn benutzen auch schneller wieder verlassen wird, was einer schnelleren Pheromonanreicherung am Ausgang des Weges gleich kommt. Entgegenkommende Ameisen wählen nun vermehrt den kürzeren Weg.

In späteren Abschnitten wird ein weiterer Mechanismus eingeführt, der die Pheromonverdampfung beschreibt und als Gegenspieler des *autokatalytischen Mechanismus*' benötigt wird. Letzterer zeichnet sich dadurch aus, dass gute Lösungswege auf neue Ameisen anziehend wirken, birgt aber das Risiko, dass in einer Gruppe schlechter Ameisen durchschnittliche Lösungen durch die besten Vertreter der Gruppe gebahnt werden. Diese vorschnelle Konvergenz auf ein lokales Maximum wird im folgenden *Stagnation* genannt. *Stagnation* kann verhindert werden, indem die Differenz

der Pheromonkonzentrationen zwischen hochfrequentierten und niedrigfrequentierten Wegstrecken durch einen Mechanismus der Pheromon-Verdunstung begrenzt wird.

2. Künstliche Ameisen und Ameisenalgorithmen

2.1 Künstliche und reale Ameisen: Unterschiede und Gemeinsamkeiten

ACO-Algorithmen nutzen zum einen die Fähigkeiten aus, die bei Ameisenkolonien beobachtet und in den vorangehenden Abschnitten dargestellt wurden. So werden kleine, einfache Einheiten (künstliche Ameisen) erzeugt, die in Form von *stigmergy* kommunizieren. Zum anderen sind künstliche Ameisen nicht nur Abstraktion ihrer realen Gegenspieler, sondern sie zeichnen sich durch Eigenschaften aus, die wir bei echten Ameisen nicht finden. Diese dienen der Adaption an die Probleme diskreter Optimierung und führen zu einem schnelleren und besseren Ergebnis.

Realen und künstlichen Ameisenkolonien ist gemein, dass sie aus vielen, unabhängigen, einfachen Einheiten bestehen, die sich in der Lösungsfindung eines globalen Problems gegenseitig unterstützen. Dabei bewegen sich beide Ameisenformen ohne zu springen fort: nur angrenzende Gebiete werden betreten. Eine künstliche Ameise, die beispielsweise auf einem Graphen G=(N,E) ausgesetzt wurde, der aus Knoten (N) und Kanten (E) besteht, darf von einem Knoten nur dann zu einem Nachbarknoten wechseln, wenn beide durch eine Kante verbunden sind. *Stigmergy* wird durch künstliche Ameisen unterstützt, indem –am Beispiel des Graphen- einer lokalen Variable ein Wert, welcher der Pheromonkonzentration entspricht, zugewiesen wird. Zusätzlich kann diese lokale, beispielsweise den Knoten des Graphen zugehörige Variable nur ausgelesen oder verändert werden, wenn die künstliche Ameise den entsprechenden Knoten besucht.
Eine weitere Analogie betrifft die Zielsetzung: Reale Ameisen suchen nach einer Lösung des speziellen Problems der Wegstreckenminimierung, ihre künstlichen Verwandten sollen allgemein ein Kostenminimum eines meist diskreten Problems aufspüren.

Unterschiede bestehen zunächst darin, dass künstliche Ameisen in einer diskreten Welt leben. Weiterhin verfügen sie im Gegensatz zu realen Ameisen über ein Gedächtnis, das die zurückliegende Route speichert. Auch die Pheromonausschüttung ist nicht identisch mit dem natürlichen Vorgang: Die Menge des Botenstoffs, der von einer künstliche Ameise deponiert wird, ist abhängig von der Qualität der gefundenen Lösung. Kürzere Wege werden beispielsweise belohnt, indem eine der Wegstrecke reziproke Menge

Pheromon hinterlegt wird. Einige Ameisenalgorithmen sehen nicht vor, dass künstliche Ameisen die Pheromonkonzentration der einzelnen Wege Schritt für Schritt auf den neuesten Stand bringen, sondern sie aktualisieren die Pheromon-Variable aller von einer Ameise benutzten Wege erst, sobald diese eine Lösung gefunden hat. Weiterhin sind Algorithmen bekannt, welche die einzelnen Ameisen mit zusätzlichen Leistungsmerkmalen wie lokalen Optimierungsmechanismen ausrüsten.

2.2 Ant System (AS)

2.2.1 Optimierungsprinzipien und Implementierung

In diesem Abschnitt soll Ant System (AS) dargestellt werden. Obwohl AS vergleichsweise unbefriedigende Lösungen für komplexe Probleme liefert, ist dieser erste ACO-Algorithmus interessant, weil er einerseits übersichtlich ist, andererseits als Grundlage verschiedener erfolgreicher Weiterentwicklungen diente.

Das Traveling Salesman Problem[1] (TSP) gilt als das erste Anwendungsgebiet von AS, da es einerseits ein gut untersuchtes Gebiet darstellt, andererseits den didaktischen Vorteil hat, dass es wie die reale Ameisenkolonie die Suche nach einer kürzesten Verbindung zum Ziel hat.

AS hat folgende, in Abb. 1 dargestellte Struktur. Der Algorithmus durchläuft $t_{\max}$ Iterationen, innerhalb derer eine Generation Ameisen der Stärke *m* erschaffen wird. Jede Ameise *k* wird zufällig einem Ursprungknoten zugewiesen, von dem aus sie *n* Schritte ausführt, welche nach einer probabilistischen Entscheidungsregel abhängen. Dabei speichert die künstliche Ameise die Reihenfolge der besuchten Städte (*tabu list*). Dies ist nötig, da jede Stadt nur einmal besucht werden darf. Erst nachdem alle Ameisen einer Generation ihr Ziel erreicht haben, erfolgt die Markierung der benutzten Wege, indem Knoten-lokale Pheromonvariablen aktualisiert werden (der Einfachheit halber sollen von nun an die Bezeichnungen artifizieller Vorgänge denen natürlicher entsprechen: Folgend soll also weiterhin von Pheromonausschüttung gesprochen werden). Für dieses verzögerte Markieren ist ein erneuter Zugriff auf die bereits erwähnte *tabu list* nötig, wobei die Pheromonausschüttung unter Berücksichtigung der Länge des Gesamtweges erfolgt. Kürzere Wege erhalten so einen größeren Pheromonbetrag. Nach

[1] Das Traveling Salesman Problem bezeichnet das Problem, eine schnellste Tour durch gegebene Städte zu finden, die paarweise durch Wege bestimmter Länge verbunden sind. Eine Stadt darf dabei nicht zweimal besucht werden. Im vorliegenden Text wird von einem *symmetrischen* TSP ausgegangen, d.h. alle Wege haben für beide Richtungen die gleichen Streckenkosten. Das TSP wird häufig durch einen Graphen G=(N,E) beschrieben, dessen Knoten N die Städte und Kanten E die Wege repräsentieren. Dabei entspricht d_{ij} der Länge der Kante, welche die Knoten i, j $\in$ N verbindet.

diesem Schritt sterben die Ameisen der aktuellen Generation und eine neue Generation Ameisen wird in der nächsten Iteration des Algorithmus erzeugt, die auf ein System markierter Wege trifft. Nach dem letzten Durchlauf des Algorithmus' wird der Weg, auf den die Ameisen der letzten Generation mehrheitlich konvergieren, als Lösung ausgegeben.

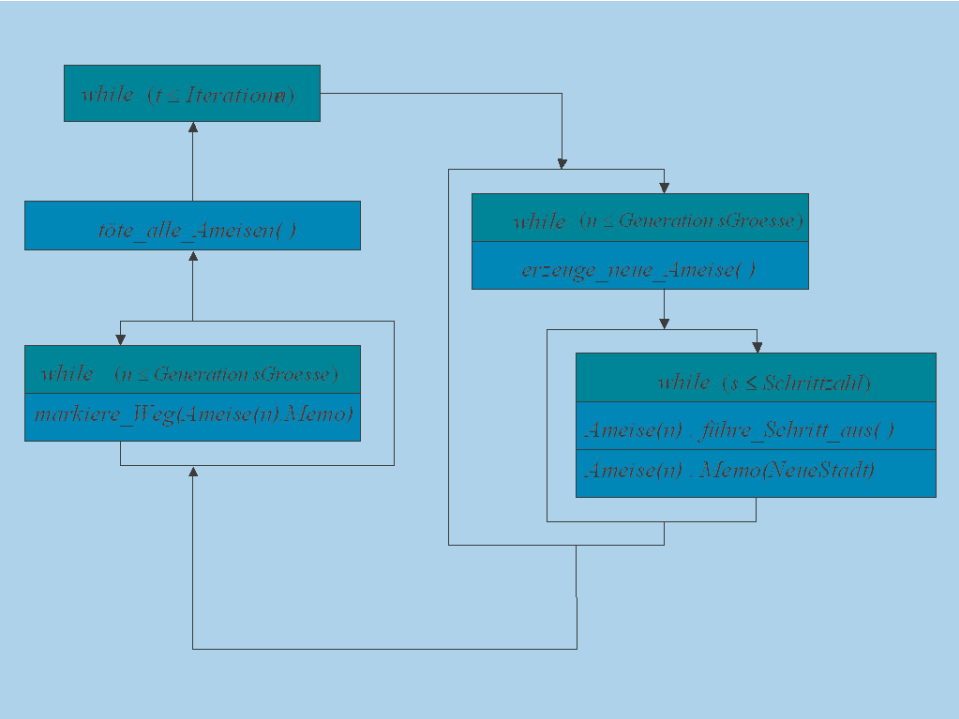

Abb. 1: Struktur von Ant System als Pseudocode

Die probabilistische Entscheidung einer Ameise der Generation t, befindlich auf einem Knoten i, den Knoten j als nächstes zu besuchen, hängt einerseits von der Pheromonkonzentration τ_{ij} und der Länge d_{ij} des verbindenden Weges ab, andererseits von Länge und Pheromonkonzentration der Wege die zu den restlichen Nachbarknoten N_i führen.

In einer Entscheidungstabelle A_i sei allen Wegen ein relatives Gewicht a_{ij} zugeordnet, welches sich errechnet als:

$$a_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in \mathrm{N}_i}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} \qquad (2)$$

Weiterhin ist $\eta_{ij} = 1/d_{ij}$ ein heuristischer Ansatz, der angibt, wie attraktiv die Wegstrecke ij für eine Ameise ist, wenn bisher noch keine Pheromone ausgeschüttet worden wären. α und β sind Parameter, die den Einfluss des heuristischen Wertes bzw. der Pheromonkonzentration festlegen. Vereinfachend entspricht β dem absoluten Einfluss des Prinzips der *implicit solution evaluation*, während α das absolute Gewicht des *autokatalytischen Mechanismus'* beschreibt. (Entsprechend würde der Fall β=0 zur *Stagnation* führen.)

Die Wahrscheinlichkeit, mit sich eine Ameise k innerhalb der t-ten Iteration des Algorithmus dafür entscheidet, von Knoten i nach Knoten j zu bewegen, beträgt:

$$p_{ij}^{k}(t) = \frac{a_{ij}(t)}{\sum_{l \in \mathrm{N}_i^k} a_{il}(t)} \qquad (3),$$

wobei N_i^k die Menge aller Nachbarknoten von Knoten i darstellt, die von Ameise k noch nicht besucht wurden.

Nachdem alle Ameisen ihre Tour abgeschlossen haben, werden die Wege nachträglich mit Pheromon der Menge $\Delta\tau$ markiert:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} 1/L^{k}(t) & if\ (i,j) \in T^{k}(t) \\ 0 & if\ (i,j) \notin T^{k}(t) \end{cases} \qquad (4)$$

Dabei ist $T^{k}(t)$ die Tour ,die von Ameise k innerhalb der Iteration t ausgeführt wurde. Ihre Gesamtlänge entspricht L^{k}.
Die neue Pheromonmenge des Weges ij nach Iteration t entspricht der Pheromonmenge τ_{ij} vor der Pheromonausschüttung, vermindert durch einen Verdampfungskoeffizienten ρ, zuzüglich der neuen Pheromonausschüttung $\Delta\tau_{ij}$ *aller* Ameisen, die diesen Weg benutzt haben:

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \qquad (5)$$

Untersuchungen haben ergeben, dass der Algorithmus optimal arbeitet, wenn den Variablen α, β, ρ die Werte 1 bzw. 5 bzw. 0,5 zugeordnet und die Anfangspheromonkonzentration $\tau_{ij}(0)$ einem kleinen positiven Betrag entspricht.

2.2.2 Evaluation

AS wurde an kleinen Varianten des Traveling Salesman Problems mit einer Größe von 30 bis 75 verbundenen Städten getestet. Dabei zeigte sich, dass AS für Probleme kleineren Umfangs gute Werte liefert: AS stellte sich als der beste Natur-inspirierte Algorithmus heraus und war anderen Algorithmen, die vergleichend untersucht wurden, mindestens ebenbürtig. Mit steigender Zahl der Städte konnte AS jedoch nie die beste bekannte Lösung erreichen, obwohl AS schnell gute Lösungen fand.

2.3 Ant Colony System (ACS)

Ant Colony System (ACS) ist eine Weiterentwicklung auf der Basis von AS. Ziel war es, AS so zu verbessern, dass der neue Algorithmus auch bei größeren Traveling Salesman Problemen ein ähnlich gutes Leistungsverhalten zeigt, wie es für kleinere Probleme bei AS bekannt ist. Hierzu sind einige Neuerungen eingeführt worden, die im folgenden beschrieben werden sollen.

2.3.1 Unterschiede zu AS

Die erste Veränderung betrifft die Entscheidungsfindung der einzelnen Ameise. Wurde bei AS eine probabilistische Entscheidung getroffen, die vom relativen Gewicht der einzelnen Wege abhing, so bedient sich ACS einer Regel, die, je nach Wert einer Zufallsvariablen *q*, deterministisch oder probabilitisch zu einer Entscheidung führt.

Ist die aus dem Intervall [0,1] mit homogen verteilter Wahrscheinlichkeit gezogene Zufallsvariable q kleiner als der Parameter $q_0 (mit: 0 \le q_0 \le 1)$, so ist der Weg s_{ij}, den eine Ameise wählt, derjenige mit dem höchsten relativen Gewicht. In diesem Fall wird der neue Weg, als Weg des maximalen Arguments, deterministisch bestimmt. Ist aber $q > q_0$, so wird der neue Weg ermittelt, indem zufällig aus allen Verbindung zu Nachbarknoten eine gewählt wird, wobei jedem Weg eine relative Wahrscheinlichkeit $p_{ij}^k(t)$ gemäß Formel (7) zugeordnet ist. Das Ergebnis wird der Variablen S zugewiesen. Diese probabilistische Form der Entscheidungsfindung entspricht der von AS.

$$s_{ij} = \begin{cases} \arg \max_{j \in N_i^k} \{\tau_{ij}(t) \cdot (\eta_{ij})^{\beta}\} & if \quad q \le q_0 \\ S & anderenfalls \end{cases} \tag{6}$$

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]\ [\eta_{ij}]^{\beta}}{\sum_{l \in N_i} [\tau_{il}(t)]\ [\eta_{il}]^{\beta}} \tag{7}$$

Die hier eingeführte Entscheidungsgrundlage soll im folgenden als *pseudo-zufällige Entscheidungsregel* bezeichnet werden. Sie zeichnet sich dadurch aus, dass ihre deterministische Komponente die Verbindung eines jeden Knotens verstärkt und etabliert, die durch die bisherige Erfahrung der Ameisen als die beste gilt. Die Konvergenz auf eine Lösung wird also gefördert. Die probabilistische Komponente hingegen ‚ermutigt' dazu, neue Wege auszukundschaften, und so möglicherweise bessere Verbindungen zu erforschen. Über q_0 ist es möglich, das Gewicht der deterministischen oder probabilistischen Strategie einzustellen und den Gesamtprozess so zu optimieren.

Eine zweite Veränderung gegenüber AS findet man bei der Pheromonausschüttung. ACS zeichnet sich dadurch aus, dass zum einen am Ende jeder Iteration eine übergeordnete Instanz die Ameise mit der besten Lösung herausfindet und deren Wege markiert. Diese Markierung- die übrigens als globale Herangehensweise nicht den Kommunikationsmerkmalen von *stigmergy* genügt- wird verzögert ausgeführt, also erst sobald alle Ameisen einer Generation ihr Ziel erreicht haben. Die ausgeschüttete Pheromonmenge ist reziprok der Länge des Gesamtweges $s_{\min}$, also nach der Güte der Lösung gewichtet.

Im Gegensatz hierzu erfolgt eine zweite, lokale Markierung, die Schritt für Schritt durchgeführt wird: Sobald eine Ameise eine Verbindung zweier Knoten wählt, wird diese mit einer konstanten Menge Pheromon τ_0 markiert. Dadurch soll vermieden werden, dass ‚ausgetretene' Pfade entstehen, die schließlich von der Mehrzahl der Ameisen benutzt werden, was *Stagnation* gleich käme.

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t); \qquad mit:\ \Delta\tau_{ij}(t) = 1/s_{\min} \tag{8}$$

$$\tau_{ij}(t, n_t) \leftarrow (1-\varphi)\tau_{ij}(t, n_t) + \varphi\tau_0 \tag{9}$$

n_t *bezeichnet die Anzahl der in dieser Generation ausgeführten Schritte;*

φ (*mit:* $0<\varphi<1$) *bezeichnet die Verdampfung nach jedem Schritt.*

Als dritte Änderung wurde bei ASC eine lokale statische Liste (*Kandidaten-Liste*) eingeführt, die zu einem gegebenen Knoten die Nachbarn in der Reihenfolge größerer Entfernung speichert. Jede Ameise entscheidet sich nun zunächst für einen Knoten der *Kandidaten-Liste*, bis diese leer ist. Erst dann werden die weniger attraktiven Knoten besucht. Es zeigt sich, dass eine kleine Kandidatenliste die Leistung des Algorithmus' steigert.

2.3.2 Evaluation

ACS wurde zur Lösung gut untersuchter symmetrischer, wie asymmetrischer Traveling Salesman Probleme herangezogen, um seine Leistungsfähigkeit mit anderen Lösungsansätzen zu vergleichen. Dabei wurden u.a. Probleme mit der Größe zwischen 75 und 1577 Städten betrachtet, für die AS nur unbefriedigende Ergebnisse lieferte. Unter allen Konkurrenten und in allen Fällen lieferte ACS hinsichtlich Qualität der Lösung und CPU-Zeit die besten Ergebnisse. Eine Übersicht zeigen Tab.1 und 2 in der Anlage.

2.4 Ameisenalgorithmen dynamischer verteilter Systeme

Im Zentrum der bisherigen Ausführungen stand das Traveling Salesman Problem als anschauliche Anwendung von ACO-Algorithmen. Neben anderen statischen Standardproblemen, wie beispielsweise dem Graph-Färbe-Problem und dem Problem der kürzesten gemeinsamen Supersequenz, gehören dynamische Systeme zu den Anwendungsgebbieten der Ameisenalgorithmen. Lösungsansätze sollen in diesem Abschnitt skizziert werden.

Führt man sich noch einmal die Prinzipien der Kommunikation sozialer Insekten vor Augen, so wird ersichtlich, dass Netzwerke ein ideales Einsatzgebiet von Ameisenalgorithmen darstellen könnten. Bei Netzwerken handelt es sich im allgemeinen um ein verteiltes System ohne zentrale Instanz, verbunden über Leitungen unterschiedlicher Kapazität und Geschwindigkeit. Datentransfer verursacht Kosten hinsichtlich Zeit und Kapazität - deshalb bietet es sich an, Daten lokal zu speichern.

Die Anwendung von Ameisenalgorithmen betrifft insbesondere das Problem der Optimierung des Datenverkehrs: Lokale Tabellen (*routing tables*), ähnlich der Entscheidungstabelle von AS, sollen derart gestaltet werden, dass bestimmte Leistungsmerkmale des Netzwerks (wie z.B. die Geschwindigkeit des Datentransfers) optimiert werden. Ein Netzwerk entspricht einem Graph G=(N,A), dessen Knoten N die Server darstellen, welche über Datenleitungen unterschiedlicher Geschwindigkeit, repräsentiert durch gewichtete Kanten A, verbunden sind. In Knoten i eintreffende Daten sollen nun schnellstmöglich ihr Ziel (Knoten Z) erreichen. Hierfür wird eine lokale Datenstruktur benötigt, die das ankommende Datenpaket bestmöglich an einen der Nachbarknoten weiterleitet. Um dieses Problem zu lösen, bietet es sich an, künstliche Ameisen über die Verbindungen des Netzwerkes laufen zu lassen, welche die benutzten Verbindungen mit Pheromonen markieren. Die Pheromonkonzentrationen der einzelnen ‚links' eines Knotens werden in der *lokalen Tabelle* gespeichert.

In die Entwicklung eines solchen Ameisenalgorithmus', müssen die Besonderheiten dynamischer Systeme einfließen. So verändert sich das Netz fortwährend - etwa durch variable Auslastung. Oft entspricht ein Netzwerk einem asymmetrischen Graph, da Geschwindigkeiten des Daten *up-* und *downloads* variieren. Weiterhin finden übergeordnete Funktionen (wie die Belohnung der besten Ameise bei ACS) in verteilten Systemen keine Anwendung, da es ineffizient ist, Daten an einem zentralen Punkt zusammen zu tragen. Ebenso erscheint es vorteilhaft, Markierungen schrittweise durchzuführen, da eine verzögerte Pheromonausschüttung den Nachteil mit sich bringt, dass die transportierte Datenmenge dadurch ansteigt, dass stets ein Gedächtnis der aktuellen Route übermittelt werden muss und zusätzlich retrograder Datenverkehr beim nachträglichen Markieren entsteht. Werden die genannten Besonderheiten jedoch beachtet, können Netzwerke durch Ameisenalgorithmen sehr effizient verwaltet werden.

Anhang: Ergebnisse

Problem	*ACS*	*GA*	*EP*	*SA*	*AG*	*Optimum*
Oliver30	**420**	421	**420**	424	**420**	420
Eil50	**425**	428	426	443	436	425
Eil75	**535**	545	542	580	561	535
KroA100	**21282**	21761	-	-	-	21282

Tab.1. Vergleich von ACS mit *genetic algorithm* (GA), *evolutionary programming* (EP), *simulated annealing* (SA) und *annealing-genetic algorithm* (AG) hinsichtlich des Optimierungsverhaltens verschiedener Standardprobleme. Angegeben sind die Integer-Werte der besten Tour-Längen, die von den einzelnen Algorithmen gefunden wurden. Hervorgehoben ist der jeweils beste Lösungsansatz.

Problem	ACS bestes Ergebnis	Optimum	% Fehler	CPU sec
d198	15888	15780	0,68	0.02
pcb442	51268	50779	0,96	0.05
att532	28147	27686	1,67	0.07
rat778	9015	8806	2,37	0.13
fl1577	22977	[22137-22249]	3.27-3.79	0.48

Tab.2. ACS-Leistungsverhalten für einige größere TS-Probleme. Die erste Spalte zeigt das beste Ergebnis aus 15 Versuchen; die zweite Spalte das optimale Ergebnis oder dessen Näherung (fl1577); Spalte drei ist der prozentuale Fehler des ACS im Vergleich mit dem optimalen Ergebnis zu entnehmen; Spalte vier gibt eine Übersicht der benötigten CPU-Zeit.

Literatur:

Marco Dorigo et al.(1997): Ant Colonies for the Traveling Salesman Problem, BioSystems 43, 73-81;

Marco Dorigo et al. (1999): Ant Algorithms for Discrete Optimization, Artificial Life, 5, 137-172.