# A M E I S E N S Y S T E M E

Arno Klein

Vortrag im Seminar „Moderne Heuristiken“, betreut durch ao. Univ.-Prof. Mag. Dr. Manfred Gronalt,
Lehrstuhl für Wirtschaftsinformatik der Technischen Universität Clausthal,
vorgetragen am 5. Juli 2001.

## E I N F Ü H R U N G

Ameisensysteme sind heuristische Suchverfahren, deren Methodik sich an dem Verhalten von Ameisen bei der Futtersuche orientiert.

1991 entwickelten Mailänder Wissenschaftler unter Führung von Marco Dorigo Algorithmen, die auf dem Verhalten der Ameisen basieren.

Die Besonderheit dieses Verfahrens ist vor allem die Dynamik des Optimierungsprozesses. So wird der Prozess auf Grund der gewonnenen Erfahrungswerte ständig angepasst. Er profitiert von gewonnenen Erkenntnissen.

Die von Dorigo und anderen entwickelten Algorithmen wurden in den letzten Jahren auch von anderen Wissenschaftlern weiterentwickelt, ergänzt und optimiert.

## D I E A M E I S E N

Obwohl sie nicht sehen können, finden Ameisen ohne größere Probleme kürzeste Wege zwischen ihrem Bau und Futterquellen. Sie hinterlassen Duftstoffe, sogenannte Pheromone, und kennzeichnen auf diese Weise die zurückgelegten Wege. Andere Ameisen orientieren sich bei der Futtersuche an den Pheromonspuren, die ihre Artgenossen gelegt haben und hinterlassen selbst neue Duftstoffe. Während eine einzelne Ameise mehr oder weniger zufällig ihren Weg wählt, erkennt ein Tier, das auf eine Pheromonspur trifft, diese und folgt ihr mit hoher Wahrscheinlichkeit, wobei stärkere Spuren bevorzugt werden. Dieser autokatalytische, d. h. sich selbst verstärkende, Prozess führt rasch dazu, dass alle Ameisen

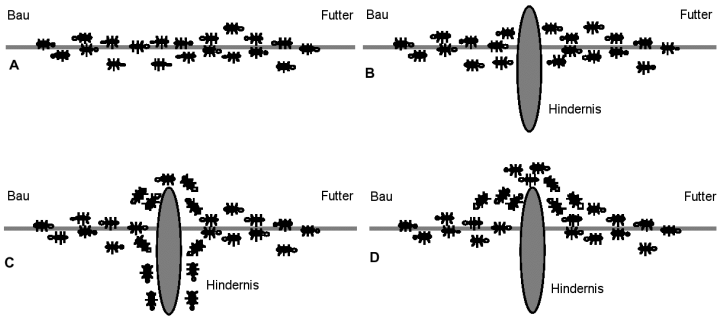


**Abb. 1.** (A) Ameisen bewegen sich zwischen ihrem Bau und einer Futterquelle. (B) Ein Hindernis taucht auf, mit gleicher Wahrscheinlichkeit kann jede einzelne Ameise den linken oder rechten Weg wählen. (C) Auf dem kürzeren Weg hinterlassen die Ameisen mehr Duftstoffe. (D) Alle Ameisen wählen den kürzeren Weg. [2]

auf dem kürzesten Weg laufen, denn die Ameisen, die auf Grund der stärkeren Pheromonspur den kürzeren Weg wählen, legen selbst natürlich auch Pheromone auf diesen Weg. [1]

Dies lässt sich sehr gut an der Abb. 1 verdeutlichen: Auf der Suche nach Futter oder auf der Rückkehr von der Futterquelle bewegen sich die Ameisen auf einem bereits

durch Pheromone gekennzeichneten Weg. (A) Nun taucht plötzlich ein Hindernis auf (B) und die Ameisen stehen vor der Wahl, ob sie links oder rechts gehen. In diesem Moment – in dem noch keine Pheromonspuren den kürzesten Weg um das Hindernis kennzeichnen – entscheiden sich die Ameisen zufällig für eine Richtung. Die Hälfte wählt den einen, die andere Hälfte den anderen Weg. Die Ameisen, die den kürzeren Weg wählen, sind schneller und hinterlassen deshalb im Laufe der Zeit auch mehr Duftstoffe, was dazu führt, dass immer mehr Ameisen den kürzeren Weg wählen (C). Dadurch steigt die Intensität der Pheromonspur auf diesem Weg noch schneller als ohnehin. Deshalb bewegen sich alle Ameisen in kürzester Zeit auf dem optimalen Pfad. (D)

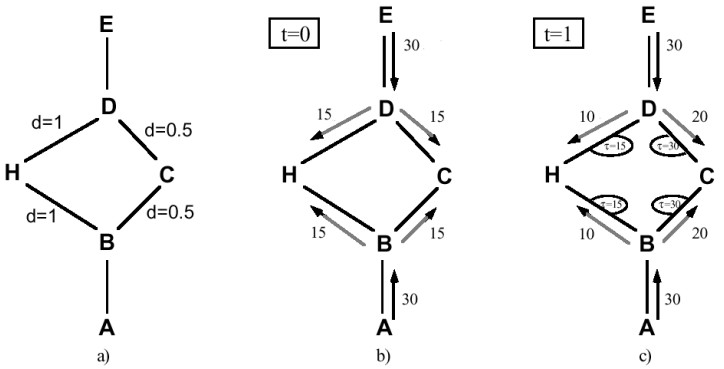


**Abb. 2.** Ein Beispiel mit virtuellen Ameisen. a) Der Startgraph mit Entfernungen. b) Zum Zeitpunkt t=0 sind keine Spuren auf den Kanten, deshalb gehen die Ameisen mit gleicher Wahrscheinlichkeit links oder rechts. c) Zum Zeitpunkt t=1 ist die Spur auf den kurzen Kanten stärker, weshalb die Ameisen diese bevorzugen. [1]

## GRUNDLEGENDES

Bei der Entwicklung der Ameisenalgorithmen nahm Dorigo einige Veränderungen zu realen Ameisen vor, denn sein Ziel war nicht die originalgetreue Nachbildung von Ameisenkolonien, sondern die Schaffung eines Optimierungswerkzeuges, das von den Erfahrungen aus der Natur profitiert.

Hervorzuheben ist, dass seine „künstlichen“ Ameisen einen Speicher haben, in dem vermerkt wird, welche Punkte sie bereits passierten. Auch sind sie nicht blind, so dass sie die Entfernung zu noch nicht besuchten Punkten berücksichtigen können. [1]

In Abb. 2 wird das Prinzip der Ameisenalgorithmen deutlich. Die Entfernungen DH und HB seien 1, DC und CB 0,5. Nehmen wir nun an, dass in jeder Zeiteinheit 30 Ameisen von A nach B kommen und 30 von E nach D, dass jede Ameise sich mit einer Geschwindigkeit von einer Längeneinheit pro Zeiteinheit bewegt und dass eine Ameise in der Zeit t eine Spur der Intensität 1 anlegt, welche in der Mitte des nächsten Zeitintervalls (t+1, t+2) vollständig verdunstet. Damit berücksichtigen die Ameisen ausschließlich die Spur der unmittelbar vorangegangenen Tiere.
Zum Zeitpunkt t = 0 ist noch keine Spur vorhanden, aber je 30 Ameisen sind an den Punkten B und D. Sie entscheiden vollkommen zufällig, ob sie über C oder H gehen, so dass durchschnittlich 15 Ameisen von jedem Punkt über C gehen und ebenso viele über H. (Abb. 2b)

Zum Zeitpunkt t = 1 finden die dreißig neuen Ameisen, die von A nach B und von E nach D kommen auf der Strecke BHD eine Spur der Intensität 15 vor, da die Ameisen, die diese Strecke gewählt haben, jetzt erst bei H angekommen sind. Hingegen haben die ersten Ameisen auf der Strecke BCD eine Spur der Stärke 30 angelegt, da sie bereits den kompletten Weg um das „Hindernis“ zurückgelegt haben.
Deshalb ist die Wahrscheinlichkeit, dass die Ameisen den Weg über C wählen, doppelt so groß, wie dass sie über H gehen. Es werden also durchschnittlich 20 Ameisen über C gehen und 10 über H. Dadurch verstärkt sich die Spur auf der Strecke BCD weiter. Dieser Prozess setzt sich fort, bis alle Ameisen die kürzeste Strecke wählen.

Die zugrundeliegende Idee ist, dass die Ameisen, wenn sie an einen Punkt kommen,

an dem sie sich zwischen zwei Wegen entscheiden müssen, stets den bevorzugen, den vorhergehende Ameisen häufiger gewählt haben. Das ist der, auf dem sich die stärkere Spur befindet. Erkennbar sind die Spuren umso stärker, je kürzer der Weg ist. [1]

## DAS AMEISENSYSTEM *

Am Beispiel eines euklidischen Traveling Salesman Problem (TSP) sollen nun die Algorithmen, die Dorigo 1991 einführte, erläutert werden. Diese wurden später ergänzt und weiterentwickelt, worauf jedoch erst in einem weiterführenden Abschnitt eingegangen werden soll.
Auch wenn die Ameisenalgorithmen hier aus Gründen der Übersichtlichkeit und Vergleichbarkeit ausschließlich am TSP erläutert werden, so wird doch deutlich, dass sie sich auch problemlos auf viele andere Optimierungsprobleme übertragen lassen.

Das TSP lässt sich als das Problem eines Handelsreisenden beschreiben, mit minimalem Aufwand eine bestimmte Anzahl an Städten jeweils einmal zu besuchen will, um dann an den Ausgangspunkt zurückzukehren.
Ein euklidisches TSP ist ein symmetrisches TSP[1], in dem die Entfernungen zwischen zwei Städten der euklidischen Distanz zwischen den Punkten im Graphen entsprechen. („Luftlinie“). Selbstverständlich lassen sich Ameisenalgorithmen auch auf jedes andere TSP, insb. auf asymmetrische TSP anwenden, jedoch soll hier der Einfachheit halber ein euklidisches TSP verwendet werden.

Wir haben ein TSP mit n Städten. Die Entfernung zwischen zwei Städten i und j sei $d_{ij}$, die euklidische Distanz der beiden Punkte des Graphen (N,E), wobei N die Menge der Städte und E die Menge der Kanten zwischen ihnen sei.

Wir definieren $b_i(t)$ (i = 1..n) als die Anzahl der Ameisen in der Stadt i zum Zeitpunkt t und die Anzahl aller Ameisen m folglich als

$$m = \sum_{i=1}^{n} b_i(t)$$

Jede Ameise habe folgende Eigenschaften:

- Sie wählt die nächste Stadt mit einer Wahrscheinlichkeit aus, die durch eine Funktion aus der Entfernung zu dieser Stadt und der Intensität der Spur auf der Kante zwischen den beiden Städten ausgedrückt wird.
- Um die Ameisen nur erlaubte Touren konstruieren zu lassen, sind Reisen zu bereits besuchten Städten verboten, was durch einen Speicher kontrolliert wird.
- Wenn eine Ameise eine Tour beendet, legt sie eine Spur auf jede Kante, die sie benutzt hat.

Wir definieren $\tau_{ij}(t)$ als die Intensität der Spur auf der Kante (i,j) zum Zeitpunkt t. Jede Ameise wählt im Zeitpunkt t eine neue Stadt, die sie zum Zeitpunkt t+1 erreicht. Wir nennen die n-malige Wiederholung des Algorithmus einen Zyklus. Nach jedem Zyklus hat jede Ameise eine Tour konstruiert. Dann wird die Intensität der Spur auf jeder Kante neu berechnet:

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij} \qquad (1)$$

mit ρ als einem Koeffizienten, so dass (1- ρ) die Verdunstung der Spur im Zeitintervall (t,t+n) ausdrückt und

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \qquad (2)$$

wobei $\Delta\tau_{ij}^{k}$ die Stärke der Spur pro Längeneinheit sei, die die k-te Ameise im Zeitinter-

* Im Wesentlichen sind die Definitionen der Ameisenalgorithmen dem Aufsatz „The Ant System: Optimization by a colony of cooperating agents“ von Dorigo et al. [1] entnommen, in dem das Ameisensystem erstmals erwähnt wurde.

[1] Bei symmetrischen TSP ist – im Ggs. zu asymmetrischen – die Entfernung zwischen zwei Städten in beide Richtungen gleich.

vall (t,t+n) auf der Kante (i,j) gelegt hat und wie folgt berechnet werde:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} & \text{wenn die k-te Ameise die Kante (i,j) in ihrer Tour benutzt (im Intervall (t,t+n))} \\ 0 & \text{im anderen Falle} \end{cases} \qquad (3)$$

mit Q als einer Konstanten und $L_k$ als der Tourlänge der k-ten Ameise.

Um die Voraussetzung zu erfüllen, dass eine Ameise alle n Städte jeweils einmal besucht, versehen wir jede Ameise mit einem Speicher, in dem alle bereits besuchten Städte abgelegt werden und der kontrolliert, dass diese Städte nicht noch einmal besucht werden. Außerdem dient der Speicher am Ende eines jeden Zyklus zur Ermittlung der besuchten Städte und der Länge einer Tour. Anschließend wird der Speicher wieder gelöscht und die Ameisen können eine neue Tour beginnen. Wir definieren $M_k$ als den Vektor, der den Speicher der k-ten Ameise enthält und folglich $M_k(s)$ als die s-te Stadt, die die k-te Ameise in ihrer aktuellen Tour besucht hat.

Um die Entfernung zu den anderen Städten bei der Wahl des nächsten Ziels zu berücksichtigen, definieren wir die Nähe $\eta_{ij}$ zwischen zwei Städten i und j als den Kehrwert ihrer Entfernung $1/d_{ij}$.

Mit diesen Definitionen lassen sich nun die Wahrscheinlichkeiten bestimmen, mit denen eine Ameise eine Verbindung zwischen zwei Städten wählt.

Dies geschieht mit der folgenden Formel, wenn $j \notin M_k$, ansonsten ist die Wahrscheinlichkeit 0:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \notin M_k} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} \qquad (4)$$

wobei $\alpha$ und $\beta$ Variablen sind, die die relative Wichtigkeit von Spurintensität (je mehr Ameisen einen Weg benutzt haben, desto besser) und Nähe (je näher eine Stadt, desto besser) bestimmen. Auf diese Weise werden Greedy-Search-Methoden mit dynamischen Optimierungsmethoden kombiniert.

## DIE ALGORITHMEN *

Auf der Grundlage der Definitionen des vorigen Kapitels wollen wir nun den sogenannten Ameisen-Zyklus-Algorithmus (ant cycle algorithm) erläutern. Dorigo führte 1991 diesen und die beiden dem Zyklus-Algorithmus ähnlichen Dichte- (ant density algorithm) und Mengen-Algorithmen (ant quantity algorithm) ein.

Beim Zyklus-Algorithmus wird zum Zeitpunkt t = 0 eine Initialisierungsphase eingeleitet, in der die Ameisen in verschiedenen Städten ausgesetzt werden, Startwerte für die Spurintensitäten bestimmt werden und die Startpunkte als erstes Element in die jeweiligen Speicher der Ameisen eingefügt werden.
Nun bewegen sich die Ameisen von einer Stadt zur nächsten, wobei die Zielstadt j mit einer Wahrscheinlichkeit bestimmt wird, die eine Funktion aus Nähe und Spurintensität ist (Formel 4). Der Nähe-Faktor beeinflusst die Wahrscheinlichkeit dahingehend, dass Städte, die nah an der aktuellen Stadt liegen bevorzugt werden, der Spur-Faktor führt dazu, dass stark benutzte Wege häufiger gewählt werden. Offensichtlich würde das Setzen von $\alpha = 0$ dazu führen, dass die Spuren nicht mehr berücksichtigt würden – wir hätten einen Greedy-Search-Algorithmus mit mehreren Startpunkten.

Nach n Durchläufen, bzw. einem Zyklus beenden alle Ameisen ihre Tour, da sie alle n Städte besucht haben. Nun werden aus den Speichern der Ameisen die Tourlängen $L_k$ berechnet und die Spurstärken $\Delta\tau_{ij}^{k}$ nach Formel 3 neu berechnet. Auch werden die

* Im Wesentlichen sind die Beschreibungen der Algorithmen [1] entnommen.

jeweils kürzesten bisher gefundenen Wege der einzelnen Ameisen (bspw. min $L_k$, k = 1..m) gespeichert und alle Speicher geleert. Der Prozess bricht ab, wenn die benutzerdefinierte maximale Zyklenzahl $NC_{MAX}$ erreicht ist oder sich alle Ameisen auf der selben Tour bewegen. Dieser zweite Fall wird Stagnation genannt, da der Algorithmus aufhört, nach anderen möglichen Lösungen zu suchen.

Der formale Algorithmus von Dorigo ist in der deutschen Übersetzung im Anhang zu finden.

Die beiden anderen erwähnten Algorithmen unterscheiden sich nur unwesentlich vom Zyklus-Algorithmus, lediglich die Intensität der Spuren wird anders bestimmt. Im Dichte- und auch im Mengen-Algorithmus wird nicht das Ende eines Zyklus abgewartet, bis die Ameise eine Spur hinterlässt, sondern nach jedem Schritt wird eine Spur gelegt.
Beim Dichte-Algorithmus legt jede Ameise eine Spur der Stärke Q auf eine benutzte Kante, beim Mengen-Algorithmus eine Spur der Stärke $Q/d_{ij}$.
Der Dichte-Algorithmus berücksichtigt also ausschließlich die Anzahl der Ameisen, die einen Weg benutzt haben, wohingegen der Mengen-Algorithmus kürzere Kanten den längeren vorzieht.

Experimente haben jedoch gezeigt, dass der zuerst beschriebene Ameisen-Zyklus-Algorithmus den anderen beiden überlegen ist, in vielen Fällen sogar recht deutlich.

## DAS KOLONIESYSTEM

Das Ameisen-Kolonie-System (ant colony system, ACS) ist den Ameisenalgorithmen sehr ähnlich, unterscheidet sich jedoch in zwei Punkten von ihnen.

Die Ameise im ACS benutzt mit einer konstanten Wahrscheinlichkeit $q_0$ den besten Pfad, der von ihrer aktuellen Stadt ausgeht und wählt mit einer Wahrscheinlichkeit $1-q_0$ den Pfad nach der Formel 1 aus, die bei den Ameisenalgorithmen benutzt wird. So werden die stärksten Spuren mehr beachtet als bei den Ameisenalgorithmen, was dazu führt, dass weniger alternative Möglichkeiten untersucht werden.

Der zweite Unterschied ist die Art und Weise, in der die Spurintensitäten berechnet werden.
Beim ACS wird zwischen globaler und lokaler Spurberechnung unterschieden. Einerseits wird nach Beendigung eines kompletten Zyklus nur auf dem global besten Pfad die Spur verstärkt, andererseits „verdunstet“ nach jedem Schritt einer Ameise die Spur auf dem benutzten Pfad, um nachfolgende Ameisen zu veranlassen, noch nicht benutzte Pfade zu testen. [4]

## DAS MAX-MIN-SYSTEM

Das Max-Min-Ameisensystem (MMAS) ist eine leichte Abänderung der Ameisenalgorithmen. Wie im ACS darf nur die global beste Ameise ihre Spur verstärken.

Den Namen gab dem System jedoch ein anderer wichtiger Unterschied zu Dorigos Algorithmen. Beim MMAS wird eine obere und untere Schranke für die Spurintensitäten, $\tau_{max}$ und $\tau_{min}$ festgelegt, um ein Stagnationsverhalten der Ameisen zu verhindern. Auf diese Weise kann es nicht vorkommen, dass eine Spur so stark wird, dass Ameisen nur noch diese wählen, oder so schwach wird, dass Ameisen diese nicht mehr benutzen.

Beim MMAS werden alle Spuren am Anfang auf den Maximalwert $\tau_{max}$ gesetzt, um die Unterschiede zwischen den Pfaden am Anfang nicht zu stark werden zu lassen, damit die Ameisen die Pfade zu Beginn recht gleichmäßig untersuchen. [4]

## EXPERIMENTE

Experimente haben gezeigt, dass Ameisensysteme, auch das ACS oder MMAS, gute Verfahren sind, um optimale Lösungen für das TSP zu konstruieren. Kombinationen aus dem MMAS und dem 3-opt-Verfahren[1] liefern beste Ergebnisse. (Abb. 3)

| Problem | MMAS | | MMAS + 3-opt | | beste bekannte Lösung |
|---|---|---|---|---|---|
| | beste | Ø | beste | Ø | |
| d198 | 15780 | 15780,4 | 15780 | 15780,2 | 15780 |
| lin318 | 42029 | 42029,0 | 42029 | 42064,6 | 42029 |
| pcb442 | 50778 | 50911,2 | 50778 | 50917,7 | 50778 |
| att532 | 27686 | 27707,9 | 27686 | 27709,7 | 27686 |
| rat783 | 8806 | 8814,4 | 8806 | 8828,4 | 8806 |
| u1060 | 224455 | 224853,5 | 224152 | 224408,4 | 224094 |
| pcb1173 | 56896 | 56956,0 | 56897 | 57029,5 | 56892 |
| d1291 | 50801 | 50821,6 | 50825 | 50875,7 | 50801 |
| fl1577 | 22286 | 22311,0 | 22311 | 22394,6 | 22249 |

**Abb. 3.** Vergleich von MMAS, MMAS mit 3-opt und der besten bekannten Lösung für symmetrische TSP aus der TSP-Bibliothek der Universität Heidelberg, 25 Durchläufe für weniger als 1000 Städte, sonst 10 Durchläufe. Die Zahlen in den Problembezeichnungen geben die Dimensionen der Probleme an (d198 = 198 Städte) [5]

Bei der Wahl der Zahl der Ameisen und ihrer Startstädte hat man in Experimenten festgestellt, dass die Ameisen dann gute und schnelle Lösungen liefern, wenn in jeder Stadt genau eine Ameise ausgesetzt wird (Anzahl der Ameisen = Anzahl der Städte, Ameisen gleichmäßig verteilen). Diese Tatsache ist bei näherer Betrachtung offensichtlich. Das Aussetzen

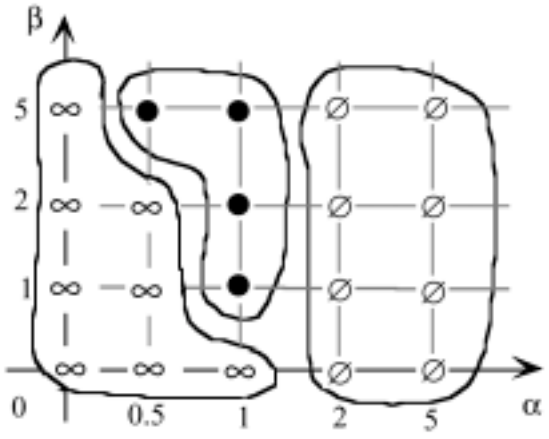


**Abb. 4.** Verhalten des Ameisen-Zyklus-Algorithmus bei unterschiedlicher Wahl der Parameter α, β. [1]

● Gute Lösungen, keine Stagnation
∞ Keine guten Lösungen, keine Stagnation
Ø Keine guten Lösungen, Stagnation

von einer Anzahl Ameisen, die ein Vielfaches der Anzahl der Städte ist, sollte offensichtlich keinen Effekt auf die Qualität der Lösung haben. Verwendet man eine andere Anzahl an Ameisen, vor allem weniger, als Städte vorhanden sind, hat das jedoch Einfluss auf die Qualität, da nicht alle Bereiche gleichmäßig untersucht werden.
Dasselbe gilt für den Fall, dass man die Ameisen nicht über die Städte verteilt, sondern alle im selben Bereich aussetzt.

Bei der Suche nach optimalen Werten für die Parameter α und β, d. h. die Gewichtung von Spurstärke und Nähe bei der Wahl der nächsten zu besuchenden Stadt, und ρ, bzw. der Verdunstung 1-ρ, hat man festgestellt, dass eine Gewichtung von $\alpha = 1$ und $\beta = 5$ und die Wahl von $\rho = 0{,}5$ (bzw. $\rho = 0{,}99$ beim Dichte- und Mengen-Algorithmus) die besten Ergebnisse liefert. (Abb. 4)

## SCHLUSSBEMERKUNGEN

Abschließend lässt sich sagen, dass die Ameisenalgorithmen und ihre Erweiterungen eine hervorragende Möglichkeit sind, um optimale Lösungen für das TSP zu konstruieren. Wenn man sie mit den bekannten verbessernden Heuristiken, wie dem 3-opt-Verfahren, kombiniert, erreicht man mit relativ geringem Rechenaufwand beste Lösungen.

[1] Beim 3-opt-Verfahren werden alle Routen untersucht, die in der 3-Nachbarschaft der konstruierten Route liegen, d. h. bei denen drei Kanten vertauscht sind. Analog wird auch häufig das 2-opt-Verfahren verwendet, bei dem zwei Kanten vertauscht werden.

## QUELLEN

[1] Dorigo, Maniezzo, Colorni, “The Ant System: Optimization by a colony of cooperating agents”, IEEE Transactions on Systems, Man and Cybernetics, Part B, Vol. 26, No. 1, 1996, S. 1-13

[2] Dorigo, Gambardella, “Ant colonies for the travelling salesman problem”, BioSystems, Vol. 43, 1997, S. 73-81

[3] Colorni, Dorigo, Maniezzo, “Distributed Optimization by Ant Colonies”, Proceedings of ECAL 91, European Conference on Artificial Life, Paris, Elsevier Publishing, 1991, S. 134-142

[4] Stützle, Dorigo, “ACO Algorithms for the Traveling Salesman Problem”, Evolutionary Algorithms in Engineering and Computer Science, Wiley & Sons, 1999

[5] Dresp, “Ameisenalgorithmen als Metastrategie in der Kombinatorischen Optimierung“, Seminararbeit, Halle, 2000.

## WEITERE INFORMATIONEN

Weitergehendes Material zu Ameisensystemen ist auf der Internetseite von Marco Dorigo unter http://iridia.ulb.ac.be/dorigo/ zu finden. Hier können auch zahlreiche Arbeiten zum Thema heruntergeladen werden.

## DIESE ARBEIT

Die vorliegende Seminararbeit und die beim Vortrag verwendeten Folien sind auch im Internet unter

**www.arno-klein.de/studium/ameisensysteme/**

zu finden.

A N H A N G

**Der Ameisen-Zyklus-Algorithmus (ant cycle algorithm)***

1. (Initialisierung)
   setze t: = 0 (t ist der Zeit-Zähler)
   setze NC := 0 (NC ist der Zyklen-Zähler)
   Für jede Kante (i, j) setze einen Startwert $\tau_{ij}(t)$ = c für die Spurintensität und $\Delta\tau_{ij}$ := 0.
   Platziere die m Ameisen auf den m Knoten.

2. setze s: = 1 (s ist der Speicher-Index)
   Für k: = 1 bis m tue
   Füge die Start-Stadt der k-ten Ameise in $M_k(s)$ ein

3. wiederhole, bis die Speicher-Liste voll ist (dieser Schritt wird (n - 1) mal wiederholt)
   setze s := s + 1
   für k := 1 bis m tue {
   wähle die Stadt j, wo die Ameise hinreist, mit der Wahrscheinlichkeit $p_{ij}^k(t)$
   bewege die Ameise zu dieser Stadt j
   füge die Stadt j in $M_k(s)$ ein }

4. für k := 1 bis m tue {
   bewege die k-te Ameise von $M_k(n)$ zu $M_k(1)$
   berechne die Länge der Tour, die durch die k-te Ameise beschrieben wird
   wenn eine neu gefundene Tour kürzer ist, als das bisherige Minimum, aktualisiere die kürzeste gefundene Tour }
   für k:=1 bis m tue {

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k} & \text{wenn (i, j)} \in \text{der Tour, die durch } M_k \text{ beschrieben wird} \\ 0 & \text{sonst} \end{cases}$$

   $\Delta\tau_{ij} := \Delta\tau_{ij} + \Delta\tau_{ij}^k$ }

5. für jede Kante (i, j) berechne $\tau_{ij}(t+n)$ gemäß der Gleichung $\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}$
   setze t := t + n
   setze NC := NC + 1
   für jede Kante (i, j) setze $\Delta\tau_{ij} := 0$

6. wenn (NC < $NC_{MAX}$) und (kein Stagnationsverhalten)
   dann
   leere alle Speicher
   Gehe zu Schritt 2
   sonst
   gib kürzeste Tour aus
   Stop

* Die deutsche Übersetzung des Algorithmus ist [5] entnommen.