F E R N U N I V E R S I T Ä T
in Hagen
FAKULTÄT FÜR WIRTSCHAFTSWISSENSCHAFT

Kombinatorische Optimierung mit Ameisensystemen

Seminar:	Problemlösen in graphischen Strukturen
	des Lehrstuhls für Betriebswirtschaftslehre, insbesondere Operations Research von Univ.Prof. Dr. Wilhelm Rödder
Studienjahr:	2008
Matrikelnummer:	693 439 0
Name, Vorname:	Schiller, Daniel
Anschrift:	Auf dem Loor 53, 51143 Köln
Abgabedatum:	8. November 2008

1. Einleitung

1.1. Zielsetzung der Arbeit

Untersuchungsgegenstand der vorliegenden Arbeit ist das Prinzip der *Optimierung mittels Ameisenkolonien (Ant Colony Optimization, ACO)*, also das Prinzip, welches das natürliche Wegfindungsverhalten von Ameisenpopulationen nachahmt, um (kombinatorische) Optimierungsprobleme zu lösen. In der Arbeit sollen der Ursprung des Prinzips für Optimierungsprobleme dargestellt und eine Einordnung zu anderen Lösungsstrategien vorgenommen werden. Anschließend werden Grundidee und Techniken der Algorithmen beschrieben, um dann verschiedene Ausprägungen darzustellen und zu analysieren. Betriebswirtschaftliche Anwendungsbeispiele werden vorgestellt. Abschließend werden die Ergebnisse zusammengefasst, bewertet und ein Ausblick in zukünftige Entwicklungen gegeben.

2. Analyse

2.1. Einordnung

Zur Lösung von *NP-schweren Problemen* (Zeitaufwand für Lösung nicht polynomiell von Problemgröße abhängig) werden häufig *Heuristiken* eingesetzt. Diese erzeugen Näherungslösungen, garantieren aber keine Optimalität, und nutzen spezielle Eigenschaften des Problems aus, um in akzeptabler Zeit eine gute Lösung zu erreichen. *Metaheuristiken* hingegen nutzen allgemeine und abstrakte algorithmische Regeln, mit deren Hilfe eine Heuristik für ein konkretes Problem definiert werden kann.

ACO zählt zu diesen Metaheuristiken, aus welcher verschiedene heuristische Algorithmen entwickelt werden konnten und stellt damit selbst ein Rahmenwerk dar. Algorithmen, welche in dieses Rahmenwerk passen werden *ACO-Algorithmen* genannt. Sie zählen zur Klasse der natürlichen Algorithmen, wie beispielsweise *Simulated Annealing* und *Genetischer Algorithmus*. Beobachtbare Vorgänge oder Prinzipien aus der Natur werden dabei nachgebildet und abgewandelt, um Lösungsstrategien zu entwickeln, neuartige Probleme zu lösen, neue Lösungen für bekannte Probleme zu finden oder effizientere Verfahren zu entwickeln. Naturorientierte Prinzipien lassen sich meist auf unterschiedlichste Probleme anwenden, sind also nicht strikt problemspezifisch[1].

Kombinatorische Probleme (Ganzzahligkeitsbedingung für die Lösung) zählen zu NP-schweren Problemen. Mittels der ACO-Metaheuristik konnten mehrere Algorithmen entwickelt werden, um mehrere Probleme dieser Klasse (v.a. graphenbasierte Minimierungsprobleme in Netzwerken) zu lösen.

In Netzwerken genügen die Entfernungen der Knoten d_j den so genannten *Bellmanschen Gleichungen*, so dass Teilwege kürzester Wege selbst wieder kürzeste Wege darstellen. Besitzt das Netzwerk keine Zyklen negativer Länge, besitzen die *Bellmanschen Gleichungen* genau eine optimale Lösung[2]. Da sich diese Gleichungen nur in Spezailfällen direkt lösen lassen, existieren eine Reihe von Algorithmen, welche teilweise speziell die Struktur des zugrunde liegenden Netzwerks ausnutzen, um eine

[1] (2)
[2] (3 S. 44 - 45)

optimale Lösung zu erreichen. Um zu erkennen, ob die Voraussetzungen für die einzelnen Verfahren erfüllt sind, müssen teilweise weitere Algorithmen zur Prüfung der Struktur vorgeschaltet werden.

2.2. Grundidee der Ameisensysteme

Ameisenalgorithmen zur Lösung kombinatorischer Optimierungsprobleme stellen eine so genannte heuristische Metastrategie dar. Sie beruhen auf dem Verhalten realer Ameisenkolonien und simulieren deren Prinzip der Markierung eines Weges durch Pheromonspuren der auf dem Weg befindlichen Individuen. Die gelegte Spur beeinflusst an Kreuzungen das Verhalten der folgenden Individuen bei der Auswahl des nächsten Wegs. Mittelfristig sammeln sich so auf kurzen Wegen stärkere Pheromonspuren an. Ameisensysteme versuchen dieses Prinzip nachzubilden, um kürzeste Wege in Netzen zu finden.

2.2.1. Reale Ameisen

Ameisenkolonien bauen für die Versorgung ein Transportnetz zwischen Futterstellen und Nest auf. Zwischen allen möglichen Wegen gibt es eine Vielzahl langer, mehrere kurze und einen oder wenige kürzeste Wege. Die Population ist bestrebt im Optimum für den Transport einer festen Menge einen kürzesten Gesamtweg zu finden.

Reale Ameisen besitzen als Individuen nur eingeschränkte Fähigkeiten. Trotz dieser Einschränkungen schafft ein Schwarm von Ameisen Leistungen, welche über das Leistungsvermögen der Individuen hinaus gehen. Grundlegende Technik bei der Wegfindung ist dabei die einfache Kommunikation der Ameisen untereinander und mit der Umgebung durch das Legen von Pheromonspuren auf ihren Wegen. Nachfolgende Ameisen werden in ihren Wegentscheidungen durch diese Spuren beeinflusst. Das Finden eines kürzesten Wegs zwischen Nest und Futterstelle folgt dabei den Schritten:

1. Anfangs sind alle Wege unmarkiert. Die Ameisen wählen alle Wege mit gleicher Wahrscheinlichkeit zur Erkundung und legen eine Pheromonspur.
2. Die Ameisen auf den kurzen Wegen erreichen zu erst das Futter und wieder das Nest.
3. Für folgende Ameisen sind die kurzen Wege stärker markiert als die nur schwach oder noch unmarkierten langen Wege. Die Ameisen entscheiden sich häufiger für die stark markierten Wege. Im Mittel wird der Großteil der Ameisen diesen Wegen folgen und die Spur auf diesen stabilisieren.

Auffällig bei Experimenten ist, dass trotzdem nie alle Ameisen die kurzen Wege nehmen. Mit einer gewissen Wahrscheinlichkeit nehmen einige Ameisen weiterhin die langen Wege, was als Suchverhalten verstanden werden kann[3], denn es ist nicht garantiert, dass die Pheromonspur bereits auf dem global kürzesten Weg konvergiert ist. Durch dieses Verhalten ist es möglich noch weitere Lösungen zu erkunden. Gleichzeitig ergaben diese Experimente aber, dass eine Ameisenkolonie nach längerer Zeit eine einmal gefundene Lösung (lokales Optimum) nur schwer wieder verlässt, wenn sich eine neue Lösung anbietet[4].

[3] (10 S. 1-6)
[4] (10 S. 5)

Vorteil dieser Schwarmintelligenz ist die „Einfachheit“ der Individuen, was eine Simulation des Verhaltens im Computer vereinfacht, bei gleichzeitig hoher Leistungsfähigkeit des Gesamtsystems.

2.2.2. Künstliche Ameisen

Um einen Algorithmus zum Finden kürzester Wege in Netzwerken aus dem Verhalten realer Ameisen abzuleiten, müssen einige Erweiterungen vorgenommen werden. Im Unterschied zum zeitlich kontinuierlichen Ablauf des Vorbilds ist eine algorithmische Implementierung mit diskreten Zeitschritten einfacher. Außerdem muss ausgeschlossen werden, dass die künstlichen Ameisen in Schleifen gefangen bleiben. Sie erhalten ein Gedächtnis für ihren bisherigen Pfad und dessen Länge[5], um zu erreichen, dass die individuellen Wegentscheidungen probalistisch erfolgen, Schleifen vermieden werden und bei der Pheromonaktualisierung rückwirkend die Qualität (Länge) der bisherigen Teillösung mit einfließen kann. Außerdem kann durch die Simulation von Verdunstung der Pheromone die Leistung des Algorithmus beim Erkunden des Lösungsraums bzw. beim Verlassen lokaler Optima verbessert werden, im Gegensatz zum empirisch beobachteten Verhalten realer Ameisen.

Die erste Umsetzung in einen Algorithmus als heuristische Lösungsstrategie für das NP-harte *Travelling-Salesman-Problem (TSP)* wurde 1991 von Dorigo et al und Colorni et al. entwickelt[6]. Als abkürzender Oberbegriff hat sich *ACO (Ant Colony Optimization)* etabliert. ACO-Algorithmen haben sich seither bei graphenbasierten kombinatorischen Optimierungsproblemen bewährt.

2.2.3. Darstellung des Grundprinzips: Ameisensystem (AS)

In der Grundversion hat ein ACO-Algorithmus folgende Eigenschaften:

- Jede Ameise entscheidet an einem Knoten individuell probalistisch über ihren nächsten Weg in Abhängigkeit von den Pheromonspuren.
- Auf dem gewählten Weg legt jede Ameise eine Pheromonspur.
- Jede Ameise erinnert sich an schon besuchte Orte, und sucht diese nicht erneut auf (um problemspezifische Nebenbedingungen einzuhalten).

Das originäre Grundprinzip eines ACO-Algorithmus, genannt *Ameisensystem (Ant System, AS)*, lässt sich durch folgende Definitionen beschreiben:

Definition 1 (Algorithmus Pseudocode)

- Initialisierung
- so lange Abbruchbedingung nicht erreich ist
  - für alle Ameisen
    - bestimme mögliche nächste Knoten
    - für alle möglichen Zielknoten
      - bestimme Wahrscheinlichkeit anhand der Pheromonspuren zu den Knoten
    - wähle durch eine Zufallszahl probalistisch einen Knoten
    - bewege Ameise
    - aktualisiere Pheromonspur auf den Kanten

[5] (10 S. 10,11)
[6] (1 S. 17), (6 S. 217)

Zur Ausformulierung der in **Definition 1** angegeben Schritte gilt:

Definition 2 **(Knotenbestimmung AS)**

Für jede Ameise k wird in jedem Iterationsschritt h am Knoten i die Wahrscheinlichkeit für die Auswahl der folgenden Knoten j bestimmt nach:

$$p_{ij,h}^{k} = \frac{\left[\tau_{ij,h}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{l \in N_i^k}\left(\left[\tau_{il,h}\right]^{\alpha}\left[\eta_{il}\right]^{\beta}\right)}$$

Dabei sind:

- $j \in N_i^k$ die Knoten j ,welche erlaubte Nachfolger des Knotens i für Ameise k sind,
- $\tau_{ij,h}$, der aktuelle Pheromonwert der Kante $\langle i,j\rangle$ bei Iteration h,
- η_{ij} eine heuristische (vorausschauende) Information zur Bewertung der Attraktivität den Knoten j als Nächsten anzusteuern,
- $\alpha, \beta \geq 0$ Steuerparameter für das Verhalten (s. Abs. 2.2.4)

Definition 3 **(Heuristische Information AS)**

Die heuristische Informationsgröße η_{ij} ist die Inverse der Kosten/Länge der betrachten Kante $\langle i,j\rangle$:

$$\eta_{ij} = \frac{1}{c_{ij}}$$

Definition 4 **(Pheromoninitialisierung AS)**

Zur Initialisierung der Pheromonspur wird oft die *„Nächster-Nachbar-Heuristik"(NN)* verwendet. Alle Kanten $\langle i,j\rangle$ des Netzes werden dabei wie folgt gleich markiert:

$$\tau_{ij,1} = \frac{m}{C^{nn}},$$

wobei m die Anzahl der Ameisen und C^{nn} die gesamten Kosten der *Nächster-Nachbar*-Lösung T^{nn} sind.

Definition 5 **(Pheromonaktualisierung AS)**

Wenn m Ameisen bis zu Iteration h die Lösungen $T_h^1, T_h^2, \ldots T_h^m$ mit den Gesamtkosten /-längen $C_h^1, C_h^2, \ldots C_h^m$ erzeugt haben, wird die Pheromonspur $\tau_{ij,h}$ auf jeder Kante $\langle i,j\rangle$ aktualisiert nach folgender Regel:

$$\tau_{ij,h+1} = \tau_{ij,h}(1-\rho) + \sum_{k=1}^{m} \Delta_{ij,h}^{k}$$

mit:

$$\Delta_{ij,h}^{k} = \begin{cases} 1/C_h^k & \langle i,j\rangle \in T_h^k \\ 0 & \text{sonst} \end{cases}$$

$$0 \leq \rho \leq 1$$

In **Definition 2** bestimmen die beiden Parameter α, β die Gewichtung der beiden Größen $\tau_{ij,h}$ und η_{ij} bei der Bestimmung der Wahrscheinlichkeiten. Durch $\beta = 0$ können die heuristischen Informationen η_{ij} über die nächsten Knoten komplett ausgeblendet werden. Bei der Berechnung der Wahrscheinlichkeiten werden nur erlaubte Knoten $j \in N_i^k$ für jede Ameise k berücksichtigt, so dass bspw. bereits besuchte Knoten nicht erneut aufgesucht werden (beim TSP). Zu große Werte von α (> 1) betonen anfängliche Zufallsschwankungen zu stark und führen zu einem schlechten Verhalten des Algorithmus[7].

Definition 3 gibt an, dass die „vorausschauende" heuristische Information η_{ij} bei der Knotenwahl die Länge c_{ij} der betrachteten Kante berücksichtigt. Durch diese Formulierung erhalten kürzere Kanten in **Definition 2** eine höhere Wahrscheinlichkeit. Es wird ein „gieriges" Verhalten belohnt, so dass Ameisen kurze Kanten bevorzugen, ähnlich dem Vorgehen bei der *Nächster-Nachbar*-Heuristik.

Definition 4 initialisiert die Pheromonspur auf allen Kanten des Netzes gleichstark, wobei die Anfangsgewichtung bereits Informationen über die Länge der Ausgangslösung aus der *Nächster-Nachbar*-Heuristik enthält. Ziel ist eine Anfangsspur zu legen, deren Stärke etwas höher ist als die zu erwartende neu gelegte Spur nach jeder Iteration. Ist die Anfangsspur zu schwach, können die ersten iterierten und noch zufälligen Lösungen die Tendenz der folgenden Lösungen stärker beeinflussen und dabei zu Konvergenz in suboptimalen Bereichen des Lösungsraums führen. Ist die anfängliche Spur zu stark, haben die Spuren der ersten Iterationen fast keinen Effekt bis die Anfangspur weit genug verdunstet ist. Damit wird wertvolle Rechenzeit vergeudet.

In **Definition 5** beschreibt der erste Term die zeitliche Entwicklung der vorhandenen Pheromonspur $\tau_{ij,h}$ auf Kante $\langle i, j \rangle$. Der Parameter ρ ist die Verdunstungsrate in jedem Iterationsschritt. Der zweite Term berechnet für jede Kante $\langle i, j \rangle$ die Aktualisierung aus den Ameisen, welche bei Iteration h diese Kante benutzt haben. Dabei legen die Ameisen auf der Kante eine Spur, deren Stärke umgekehrt proportional zur bereits zurückgelegten Länge ihrer individuellen Tour ist. Ameisen, welche bereits eine lange Strecke zurückgelegt haben, legen also eine schwächere Spur als Ameisen, welche bisher kurze (Teil-)Wege gefunden haben. Allgemein erhalten Kanten, welche von vielen Ameisen genutzt werden und Teile kurzer Touren sind eine stärkere Markierung. Durch die Einbeziehung der Weglänge in die Aktualisierung kann schnellere Konvergenz erreicht werden. Außerdem gewinnt die Verdunstung bei komplexen Problemen höhere Bedeutung für die Lösungsqualität[8].

2.2.4. Einfluss der Parameter

Durch die verschiedenen Parameter aus den o.a. Definitionen lässt sich das Verhalten des Algorithmus vielfältig steuern. Ein wichtiger Aspekt ist dabei die Wahl der „richtigen" Parameter, um gute Lösungen effizient zu erzeugen. Eine Möglichkeit zur Parameterbestimmung ist bspw. Optimierungsalgorithmen (Genetischer Algorithmus, Ameisenalgorithmus) selbst auf die Anfangsparameter anzuwenden und so mehrere Läufe durchzuführen. Ein anderer Ansatz sind Experimente und analytische Korrelations- und Regressionsanalysen zwischen Parametern, Problemstruktur und

[7] (10 S. 21)
[8] (10 S. 21)

Lösung[9]. Aus solchen Untersuchungen ergaben sich folgend aufgeführt Ergebnisse zum Einfluss der Parameter.

Wenn die Rechenzeit vorgegeben ist, hat die *Größe m der Ameisenpopulation* Auswirkungen auf das Verhalten. Je höher die Zahl der Ameisen, desto tiefer wird das Netzwerk erkundet. Gleichzeitig können aber weniger Lösungen (Iterationen) erzeugt werden. Je kleiner die Zahl der Ameisen, umso mehr Iterationen können ausgeführt und so das bereits erworbene Wissen über das Netz ausgeschöpft/stabilisiert werden[10], auf Kosten der Erkundung des Lösungsraums.

Wird die *Verdunstung ρ der Pheromonspur* simuliert, kann darüber das Erkundungsverhalten gesteuert werden. Durch schnelle Verdunstung verliert eine einmal markierte Lösung innerhalb weniger Iterationen ihre Bevorzugung, so dass auch neue Lösungen mit höherer Wahrscheinlichkeit gefunden werden können. Durch eine geringe Verdunstungsrate kann hingegen erreicht werden, dass die Lösung schnell konvergiert[11].

Die beiden Exponenten α, β betonen die Unterschiede zwischen den Kanten bei der Bestimmung der Wahrscheinlichkeiten. Bei einem paarweisen Vergleich zwischen zwei Kanten $\langle i,j\rangle, \langle i,k\rangle$ ergibt sich aus **Definition 2**:

$$\frac{p_{ij}}{p_{ik}} = \frac{\left[\tau_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\left[\tau_{ik}\right]^{\alpha}\left[\eta_{ik}\right]^{\beta}} = \left(\frac{\tau_{ij}}{\tau_{ik}}\right)^{\alpha}\left(\frac{\eta_{ij}}{\eta_{ik}}\right)^{\beta}.$$

Die Brüche $\left(\tau_{ij}/\tau_{ik}\right)$ und $\left(\eta_{ij}/\eta_{ik}\right)$ beinhalten dabei die direkten Informationen über die Unterschiede (Markierungen und Längen) der beide Kanten. Durch die beiden Exponenten werden diese Informationen bei der Auswertung gewichtet und das Verhalten des Algorithmus gesteuert. $\alpha > \beta$ steht dabei für eine höhere Bedeutung der durch Iterationen erzeugten Lösungen und dynamischen Bewertungen der Kanten. $\alpha < \beta$ bedeutet hingegen, dass die statische heuristische Information über die Kantenlängen höheren Einfluss erhält[12].

2.3. Eigenschaften

Das durch o.a. Definitionen beschriebene Verhalten des AS-Algorithmus basiert auf dem individuellen simplen Verhalten der Individuen. Jede Ameise folgt einem sehr einfachen Programm und entscheidet lokal und momentan über ihren nächsten Knoten. Dabei beeinflusst aber jede dieser einfachen individuellen Entscheidungen das Verhalten aller Ameisen im folgenden Iterationsschritt. In der Summe konvergiert das Verhalten der Ameisen, und damit die Stärke der Pheromonspur, auf bestimmten Kanten des Netzes, was dann die Lösung nach h Iteration darstellt.

Allgemein gilt für sog. *Particle-Swarm-Optimization*-Algorithmen (PSO) durch ihre stochastische Natur, dass eine gute Lösung innerhalb einer gewünschten Zeit nur mit einer bestimmten Wahrscheinlichkeit möglich ist. Dieses Verhalten verschiedener Algorithmen kann durch sog. Laufzeit- bzw. Laufängen-Verteilungen (Run-Time-Distribution, Run-Length-Distribution) charakterisiert werden. Analysen verschiedener Schwarmoptimierungen zeigen, dass ab einer bestimmten Lauflänge (Iterationen) die

[9] (5 S. 204-205)
[10] (5 S. 206)
[11] (5 S. 206-207)
[12] (5 S. 208-209)

Wahrscheinlichkeit ansteigt, dass eine Lösung mit definierter Qualität gefunden wird. Dabei haben die Problemstruktur und die genutzte Strategie in vielen Fällen starken Einfluss auf die Entwicklung dieser Wahrscheinlichkeit. Ein langsamer Anstieg der Wahrscheinlichkeit oder Stagnation unterhalb von $p = 1$ deuten bei diesen Verteilungen auf Stagnation in einem lokalen Optimum hin, welches evtl. durch eine Neuinitialisierung verlassen werden kann.[13]

Ein Vorteil des ACO-Prinzips gegenüber anderen Metastrategien ist, dass das Prinzip eine einfache Struktur aufweist. Es müssen keine speziellen Nachbarschaftsstrukturen definiert werden, wie bei Simulated Annealing oder Tabu-Search, und man muss keinen angepassten Vererbungsvorgang beschreiben, wie beim Genetischen Algorithmus. Durch diese Eigenart verspricht die ACO-Metaheuristik eine vielfältige Anwendung unter gleich bleibender Charakteristik.

Ein Problem des AS-Algorithmus ist die vorzeitige Konvergenz in lokale Optima. Untersuchungen haben ergeben, dass sich die o.a. Standardform des Algorithmus beim TSP gegenüber anderen aktuellen Algorithmen als unterlegen erweist[14]. Um ein Verlassen lokaler Optima und das Aufsuchen neuer Lösungen zu ermöglichen, wurden verschiedene abgewandelte Versionen entwickelt, s. Abschnitt 2.4. Der AS-Algorithmus wurde dabei meist als Ausgangspunkt genutzt und erweitert.

2.3.1. Konvergenz

Ein wichtiger Aspekt von Heuristiken ist ihre Konvergenz gegen eine optimale Lösung. Für die ACO-Metaheuristik stellt sich dabei das grundsätzliche Problem, dass sie sehr allgemein formuliert ist, so dass ein allgemeiner Konvergenzbeweis nicht möglich ist[15]. Vielmehr können Konvergenzbetrachtungen für einzelne ACO-Algorithmen angestellt werden.

Zwei wichtige Konvergenzarten sind:

1. **Konvergenz der Werte/Größe** bedeutet, dass der Algorithmus mit Wahrscheinlichkeit $p = 1$ eine optimale Lösung (von mehreren möglichen) findet, bei ausreichender Laufdauer.
2. **Konvergenz der Lösung** sagt aus, dass der Algorithmus mit Wahrscheinlichkeit $p = 1$ die selbe optimale Lösung wieder erreicht.

Obwohl die *zweite Art* die stärkere Bedingung darstellt, reicht bei Optimierungsproblemen die *erste Art*, da das einmalige Auffinden der optimalen Lösung ausreicht und der Algorithmus danach stoppt[16].

Für eine vereinfachte verallgemeinerte Version des AS-Algorithmus wurde die Konvergenz in beiden o.a. Fällen nachgewiesen.[17]. Dabei wurde die heuristische Information η_{ij} aus **Definition 2** entfernt und der Term τ_{ij}^{α} für die Gewichtung der Pheromonspur durch eine *allgemeine monoton steigende* Funktion $F(\tau_{ij})$ ersetzt. In **Definition 5** wurde die Pheromonaktualisierung durch eine *allgemeine monoton fallende* Funktion $Q(T)$ in Abhängigkeit von der Qualität der gefundenen Lösung T ersetzt.

[13] (8 S. 1)
[14] (10 S. 68)
[15] (10 S. 121)
[16] (10 S. 122)
[17] (10 S. 123-134)

Ergebnisse dieser Analysen sind:

1. Die heuristische Information $0 < \eta_{ij} < +\infty$ hat keinen Einfluss auf die Konvergenz an sich[18].
2. Eine Pheromonuntergrenze $0 < \tau_{min} < \tau_{max} < +\infty$ zusammen mit einer Verdunstung $\rho > 0$ und einem oberen Limit für die Pheromonaktualisierung jeder Ameise $\Delta_{ij,h}^{k} \leq \Delta_{max}$ garantieren Konvergenz der ersten Art, da jede Lösung mit einer gewissen Wahrscheinlichkeit $P(T) > 0$ gefunden wird, wenn die Algorithmus lange genug läuft. Für den MMAS- und ACS-Algorithmus (s. Abs. 2.5.3 und Abs. 2.5.4) sind diese Bedingungen erfüllt[19], für die Algorithmen AS (s. Abs. 2.2.3) und EAS (s. Abs. 2.5.2) hingegen nicht.

Die angesprochenen Arten der Konvergenz sagen noch nichts über die Performanz der Algorithmen aus. Die erste Art der Konvergenz kann ebenso durch eine reine Zufallssuche erreicht werden, bei der auch jede Lösung mit einer gewissen Wahrscheinlichkeit generiert wird. Die Nachweise zeigen aber, dass sich ACO-Algorithmen zur Konvergenz zwingen lassen.

2.4. Anwendungsprinzipen der ACO-Metaheuristik

Die auf der allgemeinen ACO-Metaheuristik beruhenden vielfältigen Algorithmen wurden bereits auf eine Vielzahl von kombinatorischen Optimierungsproblemen angewandt. Aus diesen Anwendungen ergaben sich eine Reihe von Aspekten für die Bedeutung der einzelnen Problemkomponenten und die Lösungskonstruktion[20].

2.4.1. Graph

Allgemein wird eine graphische Repräsentation des aktuellen Problems benötigt. Dessen Entscheidungsvariablen (und deren Definitionsbereich) definieren die Komponenten des Graphs. Der zu untersuchende Graph selbst wird durch die vollständig verbundenen Komponenten erzeugt. Teilweise müssen sog. Dummyknoten mit eingefügt werden.

Beim TSP bspw. sind die Städte die Komponenten und werden als Knoten definiert. Der Graph wird durch die vollständige Verbindung aller Städte aufgespannt.

2.4.2. Pheromonspur

Man kann die Optimierungsprobleme in die beiden Klassen unterteilen, bei deren Lösung entweder die relativen Positionen der Knoten zueinander wichtig sind oder die absoluten Positionen in einer Reihe eine Rolle spielen.

Zur ersten Klasse zählt das TSP, in welchem die Reihenfolge der Knoten untereinander die Qualität der Lösung definiert und zyklische Permutationen der Lösung (also zyklische Verschiebung des Startpunkts) die Qualität nicht beeinflussen. Bei diesen Problemen trifft die Spur τ_{ij} eine Aussage über die Attraktivität Knoten j direkt nach Knoten i aufzusuchen.

In der zweiten Klasse ist die absolute Position der Knoten in der Lösung entscheidend, wie es bspw. bei Maschinenbelegungsproblemen (allg. Scheduling) eine Rolle spielt.

[18] (10 S. 134)
[19] (10 S. 136)
[20] (10 S. 211-218)

Hier können die Pheromonspuren τ_{ij} bei entsprechender Problemmodellierung der Positionen und Elemente als Aussage über die Vorteilhaftigkeit interpretiert werden, Element j an Position i zu setzen.

Beide Interpretationen haben andere Regeln zur Pheromonaktualisierung zur Folge, funktionieren aber auch bei Problemen der jeweils anderen Klasse, auf Kosten der Performanz.

2.4.3. Verhältnis zwischen Erkundung und Ausnutzung

Erkundung beschreibt die Eigenschaft eines Algorithmus den Lösungsraum möglichst tief und vollständig zu erkunden, also viele unterschiedliche und im Iterationsablauf neue Lösungen zu erzeugen. *Ausnutzung* ist die Eigenschaft bereits erworbenes Wissen über den Lösungsraum zu nutzen, um Lösungen und schnelle Konvergenz zu erzielen. Zwischen beiden Eigenschaften muss ein Gleichgewicht eingestellt werden, um das Iterationsverhalten an das aktuelle Problem anzupassen. Grundlegend ist am Anfang einer Iteration eine möglichst tiefe Erkundung des Lösungsraums gewünscht, wohingegen bei späteren Iterationen das bereits erzeugte Wissen genutzt werden sollte, um die Suche auf erfolgsversprechende Bereiche zu konzentrieren. Bei einer schlechten Balance kann der Algorithmus vorzeitig konvergieren oder stagnieren.

Die Steuerung dieses Verhaltens geschieht v.a. über die Pheromonspuren. Sowohl die Verdunstungsrate ρ also auch die Nutzung der Lösungsqualität (Länge/Kosten C der Tour) für die Pheromonaktualisierung steuern dieses Verhalten. Desweiteren spielen die Begrenzung der Spuren (s. Abs. 2.5.3 MMAS) oder die Betonung einzelner Lösungen (s. Abs. 2.5.2 EAS) beim Legen der Spur eine Rolle. Über die Parameter α, β wird in **Definition 2** die Verarbeitung der Pheromonspuren τ_{ij} und der heuristischen Informationen η_{ij} gesteuert. Durch eine dynamische Anpassung der beiden Parameter kann man im Iterationsverlauf diese Verarbeitung steuern.

2.4.4. Heuristische Information

Der Sinn hinter der Nutzung heuristischer Informationen ist problemspezifisches Wissen mit auszuwerten. Dabei gibt es die beiden Möglichkeiten *statische* oder *dynamische* heuristische Informationen zu nutzen.

Bei der statischen Variante wird die heuristische Information einmal berechnet, z.B. die Längen der einzelnen Kanten ($\eta_{ij} = 1/d_{ij}$), um dann bei jedem Iterationsschritt genutzt zu werden. Die dynamische Variante nutzt bei jedem Iterationsschritt Informationen aus der aktuellen Teillösung, um daraus neue heuristische Informationen über den Graphen zu erzeugen, z.B. eine vorausschauende Abschätzung der noch zurückzulegenden Weglänge bis zum Ziel. Die höhere Genauigkeit der dynamischen heuristischen Information über die Problemstruktur wird dabei durch einen höheren Rechenaufwand bei jeder Iteration erkauft. Genaue, aber rechenaufwändige, (dynamische) heuristische Informationen können gleichwertig oder besser durch die Kombination von ACO-Algorithmen mit *Local-Search*-Heuristiken ersetzt werden[21].

[21] (10 S. 216)

2.5. Abwandlungen und Erweiterungen

Durch das simple Grundprinzip der Ameisensysteme sind vielfältige Formulierungen für unterschiedliche Probleme möglich. Neben variantenreichen Ausprägungen des Grundprinzips können auch die Parameter des jeweiligen Algorithmus zur Steuerung des Verhaltens verändert werden. In diesem Abschnitt wird eine Auswahl der Abwandlungen vorgestellt.

2.5.1. Ant-Density, Ant-Quantity, Ant-Cycle (AS)

Für den grundlegenden AS-Algorithmus wurden die verschiedenen Arten der Pheromonaktualisierung *Ant-Density, Ant-Quantity* und *Ant-Cycle* entwickelt:

- **Ant-Density:** In dieser Version nimmt die Pheromonspur in jedem Zeitschritt auf den Kanten proportional zu den Ameisen auf den jeweiligen Kanten zu. Alle Ameisen werden gleich behandelt und legen gleich gewichtete Spuren.
- **Ant-Quanitity:** In dieser Variante des AS-Algorithmus ist die Zunahme der Pheromonspur auf einer gewählten Kante durch jede Ameise umgekehrt proportional zur Länge der Kante. Im Zeitablauf werden so kurze Kanten überproportional stark markiert und nachfolgende Ameisen werden zur Auswahl dieser kurzen Teilstücke an jedem Knoten motiviert. Diese Regel ähnelt der Nächster-Nachbar-Heuristik, bei welcher auch von jedem Knoten die kürzeste noch offene Kante gewählt wird.
- **Ant-Cycle:** Bei dieser AS-Variante werden die neuen Markierungen aller Ameisen erst nach Abschluss des aktuellen Suchlaufs berechnet. Die von jeder Ameise auf ihrem Weg gelegte Spur ist dabei umgekehrt proportional zur Länge ihrer zurückgelegten Tour. Der Vorteil dieses Ansatzes gegenüber Ant-Density und Ant-Quantity ist, dass bei der Bestimmung der Markierungen globale Informationen des gesamten Schwarms genutzt werden und kurze Strecken in jedem Lauf stärker betont werden als längere.

Ant-Cycle hat sich gegenüber den anderen beiden Varianten als deutlich überlegen gezeigt[22].

[22] (1 S. 18)

2.5.2. Elitäre Strategien (EAS)

Ein *elitäres Ameisensystem (Elitist Ant System, EAS)* verstärkt die bisher beste gefundene Lösung zusätzlich zu den schon gelegten Pheromonspuren. **Definition 5** wird dabei wie folgt um einen Summanden erweitert:

Definition 5b (Pheromonaktualisierung EAS)

$$\tau_{ij,h+1} = \tau_{ij,h}(1-\rho) + \sum_{k=1}^{m} \Delta_{ij,h}^{k} + e\Delta\tau_{ij}^{bs}$$

mit:

$$e\Delta\tau_{ij}^{bs} = \begin{cases} {}^{1}\!/_{C^{bs}} & \langle i,j\rangle \in T^{bs} \\ 0 & \text{sonst} \end{cases}$$

bs steht für *best-so-far* und markiert die beste bisher gefundene Tour T^{bs} und ihre Kosten (Länge) C^{bs}. e ist ein Gewichtungsparameter für diese Information.

Eine Abwandlung dieses Prinzips sind rangbasierte Markierungen, bei denen nur die r besten Ameisen eine nach ihrem Rang r gewichtete Spur legen dürfen.

2.5.3. MAX-MIN-Algorithmus (MMAS)

Mit der Variante *MAX-MIN-Ameisensystem (MAX MIN Ant System, MMAS)* versucht man zu verhindern, dass das System zu schnell in suboptimale lokale Lösungen konvergiert und diese nicht mehr verlassen kann. Der Ansatz führt folgende Variationen in den AS-Algorithmus ein:

1. Nur die bisher beste Ameise oder die beste Ameise der aktuellen Iteration darf eine Spur legen.
2. Die Pheromonwerte der Kanten werden auf ein Intervall $[\tau_{min}, \tau_{max}]$ beschränkt.
3. Die Anfangsinitialisierung der Pheromone wird auf den Maximalwert τ_{max} gesetzt.
4. Reinitialisierung der Pheromonspur und Neustart bei stagnierender Lösung.

Durch die erste Modifikation werden v.a. die besten Lösungen ausgenutzt. Modifikation zwei soll verhindern, dass die Lösung zu schnell in der ersten guten Lösung konvergiert. Die MAX-Bedingung erreicht, dass anfänglich (zufällig) gefundene kurze Touren (evtl. lokale Optima) nicht zu stark gewichtet werden und dann nicht mehr verlassen werden könnten, die MIN-Bedingung erreicht, dass anfänglich schlechte Touren nicht zu schlecht markiert werden, so dass sie bei weiteren Durchläufen mit einer gewissen Wahrscheinlichkeit weiterhin aufgesucht werden. Die dritte Modifikation bedeutet, dass während der ersten Iterationen nach dem Start der Lösungsraum großflächig durchsucht wird.

Definition 5 wird erweitert zu:

Definition 5c (Pheromonaktualisierung MMAS)

$$\tau_{ij,h+1} = \tau_{ij,h}(1-\rho) + \Delta\tau_{ij}^{best}$$

mit:

$$\Delta\tau_{ij}^{best} = \begin{cases} 1/C^{ib} & \langle i,j\rangle \in T^{ib} \\ 0 & \text{sonst} \end{cases} \quad \text{oder} \quad \Delta\tau_{ij}^{best} = \begin{cases} 1/C^{bs} & \langle i,j\rangle \in T^{bs} \\ 0 & \text{sonst} \end{cases}$$

bs und *ib* stehen für *best-so-far* und *iteration-best* und markieren die besten zugehörigen Touren T^{bs}, T^{ib} und ihre Kosten (Längen) C^{bs}, C^{ib}. Beide Aktualisierungskonzepte können auch kombiniert werden, z.B. so dass nach einer Anzahl von Iterationen das jeweils andere Konzept genutzt wird.

Die Obergrenze wird dynamisch definiert durch $\tau_{max} = 1/(\rho C^{bs})$, also durch die Länge C^{bs} der bisher besten Tour. Die Untergrenze wird als fester Teil des Maximalwerts dynamisch definiert durch $\tau_{min} = \tau_{max}/a$. Die Untergrenze ist primär verantwortlich, um eine frühzeitige Stagnation zu verhindern. Der Obergrenze kommt bei Start und Reinitialisierung eine bedeutende Rolle zu[23].

Unter allen Varianten scheint dieser Algorithmus über einen weiten Bereich gute oder beste Lösungen zu generieren[24].

2.5.4. Ant-Colony-System (ACS)

Ant Colony System (ACS, Ameisenkoloniesystem) unterscheidet sich von AS durch folgende Aspekte.

Die Entscheidung für den nächsten Knoten *j* nach Knoten i wird durch folgende Regel bestimmt:

Definition 2b (Knotenbestimmung ACS)

Ameise k wählt an Knoten i den nächsten Knoten j nach:

$$j = \begin{cases} \text{argmax}_{l \epsilon N_i^k} \left\{ \tau_{il} [\eta_{ij}]^{\beta} \right\} & \text{wenn } q \leq q_0 \\ J & \text{sonst} \end{cases}$$

Dabei sind:

- $l \epsilon N_i^k$ die Knoten l ,welche erlaubte Nachfolger des Knotens i für Ameise k sind,
- $q \epsilon [0,1]$ eine gleichverteilte Zufallsvariable und $0 \leq q_0 \leq 1$ ein Steuerparameter
- J ein Zufallsknoten nach **Definition 2** des AS-Algorithmus

Diese Formulierung erreicht, dass jede Ameise mit der Wahrscheinlichkeit q_0 die beste aktuell mögliche Entscheidung aus dem bisherigen Wissen trifft. Mit der Wahrscheinlichkeit $(1-q_0)$ trifft sie hingegen probalistisch eine Zufallsentscheidung nach **Definition 2** zur Erkundung aller erlaubten Wege.

[23] (10 S. 75-76)
[24] (7 S. 225), (1 S. 18)

Die Pheromonaktualisierung wird zweigeteilt durchgeführt:

Definition 5d **(Pheromonaktualisierung 1 ACS)**

$$\tau_{ij,h+1} = \tau_{ij,h}(1-\rho) + \rho\Delta\tau_{ij}^{best}$$

mit:

$$\Delta\tau_{ij}^{best} = {}^{1}/_{C^{bs}}, \quad \langle i,j\rangle \in T^{bs}$$

bs steht für *best-so-far* und markiert die beste zugehörige Tour T^{bs} und ihre Kosten (Länge) C^{bs}.

Definition 5e **(Pheromonaktualisierung 2 ACS)**

$$\tau_{ij,h+1} = \tau_{ij,h}(1-\xi) + \xi\tau_{ij,1}$$

mit:

$$0 < \xi < 1$$

ξ ist ein Steuerparameter.

Definition 5d wird *nach* jeder Iteration *nur* von der Ameise mit der besten bisherigen Lösung auf ihrer Tour angewandt, um so die beste Lösung zu betonen und eine schnelle Konvergenz zu erreichen. Durch die gleichzeitige Anwendung der Verdunstung ρ auf die alte und neue Markierung wird die neue Spur als gewichtetes Mittel berechnet. Außerdem reduziert sich die Zeitkomplexität dieses Updates gegenüber einem Update auf allen Kanten. **Definition 5e** wird hingegen von *jeder* Ameise während ihrer Erkundung *sofort* auf der von ihr gerade besuchten Kante angewandt. Dadurch wird deren Markierung etwas gesenkt, um direkt nachfolgende Ameisen anzuregen andere Kanten zu erkunden[25].

2.5.5. Kandidatenlisten

Um die Berechnung möglicher nächster Knoten bei großen Probleminstanzen zu beschleunigen, können so genannte Kandidatenlisten genutzt werden. Diese schränken alle möglichen Zielknoten nach einem heuristischen Kriterium auf die aussichtsreichsten Kandidaten ein und verkürzt die notwendigen Rechenschritte.

2.5.6. Kombination mit dem Genetischem Algorithmus

Klassische Ameisenalgorithmen erzielen gute Ergebnisse in statischen Netzwerken. In dynamischen Netzwerken hingegen unterliegen sie anderen Heuristiken. Durch den Versuch die uniformen Ameisen der klassischen Methode durch sich individuell entwickelnde Ameisen zu ersetzen, wird versucht die Ergebnisse in dynamischen Umgebungen zu verbessern. Durch Kombination mit dem *Genetischen Algorithmus (GA)* wird dabei erreicht, dass sich die Individuen einer anfangs homogenen Population individuell durch ihre ‚Erfahrung' diversifizieren und diese Eigenschaft an ihre Nachkommen weitergeben können. Der Genetische Algorithmus kann dabei

[25] (10 S. 77,78)

gleichzeitig auf alle Individuen einer Population angewandt werden, sobald alle Ameisen eine Lösung gefunden haben, oder kontinuierlich auf Ameisen mit individuellen Lebensspannen.

Ergebnisse zeigen, dass klassische Ameisenalgorithmen in statischen Netzwerken in 98% der Fälle die optimale Lösung finden, gegenüber 80% bei den erweiterten Algorithmen. Wird das Netzwerk nach dem Auffinden der ersten Lösung aber verändert, so dass ein neues Optimum existiert, verlassen klassische Algorithmen das lokale Optimum nur in 18% der Fälle, wohingegen die erweiterten Algorithmen dies in 62% der Fälle schaffen können [26].

Durch Reduktion der Mutationsrate auf Null, gehen die evolutionär erweiterten Ameisenalgorithmen in klassische über. Nach dem Auffinden einer ersten Lösung durch einen klassischen Algorithmus, kann diese gewonnene Lösung bei Veränderung des Netzwerks durch Aktivierung der Mutationsrate weiterentwickelt werden. So kann eine neue Lösung evtl. schneller erzeugt werden als bei einem völligen Neustart.

2.5.7. Parallelisierung

Durch die Nutzung von Computerclustern oder paralleler Prozessoren lässt sich einfach eine höhere Rechenleistung erzielen. Algorithmen und Metaheuristiken können entsprechend parallelisiert werden, um diese technischen Möglichkeiten auszunutzen. Bei ACO-Systemen wird dabei ein Problem von mehreren Kolonien gleichzeitig gelöst. Wichtige Aspekte sind dabei der Informationsaustausch zwischen den Kolonien und welchen Einfluss dies auf die Effizienz und Qualität der Lösung hat.

Bei parallelen Algorithmen ist die Unterscheidung nach folgenden Gesichtspunkten des Informationsaustauschs zwischen Kolonien möglich:

- Informationsobjekte (bspw. erzeugte Lösungen, gelegte Pheromonspuren, Parameter der Algorithmen),
- Informationsmenge (bspw. Austausch vollständiger Informationen, Austausch nur der besten Lösungen),
- Synchronisation (bspw. Kolonien warten auf bidirektionalen Informations-austausch, Kolonien setzen nach unidirektionalem Informationsaustausch eigene Suche fort),
- Kommunikationstopologie (bspw. alle Kolonien kommunizieren miteinander, nur Nachbarn kommunizieren, keine Kommunikation)[27]

Analysen zeigen, dass parallele Algorithmen bessere Lösungen schneller erzielen als gleichartige sequentielle Algorithmen bei gleicher Rechenzeit. Kommunizierende parallele Kolonien konvergieren schnell zur selben Lösung, da sie Informationen austauschen. Unabhängige parallele Kolonien durchsuchen den Lösungsraum tiefer und können unterschiedliche Lösungen erzeugen. Außerdem benötigen sie keine Rechenzeit und algorithmischen Anpassungen für die Kommunikation. Dadurch haben diese Algorithmen die bessere Performanz gegenüber kommunizierenden.[28] Der Kommunikationstopologie kommt eine hohe Bedeutung zu, sowohl für die Effizienz als auch die Lösungsqualität.

[26] (4 S. 513)
[27] (7 S. 224-226)
[28] (7 S. 228)

3. Anwendungsgebiet

Durch die einfachen und allgemeingültigen Prinzipien können ACO-Algorithmen in vielfältigen Bereichen angewandt werden. Ihre Stärken liegen in graphenbasierten Problemen. Neben direkten betriebswirtschaftlichen Optimierungsproblemen sind auch Anwendungen in den Bereichen Data-Mining, Netzwerk-Routing, Robotersteuerung, Kommunikation oder auch Proteindesign möglich und erfolgreich[29]. Die meisten Probleme, auf welche ACO-Algorithmen angewandt wurden, lassen sich in die Kategorien:

- Routenplanung (routing)
- Zuordnung (assignment)
- Ablaufplanung (scheduling)
- Teilmengen (subset)

einteilen. In den folgenden Abschnitten sollen ausgewählte Beispiele aus diesen Kategorien vorgestellt werden.

3.1. Routenplanung

Allgemein sind Probleme der Routenplanung so definiert, dass ein oder mehrere „Agent(en)“ eine Anzahl definierter Orte ansteuern muss/müssen. Die Güte der Zielfunktion wird durch die Reihenfolge der aufgesuchten Orte bestimmt.

Die Knoten des Graphen repräsentieren normalerweise die Orte und sind vollständig verbunden. Die Pheromonspur τ_{ij} trifft eine Aussage über die Attraktivität Ort j direkt nach Ort i zu besuchen.

3.1.1. Sequential Ordering

Sequential Ordering (SOP) sucht den kürzesten gewichteten Weg in einem gerichteten Graphen, der alle Knoten verbindet. Den Kanten werden Gewichte zugeordnet. Bei der Lösung müssen Reihenfolgebedingungen eingehalten werden.

Exakte Algorithmen können diese Probleme nicht effizient lösen. Einige ACO-Algorithmen zeigen bei der Lösung exzellente Performanzeigenschaften[30].

Der Graph wird durch vollständig verbundene Knoten plus einem Start- und einem Endknoten aufgebaut. Die Nebenbedingungen definieren, dass jeder Knoten genau einmal besucht wird und vorgegebene Reihenfolgebeziehungen eingehalten werden. Die Spur τ_{ij} gibt die Vorteilhaftigkeit an, Knoten j direkt nach Knoten i aufzusuchen. Die Berücksichtigung der Nebenbedingungen wird durch eine entsprechende Einschränkung der erlaubten Nachbarschaft für jede Ameise an den Knoten erreicht.

3.1.2. Vehicle Routing

Vehicle Routing (VRP) beschreibt ein Verteilungsproblem. Eine Anzahl von Kunden hat einen positiven Bedarf und muss mit einer Flotte gleichartiger Fahrzeuge von einem Depot aus bedient werden. Zwischen den Kunden bestehen Entfernungsbeziehungen. Ziel ist es Routen zu finden, welche die Reisezeit minimieren und alle Bedarfe

[29] (9)
[30] (10 S. 154)

abdecken. Dabei starten und enden alle Touren im Depot und jeder Kunde wird von nur einem Fahrzeug bedient.

Das VRP ist ein zweistufiges Problem. Einerseits muss eine möglichst kleine Zahl von Fahrzeugen mit ihren jeweiligen Touren (Zielen) bestimmt werden, so dass alle Kunden bedient werden. Andererseits muss für jedes Fahrzeug eine kürzeste Tour gefunden werden, was dem TSP entspricht. Die erste Zielsetzung hat dabei Vorrang vor der zweiten. Erweiterungen können Zeitfenster, Rücktransporte und Umladungen beinhalten.

Zur Lösung dieses zweistufigen Problems können zwei parallel arbeitende Ameisenkolonien eingesetzt werden, welche jeweils eine der beiden Stufen optimieren.

Verschiedene Implementationen der der ACO-Metaheuristik zeigten gleich gute und teilweise bessere Ergebnisse als andere Metaheuristiken, wie Tabu-Search[31].

3.2. Zuordnung

Bei Zuordnungsproblemen muss ein Satz von Objekten einem Satz von Ressourcen unter Nebenbedingungen zugeordnet werden, so dass eine Zielfunktion minimiert wird.

3.2.1. Generalized Assignment

Generalized Assignment (GAP) bezeichnet die allgemeine Formulierung von Zuordnungsproblemen. Einer Liste von Agenten müssen Elemente aus einer Liste von Tätigkeiten zugeordnet werden. Jeder Agent hat nur eine bestimmt Kapazität und mit jeder Zuordnung sind Kosten verbunden. Ziel ist eine Zuordnung, welche unter Einhaltung der Nebenbedingungen die Kosten minimiert.

Die Knoten des Graphen repräsentieren die Agenten und Tätigkeiten. Der Graph ist vollkommen verbunden. Die Pheromonspur τ_{ij} beschreibt die Vorteilhaftigkeit einer Zuordnung von Tätigkeit i zu Agent j. Bei der Lösungskonstruktion wird zufällig eine Tätigkeit i gewählt, um dann probalistisch einen Agenten j zuzuordnen.

ACO-Algorithmen zeigen bei diesen Problemen gute Performanz, sind aber aktuellsten Algorithmen unterlegen[32].

3.2.2. Quadratic Assignment

Quadratic Assignment (QAP) ist eine Erweiterung des GAP und beschreibt eine Zuordnung zwischen Anlagen und Orten, wobei zwischen den Orten Entfernungen und zwischen den Anlagen Flüsse simultan definiert werden. Die Summe aus den Produkten der Wege und Flüsse soll bei der Zuordnung minimiert werden. Eine Vielzahl von Problemen kann in diesem Schema abgebildet werden. QAPs stellen dabei die härtesten kombinatorischen Optimierungsprobleme dar und können nur unbefriedigend mit exakten Verfahren gelöst werden. Beispiele sind die Auslegung von (integrierten) elektronischen Schaltkreisen/Mikrochips, Maschinen- oder Einrichtungsplanung und Fließbandplanung.

Die Knoten stellen alle Orte und Anlagen dar, welche vollkommen verbunden sind. Eine Lösung besteht dann aus Verbindungspaaren zwischen Anlagen und Orten, welche jede Anlage und jeden Ort genau ein Mal zuordnen. Dabei werden den Verbindungen

[31] (10 S. 159), (1 S. 19), (12 S. 363)
[32] (10 S. 165)

keine Kosten zugeordnet. Die Spuren τ_{ij} beziehen sich auf die Vorteilhaftigkeit Ort i der Anlage j zuzuweisen.

ACO-Algorithmen gehören zu den besten existierenden Verfahren zur Lösung größerer QAP-Instanzen[33].

3.3. Ablaufplanung

In der Ablaufplanung werden knappe Ressourcen zu Tätigkeiten im Zeitablauf zugeordnet. Meistens werden Fertigungsabläufe durch diese Art der Optimierung erfasst. Verschiedene ACO-Algorithmen wurden zur Lösung von unterschiedlichen Ablaufoptimierungen angewandt. Die Ergebnisse (Qualität, Performanz) der Algorithmen schwankten dabei stark mit der Struktur des zu Grunde liegenden Problems[34].

Der Graph wird normalerweise durch einen vollständig verbundenen Satz aus Knoten zur Repräsentation der Jobs/Aufträge, Tätigkeiten und Positionen aufgespannt. Nebenbedingung ist, dass alle Jobs geplant werden müssen. Die Pheromonspur τ_{ij} gibt dann die Vorteilhaftigkeit an, dass Job j an Position i ausgeführt wird.

Typische Beispiele sind Maschinenbelegungsprobleme für einzelne oder mehrere Maschinen unter unterschiedlichen Nebenbedingungen. In dieselbe Kategorie fallen Projektplanungen.

3.3.1. Job Shop, Open Shop, Group Shop

Diese Probleme stammen aus der Maschinenbelegungsplanung und sind durch eine feste Zahl Tätigkeiten charakterisiert, welche in *Tätigkeiten je Maschine* und *Tätigkeiten je Auftrag* unterteilt werden. Die verschiedenen Arten *Job Shop (JSP), Open Shop (OSP), Group Shop (GSP)* unterscheiden sich dabei nur durch unterschiedliche Nebenbedingungen an die Reihenfolge der Tätigkeiten pro Auftrag. Ziel ist die Gesamtlänge aller Aufträge zu minimieren.

Die Knoten des Graphen stellen die einzelnen Tätigkeiten dar. Zusätzlich werden ein Start- und ein Endknoten definiert. Bei der Auswahl möglicher Knoten müssen Reihenfolgebeschränkungen zwischen den Tätigkeiten beachtet werden. Die Pheromonspuren τ_{ij} können, je nach Algorithmus, die Vorteilhaftigkeit ausdrücken Tätigkeit j direkt nach i auszuführen, oder Tätigkeit j nur generell, aber nicht zwingend direkt, nach i auszuführen.

Ergebnisse verschiedener ACO-Algorithmen zeigen, dass die Standardversionen schlechter arbeiten als andere aktuelle Algorithmen. Mit speziell angepassten Versionen, welche ACO mit Ideen aus anderen Heuristiken kombinieren, konnten gute Ergebnisse erzielt werden[35].

3.3.2. Projektplanung

Bei der Projektplanung müssen Tätigkeiten so angeordnet werden, dass die Gesamtzeit minimiert wird. Für jede Tätigkeit sind Ressourcenlimits und

[33] (10 S. 161, 163)
[34] (10 S. 168) (1 S. 18)
[35] (10 S. 177)

Reihenfolgebeschränkungen einzuhalten. In der Lösung muss jeder Tätigkeit eine Anfangs- und Endzeit zugeordnet werden.

Jeder Tätigkeitsknoten wird von zwei Dummyknoten für Anfangs- und Endzeit eingerahmt. Die Pheromonspur τ_{ij} beschreibt die Vorteilhaftigkeit Tätigkeit j an Stelle i auszuführen. Angepasste ACO-Algorithmen haben teilweise exzellente Ergebnisse geliefert, verglichen mit anderen Heuristiken[36].

3.4. Teilmengen

Bei diesen Optimierungsproblemen muss eine Teilmenge von Komponenten aus der Menge aller Komponenten so ausgewählt werden, dass unter Einhaltung von Nebenbedingungen eine Zielfunktion optimiert wird. Die Reihenfolge der gewählten Objekte hat hier keinen Einfluss auf die Lösungsqualität. Deshalb werden die Pheromonspuren hier üblicherweise den Komponenten (Knoten) zugeordnet und nicht den Verbindungskanten. Außerdem müssen nicht alle Lösungen die selbe Größe (gleiche Menge an Komponenten) haben.

3.4.1. (Multiples) Rucksackproblem

Rucksackprobleme beschreiben allgemein die Auswahl mehrerer Komponenten aus einer Grundmenge unter Verbrauch von Ressourcen, um eine Zielfunktion zu optimieren. Sind mehrere Kapazitätsgrenzen einzuhalten, spricht man vom multiplen Rucksackproblem. Beispiele aus diesem Bereich sind Beladungspläne (Lkw-, Schiffsladung) und Einrichtungsplanung (z.B. Büro-, Abteilungsaufteilung).

Bei diesen Problemen bilden die auswählbaren Komponenten die vollständig verbundenen Knoten des Graphen. Die Pheromonspuren τ_j sind mit den Knoten assoziiert und sagen aus, wie erstrebenswert es ist Komponente j der Lösung zuzuordnen. Jede Ameise kann bei jedem Schritt nur aus den Komponenten wählen, welche die Kapazitätsgrenzen nicht verletzen. Dadurch können Lösungen unterschiedlicher Länge entstehen.

Anwendungen verschiedener ACO-Algorithmen erreichten bei diesen Problemen keine konkurrenzfähige Performanz[37].

3.4.2. Set Covering Problem

Bei Set-Covering-Problemen sollen n Teilmengen S_i so vereint werden, dass eine geforderte Gesamtmenge U aus den S_i dargestellt werden kann (Überdeckung). Entweder sollen die Anzahl der genutzten Teilmengen oder die Kosten bei Auswahl der Teilmengen minimiert werden. Beispiele für diesen Bereich sind Standortplanungen, Eine Auswahl von Standorten muss ein bestimmtes Einzugsgebiet abdecken.

Jeder Knoten des Graphen steht für eine Teilmenge S_i. Alle Knoten sind vollständig verbunden. Bedingung ist, dass jeder Knoten nur ein Mal besucht wird und dass die aus den gewählten Knoten erzeugte Überdeckung der S_i die Zielmenge U ergibt. Die Spur τ_j beurteilt die Vorteilhaftigkeit Teilmenge j in die Lösung aufzunehmen.

Aktuelle problemspezifische Algorithmen zeigten sich den ACO-Algorithmen überlegen[38].

[36] (10 S. 180)
[37] (10 S. 188)

4. Ergebnis und Ausblick

Die ACO-Metaheuristik stellt ein allgemeines Rahmenwerk, in welchem die natürlichen Prinzipien der Wegfindung von Ameisenkolonien abgebildet sind, um konkrete ACO-Algorithmen zur Lösung von kombinatorischen Problemen zu entwickeln.

Zwei herausstechende Merkmale sind Einfachheit und Allgemeingültigkeit der Metaheuristik. Die allgemein zugrunde liegenden und abgebildeten Prinzipien erlauben eine vielfältige Ausgestaltung und breit gefächerte Anwendung des Konzepts. Vielfältige ACO-Algorithmen wurden entwickelt. Die ersten Umsetzungen bewiesen die Anwendbarkeit der Metaheuristik zur Lösung NP-harter Optimierungsprobleme, zeigten aber gleichzeitig Probleme auf, wie die vorschnelle Konvergenz in lokale Optima. Fortschreitende Erforschung und Entwicklungsarbeit führten schrittweise Verbesserungen ein, welche teilweise nicht mehr direkt aus dem empirisch beobachteten Verhalten von Ameisen abgeleitet werden konnten, aber zu einem besseren Verhalten der Algorithmen, besserer Performanz und höherer Lösungsqualität geführt haben. Für ein breites Spektrum an Anwendungen konnten ACO-Algorithmen vergleichbare und teilweise bessere Lösungen erzielen. Vor allem bei harten Problemen, für welche noch keine andere zufrieden stellende effiziente Heuristik entwickelt wurde, stellt ACO ein weiteres Werkzeug dar, um neue Wege aufzuzeigen.

Nach ersten Erfolgen bei akademischen Benchmark-Problemen konnte die ACO-Metaheuristik erfolgreich auf reale Probleme angewandt werden[39].

4.1. Verhalten der Algorithmen

Für die ACO-Metaheuristik lässt sich keine allgemeingültige Bewertung für die Lösung realer Probleme abgeben. Durch die vielfältigen Ausgestaltungs- und Steuerungsmöglichkeiten und das weite Anwendungsgebiet kann nicht pauschal auf die Leistungsfähigkeit eines ACO-Algorithmus für ein neues Problem geschlossen werden. Generell bietet die ACO-Metaheuristik so aber das Potential auch neue Probleme effizient lösen zu können.

Um den Anwendungsbereich besser eingrenzen zu können, ist ein tieferes Verständnis des Verhaltens der Algorithmen notwendig. Dabei können einfache Benchmarkprobleme als Vergleichsbasis dienen, um ACO-Algorithmen untereinander und mit anderen Techniken zu vergleichen. Gleichzeitig erlauben es einfache Problemstrukturen das Verhalten des jeweiligen Algorithmus nachzuvollziehen und zu verstehen. Ein weiterer wichtiger Punkt ist hierbei das Erkennen der Bedingungen, unter denen sich das Verhalten eines Algorithmus verschlechtert.

Diese Erkenntnisse können nicht nur zur Eingrenzung effizient lösbarer Problemkategorien führen, sondern gleichzeitig Hinweise auf eine adäquate Problemformulierung und eine zukunftsweisende Weiterentwicklung der Algorithmen geben.

[38] (10 S. 184), (11 S. 354)
[39] (10 S. 263)

4.2. Anwendung auf neue Problemkategorien

Neben den dargestellten Problemen, auf welche ACO-Algorithmen bereits erfolgreich angewandt werden konnten, ergeben sich neue Problemkategorien für aussichtsreiche Anwendungen der ACO-Metaheuristik[40]:

- **Dynamische / stochastische Probleme:** Bei dynamischen Problemen ändern sich Systemparameter (Nebenbedingungen, Steuerparameter, Netzwerk-struktur) mit der Zeit, bspw. Regelung von (Daten-)verkehr. Die Herausforderung ist, dass ein Algorithmus eine einmal gefundene Lösung nach der Parameteränderung wieder verlassen können muss, um den Lösungsraum neu zu erkunden und altes Wissen dabei weiter zu verwenden.
- **Echtzeitoptimierung:** Zusammen mit dynamischen Problem bietet sich die Anwendung auf Optimierungsprobleme in Echtzeit an, um reale Abläufe in Netzen direkt zu steuern.
- **Probleme mit mehrfacher Zielsetzung:** Mehrfache Zielsetzungen bezeichnen Probleme, bei denen mehrere Zielfunktionen gleichzeitig optimiert werden müssen und in Konflikt zueinander stehen. Um eine Kompromisslösung zu finden, müssen dabei die Prioritäten zwischen den einzelnen Zielen definiert und berücksichtigt werden.

4.3. Entwicklungstendenzen

Die ersten ACO-Algorithmen simulierten das empirisch nachgewiesene Verhalten von Ameisenkolonien noch relativ naturgetreu. Wachsendes Verständnis zusammen mit fortschreitender Entwicklung zeigen folgende Tendenzen für die Zukunft:

- **Algorithmische Optimierung:** Eine Abkehr vom reinen „Nachbau“ des realen Ameisenverhaltens kann die algorithmische Umsetzung und Weiterentwicklung optimieren. Damit können Probleme adäquat formuliert und die rechnergestützte Ausführung optimiert werden.
- **Algorithmische Kombination:** In die ACO-Algorithmen können Elemente anderer Heuristiken erfolgreich integriert werden, z.B. Local Search und heuristische Informationen. Damit können v.a. die Komponenten der Algorithmen verbessert werden, welche zu problematischem Verhalten führen, z.B. vorschnelle Konvergenz und schlechte Erkundung.
- **Modelle höherer Ordnung:** An Stelle einer einzelnen Pheromonspur, können mehrere Spuren verschiedener Pheromonarten gelegt werden, um unterschiedliche Beziehungen zwischen den Modellkomponenten abzubilden und mehr Wissen über den Lösungsraum zu erzeugen und zu verarbeiten[41].
- **Weitere intelligente Strategien:** Außer aus dem Wegfindungsverhalten der Ameisen lassen sich auch weitere Verhaltensweisen zur Ableitung neuer Metaheuristiken und intelligenter Strategien nutzen, z.B. Arbeitsteilung und gemeinsamer Transport in einer Kolonie. Damit können natürliche Prinzipien auf weitere Problemkategorien angewandt werden.

Der Erfolg der ACO-Metaheuristik wird sich durch weitere Anwendung auf reale Probleme in Zukunft beweisen. Ihr Wert im Vergleich zu bestehenden Techniken wird sich durch die Anwendung auf harte Probleme und o.a. Weiterentwicklungen zeigen.

[40] (10 S. 264-268)
[41] (13)

5. Anhang

5.1. Abkürzungsverzeichnis

ACO	*Ant Colony Optimization (Ameisenkolonieoptimierung)*
ACS	*Ant Colony System (Ameisenkoloniesystem)*
AS	*Ant System (Ameisensystem)*
EAS	*Elitist Ant System (Elitäres Ameisensystem)*
GAP	*Generalized Assignment Problem (Allgemeines Zuordnungsproblem)*
GSP	*Group Shop Problem*
JSP	*Job Shop Problem*
MMAS	*Max Min Ant System (Max-Min-Ameisensystem)*
NN	*Nearest Neighbour (Nächster Nachbar)*
OSP	*Open Shop Problem*
PSO	*Particle Swarm Optimization (Partikelschwarumoptimierung)*
QAP	*Quadratic Assignment Problem (Quadratisches Zuordnungsproblem)*
SOP	*Sequential Order Problem (Reihenfolgeanordnungsproblem)*
TSP	*Travelling Saleman Problem (Problem des Handlungsreisenden)*
VRP	*Vehicle Routing Problem (Fahrzeugroutenproblem)*

6. Literaturverzeichnis

1. **Bankofer, Udo und Hilbert, Andreas.** Kombinatorische Optimierung mit Ameisensystemen. *OR New - Das Magazin der GOR.* (Juli 1999), 6, S. 16-19.
2. **Rödder, Wilhelm, Kulmann, Friedhelm und Reidmacher, Hans Peter.** Problemlösen in graphischen Strukturen - Optimierung mit intelligenten Strategien. Hagen : Feruniversität Hagen, Fakultät für Wirtschaftswissenschaften, (2007).
3. **Neumann, Klaus und Ohse, Dietrich.** Problemlösen in graphischen Strukturen - Optimierung in Graphen Kurseinheit 1: Grundlagen der Graphentheorie. [Hrsg.] Fakultät für Wirtschaftswissenschaften FernUniversität Hagen. Hagen : s.n., (2007).
4. **Roach, Christopher und Menezes, Ronaldo.** Evolution in Swarm Intelligence: An Evolutionary Ant-Based Optimization Algorithm. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 512-513.
5. **Pellegrini, Paola, Favaretto, Daniela und Moretti, Elena.** On MAX-MIN Ant System's Parameters. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 203-213.
6. **Birattari, Mauro, Pellegrini, Paola und Dorigo, Marco.** On the Invariance of Ant System. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 215-223.
7. **Manfrin, Max, et al.** Parallel Ant Colony Optimization for the Travelling Salesman Problem. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 224-233.
8. **Montes de Oca, Marco A, et al.** A Comparison of Particle Swarm Optimization Algorithms Based on Run-Length Distributions. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : s.n., (2006), S. 1-12.
9. **Dorigo, Marco, et al.** Ant Colony Optimization and Swarm Intelligence. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006).
10. **Dorigo, Marco und Stützle, Thomas.** *Ant Colony Optimization.* Cambridge, Massachusetts; London, England : The MIT Press, (2004).
11. **Venables, Harry und Moscardini, Alfredo.** An Adaptive Search Heuristic for the Capacitated Fixed Chrage Location Problem. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 348-355.
12. **Xiangyong, Li und Peng, Tian.** An Ant Colony System for the Open Vehicle Routing Problem. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 356-363.
13. **Montgomery, James.** Higher Order Pheromone Models in Ant Colony Optimisation. [Buchverf.] Marco Dorigo, et al. *Ant Colony Optimization and Swarm Intelligence.* Brüssel, Belgien : Springer, (2006), S. 428-435.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit eigenständig und nur mit den angegebenen und erlaubten Hilfsmitteln erstellt habe. Wörtliche und inhaltliche Zitate sind, soweit gebraucht, kenntlich gemacht.

Unterschrift, 8. November 2008