Ameisenalgorithmen zur Lösung kombinatorischer Optimierungsprobleme

Thomas Stützle*
TU Darmstadt, FB Informatik
Alexanderstr. 10
64283 Darmstadt

Holger H. Hoos†
TU Darmstadt, FB Informatik
Alexanderstr. 10
64283 Darmstadt

8. Oktober 1999

Zusammenfassung

Ameisenalgorithmen sind ein neuer, populationsorientierter Ansatz zur Lösung kombinatorischer Optimierungsprobleme, der vom Verhalten echter Ameisenkolonien inspiriert ist. Das grundlegende Verfahren basiert auf der Kooperation künstlicher Agenten ("Ameisen"), die in indirekter Form mittels eines Mechanismus kommunizieren, der den von echten Ameisen benutzten Pheromonspuren nachempfunden ist. Diese algorithmische Idee wurde 1991 von Dorigo, Maniezzo und Colorni erstmals in Form des "Ant System" umgesetzt und auf das Handlungsreisendenproblem angewandt.

In diesem Artikel beschreiben wir die grundlegenden Ideen dieses neuen algorithmischen Ansatzes. Weiterhin diskutieren wir verschiedene Erweiterungen des ersten Ansatzes, mit deren Hilfe die Leistungsfähigkeit der Ameisenalgorithmen erheblich gesteigert werden kann. Insbesondere stellen wir das $\mathcal{MAX}$–$\mathcal{MIN}$-Ant-System vor, einen von uns entwickelten Ameisenalgorithmus, der momentan für verschiedene wichtige Probleme wie dem Handlungsreisendenproblem und dem quadratischen Zuordungsproblem die besten Ergebnisse unter den bislang vorgeschlagenen Erweiterungen erzielt.

*e-mail: stuetzle@informatik.tu-darmstadt.de

†e-mail: hoos@informatik.tu-darmstadt.de

1 Einführung

Ameisenalgorithmen sind, wie der Name nahelegt, vom Verhalten realer Ameisen inspiriert. Ameisen sind als soziale Insekten mit individuell eher geringen Fähigkeiten erst durch die Kooperation in einer Kolonie zu erstaunlich komplexen Leistungen im Stande. Hierzu zählt die Fähigkeit kürzeste Wege zwischen dem Nest und einer Futterquelle zu finden; diese bildete die ursprüngliche Inspiration für die ersten Ameisenalgorithmen [Dor92, DCG99]. Ein Beispiel für diese Fähigkeit realer Ameisen liefern die Experimente von Goss et al. [GADP89]. In diesen Experimenten wurde einer in einem Labor gehaltenen Kolonie von Ameisen eine Futterquelle zur Verfügung gestellt, die diese über eine "Brücke" mit zwei unterschiedlich langen Zweigen erreichen können (siehe schematische Darstellung in Abbildung 1). Die Experimente zeigen, daß nach einiger Zeit die meisten Ameisen den kürzeren Zweig nehmen und damit den kürzesten Weg zwischen dem Nest und der Futterquelle einschlagen. Verantwortlich für dieses Verhalten ist die indirekte Kommunikation zwischen den Ameisen mittels Pheromonspuren. Beim Laufen können Ameisen ihren Pfad mit Pheromonen markieren. Während isolierte Ameisen sich fast zufällig bewegen, kann eine Ameise, die auf eine Pheromonspur trifft, diese wahrnehmen und folgt ihr mit einer Wahrscheinlichkeit, die positiv mit der Stärke der Pheromonspur korreliert. Gleichzeitig kann die Ameise selbst wieder Pheromon ablegen und so die Spur verstärken. Zu Beginn des Experiments sind keine Pheromonspuren vorhanden, folglich wählen die Ameisen jeden der Zweige mit gleicher Wahrscheinlichkeit. Da jedoch einer der beiden Zweige kürzer ist, kommen die Ameisen, die diesen benutzt haben, schneller an der Futterquelle, bzw. — auf dem Rückweg — am Nest an. Auf diese Weise werden den kürzeren Weg mehr Ameisen pro Zeiteinheit benutzen, sodaß sich dort die Pheromonkonzentration gegenüber dem längeren Pfad allmählich erhöht. Durch diesen selbstverstärkenden Prozess werden nach einiger Zeit die meisten Ameisen den kürzeren Pfad wählen.

Die Optimierung mit Ameisenalgorithmen (engl. *Ant Colony Optimization*, kurz ACO) ist ein allgemein anwendbares, heuristisches Verfahren, das von dem zuvor beschriebenen Verhalten realer Ameisenkolonien inspiriert ist [DMC91, Dor92, DMC96, DC99]. Es handelt sich um einen populationsorientierten, kooperativen Ansatz, wobei die grundlegende Idee in der Benutzung eines algorithmischen Analogons zu den Pheromonspuren natürlicher Ameisen als Kommunikationsmittel für ein Multi-Agenten-System

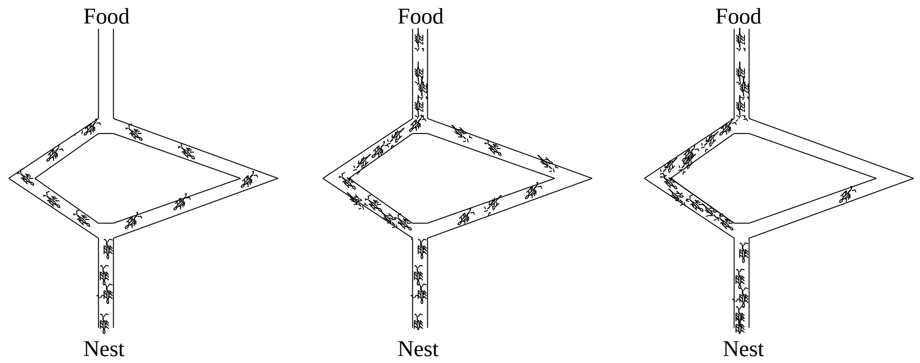

Abbildung 1: Schematische Darstellung des Brückenexperiments von Goss et al.

(eine Kolonie "künstlicher Ameisen") liegt.[1] Die "Ameisen" sind dabei einfache Agenten, die mittels stochastischer Entscheidungen, die lediglich auf der Verwendung lokal verfügbarer Information beruhen, Lösungen für die konkrete Problemstellung konstruieren. Dazu benutzen sie Information basierend auf Pheromonspuren, eventuell heuristische Informationen über die konkrete Problemstellung und können sich Teillösungen merken. Nach der Lösungskonstruktion werden dann die Pheromonspuren aktualisiert. Ameisenalgorithmen können somit als adaptive Algorithmen aufgefasst werden, deren zukünftiges Verhalten durch die in den Pheromonspuren gespeichterte Erfahrung aus früheren Iterationen beeinflußt wird. Die so realisierte Form der indirekten Kommunikation zwischen den Agenten ist charakteristisch für den algorithmischen Ansatz der Optimierung mit Ameisenalgorithmen.

Die bisherigen Anwendungen von Ameisenalgorithmen konzentrierten sich auf die Lösung $\mathcal{NP}$-harter Probleme.[2] *Ant System* (AS) [DMC96], die erste algorithmische Umsetzung dieses Prinzips zur Lösung kombinatorischer Optimierungsprobleme, wurde zunächst anhand der Beispielsanwendung auf das Handlungsreisendenproblem (engl. *Traveling Salesman Problem*, kurz TSP) eingeführt [CDM91, DMC91, Dor92, DMC96]. Dabei konnte experimentell gezeigt werden, daß der Ansatz prinzipiell zur Lösung des TSP geeignet ist; es konnten jedoch nur für relativ kleine Instanzen mit bis zu ca. 50 Städten gute Ergeb-

[1] NB: Das Ziel ist dabei nicht die Simulation des Verhalten realer Ameisenkolonien sondern die Umsetzung des Verhaltens zur effizienten Lösung harter kombinatorischer Optimierungsprobleme.

[2] Für einen Überblick über Arbeiten auf diesem Gebiet sei auch auf die *Ant Colony Optimzation home page* unter `http://iridia.ulb.ac.be/dorigo/ACO/ACO.html` verwiesen.

nisse erzielt werden. In den letzten Jahren wurden jedoch Erweiterungen dieses Algorithmus vorgestellt, mit deren Hilfe zum Teil erheblich bessere Ergebnisse erzielt werden können. Unter diesen Erweiterungen sind das *Ant Colony System* (ACS) [GD96, DG97], das $\mathcal{MAX}$–$\mathcal{MIN}$-*Ant-System* ($\mathcal{MM}$AS) [SH97] und das *rank-based version of Ant System* (AS_{rank}) [BHS99c]. Diese verbesserten Varianten von AS haben eine stärkere Ausnutzung der besten gefundenen Lösungen gemeinsam; sie unterscheiden sich aber in wesentlichen Aspekten der Suchkontrolle, insbesondere in der Kombination von Aspekten der Suchintensivierung und -diversifikation. Zusätzlich kann die Leistungsfähigkeit dieser Algorithmen deutlich gesteigert werden, indem die von den Ameisen konstruierten Lösungen mit Hilfe lokaler Suchverfahren (nach-)optimiert werden [MDC94, SH97, DG97]. Deshalb sind die erfolgreichsten ACO-Algorithmen *hybride Algorithmen*, die eine Phase der Lösungskonstruktion mit einer nachfolgenden lokalen Suche verbinden.

Generell folgen die besten ACO-Algorithmen für *statische*, $\mathcal{NP}$-harte kombinatorische Optimierungsprobleme [3] dem in Abbildung 1 dargestellten algorithmischen Schema. Nach der Initialisierung der Pheromonstärken und einiger Parameter wird eine Schleife wiederholt, bis eine Abbruchbedingung erfüllt ist (z.B. durch Erreichen einer Schranke in der Anzahl an Lösungskonstruktionen oder der CPU-Zeit). In jedem Schleifendurchlauf konstruieren alle Ameisen zuerst eine zulässige Lösung unter Berücksichtigung der aktuellen Pheromonspuren. Diese Lösung wird dann eventuell mit Hilfe eines lokalen Suchverfahrens nachoptimiert und anschließend werden die Pheromonstärken aktualisiert. Die einzelnen Komponenten werden in Abschnitt 3 anhand der Anwendung auf das TSP näher erläutert. Es sei jedoch bemerkt, daß die in [DC99, DCG99] vorgestellte ACO-Metaheuristik etwas allgemeiner als das hier angegebene algorithmische Schemata ist. So werden beispielsweise die am Ende dieses Artikels angesprochenen Anwendungen von ACO-Algorithmen auf inherent *dynamische* Probleme, wie dem Routing in Kommunikationsnetzwerken, von dem hier angegebenen algorithmen Rahmen nicht abgedeckt.

Der Rest des Artikels ist folgendermaßen aufgebaut. Im nächsten Abschnitt führen wir zunächst das TSP ein, anhand dessen im folgenden der grundlegende AS-Algorithmus erläutert wird. Anschließend beschreiben wir $\mathcal{MM}$AS, die von uns vorgeschlagene Erweiterung des AS, wobei wiederum das TSP

[3] *Statische kombinatorische Optimierungprobleme* sind diese, bei denen alle relevanten Problemdaten vor dem Start des Algorithmus verfügbar sind und sich nicht dynamisch während des Algorithmusablaufs ändern.

```
procedure Ameisenalgorithmus
    Setze Parameter, Initialisiere Pheromonstärken
    while (Abbruchbedingung nicht erfüllt) do
        Konstruiere Lösungen
        Lokale Suche        % optional
        Erneuere Pheromon
    end
end Ameisenalgorithmus
```

Abbildung 2: Algorithmischer Rahmen für Ameisenalgorithmen zur Lösung kombinatorischer Optimierungsprobleme.

als Beispielanwendung dient. Sodann verdeutlichen wir die Leistungsfähigkeit von $\mathcal{MM}$AS anhand von empirischen Ergebnissen aus Anwendungen zur Lösung von TSPs und quadratischen Zuordnungsproblemen (engl. *Quadratic Assignment Problems*, kurz QAPs). Der Artikel schließt mit einer Übersicht über bisherige Anwendungen von Ameisenalgorithmen in Abschnitt 7 und einer kurzen Zusammenfassung.

2 Das Handlungsreisendenproblem

Das TSP ist eines der bekanntesten kombinatorischen Optimierungsprobleme [LLKS85, Rei94]. Intuitiv geht es beim TSP darum, eine kürzeste Rundreise durch n Städte zu finden. Formal ist eine TSP-Instanz durch einen vollständigen gewichteten Graphen $G = (\mathcal{V}, \mathcal{A}, d)$ mit der Knotenmenge $\mathcal{V}$ und der Kantenmenge $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V}$ gegeben, wobei A für jedes Paar von Knoten eine Kante enthält; d ist eine Gewichtsfunktion, die jeder Kante (i, j) ein positives Gewicht $d(i, j)$ zuweist. Dabei entsprechen die Knoten den Städten, die Kanten den Verbindungen zwischen den Städten und die Gewichte $d(i, j)$ den Entfernungen zwischen Städten i und j. TSP-Instanzen, bei denen für alle Paare von Knoten die Gleichheit $d(i, j) = d(j, i)$ gilt, heißen *symmetrische TSPs*; im allgemeineren Fall, in dem diese Symmetrie

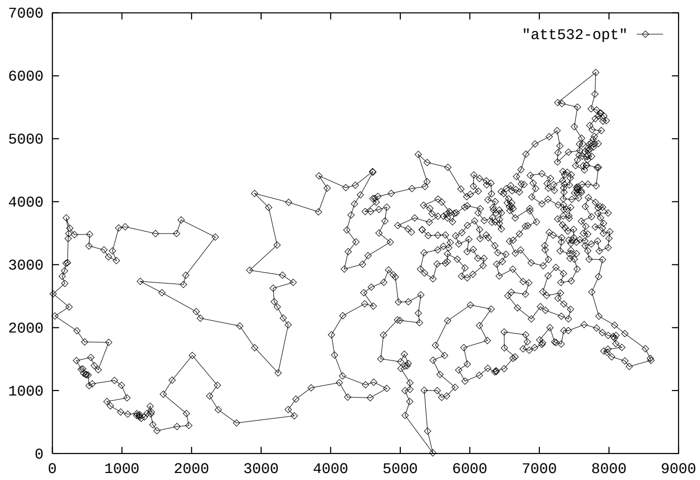

Abbildung 3: TSP-Instanz `att532`, bestehend aus 532 Städten in den USA, aus der Benchmark-Bibliothek TSPLIB. Die angegebene Tour ist optimal.

der Gewichtsfunktion nicht gegeben ist, handelt es sich um ein *asymmetrisches TSP*. *Euklidische TSPs* sind symmetrische TSP-Instanzen, deren Kantengewichte als Abstände der Knoten in der euklidischen Ebene interpretiert werden können. Abbildung 1 zeigt eine euklidische TSP-Instanz und ihre optimale Lösung (kürzeste Rundreise).

In der Entwicklung von ACO-Algorithmen spielt das TSP eine bedeutende Rolle, da es als Referenzproblem zur empirischen Analyse von AS sowie der Mehrzahl aller später entwickelten Verbesserungen diente. Dabei wurde das TSP aus mehreren Gründen gewählt: (*i*) Die biologische Motivation basiert, wie in Abschnitt 1 gezeigt wurde, auf einer sehr ähnlichen Problemstellung und ACO-Algorithmen sind entsprechend leicht auf das TSP anzuwenden; (*ii*) das TSP ist ein anwendungsrelevantes, $\mathcal{NP}$-hartes [GJ79] Optimierungsproblem; (*iii*) das TSP ist intuitiv leicht verständlich und wird generell häufig als Referenzproblem zum Testen neuer algorithmischer Ideen verwendet.

Die symmetrischen TSP-Instanzen, die für die später beschriebene experimentelle Untersuchung verwendet werden, sind sämtlichst der TSPLIB entnommen, einer Benchmarkbibliothek mit einer Vielzahl von Instanzen, von denen viele aus praktischen Anwendungen des TSP, wie dem Bohren von Leiterplatten, stammen.

3 Ant-System

Ant System (AS) [Dor92, DMC96], der erste Ameisenalgorithmus zur Lösung von kombinatorischen Optimierungsproblemen, wurde wie viele andere, von natürlichen Vorgängen inspirierte Algorithmen, für das TSP als Beispielanwendung entwickelt. Dabei sind mit jeder Kante (i, j) des gegebenen Graphen positive, reellwertige Variablen τ_{ij} assoziiert, welche die Stärke der Pheromonspuren repräsentieren. Diese werden von einer Anzahl m künstlicher Ameisen bei der Konstruktion von Lösungen der gegebenen Probleminstanz verwendet und aufgrund der gefundenen Lösungen aktualisiert. Zusätzlich zu den Pheromonspuren können die Ameisen eine heuristische Information η_{ij} zur Lösungskonstruktion verwenden; diese ist im Fall des TSP als Funktion der Entfernung zweier Städte gegeben. Jede Iteration des Algorithmus besteht aus zwei Phasen: Der Lösungskonstruktion und der Aktualisierung der Pheromonspuren. Diese beiden Phasen werden wiederholt, bis eine Abbruchbedingung, zum Beispiel eine maximale Anzahl an Iterationen, erfüllt ist.

3.1 Konstruktion der Lösungen

Zur Konstruktion der Lösungen wird jede Ameisen auf eine zufällig ausgewählte Stadt gesetzt. Wenn Ameise k in Stadt i ist, geht sie in ihrem nächsten Schritt zu einer Stadt j gemäß einer Wahrscheinlichkeit:

$$p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\cdot\eta_{ij}^{\beta}}{\sum_{k\in\mathcal{N}_i^k}\tau_{ik}^{\alpha}(t)\cdot\eta_{ik}^{\beta}} & \text{falls Ameise } k \text{ Stadt } j \text{ noch nicht besucht hat} \\ \\ 0 & \text{sonst} \end{cases} \tag{1}$$

Dabei ist $\tau_{ij}(t)$ die Stärke der Pheromonspur auf Kante (i, j) in Iteration t; $\eta_{ij} = 1/d(i, j)$ ist ein heuristisches Maß, das umso größer ist, je kleiner die Distanz zwischen Städten i und j ist; und $\mathcal{N}_i^k$ ist die zulässige Nachbarschaft, in diesem Fall die von Ameise k noch nicht besuchten Städte. Es werden also Verbindungen bevorzugt, die kurz sind und/oder eine hohe Pheromonstärke aufweisen. Zusätzlich wird die Wahrscheinlichkeitsverteilung durch die beiden Parameter α und β beeinflußt, die den relativen Einfluß der Pheromonstärke und der heuristischen Information bestimmen. Zusätzlich hat jede Ameise eine beschränktes Gedächtnis in Form einer Liste, die verwendet wird partielle Touren zu speichern und in jedem Konstruktionsschritt die zulässige Nachbarschaft zu bestimmen.

3.2 Aktualisierung der Pheromonspuren

Nachdem jede Ameise eine vollständige Lösung konstruiert und die Länge ihrer Rundreise L_k berechnet hat, werden die Pheromonspuren aktualisiert. In AS dürfen alle Ameisen eine konstante Gesamtmenge Q an Pheromon auf die von ihnen benutzten Kanten legen. Die neue Pheromonstärke $\tau_{ij}(t+1)$ einer Kante wird dann berechnet zu:

$$\tau_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{2}$$

Dabei ist ρ ist die Persistenz der Pheromonstärke, wobei $1-\rho$ der Evaporation der Pheromonspuren entspricht. Je kleiner ρ ist, desto schneller wird die indirekt in den Pheromonstärken gespeichere Information aus vorangegangenen Iterationen vergessen. $\Delta\tau_{ij}^{k}(t)$ modelliert das in Iteration t von Ameise k auf Kante (i,j) abgelegte Pheromon, dieses ist gegeben durch:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} 1/L^k(t) & \text{falls Kante } (i,j) \text{ von Ameise } k \text{ benutzt wird} \\ \\ 0 & \text{sonst} \end{cases} \tag{3}$$

Dabei ist $L^k(t)$ die Tourlänge der von Ameise k konstruierten Tour. Durch diesen Mechanismus zur Aktualisierung der Pheromonstärken erhalten Kanten, die häufig in kurzen Touren verwendet werden, mehr Pheromon und werden deshalb auch mit einer höheren Wahrscheinlichkeit in den folgenden Iterationen des Algorithmus gewählt werden. Die Motivation hierfür liegt in der Idee, daß bei der Lösung kombinatoririscher Optimierungsprobleme bessere Lösungen oft durch eine geschickte Kombination bewährter Lösungselemente (hier: Kanten) gefunden werden können.

4 $\mathcal{MAX}$–$\mathcal{MIN}$-Ant-System

Mit AS konnten für kleinere TSP Instanzen mit bis zu 50 Städten optimale oder wenigstens sehr gute Näherungslösungen gefunden werden. Jedoch sind die Laufzeiten des Algorithmus relativ hoch und die erreichbare Lösungsgüte für größere Instanzen ist nicht zufriedenstellend, weshalb in den letzten Jahren verschiedene Erweiterungen des ursprünglichen Algorithmus vorgestellt wurden. Eine dieser Erweiterungen ist $\mathcal{MM}$AS, ein neues Verfahren, das direkt auf der grundlegenden Arbeit über AS aufbaut. $\mathcal{MM}$AS unterscheidet sich in drei Gesichtspunkten von AS.

(i) Während in AS alle Ameisen an der Aktualisierung der Pheromonspuren mitwirken, ist in $\mathcal{MM}$AS nur noch eine einzige Ameise pro Iteration daran beteiligt. Diese Ameise kann diejenige sein, die die bisher beste Rundreise seit dem Start des Algorithmus gefunden hat (*global-beste*) oder diejenige, die die beste Lösung in der aktuellen Iteration des Algorithmus gefunden hat (*iterations-beste*). Durch diese Einschränkung wird erreicht, daß sich der Algorithmus schneller auf vielversprechende Kanten konzentriert. Experimentelle Ergebnisse haben dabei gezeigt, daß die Performanz von $\mathcal{MM}$AS am besten ist, falls zwischen global-besten und iterations-besten Ameisen so ausgewählt wird, daß sich die Häufigkeit, mit der die global-beste Ameise gewählt wird, während des Ablaufs des Algorithmus erhöht.

(ii) In $\mathcal{MM}$AS werden die möglichen Werte der Pheromonstärke für jede einzelne Kante auf ein Intervall $[\tau_{\min}, \tau_{\max}]$ eingeschränkt. Auf diese Weise kann eine Stagnation der Suche vermieden werden, die eintritt, wenn die Phoromonstärke auf einigen wenigen Kanten sehr hoch gegenüber den übrigen Kanten ist. In solch einem Fall würden bei der Lösungskonstruktion nur noch wenige Kanten berücksichtigt werden während die meisten anderen praktisch nicht benutzt werden. Experimentelle Ergebnisse haben dabei gezeigt, daß die untere Schranke für die Pheromonstärke dabei wichtiger ist. Für die maximale Pheromonstärke auf beliebigen Kanten (i, j) kann man generell einen Grenzwert $\tau^*_{\max} = \overline{\lim}_{t\to\infty}\tau_{ij}(t) = Q/((1-\rho)L_{opt})$ herleiten, wobei L_{opt} die Länge der optimalen Tour ist [Stü98]. Für $\tau_{\max}$, die obere Schranke der Pheromonstärken, verwenden wir eine Schätzung dieses Wertes, indem L_{opt} durch $L_{global\text{-}beste}$, die Länge der global-besten Tour, ersetzt wird. $\tau_{\min}$, die untere Schranke für die Pheromonstärke, wird um einen konstanten Faktor kleiner als $\tau_{\max}$ gewählt.

Durch die Benutzung der Schranken für die Pheromonstärke wird erreicht, daß jede Kante eine feste minimale Wahrscheinlichkeit hat, gewählt zu werden. Es sei noch bemerkt, daß man, ähnlich wie für Simulated Annealing, durch die Grenzen für die Pheromonstärken auch garantieren kann, daß approximativ die optimale Lösung gefunden werden kann.

(iii) In $\mathcal{MM}$AS werden die Pheromonstärken die Pheromonstärken mit $\tau_{\max}$ initialisiert, was zu einer höheren Exploration von Touren beim Start des Algorithmus führt und die erreichbare Lösungsgüte weiter verbessert [Stü98].

Darüber hinaus werden in $\mathcal{MM}$AS grundsätzlich lokale Suchverfahren verwendet, um die Performanz des Algorithmus weiter zu steigern. Dadurch ist $\mathcal{MM}$AS ein hybrides Verfahren, bei dem die Ameisen benutzt werden, um günstige Ausgangslösungen für die lokale Suche zu generieren. Im allgemeinen haben alle ACO-Algorithmen die Eigenschaft, daß Informationen über gute Lösungen vergangener Iterationen indirekt in Form der Pheromonspuren gespeichert werden. Die Lösungskonstruktion mittels der Ameisen wird dann benutzt, um vielversprechende Kanten geeignet zu kombinieren und so gute Anfangslösungen für die lokale Suche zu generieren.

5 Experimentelle Ergebnisse für symmetrische TSPs

In diesem Abschnitt präsentieren wir einige Ergebnisse, die mit $\mathcal{MM}$AS für symmetrische TSP Instanzen erzielt wurde und vergleichen wir die Ergebnisse mit denen eines iterierten lokalen Suchverfahren (ILS) [MOF91, JM97]. ILS basiert auf der iterativen Anwendung einer lokalen Suchmethode auf Lösungen, die aus kleinen Veränderungen an vorher gefundenen lokalen Minima hervorgehen; üblicherweise wird dabei das beste bisher gefundene lokale Minimum verwendet. Unter Verwendung der Lin-Kernighan Heuristik ist ILS der wohl beste exisitierende Approximationsalgorithmus für das symmetrische TSP [JM97]. Jedoch ist die Lin-Kernighan lokale Suche sehr aufwendig zu implementieren, weswegen wir hier eine 3-opt lokale Suche verwenden; 3-opt basiert auf der schrittweisen Verbesserung der aktuellen Lösung durch wiederholten Austausch von maximal drei Kanten. Unsere 3-opt Implementierung benutzt Standardtechniken zur Beschleunigung der lokalen Suche: Wir verwenden *don't look bits* und eine Nachbarschaftssuche mit begrenztem Radius, Techniken die zum Beispiel in [Ben92, JM97] beschrieben sind.

Für $\mathcal{MM}$AS wurden 25 Ameisen verwendet; alle Ameisen verbessern nach jeder Iteration ihre Lösung durch eine 3-opt lokale Suche. In $\mathcal{MM}$AS verwenden wir zur Beschleunigung der Lösungskonstruktion *Kandidatenlisten* [Rei94], die für jede Stadt ihre 20 nächsten Nachbarn enthalten. Soweit möglich, wählen die Ameisen bei der Konstruktion von Touren die jeweils nächste Stadt aus der entsprechenden Kandidatenmenge; falls dies nicht möglich ist (d.h., wenn alle Städte aus der Kandidatenmenge schon besucht wurden), wird die Stadt mit maximalem $\tau_{ij}(t) \cdot \eta_{ij}$ gewählt. Weiterhin wurden generell die Parameterwerte $\alpha = 1, \beta = 2$ und $\tau_{\min} = \tau_{\max}/2n$ benutzt.

In Tabelle 5 sind die Ergebnisse für einige TSP-Instanzen mit bekanntem Optimum aus der Problemsammlung TSPLIB angegeben; die Laufzeiten wurden auf einer Sun UltraSparc II Workstation (ein Prozessor, 167MHz, 192MB Arbeitsspeicher) ermittelt. Die Ergebnisse für $\mathcal{MM}$AS sind die besten, die bislang mit ACO-Algorithmen für das TSP erzielt wurden. Man sieht, daß $\mathcal{MM}$AS generall sehr hohe Lösungsgüten erreicht und für fast alle Instanzen optimale Lösungen finden kann; weiterhin erzielt $\mathcal{MM}$AS auf den meisten Instanzen eine zum Teil deutlich höhere mittlere Lösungsgüte als ILS.

Insgesamt zeigen diese Ergebnisse, daß Ameisenalgorithmen und speziell das $\mathcal{MM}$AS sehr vielversprechende Ergebnisse auf dem TSP erreichen können. Wie zuvor angedeutet, existieren für das TSP aber Lösungsverfahren, die eine bessere Performanz als der hier verwendete ILS-Algorithmus aufweisen; zu erwähnen sind dabei insbesondere ILS-Algorithmen, die für die lokale Suche die Lin-Kernighan Heuristik verwenden oder einige wenige Implementierungen genetischer Algorithmen, die mit lokalen Suchverfahren hybridisiert sind, sogenannte memetische Algorithmen [MF97, Wal98]. Es ist jedoch zu erwarten, daß die Leistungsfähigkeit von $\mathcal{MM}$AS durch die Verwendung einer noch effizienteren lokalen Suche (z.B. Lin-Kernighan) oder alternativer Möglichkeiten der Lösungskonstruktion (bzw. auch der Lösungsmanipulation wie in [GTD99] beschrieben) ebenfalls weiter gesteigert werden kann.

Tabelle 1: Ergebnisse des Vergleichs von $\mathcal{MM}$AS mit (ILS) auf symmetrischen TSP Instanzen aus der TSPLIB. Gegeben ist der Instanzname (die Zahl im Instanznamen entspricht der Anzahl der Städte), der benutzte Algorithmus, die beste gefundene Lösung, die mittlere Lösungsgüte (in Klammer die Abweichung vom Optimum in Prozent), die schlechteste gefundene Lösung, die mittlere Zeit (t_{avg}) um die beste Lösung in einem Lauf zu finden, und die maximal erlaubte CPU-Zeit (t_{max}). Die Ergebnisse beruhen auf 25 unabhängigen Läufen des Algorithmus für $n < 1000$, bzw. über 10 Läufen für die größeren Instanzen. Die jeweils besten Ergebnisse sind fettgedruckt.

Instance	Algorithm	Best	Average	Worst	t_{avg}	t_{max}
d198	$\mathcal{MM}$AS	15780	15780.4 (0.0)	15784	61.7	170
	ILS	15780	**15780.2 (0.0)**	15784	18.6	
lin318	$\mathcal{MM}$AS	42029	**42029.0 (0.0)**	42029	94.2	450
	ILS	42029	42064.6 (0.09)	42163	55.6	
pcb442	$\mathcal{MM}$AS	50778	**50911.2 (0.26)**	51047	308.9	600
	ILS	50778	50917.7 (0.28)	51054	180.7	
att532	$\mathcal{MM}$AS	27686	**27707.9 (0.08)**	27756	387.3	1250
	ILS	27686	27709.7 (0.09)	27759	436.2	
rat783	$\mathcal{MM}$AS	8806	**8814.4 (0.10)**	8837	965.2	2100
	ILS	8806	8828.4 (0.34)	8850	1395.2	
u1060	$\mathcal{MM}$AS	224455	224853.5 (0.34)	225131	2577.6	3600
	ILS	224152	**224408.4 (0.14)**	224743	1210.4	
pcb1173	$\mathcal{MM}$AS	56896	**56956.0 (0.11)**	57120	3219.5	5400
	ILS	56897	57029.5 (0.24)	57251	1892.0	
d1291	$\mathcal{MM}$AS	50801	**50821.6 (0.04)**	50838	1894.4	5400
	ILS	50825	50875.7 (0.15)	50926	1102.9	
fl1577	$\mathcal{MM}$AS	22286	**22311.0 (0.28)**	22358	3001.8	7200
	ILS	22311	22394.6 (0.65)	22505	3447.6	

6 Experimentelle Ergebnisse für das QAP

Beim quadratischen Zuordnungsproblem (engl. *Quadratic Assignment Problem*, kurz QAP) müssen n Objekte auf n verschiedene Positionen plaziert werden. Das Ziel ist es, bei bekanntem Fluß a_{ij} von Objekt i nach Objekt j und einer bekannten Entfernung b_{rs} zwischen den Positionen r und s, eine Zuordnung zu finden, die die folgende Zielfunktion minimiert:

$$S = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\phi(i)\phi(j)}$$

Dabei ist $\Phi(n)$ die Menge aller Permutationen von $\{1, \ldots, n\}$ und $\phi(i)$ gibt die Position von Objekt i in der Lösung $\phi \in \Phi(n)$ an.

Das QAP ist wie das TSP ein $\mathcal{NP}$-hartes Optimierungsproblem. Es ist eines der am schwierigsten zu lösenden Optimierungsprobleme, da generell Instanzen mit $n \geq 20$ heutzutage nicht zur Optimalität gelöst werden können. Zur praktischen Lösung von QAP sind daher heuristische Algorithmen, die sehr gute Näherungslösungen in kurzer Zeit finden können, von großer Bedeutung. Vor diesem Hintergrund ist die Anwendung von Ameisenalgorithmen zur Lösung von QAP sehr attraktiv. Dieser Ansatz wurde, unter der Verwendung von AS, bereits in [MDC94] verfolgt; aufgrund der Überlegenheit von $\mathcal{MM}$AS gegenüber AS sind die im folgenden präsentierten, mit $\mathcal{MM}$AS erzielten Ergebnisse jedoch deutlich besser.

Bei der Anwendung von $\mathcal{MM}$AS auf das QAP beziehen sich die Pheromone auf die Zuordnung eines Objekts auf eine bestimmte Position, d.h. τ_{ij} gibt die Tendenz an, Objekt i auf die Position j zu setzen. Zur Konstruktion von Lösungen wird in einem ersten Schritt zufällig ein Objekt ausgewählt, das dann im zweiten Schritt einer noch freien Position zugewiesen wird. Die Zuweisung eines Objekts auf eine Position (es wird bei dieser Anwendung keine heuristische Information berücksichtigt) für Ameise k folgt der folgenden Wahrscheinlichkeitsverteilung:

$$p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}(t)}{\sum_{\mathcal{N}_i^k} \tau_{ik}(t)} & \text{falls Position } j \text{ bei Ameise } knochfrei \\ \\ 0 & \text{sonst} \end{cases} \tag{4}$$

Als lokale Suche verwenden wir hier kurze Läufe eines *Tabu Search*-Algorithmus (Robust Tabu Search (Ro-TS) [Tai91]) für das QAP.

Im folgenden vergleichen wir Ergebnisse für das $\mathcal{MM}$AS und zwei Tabu-Search-Algorithmen für das QAP. Dabei unterscheiden wir zwischen zufällig erzeugten, unstrukturierten Instanzen und strukturierten "real-world" Instanzen, die aus Anwendungen des QAP stammen. Bei unstrukturierten Instanzen besteht insbesondere die Flußmatrix aus gleichverteilten Zufallszahlen. Bei "real-world" Instanzen dagegen weist die Flußmatrix eine deutliche Struktur auf – Interaktionen zwischen einigen wenigen Objekten sind oft hoch – und die Einträge der Flußmatrix weichen deutlich von einer Gleichverteilung ab. Die verwendeten Tabu-Search-Algorithmen sind *Robust Tabu Search (Ro-TS)* [Tai91] und *Reactive Tabu Search (RTS)* [BT94]; beide gehören zu den besten Algorithmen für das QAP (siehe [Tai95] für eine ausführliche Diskussion). Für die Tabu-Search-Algorithmen werden jeweils bis zu $1000n$ lokale Suchiterationen erlaubt; $\mathcal{MM}$AS wendet maximal 250 mal eine kurze Tabu-Suche der Länge $4n$ an. Da die lokale Suche selbst sehr laufzeitintensiv ist, ist die CPU-Zeit für alle Algorithmen fast genau die gleiche. Wegen der geringen Anzahl der lokalen Suchen verwendet $\mathcal{MM}$AS hier nur 5 Ameisen. Die anderen Parameter wurden wie in Abschnitt 5 gewählt; die einzige Ausnahme ist Parameter β, da für das QAP keine heuristische Information verwendet wird.

Aus den Ergebnissen in Tabelle 6 ist ersichtlich, daß RTS der beste Algorithmus für die nicht strukturierten `taixxa` Instanzen ist, aber auf den anderen Instanzen zum Teil deutlich schlechter als Ro-TS und $\mathcal{MM}$AS abschneidet. Insgesamt verhält sich $\mathcal{MM}$AS auf den nicht strukturierten Instanzen ähnlich wie Ro-TS. Auf den strukturierten Instanzen hingegen liefert $\mathcal{MM}$AS die besten Ergebnisse. $\mathcal{MM}$AS ist hier offensichtlich in der Lage, die Struktur dieser Probleme auszunutzen und somit auch bessere Lösungen zu finden. (Eine ausführlichere Diskussion dieser Ergebnisse findet sich in [Stü97b].)

Tabelle 2: Ergebnisse für Instanzen aus QAPLIB; die Zahl im Instanzbezeichner gibt die Problemgroße (Anzahl der Objekte) an. Als Ergebnis ist die mittlere prozentuale Abweichung von den besten bekannten Lösungen, jeweils gemittelt über 10 Versuche, angegeben.

Instanz	**Ro-TS**	**RTS**	$\mathcal{MM}$AS
nicht strukturierte Instanzen			
nug30	0.013	0.892	0.020
sko72	0.146	0.558	0.087
sko81	0.136	0.406	0.077
sko90	0.128	0.530	0.132
sko100a	0.108	0.429	0.133
tai60a	1.203	0.822	1.121
tai80a	0.900	0.497	0.986
tai100a	0.894	0.361	1.017
strukturierte Instanzen			
bur26a-h	0.002	0.092	**0.0**
kra30a	0.268	1.048	**0.0**
kra30b	0.023	0.128	**0.015**
ste36a	0.155	2.043	**0.061**
tai50b	0.342	0.426	**0.277**
tai60b	0.417	0.364	**0.137**
tai80b	1.030	1.033	**0.682**
tai100b	0.512	0.555	**0.092**

7 Weiter Arbeiten über Ameisenalgorithmen

Bisherige Arbeiten über Ameisenalgorithmen haben zu einer weiter steigenden Anzahl erfolgreicher Anwendungen auf kombinatorische Optimierungsprobleme geführt. Allgemein lassen sich Ameisenalgorithmen als Multi-Agenten Systeme charakterisieren, bei denen die einzelnen Agenten stochastische Entscheidung auf der Basis lokal verfügbarer Information treffen. Die Interaktion der Agenten basiert auf indirekter Kommunikation mittels eines Mechanismus, der als *Stigmergy* [Gra59] bekannt ist. Eine allgemeine Einführung in Ameisenalgorithmen kann in Bonabeau, Dorigo, und Theraulaz [BDT99] gefunden werden.

Die im Zusammenhang mit Ameisenalgorithmen am besten untersuchten Probleme sind das TSP und das QAP, auf das wir auch $\mathcal{MM}$AS angewandt haben. Für beide Probleme ist AS [DMC96] der zuerst angewandte Ameisenalgorithmus; für einen Überblick über bisherige Anwendungen von Ameisenalgorithmen auf diese beiden Probleme verweisen wir hier auf [SD99a] und [SD99b]. Generell sind für das TSP die bisher mit Ameisenalgorithmen erzielten Ergebnisse zwar sehr gut, sie erreichen aber bislang dennoch nicht den Leistungsstand der besten TSP-Algorithmen. Für das QAP hingegen zählen Ameisenalgorithmen zu den besten verfügbaren Lösungsverfahren.

Das *Sequential Ordering Problem* ist ein eng mit dem asymmetrischen TSP verwandtes Problem, bei dem zusätzlich Präzedenzrelationen zwischen den Knoten erfüllt werden müssen. Gambardella und Dorigo haben dieses Problem mit einer Erweiterung ihres ACS in Kombination mit einem speziell entwickelten lokalen Sucheverfahren gelöst [GD97]. Dabei konnten exzellente Ergebnisse erzielt werden und für viele Benchmarkinstanzen gelang eine Verbesserung der besten bekannten Lösungen.

Eine erste Anwendung von AS auf das *Job-Shop Scheduling Problem* ist in [CDMT94] beschrieben. Dort konnte gezeigt werden, daß Ameisenalgorithmen auch auf dieses Problem angewandt werden können, die Ergebnisse waren allerdings nicht vergleichbar zu denen von *“state-of-the-art”* Algorithmen. $\mathcal{MM}$AS wurde in [Stü97a] auf das*Flow Shop Scheduling Problem (FSP)* angewandt. Für dieses Problem wurden deutlich bessere Ergebnisse erzielt als mit früher entwickelten Simulated-Annealing-Algorithmen und die erreichbare Lösungsqualität war vergleichbar mit Tabu-Search-Algorithmen, die momentan zu den besten verfügbaren Algorithmen für das FSP gehören.

Costa und Herz [CH97] haben eine Erweiterung von AS auf Zuordungsprobleme vorgestellt und ihren Ansatz auf Graphfärbeprobleme angewandt. Die Ergebnisse waren vergleichbar mit denen anderer Ansätze, jedoch muß sich erst noch zeigen, inwieweit Ameisenalgorithmen auf diesem Problem konkurenzfähig zu den besten bekannten Algorithmen sind. Erst kürzlich wurde $\mathcal{MM}$AS von Ramalhinho Lorençou und Serra auf das generalisierte Zuordungsproblem [RS98] erweitert und erreichte sehr gute Ergebnisse auf einem weiten Bereich von Benchmarkinstanzen.

Anwendungen von AS_{rank} auf Vehicle-Routing-Probleme wurden von Bullnheimer, Hartl, und Strauss [BHS99a, BHS99b] vorgestellt. Sie erzielten gute Ergebnisse auf einigen weit verbreiteten Benchmarkinstanzen, die den von den besten Algorithmen für diese Probleme erreichten Ergebnissen nur leicht nachstanden. Eine erst kürzlich veröffentlichte Anwendung auf Vehicle-Routing-Probleme mit Zeitfenstern von Gambardella, Taillard, und Agazzi konnte sogar die bisher besten bekannten Lösungen für etliche Instanzen verbessern [GTA99].

Eine weitere, sehr vielversprechende Anwendung von ACO-Algorithmen auf das *Shortest Common Supersequence* Problem wurde von Michel und Middendorff vorgestellt [MM98]. Eine Neuheit bei diesem speziellen Algorithmus ist, daß die Ameisen bei der Lösungskonstruktion eine "lookahead"-Funktion verwenden. Zusätzlich verwenden sie eine parallele Implementierung ihres Algorithmus, die auf dem *island model*, einem oft bei der parallelen Implementierungen genetischer Algorithmen benutzten Ansatz, beruht.

Eine Großzahl von Arbeiten beschäftigen sich mit Anwendungen von ACO-Algorithmen auf Probleme aus dem Kommunikationsbereich, insbesondere mit Routing in Kommunikationsnetzwerken. Beispiele hierfür sind die Arbeiten von Schoonderwoerd et al. [SHBR96] und Di Caro und Dorigo [DD98]). Die Anwendung von ACO-Algorithmen auf diese Art von Problemen ist besonders interessant, da diese Probleme Charakteristiken wie verteilte Information, nicht-stationäre stochastische Dynamik und eine asynchrone Entwicklung des Netzwerkzustandes haben, die geradezu den charakteristischen Eigenschaften von ACO-Algorithmen entsprechen. Ein ausführlicher Überblick über Anwendungen auf Netzwerkroutingprobleme kann in Dorigo, Di Caro und Gambardella [DCG99] gefunden werden.

8 Zusammenfassung

In diesem Artikel beschreiben wir den grundlegenden Ansatz der Optimierung mit Ameisenalgorithmen, einer 1991 von Dorigo, Maniezzo und Colorni eingeführten algorithmischen Idee, die vom Verhalten natürlicher Ameisenkolonien inspiriert ist. Verschiedene Ameisenalgorithmen sind in den letzten Jahren mit wachsendem Erfolg zur Lösung einer Vielzahl von kombinatorischen Problemen eingesetzt worden. "Ant System", der erste Ameisenalgorithmus auf dem die überwiegende Anzahl aller weiteren Verfahren aufbaut, wurde auf dem Handlungsreisendenproblem eingeführt und findet für kleinere Instanzen optimale oder wenigstens sehr gute Näherungslösungen. Jedoch sind die Laufzeiten des Algorithmus relativ hoch und für größere Instanzen ist die erreichbare Lösungsgüte nicht zufriedenstellend. Vor diesem Hintergrund entwickelten wir das in diesem Artikel beschriebene *$\mathcal{MAX}$–$\mathcal{MIN}$-Ant-System*, einen neuen Ameisenalgorithmus, der wesentliche Schwächen von *Ant System* behebt und generell bessere Ergebnisse erzielt. Die hier vorgestellten Versuchsergbnisse für Handlungsreisendenprobleme und quadratische Zuordnungsprobleme belegen dies: für beide Probleme zeigt $\mathcal{MM}$AS sehr gute Ergebnisse, die mit denen der besten für diese Probleme bekannten Algorithmen vergleichbar sind. Im Licht dieser Ergebnisse erscheint $\mathcal{MM}$AS insgesamt als ein vielversprechender allgemeiner ACO-Algorithmus, der erfolgreich zur Lösung kombinatorischer Optimierungsprobleme eingesetzt werden kann.

Literatur

[BDT99] E. Bonabeau, M. Dorigo und G. Theraulaz. *From Natural to Artificial Swarm Intelligence.* Oxford University Press, 1999.

[Ben92] J.L. Bentley. Fast Algorithms for Geometric Traveling Salesman Problems. *ORSA Journal on Computing*, 4(4):387–411, 1992.

[BHS99a] B. Bullnheimer, R.F. Hartl und C. Strauss. Applying the Ant System to the Vehicle Routing Problem. In S. Voss, S. Martello, I.H. Osman und C. Roucairol, Herausgeber, *Meta-Heuristics:*

Advances and Trends in Local Search Paradigms for Optimization, Seiten 285–296. Kluwer, Boston, 1999.

[BHS99b] B. Bullnheimer, R.F. Hartl und C. Strauss. An Improved Ant System Algorithm for the Vehicle Routing Problem. *Annals of Operations Research*, 1999. Zur Veröffentlichung angenommen.

[BHS99c] B. Bullnheimer, R.F. Hartl und C. Strauss. A New Rank Based Version of the Ant System — A Computational Study. *Central European Journal for Operations Research and Economics*, 1999. Zur Veröffentlichung angenommen.

[BT94] R. Battiti und G. Tecchiolli. The Reactive Tabu Search. *ORSA Journal on Computing*, 6(2):126–140, 1994.

[CDM91] A. Colorni, M. Dorigo und V. Maniezzo. Distributed Optimization by Ant Colonies. In *Proceedings of the First European Conference on Artificial Life (ECAL 91)*, Seiten 134–142. Elsevier, 1991.

[CDMT94] A. Colorni, M. Dorigo, V. Maniezzo und M. Trubian. Ant System for Job-Shop Scheduling. *Belgian Journal of Operations Research, Statistics and Computer Science*, 34(1):39–53, 1994.

[CH97] D. Costa und A. Hertz. Ants Can Colour Graphs. *Journal of the Operational Research Society*, 48:295–305, 1997.

[DC99] M. Dorigo und G. Di Caro. The Ant Colony Optimization Meta-Heuristic. In D. Corne, M. Dorigo und F. Glover, Herausgeber, *New Ideas in Optimization*. McGraw-Hill, 1999.

[DCG99] M. Dorigo, G. Di Caro und L.M. Gambardella. Ant Algorithms for Distributed Discrete Optimization. *Artificial Life*, 1999. Zur Veröffentlichung angenommen.

[DD98] G. Di Caro und M. Dorigo. AntNet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research (JAIR)*, 9:317–365, 1998.

[DG97] M. Dorigo und L.M. Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.

[DMC91] M. Dorigo, V. Maniezzo und A. Colorni. Positive Feedback as a Search Strategy. Technischer Bericht 91-016, Dip. Elettronica, Politecnico di Milano, 1991.

[DMC96] M. Dorigo, V. Maniezzo und A. Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 26(1):29–42, 1996.

[Dor92] M. Dorigo. *Optimization, Learning, and Natural Algorithms (auf italienisch)*. Dissertation, Dip. Elettronica, Politecnico di Milano, 1992.

[GADP89] S. Goss, S. Aron, J.L. Deneubourg und J.M. Pasteels. Self-organized Shortcuts in the Argentine Ant. *Naturwissenschaften*, 76:579–581, 1989.

[GD96] L.M. Gambardella und M. Dorigo. Solving Symmetric and Asymmetric TSPs by Ant Colonies. In *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'96)*, Seiten 622–627. IEEE Press, 1996.

[GD97] L.M. Gambardella und M. Dorigo. HAS-SOP: Hybrid Ant System for the Sequential Ordering Problem. Technischer Bericht IDSIA 11-97, IDSIA, Lugano, Schweiz, 1997.

[GJ79] M.R. Garey und D.S. Johnson. *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness*. Freeman, San Francisco, CA, 1979.

[Gra59] P. P. Grassé. La Reconstruction du Nid et les Coordinations Interindividuelles chez *bellicositermes natalensis et cubitermes sp.* La Théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs. *Insectes Sociaux*, 6:41–81, 1959.

[GTA99] L.M. Gambardella, É. Taillard und G. Agazzi. MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In D. Corne, M. Dorigo und F. Glover, Herausgeber, *New Ideas in Optimization*. McGraw-Hill, 1999.

[GTD99] L.M. Gambardella, É.D. Taillard und M. Dorigo. Ant Colonies for the QAP. *Journal of the Operational Research Society*, 1999. 50:167-176, 1999.

[JM97] D.S. Johnson und L.A. McGeoch. The Travelling Salesman Problem: A Case Study in Local Optimization. In E.H.L. Aarts und J.K. Lenstra, Herausgeber, *Local Search in Combinatorial Optimization*, Seiten 215–310. John Wiley & Sons, 1997.

[LLKS85] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan und D.B. Shmoys. *The Travelling Salesman Problem*. John Wiley & Sons, 1985.

[MDC94] V. Maniezzo, M. Dorigo und A. Colorni. The Ant System Applied to the Quadratic Assignment Problem. Technischer Bericht IRIDIA/94-28, Université de Bruxelles, Belgien, 1994.

[MF97] P. Merz und B. Freisleben. Genetic Local Search for the TSP: New Results. In *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'97)*, Seiten 159–164. IEEE Press, 1997.

[MM98] R. Michel und M. Middendorf. An Island Based Ant System with Lookahead for the Shortest Common Supersequence Problem. In *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature*, Band 1498 in *LNCS*, Seiten 692–708. Springer Verlag, 1998.

[MOF91] O. Martin, S.W. Otto und E.W. Felten. Large-Step Markov Chains for the Traveling Salesman Problem. *Complex Systems*, 5(3):299–326, 1991.

[Rei94] G. Reinelt. *The Traveling Salesman: Computational Solutions for TSP Applications*, Band 840 in *LNCS*. Springer Verlag, 1994.

[RS98] H. Ramalhinho Lourenço und D. Serra. Adaptive Approach Heuristics for the Generalized Assignment Problem. Technischer Bericht Economic Working Papers Series No.304, Universitat Pompeu Fabra, Dept. of Economics and Management, Barcelona, May 1998.

[SD99a] T. Stützle und M. Dorigo. ACO Algorithms for the Quadratic Assignment Problem. In D. Corne, M. Dorigo und F. Glover, Herausgeber, *New Ideas in Optimization.* McGraw-Hill, 1999.

[SD99b] T. Stützle und M. Dorigo. ACO Algorithms for the Traveling Salesman Problem. In K. Miettinen, M.M. Mäkelä, P. Neittaanmäki und J. Periaux, Herausgeber, *Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming, Genetic Programming and Industrial Applications.* John Wiley & Sons, 1999.

[SH97] T. Stützle und H.H. Hoos. The $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System and Local Search for the Traveling Salesman Problem. In T. Baeck, Z. Michalewicz und X. Yao, Herausgeber, *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'97)*, Seiten 309–314, 1997.

[SHBR96] R. Schoonderwoerd, O. Holland, J. Bruten und L. Rothkrantz. Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior*, 5(2):169–207, 1996.

[Stü97a] T. Stützle. An Ant Approach to the Flow Shop Problem. In *Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98)*, Band 3, Seiten 1560–1564. Verlag Mainz, Aachen, 1997.

[Stü97b] T. Stützle. $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System for the Quadratic Assignment Problem. Technischer Bericht AIDA–97–4, FG Intellektik, TU Darmstadt, July 1997.

[Stü98] T. Stützle. *Local Search Algorithms for Combinatorial Problems — Analysis, Improvements, and New Applications.* Dissertation, TU Darmstadt, FB Informatik, 1998.

[Tai91] É.D. Taillard. Robust Taboo Search for the Quadratic Assignment Problem. *Parallel Computing*, 17:443–455, 1991.

[Tai95] É.D. Taillard. Comparison of Iterative Searches for the Quadratic Assignment Problem. *Location Science*, 3:87–105, 1995.

[Wal98] T. Walters. Repair and Brood Selection in the Traveling Salesman Problem. In A.E. Eiben, T. Bäck, M. Schoenauer und H.-P. Schwefel, Herausgeber, *Proc. of Parallel Problem Solving from Nature - PPSN V*, Band 1498 in *LNCS*, Seiten 813–822. Springer Verlag, 1998.