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

Zusammenfassung

Ameisenalgorithmen sind ein neues, populationsorientiertes Optimierungsverfahren, das vom Verhalten echter Ameisenkolonien bei der Futtersuche inspiriert ist. Die grundlegende Idee dieses Ansatzes besteht in der algorithmischen Übertragung der von echten Ameisen zur Kommunikation benutzten Pheromonspuren. Die grundlegende Arbeit zur Optimierung mit Ameisenkolonien bildet Ant System, die erste algorithmische Umsetzung dieses Prinzips, das anfangs auf Traveling Salesman Probleme (TSP) angewandt wurde.

In diesem Artikel beschreiben wir die grundlegenden Ideen dieses neuen Verfahrens und stellen eine Erweiterung von Ant System, das $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System, vor. Experimentelle Ergebnisse zeigen, daß das $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System äußerst erfolgreich zur Lösung $\mathcal{NP}$-schwerer Optimierungsprobleme angewandt werden kann.

1 Einführung

In den letzten Jahren zeigte sich ein starkes Interesse an Verfahren zur Lösung $\mathcal{NP}$-schwerer kombinatorischer Optimierungsproblemen, die von natürlichen Vorgängen inspiriert sind. Zu diesen Verfahren gehören Simulated Annealing [KJV83], Evolutionäre Algorithmen [Bäc96] und Neuronale Netze [HT85]. Eines der neuesten dieser naturanalogen Verfahren ist die Optimierung mit Ameisenkolonien (ACO, engl. Ant Colony Optimization) [DMC96].[1]

Dieses Gebiet ist vom Verhalten realer Ameisen bei der Futtersuche inspiriert. Diese fast blinden Tiere mit nur einfachen Fähigkeiten sind durch die Zusammenarbeit in einer Kolonie in der Lage, kooperatives Verhalten zu zeigen und ürzeste Wege zwischen zwei Punkten, zum Beispiel ihrer Heimatkolonie und einer Futterquelle, zu finden. Der Informationsaustausch zwischen den Ameisen geschieht dabei durch Geruchstoffe, sogenannte *Pheromone*. Eine Ameise kann Pheromon in variierender Stärke ablegen und auf diese Weise einen Weg markieren. Während isolierte Ameisen sich fast zufällig bewegen, kann eine Ameise, die auf eine Pheromonspur trifft, diese wahrnehmen und folgt ihr mit einer um so höheren Wahrscheinlichkeit, desto stärker diese Pheromonspur ist. Gleichzeitig kann die Ameise selbst wieder Pheromon ablegen und so die Spur verstärken. So ergibt sich ein positive Rückkopplungsschleife; je mehr Ameisen einem Pfad folgen, desto attraktiver wird dieser für andere Ameisen.

Ant System (AS) [DMC96] ist die erste algorithmische Umsetzung dieses Prinzips zur Lösung kombinatorischer Optimierungsprobleme.[2] Ant System ist ein populationsorientierter, kooperativer Algorithmus. Die grundlegende Idee basiert auf einem algorithmischen Analogon zu den Pheromonspuren natürlicher Ameisenkolonien, die als ein Kommunikationsmittel für eine Kolonie künstlicher Ameisen benutzt werden. Die künstlichen Ameisen sind einfache Agenten, die Lösungen für die konkrete Problemstellung konstruieren. Dazu benutzen sie Information basierend auf Pheromonspuren, eventuell heuristische Informationen über das konkrete Problem und können sich Teillösungen merken. Nach der Lösungskonstruktion werden die Pheromonspuren aktualisiert. Dies geschieht, indem diese zuerst um einen konstanten Faktor erniedrigt werden. Auf diese Weise wird die Evaporation der Pheromonspuren nachgebildet. In einem nachfolgenden Schritt erhalten die Lösungsattribute der konstruierten Lösungen ein *Reinforcement* dessen Stärke von der Lösungsgüte abhängt.

Bei die Anwendung des Ant System auf das Hand-

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

[1] Für einen Überblick über Arbeiten in diesem Gebiet sei auf die Ant Colony Optimzation Seite unter http://iridia.ulb.ac.be/dorigo/ACO/ACO.html verwiesen.

[2] Das Ziel dieser Algorithmen ist wohlbemerkt nicht die Simulation des Verhalten realer Ameisenkolonien, sondern die algorithmische Umsetzung dieses Verhaltens zur effizienten Lösung von Optimierungsproblemen.

lungsreisendenproblem [Rei94, LLKS85] (TSP, engl. Traveling Salesman Problem) können nur für relativ kleine Probleme mit bis zu ca. 50 Städten gute Ergebnisse erzielt werden. Das ist einer der Hauptgründe, weshalb in den letzten Jahren etliche Erweiterungen dieses Algorithmus vorgestellt wurden. Unter diesen Erweiterungen sind das *Ant Colony System* (ACS) [GD96, DG97], das *$\mathcal{MAX}$–$\mathcal{MIN}$ Ant System* ($\mathcal{MM}$AS) [SH96, SH97] und die *rank-based version of Ant System* [BHS97]. Die meisten dieser Erweiterungen unterscheiden sich dabei in wesentlichen Aspekten der Suchkontrolle, insbesondere in der Kombination von Aspekten der Intensivierung und der Diversifikation der Suche. Zusätzlich kann die Leistungsfähigkeit dieser Algorithmen noch signifikant erhöht werden, indem die von den Ameisen konstruierten Lösungen mit Hilfe lokaler Suchverfahren (nach-)optimiert werden [MDC94, SH96, SH97, DG97]. Deshalb sind die erfolgreichsten ACO-Algorithmen *hybride Algorithmen*, die eine Phase der Lösungskonstruktion mit einer nachfolgenden lokalen Suche verbinden.

Ameisenalgorithmen können als adaptive Algorithmen aufgefasst werden; dabei beeinflußt die Erfahrung aus früheren Iterationen das zukünftige Verhalten des Algorithmus. Die Pheromonspuren sollten dabei so manipuliert werden, daß Lösungsattribute, die zu besseren Lösungen führen, eine höhere Pheromonstärke erhalten. Dann werden solche Lösungsattribute auch von den Ameisen mit höherer Wahrscheinlichkeit benutzt. Dabei liegt die Idee zugrunde, daß durch eine geschickte Kombination bewährter Lösungsattribute eventuell noch bessere Lösungen gefunden werden können.

Der Rest des Artikels ist folgendermaßen aufgebaut. Im nächsten Abschnitt demonstrieren wir die von uns vorgeschlagene Erweiterung des Ant System, das $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System, am Beispiel der Anwendung auf das TSP. Anschließend, in Abschnitt 3, erläutern wir die Anwendung auf das quadratische Zuordnungsproblem (QAP, engl. Quadratic Assignment Problem). Die experimentellen Ergebnisse werden in Abschnitt 4 präsentiert und wir schließen den Artikel mit einer kurzen Zusammenfassung.

1.1 Das Handlungsreisendenproblem

Das Handlungsreisendenproblem [LLKS85, Rei94] ist wohl eines der bekanntesten kombinatorischen Optimierungsprobleme. Das Ziel ist es eine kürzeste Rundreise durch eine gegebene Anzahl von n Städten zu finden. Formal kann das TSP durch einen vollständigen gewichteten Graphen $G = (\mathcal{V}, \mathcal{A}, d)$ mit der Knotenmenge $\mathcal{V}$, der Kantenmenge $\mathcal{A}$ und einer Gewichtsfunktion $d : \mathcal{A} \mapsto \mathbb{N}$ beschrieben werden. d weisst jeder gerichteten Kante $\langle i, j\rangle$ ein ganzzahliges Gewicht $d(i, j)$ zu, das der Entfernung zwischen Städten i und j entspricht. Für symmetrische TSPs gilt, daß für alle Kanten $d(i, j) = d(j, i)$ ist, für das allgemeinere asymmetrische TSP gilt für mindestens ein Knotenpaar $d(i, j) \neq d(j, i)$. In Abbildung 1 ist als Beispiel eine euklidische TSP Instanz abgebildet.

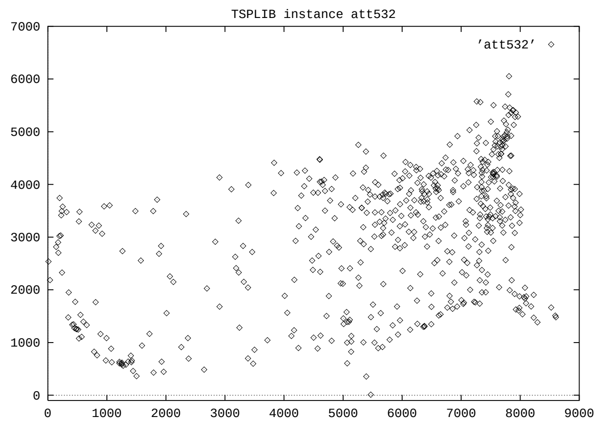

Abbildung 1: TSP Instanz `att532`, bestehend aus 532 Städten in den USA, aus der Problemsammlung TSPLIB. Diese Instanz ist sehr bekannt, da sie in vielen früheren Untersuchungen verwendet wurde. Die optimale Lösung dieser Instanz ist 27686.

1.2 Ant System

Ant System [DMC96] ist die erste Implementierung eines Algorithmus, der auf den Ideen der Optimierung mit Ameisenkolonien beruht. Wie viele andere naturanaloge Verfahren wurde der Algorithmus zuerst zur Lösung des TSP verwendet. Ant System ist ein populationsorientierter Ansatz, in dem eine Anzahl von m Ameisen verwendet wird. Zur Lösung des TSP sind mit jeder Kante $\langle i, j\rangle$ positive, reellwertige Variablen τ_{ij} verbunden, die die Stärke der Pheromonspuren angeben. Zusätzlich zu den Pheromonspuren verwenden die Ameisen eine heuristische Information, die im Fall des TSP eine Funktion der Entfernung zwischen zwei Städten ist. Der Algorithmus besteht aus zwei Phasen: Der Lösungskonstruktion und der Aktualisierung der Pheromonspuren. Diese beiden Phasen (wir bezeichnen sie als eine Iteration des Algorithmus) werden wiederholt, bis eine Abbruchbedingung, z.B. eine maximale Anzahl an Iterationen, erfüllt ist.

1.2.1 Konstruktion der Lösungen

Zur Konstruktion der Lösungen wird jede Ameisen auf eine zufällig ausgewählte Stadt gesetzt. Danach wählt jede Ameise von der Stadt, in der sie sich gerade befindet, eine der noch nicht besuchten Städte aus. In Ant System geschieht diese Auswahl gemäß der folgenden Wahrscheinlichkeitsverteilung:

$$p_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{k \text{ nicht besucht}} \tau_{ik}^{\alpha} \cdot \eta_{ik}^{\beta}} & \textit{falls } j \text{ nicht besucht} \\ 0 & \textit{sonst} \end{cases} \tag{1}$$

Dabei ist $\eta_{ij} = 1/d(i,j)$ die heuristische Information. η_{ij} ist umso größer, je kleiner die Distanz zwischen Städten i und j ist. 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. Diese bestimmen den relativen Einfluß der Pheromonstärke und der heuristischen Information.

1.2.2 Aktualisierung der Pheromonspuren

Nachdem jede Ameise eine vollständige Lösung konstruiert und die Länge ihrer Rundreise L_k berechnet hat, werden die Pheromonspuren aktualisiert. In Ant System dürfen alle Ameisen eine konstante Menge Q an Pheromon auf die von ihnen benutzten Kanten legen. Die neue Pheromonstärke τ_{ij}^{new} einer Kante wird dann berechnet zu:

$$\tau_{ij}^{new} = \rho \cdot \tau_{ij}^{old} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{2}$$

Dabei ist ρ ist die Persistenz, $1 - \rho$ gibt die Evaporation der Pheromonstärke an. Je kleiner ρ ist, desto schneller wird die Information der vorangegangenen Iterationen vergessen. Jede Ameisen legt Pheromon der Stärke $\Delta\tau_{ij} = Q/L_k$ auf jede Kante die sie besucht hat. Dies bedeutet, daß oft besuchte Kanten in kurzen Touren mehr Pheromon erhalten und deshalb auch öfter in den folgenden Iterationen des Algorithmus gewählt werden. Der Algorithmus zeichnet sich also durch eine *positive Rückkopplung* aus.

1.3 $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System

Mit Ant System 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. Dies führte in den letzten Jahren zu verschiedenen Erweiterungen des ursprünglichen Algorithmus. Eine dieser Erweiterungen ist $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System, das direkt auf der grundlegenden Arbeit über Ant System aufbaut. $\mathcal{MM}$AS unterscheidet sich in drei Gesichtspunkten von Ant System. Ein wichtiger Aspekt ist, daß, ähnlich wie in ACS, für die Aktualisierung der Pheromonspuren gemaäß Gleichung 2 nur noch eine Ameise verwendet wird. Dies kann die Ameise sein, die die bisher beste Rundreise gefunden hat (*global-beste*) oder die Ameise mit der besten Lösung der aktuellen Iteration des Algorithmus (*iterations-beste*). Dadurch wird erreicht, daß sich der Algorithmus schneller auf vielversprechende Regionen des Suchraums konzentriert. In diesem Fall kann es aber dazu kommen, daß sich die Trails auf wenigen Kanten sehr hoch sind aber andererseits auf den meisten anderen Kanten sehr schnell abfallen. In solch einem Fall werden bei der Lösungskostruktion fast nur noch Kanten berücksichtigt, die eine sehr hohe Pheromonstärke aufweisen und es kommt zur Stagnation der Suche. Zu diesem Zweck benutzen wir zusätzliche Einschränkungen für die möglichen Werte der Pheromonstärke, indem wir die erlaubten Pheromonstärken explizit auf ein Intervall $[\tau_{\min}, \tau_{\max}]$ einschränken. Die wichtigere der beiden Schranken ist die untere Grenze für die Pheromonstärke. Man kann zeigen, daß der theoretisch maximal mögliche Wert $\tau_{\max} = Q/((1-\rho)L_{global\text{-}beste})$ ist, diesen wählen wir für die obere Schranke. $\tau_{\min}$ wählen wir um einen konstanten Faktor kleiner als $\tau_{\max}$. Dadurch wird erreicht, daß jede Kante eine feste minimale Wahrscheinlichkeit hat, für die Rundreise ausgewählt zu werden. Es sei noch bemerkt, daß man, ähnlich wie für Simulated Annealing, durch die Grenzen für die Pheromonstärken auch garantieren kann, daß im Limit die optimale Lösung gefunden werden kann.

Ein weiteres wichtiges Merkmal von $\mathcal{MM}$AS ist, daß grundsätzlich lokale Suchverfahren verwendet werden, 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 Regionen des Suchraums zu identifizieren und gute Anfangslösungen für die lokale Suche zu generieren. In der folgenden Abbildung 2 geben wir ein algortihmisches Schema für $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System an.

procedure $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System
  (1) Initialisiere Pheromonspuren und Parameter
  (2) **while** (Abbruchbedingung nicht erfüllt) **do**
    for $k := 1$ to m **do**
      konstruiere Lösung für Ameise k
    end for
    Verbessere Lösungen mit lokaler Suche
    Aktualisierung der Pheromone, $\forall \tau_{ij}\ \tau_{\max} \geq \tau_{ij} \geq \tau_{\min}$
  (3) **return** beste gefundene Lösung
end $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System

Abbildung 2: Allgemeiner algorithmischer Rahmen für $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System.

2 Anwendung auf das TSP

In diesem Abschnitt gehen wir auf einige Details der Implementierung von $\mathcal{MM}$AS und der lokalen Suchverfahren für das TSP ein. Für eine Beschleunigung der Lösungskonstruktion verwenden wir Kandidatenmengen [Rei94], die für jede Stadt ihre 20 nächsten Nachbarn enthalten. Falls möglich, wählt dann eine Ameise die nächste Stadt immer nur unter denen der Kandidatenmenge aus. Ist dies nicht möglich, wird die Stadt mit maximalem Wert für $\tau_{ij} \cdot \eta_{ij}$ gewählt. Um die Aktualisierung der Pheromonspuren schneller durchzuführen, es sind insgesamt n^2 Kanten und somit auch n^2 Pheromonstärken, werden mittels der Evaporation nur die Pheromonstärken der Kanten innerhalb der Kandidatenmengen aktualisiert. Als lokales Suchverfahren benutzen wir 3-opt, das auf der schrittweisen Verbesserung der aktuellen Lösung durch wiederholten Austausch von maximal drei Kanten basiert. Um die Implemetierung des lokalen Suchverfahren zu beschleunigen, benutzen wir Standardtechniken, die zum Beispiel in [Ben92] beschrieben sind. Von diesen Techniken benutzen wir *don't look bits* und eine Nachbarschaftssuche mit begrenztem Radius innerhalb der 40 nächsten Nachbarn.[3]

3 Anwendung auf das QAP

Beim quadratischen Zuordnungsproblem müssen n Objekte genau n Positionen zugeordnet 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:

[3] In [SH97, SH98] wurde eine 2-opt lokale Suche benutzt, die schlechtere Ergebnisse als die 3-opt lokale Suche liefert. Deshalb sind die hier präsentierten Ergebnisse auch deutlich besser.

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

Dabei sind $A = (a_{ij})$ und $B = (b_{rs})$ zwei $n \times n$ Matrizen, $\Phi(n)$ ist die Menge aller Permutationen (Zuordnungen) von $\{1, \ldots, n\}$ und $\phi(i)$ gibt die Position von Objekt i in der Lösung $\phi \in \Phi(n)$ an.

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 keine heuristische Information berücksichtigt) folgt der folgenden Wahrscheinlichkeitsverteilung:

$$p_{ij} = \begin{cases} \frac{\tau_{ij}}{\sum_{k\ noch\ frei} \tau_{ik}} & \textit{falls}\ \text{Position}\ j\ \text{frei} \\ 0 & \textit{sonst} \end{cases} \quad (3)$$

Als lokale Suche verwenden wir eine kurze Läufe des Robust Tabu Search (Ro-TS) Algorithmus [Tai91] für das QAP. Eine erste Anwendung des Ant System auf das QAP ist in [MDC94] beschrieben. Auch in dieser ersten Anwendung wird eine auf 2-opt bzw. Simulated Annealing basierende lokale Suche verwendet. Jedoch werden in diesem Artikel nur sehr kleine Instanzen berücksichtigt, für die außerdem nur das jeweils beste erzielte Ergebnis angegeben sind.

4 Experimentelle Ergebnisse

In diesem Abschnitt vergleichen wir die Ergebnisse, die mit $\mathcal{MM}$AS für das TSP und das QAP erzielt mit den Resultaten für einige der besten bekannten Algorithmen. Für die beiden Algorithmen wurde zur zusätzlichen Erhöhung der Exploration des Suchraums bei einer eventuell auftretenden Stagnation eine Reinitialisierung der Pheromonstärken vorgenommen.

4.1 Experimentelle Ergebnsse für das TSP

Für das TSP vergleichen wir die Ergebnisse mit $\mathcal{MM}$AS mir denen eines iterierten lokalen Suchverfahren (ILS) [MOF91]. Verwendet man für die lokale Suche das Suchverfahren von Lin-Kernighan, ist ILS der beste exisitierende Approximationsalgorithmus für das symmetrische TSP. Die Lin-Kernighan Heuristik ist das beste bekannte lokale Suchverfahren für TSPs, seine Lösungsgüte liegt im Bereich von 1-2% über dem Optimum für geometrische Instanzen wohingegen 3-opt nur Lösungsgüten von 3-6% erreicht. Jedoch ist die Lin-Kernighan Heuristik sehr implementierungsaufwendig im Vergleich zu 3-opt, weswegen wir für das ILS Verfahren die gleiche lokale Suche wie für $\mathcal{MM}$AS verwenden. In Tabelle 1 geben wir die Ergebnisse von $\mathcal{MM}$AS und ILS für einige TSP-Instanzen mit bekanntem Optimum aus der Problemsammlung TSPLIB an. Für $\mathcal{MM}$AS wurden 25 Ameisen verwendet, die alle nach jeder Iteration eine lokale Suche anwenden. Die Parameterwerte sind $\alpha = 1, \beta = 2$ und $\tau_{\min} = \tau_{\max}/2n$. Die Laufzeiten beziehen sich auf einen Prozesser (167MHz) einer Sun UltraSparc II Workstation. Die Ergebnisse für $\mathcal{MM}$AS sind die besten, die mit ACO-Algorithmen für das TSP erzielt wurden. Man sieht auch, daß die Lösungsgüte für die meisten Instanzen leicht besser als die des ILS Verfahren ist, was als sehr erfolgversprechend angesehen werden kann.

Tabelle 1: Ergebnisse für TSPLIB Instanzen. Die bekannte optimale Lösung ist angegeben, die Zahl im Instanznamen ist die Anzahl der Städte. Angegeben ist die mittlere Lösungsgüte für jeweils 25 Versuche auf jeder Instanz und deren Standardabweichung. In der letzten Spalte ist die maximal erlaubte Laufzeit für $\mathcal{MM}$AS und ILS angegeben; zusätzlich geben wir die mittlere Anzahl der Iterationen für $\mathcal{MM}$AS an, nach der die beste Lösung gefunden wurde.

Instanz	ILS	$\mathcal{MM}$AS	time / iterat
d198 15780	15780.0	15780.0 0.37	600 181.5
lin318 42029	42067.7 57.7	**42029.0** 0.0	600 139.1
pcb442 50778	**50879.0** 75.2	50893.2 50.9	900 753.4
att532 27686	27705.5 12.3	27706.0 13.7	1800 429.9
rat782 8806	8824.3 13.9	**8812.2** 4.9	3600.0 1222.6
pcb1173 56892	56991.7 91.3	**56956.0** 73.2	5400.0 1669.2
d1291 50801	50868.0 33.3	**50821.6** 12.8	5400.0 1035.0

4.2 Experimentelle Ergebnisse für das QAP

Für die Anwendung des $\mathcal{MM}$AS auf das QAP vergleichen die Ergebnisse zu Tabu Search Algorithmen für das QAP. Bei dem Vergleich unterscheiden wir zwischen zufällig erzeugten, unstrukturierten Instanzen und 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 Suche Algorithmen werden jeweils $1000n$ Iterationen für die lokale Suche Iterationen, für $\mathcal{MM}$AS werden 250 kurze Tabu Suchen der Länge $4n$ erlaubt. Da die lokale Suche selbst sehr laufzeitintensiv ist, ist die CPU-Zeit für alle Algorithmen fast gleich. Aufgrund der geringen Anzahl der lokalen Suchen verwenden wir hier nur 5 Ameisen für $\mathcal{MM}$AS. An den Ergebnissen in Tabelle 2 sieht man, 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 ist. 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, siehe [Stü97] für eine ausführlichere Diskussion dieser Ergebnisse.

Tabelle 2: Ergebnisse für Instanzen aus QAPLIB, die Zahl im instanzbezeichner gibt die Anzahl der Objekte an. Als Ergebnis ist die mittlere Lösungsgüte für jeweils 10 Versuche gegeben.

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

5 Zusammenfassung

In diesem Artikel haben wir Anwendungen des $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System, einer Erweiterung von Ant System für das Handlungsreisendenproblem und das quadratische Zuordnungsproblem vorgestellt. Die Versuchsergebnisse zeigen, daß für beide Probleme sehr gute Ergebnisse, vergleichbar mit den besten Algorithmen für diese Probleme, erzielt werden können. $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System ist deshalb ein vielversprechender allgemeiner ACO-Algorithmus der erfolgreich zur Lösung kombinatorischer Optimierungsprobleme eingesetzt werden kann.

Literatur

[Bäc96] T. Bäck. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, New York, 1996.

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

[BHS97] B. Bullnheimer, R.F. Hartl, and C. Strauss. A New Rank Based Version of the Ant System — A Computational Study. Technical report, University of Viena, Institute of Management Science, 1997.

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

[DG97] M. Dorigo and 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.

[DMC96] M. Dorigo, V. Maniezzo, and 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.

[GD96] L.M. Gambardella and M. Dorigo. Solving Symmetric and Asymmetric TSPs by Ant Colonies. In *Proc. of ICEC'96*, pages 622–627, 1996.

[HT85] J.J. Hopfield and D. Tank. Neural Computations of Decisions in Optimization Problems. *Biological Cybernetics*, 52:141–152, 1985.

[KJV83] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

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

[MDC94] V. Maniezzo, M. Dorigo, and A. Colorni. The Ant System Applied to the Quadratic Assignment Problem. Technical Report IRIDIA/94-28, Universit Libre de Bruxelles, Belgium, 1994.

[MOF91] O. Martin, S.W. Otto, and 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*, volume 840 of *Lecture Notes in Computer Science.* Springer Verlag, 1994.

[SH96] T. Stützle and H.H. Hoos. Improving the Ant-System: A Detailed Report on the $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System. Technical Report AIDA–96–12, FG Intellektik, TH Darmstadt, August 1996.

[SH97] T. Stützle and H.H. Hoos. The $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System and Local Search for the Traveling Salesman Problem. In *Proc. of ICEC'97*, pages 309–314, 1997.

[SH98] T. Stützle and H.H. Hoos. $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System and Local Search for Combinatorial Optimization Problems. In *Proc. of the Metaheuristics Conference*, 1998. to appear.

[Stü97] T. Stützle. $\mathcal{MAX}$–$\mathcal{MIN}$ Ant System for the Quadratic Assignment Problem. Technical Report AIDA–97–4, FG Intellektik, TU Darmstadt, July 1997.

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

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