Infos zu Ameisensystemen & TSP

Inhalt

[Ameisensysteme 1](#_Toc305746042)

[TSP 6](#_Toc305746043)

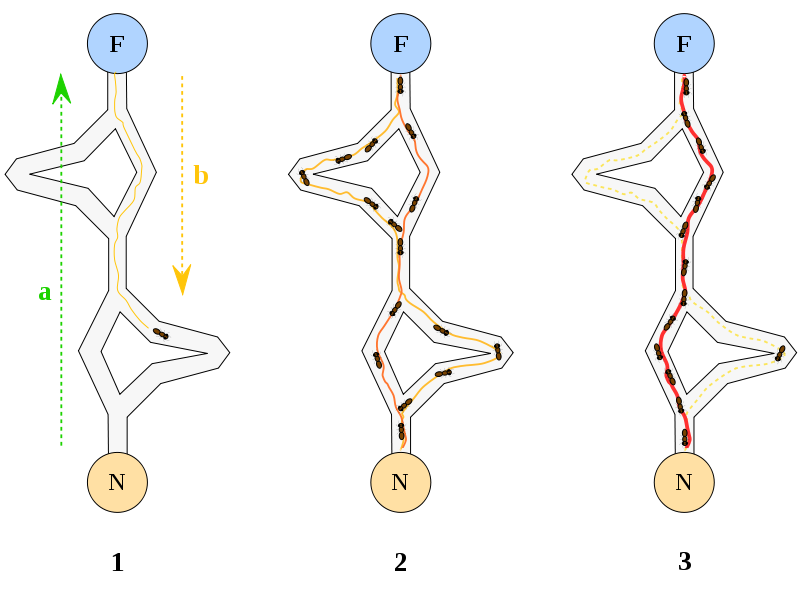
[Quellen 9](#_Toc305746044)

# Ameisensysteme

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

**Reale Ameisen**

* basieren auf dem modellhaften Verhalten von realen Ameisen bei der Futtersuche



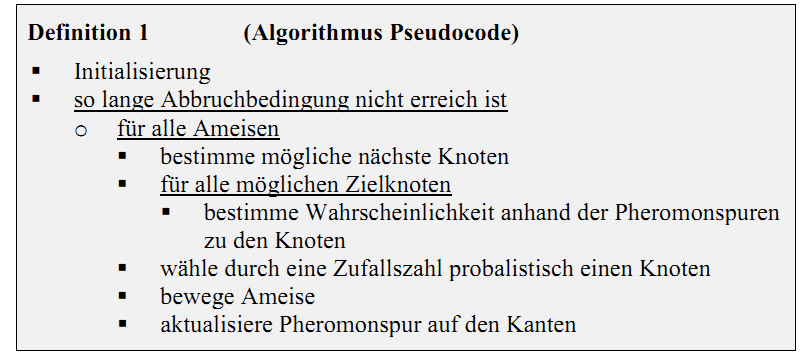
1. die erste Ameise findet eine Futterquelle (F), benutzt den Weg (a), dann erreicht sie das Nest (N), und hinterlässt eine Pheromonspur
2. andere Ameisen folgen der ersten auf 4 möglichen Pfaden
3. Die Ameisen folgen dem kürzesten Pfad

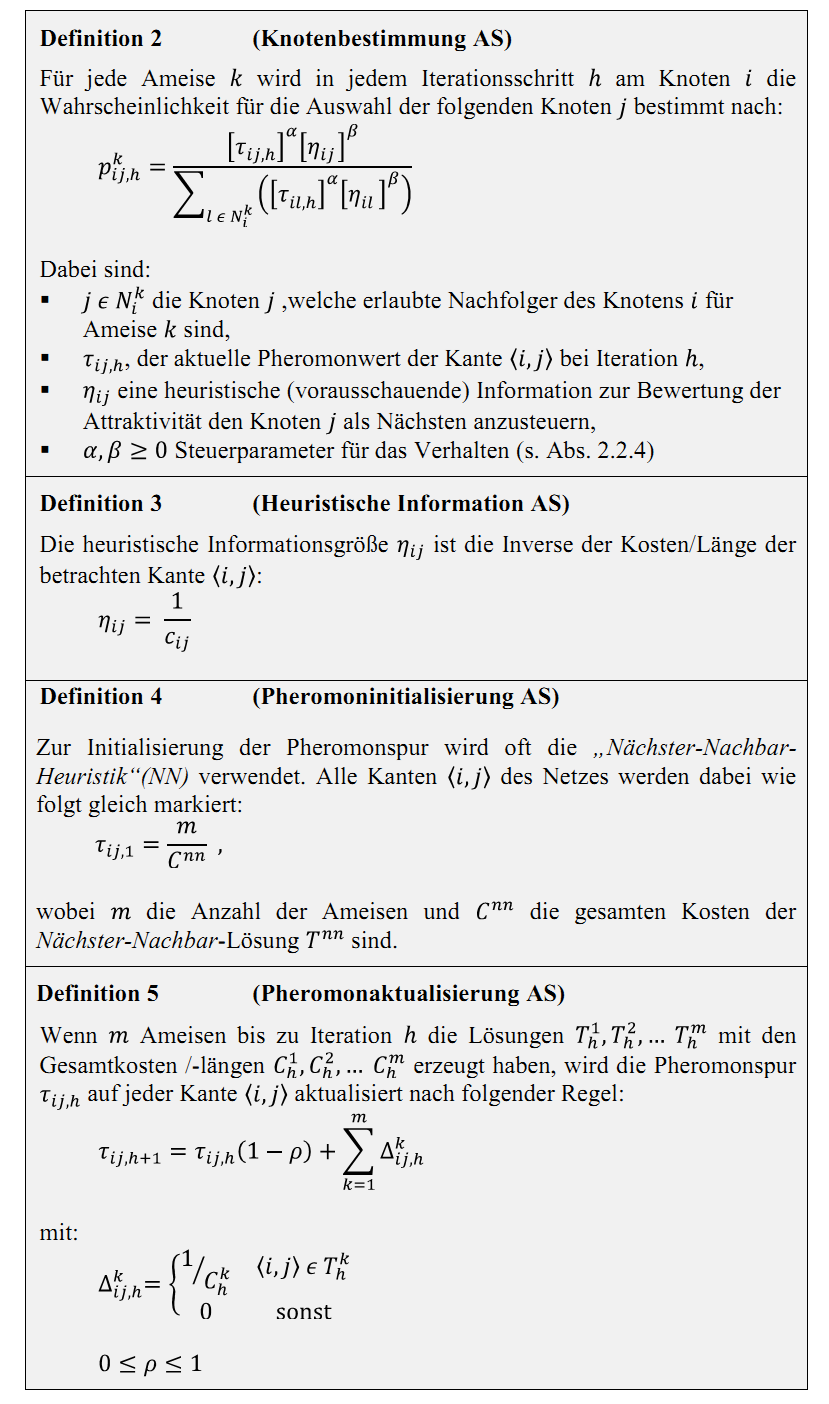
* Ant Colony Optimization (ACO) = Inspiration vom Verhalten von Ameisen bei der Futtersuche
* sobald eine Ameise eine Futterquelle gefunden hat, markiert sie den Weg zwischen ihrem Nest und der Futterquelle in regelmäßigen Abständen mit einer Substanz (Pheromon)
* anderen Ameisen orientieren sich am Duft-Pfad
* Orientierung an der Intensität der Duftspur
* Kommunikation zwischen mehreren relativ einfachen Agenten
  + Versuch: Nachahmung im ACO-System
* nicht alle Ameisen nehmen die kurzen Wege
* mit einer gewissen Wahrscheinlichkeit nehmen einige Ameisen weiterhin die langen Wege 🡪 Suchverhalten
* nicht garantiert, dass die Pheromonspur bereits auf dem global kürzesten Weg konvergiert ist
* nach längerer Zeit: Ameisenkolonie verlässt eine einmal gefundene Lösung (lokales Optimum) nur schwer, trotz neuer Lösung
* Vorteil:
  + Schwarmintelligenz = „Einfachheit“ der Individuen
    - Vereinfachung der Simulation des Verhaltens im Computer
    - bei gleichzeitig hoher Leistungsfähigkeit des Gesamtsystems

**Künstliche Ameisen**

* Unterschied zur Realität:
  + algorithmische Implementierung mit diskreten Zeitschritten einfacher
  + Vermeidung, dass die künstlichen Ameisen in Schleifen gefangen bleiben
  + Erhalten ein Gedächtnis für ihren bisherigen Pfad und dessen Länge
    - Individuelle Wegentscheidungen = probalistisch
    - Schleifen vermieden werden
    - bei der Pheromonaktualisierung rückwirkend die Qualität (Länge) der bisherigen Teillösung mit einfließen kann
  + durch die Simulation von Verdunstung der Pheromone:
    - Verbessern der Leistung des Algorithmus beim Erkunden des Lösungsraums bzw. beim Verlassen lokaler Optima

**Darstellung des Grundprinzips: Ameisensystem (AS)**





**Einfluss der Parameter**

* durch die verschiedenen Parameter:
  + Verhalten des Algorithmus lässt sich vielfältig steuern
* Wichtig:
  + Wahl der „richtigen“ Parameter, um gute Lösungen effizient zu erzeugen

1. Möglichkeit:

Parameterbestimmung bspw. Ameisenalgorithmus auf Anfangsparameter anzuwenden und Durchführung mehrerer Läufe

1. Möglichkeit:

Experimente und analytische Korrelations- (Ermittlung der Strenge des Zusammenhanges zwischen den einzelnen Variablen) und Regressionsanalysen (Feststellung von Beziehungen zwischen einer abhängigen und einer oder mehreren unabhängigen Variablen) zwischen Parametern, Problemstruktur und Lösung

* Ergebnisse:
  + wenn Rechenzeit vorgegeben ist:
    - die Größe der Ameisenpopulation hat Auswirkungen auf das Verhalten
      * Je höher die Zahl der Ameisen, desto tiefer wird das Netzwerk erkundet

🡪 weniger Lösungen (Iterationen) werden erzeugt

* + - * Je kleiner die Zahl der Ameisen, umso mehr Iterationen können ausgeführt werden
      * das bereits erworbene Wissen über das Netz ausgeschöpft/stabilisiert werden

🡪 auf Kosten der Erkundung des Lösungsraums

* + wird Verdunstung der Pheromonspur simuliert:
    - Steuerung des Erkundungsverhalten
    - durch schnelle Verdunstung verliert eine einmal markierte Lösung innerhalb weniger Iterationen ihre Bevorzugung

🡪 Finden neuer Lösungen mit höherer Wahrscheinlichkeit

* durch eine geringe Verdunstungsrate

🡪 schnelle Konvergenz der Lösung

* Exponenten α & β betonen Unterschiede zwischen den Kanten bei der Bestimmung der Wahrscheinlichkeiten
  + α > β
    - höhere Bedeutung der durch Iterationen erzeugten Lösungen und dynamischen Bewertungen der Kanten
  + α < β:
    - statische heuristische Information erhält über die Kantenlängen höheren Einfluss

**Vor- und Nachteil des ACO**

* Vorteil:
  + einfache Struktur, keine Definierung von speziellen Nachbarschaftsstrukturen
  + keine Beschreibung von angepassten Vererbungsvorgängen
  + vielfältige Anwendung unter gleich bleibender Charakteristik
* Nachteil:
  + vorzeitige Konvergenz in lokale Optima

**Anwendung des Algorithmus in der Realität**

* durch einfache und allgemeingültige Prinzipien können ACO-Algorithmen in vielfältigen Bereichen angewandt werden
* Stärke = graphenbasierte Probleme
* Einteilung in Kategorien:
  + Routenplanung (routing)
    - „Agent(en)“ muss/müssen eine Anzahl definierter Orte ansteuern
    - Zielfunktion wird durch die Reihenfolge der aufgesuchten Orte bestimmt
    - Bsp.: Busrouten, Müllabfuhr, Post- und Auslieferungsrouten
  + Zuordnung (assignment)
    - ein Satz von Objekten muss einem Satz von Ressourcen unter Nebenbedingungen zugeordnet werden
    - Zielfunktion wird minimiert
  + Ablaufplanung (scheduling)
    - Zuordnung von knappen Ressourcen zu Tätigkeiten im Zeitablauf
  + Teilmengen (subset)
    - eine Teilmenge von Komponenten muss aus der Menge aller Komponenten ausgewählt werden (unter Einhaltung von Nebenbedingungen)
      * Optimierung der Zielfunktion
    - Reihenfolge hat keine Einfluss auf die Lösungsqualität
    - Komponenten beinhalten die Pheromonspuren
    - nicht alle Lösungen müssen die selbe Größe (gleiche Menge an Komponenten) haben

# TSP

**Was ist das Handelsreisendenproblem?**

* Travelling Salesman Problem (TSP)
* Handelsreisender möchte eine vorgegebene Menge von Städten nacheinander besuchen
* Am Ende = Rückkehr zum Ausgangsort
* Reihenfolge, in der er die Städte besucht, kann er selbst festlegen
* Reise nennt man Tour durch alle Städte
* hat einen Plan zur Verfügung, der ihm alle Entfernungen (z.B. in Kilometer) zwischen jeweils zwei Städten anzeigt
* Jede Tour liefert Gesamtlänge, die der Handelsreisende für diese Tour zurücklegen muss
* Er kann zwei Ziele verfolgen:
  + Das höchste Ziel:
    - Tour mit minimaler Gesamtlänge
  + Falls ihm dieses Ziel zu ehrgeizig ist:
    - Tour mit möglichst kleiner Gesamtlänge



* Optimaler Reiseweg eines Handlungsreisenden durch die 15 größten Städte Deutschlands. Die angegebene Route ist die kürzeste von 43.589.145.600 möglichen.

**Bedeutung des Handelsreisendenproblems**

* eines der bekanntesten Probleme der Theoretischen und Praktischen Informatik
* Gründe:
  + Einfach zu verstehen
  + Schwer zu lösen
  + Man kann praktisch testen, was Computerprogramme für bestimmte Beispiele leisten können:
    - In welcher Laufzeit finden sie Touren mit welcher Qualität?

**Anwendung en in der Praxis**

* Netzfahrplan
  + Verkehrsmittel fährt vorgegebene Haltestellen an
* Lagerlogistik
  + Einzelposten einer Bestellung werden im Lager zusammengesucht
* Tourenplanung
  + Einsatzfahrzeug fährt Depots ab
* Design von Mikrochips
  + Städte sind Bohrlöcher
* Genom-Sequenzierung
  + Städte sind DNA-Teilstränge

**Wie ﬁndet man eine minimale Tour?**

* Naive Methode
  + Probiere alle möglichen Touren durch
  + Auswahl der Tour mit der kleinsten Länge
  + Wie viele Touren hat man durchzuprobieren?
    - Beispiel mit 10 Städten
      * Am Startort hat er 9 Möglichkeiten
      * Am ersten Durchgangsort hat er nur noch 8 Möglichkeiten usw.
      * Am 9-ten Durchgangsort hat er nur noch eine Möglichkeit
      * Insgesamt:
      * 9 \* 8 \* 7 \* 6 \* 5 \* 4 \* 3 \* 2 \* 1 = 9! = 362.880 Touren durchzuprobieren
      * für beliebige Anzahl n von Städten = (n-1)! Touren durchprobieren
      * Bsp.:
        + n = 5: 24 mögliche Touren
        + n = 30: 8, 84 \* 1030 mögliche Touren
        + n = 70: 1, 71 \* 1098 mögliche Touren
      * Vergleich: Im Universum gibt es etwa 1078 Atome
      * besten Hochleistungsrechner für 30 Städte = Millionen Jahre

**Erweiterung der naiven Methode**

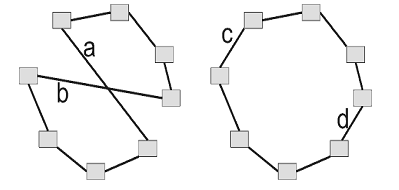
* Ziel:
  + Entferne Touren von der Suche, von denen man während der Suche schon weiß, dass sie nicht mehr minimal sein können
* Beispiel:
  + während der Suche wurde schon eine Tour mit Gesamtkosten 10 Einheiten gefunden
  + zwischen Stadt A und B sind die Kosten 11 Einheiten
  + Strecken AB & BA kommen für eine minimale Tour nicht mehr in Frage

**Wie findet man eine möglichst gute Tour?**

* Naive Methode
  + Gehe in jedem Schritt zur nächstgelegenen Stadt
  + wenn alle Städte besucht, gehe wieder zum Startort
  + Nachteil:
    - am Anfang sehr kurze Entfernungen
    - am Ende Auswahl an nicht besuchten Städten sehr klein
      * zurückgelegte Entfernungen können sehr groß sein
  + Bezeichnung der Methode: Greedy-Methode (”greedy” heißt ”gierig”)
  + Was benötigt man für die Greedy-Methode?
    - Liste, die zu jeder Stadt die nächstgelegene Stadt angibt
  + Laufzeit?
    - Beispiel: 10 Städte
    - für jede der 10 Städte hat er 9 Entfernungen zu vergleichen
    - Notieren der minimalen Entfernung
    - 9 \* 10 = 90 Schritte
    - für beliebige Anzahl n von Städten = n \* (n-1) = n²-n Schritte durchzuführen
    - Hochleistungsrechner in wenigen Tagen = 1 Million Städte

**Erweiterung der naiven Methode**

* Starte mit einer (schon recht guten) Tour, die man durch die naive Methode erhalten hat
* Ersetze (wenige) existente Städteverbindungen durch (genauso viele) nicht-existente Städteverbindungen
  + Tour liegt vor, die aber besser ist als die ursprüngliche Tour
  + Dies kann für zwei Städteverbindungen so aussehen:



* Laufzeit hängt sehr stark von der Anzahl der Städte ab

# Quellen

* <http://de.wikipedia.org/wiki/Metaheuristik>
* <http://de.wikipedia.org/wiki/Ameisensystem>
* <http://upload.wikimedia.org/wikipedia/commons/thumb/a/af/Aco_branches.svg/800px-Aco_branches.svg.png>
* <http://www.advanced-planning.de/advancedplanning-320.htm>
* <http://de.wikipedia.org/wiki/Ameisensystem>
* <http://www.schillrich.de/dateien/SA_OR_AntSystems.pdf>
* <http://de.wikipedia.org/wiki/Problem_des_Handlungsreisenden>
* <http://upload.wikimedia.org/wikipedia/commons/thumb/c/c4/TSP_Deutschland_3.png/220px-TSP_Deutschland_3.png>
* <http://www.informatik.uni-kiel.de/~gej/publ/halle2.pdf>