

Anwendungen von Ameisensystemen

Christopher Blöcker

Contents

1	Anwendungsbeispiele	3
1.1	Traveling Salesman Problem (<i>TSP</i>)	3
1.2	Job Shop Problem (Verallgemeinertes <i>TSP</i>)	3
1.3	Vertex Cover (<i>VC</i>)	4
1.4	Graph Coloring	4
1.5	Graph Tree Partitioning	4
1.6	Quadratic Assignment	5
1.7	Bin Packing	5
1.8	Protein Folding	6
1.9	Shortest Common Supersequence	6
2	Vertex Cover (<i>VC</i>)	7
2.1	Beispiel	7
2.2	Formale Definition des Problems	7
2.3	Problem	8
2.4	Lösung des Problems	9
2.4.1	Ableiten der Informationen	9
2.4.2	Zielfunktion und Nebenbedingung	10
2.4.3	Modifikation des Graphen	11
2.4.4	Konstruktion der Lösung	12
2.4.5	Ablauf des Verfahrens	15
3	Vergleich Algorithmik und KI	16
3.1	Algorithmik	16
3.2	KI (Ameisensysteme)	16
3.3	Vergleich anhand eines konkreten Beispiels	17
3.4	Beispiel	17

1 Anwendungsbeispiele

Ameisensysteme können zur Lösung vieler Probleme verwendet werden. Vielen der Probleme, für die Ameisensysteme verwendet werden, ist gemeinsam, dass sie algorithmisch nur schwer lösbar sind. Eine Anwendung auf effizient lösbare Probleme ist dennoch denkbar.

Die Beschreibung einiger Anwendungsfälle soll einen Überblick darüber verschaffen, für welche Art von Problemen Ameisensysteme zur Lösung verwendet werden können. Außerdem sollen die Beschreibungen dabei helfen, zu erkennen, wann ein Problem vorliegt, für das ein Ameisensystem zur Lösung in Betracht gezogen werden kann.

1.1 Traveling Salesman Problem (*TSP*)

Gegeben sei ein kantenbewerteter Graph $G = (V, E)$ mit

- V : Menge der Knoten von G , $|V| = n$
- $E \subseteq V \times V$: Menge der Kanten von G
- $\phi : V \times V \rightarrow \mathbb{N}$: Bewertungsfunktion der Kanten

Gesucht ist eine geordnete Folge $R = (v_{i_1}, v_{i_2}, \dots, v_{i_n})$ der n Knoten von G derart, dass aufeinanderfolgende Knoten adjazent sind und die Gesamtlänge $\Phi(R)$ minimal ist.

Beispiel: Ein Tourist möchte eine Reise durch die 10 größten Städte Deutschlands unternehmen und sucht dabei die kürzeste Rundreise.

1.2 Job Shop Problem (Verallgemeinertes *TSP*)

Gegeben seien n Jobs und m Maschinen.

Gesucht ist eine Zuordnung der Jobs auf die Maschinen derart, dass jeder Job auf jeder Maschine bearbeitet wird und die entstehenden Durchlaufkosten minimal sind.

Beispiel: Die Bauteile für ein Auto müssen lackiert, gebohrt und zurechtgeschnitten werden. Es soll nun geplant werden, welches Bauteil wann auf welcher Maschine bearbeitet wird.

1.3 Vertex Cover (VC)

Gegeben sei ein ungerichteter, knotenbewerteter Graph $G = (V, E)$ mit

- V : Menge der Knoten von G
- $E \subseteq V \times V$: Menge der Kanten von G
- $\phi : V \rightarrow \mathbb{N}$: Bewertungsfunktion der Knoten

Gesucht ist Eine Teilmenge $S \subseteq V$ mit

$$\bigwedge_{\{v_i, v_j\} \in E} v_i \in S \vee v_j \in S$$

wobei $\Phi(S)$ minimal sein soll.

Beispiel: In einem Museum sollen Überwachungskameras installiert werden. Mit einem Vertex Cover kann bestimmt werden, wo diese Kameras anzubringen sind, damit möglichst wenige benötigt werden.

1.4 Graph Coloring

Gegeben sei ein Graph $G = (V, E)$ mit

- V : Menge der Knoten von G
- $E \subseteq V \times V$: Menge der Kanten von G

Gesucht ist eine Färbung der Knoten derart, dass benachbarte Knoten nicht dieselbe Farbe erhalten und die Anzahl der benötigten Farben minimal ist.

1.5 Graph Tree Partitioning

Gegeben sei ein kantenbewerteter Graph $G = (V, E)$, eine untere Grenze w_{min} sowie eine obere Grenze w_{max} mit

- V : Menge der Knoten von G
- $E \subseteq V \times V$: Menge der Kanten von G
- $w_{min}, w_{max} \in \mathbb{R}^+, w_{min} < w_{max}$

Gesucht ist eine Unterteilung (Partitionierung) von G in einen Wald derart, dass das Gewicht jedes Baumes innerhalb der Grenzen $[w_{min}, w_{max}]$ liegt.

Beispiel: Beim Design eines Telekommunikationsnetzes soll bestimmt werden, welche Kunden durch welchen Zugangspunkt versorgt werden. Dabei dürfen es weder zu viele sein, da sonst die Kapazität überschritten wäre, noch zu wenige, da wirtschaftlich gearbeitet werden soll.

1.6 Quadratic Assignment

Gegeben sei eine Menge von n Items und eine Menge von n Positionen.

Gesucht ist eine Zuordnung der Items auf die Positionen derart, dass die Verbindungskosten zwischen den Positionen minimiert werden.

Beispiel: Ein Unternehmen hat Grundstücke gekauft und will darauf Fabriken errichten. Die Fabriken sollen so angeordnet werden, dass die Transportkosten für die Materialien minimal sind.

1.7 Bin Packing

Gegeben seien n gewichtete Gegenstände sowie eine unbegrenzte Anzahl von Behältern derselben Kapazität k mit

- $G = \{g_1, g_2, \dots, g_n\}$: Menge von n Gegenständen
- $\phi : G \rightarrow \mathbb{N}$: Bewertungsfunktion der Gegenstände
- Die Kapazität der Behälter $k \geq \max\{\phi(G)\}$

Gesucht ist eine Befüllung der Behälter mit den Gegenständen derart, dass die Anzahl der benötigten Behälter minimal ist und die Kapazität k bei keinem der Behälter überschritten wird.

Beispiel: Von einem Warenlager aus müssen Filialen einer Supermarktkette beliefert werden. Die Lastwagen sollen so beladen werden, dass möglichst wenige für die Lieferungen benötigt werden.

1.8 Protein Folding

Gegeben sei die Aminosäuren-Sequenz eines Proteins.

Gesucht ist die Struktur des daraus entstehenden Proteins, um vorhersagen zu können, welche Wirkung es auf chemische Prozesse hat.

1.9 Shortest Common Supersequence

Gegeben sei eine endliche Menge L von Wörtern über einem Alphabet Σ

$$L = \{w_1, w_2, \dots, w_n\}$$

Gesucht ist ein Wort $w^* \in \Sigma^*$ derart, dass gilt

$$\bigwedge_{w \in L} u \cdot w \cdot v = w^*$$

mit

$$u, v \in \Sigma^*$$

Jedes Wort $w \in L$ soll ein Teilwort des Wortes w^* sein. Außerdem soll $|w^*|$ minimal sein.

2 Vertex Cover (VC)

Das Vertex Cover Problem ist ein graphentheoretisches Problem von praktischer Relevanz.

Beim Verständnis der Funktionsweise von Ameisensystemen soll die detaillierte Beschreibung eines Ameisensystems zur Lösung des Vertex Cover Problems helfen.

Nach einem einfachen Beispiel folgt die formale Definition des Problems sowie die Beschreibung des Ameisensystems zur Lösung des Problems.

2.1 Beispiel

Nachdem in einer Bank ein Überfall stattgefunden hat, muss die Polizei die Räuber fassen, um das gestohlene Geld sicherzustellen.

Da die Räuber mit dem Auto auf der Flucht sind, muss dazu das Straßennetz der Stadt überwacht werden.

Statt in jeder Straße einen Polizeiwagen zu postieren, können aber auch die Kreuzungen überwacht werden.

Ziel ist es dabei, so wenige Kreuzungen wie möglich zu überwachen, um die Kosten der Fahndung gering zu halten und möglichst wenige Einsatzwagen zu benutzen.

Eine Knotenüberdeckung (bzw. ein Vertex Cover) gibt dabei an, welche Kreuzungen zu überwachen sind.

2.2 Formale Definition des Problems

Sei $G = (V, E)$ ein ungerichteter, knotenbewerteter Graph mit

- V : Menge der Knoten von G
- $E \subseteq V \times V$: Menge der Kanten von G
- $\phi : V \rightarrow \mathbb{N}$: Die Bewertungsfunktion der Knoten

Gesucht: Eine Teilmenge $S \subseteq V$ mit

$$\bigwedge_{\{v_i, v_j\} \in E} v_i \in S \vee v_j \in S$$

wobei $|S|$ minimal sein soll.

Eine Knotenüberdeckung ist eine Teilmenge der Knoten eines Graphen derart, dass wenigstens ein Endpunkt jeder Kante in der Überdeckung liegt.

Dabei ist eine Knotenüberdeckung besser, je kleiner sie ist bzw. je geringer die verursachten Kosten sind.

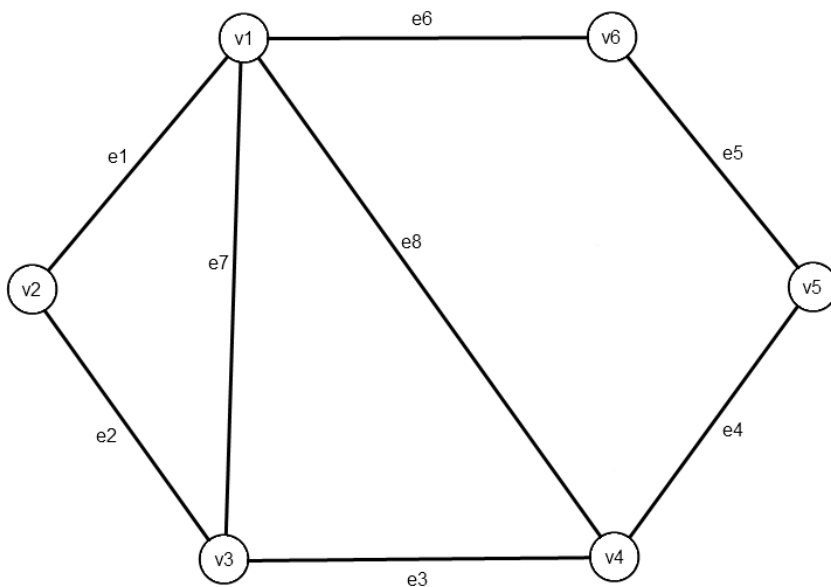
Dabei ist es im allgemeinen algorithmisch nur schwer entscheidbar, ob in einem gegebenen Graphen eine Knotenüberdeckung mit einer bestimmten Anzahl von Knoten existiert (triviale Überdeckungen existieren immer).

In einem Graphen ein möglichst kleines Vertex Cover zu bestimmen ist ein *NP*-Vollständiges Problem.

2.3 Problem

Gegeben sei der folgende Graph $G = (V, E)$ mit

- $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$
- $E = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_6\}, \{v_6, v_1\}, \{v_1, v_3\}, \{v_1, v_4\}\}$



Gesucht ist eine minimale Knotenüberdeckung von G .

2.4 Lösung des Problems

Um das Problem zu lösen, müssen zunächst Informationen aus G abgeleitet werden.

Danach werden mit Hilfe dieser Informationen Regeln definiert, nach denen die Ameisen Lösungskandidaten konstruieren.

Für das Verfahren werden dann Modifikationen am Problemgraphen durchgeführt, die aufgrund der Funktionsweise notwendig sind.

Zum Finden der Lösung laufen die Ameisen unabhängig voneinander und konstruieren jeweils einen Lösungskandidaten. Die Phasen, in denen dies geschieht werden als Zyklen bezeichnet. Nach jedem Zyklus werden anhand der konstruierten Lösungskandidaten Aktualisierungen durchgeführt, die die Daten verändern, anhand derer die Konstruktion der Lösungskandidaten stattfindet.

Das Verfahren wird abgebrochen wenn

- Die zu Beginn des Verfahrens festgelegte Maximalzahl der Zyklen erreicht ist
- Alle Ameisen nur noch denselben Lösungskandidaten konstruieren. In diesem Fall spricht man von Stagnation.

2.4.1 Ableiten der Informationen

1. Aus G wird eine $n \times m$ Inzidenz-Matrix der Form $[a_{ij}]$ abgeleitet, dabei soll gelten

$$a_{ij} = \begin{cases} 1 & \text{wenn Knoten } i \text{ ein Endpunkt von Kante } j \text{ ist} \\ 0 & \text{sonst} \end{cases}$$

Dabei entsteht folgende Matrix:

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

2. Es wird ein Vektor der Dimension n abgeleitet, der die Kosten der Knoten enthält mit

$$b_i = \phi(v_i)$$

Dabei entsteht folgender Vektor:

$$b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Interpretation des Vektors b :

Für das Überwachen jeder Kreuzung wird ein Polizeiwagen benötigt. Eine Bewertung mit einem anderen Wert ist möglich, z.B. wenn für eine besonders große Kreuzung zwei (oder mehr) Wagen benötigt werden.

3. Es wird ein Vektor y der Dimension n definiert mit

$$y_i = \begin{cases} 1 & \text{wenn } v_i \in S \\ 0 & \text{sonst} \end{cases}$$

Sei z.B. $S = \{v_1, v_2, v_3, v_5\}$

Dann ergibt sich daraus

$$y = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

2.4.2 Zielfunktion und Nebenbedingung

Für eine optimale Lösung müssen die Kosten c minimiert werden. Diese lassen sich wie folgt berechnen

$$c = \sum_{i=1}^n b_i \cdot y_i$$

Es soll erreicht werden $c \rightarrow \min$.

Außerdem muss gelten, dass wenigstens ein Endpunkt jeder Kante zum Vertex Cover gehört. Dies wird sichergestellt durch die folgende Bedingung

$$\bigwedge_{1 \leq i \leq n} \left(\sum_{j=1}^m a_{ij} \cdot y_j \right) \geq 1$$

2.4.3 Modifikation des Graphen

Der Graph wird so erweitert, dass er vollständig ist. Dazu werden zur Kantenmenge von G die Kanten des Komplementärgraphen \overline{G} hinzugefügt. $G' = (V', E')$ mit

- V' : Menge der Knoten von G
- $E' = E \cup \overline{E}$: Menge der Kanten von G

Dann werden zwei Dummy-Nodes v_s und v_z hinzugefügt wobei v_s derjenige Knoten sein soll, an dem die Ameisen ihren Lauf starten und v_z derjenige Knoten sein soll, an dem die Ameisen ihren Lauf beenden.

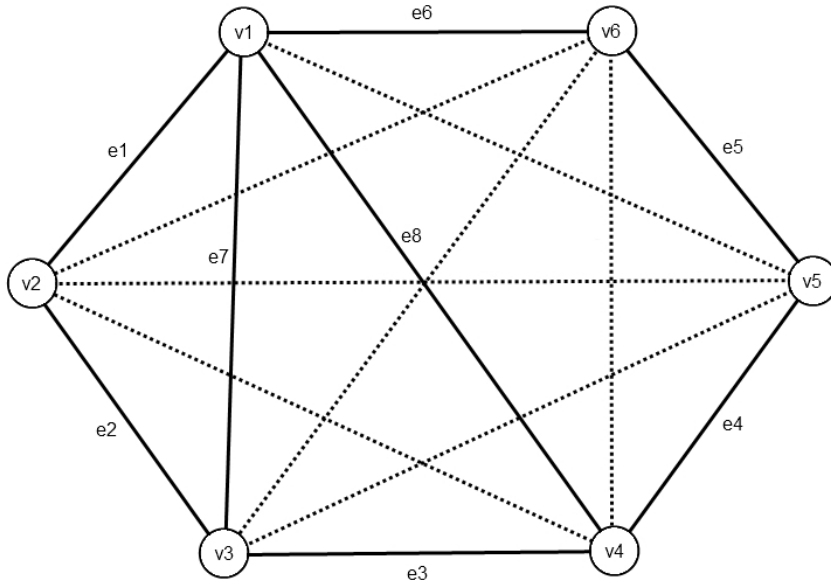
$$V' = V \cup \{v_s, v_z\}$$

v_s und v_z werden mit allen anderen Knoten verbunden.

$$\bigwedge_{v_i \in V} E' = E' \cup \{\{v_i, v_s\}, \{v_i, v_z\}\}$$

Die Erweiterung des Graphen, sodass er vollständig ist, ist notwendig, damit die Ameisen von jedem Knoten aus zu jedem anderen übergehen können. Beim Vertex Cover Problem ist es nicht notwendig, dass zur Lösung gehörende Knoten jeweils paarweise adjazent sind. Die hinzugefügten Kanten müssen für eine gültige Lösung nicht überdeckt werden, sie dienen lediglich zur Lösungskonstruktion.

Das Einführen der Dummy-Nodes ist erforderlich, da ansonsten für die Ameisen ein Startknoten festgelegt werden müsste, was zu einer nicht optimalen Lösung führen kann.



Der aus G entstehende, vollständige Graph G' .

2.4.4 Konstruktion der Lösung

Die Ameisen benötigen ein Kriterium anhand dessen sie während ihres Laufes entscheiden, welcher Knoten als nächstes besucht werden soll.

Dieses Kriterium setzt sich aus Pheromonspuren (τ) sowie heuristischen Informationen (η) zusammen.

1. Die heuristische Information

Die heuristische Information wird für jeden Knoten berechnet und ergibt sich aus

$$\eta_i = \frac{a_i}{b_i}$$

wobei

- a_i angibt, wieviele mit v_i inzidente Kanten noch nicht überdeckt worden sind
- b_i angibt, wie hoch die Kosten für das Hinzufügen von v_i zur Lösung sind

Erläuterung:

$$\frac{b_i}{e_i}$$

würde angeben, wie hoch die Kosten pro überdeckter Kante beim Hinzufügen von v_i zur Lösung wären. Je kleiner dieser Wert ist, desto besser. Da aber für die heuristischen Informationen hohe Werte "gut" sein sollen, wird der Kehrwert gebildet.

Als heuristische Information für den gegebenen Graphen erhält man

$$\eta = \begin{bmatrix} 4 \\ 2 \\ 3 \\ 3 \\ 2 \\ 2 \end{bmatrix}$$

Der Knoten v_1 hat hier den "besten" heuristischen Wert ($\eta_1 = 4$), wäre also am interessantesten für die Ameisen, wenn sich diese nur anhand der heuristischen Information entscheiden müssten.

Zwischenbemerkung:

Würde nur die heuristische Information vorliegen, so läge ein Greedy-Algorithmus vor.

Es ist jedoch gezeigt worden, dass VC von einem Greedy-Algorithmus (im allgemeinen) nicht optimal gelöst werden kann. Gleiches gilt aufgrund der Definition der Klasse NPV für sämtliche NP -Vollständigen Probleme.

2. Die Pheromonspuren

Die Pheromonspuren werden an den Knoten abgelegt, was sich aus der Definition des Problems ergibt. (Zum Finden einer Lösung müssen die Knoten mit einem Gütemaß gekennzeichnet werden.)

Während des Verfahrens werden die Pheormone auf folgende Weise genutzt bzw. modifiziert

- (a) Zu Beginn des Verfahrens werden die Pheromonwerte alle mit dem gleichen Wert initialisiert, dieser ist verschieden von 0
- (b) Nach jedem Zyklus werden zunächst alle Pheromonspuren verringert (auch bezeichnet als Verdampfung bzw. Evaporation)

- (c) Danach legt jede Ameise neue Pheromone ab, wobei sich die Intensität der neu abgelegten Pheromone aus der Güte des gefundenen Lösungskandidaten der jeweiligen Ameise ergibt

Diese Aktionen laufen wie folgt ab

- (a) Initialisierung der Pheromonwerte

$$\tau_i \leftarrow \tau_0$$

$$\tau_0 \in \mathbb{R}^+$$

- (b) Verringern der Pheromonwerte nach jedem Zyklus

$$\tau_i \leftarrow (1 - \rho)\tau_i$$

Dabei handelt es sich bei ρ um ein Parameter, der angibt, wieviel Prozent der Pheromone nach jedem Zyklus verdampfen sollen. ρ wird vor dem Start des Verfahrens festgelegt und bleibt dann konstant.

- (c) Ablegen neuer Pheromone durch die Ameisen

Die Ameisen legen abhängig von den Kosten ihres konstruierten Lösungskandidaten Pheromone ab. Dabei ist die Intensität der abgelegten Pheromone umso stärker, je geringer die Kosten der Lösung sind.

Die neuen Pheromonwerte ergeben sich aus

$$\tau_i \leftarrow \tau_i + \frac{y_i}{c}$$

wobei die Kosten c berechnet werden aus

$$c = \sum_{i=1}^n b_i \cdot y_i$$

3. Kombination der heuristischen Information und Pheromonspuren

Bevor die Ameisen ihre Schritte ausführen können, muss jeder mögliche nächste Knoten bewertet werden.

Dazu wird die Nachbarschaft N_i eines Knotens v_i definiert als diejenigen Knoten, die bisher nicht besucht worden sind.

Nun wird jedem Knoten der Nachbarschaft des aktuellen Knotens eine Wahrscheinlichkeit zugeordnet. Diese gibt an, wie wahrscheinlich es ist, dass der jeweilige Knoten als nächstes zur Lösung hinzugefügt wird.

Um diese Wahrscheinlichkeit zu berechnen, werden die heuristischen Informationen η mit den Pheromonen τ "kombiniert".

Jedem Knoten $v_j \in N_i$ wird eine Wahrscheinlichkeit p_j zugeordnet, die sich berechnet lässt aus

$$p_j = \frac{\tau_j \cdot \eta_j^\beta}{\sum_{v_j \in N_i} \tau_j \cdot \eta_j^\beta}$$

wobei

- τ_j angibt, wie hoch die Pheromonkonzentration an Knoten v_j ist
- η_j angibt, welcher heuristische Wert für Knoten v_j berechnet worden ist
- β ein Parameter ist, der zur Gewichtung der heuristischen Information und der Pheromonspuren dient, β wird vor dem Start des Verfahrens festgelegt und ist konstant.

2.4.5 Ablauf des Verfahrens

Vor dem Start des Verfahrens müssen einige Werte angegeben werden, die während des Verfahrens konstant bleiben.

- Anzahl der Ameisen, die zur Konstruktion der Lösungskandidaten verwendet werden sollen
- Maximale Zyklusanzahl
- ρ , um festzulegen, welcher Anteil der Pheromone nach jedem Zyklus verdampfen soll
- β , um die Gewichtung von heuristischer Information und der Pheromonspuren vorzunehmen
- τ_0 als Initialwert für die Pheromonspuren

Nach dem Initialisieren der Pheromonspuren mit dem Wert τ_0 beginnt jede Ameise unabhängig von den anderen, einen Lösungskandidaten zu konstruieren.

Dabei werden in jedem Schritt alle möglichen Folgeknoten bewertet und entsprechend den zugeordneten Wahrscheinlichkeiten einer ausgewählt. In diesem Schritt werden auch M , y und η aktualisiert, wobei diese Informationen nicht global verwaltet werden, sondern jede Ameise dies selbst tun muss,

da sie sich abhängig vom bisher konstruierten Lösungskandidaten ändern. τ und b sind hingegen Informationen, die global verwaltet werden, da τ bezogen auf die Zyklen und b bezogen auf das Verfahren konstant sind.

Nach der Auswahl eines Knotens und dem Hinzufügen zur Lösung werden M und y aktualisiert und η neu berechnet.

Es werden so lange Knoten zur Lösung hinzugefügt bis ein zulässiger Lösungskandidat entsteht.

Haben alle Ameisen einen gültigen Lösungskandidaten gefunden, so wird τ aktualisiert und ein neuer Zyklus gestartet.

Dies wird so lange wiederholt, bis eine der beiden Abbruchsbedingungen eintritt.

3 Vergleich Algorithmik und KI

Es gibt unterschiedliche Ansätze, um das Vertex Cover Problem zu lösen. Zum einen können Ameisensysteme verwendet werden und zum anderen kann eine Lösung über Vollständige Enumeration bestimmt werden.

Die Gegenüberstellung dieser beiden Ansätze soll zeigen, welche Vorteile der Lösungsansatz mit Hilfe von Ameisensystemen bietet.

3.1 Algorithmik

Die Lösung für ein Problem kann immer dadurch gefunden werden, dass mittels Vollständiger Enumeration alle Lösungskandidaten ausprobiert werden. Dabei ist garantiert, dass die optimale Lösung gefunden wird.

Für kleine Beispiele ist diese Vorgehensweise praktikabel.

Bei großen Beispielen tritt jedoch das Problem auf, dass die Rechenzeit exponentiell mit der Eingabegröße steigt. Daher ist diese Herangehensweise im allgemeinen nicht praktikabel.

3.2 KI (Ameisensysteme)

Verwendet man Ameisensysteme zur Lösung, so werden nicht alle möglichen Lösungen ausprobiert. Das Verfahren wird durch die Kombination der heuristischen Information und der Pheromonspuren derart gelenkt, dass der Suchraum

eingeschränkt wird und die konstruierten Lösungen so gut wie die bisher beste gefundene sind oder besser. Dabei kann nicht garantiert werden, dass immer die optimale Lösung gefunden wird. Die gefundene Lösung ist in der Regel jedoch eine gute Näherung der optimalen Lösung.

3.3 Vergleich anhand eines konkreten Beispiels

Das Problem *ste36a* wurde sowohl mit Hilfe von Vollständiger Enumeration als auch mit Hilfe von Ameisensystemen gelöst.

ste36a ist ein Beispiel vom Quadratic Assignment Problem.

Bei *ste36a* geht es darum, 36 Computer-Komponenten derart auf einer Platine anzuordnen, dass die Verkabelungskosten dieser Komponenten minimal sind.

Insgesamt müssen dabei 2625 Verbindungen hergestellt werden.

Die optimale Lösung wurde mit Hilfe Vollständiger Enumeration nach 180 Stunden gefunden.

Das zur Lösung verwendete Ameisensystem hatte diese Lösung nach bereits 10 Sekunden gefunden.

Für die Simulation wurde ein Pentium *III* Rechner mit *500Mhz* verwendet.

3.4 Beispiel

Angenommen, ein Tourist möchte die 100 größten deutschen Städte besuchen.

Würde man zum Finden der optimalen Lösung Vollständige Enumeration verwenden, so müssten 100! Möglichkeiten ausprobiert werden.

$$\begin{aligned} 100! = & 93.326.215.443.944.152.681.699.238.856.266.700.490.715.968.264. \\ & 381.621.468.592.963.895.217.599.993.229.915.608.941.463.976.156. \\ & 518.286.253.697.920.827.223.758.251.185.210.916.864.000.000.000. \\ & 000.000.000.000.000 \end{aligned}$$

Das sind etwa 10^{157} Möglichkeiten.

Könnte man pro Sekunde 1.000.000 Möglichkeiten durchprobieren, müsste

man etwa 10^{151} Sekunden rechnen.

Das sind etwa 10^{145} Tage.

Und etwa 10^{142} Jahre.

Zum Vergleich: Das Universum ist ca. $13 \cdot 10^9$ Jahre alt.

Hier bietet sich die Verwendung eines Ameisensystems an. Dabei wäre es möglich, dass bereits nach kurzer Rechenzeit eine gute Lösung gefunden wäre.

References

- [1] Marco Dorigo / Thomas Stützle, Ant Colony Optimization, MIT Press 2004
- [2] Michael Sipser, Introduction to the Theory of Computation Second Edition, Thomson 2006
- [3] Rainer Lang, Vorlesung Berechenbarkeit und Komplexität, WS 2009/2010