Can ants solve optimization problems?

Dipl. Inf. Gunnar Völkel

Theoretische Informatik
Universität Ulm

17.02.2011
Overview

1. Introduction
   - Foraging behavior of ants
   - Combinatorial Optimization Problem: TSP

2. Ant Colony Optimization
   - Construction Graph & Construction Scheme
   - Algorithm Scheme

3. Theoretical aspects
   - Markov property
   - Model convergence

4. Conclusion
   - Comparison with Genetic Algorithms
   - Summary
Introduction

1. Foraging behavior of ants
2. Combinatorial Optimization Problem: TSP
Foraging behavior of ants

Figure: Argentine ant

- in many ant species: single ants able to deposit *pheromone* perceivable by other ants
- pheromone laying while moving from nest to food source (and vice versa)

⇒ pheromone trails emerge
- guidance for other foragers to food source
- even shortest way to food source optimization

---

1 © Joyce Gross
- experiment without food source (Deneubourg et al. [1990])
- pheromone trail establishes on one of the branches

Figure: Percentage of ants per 3-min period passing on the two branches of the bridge (inset). Colony of 1000 workers. (Deneubourg et al. [1990])
Shortcut Experiment

- experiment with food source (Goss et al. [1989])
- advantage gain for the shorter branch due to length difference (ants return earlier)

**Figure:** An ant colony selecting the short branches on both modules of the bridge; (a) one module of the bridge, (b) and (c): photos taken 4 and 8 mins after start. (Goss et al. [1989])
Combinatorial Optimization Problem

Search Space $S$

Solution Components

$f(s) \rightarrow \min$

$\omega(s) = \text{true} \quad \forall \omega \in \Omega$

$s \in S$
Combinatorial Optimization Problem
Traveling Salesman Problem (TSP)

shape of Germany: http://www.worldatlas.com/webimage/countrys/europe/printpage/deoutline.htm (January 2011)
Combinatorial Optimization Problem
Traveling Salesman Problem (TSP)

shape of Germany: http://www.worldatlas.com/webimage/countrys/europe/printpage/deoutline.htm (January 2011)
many important problems \textit{NP-complete}, e.g.

- Traveling Salesman Problem (TSP)
- Rucksackproblem (KNAPSACK)

⇒ worst-case runtime for exact algorithms: exponential in size of problem instance

approximate algorithms for near optimal solutions at low computational cost

- problem-specific heuristics
- metaheuristics
Ant Colony Optimization

- Construction Graph & Construction Scheme
- Algorithm Scheme
Ant Colony Optimization

- metaheuristic inspired by foraging behavior of ants
- developed by M. Dorigo et al. since 1991–1992 (recent survey paper: Dorigo and Stützle [2009])

Probabilistic solution construction procedure using

- (artificial) pheromone trails: dynamically changing at runtime to reflect “search experience”
- heuristic information about the problem

One (artificial) ant constructs one solution
Ant Colony Optimization

- metaheuristic inspired by foraging behavior of ants
- developed by M. Dorigo et al. since 1991–1992 (recent survey paper: Dorigo and Stützle [2009])

- probabilistic solution construction procedure using
  - (artificial) pheromone trails: dynamically changing at runtime to reflect “search experience”
  - heuristic information about the problem
- one (artificial) ant constructs one solution
Construction Graph
Construction Graph

Ant Colony Optimization

→ Construction Graph & Construction Scheme
Construction Graph for TSP with 6 locations
Construction Scheme

Pheromone on edges:

\[ 0 \leq \tau_{ij} \in \mathbb{R} \quad \forall i, j : i \neq j \]
Probabilistic decision:

\[ p_{12} = \frac{\tau_{12}}{\tau_{12} + \tau_{13} + \tau_{14} + \tau_{15} + \tau_{16}} \]
\[ p_{13} = \frac{\tau_{13}}{\tau_{12} + \tau_{13} + \tau_{14} + \tau_{15} + \tau_{16}} \]
\[ p_{14} = \frac{\tau_{14}}{\tau_{12} + \tau_{13} + \tau_{14} + \tau_{15} + \tau_{16}} \]
\[ p_{15} = \frac{\tau_{15}}{\tau_{12} + \tau_{13} + \tau_{14} + \tau_{15} + \tau_{16}} \]
\[ p_{16} = \frac{\tau_{16}}{\tau_{12} + \tau_{13} + \tau_{14} + \tau_{15} + \tau_{16}} \]
Probabilistic decision:

\[ p_{42} = \frac{\tau_{42}}{\tau_{42} + \tau_{43} + \tau_{45} + \tau_{46}} \]

\[ p_{43} = \frac{\tau_{43}}{\tau_{42} + \tau_{43} + \tau_{45} + \tau_{46}} \]

\[ p_{45} = \frac{\tau_{45}}{\tau_{42} + \tau_{43} + \tau_{45} + \tau_{46}} \]

\[ p_{46} = \frac{\tau_{46}}{\tau_{42} + \tau_{43} + \tau_{45} + \tau_{46}} \]
Probabilistic decision:

\[
p_{23} = \frac{\tau_{23}}{\tau_{23} + \tau_{25} + \tau_{26}}
\]

\[
p_{25} = \frac{\tau_{25}}{\tau_{23} + \tau_{25} + \tau_{26}}
\]

\[
p_{26} = \frac{\tau_{26}}{\tau_{23} + \tau_{25} + \tau_{26}}
\]
Probabilistic decision:

\[
p_{35} = \frac{\tau_{35}}{\tau_{35} + \tau_{36}}
\]
\[
p_{36} = \frac{\tau_{36}}{\tau_{35} + \tau_{36}}
\]
Probabilistic decision:
\[ p_{65} = \frac{\tau_{65}}{\tau_{65}} \]
Construction finished!

Solution

\[ s = \{c_{14}, c_{42}, c_{23}, c_{36}, c_{65}\} \]

\[ \begin{align*}
&c_1^4 \\
&c_2^4 \\
&c_3^3 \\
&c_6^3 \\
&c_5^6
\end{align*} \]
Algorithm Scheme

Iteration $n$
Algorithm Scheme

Iteration $n$

select best

Ant Colony Optimization → Algorithm Scheme
Algorithm Scheme

Iteration $n$

select best

update pheromone

Ant Colony Optimization → Algorithm Scheme
Algorithm Scheme

Iteration $n$

select best

update pheromone

Ant Colony Optimization $\rightarrow$ Algorithm Scheme
Algorithm 1: GBAS

Initialize pheromone trails $\tau_{kl}$ on the arcs $(k, l) \in \mathcal{E}$;

for iteration $n = 1, 2, \ldots$ do

for ant $a = 1, \ldots, A$ do

$u_a = \{\}$ ;

while $\exists (\text{last}(u_a), l) \in \mathcal{C}_\mathcal{E}(u_a)$ do

$k = \text{last}(u_a)$ ;

with distribution $P_\tau$ derived from $p_{kl} = \frac{\tau_{kl}}{\sum_{(k,j) \in \mathcal{C}_\mathcal{E}(u_a)} \tau_{kj}}$ do

$(k, l^*) \in P_\tau \mathcal{C}_\mathcal{E}(u_a)$ ;

end

$u_a = u_a \oplus (k, l^*)$ ;

end

end

do pheromone update ;

end
Pheromone

- pheromone initialization:
  \[ \tau_{kl} = \tau_{kl}(1) = \frac{1}{|E|} \quad \forall (k, l) \in E \]

- best-so-far solution for pheromone update

- pheromone update rule:
  \[ \tau_{kl}(n + 1) = \begin{cases} 
  (1 - \rho_n)\tau_{kl}(n) + \rho_n \cdot R(\hat{s}(n)), & c_k^l \in \hat{s}(n) \\
  (1 - \rho_n)\tau_{kl}(n), & \text{otherwise} 
\end{cases} \]

\( \hat{s}(n) \)  best-so-far solution

\( R(s) \)  (fitness proportional) reward for \( s \)

\( \rho_n \)  evaporation rate, e.g.
\[ \rho_n = \frac{c_n}{n \log(n+1)} \quad (n \geq 1), \quad 0 < \lim_{n \to \infty} c_n < 1 \]
Probabilistic decision rules

previous described probabilistic decision rule:

\[
p_{ij} = \begin{cases} 
0, & (i,j) \text{ infeasible} \\
\frac{\tau_{ij}}{\sum_{(i,r) \in E} \tau_{ir}}, & \text{otherwise, } \forall (i,r) \text{ feasible}
\end{cases}
\]

with usage of heuristic information \(\eta_{ij}(u)\):

\[
p_{ij} = \begin{cases} 
0, & (i,j) \text{ infeasible} \\
\frac{\tau_{ij} \cdot \eta_{ij}(u)^\beta}{\sum_{(i,r) \in E} \tau_{ir} \cdot \eta_{ir}(u)^\beta}, & \text{otherwise, } \forall (i,r) \text{ feasible}
\end{cases}
\]

(\(\eta_{ij}\) may depend on partial solution \(u\))
previous described probabilistic decision rule:

\[ p_{ij} = \begin{cases} 
0, & (i,j) \text{ infeasible} \\
\frac{\tau_{ij}}{\sum_{(i,r) \in E} \tau_{ir}}, & \text{otherwise, } \forall (i,r) \text{ feasible}
\end{cases} \]

with usage of heuristic information \( \eta_{ij}(u) \):

\[ p_{ij} = \begin{cases} 
0, & (i,j) \text{ infeasible} \\
\frac{\tau_{ij} \cdot \eta_{ij}(u)^{\beta}}{\sum_{(i,r) \in E} \tau_{ir} \cdot \eta_{ir}(u)^{\beta}}, & \text{otherwise, } \forall (i,r) \text{ feasible}
\end{cases} \]

(\( \eta_{ij} \) may depend on partial solution \( u \))
<table>
<thead>
<tr>
<th>Problem type</th>
<th>Problem name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routing</td>
<td>Traveling salesman (TSP)</td>
</tr>
<tr>
<td></td>
<td>Vehicle routing</td>
</tr>
<tr>
<td></td>
<td>Sequential ordering</td>
</tr>
<tr>
<td>Assignment</td>
<td>Quadratic assignment</td>
</tr>
<tr>
<td></td>
<td>Course timetabling</td>
</tr>
<tr>
<td></td>
<td>Graph coloring</td>
</tr>
<tr>
<td>Scheduling</td>
<td>Project scheduling</td>
</tr>
<tr>
<td></td>
<td>Car sequencing</td>
</tr>
<tr>
<td>Subset</td>
<td>Set covering</td>
</tr>
<tr>
<td></td>
<td>Multiple knapsack</td>
</tr>
<tr>
<td></td>
<td>Maximum clique</td>
</tr>
<tr>
<td>Machine learning</td>
<td>Classification rules</td>
</tr>
<tr>
<td></td>
<td>Bayesian networks</td>
</tr>
<tr>
<td></td>
<td>Neural networks</td>
</tr>
<tr>
<td>Bioinformatics</td>
<td>Protein folding</td>
</tr>
<tr>
<td></td>
<td>DNA Sequencing</td>
</tr>
</tbody>
</table>

**Figure:** Reduced list of applications from Dorigo and Stützle [2009]
3 Theoretical aspects

- Markov property
- Model convergence
\( \tau(n) \in (\mathbb{R}_0^+)_{|\mathcal{E}|} \) vector of pheromone values on the arcs of \( \mathcal{G}_c \) at the beginning of iteration \( n \)

\( \hat{s}(n-1) \) best found solution at the beginning of iteration \( n \), i.e. at the end of iteration \( n - 1 \) \((\hat{s}(0) \text{ defined arbitrarily})\)

- since paths are enumerable:

\[
(\tau(n), \hat{s}(n-1)) \in (\mathbb{R}_0^+)_{|\mathcal{E}|} \times \mathbb{N}
\]
Lemma (Markov property)

For GBAS, the stochastic process with states

\[ X_n = (\tau(n), \hat{s}(n-1)) \in (\mathbb{R}_0^+)^{|\mathcal{E}|} \times \mathbb{N} \quad (n = 1, 2, \ldots) \]

is an inhomogeneous Markov process in discrete time.
Theorem (Gutjahr 2002)

Let
\[ \rho_n \leq 1 - \frac{\log n}{\log(n + 1)} \quad (n \geq N, \text{ for some } N \geq 1) \quad (1) \]
\[ \sum_{n=1}^{\infty} \rho_n = \infty \quad (2) \]

Then, with probability one,
\[ \lim_{n \to \infty} \left( \tau(n), \hat{s}(n - 1) \right) = \left( \tau[s^*], s^* \right) = X_n \]

where \( s^* \) is an optimal path and \( \tau[s^*] \) is defined by
\[ \tau_{kl}[s^*] = \begin{cases} R(s^*), & \text{if } (k, l) \in s^* \\ 0, & \text{otherwise} \end{cases} \quad (3) \]

In particular, with probability one, \( \exists s^* \) such that for fixed ant \( a \)
\[ P_{s^*}(n) = P(\text{“constructing solution } s^* \text{ in iteration } n\text{”}) \to 1 \quad (n \to \infty) \]
Model convergence

Algorithm state illustration

\[ \hat{s}(n) \neq s^*, \]
\[ s^{(a)}(n) \neq s^* \quad \forall a \]
\[ \hat{s}(n) = s^*, \]
\[ s^{(a)}(n) \neq s^* \quad \exists a \]
\[ \hat{s}(n) \neq s^*, \]
\[ s^{(a)}(n) \neq s^* \quad \forall a \]
Model convergence

Algorithm state illustration

\[ \hat{s}(n) = s^*, \]
\[ s^{(a)}(n) \neq s^* \ \forall a \]

\[ \hat{s}(n) = s^*, \]
\[ s^{(a)}(n) = s^* \ \forall a \]
Model convergence

Figure: Pheromone at initialization

Figure: Pheromone during algorithm

Figure: Pheromone in state \((\tau[s^*], s^*)\)
convergence guarantee by suitable speed of “cooling” (i.e. reduction of influence of randomness)

\[ \rho_n = \frac{c_n}{n \log(n + 1)} \quad (n \geq 1), \quad 0 < \lim_{n \to \infty} c_n < 1 \]

geometric pheromone decrement on not reinforced arcs:

\[ \tau(n + r) = (1 - \rho)^r \cdot \tau(n) \]

⇒ premature convergence to suboptimal solutions

convergence result as strong as the well-known convergence property of Simulated Annealing
convergence guarantee by suitable speed of “cooling” (i.e. reduction of influence of randomness)

\[ \rho_n = \frac{c_n}{n \log(n+1)} \quad (n \geq 1), \quad 0 < \lim_{n \to \infty} c_n < 1 \]

geometric pheromone decrement on not reinforced arcs:

\[ \tau(n + r) = (1 - \rho)^r \cdot \tau(n) \]

\[ \Rightarrow \quad \text{premature convergence to suboptimal solutions} \]

convergence result as strong as the well-known convergence property of Simulated Annealing
convergence guarantee by suitable speed of “cooling” (i.e. reduction of influence of randomness)

$$\rho_n = \frac{c_n}{n \log(n + 1)} \quad (n \geq 1), \quad 0 < \lim_{n \to \infty} c_n < 1$$

geometric pheromone decrement on not reinforced arcs:

$$\tau(n + r) = (1 - \rho)^r \cdot \tau(n)$$

⇒ premature convergence to suboptimal solutions

convergence result as strong as the well-known convergence property of *Simulated Annealing*
Conclusion

- Comparison with Genetic Algorithms
- Summary
### Comparison with Genetic Algorithms (GA)

<table>
<thead>
<tr>
<th>Algorithm kind</th>
<th>Genetic Algorithm</th>
<th>Ant Colony Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>(local) search algorithm (complete solutions)</td>
<td>probabilistic solution construction algorithm</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm state</th>
<th>in individuals of population</th>
<th>in pheromone matrix</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Selection</th>
<th>fittest portion of individuals for next iteration (direct)</th>
<th>“update solutions” for the pheromone matrix (indirect)</th>
</tr>
</thead>
</table>

#### When to favor ACO over GA?
- no suitable mutation or crossover operators deducible due to complex constraints between solution components (TSP?)
- no fixed length chromosome encoding for solutions (VRP?)
most combinatorial optimization problems hard to solve

Ant Colony Optimization (ACO) metaheuristic:
- repeated probabilistic construction procedure
- indirect “communication” about solution component quality via pheromone

many applications to combinatorial optimization problems

theoretic aspects of ACO:
- notion of “Model Convergence”
- no runtime complexity results for hard problems yet
Thank you for your attention.
References


Part II

Additional Slides
Algorithm
- Metaheuristics
- Variable Aspects
- Model Convergence: Proof Outline

Example Problems
- Traveling Salesman Problem
- KNAPSACK Problem

ACO variants
- Ant System
- Max-Min Ant System
5 Algorithm

- Metaheuristics
- Variable Aspects
- Model Convergence: Proof Outline
<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Metaheuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>a problem-specific method to solve or assist in solving a combinatorial optimization problem (approximately) by</td>
<td>a problem-independent algorithmic scheme to solve a combinatorial optimization problem (approximately) by</td>
</tr>
<tr>
<td>using knowledge to select most favorable “search directions” first (search based or brute force algorithms)</td>
<td>guidance of the objective function (and usage of heuristics)</td>
</tr>
<tr>
<td>provide a criterion for omitting unpromising “search directions”</td>
<td>probabilistic construction of solutions and simultaneous learning about the problem instance</td>
</tr>
<tr>
<td></td>
<td>probabilistic alteration of solutions while getting feedback from the objective function</td>
</tr>
</tbody>
</table>
Heuristic vs Metaheuristic

Heuristic

- a **problem-specific** method to solve or assist in solving a combinatorial optimization problem (approximately) by
  - using **knowledge to select** most favorable “search directions” first (search based or brute force algorithms)
  - provide a **criterion for omitting** unpromising “search directions”

Metaheuristic

- a **problem-independent** algorithmic scheme to solve a combinatorial optimization problem (approximately) by
  - **guidance** of the objective function (and usage of heuristics)
  - **probabilistic construction** of solutions and simultaneous **learning** about the problem instance
  - **probabilistic alteration** of solutions while getting feedback from the objective function
Approximate algorithms

in general two main types:

**Construction algorithms**
- start with empty initial solution
- incremental build of solution by adding components until a complete solution is reached
- example: *Greedy Construction Heuristic*

**Local search**
- start with complete initial solution
- modifying current best solution to improve it
- current best solution replaced by improved one
- example: *Iterative Improvement*

also possible: stopping of exact algorithms before completion
Approximate algorithms

in general two main types:

### Construction algorithms
- start with empty initial solution
- incremental build of solution by adding components until a complete solution is reached
- example: **Greedy Construction Heuristic**

### Local search
- start with complete initial solution
- modifying current best solution to improve it
- current best solution replaced by improved one
- example: **Iterative Improvement**

also possible: stopping of exact algorithms before completion
Approximate algorithms

in general two main types:

- **Construction algorithms**
  - start with empty initial solution
  - incremental build of solution by adding components until a complete solution is reached
  - example: *Greedy Construction Heuristic*

- **Local search**
  - start with complete initial solution
  - modifying current best solution to improve it
  - current best solution replaced by improved one
  - example: *Iterative Improvement*

- also possible: stopping of exact algorithms before completion
- solution evaluation and comparison
- feasible continuation rules
- solution post-processing (completion)
- (start node selection)
Variable Strategic Aspects

- problem independent:
  - pheromone initialization rule $\tau_0(c^j_i)$
  - pheromone update rule
    - selection of update solution, e.g. best-so-far, iteration-best
    - update expression, e.g. using $\rho(n)$, $\tau_{\text{min}}(n)$

- problem specific:
  - heuristic function
  - local optimization
1. \( \tau_{kl}(n) \geq \frac{\text{const}}{\log n} \)

2. \( P_{\text{never}} = P(\text{"no optimal path is ever traversed"}) = 0 \)

3. \((\tau[s^*], s^*)\) is an attractor of \((\tau(n), \hat{s}(n-1))\) with attraction domain \(S_{|E|} \times \{s^*\}\) where

\[
S_p = \left\{ (x_1, \ldots, x_p) \mid x_i \geq 0, \sum_{i=1}^{p} x_i = 1 \right\}
\]

\( \forall (k, l) \notin s^* : \lim_{r \to \infty} \tau_{kl}(m + r) = 0 \)

\( \forall (k, l) \in s^* : \lim_{r \to \infty} \tau_{kl}(m + r) = R(s^*) \)

4. With probability one \( \exists s^* \) optimal such that for a fixed ant

\( P_{s^*}(n) \to 1 \quad (n \to \infty) \)
Model Convergence

Proof Outline

1. \( \tau_{kl}(n) \geq \frac{\text{const}}{\log n} \)

2. \( P_{\text{never}} = P(\text{"no optimal path is ever traversed"}) = 0 \)

3. \((\tau[s^*], s^*)\) is an attractor of \((\tau(n), \hat{s}(n - 1))\) with attraction domain \(S_{|E|} \times \{s^*\}\) where

\[
S_p = \left\{(x_1, \ldots, x_p) \mid x_i \geq 0, \sum_{i=1}^{p} x_i = 1\right\}
\]

- \(\forall (k, l) \notin s^* : \lim_{r \to \infty} \tau_{kl}(m + r) = 0\)
- \(\forall (k, l) \in s^* : \lim_{r \to \infty} \tau_{kl}(m + r) = R(s^*)\)

4. with probability one \(\exists s^*\) optimal such that for a fixed ant

\(P_{s^*}(n) \to 1 \quad (n \to \infty)\)
Model Convergence
Proof Outline

1. \[ \tau_{kl}(n) \geq \frac{\text{const}}{\log n} \]

2. \[ P_{\text{never}} = P\left(\text{“no optimal path is ever traversed”}\right) = 0 \]

3. \( (\tau[s^*], s^*) \) is an attractor of \( (\tau(n), \hat{s}(n-1)) \) with attraction domain \( S_{|E|} \times \{s^*\} \) where

\[
S_p = \left\{ (x_1, \ldots, x_p) \mid x_i \geq 0, \sum_{i=1}^{p} x_i = 1 \right\}
\]

- \( \forall (k, l) \notin s^* : \lim_{r \to \infty} \tau_{kl}(m + r) = 0 \)
- \( \forall (k, l) \in s^* : \lim_{r \to \infty} \tau_{kl}(m + r) = R(s^*) \)

4. with probability one \( \exists s^* \) optimal such that for a fixed ant

\[ P_{s^*}(n) \to 1 \quad (n \to \infty) \]
1. \( \tau_{kl}(n) \geq \frac{\text{const}}{\log n} \)

2. \( P_{\text{never}} = P(\text{"no optimal path is ever traversed"}) = 0 \)

3. \( (\tau[s^*], s^*) \) is an attractor of \( (\tau(n), \hat{s}(n-1)) \) with attraction domain \( S_{|E|} \times \{s^*\} \) where

\[
S_p = \left\{ (x_1, \ldots, x_p) \mid x_i \geq 0, \sum_{i=1}^{p} x_i = 1 \right\}
\]

- \( \forall (k,l) \notin s^* : \lim_{r \to \infty} \tau_{kl}(m+r) = 0 \)
- \( \forall (k,l) \in s^* : \lim_{r \to \infty} \tau_{kl}(m+r) = R(s^*) \)

4. with probability one \( \exists s^* \) optimal such that for a fixed ant

\[
P_{s^*}(n) \to 1 \quad (n \to \infty)
\]
Example Problems

- Traveling Salesman Problem
- KNAPSACK Problem
Traveling Salesman Problem (TSP)

\[
\text{TSP} \left\{ \begin{array}{l}
- \left( d(X_k, X_1) + \sum_{i=1}^{k} d(X_i, X_{i+1}) \right) \rightarrow \max \\
(X_1, \ldots, X_k) \in \Pi(\{v_1, v_2, \ldots, v_k\})
\end{array} \right.
\]

\begin{align*}
V &= \{v_1, \ldots, v_k\} \quad \text{set of cities to visit} \\
G &= (V, E) \quad \text{complete graph with cities as vertices (one edge between each city pair)} \\
\Pi(\{v_1, \ldots, v_k\}) &= \text{set of permutations of } V = \{v_1, \ldots, v_k\} \\
d : V \times V \rightarrow \mathbb{R}^+ &= \text{distance function (via distance matrix } D \in (\mathbb{R}^+)^{n \times n} \text{ possible)}
\end{align*}

Example Problems

KNAPSACK Problem

\[
\begin{align*}
\sum_{i=1}^{k} X_i \cdot u_i & \rightarrow \text{max} \\
(X_1, \ldots, X_k) & \in \{0,1\}^k \\
\sum_{i=1}^{k} X_i \cdot w_i & \leq W
\end{align*}
\]

Objects $1, 2, \ldots, k \in \mathbb{N}^+$

$X_i$ decision whether object $i \in \{1, \ldots, k\}$ is included ($D_i = \{0,1\}$)

$u_i \in \mathbb{R}^+$ value for usefulness of object $i \in \{1, \ldots, k\}$

$w_i \in \mathbb{N}^+$ weight of object $i \in \{1, \ldots, k\}$

$W \in \mathbb{N}^+$ capacity of knapsack

Example Problems

ACO variants

- Ant System
- Max-Min Ant System
Ant System (AS)

- update selection set:
  \[ \Delta S = \{ w^1, w^2, \ldots, w^S \} \]

- pheromone update mechanism:
  \[ \tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \frac{\rho}{S} \cdot \sum_{s=1}^{S} \Delta \tau_{ij}^s \]
  \[ \Delta \tau_{ij}^s = \begin{cases} 
  C \cdot f(w^s) & \text{if } (i, j) \in w^s \\
  0 & \text{otherwise}
  \end{cases} \]

- explanations:
  - \( \rho \) evaporation rate \( \rho \in (0, 1) \)
  - \( w^s \) feasible path of ant \( s \in \{1, \ldots, S\} \)
  - \( C \) a constant \( C > 0 \)
MAX-MIN Ant System (MMAS)

- update selection set ($\mathcal{MMAS}_{bs}$ or $\mathcal{MMAS}_{ib}$):
  \[ \Delta S = \{ w^{bs} \} \quad \text{or} \quad \Delta S = \{ w^{ib} \} \]

- pheromone update mechanism:
  \[ \tau_{ij} = \left[ (1 - \rho) \cdot \tau_{ij} + \rho \cdot \Delta \tau_{ij}^{\text{best}} \right] \frac{\tau_{\text{max}}}{\tau_{\text{min}}} \]
  \[ [\tau]_a^b = \begin{cases} a, & \tau < a \\ \tau, & a \leq \tau \leq b \\ b, & \tau > b \end{cases} \]

  \[ \Delta \tau_{ij}^{\text{best}} = \begin{cases} C \cdot f(w^{\text{best}}) & \text{if } (i, j) \in w^{\text{best}} \\ 0 & \text{otherwise} \end{cases} \]

- explanations:

  - $\tau_{\text{min}}, \tau_{\text{max}}$ min./max. pheromone value $\tau_{\text{max}} > \tau_{\text{min}} > 0$
  - $w^{\text{best}}$ best-so-far ($w^{bs}$) or iteration-best ($w^{ib}$)
  - $\rho$ evaporation rate $\rho \in (0, 1)$
  - $C$ a constant $C > 0$