Ant Colony Optimization An Introduction.

Thomas Stützle

stuetzle@informatik.tu-darmstadt.de

Technische Universität Darmstadt
Fachbereich Informatik
Ant Colony Optimization (ACO) is a population-based, general search technique for the solution of difficult combinatorial problems which is inspired by the pheromone trail laying behavior of real ant colonies.
Outline of the talk

- Biological Inspiration
- ACO algorithms
- ACO applications
- ACO metaheuristic
- ACO theory
- Conclusions
Biological Inspiration

Pheromone trail following behavior

- in many ant species the visual perceptive faculty is only very rudimentarily developed.
- most communication among individuals is based on chemicals called pheromone.
- a particular type in some ant species is *trail pheromone*, used for marking and following paths.

~ collective trail laying / trail following behavior is the inspiring source of Ant Colony Optimization.
laboratory colonies of *Iridomyrmex humilis*

ants deposit pheromone while walking from food sources to nest and vice versa

ants tend to choose, in probability, paths marked by strong pheromone concentrations
Experimental results

- equal length bridges: convergence to a single path
Experimental results

- equal length bridges: convergence to a single path
- different length paths: convergence to short path
A stochastic model was derived from the experiments and verified in simulations.

The functional form of transition probability is given by:

\[ p_{i,a} = \frac{(k + \tau_{i,a})^\alpha}{(k + \tau_{i,a})^\alpha + (k + \tau_{i,a'})^\alpha} \]

- \( p_{i,a} \): Probability of choosing branch \( a \) when being at decision point \( i \)
- \( \tau_{i,a} \): Corresponding pheromone concentration
Towards artificial ants

- real ant colonies are solving *shortest path problems*
- Ant Colony Optimization takes elements from real ant behavior to solve more complex problems than real ants
- In ACO, artificial ants are *stochastic solution construction procedures* that probabilistically build solutions exploiting
  - (artificial) *pheromone trails* which change dynamically at run time to reflect the agents’ acquired search experience
  - *heuristic information* on the problem instance being solved
Travelling Salesman Problem
Stochastic solution construction

\[
\begin{align*}
\tau_{ij}, \eta_{ij} \\
\end{align*}
\]
Stochastic solution construction

- use memory to remember partial tours
- being at a city $i$ choose next city $j$ probabilistically among feasible neighbors
- probabilistic choice depends on pheromone trails $\tau_{ij}$ and heuristic information $\eta_{ij} = 1/d_{ij}$
- “usual” action choice rule at node $i$

\[
p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta} \quad \text{if } j \in \mathcal{N}_i^k
\]

Pheromone update

- use positive feedback to reinforce components of good solutions
  - better solutions give more feedback
- use pheromone evaporation to avoid unlimited increase of pheromone trails and allow forgetting of bad choices
  - pheromone evaporation $0 < \rho \leq 1$
**Pheromone update (2)**

Example of pheromone update

\[
\tau_{ij}(t) = (1 - \rho) \cdot \tau_{ij}(t - 1) + \sum_{k=1}^{m} \Delta \tau_{ij}^k
\]

\[\Delta \tau_{ij}^k = 1/L_k\] if arc \((i, j)\) is used by ant \(k\) on its tour

\(L_k\): Tour length of ant \(k\)

\(m\): Number of ants

The resulting algorithm is called **Ant System**

---

Ant System

- **Ant System** is the first ACO algorithm proposed in 1991 by Dorigo et al.
- encouraging initial results on TSP but inferior to state-of-the-art

**Role of Ant System**

- “proof of concept”
- stimulation of further research on algorithmic variants and new applications
**Ant System: The algorithm**

procedure *Ant System for TSP*

Initialize pheromones

while (termination condition not met) do

for $i = 1$ to $n - 1$ do

for $k = 1$ to $m$ do

ApplyProbabilisticActionChoiceRule($M^k, \tau, \eta$)

end

end

GlobalPheromoneTrailUpdate

end

end *Ant System for TSP*
## ACO Algorithms: Overview

<table>
<thead>
<tr>
<th>ACO algorithm</th>
<th>Authors</th>
<th>Year</th>
<th>TSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ant System</td>
<td>Dorigo, Maniezzo &amp; Coloni</td>
<td>1991</td>
<td>yes</td>
</tr>
<tr>
<td>Elitist AS</td>
<td>Dorigo</td>
<td>1992</td>
<td>yes</td>
</tr>
<tr>
<td>Ant-Q</td>
<td>Gambardella &amp; Dorigo</td>
<td>1995</td>
<td>yes</td>
</tr>
<tr>
<td>Ant Colony System</td>
<td>Dorigo &amp; Gambardella</td>
<td>1996</td>
<td>yes</td>
</tr>
<tr>
<td>MMAS</td>
<td>Stützle &amp; Hoos</td>
<td>1996</td>
<td>yes</td>
</tr>
<tr>
<td>Rank-based AS</td>
<td>Bullnheimer, Hartl &amp; Strauss</td>
<td>1997</td>
<td>yes</td>
</tr>
<tr>
<td>ANTS</td>
<td>Maniezzo</td>
<td>1998</td>
<td>no</td>
</tr>
<tr>
<td>Best-Worst AS</td>
<td>Cordón, et al.</td>
<td>2000</td>
<td>yes</td>
</tr>
<tr>
<td>Hyper-cube ACO</td>
<td>Blum, Roli, Dorigo</td>
<td>2001</td>
<td>no</td>
</tr>
</tbody>
</table>
Elitist Ant System  

Dorigo, 1992

- strong additional reinforcement of best found solution $T^{gb}$
  (best-so-far solution)

- pheromone update rule becomes

$$
\tau_{ij}(t) = (1 - \rho) \cdot \tau_{ij}(t - 1) + \sum_{k=1}^{m} \Delta \tau_{ij}^k + e \cdot \Delta \tau_{ij}^{gb}
$$

- $\tau_{ij}^{gb}$ is defined analogous to $\Delta \tau_{ij}^k$
**Ant Colony System (ACS)**  
Gambardella, Dorigo 1996, 1997

- **pseudo-random proportional action choice rule:**
  - with probability $q_0$ an ant $k$ located at city $i$ chooses successor city $j$ with maximal $\tau_{ij}(t) \cdot [\eta_{ij}]^\beta$ (**exploitation**)
  - with probability $1 - q_0$ an ant $k$ chooses the successor city $j$ according to action choice rule used in Ant System (**biased exploration**)

global pheromone update rule

only the best-so-far solution $T^{gb}$ deposits pheromone after each iteration

$$\tau_{ij}(t) = (1 - \rho) \cdot \tau_{ij}(t - 1) + \rho \cdot \Delta \tau^{gb}_{ij}(t - 1),$$

where $\Delta \tau^{gb}_{ij}(t) = 1 / L^{gb}$.

pheromone update only affects edges contained in $T^{gb}$!
**Ant Colony System (3)**

- **local pheromone update rule**
  - ants “eat away” pheromone on the edges just crossed
  
  \[
  \tau_{ij} = (1 - \xi) \cdot \tau_{ij} + \xi \cdot \tau_0
  \]

- \( \tau_0 \) is some small constant value
- increases exploration by reducing the desirability of frequently used edges
extension of Ant System with stronger exploitation of best solutions and additional mechanism to avoid search stagnation

only the iteration-best or best-so-far ant deposit pheromone

\[ \tau_{ij}(t + 1) = (1 - \rho) \cdot \tau_{ij}(t) + \Delta \tau_{ij}^{\text{best}} \]
**MAX–MIN Ant System (2)**

- additional limits on the feasible pheromone trails
  - for all $\tau_{ij}(t)$ we have: $\tau_{\text{min}} \leq \tau_{ij}(t) \leq \tau_{\text{max}}$
  - counteracts stagnation of search through aggressive pheromone update
  - heuristics for determining $\tau_{\text{min}}$ and $\tau_{\text{max}}$

- pheromone values are initialized to $\tau_{\text{max}}$
  - $\leadsto$ stronger exploration at the start of the algorithm

- pheromone trail re-initialization to increase exploration
Comparison of ACO algorithms

- all algorithms construct $10.000 \cdot n$ tours for symmetric TSPs, $20.000 \cdot n$ tours for asymmetric TSPs
- only solution construction

<table>
<thead>
<tr>
<th>Instance</th>
<th>opt</th>
<th>MMAS</th>
<th>ACS</th>
<th>$ASe$</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>eil51.tsp</td>
<td>426</td>
<td>427.6</td>
<td>428.1</td>
<td>428.3</td>
<td>437.3</td>
</tr>
<tr>
<td>kroA100.tsp</td>
<td>21282</td>
<td>21320.3</td>
<td>21420.0</td>
<td>21522.83</td>
<td>22471.4</td>
</tr>
<tr>
<td>d198.tsp</td>
<td>15780</td>
<td>15972.5</td>
<td>16054.0</td>
<td>16205.0</td>
<td>16705.6</td>
</tr>
<tr>
<td>lin318.tsp</td>
<td>42029</td>
<td>42220.2</td>
<td>42570.0</td>
<td>43422.8</td>
<td>45535.2</td>
</tr>
<tr>
<td>ry48p.atsp</td>
<td>14422</td>
<td>14553.2</td>
<td>14565.4</td>
<td>14685.2</td>
<td>15296.4</td>
</tr>
<tr>
<td>ft70.atsp</td>
<td>38673</td>
<td>39040.2</td>
<td>39099.0</td>
<td>39261.8</td>
<td>39596.3</td>
</tr>
<tr>
<td>kro124p.atsp</td>
<td>36230</td>
<td>36773.5</td>
<td>36857.0</td>
<td>37510.2</td>
<td>38733.1</td>
</tr>
<tr>
<td>ftv170.atsp</td>
<td>2755</td>
<td>2828.8</td>
<td>2826.5</td>
<td>2952.4</td>
<td>3154.5</td>
</tr>
</tbody>
</table>
Comparison of ACO algorithms

Combining ACO with local search

- current wisdom suggests that good strategy for attacking combinatorial problems is a coupling of a constructive heuristic and a local search
- problem: finding a good coupling
- ACO plus local search seems to be such a good coupling as experimental results suggest

Advantages

- local search fine-tunes constructed solutions
- ACO algorithm alleviates "initialization problem" of local search
Experimental results

- much improved performance; e.g. $\text{MAX-MIN}$ Ant System+3-opt:
  - instances with up to 500 cities: optimal found within few seconds/minutes
  - for large instances (tested up to 3038 cities) solutions within 0.25% of optimal can be found in reasonable time ($< 1$ hour)
- performance is close to state-of-the-art
Experimental results
Observations

- common features among extensions
  - strong exploitation of best found solutions
  - the most efficient extensions use local search
- differences concern essential aspects of the search control
- best performing variants add explicit diversification features
**ACO applications: QAP**

Design a keyboard layout!

- **Goal:** find a keyboard layout that minimizes the average writing time!

- **Distance:** time for pressing two keys consecutively

- **Flow:** frequency of a letter given another letter

- **Objective function:** average writing time

\[
\sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} f_{i,j} \phi_i \phi_j
\]

\(\phi_k\) is the letter assigned to the key \(k\).

MMAS for the QAP

Two decisions have to be made:

- which letter (item) is to be chosen next?
- to which key (location) is this letter assigned?

In MAX–MIN Ant System:

- items are chosen randomly
- \( \tau_{ij} \) gives desirability of assigning item \( i \) to location \( j \)

\[
p_{ij}^{k}(t) = \frac{\tau_{ij}(t)}{\sum_{l \in N_{i}^{k}} \tau_{il}(t)}
\]
**Experimental results – 1**

<table>
<thead>
<tr>
<th>Problem instance</th>
<th>Ro-TS</th>
<th>GH</th>
<th>HAS-QAP</th>
<th>$\mathcal{M}\mathcal{M}\mathcal{A}\mathcal{S}$-QAP$_{TS}$</th>
<th>$\mathcal{M}\mathcal{M}\mathcal{A}\mathcal{S}$-QAP$_{2\text{-opt}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>random problems with uniformly distributed matrix entries (i)</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tai40a</td>
<td>0.990</td>
<td>0.884</td>
<td>1.989</td>
<td>0.794</td>
<td>1.509</td>
</tr>
<tr>
<td>tai50a</td>
<td>1.125</td>
<td>1.049</td>
<td>2.800</td>
<td>1.060</td>
<td>1.795</td>
</tr>
<tr>
<td>tai60a</td>
<td>1.203</td>
<td>1.159</td>
<td>3.070</td>
<td>1.137</td>
<td>1.882</td>
</tr>
<tr>
<td>tai80a</td>
<td>0.900</td>
<td>0.796</td>
<td>2.689</td>
<td>0.836</td>
<td>1.402</td>
</tr>
<tr>
<td><em>random flows on grids (ii)</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nug30</td>
<td>0.013</td>
<td>0.007</td>
<td>0.098</td>
<td>0.013</td>
<td>0.039</td>
</tr>
<tr>
<td>sko49</td>
<td>0.076</td>
<td>0.040</td>
<td>0.141</td>
<td>0.068</td>
<td>0.115</td>
</tr>
<tr>
<td>sko56</td>
<td>0.088</td>
<td>0.060</td>
<td>0.101</td>
<td>0.075</td>
<td>0.098</td>
</tr>
<tr>
<td>sko64</td>
<td>0.071</td>
<td>0.092</td>
<td>0.129</td>
<td>0.071</td>
<td>0.099</td>
</tr>
<tr>
<td>sko72</td>
<td>0.146</td>
<td>0.143</td>
<td>0.277</td>
<td>0.090</td>
<td>0.172</td>
</tr>
<tr>
<td>sko81</td>
<td>0.136</td>
<td>0.136</td>
<td>0.144</td>
<td>0.062</td>
<td>0.124</td>
</tr>
<tr>
<td>sko90</td>
<td>0.128</td>
<td>0.196</td>
<td>0.231</td>
<td>0.114</td>
<td>0.140</td>
</tr>
</tbody>
</table>
## Results – 2

<table>
<thead>
<tr>
<th>Problem instance</th>
<th>Ro-TS</th>
<th>GH</th>
<th>HAS-QAP</th>
<th>$\mathcal{M}_{\text{MAS-QAP}}$</th>
<th>$\mathcal{M}<em>{\text{MAS-QAP}</em>{\text{opt}}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>real-life instances (iii)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bur26a-h</td>
<td>0.002</td>
<td>0.043</td>
<td>0.0</td>
<td>0.006</td>
<td>0.0</td>
</tr>
<tr>
<td>kra30a</td>
<td>0.268</td>
<td>0.134</td>
<td>0.630</td>
<td>0.134</td>
<td>0.157</td>
</tr>
<tr>
<td>kra30b</td>
<td>0.023</td>
<td>0.054</td>
<td>0.071</td>
<td>0.044</td>
<td>0.066</td>
</tr>
<tr>
<td>ste36a</td>
<td>0.155</td>
<td>n.a.</td>
<td>n.a.</td>
<td>0.061</td>
<td>0.126</td>
</tr>
<tr>
<td>ste36b</td>
<td>0.081</td>
<td>n.a.</td>
<td>n.a.</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>randomly generated real-life like instances (iv)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tai40b</td>
<td>0.531</td>
<td>0.211</td>
<td>0.0</td>
<td>0.402</td>
<td>0.0</td>
</tr>
<tr>
<td>tai50b</td>
<td>0.342</td>
<td>0.214</td>
<td>0.192</td>
<td>0.172</td>
<td>0.009</td>
</tr>
<tr>
<td>tai60b</td>
<td>0.417</td>
<td>0.291</td>
<td>0.048</td>
<td>0.005</td>
<td>0.005</td>
</tr>
<tr>
<td>tai80b</td>
<td>1.031</td>
<td>0.829</td>
<td>0.667</td>
<td>0.591</td>
<td>0.266</td>
</tr>
<tr>
<td>tai100b</td>
<td>0.512</td>
<td>n.a.</td>
<td>n.a.</td>
<td>0.230</td>
<td>0.114</td>
</tr>
</tbody>
</table>
**ACO for static problems**

**procedure** *Ant Colony Optimization*

Initialize parameters, pheromone trails

**while** (termination condition not met) **do**

ConstructSolutions

ApplyLocalSearch  *optional*

GlobalUpdateTrails

**end**

**end** *Ant Colony Optimization*
ACO applications: routing

- goal: build routing tables to direct data traffic optimizing some measure of network performance
- a routing table entry $r_{ijd}$ gives for node $i$ and destination node $d$ of a data packet the next node $j$ to move to
- routing is difficult because routing costs are dynamic
- adaptive routing is difficult because changes in control policy result in changes in the performance measures
ACO algorithm: AntNet

- Ants are launched at regular intervals from each node to randomly chosen destinations.
- Ants are routed probabilistically biased by artificial pheromone values and heuristic information.
- Ants memorize path and trip time.
- When reaching destination node, ants retrace path and update pheromone trails.
- Data packets are routed deterministically.

AntNet is distributed and asynchronous.
Experimental results

- comparisons were done to a number of state-of-the-art algorithms
- experiments were run under varying traffic conditions
- AntNet compared very favorably to other algorithms and was shown to be very robust
- reason: ACO matches well the problem characteristics
  - stochastic state transitions
  - only distributed information is available
Applications of ACO

- Main application fields
  - \( \mathcal{NP} \)-hard problems
  - dynamic shortest path problems (network routing)
- World class performance on
  - sequential ordering, vehicle routing, quadratic assignment, open-shop scheduling, shortest common super-sequence problem, 2D-HP protein folding, packet-switched network routing, mobile ad-hoc network routing..
- Very good performance on many other problems
- Industrial applications
  - AntOptima
ACO metaheuristic was proposed *a posteriori* as a common framework for ACO applications.

Artificial ants in ACO are seen as stochastic solution construction procedures.

Solution construction is biased by:

- Pheromone trails which change at run-time
- Heuristic information on the problem instance
- The ants’ memory
- Additional capabilities via daemon actions
The ACO metaheuristic

procedure ACO metaheuristic
  ScheduleActivities
    ManageAntsActivity()
    EvaporatePheromone()
    DaemonActions() {Optional}
  end ScheduleActivities
end ACO metaheuristic
ACO as tree search

ACO metaheuristic is centrally based on notion of construction graph.

Alternative point of view: ants as stochastic tree search procedures.

Highlights ants as stochastic construction procedures = stochastic search tree exploration.

This latter view highlights natural relationships of ACO to tree search procedures such as Branch-and-Bound, backtracking, beam-search, etc.
ACO Theory

- convergence proofs  *Gutjahr 1999–2005; Stützle, Dorigo 2001-2004*
- formal relationships to other methods
  - model-based search framework  *Zlochin et al. 2005*
  - relationship to methods such as PBIL, *stochastic gradient descent*, cross-entropy method or EDAs  *Meuleau, Dorigo, 2002*
- models of ACO behavior
  - modelling the dynamics of ACO  *Merkle, Middendorf 2002-04*
  - examination of search bias  *Blum et al. 2002-05*
**ACO Theory—Convergence proofs**

Gutjahr (1999) has proved **convergence to the optimal solution** of a Graph-based Ant System:

- Let $P_t$ be the probability with which an agent traverses the optimal tour at iteration $t$.
- For each $0 < \epsilon < 1$ and for a fixed evaporation rate, if the number of agents is large enough, then
  
  $$P_t \leq 1 - \epsilon, \quad \forall t \leq t_0, t_0 = t_0(\epsilon)$$

- For each $0 < \epsilon < 1$ and for a fixed number of agents, if the evaporation rate is small enough, then
  
  $$P_t \leq 1 - \epsilon, \quad \forall t \leq t_0, t_0 = t_0(\epsilon)$$
Stützle and Dorigo (2001/02) proved convergence properties for a class of ACO algorithms called $\text{ACO}_{\tau_{\text{min}}}$:

- for any $\epsilon > 0$ they prove that the probability of finding at least once an optimal solution is $P^*(t) \geq 1 - \epsilon$
- $P^*(t)$ tends to one for $t \to \infty$

these results are directly extendable to two of the experimentally most successful ACO algorithms: $\text{MAX-MIN}$ Ant System and Ant Colony System

Gutjahr (2002) extends these proofs by to convergence with probability one
Recent Trends in ACO

- continued interest in applications
- new applications
  - multi-objective optimization
  - dynamic optimization problems
  - stochastic optimization problems
- increasing interest in theoretical issues
- methodology for applying ACO
Conclusions

Ant Colony Optimization is becoming a mature field with a variety of algorithms available, many successful applications, theoretical results, and dedicated workshops (ANTS’1998 — 2006) and journal special issues.
Conclusions

Ant Colony Optimization is becoming a mature field
- variety of algorithms available
- many successful applications
- theoretical results
- dedicated workshops (ANTS’1998 — 2006) and journal special issues

ACO is the most successful technique of the wider field of Swarm Intelligence
Interested in more?

Ant Colony Optimization

Marco Dorigo and Thomas Stützle

STOCHASTIC LOCAL SEARCH FOUNDATIONS AND APPLICATIONS

Holger Hoos, Thomas Stützle