

DM865 – Spring 2018  
Heuristics and Approximation Algorithms

## Ant Colony Optimization

Marco Chiarandini

Department of Mathematics & Computer Science  
University of Southern Denmark

# Outline

1. Adaptive Iterated Construction Search
2. Ant Colony Optimization
  - Context
  - Inspiration from Nature
3. The Metaheuristic
4. ACO Variants
5. Analysis
  - Theoretical
  - Experimental

# Outline

1. Adaptive Iterated Construction Search
2. Ant Colony Optimization
  - Context
  - Inspiration from Nature
3. The Metaheuristic
4. ACO Variants
5. Analysis
  - Theoretical
  - Experimental

# Adaptive Iterated Construction Search

**Key Idea:** Alternate construction and local search phases as in GRASP, exploiting experience gained during the search process.

## Realisation:

- Associate *weights* with possible decisions made during constructive search.
- Initialize all weights to some small value  $\tau_0$  at beginning of search process.
- After every cycle (= constructive + local local search phase), update weights based on solution quality and solution components of current candidate solution.

## Adaptive Iterated Construction Search (AICS):

initialise weights

**while** *termination criterion* is not satisfied: **do**

    generate candidate solution  $s$  using

        subsidiary randomized constructive search

    perform subsidiary local search on  $s$

    adapt weights based on  $s$

## Subsidiary constructive search:

- The solution component to be added in each step of *constructive search* is based on:
  - i) *weights* and
  - ii) heuristic function *h*.
- *h* can be standard heuristic function as, e.g., used by greedy heuristics
- It is often useful to design solution component selection in constructive search such that any solution component may be chosen (at least with some small probability) irrespective of its weight and heuristic value.

### Subsidiary local search:

- As in GRASP, local search phase is typically important for achieving good performance.
- Can be based on Iterative Improvement or more advanced LS method (the latter often results in better performance).
- Tradeoff between computation time used in construction phase *vs* local search phase (typically optimized empirically, depends on problem domain).

## Weight updating mechanism:

- Typical mechanism: increase weights of all solution components contained in candidate solution obtained from local search.
- Can also use aspects of search history;  
*e.g.*, *current candidate solution* can be used as basis for weight update for additional intensification.



Example: A simple AICS algorithm for the TSP (1/2)

[ Based on Ant System for the TSP, Dorigo et al. 1991 ]

- Search space and solution set as usual (all Hamiltonian cycles in given graph  $G$ ). However represented in a construction tree  $T$ .
- Associate weight  $\tau_{ij}$  with each edge  $(i, j)$  in  $G$  and  $T$
- Use heuristic values  $\eta_{ij} := 1/w_{ij}$ .
- Initialize all weights to a small value  $\tau_0$  (parameter).
- *Constructive search* start with randomly chosen vertex and iteratively extend partial round trip  $\pi$  by selecting vertex not contained in  $\pi$  with probability

$$\Pr(\pi_j) = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N'(i)} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta}$$

Example: A simple AICS algorithm for the TSP (2/2)

- *Subsidiary local search* = typical iterative improvement
- *Weight update* according to

$$\tau_{ij} := (1 - \rho) \cdot \tau_{ij} + \Delta(ij, s')$$

where  $\Delta(i, j, s') := 1/f(s')$ , if edge  $ij$  is contained in the cycle represented by  $s'$ , and 0 otherwise.

- Criterion for weight increase is based on intuition that edges contained in short round trips should be preferably used in subsequent constructions.
- Decay mechanism (controlled by parameter  $\rho$ ) helps to avoid unlimited growth of weights and lets algorithm forget past experience reflected in weights.
- (Just add a population of cand. solutions and you have an Ant Colony Optimization Algorithm!)

# Outline

1. Adaptive Iterated Construction Search
2. Ant Colony Optimization
  - Context
  - Inspiration from Nature
3. The Metaheuristic
4. ACO Variants
5. Analysis
  - Theoretical
  - Experimental

# Swarm Intelligence

## Definition: Swarm Intelligence

**Swarm intelligence** deals with systems composed of many **individuals** that coordinate using decentralized control and self-organization.

In particular, it focuses on the collective behaviors that **emerges** from the local interactions of the individuals with each other and with their environment and without the presence of a coordinator

## Examples:

### Natural swarm intelligence

- colonies of ants and termites
- schools of fishes
- flocks of birds
- herds of land animals

### Artificial swarm intelligence

- artificial life (boids)
- robotic systems
- computer programs for tackling optimization and data analysis problems.

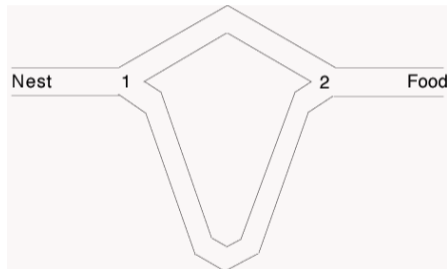
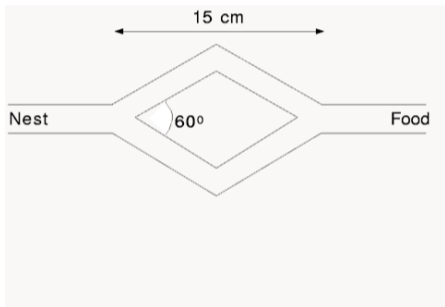
# Swarm Intelligence

Research goals in Swarm Intelligence:

- **scientific**  
modelling swarm intelligence systems to understand the mechanisms that allow coordination to arise from local individual-individual and individual-environment interactions
- **engineering**  
exploiting the understanding developed by the scientific stream in order to design systems that are able to solve problems of practical relevance

## The Biological Inspiration

Double-bridge experiment [Goss, Aron, Deneubourg, Pasteels, 1989]



- If the experiment is repeated a number of times, it is observed that each of the two bridges is used in about 50% of the cases.
- About 100% the ants select the shorter bridge

# Self-organization

Four basic ingredients:

- 1 Multiple interactions
- 2 Randomness
- 3 Positive feedback (reinforcement)
- 4 Negative feedback (evaporating, forgetting)

Communication is necessary

- Two types of communication:
  - **Direct:** antennation, trophallaxis (food or liquid exchange), mandibular contact, visual contact, chemical contact, etc.
  - **Indirect:** two individuals interact indirectly when one of them modifies the environment and the other responds to the new environment at a later time.  
This is called **stigmergy** and it happens through **pheromone**.

# Stigmergy

- "The coordination of tasks and the regulation of constructions does not depend directly on the workers, but on the constructions themselves. The worker does not direct his work, but is guided by it. It is to this special form of stimulation that we give the name STIGMERGY (stigma, sting; ergon, work, product of labour = stimulating product of labour)." Grassé P. P., 1959

*Stigmergy*  
*Stimulation of workers*  
*by the performance*  
*they have achieved*  
*Grassé P. P., 1959*



# Mathematical Model

[Goss et al. (1989)] developed a model of the observed behavior:

Assuming that at a given moment in time,

- $m_1$  ants have used the first bridge
- $m_2$  ants have used the second bridge,

The probability  $\Pr[X = 1]$  for an ant to choose the first bridge is:

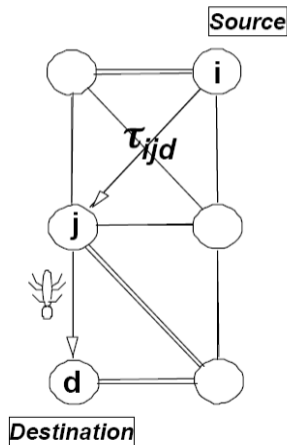
$$\Pr[X = 1] = \frac{(m_1 + k)^h}{(m_1 + k)^h + (m_2 + k)^h}$$

(parameters  $k$  and  $h$  are to be fitted to the experimental data)

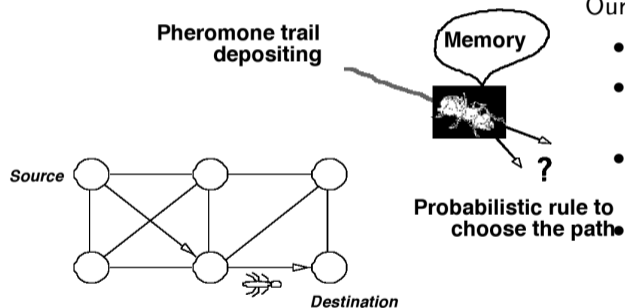
## Why Does it Work?

Three important components:

- TIME: a shorter path receives pheromone quicker (this is often called: "differential length effect")
- QUALITY: a shorter path receives more pheromone
- COMBINATORICS: a shorter path receives pheromone more frequently because it is likely to have a lower number of decision points



# From Real to Artificial Ants



## Our Basic Design Choices

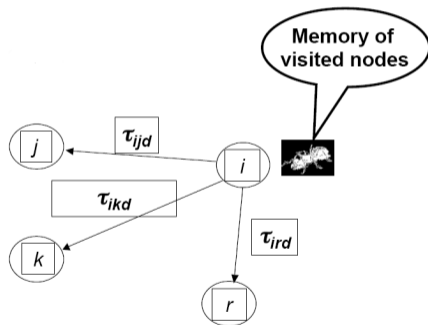
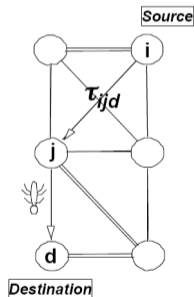
- Ants are given a **memory** of visited nodes
- Ants **build solutions probabilistically** (without updating pheromone trails)
- Ants **deterministically** retrace backward the forward path to **update pheromone**
- Ants **deposit** a quantity of pheromone **function of the quality** of the solution they generated

# From Real to Artificial Ants

Using Pheromone and Memory to Choose the Next Node

For ant  $k$ :

$$p_{ijd}^k(t) = f(\tau_{ijd}(t))$$

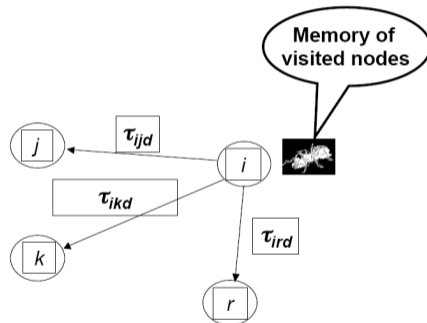


# From Real to Artificial Ants

## Ants' Probabilistic Transition Rule

For ant  $k$ :

$$p_{ijd}^k(t) = \frac{[\tau_{ijd}(t)]^\alpha}{\sum_{h \in J_i^k} [\tau_{ihd}(t)]^\alpha}$$



- $\tau_{ijd}$  is the amount of pheromone trail on edge  $(i, j, d)$
- $J_i^k$  is the set of feasible nodes ant  $k$  positioned on node  $i$  can move to

# From Real to Artificial Ants

## Ants' Pheromone Trail: Deposition and Evaporation

Evaporation:

$$\tau_{ijd}(t+1) \leftarrow (1 - \rho) \cdot \tau_{ijd}(t)$$

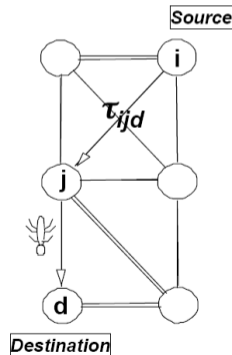
Deposition

$$\tau_{ijd}(t+1) \leftarrow \tau_{ijd}(t+1) + \Delta_{ijd}^k(t)$$

$(i, j)$ 's are the links visited by ant  $k$ , and

$$\Delta_{ijd}^k(t) \sim \text{quality}^k$$

eg:  $\text{quality}^k$  proportional to the inverse of the time it took

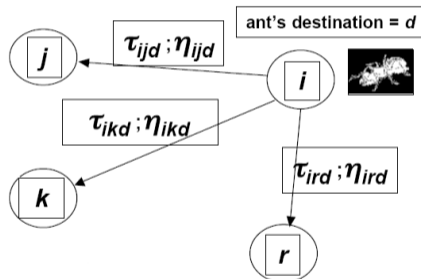


## From Real to Artificial Ants

### Using Pheromones and Heuristic to Choose the Next Node

For ant  $k$

$$p_{ij d}^k(t) = f(\tau_{ij d}(t), \eta_{ij d}(t))$$

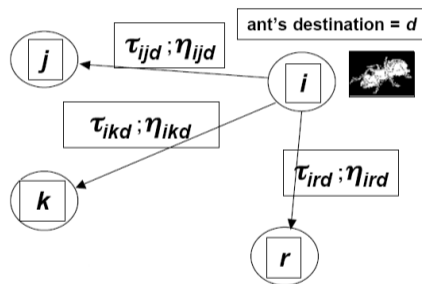


- $\tau_{ij d}$  is a value stored in a pheromone table
- $\eta_{ij d}$  is a heuristic evaluation of link  $(i, j, d)$  which introduces problem specific information

# From Real to Artificial Ants

## Ants' Probabilistic Transition Rule (Revised)

$$p_{ij d}^k(t) = \frac{[\tau_{ij d}(t)]^\alpha \cdot [\eta_{ij d}(t)]^\beta}{\sum_{h \in J_i^k} [\tau_{ih d}(t)]^\alpha \cdot [\eta_{ih d}(t)]^\beta}$$



- $\tau_{ij d}$  is the amount of pheromone trail on edge  $(i, j, d)$
- $\eta_{ij d}$  is the heuristic evaluation of link  $(i, j, d)$
- $J_i^k$  is the set of feasible nodes ant  $k$  positioned on node  $i$  can move to



# From Real to Artificial Ants

## Simple Ant Colony Optimization Algorithm

1. Ants are launched at regular instants from each node to randomly chosen destinations
2. Ants build their paths probabilistically with a probability function of:
  - artificial pheromone values
  - heuristic values
3. Ants memorize visited nodes and costs incurred
4. Once reached their destination nodes, ants retrace their paths backwards, and update the pheromone trails
5. Repeat from 1.

The pheromone trail is the stigmergic variable

# Artificial versus Real Ants:

## Main Differences

### Artificial ants:

- Live in a discrete world
- Deposit pheromone in a problem dependent way
- Can have extra capabilities:
  - local search, lookahead, backtracking
- Exploit an internal state (memory)
- Deposit an amount of pheromone function of the solution quality
- Can use heuristics

# Outline

1. Adaptive Iterated Construction Search
2. Ant Colony Optimization
  - Context
  - Inspiration from Nature
3. **The Metaheuristic**
4. ACO Variants
5. Analysis
  - Theoretical
  - Experimental

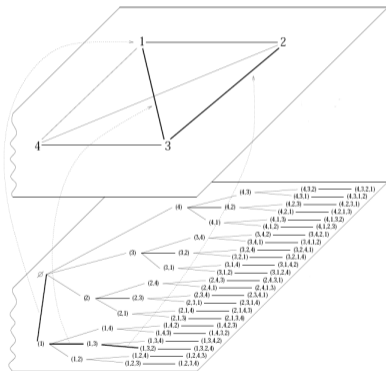
# Ant Colony Optimization

## The Metaheuristic

- The optimization problem is transformed into the problem of finding the best path on a weighted graph  $G(V, E)$  called **construction graph**
- The artificial ants incrementally build solutions by moving on the construction graph.
- The solution construction process is
  - **stochastic**
  - biased by a **pheromone model**, that is, a set of parameters associated with graph components (either nodes or edges) whose values are modified at runtime by the ants.
- All *pheromone trails* are initialized to the same value,  $\tau_0$ .
- At each iteration, *pheromone trails* are updated by decreasing (*evaporation*) or increasing (*reinforcement*) some trail levels on the basis of the solutions produced by the ants

# Ant Colony Optimization

Example: A simple ACO method for the TSP



- *Construction graph*
- To each edge  $ij$  in  $G$  associate
  - pheromone trails  $\tau_{ij}$
  - heuristic values  $\eta_{ij} := \frac{1}{c_{ij}}$
- Initialize pheromones
- *Probabilistic construction:*

$$p_{ij} = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta}, \alpha \text{ and } \beta \text{ are parameters}$$

- *Update pheromone trail levels*

## ACO Metaheuristic

- Population-based method in which artificial ants iteratively construct candidate solutions.
- Solution construction is probabilistically biased by pheromone trail information, heuristic information and partial candidate solution of each ant (memory).
- Pheromone trails are modified during the search process to reflect collective experience.

### Ant Colony Optimization (ACO):

initialize pheromone trails

**while** termination criterion is not satisfied **do**

    generate population  $P$  of candidate solutions

        using subsidiary randomized constructive search

    apply subsidiary local search on  $P$

    update pheromone trails based on  $P$

## Note

- In each cycle, each ant creates one candidate solution using a **constructive search procedure**.
- Ants build solutions by performing randomized walks on a **construction graph**  $G = (V, E)$  where  $V$  are solution components and  $G$  is **fully connected**.
- All **pheromone trails** are initialized to the same value,  $\tau_0$ .
- **Pheromone update** typically comprises uniform decrease of all trail levels (**evaporation**) and increase of some trail levels based on candidate solutions obtained from construction + local search.
- **Subsidiary local search** is (often) applied to individual candidate solutions.
- **Termination criterion** can include conditions on make-up of current population, e.g., variation in solution quality or distance between individual candidate solutions.

### Example: A simple ACO algorithm for the TSP (Revised)

- Search space and solution set: all Hamiltonian cycles in given graph  $G$ .
- Associate pheromone trails  $\tau_{ij}$  with each edge  $(i, j)$  in  $G$ .
- Use heuristic values  $\eta_{ij} := \frac{1}{c_{ij}}$
- Initialize all weights to a small value  $\tau_0$  ( $\tau_0 = 1$ ).
- *Constructive search*: Each ant starts with randomly chosen vertex and iteratively extends partial round trip  $\pi^k$  by selecting vertex not contained in  $\pi^k$  with probability

$$\Pr(\pi_j) = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta}$$

$\alpha$  and  $\beta$  are parameters.



## Example: A simple ACO algorithm for the TSP (2)

- *Subsidiary local search*: Perform iterative improvement based on standard 2-exchange neighborhood on each candidate solution in population (until local minimum is reached).
- *Update pheromone trail levels* according to

$$\tau_{ij} := (1 - \rho) \cdot \tau_{ij} + \sum_{s \in sp'} \Delta_{ij}(s)$$

where  $\Delta_{ij}(s) := 1/C^s$  if edge  $(i, j)$  is contained in the cycle represented by  $s$ , and 0 otherwise.

*Motivation*: Edges belonging to highest-quality candidate solutions and/or that have been used by many ants should be preferably used in subsequent constructions.

- *Termination*: After fixed number of cycles (= construction + local search phases).

# Outline

1. Adaptive Iterated Construction Search
2. Ant Colony Optimization
  - Context
  - Inspiration from Nature
3. The Metaheuristic
4. **ACO Variants**
5. Analysis
  - Theoretical
  - Experimental

# ACO Variants

## Variants of ACO tested on the TSP

- Ant System AS (Dorigo et al., 1991)
- Elitist AS (EAS)(Dorigo et al., 1991; 1996)
  - The iteration best solution adds more pheromone
- Rank-Based AS (ASrank)(Bullnheimer et al., 1997; 1999)
  - Only best ranked ants can add pheromone
  - Pheromone added is proportional to rank
- Max-Min AS (MMAS)(Stützle & Hoos, 1997)
  
- Ant Colony System (ACS) (Gambardella & Dorigo, 1996; Dorigo & Gambardella, 1997)
- Approximate Nondeterministic Tree Search ANTS (Maniezzo 1999)
- Hypercube AS (Blum, Roli and Dorigo, 2001)

# Ant System

- Initialization:

$$\tau_{ij} = \tau_o = \frac{m}{C^{NN}}$$

Motivation: slightly more than what evaporates

- Construction:  $m$  ants in  $m$  randomly chosen cities

$$\Pr(\pi_j) = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta}, \quad \alpha \text{ and } \beta \text{ parameters}$$

- Update

$$\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij} \quad \text{to all the edges}$$

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^m \Delta_{ij}^k \quad \text{to the edges visited by the ants, } \Delta_{ij}^k = \frac{1}{C^k}$$

# Elitist Ant System

- Update

$$\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij}$$

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^m \Delta_{ij}^k + e \cdot \Delta_{ij}^{bs}$$

to all the edges

to the edges visited by the ants

$$\Delta_{ij}^{bs} = \begin{cases} \frac{1}{C^{bs}} & (ij) \text{ in tour } k, bs \text{ best-so-far} \\ 0 & \text{otherwise} \end{cases}$$

# Rank-based Ant System

- Update: only  $w - 1$  best ranked ants + the best-so-far solution deposit pheromone:

$$\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij} \quad \text{to all the edges}$$
$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{w-1} \Delta_{ij}^k + w \cdot \Delta_{ij}^{bs} \quad \text{to the edges visited by the ants}$$

$$\Delta_{ij}^k = \frac{1}{C^k}$$

$$\Delta_{ij}^{bs} = \frac{1}{C^{bs}}$$

# MAX-MIN Ant System (MMAS)

Peculiarities in pheromone management:

- Update

$$\begin{aligned} \tau_{ij} &\leftarrow (1 - \rho) \cdot \tau_{ij} && \text{to all the edges} \\ \tau_{ij} &\leftarrow \tau_{ij} + \Delta_{ij}^{bs} && \text{only to the edges visited by the best ant} \end{aligned}$$

Meaning of **best** alternates during the search between:

- best-so-far
- iteration best
- bounded values  $\tau_{min}$  and  $\tau_{max}$
- $\tau_{max} = \frac{1}{\rho C^*}$  and  $\tau_{min} = \frac{\tau_{max}}{a}$
- Reinitialization of  $\tau$  if:
  - stagnation occurs
  - idle iterations

Results obtained are better than AS, EAS, and ASrank, and of similar quality to ACS's

# Ant Colony System (ACS)

Three main ideas:

- Different state transition rule

$$j = \begin{cases} \arg \max_{l \in N_i^k} \{\tau_{il} \eta_{il}^\beta\} & \text{if } q \leq q_0 \\ p_{ij} = \frac{[\tau_{ij}]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}]^\alpha \cdot [\eta_{il}]^\beta} & \text{otherwise} \end{cases}$$

- **Global** pheromone update

$$\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij} + \rho \Delta_{ij}^{bs}(s)$$

to only  $(ij)$  in best-so-far tour ( $O(n)$  complexity)

- **Local** pheromone update: happens during tour construction to avoid other ants to make the same choices:

$$\tau_{ij} \leftarrow (1 - \epsilon) \cdot \tau_{ij} + \epsilon \tau_0 \quad \epsilon = 0.1, \tau_0 = \frac{1}{nC^{NN}}$$

Parallel construction preferred to sequential construction



# Approximate Nondeterministic Tree Search

- Use of lower bound to compute heuristic value
  - Add an arc to the current partial solution and estimate LB of complete solution
- Different solution construction rule

$$p_{ij}^k = \frac{\alpha\tau_{ij} + (1 - \alpha)\eta_{ij}}{\sum_{l \in N_i^k} \alpha\tau_{il} + (1 - \alpha)\eta_{il}}$$

- Different pheromone trail update rule

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{i=1}^k \Delta_{ij}^k$$

$$\Delta_{ij}^k = \begin{cases} \theta \left( 1 - \frac{C^k - LB}{L_{avg} - LB} \right) & \text{if } (ij) \text{ belongs to } T^k \\ 0 & \text{otherwise} \end{cases}$$

## Strongly Invariant ACO

Considers instances which are equivalent up to a linear transformation of units.

The siACO is an algorithm with the property that its internal state at each iteration is the same on equivalent instances.

For AS:

- Use heuristic values  $\eta_{ij} := \frac{C^{NN}}{n \cdot c_{ij}}$
- Update according to

$$\tau_{ij} := (1 - \rho) \cdot \tau_{ij} + \sum_{s \in sp'} \Delta_{ij}(s)$$

where  $\Delta_{ij}(s) := \frac{C^{NN}}{m \cdot C^s}$   
if edge  $(i, j)$  is contained in the cycle represented by  $s$ , and 0 otherwise.

Can be extended to other ACO versions and to other problems: QAP and Scheduling

# Outline

1. Adaptive Iterated Construction Search
2. Ant Colony Optimization
  - Context
  - Inspiration from Nature
3. The Metaheuristic
4. ACO Variants
5. Analysis
  - Theoretical
  - Experimental

## Analytical studies

- [Gutjahr, *Future Generation Computer Systems*, 2000, *Information Processing Letters* 2002] and [Stützle and Dorigo, *IEEE Trans. on Evolutionary Computation*, 2002] have proved **convergence** with prob 1 to the optimal solution of different versions of ACO
- Runtime analysis of Different MMAS ACO algorithms on Unimodal Functions and Plateaus [Neumann, Sudholt and Witt, *Swarm Intelligence*, 2009]
- [Meuleau and Dorigo, *Artificial Life Journal*, 2002] have shown that there are strong relations between ACO and **stochastic gradient descent** in the space of pheromone trails, which converges to a local optimum with prob 1
- [Zlochin et al. *TR*, 2001] have shown the tight relationship between ACO and **estimation of distribution algorithms**
- Studies on pheromone dynamics [Merkle and Middendorf, *Evolutionary Computation*, 2002]

## Things to check

- Parameter Tuning
- Synergy
- Pheromone Development
- Strength of local search (exploitation vs exploration)
- Heuristic Information (linked to parameter  $\beta$ )  
Results show that with  $\beta = 0$  local search can still be enough
- Lamarckian vs Darwinian Pheromone Updates
- Run Time impact

# Parameter tuning

Our experimental study of the various ACO algorithms for the TSP has identified parameter settings that result in good performance. For the parameters that are common to almost all the ACO algorithms, good settings (if no local search is applied) are given in the following table.

| ACO algorithm      | $\alpha$ | $\beta$ | $\rho$ | $m$ | $\tau_0$                |
|--------------------|----------|---------|--------|-----|-------------------------|
| AS                 | 1        | 2 to 5  | 0.5    | $n$ | $m/C^{nm}$              |
| EAS                | 1        | 2 to 5  | 0.5    | $n$ | $(e+m)/\rho C^{nm}$     |
| AS <sub>rank</sub> | 1        | 2 to 5  | 0.1    | $n$ | $0.5r(r-1)/\rho C^{nm}$ |
| M/MAS              | 1        | 2 to 5  | 0.02   | $n$ | $1/\rho C^{nm}$         |
| ACS                | —        | 2 to 5  | 0.1    | 10  | $1/nC^{nm}$             |

Here,  $n$  is the number of cities in a TSP instance. All variants of AS also require some additional parameters. Good values for these parameters are:

**EAS:** The parameter  $e$  is set to  $e = n$ .

**AS<sub>rank</sub>:** The number of ants that deposit pheromones is  $w = 6$ .

**M/MAS:** The pheromone trail limits are  $\tau_{max} = 1/\rho C^{br}$  and  $\tau_{min} = \tau_{max}(1 - \sqrt[3]{0.05})/((avg - 1) \cdot \sqrt[3]{0.05})$ , where  $avg$  is the average number of different choices available to an ant at each step while constructing a solution (for a justification of these values see Stützle & Hoos (2000)). When applied to small TSP instances with up to 200 cities, good results are obtained by using always the iteration-best pheromone update rule, while on larger instances it becomes increasingly important to alternate between the iteration-best and the best-so-far pheromone update rules.

**ACS:** In the local pheromone trail update rule:  $\zeta = 0.1$ . In the pseudorandom proportional action choice rule:  $q_0 = 0.9$ .

It should be clear that in individual instances, different settings may result in much better performance. However, these parameters were found to yield reasonable performance over a significant set of TSP instances.

## How Many Ants?

