DM63
HEURISTICS FOR
COMBINATORIAL OPTIMIZATION

Lecture 10

Ant Colony Optimization

Marco Chiarandini

Outline

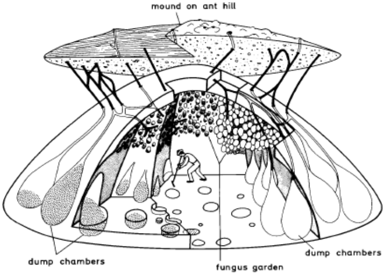
Outline

Insects, Social Insects, and Ants

- 10^{18} living insects (rough estimate)
- about 2% of all insects are social
- Social insects are: All ants All termites Some bees Some wasps
- 50% of all social insects are ants
- Avg weight of one ant between 1 and 5 mg
- Tot weight ants $\sim$ Tot weight humans
- Ants have colonized Earth for 100 million years, Homo sapiens sapiens for 50,000 years

Ants

- Fungus growers
- Breeding ants
- Weaver ants
- Harvesting ants
- Army ants
- Slavemaker ants



Ant Colony Societies

- Ant colony size: from as few as 30 to millions of workers
- Work division: Reproduction ⇒ queen

  Defense ⇒ soldiers
  Food collection ⇒ specialized workers
  Brood care ⇒ specialized workers
  Nest brooming ⇒ specialized workers
  Nest building ⇒ specialized workers
  Nest building ⇒ specialized workers

How Do Ants and Social Insects Coordinate their Activities?

- Self-organization:
  - Set of dynamical mechanisms whereby structure appears at the global level as the result of interactions among lower-level components
  - The rules specifying the interactions among the system's constituent units are executed on the basis of purely local information, without reference to the global pattern, which is an emergent property of the system rather than a property imposed upon the system by an external ordering

Self-organization

Four basic ingredients:

- Multiple interactions
- Randomness
- Positive feedback
  E.g., recruitment and reinforcement
- Negative feedback
  E.g., limited number of available foragers

Characteristics of a Self-organized System

- Creation of spatio-temporal structures
  E.g., foraging trails, nest architectures, social organization
- Multistability
  (i.e., possible coexistence of several stable states) E.g., ants exploit only one of two identical food sources
- Existence of bifurcations when some parameters change
  E.g., termites move from a non-coordinated to a coordinated phase only if their density is higher than a threshold value

How Do Social Insects Achieve Self-organization?

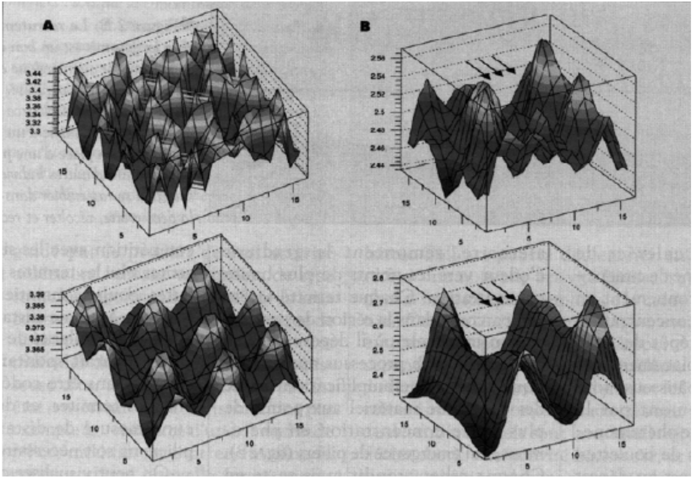
- Communication is necessary
- Two types of communication:
  Direct: antennation, trophallaxis (food or liquid exchange), mandibular contact, visual contact, chemical contact, etc.
  Indirect: two individuals interact indirectly when one of them modifies the environment and the other responds to the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

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)."
  Grass P. P., 1959

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

Stigmergy + External Forces:
Simulation of the Nest Building Activity



Deneubourg, 1977

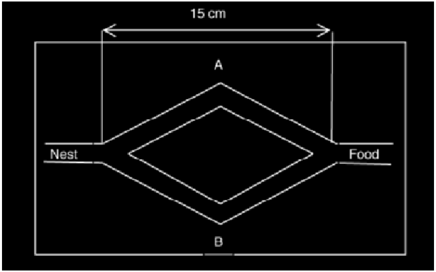
Termites' Nests

Sign-based Stigmergy

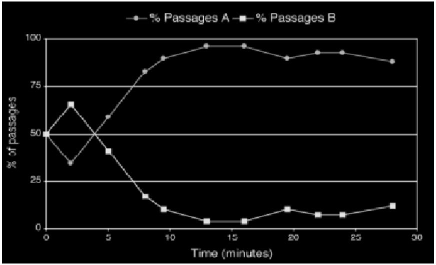
Example: Trail following and ants foraging behavior while walking, ants and termites

- May deposit a pheromone on the ground
- Follow with high probability pheromone trails they sense on the ground

Ants Foraging Behavior
Example: The Double Bridge Experiment



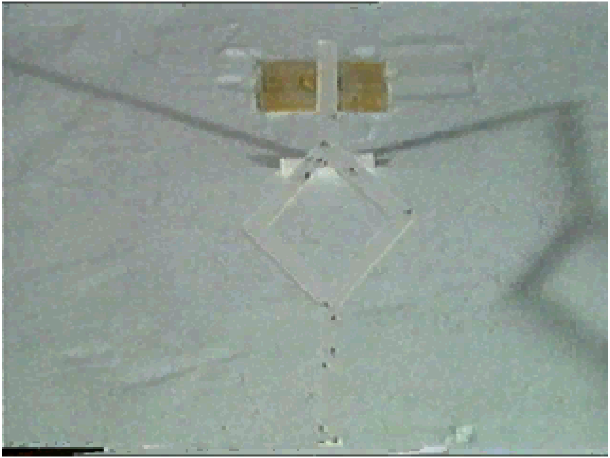
Simple bridge



% of ant passages on the two branches

Goss et al., 1989, Deneubourg et al., 1990

Double Bridge Experiment



Movie by Jean-Louis Deneubourg

"Artificial" Stigmergy

Indirect communication mediated by modifications of environmental states which are only locally accessible by the communicating agents

Dorigo & Di Caro, 1999

- Characteristics of artificial stigmergy:
  - Indirect communication
  - Local accessibility

What Are Ant Algorithms?

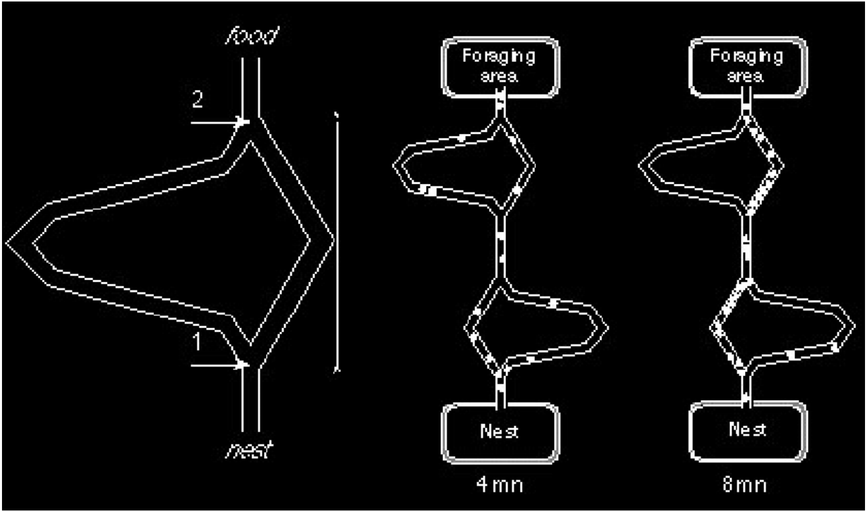
- Ant algorithms are multi-agent systems
  that exploit artificial stigmergy as
  a means for coordinating artificial ants
  for the solution of computational problems

Real Ants Inspire Ant Algorithms

- Foraging ⇒ ACO:
  - Shortest path
  - Combinatorial optimization
  - Network routing
- Division ⇒ Adaptive task allocation of labor
- Cemetery organization and brood sorting
  - Robot clustering
  - Graph partitioning
- Cooperative transport
  - Robotic implementations

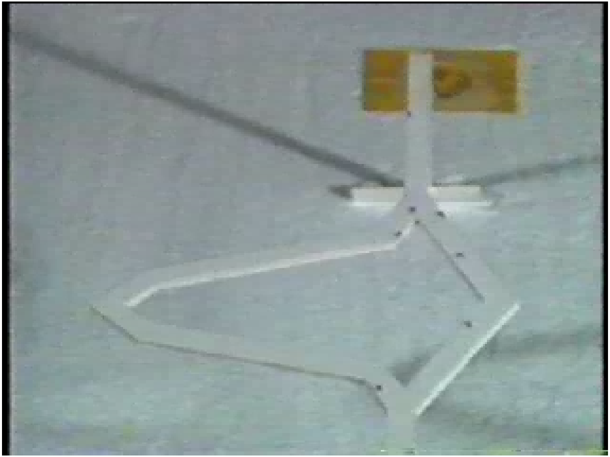
Ant behavior ⇒ Model ⇒ Derived Application

Asymmetric Bridge Experiment



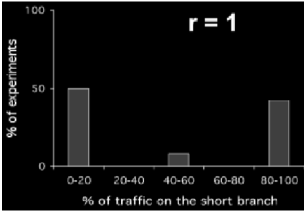
Goss et al., 1989

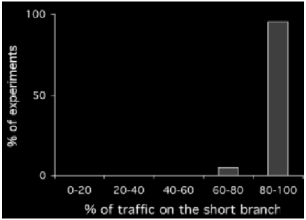
Asymmetric Bridge Experiment



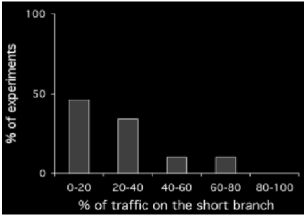
Goss et al., 1989

Some Results



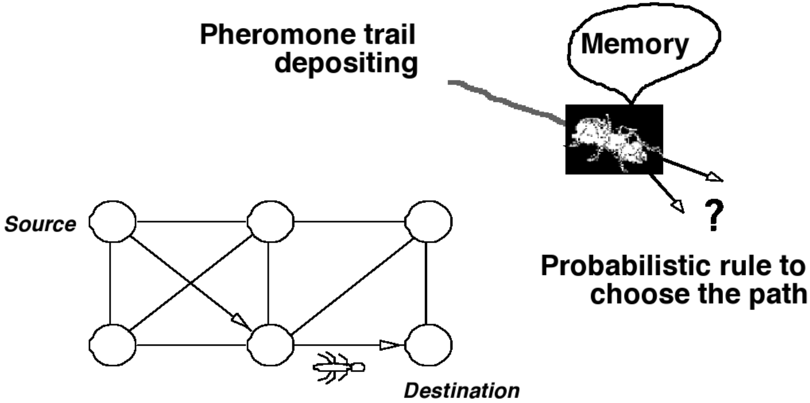


r is the length ratio among the two bridges



short edge added later

Artificial Ants and the Shortest Path Problem

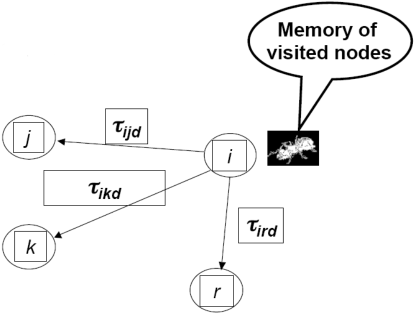


Our Design Choices

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

Using Pheromone and Memory to Choose the Next Node

$$p^k_{ijd}(t) = f\big(\tau_{ijd}(t)\big)$$



Ants' Probabilistic Transition Rule

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

- τ_{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

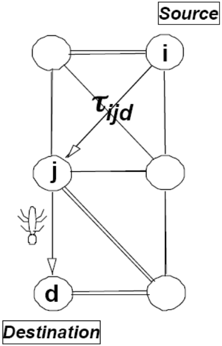
Ants' Pheromone Trail Depositing

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

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

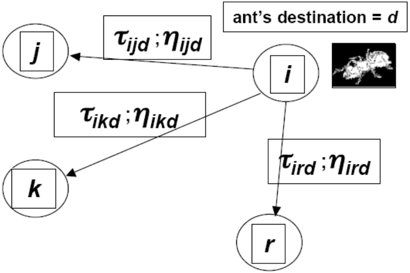
$$\Delta\tau_{ijd}^{k}(t) = \text{quality}^{k}$$

where quality^k is set proportional to the inverse of the time it took ant k to build the path from i to d via j.



Using Pheromones and Heuristic to Choose the Next Node

$$p^k_{ijd}(t) = f(\tau_{ijd}(t), \eta_{ijd}(t))$$



- τ_{ijd} is a value stored in a pheromone table
- η_{ijd} is an heuristic evaluation of link (i, j, d) which introduces problem specific information

The Simple Ant Colony Optimization Algorithm

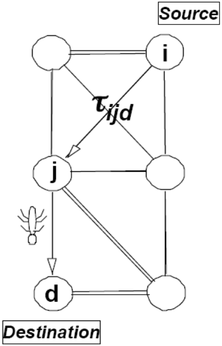
- Ants are launched at regular instants from each node to randomly chosen destinations
- Ants build their paths probabilistically with a probability function of: (i) artificial pheromone values, and (ii) heuristic values
- Ants memorize visited nodes and costs incurred
- Once reached their destination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

The pheromone trail is the stigmergic variable

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



How Does it Work?

- It works very well on
  - shortest path problems with dynamic costs (e.g., routing in telecommunications networks)
  - constrained shortest path problems (e.g., NP-hard problems)

Artificial versus Real Ants:
Main Similarities

- Colony of individuals
- Exploitation of stigmergy & pheromone trail
  - Stigmergic, indirect communication
  - Pheromone evaporation
  - Local access to information
- Shortest path & local moves (no jumps)
- Stochastic and myopic state transition

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 local heuristic

Ant Colony Optimization 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.
- 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:

 generate population sp of candidate solutions
  using *subsidiary randomized constructive search*

 perform *subsidiary perturbative search* on sp

 update pheromone trails based on sp

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, τ_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 + perturbative search.
- *Subsidiary perturbative 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 (1)

- Search space and solution set as usual (all Hamiltonian cycles in given graph G).
- Associate pheromone trails τ_{ij} with each edge (i, j) in G.
- Use heuristic values $\eta_{ij} := \frac{1}{c_{ij}}$ (better: $\eta_{ij} := \frac{C^{NN}}{n \cdot c_{ij}}$)
- Initialize all weights to a small value τ_0 ($\tau_0 = 1$).
- *Constructive search:* Each ant starts with randomly chosen vertex and iteratively extends partial round trip π^k by selecting vertex not contained in π^k with probability

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

α and β are parameters.

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

- *Subsidiary perturbative 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

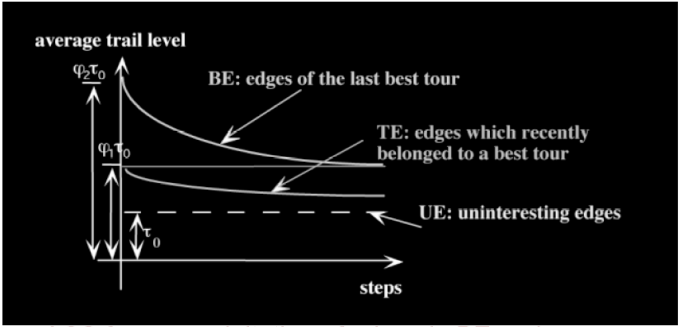
  $$\tau_{ij} := (1-\rho) \cdot \tau_{ij} + \sum_{s' \in sp'} \Delta_{ij}(s')$$

  where $\Delta_{ij}(s') := 1/g(s')$ (better $\Delta_{ij}(s') = \frac{C^{NN}}{m \cdot g(s')}$)
  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.

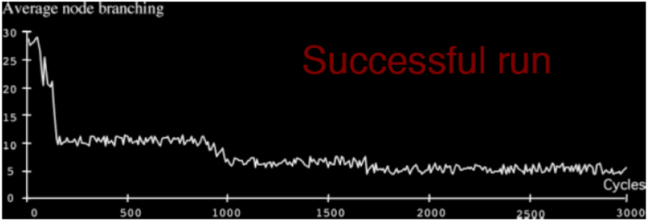
- *Termination:* After fixed number of cycles
  ($=$ construction $+$ perturbative search phases).

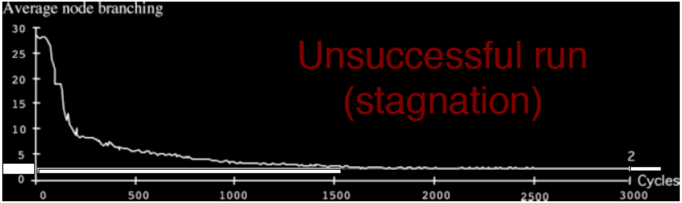
How does ACO work?



ACS favors exploitation of edges in BE and exploration of edges in TE

Analysis
Evolution of Avg Node Branching

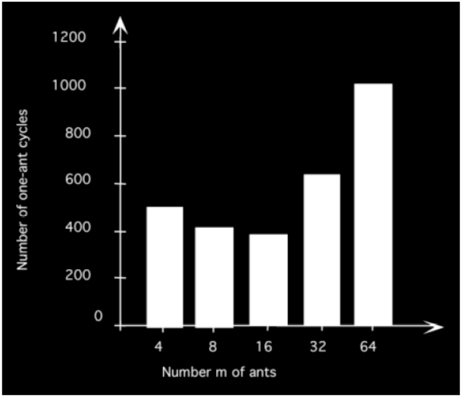




Analysis

How Many Ants?

In general, it was found that a good value for the number of ants is equal to the number of



Number of tours generated to find the optimal solution as a function of the number m of ants used

Analysis

Other things to check

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

ACO Variants

- 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)

Max-Min AS (MMAS)

- Only iteration best or best-so-far ants can add pheromone
- Pheromone trails have explicit upper and lower limits
- Pheromone trail initialized to upper limit
- Pheromone trail are re-initialized when stagnation
- Results obtained are better than AS, EAS, and ASrank, and of similar quality to ACS's

ANTS

- 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

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

- Different pheromone trail update rule

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

Ant Colony System (ACS)

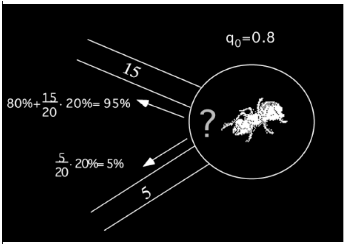
Three main ideas:

- Different state transition rule
- Different global pheromone trail update rule
- New local pheromone trail update rule

ACS's State Transition Rule

Next state:

- with probability q_0
  - exploitation
- with probability $(1 - q_0)$
  - biased



ACS's State Transition Rule

$$j = \begin{cases} \arg\max_{j \in J_i^k} \left\{ \left[\tau_{ij}(t)\right] \cdot \left[\eta_{ij}\right]^{\beta} \right\} & \text{if } q \le q_0 \quad \text{(Exploitation)} \\ J & \text{otherwise} \quad \text{(Exploration)} \end{cases}$$

where J is a stochastic variable distributed as follows:

$$p_{ij}^k(t) = \frac{\left[\tau_{ij}(t)\right] \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{l \in J_i^k} \left[\tau_{il}(t)\right] \cdot \left[\eta_{il}\right]^{\beta}}$$

and β and q_0 are parameters

The ACS Algorithm

Loop
Randomly position #ants ants on #cities cities
For step=1 to #cities
For k=1 to #ants
Apply the **state transition rule**
Apply the **online trail updating rule**
End-for
End-for
Apply the **offline trail updating rule**
Until End_condition

ACS's Offline Trail Updating

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\cdot\tau_{ij}(t) + \rho\cdot\Delta\tau_{ij}(t)_{Global}$$

where

$$\Delta\tau_{ij}(t)_{Global} = \frac{1}{L^+}$$

L^+ = best tour so far

Only edges belonging to the best tour so far are updated

ACS's Online Trail Updating

If an edge (*i*,*j*) is visited by an ant

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\cdot\tau_{ij}(t) + \rho\cdot\tau_0$$

Only edges visited by ants are updated

Ant Colony Optimisation . . .

- has been applied very successfully to a wide range of combinatorial problems. Check: `http://www.aco-metaheuristic.org/`
- underlies new high-performance algorithms for *dynamic optimisation problems*, such as routing in telecommunications networks

For further details on Ant Colony Optimisation, see the book by Dorigo and Stützle [2004].

ACO: Theoretical results

- 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
- 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 optima with prob 1
- Birattari et al. (TR, 2000; ANTS 2002) have shown the tight relationship between ACO and dynamic programming
- Zlochin et al. (TR, 2001) have shown the tight relationship between ACO and estimation of distribution algorithms

Application Examples

Linear permutations problems: SMTWTP

Construction graph:
Fully connected and the set of vertices consists of the n jobs and the n positions to which the jobs are assigned.

Constraints: all jobs have to be scheduled.

Pheromone Trails: τ_{ij} expresses the desirability of assigning job i in position j (cumulative rule)

Heuristic information: $\eta_{ij} = \frac{1}{h_i}$ where h_i is a dispatching rule.

Generalized Assignment Problem (GAP)

Input:

- a set of jobs $J = \{1, \ldots, n\}$ and a set of agents $I = \{1, \ldots, m\}$.
- the cost c_{ij} and the resource requirement a_{ij} of a job j assigned to agent i
- the amount b_i of resource available to agent i

Task: Find an assignment of jobs to tasks $\sigma : J \rightarrow I$ such that:

$$\begin{array}{ll} \min & f(\sigma) = \sum_{j \in J} c_{\sigma(j)j} \\ \text{s.t.} & \sum_{j \in J, \sigma(j) = i} a_{ij} \leq b_i \quad \forall i \in I \end{array}$$

Application Examples (2)

Assignment problems: Generalized Assignment Problem

Construction Graph:
a complete graph with vertices $I \cup J$ and costs on edges. An ant walk must then consist of n couplings (i, j).

(alternatively the graph is given by $E \times T$ and ants walk through the list of jobs choosing agents. An order must be decided for the jobs.)

Constraints:

- if only feasible: the capacity constraint can be enforced by restricting the neighborhood, ie, N_i^k for a ant k at job i contains only those agents where job i can be assigned.
- if also infeasible: then no restriction

Pheromone:

Two choices:

- which job to consider next
- which agent to assign to the job

Pheromone and heuristic on:

- desirability of considering job $i2$ after job $i1$
- desirability of assigning job i on agent j

Application Examples (3)

Subset problems: Set Covering

Construction graph:
Fully connected with set of vertices that corresponds to the set of columns plus a dummy vertex from where all the ants depart.

Constraints: each vertex can be visited at most once and all rows must be covered.

Pheromone Trails: associated with components (vertices); τ_j measures the desirability of including column j in solution.

Heuristic information: on the components as function of the ant's partial solution.
$\eta_j = \frac{e_j}{c_j}$ where e_j is the # of additional rows covered by j.

Connection between ACO and other Metaheuristics

Greedy Randomized "Adaptive" Search Procedure (GRASP):

While *termination criterion* is not satisfied:

- generate candidate solution s using
  subsidiary greedy randomized constructive search
- perform *subsidiary perturbative search* on s

Adaptive Iterated Construction Search:

initialise weights

While *termination criterion* is not satisfied:

- generate candidate solution s using
  subsidiary randomized constructive search
- perform *subsidiary perturbative search* on s
- *adapt weights* based on s

Squeaky Wheel:
Construct, Analyse, Prioritize

Iterated Greedy (IG):
destruct, reconstruct, acceptance criterion