Chapter 17: Ant Algorithms

Computational Intelligence: Second Edition

Contents

- About Ants
- Ant Colony Optimization Meta-Heuristic
- Cemetery Organization and Brood Care
- Division of Labor
- Advanced Topics

Some Facts

- Ants appeared on earth some 100 million years ago
- The total ant population is estimated at 10^{16} individuals [10]
- The total weight of ants is in the same order of magnitude as the total weight of human beings
- Most of these ants are social insects, living in colonies of 30 to millions of individuals
- Many studies of ant colonies have been done, to better understand the collective behaviors found in ant colonies:
  - foraging behavior,
  - division of labour,
  - cemetery organization and brood care, and
  - nest construction

The First Pioneers

- Eugéne Marais (1872-1936), one of the first to study termite colonies, published his work in *The Soul of the Ant* [12]
- Maurice Maeterlinck (1862–1949), published *The Life of the White Ant* [11], largely drawn from Marais's articles (see the discussion in [12])
- Pierre-Paul Grassé [8] (1959) postulated on the mechanics of termite communication in studies of their nest construction behavior
- Grassé determined that a form of indirect communication exists between individuals, which he termed *stigmergy*
- Deneubourg *et al.* [3] (1990) studied pheromonal communication as an example of stigmergy
- From these studies, Dorigo implemented the first algorithmic models of foraging behavior in 1992 [4]

Foraging Behavior of Ants

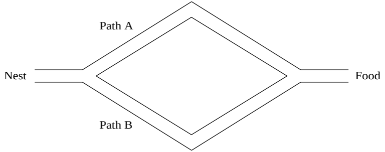
- How do ants find the shortest path between their nest and food source, without any visible, central, active coordination mechanisms?
- Initial studies of foraging behavior showed:
  - Initial random or chaotic activity pattern in the search for food
  - When food source is located, activity patterns become more organized
  - More and more ants follow the same path to the food source
  - Auto-magically, most ants follow the same, shortest path.
- This emergent behavior is the result of a recruitement mechanism, via pheromone trail following

Foraging Behavior of Ants (cont)

- **Autocatalytic behavior** – positive feedback:
  - Forager ants lay pheromones along followed trails
  - Paths with a larger pheromone concentration have a higher probability of being selected
  - As more ants follow a specific trail, the desirability of that path is reinforced by more pheromone being deposited
  - This attracts more ants to follow that path
- **Stigmergy**
  - Indirect communication where ants modify their environment by laying pheromones to influence the behavior of other ants

Foraging Behavior of Ants (cont)

- Bridge experiment (figure 17.1)



- Probability of the next ant to choose path A:

$$P_A(t+1) = \frac{(c + n_A(t))^\alpha}{(c + n_A(t))^\alpha + (c + n_B)^\alpha} = 1 - P_B(t+1)$$

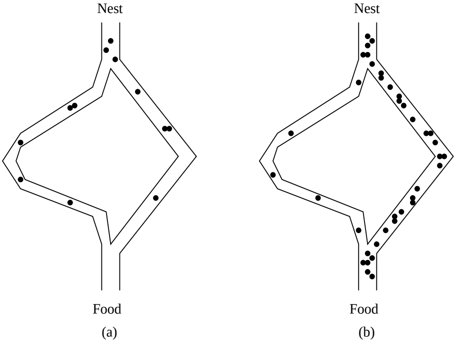
  - $n_A(t)$ and $n_B(t)$ are the number of ants on paths A and B respectively at time step t
  - c quantifies the degree of attraction of an unexplored branch
  - α biases towards pheromone deposits in the decision process

Foraging Behavior of Ants (cont)

- The larger the value of α, the higher the probability that the next ant will follow the path with a higher pheromone concentration
- The larger the value of c, the more pheromone deposits are required to make the choice of path non-random
- The **decision rule**:
  if $U(0,1) \leq P_A(t+1)$ *then follow path A otherwise follow path B*

Foraging Behavior of Ants (cont)

- **Extended binary bridge** (figure 17.2):



- **Differential pathlength effect**:
  - The probability of selecting the shorter path increases with the length ratio between the two paths

Foraging Behavior of Ants (cont)

- Each ant is a stimulus-response agent, following simple production rules (Algorithm 17.1):

Let $r \sim U(0,1)$;
for *each potential path* A **do**
    Calculate P_A using;
    Sort P_A in decreasing order;
    if $r \leq P_A$ **then**
        Follow path A;
        Break;
    end
end

Stigmergy and Artificial Pheromone

- Generally stated, stigmergy is a class of mechanisms that mediate animal-to-animal interactions [14]
- A form of indirect communication mediated by modifications to the environment
- Forms of stigmergy:
  - *Sematectonic stigmergy* refers to communication via changes in the physical characteristics of the environment – nest building, brood sorting
  - *Sign-based* stigmergy facilitates communication via a signaling mechanism, implemented via chemical compounds deposited by ants – foraging

Artificial Stigmergy

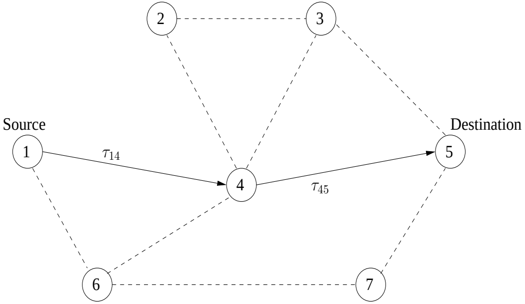
- *The indirect communication mediated by numeric modifications of environmental states which are only locally accessible by the communicating agents* [5]
- The essence of modeling ant behavior is to find a mathematical model that accurately describes the stigmergetic characteristics of the corresponding ant individuals
- Define stimergic variables which encapsulate the information used by artificial ants to communicate indirectly:
    - for foraging behavior, artificial pheromone

Ant Algorithms

- Ant algorithms are population-based systems inspired by observations of real ant colonies
- Cooperation among individuals in an ant algorithm is achieved by exploiting the stigmergic communication mechanisms observed in real ant colonies
- Foraging-based ant algorithms are generally referred to as *Ant Colony Optimization Meta-Heuristics*

Simple Ant Colony Optimization (SACO)

- An algorithmic implementation of the binary bridge experiment
- Consider the general problem of finding a shortest path between two nodes (figure 17.3):



Simple Ant Colony Optimization (Algorithm 17.2)

Initialize $\tau_{ij}(0)$ to small random values and let $t = 0$;
Place n_k ants on the origin node;
repeat
    for *each ant* $k = 1, \ldots, n_k$ **do**
        Construct a path $x^k(t)$;
    end
    for *each link* (i, j) *of the graph* **do**
        Pheromone evaporation;
    end
    for *each ant* $k = 1, \ldots, n_k$ **do**
        for *each link* (i, j) *of* $x^k(t)$ **do**
            Update τ_{ij};
        end
    end
    $t = t + 1$;
until *stopping condition is true* ;

SACO: Path Construction

- For each iteration, each ant incrementally constructs a path (solution)
- At node i, ant k selects the next node $j \in \mathcal{N}_i^k$, based on the transition probability:

$$p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)}{\sum_{j \in \mathcal{N}_i^k} \tau_{ij}^{\alpha}(t)} & \text{if } j \in \mathcal{N}_i^k \\ 0 & \text{if } j \notin \mathcal{N}_i^k \end{cases}$$

  where $\mathcal{N}_i^k$ is the set of feasible nodes connected to node i, with respect to ant k
- If $\mathcal{N}_i^k = \emptyset$, the predecessor to node i is included in $\mathcal{N}_i^k$
- Loops are removed once the destination node has been reached

SACO: Pheromone Evaporation

- To improve exploration abilities, and to prevent premature convergence:

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

  with $\rho \in [0, 1]$
- ρ specifies the rate at which pheromones evaporate, causing ants to "forget" previous decisions
- ρ controls the influence of search history
- For large values of ρ, pheromone evaporates rapidly, while small values of ρ result in slower evaporation rates
- Large values therefore implies more exploration, more random search

SACO: Pheromone Update

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \sum_{k=1}^{n_k} \Delta\tau_{ij}^k(t)$$

where

$$\Delta\tau_{ij}^k(t) = \frac{1}{L^k(t)}$$

$L^k(t)$ is the length of the path constructed by ant k at time step t
n_k is the number of ants

SACO: Important Notes

- Solution construction is the result of cooperative behavior that emerges from the simple behaviors of individual ants
- Each ant chooses the next link of its path based on information provided by other ants, in the form of pheromone deposits, referring to the autocatalytic behavior exhibited by forager ants
- The information used to aid in the decision making process is limited to the local environment of the ant

Ant System (AS)

- First ant colony optimization (ACO) algorithm developed by Dorigo [4]
- Transition probability:

$$p_{ij}^{k}(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{u \in \mathcal{N}_i^k(t)} \tau_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)} & \text{if } j \in \mathcal{N}_i^k(t) \\ 0 & \text{if } j \notin \mathcal{N}_i^k(t) \end{cases}$$

where
τ_{ij} represents the *a posteriori* effectiveness of the move from node i to node j
η_{ij} represents the *a priori* effectiveness of the move from i to j – desirability of the move

AS: Exploration–Explotation Trade-off

- A balance between pheromone intensity, τ_{ij}, and heuristic information, η_{ij}
- If $\alpha = 0$:
  - No pheromone information is used, i.e. previous search experience is neglected
  - The search then degrades to a stochastic greedy search
- If $\beta = 0$:
  - The attractiveness of moves is neglected
  - The search algorithm is similar to SACO
- Heuristic information adds an explicit bias towards the most attractive solutions, e.g.

$$\eta_{ij} = \frac{1}{d_{ij}}$$

AS: Pheromone Updates

- Three variations in the way pheromone deposits are calculated
- Ant-cycle AS:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{Q}{f(x^k(t))} & \text{if link } (i,j) \text{ occurs in path } x^k(t) \\ 0 & \text{otherwise} \end{cases}$$

- Ant-density AS:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} Q & \text{if link } (i,j) \text{ occurs in path } x^k(t) \\ 0 & \text{otherwise} \end{cases}$$

- Ant-quantity AS:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{Q}{d_{ij}} & \text{if link } (i,j) \text{ occurs in path } x^k(t) \\ 0 & \text{otherwise} \end{cases}$$

AS: Algorithm

$t = 0$ and initialize all parameters, i.e. $\alpha, \beta, \rho, Q, n_k, \tau_0$;
Place all ants, $k = 1, \ldots, n_k$ and Initialize pheromone on each link;
repeat
  for *each ant* $k = 1, \ldots, n_k$ **do**
    Construct a path, $x^k(t)$;
    Compute $f(x^k(t))$;
  end
  for *each link* (i, j) **do**
    Apply pheromone evaporation, then update pheromone;
    $\tau_{ij}(t+1) = \tau_{ij}(t)$;
  end
  $t = t + 1$;
until *stopping condition is true* ;
Return $x^k(t) : f(x^k(t)) = \min_{k'=1,\ldots,n_k}\{f(x^{k'}(t))\}$;

AS: Elitist Pheromone Updates

- The best ants add pheromone proportional to quality of their paths

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \Delta\tau_{ij}(t) + n_e \Delta\tau_{ij}^{e}(t)$$

  where

$$\Delta\tau_{ij}^{e}(t) = \begin{cases} \frac{Q}{f(\tilde{x}(t))} & \text{if } (i,j) \in \tilde{x}(t) \\ 0 & \text{otherwise} \end{cases}$$

  e is the number of elite ants
  $\tilde{x}(t)$ is the current best route
- Objective is to direct the search of all ants to construct a solution to contain links of the current best route(s)

Ant Colony System (ACS)

- ACS differs from AS in four aspects:
  1. a different transition rule is used,
  2. a different pheromone update rule is defined,
  3. local pheromone updates are introduced, and
  4. candidate lists are used to favor specific nodes

ACS: Transition Rule

- The pseudo-random-proportional action rule:

$$j = \begin{cases} \arg\max_{u \in \mathcal{N}_i^k(t)} \{\tau_{iu}(t)\eta_{iu}^{\beta}(t)\} & \text{if } r \leq r_0 \\ J & \text{if } r > r_0 \end{cases}$$

where $r \sim U(0,1)$, and $r_0 \in [0,1]$ is a user-specified parameter

- $J \in \mathcal{N}_i^k(t)$ is a node randomly selected according to probability

$$p_{iJ}^k(t) = \frac{\tau_{iJ}(t)\eta_{iJ}^{\beta}(t)}{\sum_{u \in \mathcal{N}_i^k} \tau_{iu}(t)\eta_{iu}^{\beta}(t)}$$

$\mathcal{N}_i^k(t)$ is a set of valid nodes to visit

ACS: Exploration–Exploitation Trade-off

- Transition rule creates a bias towards nodes connected by short links and with a large amount of pheromone
- Parameter r_0 is used to balance exploration and exploitation:
  - if $r \leq r_0$, the algorithm exploits by favoring the best edge
  - if $r > r_0$, the algorithm explores
  - the smaller the value of r_0, the less best links are exploited, while exploration is emphasized more
- The transition rule is the same as that of AS when $r > r_0$

ACS: Pheromone Update

- Global update rule:
  - Only the globally best ant, $x^+(t)$, is allowed to reinforce pheromone concentrations on the links of the corresponding best path

    $$\tau_{ij}(t+1) = (1-\rho_1)\tau_{ij}(t) + \rho_1 \Delta\tau_{ij}(t)$$

    where

    $$\Delta\tau_{ij}(t) = \begin{cases} \frac{1}{f(x^+(t))} & \text{if } (i,j) \in x^+(t) \\ 0 & \text{otherwise} \end{cases}$$

    with $f(x^+(t)) = |x^+(t)|$, in the case of finding shortest paths
  - Favors exploitation
  - $x^+(t)$ as the iteration-best vs global-best

ACS: Pheromone Update (cont)

- Pheromone evaporation:
  - For small values of ρ_1, the existing pheromone concentrations on links evaporate slowly, while the influence of the best route is dampened
  - For large values of ρ_1, previous pheromone deposits evaporate rapidly, but the influence of the best path is emphasized
  - The effect of large ρ_1 is that previous experience is neglected in favor of more recent experiences – more exploration
  - If ρ_1 is adjusted dynamically from large to small values exploration is favored in the initial iterations of the search, while focusing on exploiting the best found paths in the later iterations

ACS: Pheromone Update (cont)

- Local update rule:
  - Applied by each ant as soon as a new link is added to the path:

$$\tau_{ij}(t) = (1 - \rho_2)\tau_{ij}(t) + \rho_2 \tau_0$$

with ρ_2 also in $(0, 1)$, and τ_0 is a small positive constant

ACS: Neighborhood Set

- $\mathcal{N}_i^k(t)$ is organized to contain a list of candidate nodes
- Candidate nodes are preferred nodes, to be visited first
- Let $n_l < |\mathcal{N}_i^k(t)|$ denote the number of nodes in the candidate list
- The n_l nodes closest to node i, i.t.o. cost, are included in the candidate list and ordered by increasing distance
- When a next node is selected, the best node in the candidate list is selected
- If the candidate list is empty, then node j is selected from the remainder of $\mathcal{N}_i^k(t)$

Max-Min AS (MMAS)

- AS was shown to prematurely stagnate for complext problems:
  - All ants follow exactly the same path
  - Occurs when ants explore little, and too rapidly exploit highest pheromone concentrations
- MMAS was developed to address the premature stagnation problem of AS
- Differences between MMAS and AS:
  - MMAS restricts pheromone intensities within given intervals
  - Only the best ant may reinforce pheromones
  - Initial pheromones are set to the max allowed value
  - A pheromone smoothing mechanism is used

MMAS: Pheromone Update

- Global update is similar to that of ACS
  - If based on only the global-best path, may exploit too much
  - If based on only the iteration-best, more exploration
  - Used mixed strategies
  - At point of stagnation, all τ_{ij} are initialized to max value, after which interation-best is applied for a number of iterations.
- Point of stagnation:

$$\frac{\sum_{i \in V} \lambda_i}{n_G} < \epsilon, \quad \epsilon > 0$$

where λ_i is the number of links leaving node i with τ_{ij}-values greater than $\lambda\delta_i + \tau_{i,min}$; $\delta_i = \tau_{i,max} - \tau_{i,min}$

$$\tau_{i,min} = \min_{j \epsilon \mathcal{N}_i}\{\tau_{ij}\}$$

$$\tau_{i,max} = \max_{j \epsilon \mathcal{N}_i}\{\tau_{ij}\}$$

MMAS: Pheromone Update

- Clamping of pheromone:
  - If after application of the global update rule $\tau_{ij}(t+1) > \tau_{max}$, $\tau_{ij}(t+1)$ is explicitly set equal to τ_{max}
  - If $\tau_{ij}(t+1) < \tau_{min}$, $\tau_{ij}(t+1)$ is set to τ_{min}
  - Upper bound helps to avoid stagnation. How?
  - What is the advantage of having a lower pheromone limit?
- Local update, applied by each ant after adding a new link to the path:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \Delta\tau_{ij}(t)$$

MMAS: Pheromone Smoothing

- Stagnation still occurred, due to large differences between min and max pheromones
- Smoothing strategy used to reduce the differences between high and low pheromone concentrations
- At point of stagnation, all pheromone concentrations are increased proportional to the difference with the maximum bound:

$$\Delta\tau_{ij}(t) \propto (\tau_{max}(t) - \tau_{ij}(t))$$

- Stronger pheromone concentrations are proportionally less reinforced than weaker concentrations
- Increases the chance of links with low pheromone intensity to be selected as part of a path, and thereby increases the exploration abilities of the algorithm

Ant-Q (AQ)

- Local update rule is inspired by Q-learning
- The pheromone notion is dropped to be replaced by Ant-Q values (AQ-values)
- The goal of Ant-Q is to learn AQ-values such that the discovery of good solutions is favored in probability
- The AQ-values express how useful it is to move to node j from the current node i
- Transition rule:
  - Let $\mu_{ij}(t)$ denote the AQ-value on the link between nodes i and j at time step t. Then,

$$j = \begin{cases} \arg\max_{u \in \mathcal{N}_i^k(t)} \{\mu_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)\} & \text{if } r \leq r_0 \\ J & \text{otherwise} \end{cases}$$

AQ (cont)

- Different ways of selecting J:
  - *pseudo-random* action choice rule – J is a node randomly selected from the set $\mathcal{N}_i^k(t)$ according to the uniform distribution
  - *pseudo-random-proportional* action choice rule – $J \in V$ is selected according to the distribution

$$p_{ij}^k(t) = \begin{cases} \frac{\mu_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{u \in \mathcal{N}_i^k(t)} \mu_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)} & \text{if } j \in \mathcal{N}_i^k(t) \\ 0 & \text{otherwise} \end{cases}$$

  - *random-proportional* action choice rule – $r_0 = 0$

AQ: Learning Q-Values

- Local update rule:

$$\mu_{ij}(t+1) = (1-\rho)\mu_{ij}(t) + \rho\left(\Delta\mu_{ij}(t) + \gamma \max_{u \in \mathcal{N}_j^k(t)} \{\mu_{ju}(t)\}\right)$$

ρ is referred to as the discount factor
γ is the learning step size

  - Applied for each ant k after each new node j has been selected, but with $\Delta\mu_{ij}(t) = 0$
  - AQ-value associated with link (i, j) is reduced by a factor of $(1-\rho)$ each time that the link is selected to form part of a candidate solution
  - AQ-value is reinforced with an amount proportional to the AQ-value of the best link, (j, u), leaving node j

AQ: Learning Q-Values (cont)

- Global update rule – delayed reinforcement:
  - Applied after all ants have constructed a candidate solution
  - Local update rule is applied, but with

$$\Delta\mu_{ij}(t) = \begin{cases} \frac{1}{f(x^+(t))} & \text{if } (i,j) \in x^+(t) \\ 0 & \text{otherwise} \end{cases}$$

where $x^+(t)$ is either the iteration-best or global-best

Fast Ant System (FANT)

- Main differences between FANT and other ACO algorithms:
  1. FANT uses only one ant
  2. A different pheromone update rule is applied which does not make use of any evaporation
- Uses the AS transition rule, but with $\beta = 0$
- Pheromone update rule:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + w_1 \Delta\tilde{\tau}_{ij}(t) + w_2 \Delta\hat{\tau}^{+}_{ij}(t)$$

$$\Delta\tilde{\tau}_{ij}(t) = \begin{cases} 1 & \text{if } (i,j) \in \tilde{x}(t) \\ 0 & \text{otherwise} \end{cases}$$

$$\Delta\hat{\tau}_{ij}(t) = \begin{cases} 1 & \text{if } (i,j) \in \hat{x}(t) \\ 0 & \text{otherwise} \end{cases}$$

w_1 and w_2 determine the relative reinforcement by the current best solution and the best solution found so far

FANT: Pheromone Initialization

- All $\tau_{ij}(0) = 1$
- When a new $\hat{x}(t)$ is obtained, all pheromones are reinitialized to $\tau_{ij}(t) = 1$
- This step exploits the search area around the global best path, $\hat{x}(t)$
- Balancing exploration–exploitation?

Antabu

- Adapts AS to include a local search using tabu search
- Global update rule is changed such that each ant's pheromone deposit on each link of its constructed path is proportional to the quality of the path:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \left(\frac{\rho}{f(x^k(t))}\right)\left(\frac{f(x^-(t)) - f(x^k(t))}{f(\hat{x}(t))}\right)$$

$f(x^-(t))$ is the cost of the worst path found so far
$f(\hat{x}(t))$ is the cost of the best path found so far
$f(x^k(t))$ is the cost of the path found by ant k

AS-rank

- A modification of AS to:
  1. allow only the best ant to update pheromone concentrations on the links of the global-best path,
  2. to use elitist ants, and
  3. to let ants update pheromone on the basis of a ranking of the ants

AS-rank: Pheromone Update

- Global update rule changes to

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + n_e\Delta\hat{\tau}_{ij}(t) + \Delta\tau^r_{ij}(t), \quad \Delta\hat{\tau}_{ij}(t) = \frac{Q}{f(\hat{x}(t))}$$

with $\hat{x}(t)$ the best path constructed so far

$$\Delta\tau^r_{ij}(t) = \sum_{\sigma=1}^{n_e} \Delta\tau^\sigma_{ij}(t)$$

$$\Delta\tau^\sigma_{ij}(t) = \begin{cases} \frac{(n_e-\sigma)Q}{f(x^\sigma(t))} & \text{if } (i,j) \in x^\sigma(t) \\ 0 & \text{otherwise} \end{cases}$$

and $f(x^1(t)) \leq f(x^2(t)) \leq \ldots \leq f(x^{n_k}(t))$
σ is the rank of the corresponding ant

ANTS

- Uses the transition probability of AS, but with

$$\Delta\tau_{ij}^{k}(t) = \tau_0 \left(1 - \frac{f(x^k(t)) - \epsilon}{\overline{f}(t) - \epsilon}\right)$$

where $\overline{f}(t)$ is a moving average on the cost of the last $\hat{n}_t$ global-best solutions found by the algorithm

Cemetery Organization and Brood Care: Real Ants

- Cemetery organization:
  - Many ant species exhibit the behavior of clustering corpses to form cemeteries [2]
  - Each ant seems to behave individually, moving randomly in space while picking up or depositing (dropping) corpses
  - The decision to pick up or drop a corpse is based on local information of the ant's current position
  - This very simple behavior of individual ants results in the emergence of a more complex behavior of cluster formation
- Brood care:
  - Larvae are sorted in such a way that different brood stages are arranged in concentric rings
  - Smaller larvae are located in the center, with larger larvae on the periphery

Basic Ant Clustering Algorithm

- The local behaviors:
  - Items in less dense areas should be picked up
  - Items should be dropped in a different location where more of the same type exist
- Assumptions:
  - Only one type of item
  - Items are randomly distributed on a two-dimensional grid
  - Each grid-point contains only one item
  - Ants are placed randomly on the lattice
  - Ants move in random directions one cell at a time

Basic Ant Clustering Algorithm (cont)

- An unladen ant decides to pick up an item based on the probability

  $$P_p = \left(\frac{\gamma_1}{\gamma_1 + \lambda}\right)^2$$

  where λ is the fraction of items the ant perceives in its neighborhood

  $\gamma_1 > 0$

- Each loaded ant has a probability of dropping the carried object, given by

  $$P_d = \left(\frac{\lambda}{\gamma_2 + \lambda}\right)^2$$

  provided that the corresponding cell is empty

  $\gamma_2 > 0$

Basic Ant Clustering Algorithm (cont)

- Calculating item frequency:
  - Each ant keeps track of the last T time steps
  - Then,

$$\lambda = n_\lambda / T$$

  where n_λ is the number of encountered items
- Extended to more than one item type:
  - Let A and B denote two types of items
  - Use the same dropping and pick-up probabilities as above, but with λ replaced by λ_A or λ_B depending on the type of item encountered

Lumer-Faieta Algorithm

- A generalized model to cluster data vectors with real-valued elements
- Measure of similarity, $d(\mathbf{y}_a, \mathbf{y}_b)$ – Euclidean distance
- How should items be grouped such that
  - intra-cluster distances are minimized; that is, the distances between data vectors within a cluster should be small to form a compact, condensed cluster, and
  - inter-cluster distances are maximized; that is, the different clusters should be well separated

Lumer-Faieta Algorithm: Density Estimation

- Local density estimation with respect to vector $\mathbf{y}_a$:

$$\lambda(\mathbf{y}_a) = \max\left\{0, \frac{1}{n_{\mathcal{N}}^2} \sum_{\mathbf{y}_b \in \mathcal{N}_{n_{\mathcal{N}} \times n_{\mathcal{N}}}(i)} \left(1 - \frac{d(\mathbf{y}_a, \mathbf{y}_b)}{\gamma}\right)\right\}$$

  where $\gamma > 0$ defines the scale of dissimilarity between items $\mathbf{y}_a$ and $\mathbf{y}_b$
- The constant γ determines when two items should, or should not be located next to each other:
  - If γ is too large, it results in the fusion of individual clusters, clustering items together that do not belong together
  - If γ is too small, many small clusters are formed
  - γ has a direct influence on the number of clusters formed

Lumer-Faieta Algorithm: The Probabilities

- Pick-up probability:

$$P_p(\mathbf{y}_a) = \left(\frac{\gamma_1}{\gamma_1 + \lambda(\mathbf{y}_a)}\right)^2$$

- Drop probability:

$$P_d(\mathbf{y}_a) = \begin{cases} 2\lambda(\mathbf{y}_a) & \text{if } \lambda(\mathbf{y}_a) < \gamma_2 \\ 1 & \text{if } \lambda(\mathbf{y}_a) \geq \gamma_2 \end{cases}$$

Lumer-Faieta Algorithm 17.7

Place each $\mathbf{y}_a$ randomly on a cell, and n_k ants on randomly selected sites;
Initialize values of $\gamma_1, \gamma_2, \gamma$ and n_t;
for $t = 1$ *to* n_t **do**
  for *each ant,* $k = 1, \cdots, n_k$ **do**
    if *ant* k *is unladen and the site is occupied by item* $\mathbf{y}_a$ **then**
      Compute $\lambda(\mathbf{y}_a)$ and $P_p(\mathbf{y}_a)$;
      if $U(0,1) \leq P_p(\mathbf{y}_a)$ **then** Pick up item $\mathbf{y}_a$;
    end
    else
      if *ant* k *carries item* $\mathbf{y}_a$ *and site is empty* **then**
        Compute $\lambda(\mathbf{y}_a)$ and $P_d(\mathbf{y}_a)$;
        if $U(0,1) \leq P_d(\mathbf{y}_a)$ **then** Drop item $\mathbf{y}_a$;
      end
    end
    Move to a randomly selected unoccupied neighboring site;
  end

Lumer-Faieta Algorithm: Aspects to Consider

- The grid size:
  - There should be more sites than data vectors, since items are not stacked
  - What will happen if the number of cites is approximately the same as the number of vectors?
- The number of ants:
  - There should be fewer ants than data vectors
  - What will happen if there are more ants than data vectors?
- Patch size, $n_{\mathcal{N}}$:
  - What will happen for large vs small patch sizes?

Modifications to Lumer-Faieta Algorithm

- Different moving speeds:
  - Fast-moving ants form coarser clusters by being less selective in their estimation of the average similarity of a data vector to its neighbors
  - Slower agents are more accurate in refining the cluster boundaries

$$\lambda(\mathbf{y}_a) = \max\left\{0, \frac{1}{n_{\mathcal{N}}^2} \sum_{\mathbf{y}_b \in \mathcal{N}_{n_{\mathcal{N}} \times n_{\mathcal{N}}}(i)} \left(1 - \frac{d(\mathbf{y}_a, \mathbf{y}_b)}{\gamma(1 - \frac{v-1}{v_{max}})}\right)\right\}$$

$v \sim U(1, v_{max})$ and v_{max} is the maximum moving speed

Modifications to Lumer-Faieta Algorithm (cont)

- Short-term memory:
  - Remembers where a limited number of carried items have been dropped
  - Position of the best matching memorized data item biases the direction of the ant's walk
  - Helps to group together similar items
- Behavioral switches:
  - Ants are not allowed to start destroying clusters if they have not performed an action for a number of time steps
  - Allows the algorithm to escape from local optima

Modifications to Lumer-Faieta Algorithm (cont)

- Distance/Dissimilarity Measures:
  - For floating-point vectors, the Euclidean distance:

$$d_E(\mathbf{y}_a, \mathbf{y}_b) = \sqrt{\sum_{l=1}^{n_y} (y_{al} - y_{bl})^2}$$

  - Alternative is the cosine similarity:

$$d_C(\mathbf{y}_a, \mathbf{y}_b) = 1 - \text{sim}(\mathbf{y}_a, \mathbf{y}_b)$$

where

$$\text{sim}(\mathbf{y}_a, \mathbf{y}_b) = \frac{\sum_{l=1}^{n_y} y_{al} y_{bl}}{\sqrt{\sum_{l=1}^{n_y} y_{al}^2 \sum_{l=1}^{n_y} y_{bl}^2}}$$

As $\mathbf{y}_a$ and $\mathbf{y}_b$ become more similar, $\text{sim}(\mathbf{y}_a, \mathbf{y}_b)$ approaches 1.0, and $d_C(\mathbf{y}_a, \mathbf{y}_b)$ becomes 0

Modifications to Lumer-Faieta Algorithm (cont)

- Pick-up and dropping probabilities:
  - Yang and Kamel [16]:

$$P_p = 1 - f_s(\lambda(\mathbf{y}_a)), \quad P_d = f_s(\lambda(\mathbf{y}_a))$$

    where f_s is the sigmoid function
  - Handl *et al.* [9]:

$$P_p = \begin{cases} 1 & \text{if } \lambda(\mathbf{y}_a) \leq 1 \\ \frac{1}{\lambda(\mathbf{y}_a)^2} & \text{otherwise} \end{cases}$$

$$P_d = \begin{cases} 1 & \text{if } \lambda(\mathbf{y}_a) > 1 \\ \lambda(\mathbf{y}_a)^4 & \text{otherwise} \end{cases}$$

Modifications to Lumer-Faieta Algorithm (cont)

- Heterogeneous ants:
  - Ants with different behaviors
  - Different values for $n_{\mathcal{N}}$ and γ for each ant
  - For each ant select values for these parameters randomly within defined ranges
  - Start with small $n_{\mathcal{N}}$, increasing over time, to prevent large numbers of small clusters to form

Division of Labor: In Insect Colonies

- Division of labor occurs in biological systems when all the individuals of that system are co-adapted through divergent specialization, in such a way that there is a fitness gain as a consequence of such specialization [13]
- Within insect colonies, a number of tasks are done, including reproduction, caring for the young, foraging, cemetery organization, waste disposal, and defense
- Task allocation and coordination occur mostly without any central control
- Individuals respond to simple local cues, e.g. the pattern of interactions with other individuals [7], or chemical signals
- Task allocation is dynamic
- Task switching occurs when environmental conditions demand such switches

In Insect Colonies (cont)

- **Temporal polyethism** (age subcaste):
  - Individuals of the same age tend to perform the same tasks, and form an age caste
  - As an example, in honey bee colonies, younger bees tend to the hive while older bees forage
- **Worker polymorphism**:
  - Workers have different morphologies
  - Workers of the same morphological structure belong to the same morphological caste, and tend to perform the same tasks
  - For example, minor ants care for the brood, while major ants forage
- **Individual variability**:
  - Even for individuals in an age or morphological caste, differences may occur among individuals in the frequency and sequence of task performance

Task Allocation Based on Response Thresholds

- Response thresholds refer to the likelihood of reacting to task-associated stimuli
- Individuals with a low threshold perform a task at a lower level of stimulus than individuals with high thresholds
- Individuals become engaged in a specific task when the level of task-associated stimuli exceeds their thresholds
- If a task is not performed by individuals, the intensity of the corresponding stimulus increases
- Intensity decreases as more ants perform the task
- The task-associated stimuli serve as stigmergic variable

Single Task Allocation

- Let s_j be the intensity of task-j-associated stimuli
- A response threshold, θ_{kj}, determines the tendency of individual k to respond to the stimulus, s_j, associated with task j
- Individual k engages in task j with probability [1, 6, 15]

$$P_{\theta_{kj}}(s_j) = \frac{s_j^\omega}{s_j^\omega + \theta_{kj}^\omega}$$

  where $\omega > 1$ determines the steepness of the threshold
- For $s_j << \theta_{kj}$, $P_{\theta_{kj}}(s_j)$ is close to zero, and the probability of performing task j is very small
- For $s_j >> \theta_{kj}$, the probability of performing task j is close to one

Single Task Allocation (cont)

- Assume only one task
- The probability that an inactive ant will become active

$$P(\vartheta_k = 0 \rightarrow \vartheta_k = 1) = \frac{s^2}{s^2 + \theta_k^2}$$

$\vartheta_k = 0$ indicates that ant k is inactive

$\vartheta_k = 1$ indicates that the ant is performing the task

- An active ant spends an average $1/p$ time performing the task
- Change in stimulus intensity:

$$s(t+1) = s(t) + \sigma - \gamma n_{act}$$

σ is the increase in demand

γ is the decrease associated with one ant performing the task

n_{act} is the number of active ants

Single Task Allocation (cont)

- The more ants engaged in the task, the smaller the intensity, s, and consequently, the smaller the probability that an inactive ant will take up the task
- If there are not enough ants busy with the task (i.e. $\sigma > \gamma n_{act}$), the probability increases that inactive ants will participate in the task
- Multiple tasks:
    - Let there be n_j tasks
    - Let n_{kj} be the number of workers of caste k performing task j
    - Each individual has a vector of thresholds, θ_k
    - After $1/p$ time units of performing task j, the ant stops with this task, and selects another task

Adaptive Task Allocation

- Thresholds are adapted using a simple reinforcement mechanism
  - A threshold decreases when the corresponding task is performed
  - A threshold increases when the task is not performed
- Let ξ and ϕ respectively represent the learning coefficient and the forgetting coefficient
- If ant k performs task j in the next time unit, then

$$\theta_{kj}(t+1) = \theta_{kj}(t) - \xi$$

- If ant k does not perform task j, then

$$\theta_{kj}(t+1) = \theta_{kj}(t) + \phi$$

Adaptive Task Allocation (cont)

- If t_{kj} is the fraction of time that ant k spent on task j, then it spent $1 - t_{kj}$ time on other tasks:

$$\theta_{kj}(t+1) = \theta_{kj}(t) - t_{kj}\xi + (1 - t_{kj})\phi$$

- The more ant k performs task j, the smaller θ_{kj} becomes
- The smaller θ_{kj} becomes, more more specialized the ant becomes in performing task j

Advanced Topics

- The ACO-MH was developed to solve uni-objective, static optimization problems where the search space can be represented by a graph – thus, discrete optimization problems
- How can the ACO-MH be adapted to be applied to
  - continuous-valued search domains?
  - multi-objective optimization problems?
  - dynamic environments?
  - constrained problems?

Continuous Ant Colony Optimization (CACO)

- The ACO-MH was originally developed to solve discrete optimization problems, where the values assigned to variables of the solution are constrained by a fixed finite set of discrete values
- How can the ACO-MH be applied to continuous optimization problems?
    - Need to find a way to map the continuous space problem to a graph search problem
- CACO performs a bi-level search:
    - A local search component to exploit good regions
    - A global search component to explore bad regions

CACO: Algorithm 17.8

Create n_r regions and set $\tau_i(0) = 1, \; i = 1, \ldots, n_r$;
repeat
    Evaluate fitness, $f(\mathbf{x}_i)$, of each region;
    Sort regions in descending order of fitness;
    Send 90% of n_g global ants for crossover and mutation;
    Send 10% of n_g global ants for trail diffusion;
    Update pheromone and age of n_g weak regions;
    Send n_l ants to probabilistically chosen good regions;
    Do local search to exploit good region;
until *stopping condition is true* ;
Return region $\mathbf{x}_i$ with best fitness as solution;

CACO: Algorithm 17.8 – Local Search

```
for each local ant do
    if region with improved fitness is found then
        Move ant to better region;
        Update pheromone;
    end
    else
        Increase age of region;
        Choose new random direction;
    end
    Evaporate all pheromone;
end
```

CACO: Initialization

- Search is performed by n_k ants:
  - n_l ants perform local search in good area
  - n_g ants perform global search to explore
- Initialization: Create n_r regions
  - Each region represents a point in the continuous search space
  - If $\mathbf{x}_i$ denotes the i-th region, then $x_{ij} \sim U(x_{min,j}, x_{max,j})$

  Set pheromone for each region to 1
- Fitness is evaluated using the continuous function being optimized

CACO: Global Search

- Most ants perform crossover to produce new regions:
  - For each variable x'_{ij} of the offspring, choose a random weak region $\mathbf{x}_i$
  - Let $x'_{ij} = x_{ij}$, with probability P_c
  - Mutate the offspring $\mathbf{x}'_i$

$$x'_{ij} \leftarrow x'_{ij} + N(0, \sigma^2), \quad \sigma \leftarrow \sigma_{max}(1 - r^{(1-t/n_t)^{\gamma_1}})$$

    where $r \sim U(0,1)$, σ_{max} is the maximum step size, t is the current time step, n_t is the maximum number of iterations, and γ_1 controls the degree of nonlinearity
- Trail diffusion using arithmetic crossover:
  - Randomly select two weak regions as parents, $\mathbf{x}_i$ and $\mathbf{x}_l$, then

$$x'_j = \gamma_2 x_{ij} + (1 - \gamma_2) x_{lj}$$

    where $\gamma_2 \sim U(0,1)$

CACO: Local Search

- Each ant k of the n_l local ants selects a region $\mathbf{x}_i$ based on the probability

$$p_i^k(t) = \frac{\tau_i^{\alpha}(t)\eta_i^{\beta}(t)}{\sum_{j \in \mathcal{N}_i^k} \tau_j^{\alpha}(t)\eta^{\beta}(t)}$$

- Selection biases towards good regions
- The ant moves a distance away from the selected region, $\mathbf{x}_i$, to a new region, $\mathbf{x}_i^{'}$, using

$$\mathbf{x}_i^{'} = \mathbf{x}_i + \Delta\mathbf{x}$$

where $\Delta x_{ij} = c_i - m_i a_i$ with c_i and m_i user-defined parameters, and a_i the age of region $\mathbf{x}_i$

CACO: Region Age & Pheromone Update

- The age of a region indicates the "weakness" of the corresponding solution
- If $\mathbf{x}_i^{'}$, does not have a better fitness than $\mathbf{x}_i$, the age of $\mathbf{x}_i$ is incremented
- If $\mathbf{x}_i^{'}$, has a better fitness, the position vector of the i-th region is replaced with the new vector $\mathbf{x}_i^{'}$
- The direction in which an ant moves remains the same if a region of better fitness is found
- If the new region is less fit, a new direction is randomly chosen
- Pheromone is updated by adding an amount to each τ_i proportional to the fitness of the corresponding region

Multi-Objective Optimization

- How can the ACO-MH be used to balance more than one sub-objective?
- Main methods:
  - Multi-pheromone approach – one pheromone for each objective
  - Multi-colony approach – one colony for each objective

Multi-Pheromone Approach: Transition Rule

- A pheromone matrix is used for each objective
- A heuristic matrix is used for each objective
- Assume two objectives, and pheromone matrices τ_1 and τ_2
- Each objective has a separate heuristic information matrix
- AS trasistion rule changes to

$$p_{ij}(t) = \begin{cases} \frac{\tau_{1,ij}^{\psi\alpha}(t)\tau_{2,ij}^{(1-\psi)\alpha}(t)\eta_{1_ij}^{\psi\beta}(t)\eta_{2_ij}^{(1-\psi)\beta}(t)}{\sum_{u\in\mathcal{N}_i(t)}\tau_{1,iu}^{\psi\alpha}(t)\tau_{2,iu}^{(1-\psi)\alpha}(t)\eta_{1_iu}^{\psi\beta}(t)\eta_{2_iu}^{(1-\psi)\beta}(t)} & \text{if } j \in \mathcal{N}_i(t) \\ 0 & \text{if } j \notin \mathcal{N}_i(t) \end{cases}$$

ψ is calculated for each ant as the ratio of the ant index to the total number of ants

Multi-Pheromone Approach: Pheromone Update

- Only ants that generated non-dominated solutions may update both pheromone matrices
- Each such ant deposits $\frac{1}{n_{\mathcal{P}}}$
- $n_{\mathcal{P}}$ is the number of ants that constructed a non-dominated solution
- All non-dominated solutions are maintained in an archive

Multi-Pheromone Approach, but Single Heuristic

- The ACS transition rule is changed to

$$j = \begin{cases} \arg\max_{u \in \mathcal{N}_i(t)} \{ \left(\sum_{c=1}^{n_c} w_c \tau_{c,iu}(t)\right)^{\alpha} \eta_{iu}^{\beta}(t) \} & \text{if } r \leq r_0 \\ J & \text{if } r > r_0 \end{cases}$$

n_c is the number of objectives
w_c is the weight assigned to the c-th objective

- J is selected on the basis of the probability

$$p_{iJ}^{c}(t) = \frac{\left(\sum_{c=1}^{n_c} w_c \tau_{c,iJ}(t)\right)^{\alpha} \eta_{iJ}^{\beta}(t)}{\sum_{u \in \mathcal{N}_i} \left(\sum_{c=1}^{n_c} w_c \tau_{c,iu}(t)\right)^{\alpha} \eta_{iu}^{\beta}(t)}$$

Single-Pheromone, Multiple Heuristic

- Only one pheromone matrix
- One heuristic matric for each objective
- AS transition rule changes to

$$p_{ij}(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\prod_{c=1}^{n_c}(\eta_{c,ij})^{\beta_c}}{\sum_{u \in \mathcal{N}_i(t)} \tau_{iu}^{\alpha}(t)\prod_{c=1}^{n_c}(\eta_{c,iu})^{\beta_c}} & \text{if } j \in \mathcal{N}_i(t) \\ 0 & \text{if } j \notin \mathcal{N}_i(t) \end{cases}$$

- All visited links are updated by each ant with an amount $\frac{Q}{f_k(\mathbf{x})}$

Multi-Colony Approach

- For n_c objectives, n_c colonies are used
- One colony for each objective
- Need an information sharing mechanism among the colonies:
  - Each colony implements ACS to optimize one of the objectives
  - Separate pheromone matrices are maintained by the different colonies
  - Local and global updates:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \gamma\tau_0 f_{ij}(t)$$

    $\gamma \in (0, 1)$, and τ_0 is the initial pheromone on link (i, j)
    $f_{ij}(t)$ is referred to as a fitness value
  - Fitness value calculated differently for local and global updates based on a sharing mechanism

Multi-Colony Approach: Sharing Mechanism

- Sharing mechanism is used to exchange information among colonies
- Applied before pheromone updates
- Local sharing:
  - Applied after each next node is added to the solution
  - Rank all partial solutions in classes of non-dominance using non-dominated sorting
  - Each non-dominance class forms a Pareto front, $\mathcal{PF}_p$ containing $n_p = |\mathcal{PF}_p|$ non-dominated solutions
  - The sharing mechanism assigns a fitness value f_p^k to each solution in Pareto front $\mathcal{PF}_p$

Multi-Colony Approach: Sharing Mechanism, Algorithm 17.9

for *each Pareto front* $\mathcal{PF}_p, p = 1, \ldots, n_p^*$ **do**
    for *each solution* $\mathbf{x}^a \in \mathcal{PF}_p$ **do**
        for *each solution* $\mathbf{x}^b \in \mathcal{PF}_p$ **do**
            Calculate d_{ab};
            Calculate sharing value σ_{ab};
        end
        Calculate niche count ξ_a;
        Calculate fitness value f_p^a;
    end
    Calculate f_p;
end

Multi-Colony Approach: Local Sharing

- Normalized Euclidean distance between solution vectors $\mathbf{x}^a$ and $\mathbf{x}^b$

$$d_{ab} = \sqrt{\sum_{l=1}^{L} \left(\frac{x_l^a - x_l^b}{x_{max,l} - x_{min,l}} \right)^2}$$

$L \leq n_x$ is the length of the current paths

- A sharing value σ_{ab} is calculated for each distance

$$\sigma_{ab} = \begin{cases} 1 - \left(\frac{d_{ab}}{\sigma_{share}} \right)^2 & \text{if } d_{ab} < \sigma_{share} \\ 0 & \text{otherwise} \end{cases}$$

$$\sigma_{share} = \frac{0.5}{\sqrt{\frac{L}{n_p^*}}}$$

n_p^* is the desired number of Pareto optimal solutions

Multi-Colony Approach: Niche Count, Fitness

- For each solution, using the sharing values, a niche count is calculated

$$\xi_a = \sum_{b=1}^{n_p} \sigma_{ab} \text{ and } f_p^a = \frac{f_p}{\xi_a}$$

- Fitness value of each solution

$$f_{p+1} = f_{min,p} - \epsilon_p$$

where $f_{p+1} = f_{min,p} - \epsilon_p$ and

$$f_{min,p} = \begin{cases} \min_{a=1,\ldots,n_p}\{f_p^a\} & \text{if } p > 1 \\ f_1 & \text{if } p = 1 \end{cases}$$

- $f_{ij}(t)$ is the fitness value of the Pareto front to which the corresponding solution belongs, i.e. f_p

Multi-Colony Approach: Global Sharing

- Applied after all paths have been constructed
- Rank solutions using non-dominated sorting
  - Find all non-dominated solutions
  - Add them to the current front
  - Remove these inidividuals
  - Advance to the next front and repeat the steps
- f_{ij} calculated as for local sharing, but now wrt the complete solutions

Dynamic Environments

- ACO-MH is inefficient in tracking solutions, due to the pheromone trail following behavior of ants
- To enable tracking of solutions in dynamic environments, exploration abilaties must be improved
- With reference to ACS:
  - Use a small value of r_0
  - Increase β
- Change pheromone update strategy:
  - Only links that form part of a solution are updated
  - The update of these links include evaporation
  - Pheromones on links not part of a solution does not evaporate

Dynamic Environments (cont)

- Reinitialization of pheromone:
  - Reinitialize after change detection
  - Keep reference to the previous best solution found
  - If location of change can be identified, the links in that area can be initialized to a maximum value
- Multi-colony approach:
  - Colonies are replled by the pheromone of other colonies
  - Promotes exploration

Dynamic Environments (cont)

- Change pheromone update rules:
  - Local update rule

$$\tau_{ij}(t+1) = (1 - \rho_1(\tau_{ij}(t)))\tau_{ij}(t) + \Delta\tau_{ij}(t)$$

where $\rho_1(\tau_{ij})$ is monotonically increasing,

$$\rho_1(\tau_{ij}) = \frac{1}{1 + e^{-(\tau_{ij}+\theta)}}$$

with $\theta > 0$.
  - High pheromone values are evaporated more than low pheromone values

- Global update rule, only wrt global-best and global-worst:

$$\tau_{ij}(t+1) = (1 - \rho_2(\tau_{ij}(t)))\tau_{ij}(t) + \gamma_{ij}\Delta\tau_{ij}(t)$$

where

$$\gamma_{ij} = \begin{cases} +1 & \text{if } (i,j) \text{ is in the global-best solution} \\ -1 & \text{if } (i,j) \text{ is in the global-worst solution} \\ 0 & \text{otherwise} \end{cases}$$

Alternative update strategies:

$$\tau_{ij}(t+1) = (1 - \gamma_i)\tau_{ij} + \gamma_i \frac{1}{n_G - 1}$$

n_G is the number of nodes in the representation graph

Dynamic Environments (cont)

- **Restart strategy**: $\gamma_i = \lambda_R$ where $\lambda_R \in [0, 1]$
- η-**strategy**: Heuristic information is used to decide degree of pheromone evaporation

$$\gamma_i = \max\{0, d_{ij}^{\eta}\}, \quad d_{ij}^{\eta} = 1 - \frac{\overline{\eta}}{\lambda_\eta \eta_{ij}}, \ \lambda_\eta \in [0, \infty)$$

$$\overline{\eta} = \frac{1}{n_G(n_G - 1)} \sum_{i=1}^{n_G} \sum_{j=1, j \neq i} \eta_{ij}$$

γ_i is proportional to the distance from the changed component
all links incident to the component are updated

Dynamic Environments (cont)

- τ-**strategy**: evaporates pheromone on links closer to the changed component more

$$\gamma_i = \min\{1, \lambda_\tau d_{ij}^\tau\},\ \lambda_\tau \in [0, \infty)$$

$$d_{ij}^\tau = \max_{\mathcal{N}_{ij}} \left\{ \prod_{(x,y)\in\mathcal{N}_{ij}} \frac{\tau_{xy}}{\tau_{max}} \right\}$$

N_{ij} is the set of all paths from i to j

Applying the ACO-MH

- Can be applied to optimization problems for which the following problem-dependent aspects can be defined:

1. An appropriate **graph representation** to represent the discrete search space. A solution representation scheme also has to be defined.
2. An **autocatalytic (positive) feedback process**; that is, a mechanism to update pheromone concentrations such that current successes positively influence future solution construction.
3. **Heuristic desirability** of links in the representation graph.
4. A **constraint-satisfaction method** to ensure that only feasible solutions are constructed.
5. A solution construction method which defines the way in which solutions are built, and a state transition probability.

Applying the ACO-MH to the TSP

- Given a set of n_π cities
- The objective is to find a minimal length closed (Hamiltonian) tour that visits each city once
- Let π represent a solution as a permutation of the cities $\{1, \ldots, n_\pi\}$, where $\pi(i)$ indicates the i-th city visited
- $\Pi(n_\pi)$ is the set of all permutations of $\{1, \ldots, n_\pi\}$, i.e. the search space
- Formally, $\pi^* = \arg\min_{\pi \in \Pi(n_\pi)} f(\pi), \quad f(\pi) = \sum_{i,j=1}^{n_\pi} d_{ij}$
- Let $D = [d_{ij}]_{n_\pi \times n_\pi}$ denote the distance matrix

Applying the ACO-MH to the TSP (cont)

- Problem representation:
  - The representation graph is the 3-tuple, $G = (V, E, D)$
  - V is the set of nodes, each representing one city
  - E represents the links between cities
  - D is the distance matrix which assigns a weight to each link $(i, j) \in E$
  - A solution is represented as an ordered sequence $\pi = (1, 2, \ldots, n_\pi)$ which indicates the order in which cities are visited

Applying the ACO-MH to the TSP (cont)

- Heuristic Desirability:
  - The desirability of adding city j after city i is calculated as

$$\eta_{ij}(t) = \frac{1}{d_{ij}(t)}$$

- Constraint satisfaction:
  1. All cities must be visited – require each solution to contain n cities
  2. Each city is visited once only – $\mathcal{N}_i^k(t)$ only contains nodes not yet visited

E. Bonabeau, M. Dorigo, and G. Theraulaz.

Swarm Intelligence: From Natural to Artificial Systems.

Oxford University Press, 1999.

L. Chrétien.

Organisation Spatiale du Matériel Provenant de l'excavation du nid chez Messor Barbarus et des Cadavres d'ouvrières chez "Lasius niger" (Hymenopterae: Formicidae).

PhD thesis, Université Libre de Bruxelles, 1996.

J-L. Deneubourg, S. Aron, S. Goss, and J-M. Pasteels.

The Self-Organizing Exploratory Pattern of the Argentine Ant.

Journal of Insect Behavior, 3:159–168, 1990.

M. Dorigo.

Optimization, Learning and Natural Algorithms.

PhD thesis, Politecnico di Milano, 1992.

M. Dorigo and G. Di Caro.

Ant Colony Optimization: A New Meta-Heuristic.

In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 2, page 1477, July 1999.

M. Dorigo, V. Maniezzo, and A. Colorni.

Ant System: Optimization by a Colony of Cooperating Agents.

IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1):29–41, 1996.

D.M. Gordon and N.J. Mehdiabadi.

Encounter Rate and Task Allocation in Harvester Ants.

Behavioral Ecololgy and Sociobiology, 45:370–377, 1999.

P-P. Grassé.

La Reconstruction du nid et les Coordinations Individuelles chez *Bellicositermes Natalensis* et *Cubitermes sp.* la Théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs.

Insectes Sociaux, 6:41–80, 1959.

J. Handl, J. Knowles, and M. Dorigo.

Ant-Based Clustering: A Comparative Study of Its Relative Performance with Respect to k-Means, Average Link and 1D-SOM.

Technical Report TR/IRIDIA/2003-24, Université Libre de Bruxelles, 2003.

B. Hölldobler and E.O. Wilson.

Journey of the Ants: A Story of Scientific Exploration.

Harvard University Press, 1994.

M. Maeterlinck.

The Life of the White Ant.

Dodd-Mead, New York, 1927.

E.N. Marais.

Die Siel van die Mier (The Soul of the Ant).

J.L. van Schaik, Pretoria, South Africa, fifth edition, 1948. (first published in 1937).

A.B. Sendova-Franks and N.R. Franks.

Self-Assembly, Self-Organization and Division of Labour.

Philosophical Transactions of the Royal Society of London, 354:1395–1405, 1999.

G. Théraulaz and E. Bonabeau.

A Brief History of Stigmergy.

Artificial Life, 5:97–116, 1999.

G. Théraulaz, E. Bonabeau, and J-L. Deneubourg.

Response Threshold Reinforcement and Division of Labour in Insect Societies.

Proceedings of the Royal Society of London, Series B, 265:327–332, 1998.

Y. Yang and M. Kamel.

Clustering Ensemble using Swarm Intelligence.

In *Proceedings of the IEEE Swarm Intelligence Symposium*, pages 65–71, 2003.