Natural Computing

Lecture 11

Michael Herrmann
mherrman@inf.ed.ac.uk
phone: 0131 6 517177
Informatics Forum 1.42

25/10/2011

ACO II

ACO (in brief)

ACO

- Represent solution space
- Set parameters, initialize pheromone trails
- SCHEDULE_ACTIVITIES
  - ConstructAntSolutions
  - DaemonActions {optional}
  - UpdatePheromones
  - Check termination condition
- END_SCHEDULE_ACTIVITIES

ACO algorithm

Algorithm 1 The framework of a basic ACO algorithm

input: An instance P of a CO problem model $\mathcal{P} = (\mathcal{S}, f, \Omega)$.
InitializePheromoneValues($\mathcal{T}$)
$\mathfrak{s}_{bs} \leftarrow \text{NULL}$ — init best-so-far solution
while termination conditions not met **do**
  $\mathfrak{S}_{iter} \leftarrow \emptyset$ — set of valid solutions
  for $j = 1, \ldots, n_a$ **do** — loop over ants
    $\mathfrak{s} \leftarrow$ ConstructSolution($\mathcal{T}$)
    if $\mathfrak{s}$ is a valid solution **then**
      $\mathfrak{s} \leftarrow$ LocalSearch($\mathfrak{s}$) {optional}
      if $(f(\mathfrak{s}) < f(\mathfrak{s}_{bs}))$ or $(\mathfrak{s}_{bs} = \text{NULL})$ **then** $\mathfrak{s}_{bs} \leftarrow \mathfrak{s}$ — update best-so-far
      $\mathfrak{S}_{iter} \leftarrow \mathfrak{S}_{iter} \cup \{\mathfrak{s}\}$ — store valid solutions
    end if
  end for
  ApplyPheromoneUpdate($\mathcal{T}$,$\mathfrak{S}_{iter}$,$\mathfrak{s}_{bs}$)
end while
output: The best-so-far solution $\mathfrak{s}_{bs}$

Variants: Ant System (AS)

Pheromone trails: Initialise $\tau_{ij} = \tau_0 \ll 1$

$$\tau_{ij} \leftarrow \rho\tau_{ij} + (1-\rho)\sum_{k=1}^{n} \Delta\tau_{ij}^{k}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{L_k} & \text{if ant } k \text{ used edge } (i,j) \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}$$

Probability rule

$$p\left(c_{ij}|s_k^p\right) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{c_{im}\in N\left(s_i^k\right)} \tau_{ij}^{\alpha}\eta_{ij}^{\beta}} & \text{if } j \in N\left(s_i^k\right) \\ 0 & \text{otherwise} \end{cases}$$

c_{ij}: graph edge, s_i^k partial solution of ant k sofar incl. arrival to i, N set of possible continuations of s_i^k (e.g. towards j if $j \in N\left(s_i^k\right)$.

see: Dorigo et al. 1991/92 and www.scholarpedia.org/article/Ant_colony_optimization

Variants: Ant Colony System (ACS)

Pheromone trails: Initialise $\tau_{ij} = \tau_0 \ll 1$

Local pheromone update

$$\tau_{ij} \leftarrow \rho\tau_{ij} + (1-\rho)\sum_{k=1}^{n} \Delta\tau_0$$

in addition to *global update* (best ant contribution)

$$\tau_{ij} \leftarrow \rho\tau_{ij} + (1-\rho)\,\Delta\tau_{ij}^{\text{best}}$$

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{L_k} & \text{if best ant used edge } (i,j) \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}$$

Pseudorandom propotional rule: Use probability rule with prob. q_0

$$p\left(c_{ij}|s_k^p\right) = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{c_{im}\in N\left(s_i^k\right)} \tau_{ij}^{\alpha}\eta_{ij}^{\beta}} & \text{if } j \in N\left(s_i^k\right) \\ 0 & \text{otherwise} \end{cases}$$

or make a random (admissible) transition otherwise.

see: Dorigo and Gambardella 1997

Variants: Max-Min Ant System (MMAS)

Best ant adds to the pheromone trails (iteration best or best so far)

Initialise e.g. $\tau_{ij} = \tau_{\max} = \frac{1}{pL^*}$ where L^* best sofar known optimum

$$\tau_{ij} \leftarrow \rho\tau_{ij} + (1-\rho)\sum_{k=1}^{n} \Delta\tau_{ij}^{\text{best}}$$

Pheromone production by the best ant only

$$\Delta\tau_{ij}^{best} = \begin{cases} \frac{1}{L_k} & \text{if best ant used edge } (i,j) \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}$$

Minimum and maximum value of the pheromone are explicitely limited by $\tau_{\min}$ and $\tau_{\max}$ (truncation).

Pseudorandom propotional rule

$$p\left(c_{ij}|s_k^p\right) = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{c_{im}\in N\left(s_i^k\right)} \tau_{ij}^{\alpha}\eta_{ij}^{\beta}} & \text{if } j \in N\left(s_i^k\right) \\ 0 & \text{otherwise} \end{cases}$$

see: Dorigo and Gambardella 1997

Variants: Max-Min Ant System (MMAS)

Comments

- τ_{min} is small and guarantees continuous exploration (may be reduced in order to enforce convergence.
- τ_{max} causes all non-visited regions to be equally attractive (pheromone update can be restricted to the links that were actually visited, but you will need fast evaporation in this case)
- If τ_{max} is not very large it can prevent over-exploration of frequently used regions or may be set to the largest pheromone value that is possible in the given ant system.
- In the face of stagnation the paths can reinitialized by τ_{max}
- Essentially two more parameters to play with.

ACO variants

ACO variant	Authors	Year
Elitist AS (EAS)	Dorigo	1992
Continuous ACO (CACO)	Bilchev and I.C. Parmee Dorigo, Maniezzo, and Colorni	1995 1996
Ant Colony System (ACS)	Dorigo and Gambardella	1997
Rank-based AS (RAS)	Bullnheimer, Hartl, and Strauss	1999
Max-Min Ant System (MMAS)	Stützle and Hoos	2000
Continous Orthogonal (COAC)	Hu, Zhang, and Li	2008
Hyper-Cube Framework (HCF)	Blum and Dorigo	2004

Properties of ACO in a numerical experiment

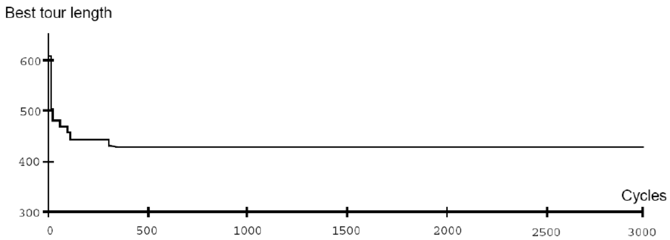


Fig. 3. Evolution of best tour length (Oliver30). Typical run.

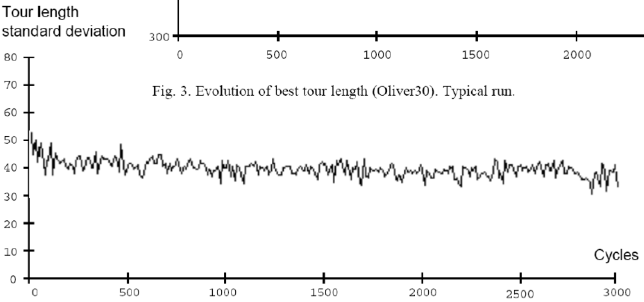


Fig. 4. Evolution of the standard deviation of the population's tour lengths (Oliver30). Typical run.

Properties of ACO in a numerical experiment

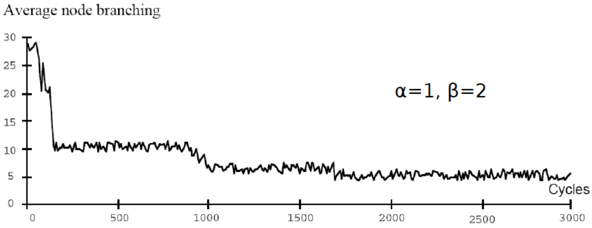


Fig. 5. Evolution of the average node branching of the problem's graph (Oliver30). Typical run.

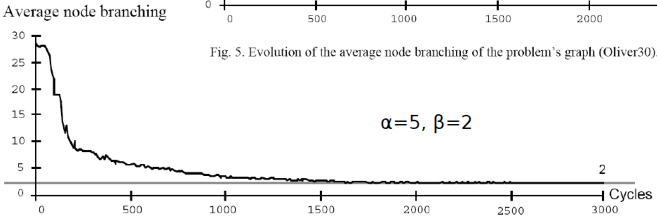


Fig. 7. Average node branching of a run going to stagnation behavior (Oliver30). Typical run obtained setting α=5 and β=2.

Dorigo et al.: Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics B26:1 (1996) 1-13

Numerical experiment

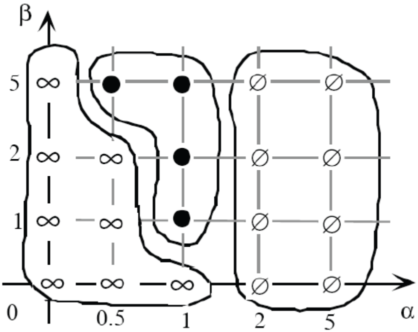


Fig. 8. *Ant-cycle* behavior for different combinations of α-β parameters.

● - The algorithm finds the best known solution without entering the stagnation behavior.
∞ - The algorithm doesn't find good solutions without entering the stagnation behavior.
∅ - The algorithm doesn't find good solutions and enters the stagnation behavior.

Dorigo et al.: Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics B26:1 (1996) 1-13

Theoretical results: Overview

- Convergence in probability of an ACO algorithm (Gutjahr 2000) [Theoretical bounds]
- Run-time analysis
- Understanding ACO: Search biases
- Hypercube framework
- Relations to other optimization algorithms (later)

ACO: Convergence

- For simplified algorithms; bounds not very tight
- Given a lower bound for the pheromones the algorithm explores everywhere and must therefore find an optimal solution given sufficiently long time, i.e.

Theorem (Dorigo & Stuetzle): Let $p^*(t)$ be the probability that ACO (best-so-far update and lower bound for pheromones) finds an optimal solution at least once within the first t iterations. Then, for an arbitrarily small $\varepsilon > 0$ and for a sufficiently large t it holds that: $p^*(t) = 1 - \varepsilon$ and asymptotically $\lim_{t\to\infty} p^*(t) = 1$.

- Logarithmic decrease of τ_{min} is also OK, i.e. $\tau_{\text{min}} = c/\log(t)$
- Safe convergence times can be very large!
- This was for MMAS. For standard AS, ρ should approach 1 (for large t) as $1 - c/(t\log(t))$ which might be too fast to reach good solutions unless c is appropriately chosen

Run-time analysis

- Results for very simple problems such as "needle in a haystack" (only one optimum and flat elsewhere) and "all-ones" (ONE-MAX: one optimum reachable from everywhere)
- For large evaporation rate (unrealistic!): identical to (1+1)-ES (i.e. weakly exponential)
- For small evaporation rates polynomial complexity can be achieved (for ONE-MAX) Neumann & Witt (2007)
- Comparing MMAS and 1-ANT (only one ant with rescaled pheromone update): Depends on evaporation rate. Run-time can be $O(n^2)$ for MMAS when 1-ANT has already exponential run-time (Gutjahr and Sebastiani 2007)

Search Biases

- A desirable search bias towards good zones of the search space is given by the pheromones
- *Negative search bias caused by selection fixpoints*
- *Negative search bias caused by an unfair competition*
- Note: For these theoretical considerations sometimes local heuristic information is ignored (e.g. setting $\beta = 0$), i.e. the question is: How does ACO work beyond local search?

M. Dorigo, C. Blum: Theoretical Computer Science 344 (2005) 243 – 278

Selection fix-points: Ant number

Why using more than one ant per iteration? Wouldn't the algorithm work with only one ant?

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

$$\Delta\tau_{ij} = P\left(c_{ij}|s_i^k\right) = \frac{\tau_{ij}}{\sum_k \tau_{ik}(t)}$$

Pheromones tend towards $\tau_{ij} = \Delta\tau_{ij}$

Effect increases with problem size. (Merkle & Middendorf 2004)

Single-ant algorithms need to use very slow evaporation and are therefore less likely to explore well before stagnation. Or, if stagnation is prevented (e.g. by $\tau_{\min} > 0$) they are very slow.

Several ants building a common pheromone matrix may contribute various "building blocks" to a solution that is likely to be followed by the ants of later generations.

Competition between the ants is a driving force of ACO algorithms.

Analogous to but not generally the same as in GA: Ants are more likely to test various combinations of good part-solutions ("building blocks")

Selection fix-points: Constraints

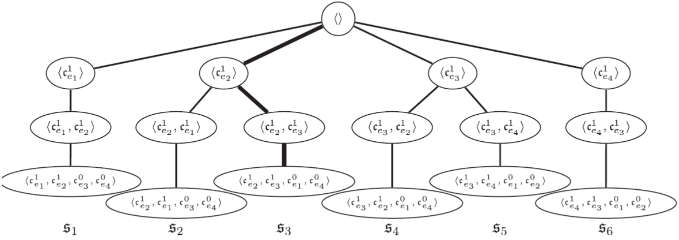
- The adaptation of the pheromone matrix depends also on the number of ants having passed a solution component
- In unconstrained problems all nodes of the underlying graph have the same degree
- In constrained problems the degree may differ such that poor regions with low degrees become more attractive than good regions with high degree
- One can construct examples where the increased exploration of the bad regions lead to a fixed point of the algorithm or a local minimum of the search problem

Bias by an "unfair" competition

- Unconstrained ACOs always improve the iteration quality (expected value of the ants' performance) or are stationary
- Constrained problem: minimal k-cardinality sub-tree

- Minise w.r.t. weights: $w(e_1) = w(e_4) = 1$ and $w(e_2) = w(e_3) = 2$
- Optimal solution shoud contain e_1 or e_4.

- Start solution with empty set and add nodes sequentially

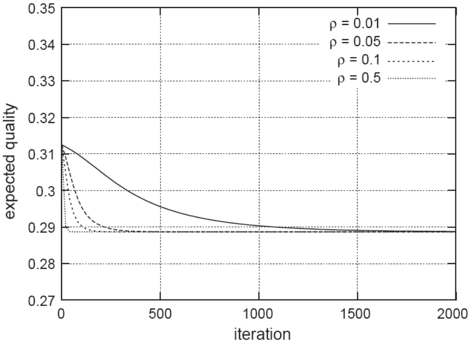


Fitness values $f(\mathfrak{s}_1) = f(\mathfrak{s}_2) = f(\mathfrak{s}_5) = f(\mathfrak{s}_6) = 3$ and $f(\mathfrak{s}_3) = f(\mathfrak{s}_4) = 4$

- $k = 2$ gives ($2 \times$ "on", $2 \times$ "off")
- The outer routes will still get less pheromone (for AS), good solutions are not equally sampled
- The branching paths get twice as much update although they lead to a larger expected cost

Bias by an unfair competition

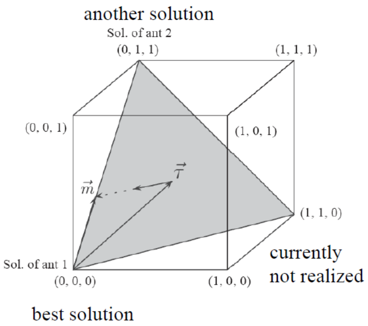
- Quality decreases (!) when starting from a homogeneous initial pheromone matrix
- The impact of the pheromone value update increases and the expected iteration quality decreases faster
- Note: Here quality is fitness (inverse costs)



Search space: "Hyper-cube framework"

- Given a solution (path) $s = (s_1, \ldots, s_n)$
- The solution is a subsets of the edges E of a graph $G = (N, E)$
- Partitioning of E: if a link belongs to s: 1 otherwise 0

- For TSP, s can be represented by a binary vector with dimension $M = n(n-1)/2$ [in case of TSP, generally this would be total number of available solution components]
- Pheromones are updated in the span of admissible solutions



- The HCF is not an algorithm, but a framework which applies for several variants

Search space: "Hyper-cube framework"

Pheromone update ($c_j = s_j^i$ if it is a component of solution i)

$\tau_j \leftarrow \rho\tau_j + \sum_{i=1}^{k} \Delta\tau_j^i$ where $\Delta\tau_j^i = \begin{cases} \frac{1}{f(s^i)} & \text{if } c_j \in s^i \\ 0 & \text{otherwise} \end{cases}$

$\lim_{t\to\infty} \tau_i(t) \leq \frac{1}{1-\rho} \cdot \frac{k}{f\left(s^{\text{opt}}\right)}$ Maximal if all k ants follow forever the optimal solution: $\tau_i = \rho\tau_i + \frac{k}{f\left(s^{\text{opt}}\right)}$

$\tau = (\tau_1, \ldots, \tau_M)$ is an M-dimensional vector: $\tau \in \left[\tau_{\text{min}}, \tau_{\text{max}}\right]^M$

M: number of solution components (e.g. all edges of a graph)

$\tau = \sum \alpha^i s^i$, $\alpha^i \in \left[\tau_{\text{min}}, \tau_{\text{max}}\right]$, $s^i \in \{0,1\}^M$, w.l.o.g. $\alpha_j \in [0,1]$

only used components of s^i being 1, elsewhere 0s

Blum, Roli, Dorigo (2001) HC-ACO. 4th Metaheuristics Int. Conf., 399-403.

Search space: "Hyper-cube framework"

- A binarised solution $s = (s_1, \ldots, s_M)$ is a subset of the edges E of a graph $G = (N, E)$ indicated by s_i being 1 or 0.
- Pheromone normalisation

$$\tau_j \leftarrow \rho\tau_j + \sum_{i=1}^{k} \Delta\tau_j^i \text{ where } \Delta\tau_j^i = \begin{cases} \dfrac{\frac{1}{f(s^i)}}{\sum_{l=1}^{k} \frac{1}{f(s^l)}} & \text{if } c_j \in s^i \\ 0 & \text{otherwise} \end{cases}$$

- Hyper-cube update rule ($\mu = 1 - \rho$)

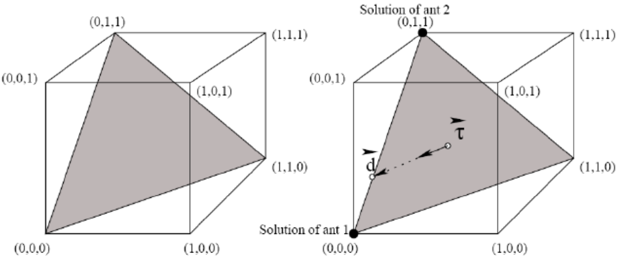
$$\tau \leftarrow \tau + \mu\,(d - \tau)$$

$$d = (d_1, \ldots, d_M) \text{ where } d_j = \sum_{i=1}^{k} \frac{\frac{1}{f(s^i)}}{\sum_{l=1}^{k} \frac{1}{f(s^l)}},\ j = 1, \ldots, M$$

- Pheromones are updated in the span of the solutions

Search space: "Hyper-cube framework"

$$\tau \leftarrow \tau + \mu \left(d - \tau \right)$$



The pheromone vector moves a bit towards the weighted mean of the solutions produced by the current iteration.

Benefits of the Hyper-cube framework

- Probabilistic interpretation (in the absence of constraints)
- Proof that expected quality of solutions strictly increases (without the assumption of an infinitesimal step size as in standard gradient methods!)
- A diversification scheme
  - global desirability: $v_j^{\text{des}} \leftarrow \max\left\{\frac{1}{f(s)} : s \in S_{\text{ants}}, s_j = 1\right\}$
  - global frequency: $v_j^{\text{fr}} \leftarrow \sum_{s\in S_{\text{ants}}} s_j$

  S_{ants} : all solutions generated since the start
- At stagnation the algorithm may be restarted with a pheromone $(n \times n)$ matrix (or vector in $n(n-1)/2$ dimensions) constructed from v^{des} or the regularised inverse of v^{fr} in order to keep good solutions, but also to favour regions where few ants have been before.

Conclusion on ACO

- Cooperative scheme: Building a common solution
- Prone to premature convergence (can be controlled by minimal pheromone level, by adjusting evaporation rate or reinitialising the pheromone levels)
- some theory exists, points to analogies among metaheuristic serach algorithms
- Applications in scheduling, route planing and any problem that can be brought into this form

next time: PSO