DM63
HEURISTICS FOR
COMBINATORIAL OPTIMIZATION

Lecture 11

Ant Colony Optimization Exercises

Marco Chiarandini

Outline

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 (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:
  | 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 $\eta_{ij} := \frac{1}{c_{ij}}$
- 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 $\Delta_{ij}(s) := 1/C^s$
  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 + perturbative search phases).

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)

Ant System on TSP

- Initialization:

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

  Motivation: sligthly more than what evaporates
- Construction: m ants in m randomly chosen cities

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

- Update

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

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

Elitist Ant System on TSP

- Update

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

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m}\Delta_{ij}^{k} + e\cdot\Delta_{ij}^{bs} \qquad \text{to the edges visited by the ants}$$

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

Rank-based Ant System on TSP

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

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

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

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

$\mathcal{MAX}-\mathcal{MIN}$ Ant System (MMAS)

Peculiarities in pheromone management:

- Update

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

  Meaning of best alternates during the search between:
  - best-so-far
  - iteration best
- bounded values τ_{min} and τ_{max}
- $\tau_{max} = \frac{1}{\rho C^*}$ and $\tau_{min} = \frac{\tau_{max}}{a}$
- Reinitialization of τ 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) on the TSP

Three main ideas:

- Different state transition rule

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

- 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:

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

  Parallel construction preferred to sequential construction

Approximate Nondeterministic Tree Search on TSP

- 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

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

- Different pheromone trail update rule

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{i=1}^{k} \Delta_{ij}^k \qquad \Delta_{ij}^k = \begin{cases} \theta(1 - \frac{C^k - LB}{L_{avg} - LB} & \text{if } (ij) \text{ in belongs to } T^k \\ 0 & \text{otherwise} \end{cases}$$

Strongly Invariant ACO on TSP

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

The siACO is an algorithm that enjoy the property of 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.

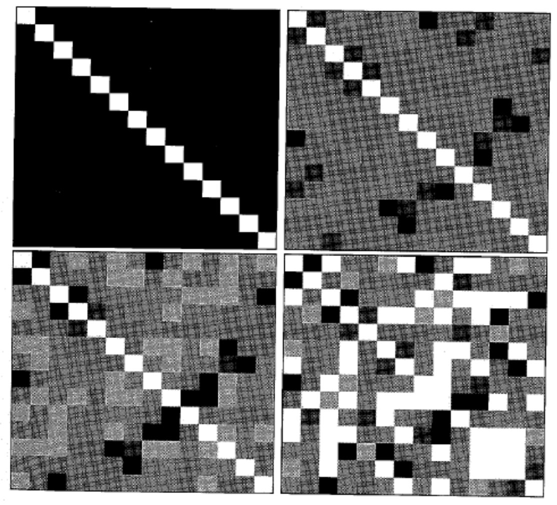
Can be extended to other ACO versions and to other problems: QAP and Scheduling

Analysis

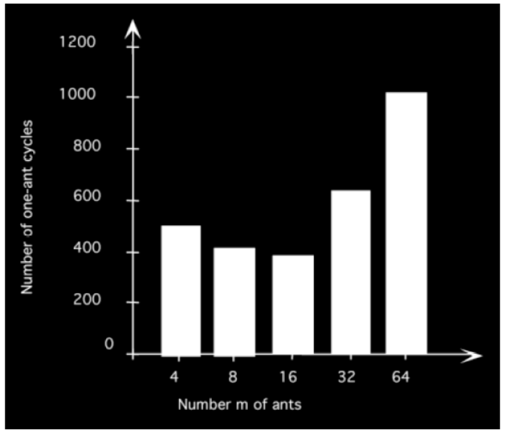
Other things to check

- 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

Pheromone development

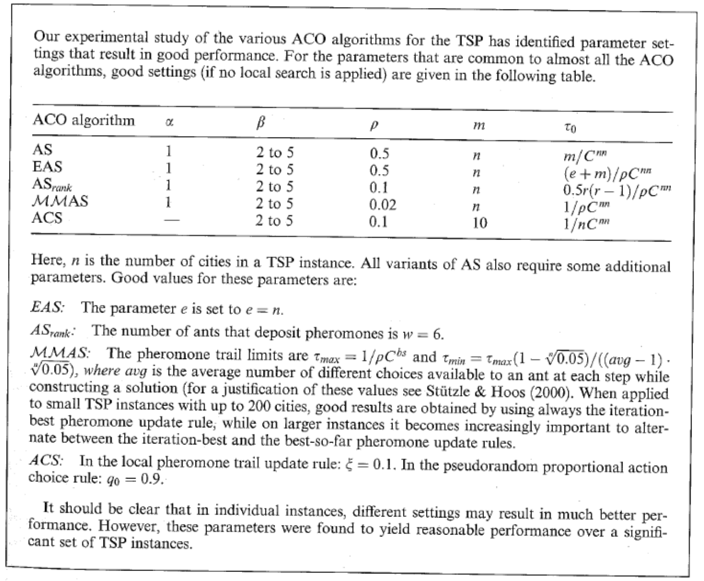


Analysis



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

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	α	β	ρ	m	τ_0
AS	1	2 to 5	0.5	n	m/C^{nn}
EAS	1	2 to 5	0.5	n	$(e+m)/\rho C^{nn}$
AS$_{rank}$	1	2 to 5	0.1	n	$0.5r(r-1)/\rho C^{nn}$
$\mathcal{MM}$AS	1	2 to 5	0.02	n	$1/\rho C^{nn}$
ACS	—	2 to 5	0.1	10	$1/nC^{nn}$

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$_{rank}$: The number of ants that deposit pheromones is $w = 6$.

$\mathcal{MM}$AS: The pheromone trail limits are $\tau_{max} = 1/\rho C^{bs}$ and $\tau_{min} = \tau_{max}(1 - \sqrt[n]{0.05})/((avg - 1) \cdot \sqrt[n]{0.05})$, *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: $\xi = 0.1$. 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.

Ant Colony Optimization . . .

- A framework and several problem applications are available from: `http://www.aco-metaheuristic.org/`
- good performance also in *dynamic optimization problems*, such as routing in telecommunications networks

For further details on Ant Colony Optimization, 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 agents $\sigma : J \to I$ such that:

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

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 $I \times J$ 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 i_2 after job i_1
- 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
```

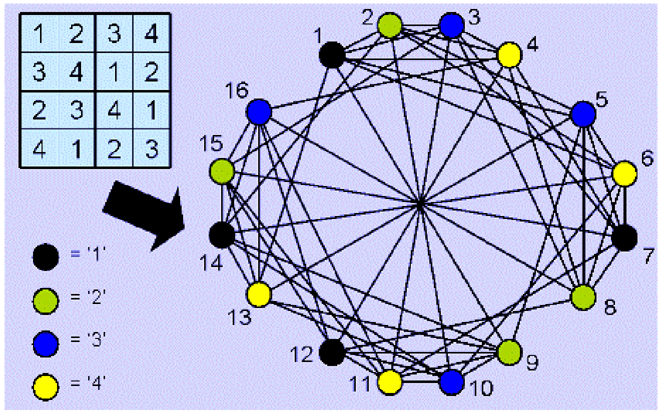
Adaptive Iterated Construction Search:

```
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, Analyze, Prioritize

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

Sudoku



GCP into SAT

Set Covering into MAX-SAT

Sudoku into Exact Hitting Set

Exact Covering: Set partitioning with $\vec{c} = \vec{1}$

- A = 1, 4, 7;
- B = 1, 4;
- C = 4, 5, 7;
- D = 3, 5, 6;
- E = 2, 3, 6, 7; and
- F = 2, 7.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} y_j \\ & \sum_{j=1}^{n} a_{ij} y_j = 1 \quad \forall i \\ & y_j \in \{0,1\} \end{aligned}$$

$$\begin{array}{c|cccccc} & A & B & C & D & E & F \\ \hline 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 1 & 1 \\ 3 & 0 & 0 & 0 & 1 & 1 & 0 \\ 4 & 1 & 1 & 1 & 0 & 0 & 0 \\ 5 & 0 & 0 & 1 & 1 & 0 & 0 \\ 6 & 0 & 0 & 0 & 1 & 1 & 0 \\ 7 & 1 & 0 & 1 & 0 & 1 & 1 \end{array}$$

The dual of Exact Covering is the Exact Hitting Set

- A = 1, 2
- B = 5, 6
- C = 4, 5
- D = 1, 2, 3
- E = 3, 4
- F = 4, 5
- G = 1, 3, 5, 6

$$\begin{aligned} \max \quad & \sum_{j=1}^{n} x_j \\ & \sum_{j=1}^{n} a_{ij} x_j = 1 \quad \forall i \\ & x_j \in \{0,1\} \end{aligned}$$

$$\begin{array}{c|ccccccc} & A & B & C & D & E & F & G \\ \hline 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 4 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 5 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 6 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Asymmetric TSP into Symmetric TSP

Capacited Vehicle Routing (CVRP)

Input:

- complete graph $G(V, A)$, where $V = \{0, \dots, n\}$
- vertices $i = 1, \dots, n$ are customers that must be visited
- vertex $i = 0$ is the single depot
- arc/edges have associated a cost c_{ij} $(c_{ik} + c_{kj} \geq c_{ij}, \forall\, i, j \in V)$
- costumers have associated a non-negative demand d_i
- a set of K identical vehicles with capacity C $(d_i \leq C)$

Task: Find collection of K circuits with minimum cost, defined as the sum of the costs of the arcs of the circuits and such that:

- each circuit visit the depot vertex
- each customer vertex is visited by exactly one circuit; and
- the sum of the demands of the vertices visited by a circuit does not exceed the vehicle capacity C.

Lower bound to K: $K \geq K_{min}$ where K_{min} is the number of bins in the associated *Bin Packing Problem*)

Construction Heuristics

Construction heuristics specific for TSP

- Heuristics that Grow Fragments
  - Nearest neighborhood heuristics
  - Double-Ended Nearest Neighbor heuristic
  - Multiple Fragment heuristic (aka, greedy heuristic)
- Heuristics that Grow Tours
  - Nearest Addition
  - Farthest Addition
  - Random Addition
  - Nearest Insertion
  - Farthest Insertion
  - Random Insertion
  - Clarke-Wright savings heuristic
- Heuristics based on Trees
  - Minimum span tree heuristic
  - Christofides' heuristics
  - Fast recursive partitioning heuristic

CVRP

Construction Heuristics

- Nearest neighbors
- Savings heuristics (Clarke and Wright)
- Insertion heuristics
- Route-first cluster-second
- Cluster-first route-second
  - Sweep algorithm
  - Generalized assignment
  - Location based heuristic
  - Petal algorithm

Perturbative Search

- Solution representation: sets of integer sequences, one per route
- Neighborhoods structures:
  - intra-route: 2-opt, 3-opt
  - inter-routes: λ-interchange, relocate, exchange, CROSS, ejection chains, GENI