Genetic Algorithms and Genetic Programming

Lecture 11: (30/10/09)

Ant Colony Optimization

School of informatics

Michael Herrmann

michael.herrmann@ed.ac.uk, phone: 0131 6 517177, Informatics Forum 1.42

Overview: Remainder of the course

I. GA (1-7)
II. GP (8-10)
III. ACO (11-13): Ant colony optimization
IV. PSO (14-15): Particle swarm optimization and differential evolution
V. NC (16): Overview on DNA computing, Membrane computing, Molecular computing, Amorphous computing, Organic computing, ….
VI. Wrapping up: Metaheuristic search (17)

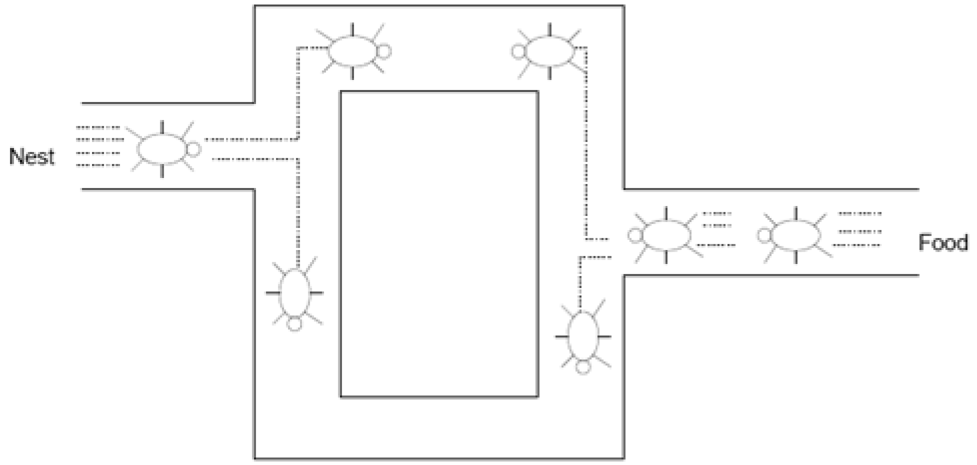
Not included:

artificial neural networks, quantum computing, cellular automata, artificial immune systems

蟻 蚁

Ant Colony Optimisation

Biological inspiration: ants find the shortest path between their nest and a food source using **pheromone trails**.



Ant Colony Optimisation is a population-based search technique for the solution of combinatorial optimisation problems which is inspired by this behaviour.

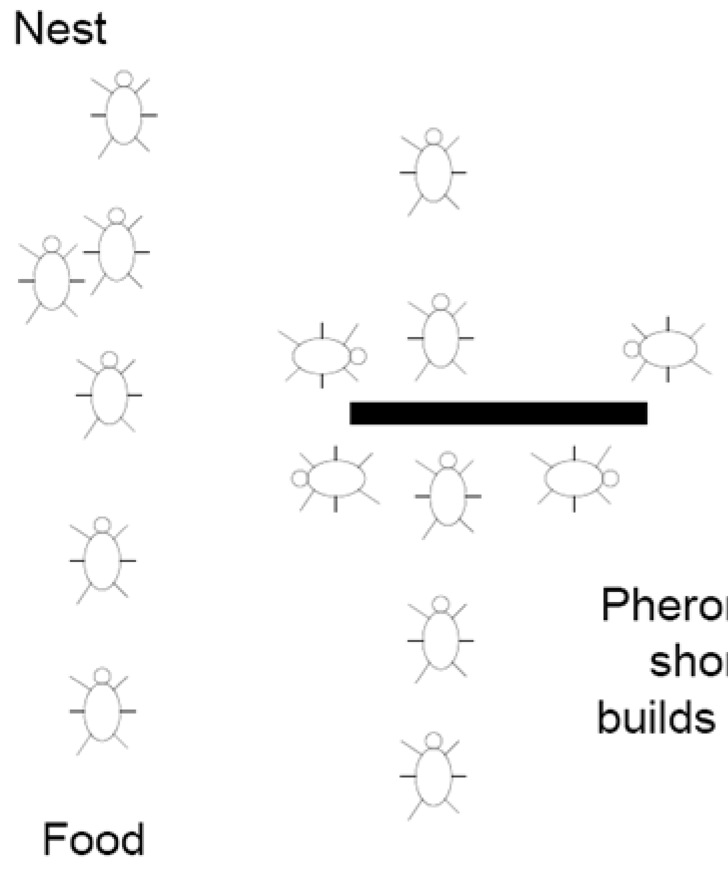
Marco Dorigo (1992). Optimization, Learning and Natural Algorithms. *Ph.D.Thesis*, Politecnico di Milano, Italy, in Italian.
"The Metaphor of the Ant Colony and its Application to Combinatorial Optimization"

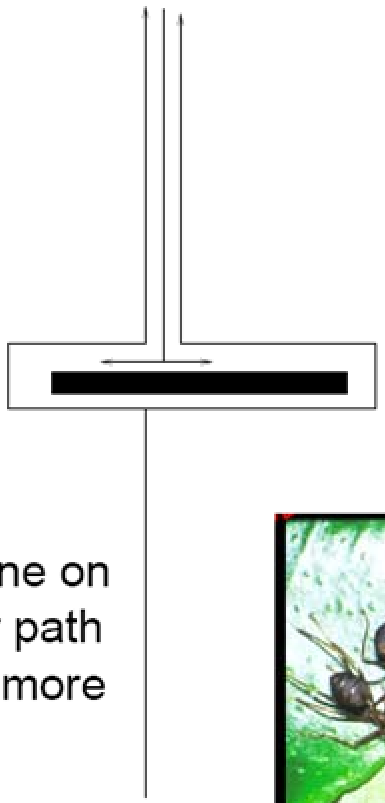
Based on theoretical biology work of Jean-Louis Deneubourg (1987) *From individual to collective behavior in social insects* . Birkhäuser Verlag, Boston.

What, how?

- Real ants find shortest routes between food and nest
- They hardly use vision (almost blind)
- They lay pheromone trails, chemicals left on the ground, which act as a signal to other ants – **STIGMERGY**
- If an ant decides, with some probability, to follow the pheromone trail, it itself lays more pheromone, thus reinforcing the trail.
- The more ants follow the trail, the stronger the pheromone, the more likely ants are to follow it.
- Pheromone strength decays over time (half-life: a few minutes)
- Pheromone builds up on shorter path faster (it doesn't have so much time to decay), so ants start to follow it.

Pheromone Builds Up on Short Paths





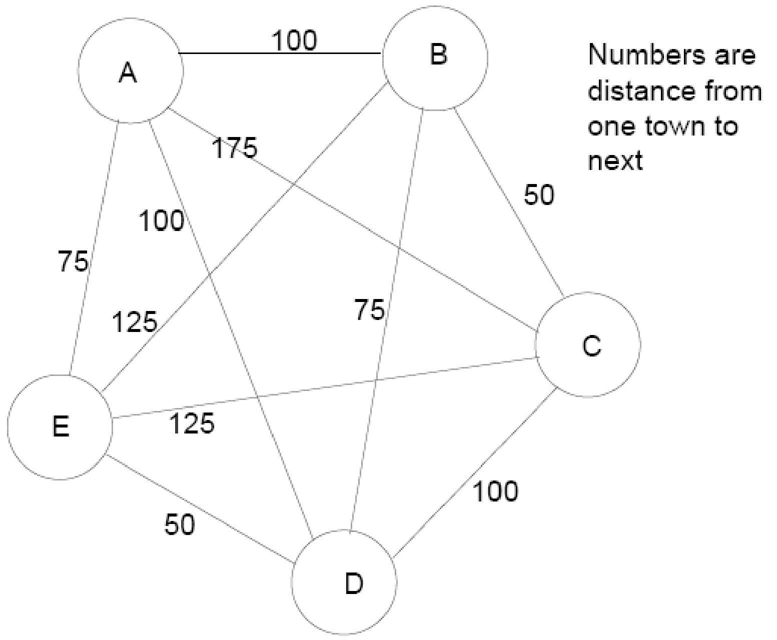
http://www.scs.carleton.ca/~arpwhite/courses/95590Y/notes/SI%20Lecture%203.pdf

Artificial Ant Systems

- Do have some memory (data structures)
- Are able to sense “environment” if necessary (not just pheromone)
- Use discrete time
- Are optimisation algorithms

So can we apply them to an optimisation problem: Travelling Salesperson Problem

Ant System for the TSP: An Example



Find the tour that minimises the distance travelled in visiting all towns.

Ant System for the TSP

- Each ant builds its own tour from a starting city
- Each ant chooses a town to go to with a probability: this is a function of the town's distance and the amount of pheromone on the connecting edge
- Legal tours: transitions to already visited towns disallowed till tour complete (keep a tabu list)
- When tour completed, lay pheromone on each edge visited
- Next city j after city i chosen according to Probability Rule

Probability Rule

$$\boxed{p(i,j) = \frac{[\tau(i,j)].[\eta(i,j)]^{\beta}}{\sum_{g\in\text{allowed}}[\tau(i,g)].[\eta(i,g)]^{\beta}}}$$

- Strength of pheromone $\tau(i,j)$ is favourability of j following i
  Emphasises "**global** goodness": the **pheromone matrix**
- Visibility $\eta(i,j) = 1/d(i,j)$ is a simple heuristic guiding construction of the tour. In this case it's greedy – the nearest town is the most desirable (seen from a **local** point of view)
- β is a constant, e.g. 2
- $\sum_{g\in\text{allowed}}$: normalise over all the towns g that are still permitted to be added to the tour, i.e. not on the tour already
- So τ and η trade off global and local factors in construction of tour

Pheromone

- Pheromone trail evaporates a small amount after every iteration

$$\tau(i,j) = \rho\,\tau(i,j) + \Delta\tau_{ij}$$

  where $0 < \rho < 1$ is an evaporation constant

- The density of pheromone laid on edge (i,j) by the m ants at that timestep is

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

- $\Delta\tau_{ij}^{k} = Q/L_k$ if kth ant uses edge (i,j) in its tour, else 0. Q is a constant and L_k is the length of k's tour. Pheromone density for k's tour.

- Initialise: set pheromone strength to a small value

- Transitions chosen to trade off visibility (choose close towns with high probability – greedy) and trail intensity (if there's been a lot of traffic the trail must be desirable).

- In one iteration all the ants build up their own individual tours (so an iteration consists of lots of moves/town choices/timesteps – until the tour is complete) and pheromone is laid down once all the tours are complete

- Remember: we're aiming for the shortest tour – and expect pheromone to build up on the shortest tour faster than on the other tours

Algorithm

- Position ants on different towns, initialise pheromone intensities on edges.
- Set first element of each ant's tabu list to be its starting town.
- Each ant moves from town to town according to the probability $p(i, j)$
- After n moves all ants have a complete tour, their tabu lists are full; so compute L_k and $\Delta\tau_{ij}^k$. Save shortest path found and empty tabu lists. Update pheromone strengths.
- Iterate until tour counter reaches maximum or until *stagnation* – all ants make same tour.

Can also have different pheromone-laying procedures, e.g. lay a certain quantity of pheromone Q at each timestep, or lay a certain density of pheromone Q/d_{ij} at each timestep.

Applications

- Bus routes, garbage collection, delivery routes
- Machine scheduling: Minimization of transport time for distant production locations
- Feeding of lacquering machines
- Protein folding
- Telecommunication networks: Online optimization
- Personnel placement in airline companies
- Composition of products

Tweaks

This works well for short tours, say up to about 30 cities

BUT try including exploration and exploitation

ACS algorithm

- ϵ of the time (say 0.1), from city i choose a random allowed city j to move to.
- Rest of time choose greedily among cities using $\max_j p(i, j)$.
- Global updating rule: only globally best ant gets to deposit pheromone at the end of each iteration (i.e. the ant with the shortest tour so far)
- Local updating rule: while building a solution ants change pheromone on edges visited in a similar fashion but with $\Delta\tau(i, j) = \tau_0 = 1/nL$, a constant ($n$ is the number of cities and L is an approximation of the optimal tour length as found e.g. by a greedy algorithm). (Also Ant-Q, with a Q-learning-like update rule.)

Performance

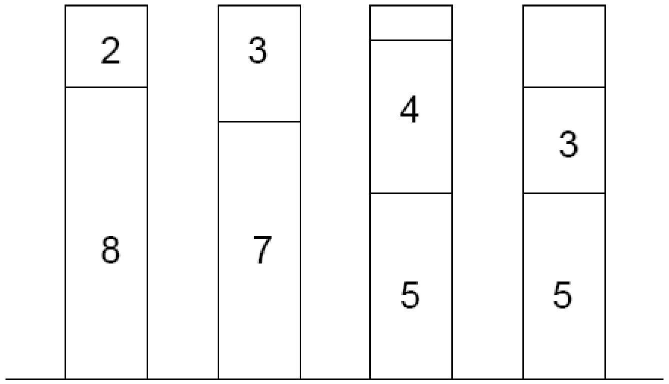
Problem	ACS (avge)	SA (avge)	EN (avge)	SOM (avge)
50-city set 1	5.88	5.88	5.98	6.06
50-city set 2	6.05	6.01	6.03	6.25
50-city set 3	5.58	5.65	5.70	5.83
50-city set 4	5.74	5.81	5.86	5.87
50-city set 5	6.18	6.33	6.49	6.70

ACS – ant colony system, SA–simulated annealing, EN–elastic net, SOM–self-organising map

From Dorigo and Gambardella: Ant Colony System: A cooperative learning approach to the TSP. IEEE Trans. Evol. Comp 1 (1) 53–66 1997.

Can do larger problems, e.g. finds optimal in 100-city problem KroA100, close to optimal on 1577-city problem fl1577.

Bin Packing Problems



- Packing a number of items in bins of a fixed capacity
- Bins have capacity C, set of items S with size/weight w_i
- Pack items into as few bins as possible
- Lower bound on no. bins: $L_1 = \lceil \sum w_i / C \rceil$ ($\lceil x \rceil$ is smallest integer $\geq x$)
- Slack $= L_1 C - \sum w_i$

Solving the BPP

- Greedy algorithm: first fit decreasing (FFD):
  - Order items in order of non-increasing weight/size
  - Pick up one by one and place into first bin that is still empty enough to hold them
  - If no bin is left that the item can fit in, start a new bin
- Or apply Ant Colony Optimisation: what is the trail/pheromone? what is the "visibility"?

Applying ACO to the BPP

1. How can good packings be reinforced via a pheromone matrix?
2. How can the solutions be constructed stochastically, with influence from the pheromone matrix and a simple heuristic?
3. How should the pheromone matrix be updated after each iteration?
4. What fitness function should be used to recognised good solutions?
5. What local search technique should be used to improve the solutions generated by the ants?

AntBin: Levine and Ducatelle

AntBin 1: Pheromone Matrix

- BPP as an ordering problem? TSP is an ordering problem – put cities into some order. But in BPP many orderings are possible:

$$\begin{aligned} &|82|73|54|53| \\ =\ &|53|73|82|54| \\ =\ &|35|73|28|54| \end{aligned}$$

- BPP as a grouping problem? $\tau(i,j)$ expresses the favourability of having items of size i and j in the same bin – **possibly**
- Pheromone matrix works on item sizes, not items themselves
- There can be several items of **size** i or j, but there are fewer item sizes than there are items, so small pheromone matrix
- Pheromone matrix encodes good packing patterns – combinations of sizes

AntBin 2: Building Solutions

- Every ant k starts with an empty bin b
- New items j are added to k's partial solution s stochastically:

$$p_k(s,b,j) = \frac{[\tau_b(j)]^\alpha . [\eta(j)]^\beta}{\sum_{g \in \text{allowed}} [\tau_b(g)]^\alpha . [\eta(g)]^\beta}$$

- The allowed items are those that are still small enough to fit in bin b.
- $\eta(j)$ is the weight/size of the item, so $\eta(j) = j$ – prefer largest
- $\tau_b(j)$ is the sum of pheromone between item of size j and the items already in bin b divided by the number of items in bin b
- α and β are empirical parameters, e.g. 1 and 2, giving the relative weighting of local and global terms

AntBin 3: Pheromone Updating

- Pheromone trail evaporates a small amount after every iteration (i.e. when all ants have solutions)

$$\tau(i,j) = \rho\, \tau(i,j) + m\, f(s_{\text{best}})$$

- Minimum pheromone level set by parameter τ_{min}, evaporation parameter ρ
- The pheromone is increased for every time items of size i and j are combined in a bin in the best solution (combined m times)
- Only the iteration best ant increases the pheromone trail (quite aggressive, but allows exploration)
- Occasionally (every γ iterations) update with the global best ant instead (strong exploitation)

AntBin 4: Evaluation Function

- Total number of bins in solution? Would give an extremely unfriendly evaluation landscape – no guidance from $N+1$ bins to N bins – there may be many possible solutions with just one bin more than the optimal
- Need large reward for full or nearly full bins

$$f(s_k) = \frac{\sum_{b=1}^{N} (F_b/C)^2}{N}$$

N the number of bins in s_k, F_b the sum of items in bin b, C the bin capacity

- Includes how full the bins are and number of bins
- Promotes full bins with the spare capacity in one "big lump" not spread among lots of bins

Improvement Through Local Search

- While building tour, apply an improvement heuristic at each step to each ant's partial tour.
- For example: use 3-opt: cut the tour in three places (remove three links) and attempt to connect up the cities in alternative ways that shorten the path.
- Reduces time, almost always finds optimal path.

AntBin 5: Local Search

- In every ant's solution, the n_{bins} least full bins are opened and their contents are made free
- Items in the remaining bins are replaced by larger free items
- This gives fuller bins with larger items and smaller free items to reinsert
- The free items are reinserted via FFD (first-fit-decreasing)
- The procedure is repeated until no further improvement is possible
- Deterministic and fast local search procedure
- ACO gives coarse-grained search, local search gives finer-grained search

Setting the Parameters

- Ducatelle used 10 existing problems for which solutions known to investigate parameter setting
- $\beta = 2$ • $n_{\text{ants}} = 10$
- $n_{\text{bins}} = 3$ to be opened in local search
- $\tau_{\min} = 0.001$ • $\rho = 0.75$
- Alternate global and iteration best ant laying pheromone 1/1
- $n_{\text{iter}} = 50000$
- Local search: replace 2 current items by 2 free items; then 2 current by 1 free; then 1 current by 1 free

Comparison Results

Compare with published solutions to some standard problems (see reference). For how many problem instances does AntBin find the optimal solution?

	N_{inst}	Scholl	Flszar	Alvim	AntBin
uniform	80	-	79	79.6	79.2
triplets	80	-	80	80.0	7.0
scholl_1	720	709	710	718.8	719.0
scholl_2	480	476	477	480.0	477.0
scholl_3	10	10	9	10.0	10.0
schwerin	200	-	-	200.0	200.0
waescher	17	-	-	10.0	13.1

Triplets hard. Optimal solution: 3 items, two smaller than $\frac{1}{3}$ bin, one larger than $\frac{1}{3}$ bin.

Can fit 3 small items in bin or 2 large items. Optimal solution often missed.

Applying Ant Methods to Optimisation

What we need to set up an ACO

- *Problem representation* that allows the solution to be built up incrementally
- *Desirability heuristic* η to help in building up the solution
- *Constraints* that permit only feasible/valid solutions to be constructed
- *Pheromone update rule* incorporating quality of the solution
- *Probability rule* that is a function of desirability and pheromone strength

Points

Encourage ants to follow best tour by getting best ant to lay pheromone (global-best ant or, in some versions of ACO, iteration-best ant). Ants search in the neighbourhood of this tour.

Local Updating (the ants lay pheromone as they go along without waiting till end of tour). Can set up the evaporation rate so that local updating "eats away" pheromone, and thus visited edges are seen as less desirable, encourages exploration. (Because the pheromone added is quite small compared with the amount that evaporates.)

Heuristic (meta-)improvements like 3-opt – not really "ant"

"Guided parallel stochastic search in region of best tour" [Dorigo and Gambardella]

Reading: 2 Dorigo papers linked to web page

Next: ACO applied to other problems