Genetic Algorithms and Genetic Programming Lecture 14

Gillian Hayes

14th November 2006

School of informatics

Ant Colony Optimisation and Bin Packing Problems

- Ant Colony Optimisation - review
- Pheromone Trail and Heuristic
- The Bin Packing Problem (BPP)
- Applying ACO to the BPP
- Adding a local search procedure
- Comparing to other approaches
- Summing Up

Ant Colony Optimisation

Probability of choosing next town in TSP (see lecture 13)

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

Two components:

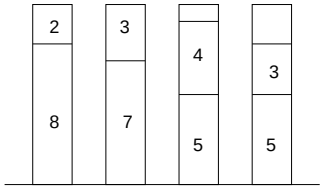
- Strength of pheromone $\tau(i,j)$ is favourability of j following i
- Visibility $\eta(i,j) = 1/d(i,j)$ is a simple heuristic guiding construction

Need to find an interpretation and formula for pheromone strength deposited at each iteration – "positive feedback process"

Need to find an interpretation and formula for the heuristic – the "greedy force"

See $\tau(i,j)$ as the components of a matrix: the **pheromone matrix**

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:

$$|82|73|54|53|$$
$$= |53|73|82|54|$$
$$= |35|73|28|54|$$

- BPP as a grouping problem? $\tau(i,j)$ expresses the favourability of having items of size i and j in the same bin – YES
- 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:

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

- 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
- β is an empirical parameter, e.g. 2

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

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
- $\tau_{\text{min}} = 0.001$ • $\rho = 0.75$
- 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.

Waescher and Gau New Optima

Instance	Bins	Iterations	Time
TEST0082	24	21	0.75s
TEST0058	20	344	9.95s
TEST0005	28	2465	81.4s
TEST0030	27	28957	941.5s
TEST0014	23	1226866	37451s

See G. Waescher, T. Gau: Heuristics for the integer one-dimensional cutting stock problem: A computational study. OR Spectrum, 18(3), 131–144, 1996. Randomly generates 4000 cutting stock problems.

See reading for comparison with other algorithms and application to Cutting Stock Problem.

Conclusions

- Competitive algorithm for some problem classes
- Does very badly at triplets and other zero-slack problems due to the simple local search being used
- Can solve both BPP and cutting-stock problem
- Does very well at Waescher and Gau's (1996) problem set, identifies 5 new optima
- How to improve:
  - Do better local search
  - Put onto multiprocessors

Reading: Levine and Ducatelle paper linked to web page