**ECO Notes**

List of topics

**Algorithms**

* Local search algorithms
* Population Based search algorithms
  + Including Multi-objective
* Swarm Algorithms
  + ANT/PSO
* Hyper Heuristics

**Concepts/Methodology**

* Why some problems
* Concepts relating to search : exploitation/exploration
* Designing an algorithm
* Maintaining diversity
* Evaluating an algorithm
* Practical examples

Why are problems hard

* Size of search space
* Constraints
* Dynamic nature of environment
* Hard to find feasible solutions
* Only ‘models’ available (and they might not be very good)
* Examples of real world problems

Search and search spaces

* Characteristics of search spaces
* Representing problems as search problems
* Concepts:
  + Optima
  + Local optima
  + Exploration
  + Exploitation

Local Search Algorithms

* Hillclimbers; Simulated Annealing: Tabu Search
* For all above:
  + Representation
  + Move operators/acceptance criteria
  + Main concepts
* Advantages/Disadvantages
* Comparisons

Population based algorithms

* Population based algorithms
  + Evolutionary Algorithms (includes Multi-Objective, map elites, novelty search)
    - Comparisons: similarities and differences
    - Advantages compared to non-population based algorithms
    - Relative strengths/weakness
  + Evolution Strategies
  + Ant Colonies/PSO
* Hyper-heuristics

Population Algorithms

* Representations(direct, indirect)
* Operators & Role
  + Crossover/Mutation/Selection
  + Examples of each
  + Role: Exploration/Exploitation
* Approaches to maintain diversity
  + Why its important
  + Implicit & Explicit Methods
  + Quality-Diversity algorithms: Novelty search, map-elites
* Multi-Objective Algorithms

Experimental & Practical Issues

* Design vs repetitive problems
* Evaluation :
  + Need to repeat experiments
* Measures of algorithm quality(success rate etc)
* Statistical concepts/metrics
* Data: benchmarks, real-world problems
* Multi-objective trade offs

Examples

* Examples
  + From lectures: TSP, bin-packing, knapsack
* From reading list: hearing-aid implants
* Issues concerning practical application of EC techniques
  + Acceptance by industry
  + Factors affecting uptake
    - Stochasticity/black-box.

**Notes from Lectures**

Problems vs Problem solvers

* Problem Instance : A particular problem we want to find a solution to.
* An instance has many possible solutions
  + Some good
  + Some bad
* The problem solver searches though the set of solutions to find a good one
* Optimisation = search

Black Box problem solvers

* Feed in an instance of the problem.
* Outputs the answer
* We don’t necessarily how that answer was determined.
* Blackbox algorithms are useful solvers:
  + The same algorithm can be applied to many instances without changing the algorithm
* But they can be mistrusted
  + Don’t know how a solution was formed
  + Don’t understand the process
  + Don ‘t know if the solution is reliable/robust
* Despite this, they are very commonly used.
* Ingredients: black box optimisers
  + Input
    - A method of representing a possible solution to our problem.
  + Quality Measure
    - A way of assigning a score(fitness) to a solution to indicate how good it is
  + Solver
    - Searches to find the ‘best’ solution for our problem.
    - The search-space defines the set of all possible solutions
  + Output
    - Best solution for our problem

Representation & Search Space

* The representation is a method of writing down a potential solution to a problem

Solution Quality

* The quality measure depends on the problem objective: what you are trying to achieve:
  + In the TSP, minimise the distance travelled
* An evaluation (fitness) function allows you to
  + Quantify the quality of possible solution
  + Compare the quality of a number of solutions
  + It is helpful if the evaluation function tells you how much better a solution is that another.

Stochastic Local Search

* Simple stochastic algorithms for solving problems:
  + Hill climbers
  + Simulated Annealing
  + Tabu Search
* Often naïve, but very good way of obtaining near – optimal solutions
* Doesn’t guarantee best solution

Search Spaces

* Global optima is the best solution
* May be multiple optima
* There may be many local optima
* Smooth search space landscapes are relatively easy to search
* Rugged (noisy) search spaces are much more difficult
  + Many peaks so no obvious path to global optima

Hill climbing

* Start at a random place in the landscape
* Move uphill though the solution space towards better solutions
* Stages of a hillclimbers:
  + 1. Pick a solution from the search space , x. Evaluate it.
  + 2. Create a solution y by applying a move operator to x.
  + 3. Evaluate the new solution y.
  + 4. If y is better than x, replace x with y, else discard it.
  + 5. Return to step 2 until some stopping criterion applies
* Iterated hill climbing – Repeat several hill climbs, each time starting from a different point in the search space.

Hill climbing advantages and disadvantages

* It works well if there are not too many local optima.
* But if the fitness function is very ‘noisy’ with many small peaks, stochastic hill climbing is not good
  + In general don’t know if fitness function is noisy
* Very easy to implement and often finds fairly good solutions quickly

Effect of neighbourhood size

* Small neighbourhoods can be searched quickly but high risk of getting stuck in a local optimum
* Large neighbourhoods are expensive to search but less risk of getting stuck
* Choice depends on problem

Simmulated Annealing

* It works by searching neighbourhoods via a move operator
* Reduces the chance of getting stuck in a local optima
  + Down aswell as uphill
* Stages of simulated annealing
  + 1. Pick a solution and evaluate it. Call this the current solution
  + 2. Apply a move operator to get a new solution and evaluate it.
  + 3. If the new solution is better than the current solution keep it
  + 4. If it is worse, accept it with some small probability that decreases over time
  + 5. Decrease the probability of accepting bad solutions
  + 6. Return to step 2 until some stopping criterion applies
* Deciding to accept solutions
  + The probability of accepting a worse move is determined by the system temperature.
  + At high temperatures there is a high probability of accepting inferior moves
  + Accepting inferior moves allows the system to escape from local optima

Advantages/Disadvantages of SA

* Advantages
  + SA is still very simple to implement
  + Shown to give good results over a large spectrum of difficult problems
  + Theoretically, shouldn’t get stuck as can move out of local optima
* Disadvantages
  + More parameters to tune than HC
  + Lack of memory – information learned during the search process always discarded.
  + Can get stuck in cycles

Tabu search

* Extends local search by adding memory:
  + Maintains a list of recent decisions made by the algorithm
  + Is used to guide the search away from previously visited areas
    - Adds diversity to the search
    - Avoids cycling
* The set of moves which are forbidden is called the ‘tabu list’
* Memory sized fixed and continuously updated so moves become allowed later
* Inferior solutions selected if all the better ones are tabu.
* Specify a neighbourhood search algorithm by
  + An initial state
  + A search neighbourhood
  + A move evaluator
  + Tabu criteria
  + Aspiration criteria
  + One or more tabu lists
* Tabu search Meta-heuristic
  + Construct a candidate list from the search neighbourhood
  + Pick a move that doesn’t violate tabu criteria.
    - But allow if it meets aspiration criteria
* Update tabu-list if necessary.
* Tabu criteria
  + Don’t allow reversal of moves recently made
  + Don’t allow move if solution quality is the same as on a previous iteration
  + Same solution has been visited recently
  + Don’t allow moves that have been made often
* Aspiration criteria (Long Term Memory)
* Based on quality:
  + Best in neighbourhood
  + Similar to existing solution (intensification)
  + Dissimilar to exiting solution (diversification)