Evolutionary Computing

Chapter 10

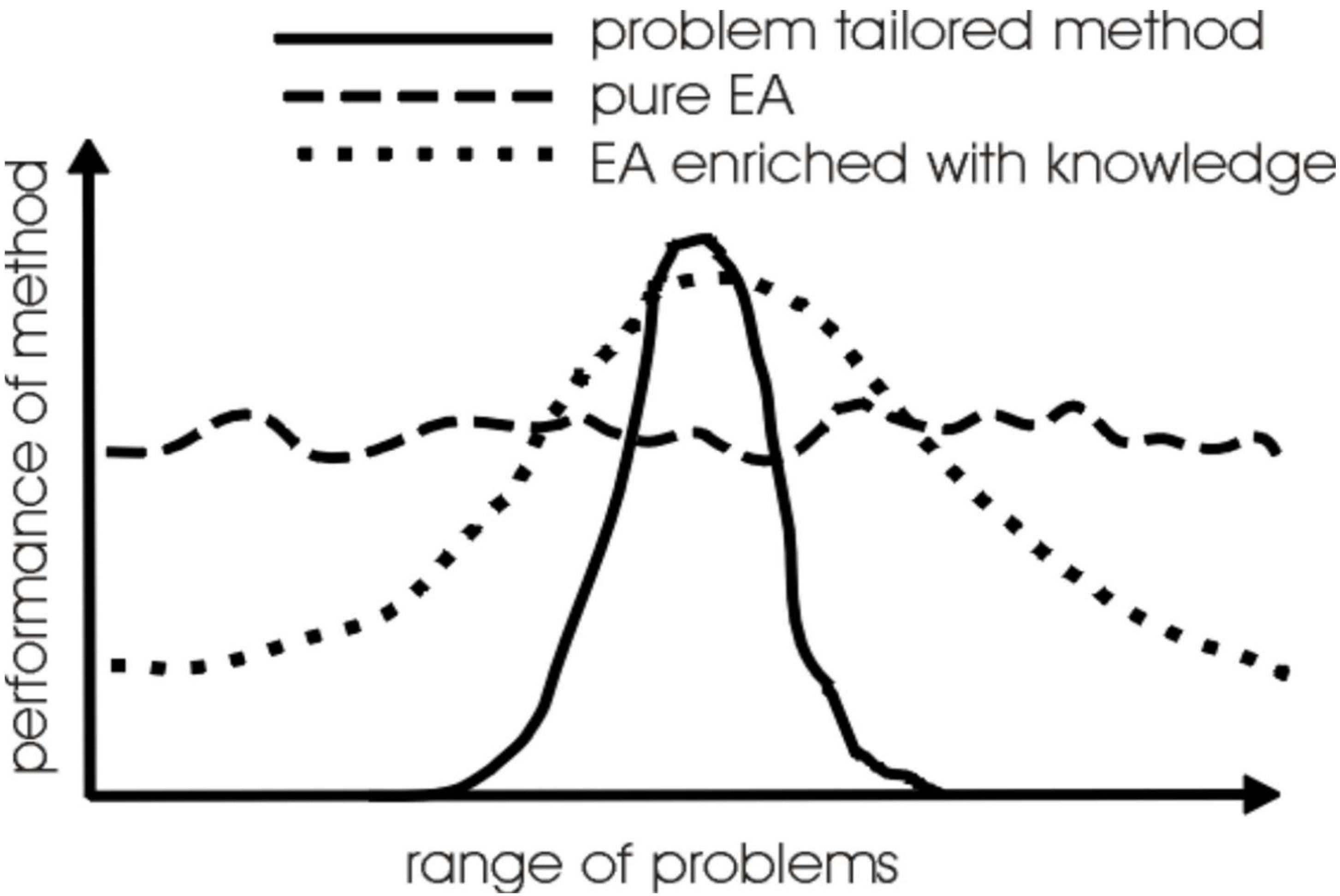
Chapter 10: Hybridisation with Other Techniques: Memetic Algorithms

- Why to Hybridise
- What is a Memetic Algorithm?
- Where to hybridise
- Incorporating good solutions
- Local Search and graphs
  - Lamarckian vs. Baldwinian adaptation
- Diversity
- Operator choice
- Adaptive Memetic Algorithm

Why Hybridise

- Might want to put in EA as part of larger system
- Might be looking to improve on existing techniques but not re-invent wheel
- Might be looking to improve EA search for good solutions

Why Hybridise
Michalewicz's view on EAs in context

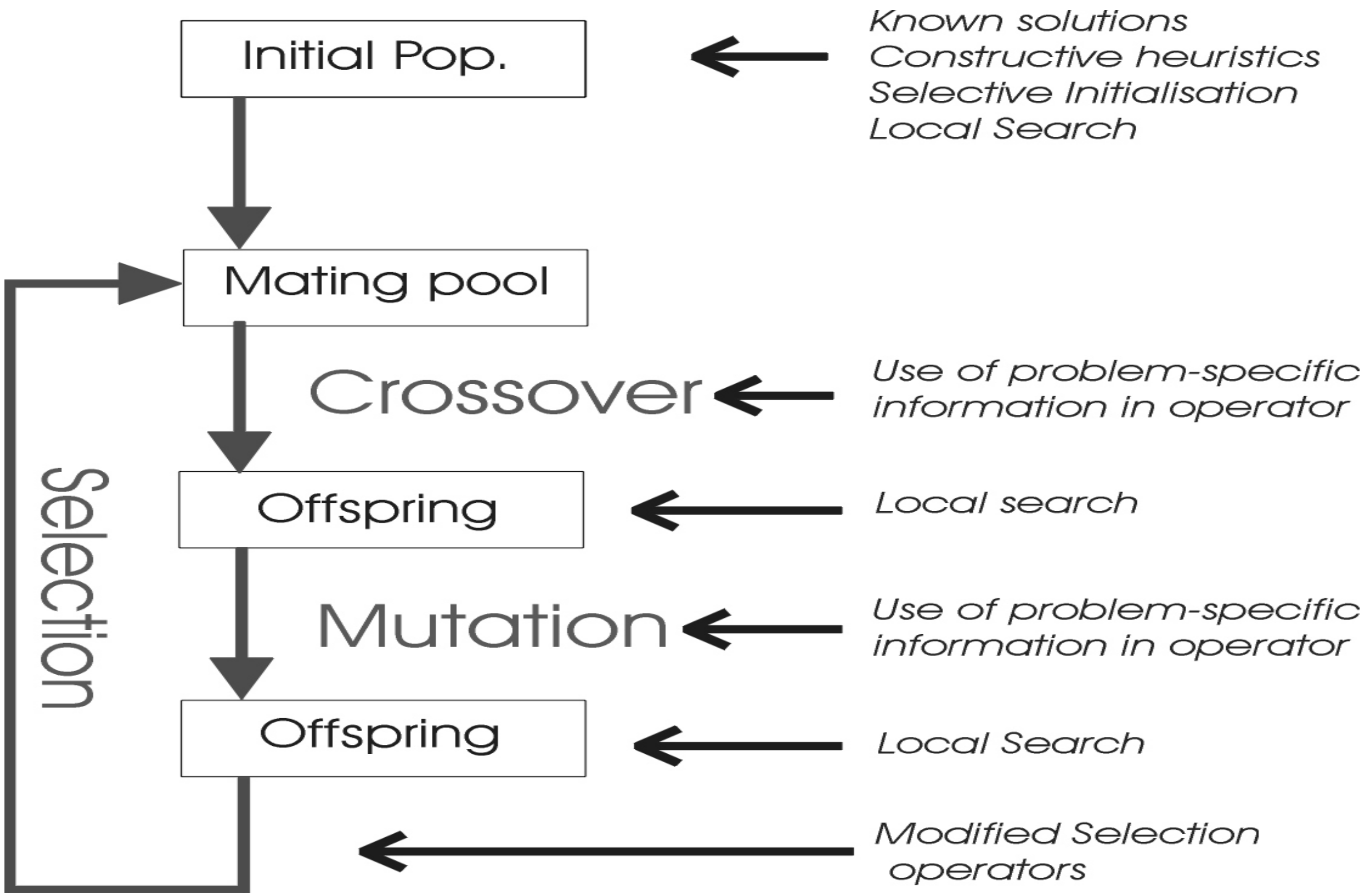


Adapted from A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing 2014

What is a Memetic Algorithm?

- The combination of Evolutionary Algorithms with Local Search Operators that work within the EA loop has been termed “Memetic Algorithms”
- Term also applies to EAs that use instance-specific knowledge in operators
- Memetic Algorithms have been shown to be orders of magnitude faster and more accurate than EAs on some problems, and are the “state of the art” on many problems

Where to Hybridise:



Adapted from A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing 2014

Incorporating good solutions: Heuristics for Initialising Population

- Bramlette ran experiments with limited time scale and suggested holding a *n*-way tournament amongst randomly created solutions to pick initial population

  (n.b. NOT the same as taking the best *popsize* of *n.popsize* random points)

- Multi-Start Local Search is another option: pick *popsize* points at random to climb from

- Constructive Heuristics often exist

Incorporating good solutions: Initialisation Issues

- Another common approach would be to initialise population with solutions already known, or found by another technique (beware, performance may appear to drop at first if local optima on different landscapes do not coincide)

- Surry & Radcliffe (1994) studied ways of “inoculating” population with solutions gained from previous runs or other algorithms/heuristics
  - found ***mean*** performance increased as population was biased towards known solutions,
  - but ***best*** performance came from more random solutions

Incorporating good solutions: “Intelligent” Operators

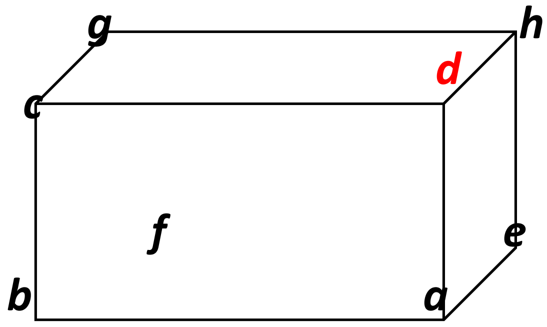
- It is sometimes possible to incorporate problem or instance specific knowledge within crossover or mutation operators
  - E.g. Merz’s DPX operator for TSP inherits common sub tours from parents, then connects them using a nearest neighbour heuristic
  - Smith (97) evolving microprocessor instruction sequences: group instructions (alleles) into classes so mutation is more likely to switch gene to value having a similar effect
  - Many other examples in literature

Incorporating good solutions
Local Search Acting on Offspring

- Can be viewed as a sort of "lifetime learning"
- Lots of early research done using EAs to evolve the structure of Artificial Neural Networks and then Back-propagation to learn connection weights
- Often used to speed-up the "endgame" of an EA by making the search in the vicinity of good solutions more systematic than mutation alone

Local Search and graphs: Local Search

- Defined by combination of ***neighbourhood*** and ***pivot rule***
- Related to landscape metaphor
- *N(x)* is defined as the set of points that can be reached from *x* with one application of a move operator
  - e.g. bit flipping search on binary problems



N(d) = {a,c,h}

Local Search and graphs: Landscapes & Graphs

- The combination of representation and operator defines a graph *G(V,E)* on the search space (useful for analysis)
- *V*, the set of vertices, is the set of all points that can be represented (the potential solutions)
- *E*, the set of edges, is the possible transitions that can arise from a single application of the operator
    - note that the edges in *E* can have weights attached to them, and that they need not be symmetrical

Local Search and graphs: Example Graphs for Binary Problems

- Example : 3 dimensional binary problem as above
  - *V = {a,b,c,d,e,f,g,h,}*
  - Search by flipping each bit in turn
    - E_1 = *{ ab, ad, ae, bc, bf, cd, cg, dh, fg, fe, gh, eh}*
    - symmetrical and all values equally likely
    - E_2 = *{ac,bd,af,be,dg, ch, fh, ge, ah, de, bg, cf}*
    - E_3 = *{ag, bh, ce, df}*
  - Bit flipping mutation with prob *p* per bit implies weights for edges

Local Search and graphs: Graphs

- The ***Degree*** of a graph is the maximum number of edges coming into/out of a single point, - the size of the biggest neighbourhood
  - single bit changing search: degree is l
  - bit-wise mutation on binary: degree is 2^l -1
  - 2-opt: degree is $O(N^2)$
- Local Search algorithms look at points in the neighbourhood of a solution, so complexity is related to degree of graph

Local Search and graphs: Pivot Rules

- Is the neighbourhood searched randomly, systematically or exhaustively ?
- does the search stop as soon as a fitter neighbour is found (***Greedy Ascent***)
- or is the whole set of neighbours examined and the best chosen (***Steepest Ascent***)
- of course there is no one best answer, but some are quicker than others to run ....

Local Search and graphs: Variations of Local Search

- Does the search happen in representation space or solution space ?
- How many iterations of the local search are done ?
- Is local search applied to the whole population?
  - or just the best ?
  - or just the worst ?

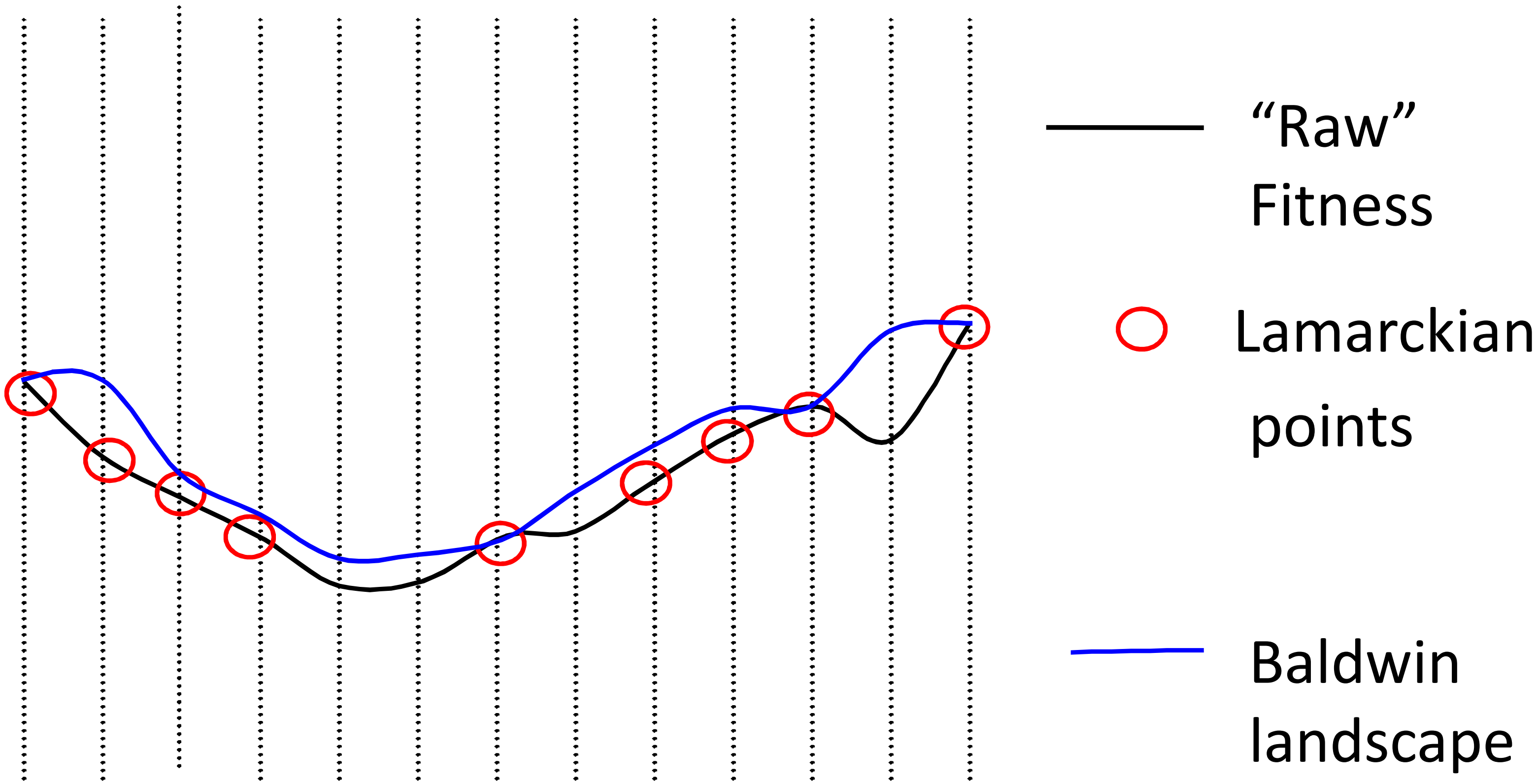
Local Search and graphs: Two Models of Lifetime Adaptation

- Lamarckian
  - traits acquired by an individual during its lifetime can be transmitted to its offspring
  - e.g. replace individual with fitter neighbour
- Baldwinian
  - traits acquired by individual cannot be transmitted to its offspring
  - e.g. individual receives fitness (but not genotype) of fitter neighbour

Local Search and graphs: The Baldwin effect

- LOTS of work has been done on this
  - the central dogma of genetics is that traits acquired during an organisms lifetime ***cannot*** be written back into its gametes
  - e.g. Hinton & Nowlan '87, ECJ special issue etc
- In MAs we are not constrained by biological realities so can do Lamarckism

Local Search and graphs: Induced landscapes



Local Search and graphs: Information Use in Local Search

- Most Memetic Algorithms use an operator acting on a single point, and only use that information
- However this is an arbitrary restriction
    - Jones (1995), Merz & Friesleben (1996) suggest the use of a crossover hillclimber which uses information from two points in the search space
    - Krasnogor & Smith (2000) - see later - use information from whole of current population to govern acceptance of inferior moves
    - Could use Tabu search with a common list

Diversity

- Maintenance of diversity within the population can be a problem, and some successful algorithms explicitly use mechanisms to preserve diversity:
  - Merz's DPX crossover explicitly generates individuals at same distance to each parent as they are apart
  - Krasnogor's Adaptive Boltzmann Operator uses a Simulated-Annealing like acceptance criteria where "temperature" is inversely proportional to population diversity

Diversity: Boltzman MAs: acceptance criteria (1/2)

- Assuming a maximisation problem,

  Let Δf = fitness of neighbour – current fitness

$$P(accepting\ \ neighbour) = \begin{cases} 1 & \Delta f > 0 \\ e^{\frac{k\Delta f}{f_{\max} - f_{avg}}} & \Delta f < 0 \end{cases}$$

Diversity: Boltzman MAs: acceptance criteria (2/2)

- Induced dynamic is such that:
  - Population is diverse => spread of fitness is large, therefore *temperature* is low, so only accept improving moves => *Exploitation*
  - Population is converged => temperature is high, more likely to accept worse moves => *Exploration*
- Krasnogor showed this improved final fitness and preserved diversity longer on a range of TSP and Protein Structure Prediction (PSP) problems

Choice of Operators

- There are theoretical advantages to using a local search with a move operator that is DIFFERENT to the move operators used by mutation and crossover cf. Krasnogor (2002)
- Can be helpful since local optimum on one landscape might be point on a slope on another
- Easy implementation is to use a range of local search operators, with mechanism for choosing which to use. (Similar to Variable Neighbourhood Search)
- This could be learned & adapted on-line (e.g. Krasnogor & Smith 2001)

Hybrid Algorithms Summary

- It is common practice to hybridise EA's when using them in a real world context.
- This may involve the use of operators from other algorithms which have already been used on the problem (e.g. 2-opt for TSP), or the incorporation of domain-specific knowledge (e.g. PSP operators)
- Memetic algorithms have been shown to be orders of magnitude faster and more accurate than GAs on some problems, and are the "state of the art" on many problems

Adaptive Memetic Algorithm

- Most important in MA incorporating local search or heuristic improvement is choice of improving move operator

- Careful consideration
  – Using domain-specific information
  – Use of multiple local search operators in tandem
  – Adding a gene indicating which local search operator to use (inherited from parents, subject to mutation)

Adaptive Memetic Algorithm MA generations

- Meuth et al. defined different MA generations:
  - First: “Global search paired with local search”
  - Second: “Global search with multiple local optimizers. Memetic information (choice of optimizer) passed to offspring (Lamarckian evolution)”
  - Third: “Global search with multiple local optmizers. Memetic information (choice of local optimizer) passes to offspring (Lamarckian evolution). A mapping between evolutionary trajectory and choice of local optimizer is learned”

Warning: Memetic Overkill

- Craenen and Eiben (CEC 2005) solve CSPs with hybrid EAs, i.e., memetic algorithms
- 3 out of best 4 MAs become better after “switching off evolution”:
  - No selection (uniform random choices)
  - No population (pop size = 1)
- Irony: heuristics were added to EAs to improve them, removing the “E” gives the best result