Memetic Algorithms

A Tutorial given in the Seventh International Conference on Parallel Problem Solving From Nature (PPSN VII). Granada, Spain, September 2002

Natalio Krasnogor

Automated Scheduling, Optimisation and Planning Group
and
Computational Biophysics and Chemistry Group
University of Nottingham
Nottingham,U.K.
dirac.chem.nott.ac.uk/~natk/Public/index.html
Natalio.Krasnogor@nottinghahm.ac.uk

This tutorial is divided in five parts:

1. **PART 1:** Operational definition of Memetic Algorithms, overview of a few success / documented stories. (≈ 25 mins.)

2. **PART 2:** Abstract Conceptual framework (Syntactic model + taxonomy) (≈ 25 mins). *Coffee Break, (≈ 10 mins.)*

3. **PART 3:** Design issues for a robust engineering of competent MAs. (≈ 30 mins.)

4. **PART 4:** Revisiting the "Memetic" metaphor, future lines of research and Q&A. (≈ 30 mins.)

5. **PART 5:** Open discussion. (≈ 15 mins.)

TOTAL TIME ≈ 135 mins.

But First....

Acknowledgments

To D.A. Pelta, W.E. Hart, P. Merz, J. Knowles, J.E. Smith, E.K. Burke, J.D. Hirst, S. Gustafson, S. Ahmadi, M. Land for their work and their helpful (sometimes daily) discussions.

To the researchers that by submitting their papers to WOMA made this workshop *THE* MA workshop.

Part 1

- Operational definition for Memetic Algorithms

- Overview of a few success / documented stories.

Memetic Algorithms:

Evolutionary algorithms which include a stage of individual optimization or learning (usually in the form of local search) as part of their search strategy are Memetic Algorithms.

Some, but not all, of the earliest references can be traced back to:
[HN87],[MGSK88],[JSG89],[Mos89].

- GA+LS were named MAs by Moscato in [Mos89]

- GA+LS were present from the very beginnings of EC

- MAs are also called: Hybrid Genetic Algorithms, Genetic Local Search, Lamarckian Genetic Algorithms, Baldwinian Genetic Algorithms.

- MAs are inspired by models of adaptation in natural systems that combine evolutionary adaptation of populations of individuals(EA) with individual learning within a lifetime(LS). Others consider the LS process as *development*.

- MAs are inspired by Dawkin's Meme which represents a unit of cultural evolution that can exhibit refinement. **So far, MAs have not been used in this sense!**

The most basic MA can be seen below:

```
Memetic_Algorithm():
Begin
  t = 0;
  /* Initialize the evolutionary clock (generations) */
  Generate an initial population P(t);
  Repeat Until ( Termination Criterion Fulfilled ) Do
    Recombine;
    Mutate;
    Improve_by_local_search;
    Select_for_next_generation;
    t = t + 1;
  Od
  Return best solution(s);
End.
```

Some Examples From the Literature:

In [AV97] a short review on early MAs for the TSP is presented, where an MA was defined by the following skeleton code:

```
Genetic_Local_Search(P ∈ S^l):
Begin
  /* λ, μ, m ≥ 1 */
  For i := 1 To μ Do
    Iterative_Improvement(s_i);
  Od
  stop_criterion := false;
  While ( ¬ stop_criterion ) Do
    P' := ∅;
    For i := 1 To λ Do
      /* Mate */
      M_i ∈ P^m;
      /* Recombine */
      s_i ∈ H_m(M_i);
      /* Improve */
      Iterative_Improvement(s_i);
      P' := P' ∪ {s_i};
    Od
    /* Select */
    P :∈ (P ∪ P')^μ;
    evaluate stop_criterion;
  Od
End.
```

GLS_Based_Memetic_Algorithm[HM99]:

Begin

Initialize population;

For i := 1 **To** sizeOf(population) **Do**

$individual := population_i$;

$individual := Local - Search - Engine(individual)$;

$Evaluate(individual)$;

Od

Repeat Until (termination_condition) **Do**

For j := 1 **To** #recombinations **Do**

selectToMerge a set $S_{par} \subseteq population$;

$offspring = Recombine(S_{par}, x)$;

$offspring = Local - Search - Engine(offspring)$;

$Evaluate(offspring)$;

Add offspring to population;

Od

For j := 1 **To** #mutations **Do**

selectToMutate an individual in population;

$Mutate(individual)$;

$individual = Local - Search - Engine(individual)$;

$Evaluate(individual)$;

Add individual to population;

Od

$population = SelectPop(population)$;

Od

End.

In [FM96a]:

STSP-GA:
Begin
  Initialize population P with Nearest-Neighbor(...) ;
  For i := 1 **To** $popsize(P)$ **Do**
    $Lin - Kernighan - Opt(individuali), i \in P$;
  Od
  Repeat Until (converged) **Do**
    For i := 0 **To** #crossover **Do**
      Select two parents $i_a, i_b \in P$ randomly;
      $i_c = DPX - STSP(i_a, i_b)$;
      Lin-Kernighan-Opt(i_c) ;
      With probability m_p do Mutation-STSP(i_c);
      Replace an individual of P by i_c;
    Od
  Od
End.

In [FF93]:

Genetic_Hybrid_Algorithm(H_1, H_2):
Begin
  $P := \emptyset$;
  For i := 1 **To** m **Do**
    Generate a random permutation p;
    Add $H_1(p)$ to P;
  Od
  Sort P;
  For i := 1 **To** number_of_generations **Do**
    For j := 1 **To** number_of_offsprings **Do**
      select two parents p_1, p_2 from P;
      $child := crossover(p_1, p_2)$;
      Add $H_2(child)$ to P;
    Od
    Sort P;
    $Cull(P, number\_of\_offsprings\_per\_generation)$;
  Od
  Return the best $p \in P$;
End.

In [DH98]:

GL_for_Coloring:
Begin
/* f, F^*: fitness function and */
/* best value encountered so far */
/* s^*: best individual encountered so far */
/* $i, MaxIter$: current and */
/* maximum number of iterations */
/* $best(P)$: returns the best individual */
/* of the population P */
i=0;
generate(P_0);
$s^* := best(P_0)$;
$f^* := f(s^*)$;
While ($f^* > 0$ and $i < maxIter$) **Do**
$P'_i := crossing(P_i, T_x)$;
/* using specialized crossover */
$P_{i+1} := mutation(P'_i)$;
/* using tabu search */
If ($f(best(P_{i+1})) < f^*$) **Then**
$s^* := best(P_{i+1})$;
$f^* := f(s^*)$;
Fi
i := i+1;
Od
End.

In [KS00]:

```
PF_MA:
Begin
  Random initialize population Parents;
  Repeat Until ( Finalization_criteria_met ) Do
    Local_Search(Parents) ;
    mating_pool := Select_mating(Parents);
    offspring := Cross(mating_pool);
    Mutate(offspring);
    Parents := Select(Parents + offspring);
  Od
End.
```

Similarly, other works with MAs have used local search before or after mutation, before or after crossover.

In many cases the local search was a simple hill-climber algorithm (by means of a specific move operator), or more sophisticate methods like simulated annealing, tabu search, GRASP, FANS, etc.

On some cases the local search was used only to bias the selection procedure *a la Baldwin* (more on this later). However, the majority of MAs are *Lamarckian* MAs.

Design Issues for Memetic Algorithms Engineering

MA specific:

- Where/when do we apply local search?
- Which individuals in the population will be improved by local search?
- How long/wide should the local search be?
- What neighborhoods, acceptance criteria, etc the LS will use?
- How do we integrate the standard genetic operators with local search?
- Shall we use a Baldwinian or Lamarckian model for optimization?
- Which is the fitness landscape an MA is exploring?
- How do we engineer an MA that can jump out of deep local optima or traverse large neutral plateaus?
- What is needed for an MA to be able to cope with Multi-objective problems?
- How premature convergence can be averted?
- Help from theory?

EA general:

- Do we use specialized crossover,mutations?
- Do we use panmitic EA or geographically distributed or tree-topology or ..?
- Shall we used a problem-specific representation or would a binary representation be enough?
- etc.

Part 2

- Syntactic Model for Memetic Algorithms
- MAs taxonomy

The purpose of the next slides is to present an abstract formalism that captures the points made before.

With this formalism we will classify existing MAs according to their architecture.

We will also be able to discuss several architectural issues and venture into new regions of the design space of Memetic Algorithms.

A formalism for an Evolutionary Algorithm

$EA = (P^0, \delta^0, \lambda, \mu, l, F, G, U)$ where

- $P^0 = (a_1^0, \ldots, a_\mu^0) \in I^\mu$ Initial population of solutions
- $I = \{q_1, \ldots, q_n\}^l$ $n-$ary finite discrete problem representation
- $\delta^0 \subseteq \Re$ Initial operator parameter set
- $\mu \in \mathbf{N}$ Population size
- $\lambda \in \mathbf{N}$ Number of offspring
- $l \in \mathbf{N}$ Length of representation
- $F : I \mapsto \Re^+$ Fitness function (objective)
- $G : I^\mu \mapsto I^\lambda$ Generating Function
- $U : I^\mu \times I^\lambda \mapsto I^\mu$ Updating function

Examples of G as generating functions are mutation and crossover operators:

- $R : I^{\mu} \times \delta \mapsto I$

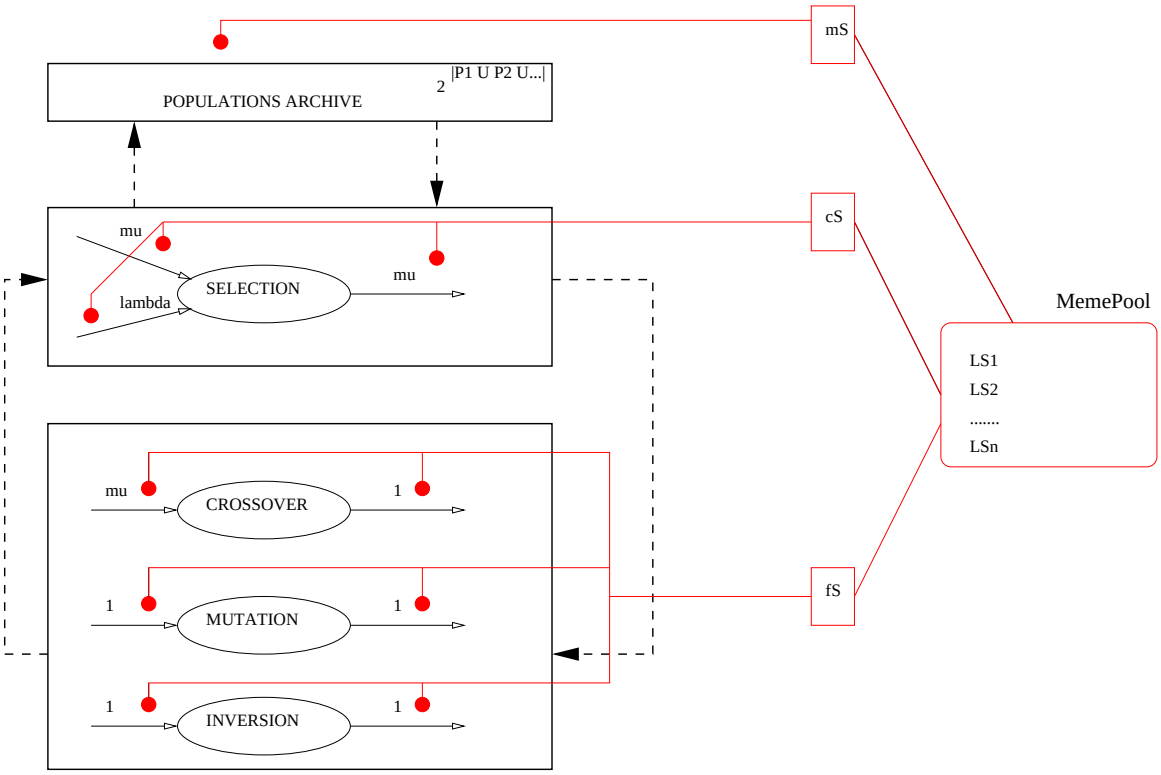
- $M : I \times \delta \mapsto I$.

If $O \in I^{\lambda}$ denotes the set of offspring then an iteration of the *GA* is given by:

$$\begin{aligned} O_i^t &= M(R(P^t, \delta^t), \delta^t) \\ & \quad \forall i \in \{1, \ldots, \lambda\} \\ P^{t+1} &= U(O^t \cup P^t), \end{aligned} \tag{1}$$

where t is the time step.

Overview of a Memetic Algorithm:



The local search operators will be members of a set, LSS, of available local search strategies to the MA:

$$LSS = \{L_1, \ldots, L_m\}$$

In the majority of MAs in the literature $1 \leq |LSS| \leq 2$

Regardless the number of *helpers/memes* at hand we need to coordinate their activities with the generating functions (i.e., crossover and mutation).

- The *fine grain scheduler* coordinates a helper's activity with crossover and mutation.

- The *coarse grain scheduler* organizes the application of a helper to the **set** of parents, offsprings or their union.

The fine grain scheduler, which coordinates the activity of a local search with a generating function, has the following signature:

$$fS : (I^{c1} \times \delta \mapsto I) \times LSS \times I^{c1} \times \delta \times \zeta \mapsto I$$

The coarse grain scheduler, that coordinates the activity of a local search and the update function, is defined by:

$$cS : (I^{\mu} \times I^{\lambda} \mapsto I^{\mu}) \times LSS \times I^{\mu} \times I^{\lambda} \times \delta \times \zeta \mapsto I^{\mu}$$

Why do we distinguish between $fS()$ and $cS()$?

- $fS()$ "knows" just one individual at a time (in the case when it coordinates the application of a helper with mutation) or two individuals (for the case of crossover).

- $cS()$ can provide population statistics to the helper it is coordinating with the update function.

Note also that both schedulers have access to the set of helpers, not just to a unique and fixed local searcher.

Then, an iteration of the *Memetic Algorithm* becomes:

$$\begin{aligned} O_i^t &= fS_M(M, L_j^{t,i}, fS_R(R, L_k^{t,i}, P^t, \delta^t, \zeta^t), \delta^t, \zeta^t) \\ &\quad \forall i \in \{1, \ldots, \lambda\} \\ P^{t+1} &= cS(U, L_h^{t,P^t,O^t}, P^t, O^t, \delta^t, \zeta^t) \\ &\quad j, k, h \in, \{1, \ldots, \mid LSS \mid\} \end{aligned} \tag{2}$$

and t as before.

A natural generalization on these concepts calls for a third kind of scheduler: *meta-scheduler*(mS)

- MAs that employ a mS can apply a helper/meme to an already selected population using information from archived populations: **evolutionary memory**

The *meta scheduler*(mS) has the following signature:

$mS : LSS \times H_P^t \times I^\mu \times \delta \times \zeta_g^t \mapsto I^\mu$ where $H_P^t \subseteq 2^{P_1 \cup P_2 \cup ... \cup P_{t-1}}$.

With the introduction of this scheduler, a new class of meta-heuristics is available given by the many possible instantiations of:

$$\begin{aligned} O_i^t &= fS_M(M, L_j^{t,i}, fS_R(R, L_k^{t,i}, P^t, \delta^t, \zeta^t), \delta^t, \zeta^t) \\ &\quad \forall i \in \{1, \ldots, \lambda\} \\ P^{t+1} &= mS(L_g^{t,H_P^t}, H_P^t, cS(U, L_h^{t,P^t,O^t}, P^t, O^t, \delta^t, \zeta^t), \\ &\delta^t, \zeta^t) \\ &\quad j, k, h, g \in \{1, \ldots, \mid LSS \mid\} \end{aligned} \tag{3}$$

and t as before.

With this model our MAs are (potentially) endowed with:

- A Population that evolves by means of Crossover, Mutation, Selection, i.e., the EA *stuff*
- A Population that is improved by means of one or more Memes
- Mechanisms (i.e., the schedulers) to coordinate the application of Memes
- Memes can be applied at the individual level, the population level, the populations archive level (PAL).
- The assignment of a meme to $\{individual, population, PAL\}$ can be static or dynamic.
- Memes can be:
  - Static : the local search always performs the same operations.
  - Adaptive : the local search parameters are adapted.
  - Self-Generating: the local search itself is generated, i.e., a *true meme*.

Taxonomy

We can go back to the MAs in the literature and classify them accordingly to what part(s) of the model presented here they use. We use the three schedulers as indexes into a taxonomic architecture of MAs:

if A is a MA then $D(A)$ is its index (binary representation) in the taxonomy.

$D(A) = b(mS), b(cS), b(fS_r), b(fS_m)$ where:

$$b_x = \begin{cases} 0 & \text{if scheduler x is absent} \\ 1 & \text{if scheduler x is present} \end{cases}$$

for $x \in \{mS, cS, fS_r, fS_m\}$.

The ordering of bits assigns the least significant to the fS associated to mutation, then to the one associated to crossover, then to the update function, then to PAL, in relation to the (potential) cardinality of the local searchers scheduled.

Classification of some MAs found in the literature for the following problems:

- $\triangle \equiv$ *TSP*
- $\diamondsuit \equiv$ *QAP*
- $\star \equiv$ *MGC*
- $\circ \equiv$ *BPQ*
- $\sharp \equiv$ *PFP* and *Protein Docking*
- $\bowtie \equiv$ *General Studies*
- $\oplus \equiv$ *Other Applications*

- **D=0 : EA**
  uncountable many papers.

- **D=1 : EA** + fS_M
  $\star$[CHD95], $\oplus$[BN99, MR99]

- **D=2 : EA** + fS_R
  $\triangle$[AV97, FM96a, FM96b, NK97, WRE$^+$98], $\diamondsuit$[FF93], $\star$[FF97, DH98]

- **D=3 : EA** + fS_C + fS_M
  $\triangle$[HM99, Mos99, MF97], $\bowtie$[MF98], $\oplus$[DAS97, BSS99]

- **D=4 : EA** + cS
  $\sharp$[KS00, RHHB97], $\triangle$[KS00],$\bowtie$[Har94, Lan98, TG97, MIG99]

- **D=5 : EA** + cS + fS_M
  $\oplus$[DBH98]

- **D=6 : EA** + cS + fS_X
  $\diamondsuit$[MF99a],$\circ$[MF99b]

- **D=7 : EA** $+ cS + fS_X + fS_M$
  $\bowtie$[Sal98],$\oplus$[VFJ]

- **D=8 : EA** $+ mS$
  none

- **D=9 : EA** $+ mS + fS_M$
  none

- **D=10: EA** $+ mS + fS_X$
  $\oplus$[KC00]

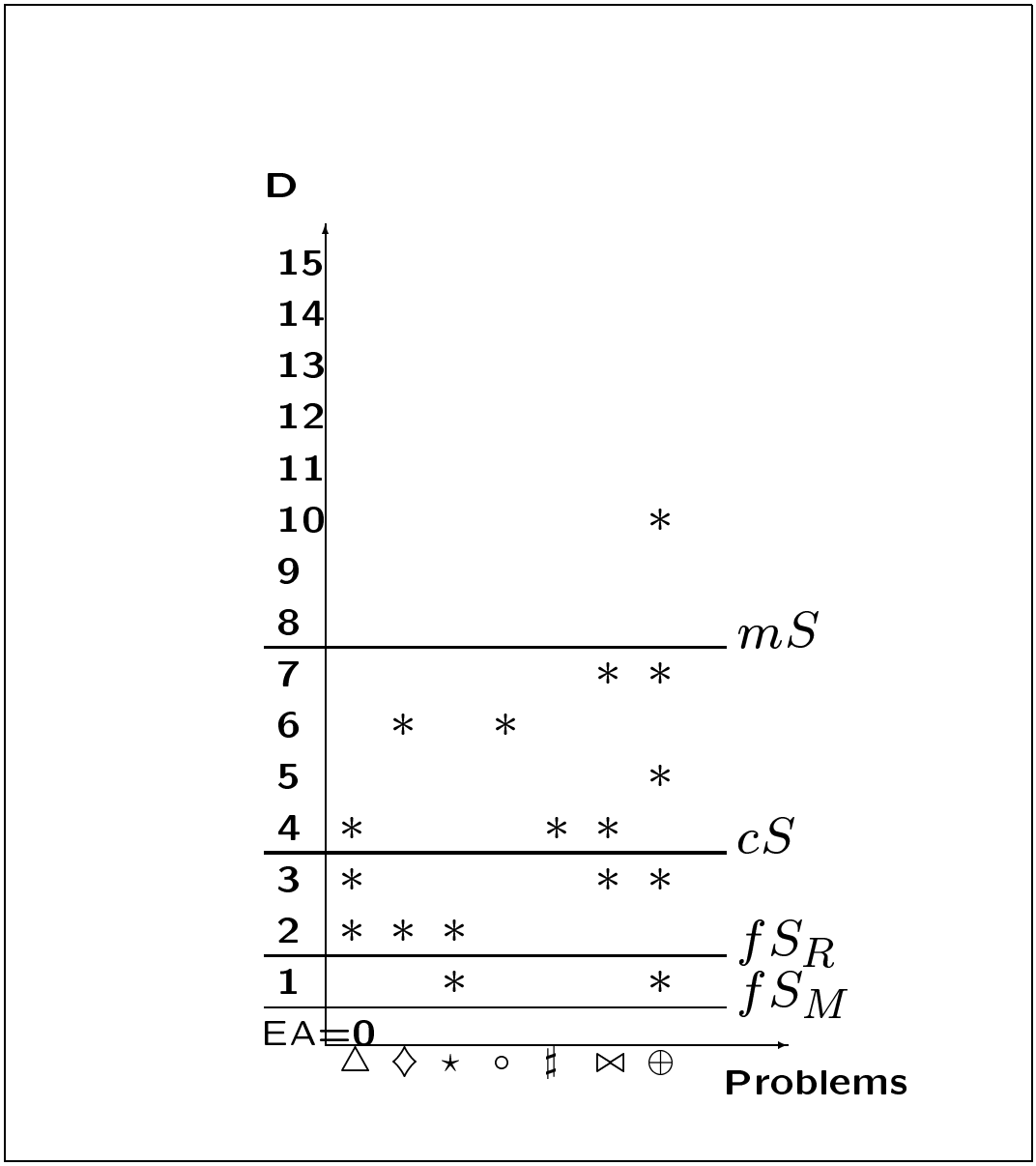
- **D=11: EA** $+ mS + fS_X + fS_M$
  none

- **D=12: EA** $+ mS + cS$
  none

- **D=13: EA** $+ mS + cS + fS_M$
  none

- **D=14: EA** $+ mS + cS + fS_X$
  none

- **D=15: EA** $+ mS + cS + fS_X + fS_M$
  none



Optimization problems versus the taxonomic index D of the Memetic Algorithm architectures applied to them.

Memetic Algorithms, Hyperheuristics and VNS

As shown in previous slides, the model presented captures a wide range of algorithms. In general we find that:

- if $\mu, \lambda > 1$ & $|LSS| \geq 1$ & dynamic/static schedulers $\Longrightarrow$ Memetic Algorithms
- if $\mu = \lambda = 1$ & $|LSS| > 1$ & dynamic schedulers $\Longrightarrow$ Hyperheuristics [CKS01, CKS02]
- if $\mu = \lambda = 1$ & $|LSS| > 1$ & static schedulers $\Longrightarrow$ VNS

As the free lunch theorems[WM95, Cul98] suggest no algorithm is best on all cases, which implies that more complex architectures (or for that case simpler ones) are not always the best choice.

Part 3

- Design issues for a robust engineering of competent MAs

We will try to address some of the following issues:

- Where/when do we apply local search?
- Which individuals in the population will be improved by local search?
- How long/wide should the local search be?
- What neighborhoods, acceptance criteria, etc the LS will use?
- How do we integrate the standard genetic operators with local search?
- Shall we use a Baldwinian or Lamarckian model for optimization?
- Which is the fitness landscape an MA is exploring?
- How do we engineer an MA that can jump out of deep local optima or traverse large neutral plateaus?
- What is needed for an MA to be able to cope with Multi-objective problems?
- How premature convergence can be averted?
- Help from theory?

In *Adaptive Global Optimization with Local Search* [Har94] and *Evolutionary Algorithms with Local Search for Combinatorial Optimization* [Lan98] the authors studied:

- Which individuals in the population will be improved by local search?
- How long/wide should the local search be?
- Shall we use a Baldwinian or Lamarckian model for optimization?

In *Memetic Algorithms for Combinatorial Optimization Problems:Fitness Landscapes and Effective Search Strategies* [Mer00] the author studied:

- The relationship between fitness landscapes and MA performance

In *Studies on the Theory and Design Space of Memetic Algorithms*[Kra02] the author studied:

- What neighborhoods, acceptance criteria, etc the LS will use?
- How do we integrate the standard genetic operators with local search?
- How do we engineer an MA that can jump out of deep local optima or traverse large neutral plateaus?
- How premature convergence can be averted?

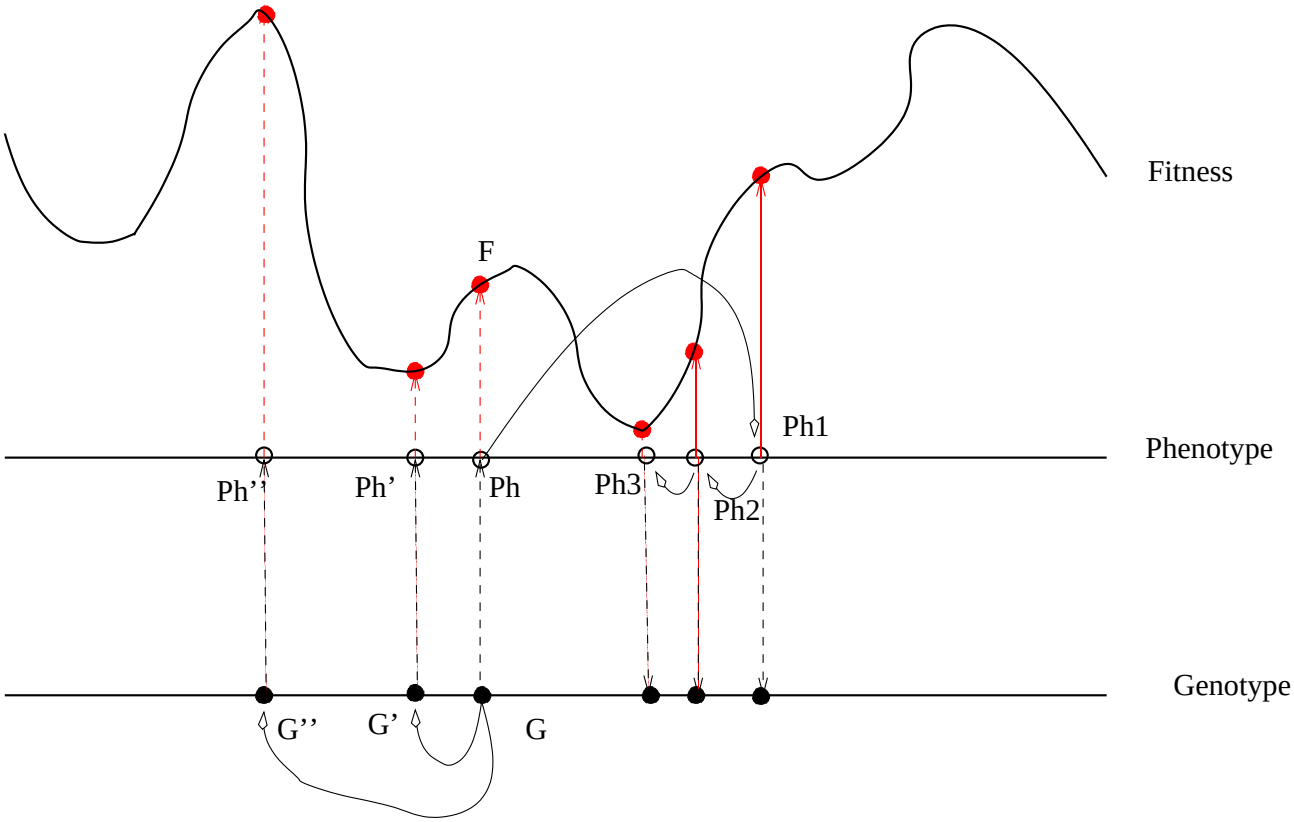
In *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization* [Kno02] the author studied:

- What is needed for an MA to be able to cope with Multi-objective problems?

In *Studies on the Theory and Design Space of Memetic Algorithms*[Kra02] and *Problemas de Otimizacão NP, Aproximabilidade e Computacão Evolutiva:Da Prática à Teoria* [Mos01] the authors studied:

- The impact of complexity theory and Kolmogorov theory for MAs

Baldwinian Vs. Lamarckian



Lamarckian learning:

- A genetic modification (by means of LS) that is beneficial for the individual is acquired in its life span.

- the modification is subject to propagation and selection into future generations.

- This is achieved by coding back into the genome any improvement found by the learning/optimization mechanism.

- This process is in direct opposition to the central dogma of biology as stated by August F.L. Weismann in the XIX century[Wei93] and is only possible when the inverse mapping, i.e., phenotype $\mapsto$ genotype, is computable **and** practical (i.e., not too expensive).

Baldwinian learning:

- Concurrently discovered by Baldwin[Bal96], Morgan[Mor96] and Osborn [Osb96] at the end of the XIX century.

- Baldwinian learning in MAs[Tur95],[Tur96a], [Tur96b], [TWA96] can be regarded as the application of LS with the sole purpose of fitness evaluation or development (see for example [HKB95],[KM98] and references therein).

- Baldwinian learning shows its effect through a process that Waddington [Wad57] summarizes as *Adaptation during development + canalization of development + genetic assimilation*. (referred to [Tur96b] for a position paper on Baldwinian Learning.)

If LS **directly modifies** the genotypes the EA is working with, then it is a Lamarckian MA regardless of which stage the LS was applied. If LS is **only used to bias the search** of the EA then it is a Baldwinian MA.

What is the impact of individual optimization on population evolution?

Assume[Mit97]

- Individual learning has no direct influence on individual DNA
- But ability to learn reduces need to "hard wire" traits in DNA

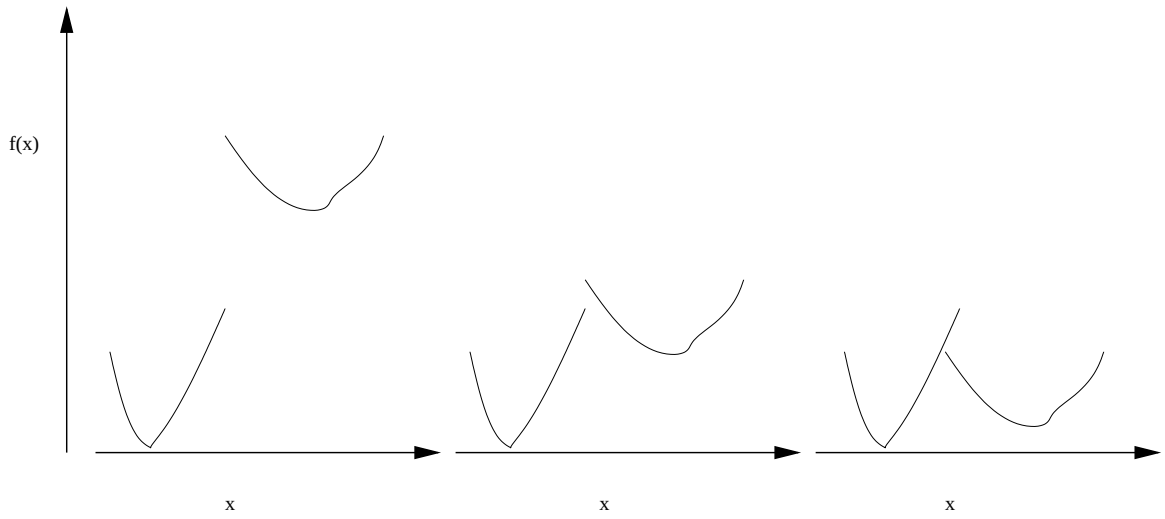
Then

- Ability of individuals to learn will support more diverse gene pool
- Because learning allows individuals with various "hard wired" traits to be successful
- More diverse gene pool will support faster evolution of gene pool

$\rightarrow$ individual learning (indirectly) increases rate of evolution

See classical experiment by Hinton and Nowlan[HN87]

Frequency of Local Search



Three hypothetical cases where local search can(?) help the evolutionary process:

- On the left, LS will probably **not** improve the EA's efficiency. Although the LS will move points toward local optima this can be achieved by the EA's selection itself as the range of values is wide.

- On the right, vigorous LS **will** probably enhance the EA as the points drawn from the two basins of attraction will not distinguish where the global optimum really lives.

- The curve in the middle is an intermediate case where an MA with **moderate** amount of LS can be most efficient.

Hart in [Har94] run a set of experiments on:

- Griewank
- Modified Griewank
- Rastrigin

He compared:

- Monte Carlo(MC)
- MultiStart(MS)
- Conjugate Gradient(CG)
- Genetic Algorithm(GA)
- MA= GA+CG(0.0625)
- MA= GA+CG(0.25)
- MA= GA+CG(1.0)
- Solis-West(SW)
- MA= GA+Sw(0.0625)
- MA= GA+CG(0.25)
- MA= GA+CG(1.0)

The number between parenthesis represents the frequency of local search for the MA. The MAs used were in class D=4 of the taxonomy described before.

Although the cS schedules local search for each individual at a time, it assigns to every individual the same probability of local search (Λ).

As it can be expected, no single algorithm was found to be best for all experiments, but it was possible to conclude that the MAs with low frequency of local search were more adequate for solving problems with large $\epsilon-$accuracy (previous figure to the left). On the other hand when the expected accuracy for solutions was high, then MAs with frequent local searches were better (previous figure to the right).

Hart concluded that:

> Refining individuals with local search can improve the efficiency of the GA-LS hybrids in two ways. First, the local searches may generate better solutions more efficiently than the GA's competitive selection. Second, the fitnesses of the refined solutions may reflect the domain-wide characteristics of the objective function more accurately, especially when complete local searches are performed.

On a different experiment he showed that if elitism is introduced then:

- the required frequency of local search must be reduced.

- there were cases were the elitist version of a GA was more efficient than the MA

and he concludes:

> There may be some functions for which local search does not improve the efficiency of the GA. In the absence of prior information about the function, these results recommend the use of elitism in the GA-LS hybrids.

Adaptive Local Search Frequency:

These methods aim to reduce the number of local searches used in each generation when there are redundant solutions in the population or when several solutions are clustered around different basins of attractions.

The cS is allowed to change the frequency of LS, Λ, accordingly to several schemes:

- **Complete method:** $\Lambda_i = \frac{\Lambda}{N_i}$ where N_i is the number of individuals in the population (known to the cS) which represent identical solutions to the problem. Expensive $O(N^2)$.
- **Local approximation:** $\Lambda_i = \frac{\Lambda_i}{1+\eta(N-1)\delta(p_1,p_2)}$ where $\eta \in [0,1]$ and $\delta(.,.)$ is the Kronecker function. Utilizes information from previous crossovers between p_1, p_2.
- **Global approximation:** $\Lambda_i = \frac{\Lambda_i}{1-\frac{N'}{N}}$ where N' is the number of duplicate pairs of parents.
- **Redundancy from distance matrix:** This is a sophisticate (and expensive) method of updating Λ. Based on F statistics and has resemblance to Goldberg and Richardson[GR87] *Fitness Sharing*.

Conclusions for the problems and algorithms mentioned before:

- Any of the methods in the previous slides produce better results than the traditionally fixed $\Lambda = 1$ local search.

- From Hart's experiments it is possible to conclude that large populations that use fixed frequency of local search benefit from low frequencies for LS.

- Traditionally MAs use small populations with very frequent local searches. Here we see a departure from conventional wisdom: use large populations and infrequent local searches.

- If elitism and fixed frequencies are used then reduce further Λ.

Fitness Distance Correlation

P.Merz in [Mer00] and other papers extensively studied the fitness landscapes MA explore.

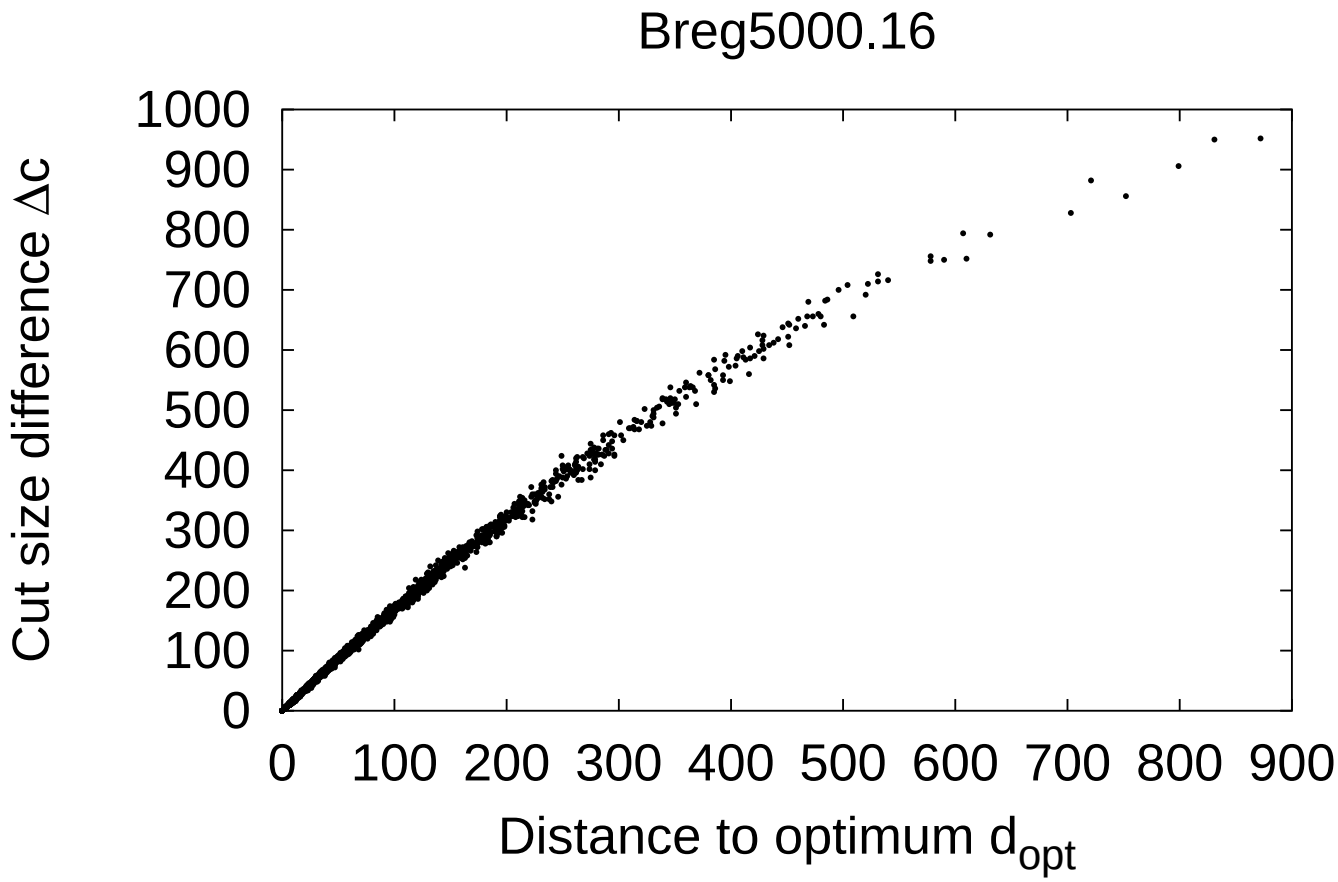
Fitness Distance Correlation:

$$\varrho(f, d_{opt}) = \frac{Cov(f,d_{opt})}{\sigma(f)\sigma(d_{opt})} \rightarrow$$

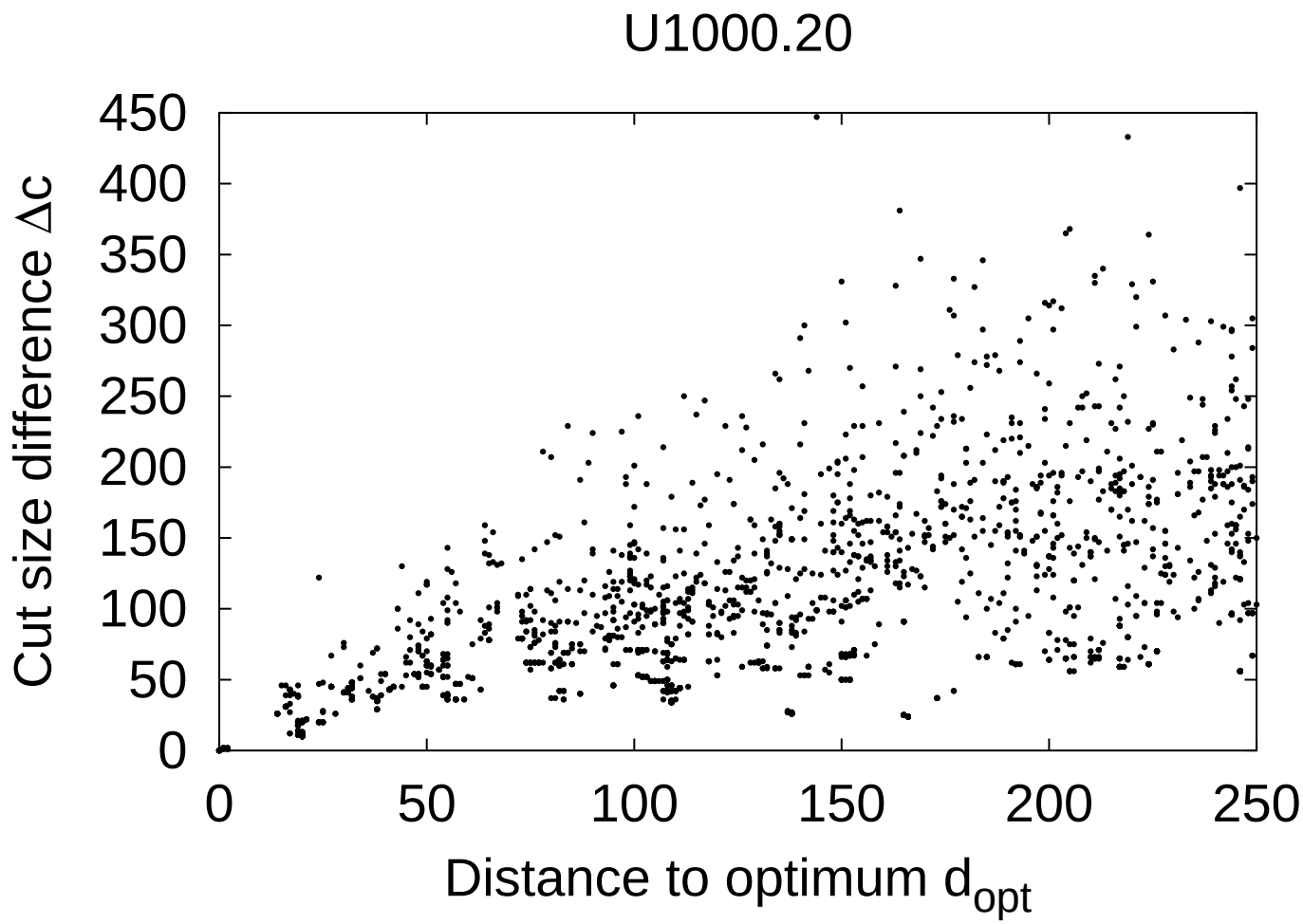
$$\varrho(f, d_{opt}) \approx \frac{1}{\sigma(f)\sigma(d)} \frac{1}{m} \sum_{i=m}^{i=1} (f_i - \overline{f})(d_i - \overline{d})$$

where we are given a set of points $x_1, x_2, \ldots, x_m$ that generate the time-series $f(x_1), f(x_2), \ldots, f(x_m)$, $d_i = d_{opt}(x_i)$ the shortest distance to a global optimum.

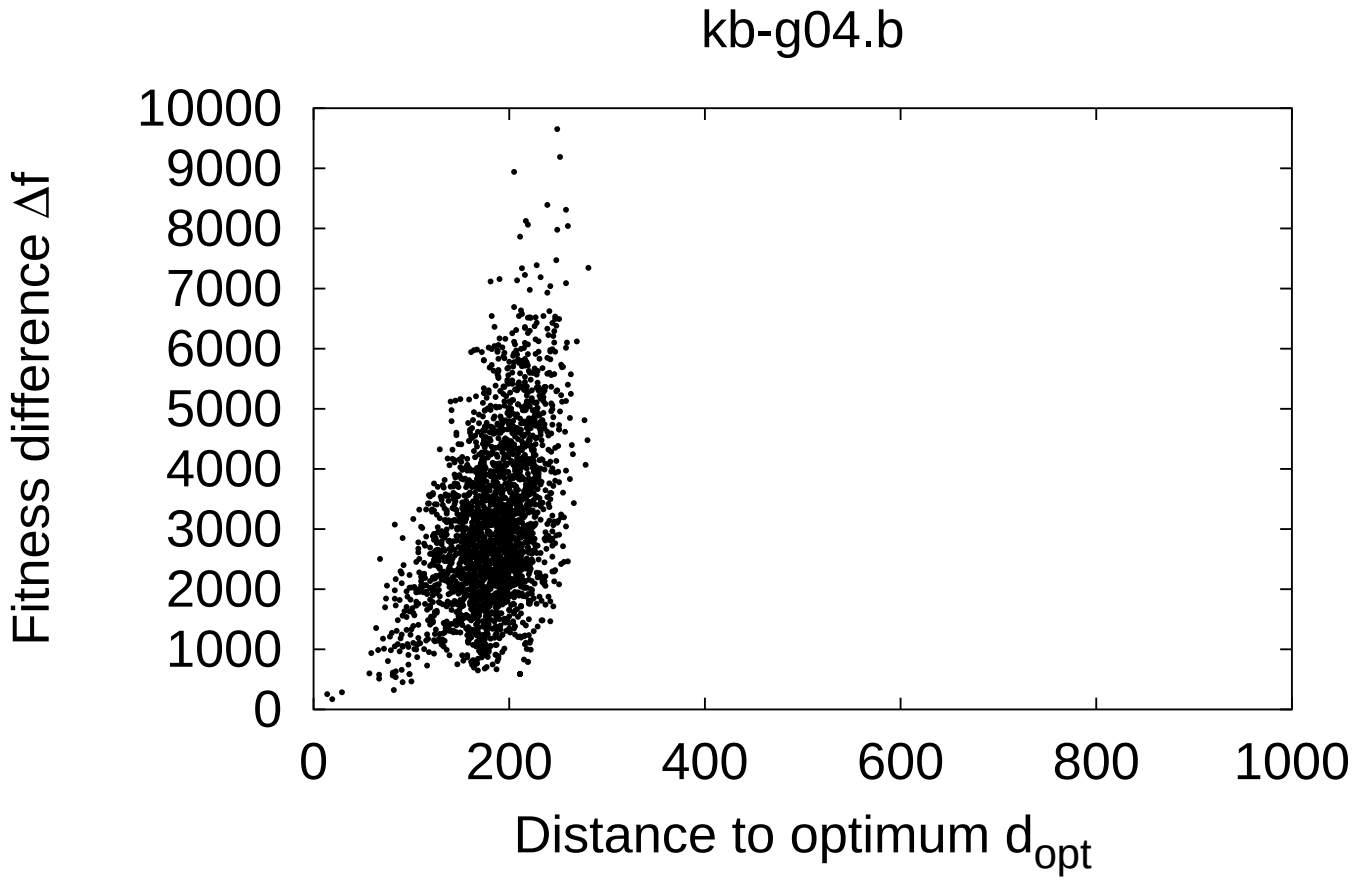
What can we learn from plotting $\varrho(.,.)$?



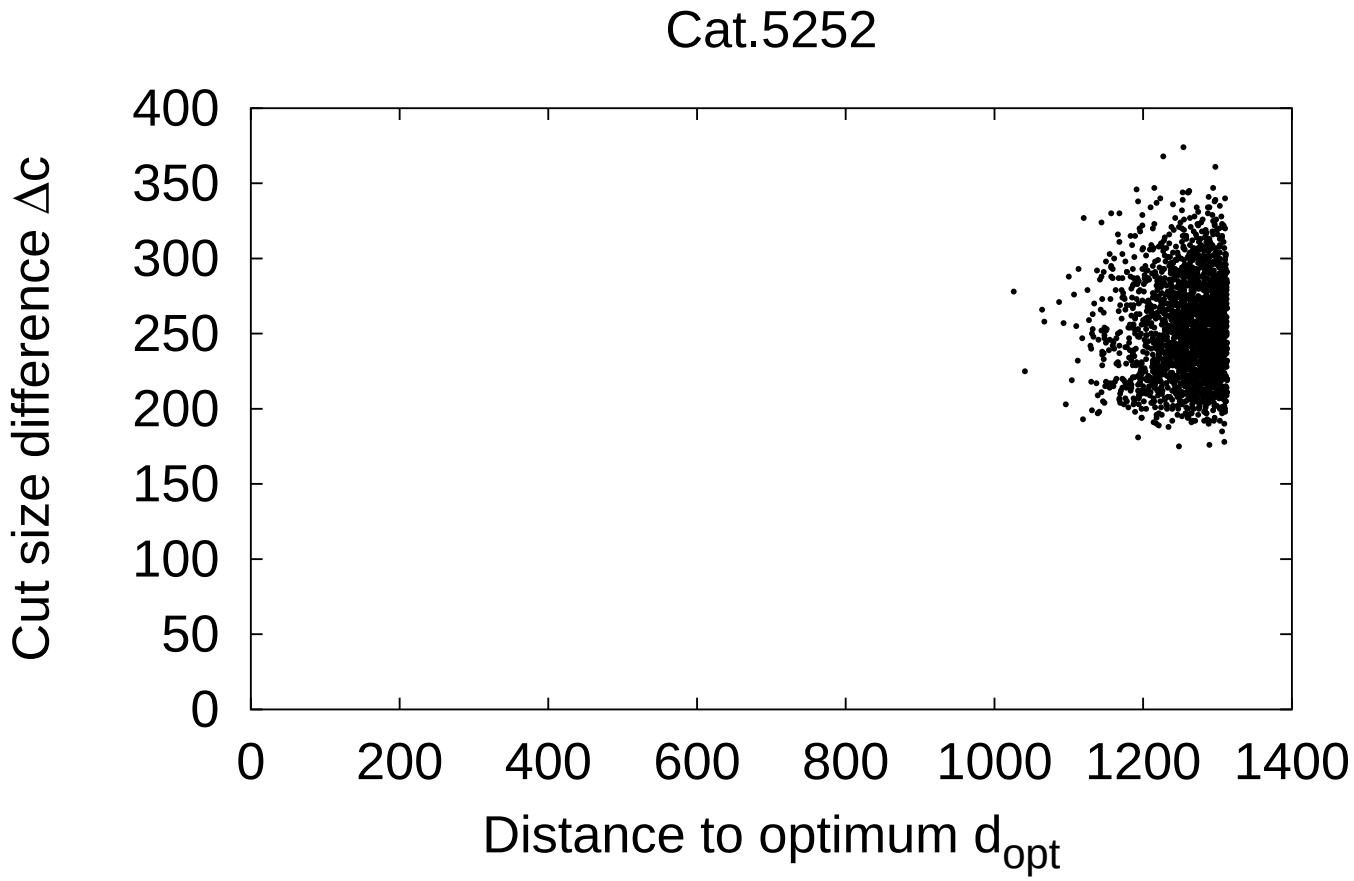
Very good information



Good information



Some information



No information

A word of caution:

- in general it is difficult (if not impossible) to adequately assess the distance to an optimum. Local optima are used and these need to be carefully chosen.

- in general people do not use the operators their algorithms employ to measure the distance to (sub)optima.

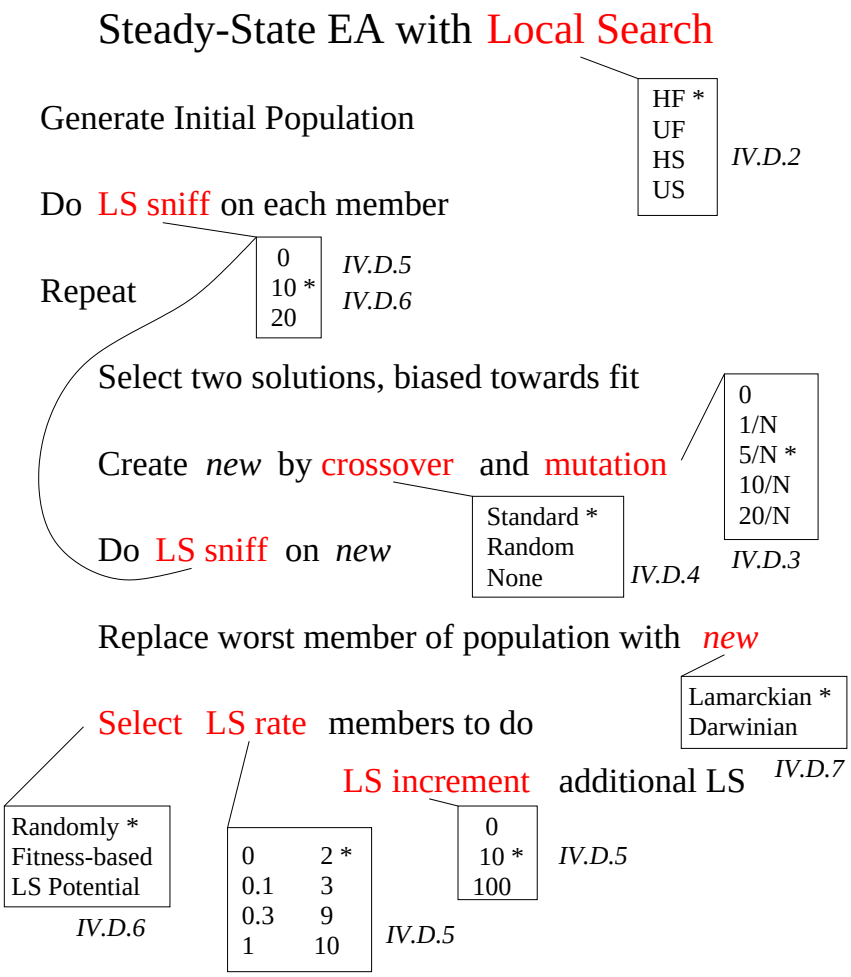
- the metric used to measure distance must be related to the operators the algorithm employ(!)

- watch out for *misinformation*, that is, anti-correlation(!).

Quality of Sampling

In the figure (from [Lan98]) the EA sample is misleading: the sampled points correlate negatively with the quality of local optima $\Longrightarrow$. We need to (better) integrate EA with LS.

A Tighter integration of Local and Global Search (from M.W.S. Land Ph.D thesis[Lan98]):



Some conclusions from Land's work:

- Land's MA proposal integrates more tightly Local and Global Search by using intelligently fS_X and cS. (for additional theoretical considerations on this issue see [GV99])

- Crossover is redundant for a MA working on Graph Bisection as LS does the same job! $\Longrightarrow$ **compositionality hypothesis**.

- Proposed rule of thumb: *When LS is used with Lamarckian evolution, the appropriate size of mutation will be at least as large as the typical basin size*. This implies a sometimes **massive** mutation (in effect macromutation).

- If complete LS takes place then there is evidence that Baldwinian evolution is competitive and robust $\Longrightarrow$ related to the compositionality hypothesis as LS will always find the building blocks if it is a complete search.

If we want to better integrate the genetic operators with the local searchers why not produce *crossover-aware-local-searchers* and *mutation-aware-local-searchers* by means of the fine-grain schedulers (fS_M, fS_X) mentioned in previous slides?:

- A local searcher that is used in an MA accepts a new point as its next search point only if it improves the objective function.

- Whether this disrupts the search done by crossover or mutation is seldom (never?) taken into consideration.

Krasnogor in [Kra02] proposes to make the local searcher(s) and the genetic operators to act in unison and synergetically by *construction*:

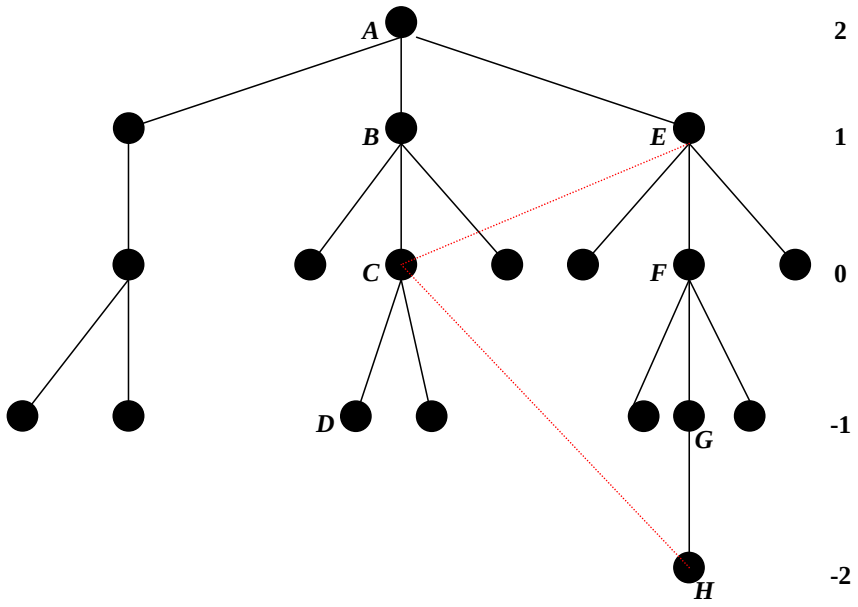
- crossover-aware LS: fS_X may enable the local searcher to crossover any *new point* that it is considering with a *representative sample of the population*. Then, the next point where to move to is the one that produces the better average fitness resulting from crossing over that candidate point with the sample of the population.

```
Crossover_Aware_Local_Search (x,S,n(),X()):
Begin
  /* x is the current solution */
  /* S is a small sample of the population */
  /* n() is a polynomial-time computable neighborhood */
  /* X() is the crossover operator in use */
  Repeat Until ( finalization criteria is met ) Do
    For n := a neighbor ∈ n(x) To last neighbor Do
      Produce P' by crossing-over n with ;
      every member of S using X();
      Compute the average fitness of P', store in F;
      If (F > best_average_fitness) Then
        next_solution = n;
        best_average_fitness = F;
      Fi
    Od
    x = next_solution;
    /* The returned solution is the one for which */
    /* its crossover with a reduced sample of the */
    /* population produces the best average fitness */
  Od
End.
```

- mutation-aware LS: fS_M may enable the local searcher to maximize the variance of the fitness obtained when the selected point is the object of a reduced number of mutations. A higher variance helps to keep diversity.

```
Mutation_Aware_Local_Search (x,n(),M()):
Begin
  /* x is the current solution */
  /* n() is a polynomial-time computable neighborhood */
  /* M() is the mutation operator in use */
  Repeat Until ( finalization criteria is met ) Do
    For n := a neighbor ∈ n(x) To last neighbor Do
      Produce P' by generating a number of mutants;
      of n by means of M();
      Compute the fitness variance of P', store in V;
      If (V > best_fitness_variance) Then
        next_solution = n;
        best_fitness_variance = V;
      Fi
    Od
    x = next_solution;
    /* The returned solution is the one for which */
    /* its reduced sample of mutants produces */
    /* the largest variance in fitness */
  Od
End.
```

Revisiting Local Optima, Basins of Attraction and the Dynamics of Search



Schematic representation of a multi-dimensional fitness landscape. This graph was called *disconnectivity graph* in [BK97]. Each level of the graph (e.g., 2,1,0,-1,-2) corresponds to a fitness barrier ϵ. Each fat dot represents the basin of attraction of a local optimum. Every walk *within* a given basin does not cross fitness barriers higher than the corresponding ϵ.

Dynamics of search:

1. The MA starts the search with samples from different vertices of the DG.

2. Then it will collapse the population to a particular basin (when it is converged)

An hypothetical trajectory is: $A \to B \to C \to D$. If we want the MA to explored the unnamed basin connected to D by C then it will need to **bypass** a fitness barrier given by ϵ_0.

Another case: if the MA is trapped in B (and the global optimum is in H) then we need to provide the MA with a mechanism to jump across the fitness barrier with an ϵ_2 and then $A \to E \to F \to G \to H$.

Two mechanisms that allow the MA to do just that:

- adaptive helper based on a Monte-Carlo local searcher[Kra02]

- adaptive helper based on fuzzy-logic local searcher[KP02]

The adaptive helper based on a Monte-Carlo local searcher motivated by:

- In the memetic algorithm literature, keeping population diversity while using local search together with a GA is always an issue to be addressed, either implicitly or explicitly. Usually this takes the form of complex operators or sophisticated book-keeping and/or guiding strategies + sometimes population structures.

- In the multi-agent literature, different annealing schemes were proposed together with different ways of sharing either solutions, annealing schedules or temperatures. Several inter individual coupling mechanisms were investigated.

- Boese [Boe96] showed that for **optimization** the optimal annealing schedule does not cool-down monotonically but rather oscillates!.

The temperature in the pseudocode below is $temperature = \frac{1}{|P_{MaxFitness} - P_{AvgFitness}|}$:

ApplyMove(indip):
Begin
  /* This is a Maximizing process */
  prevFitness = fitness(indip);
  Modify(indip);
  nFitness = fitness(indip);
  If (prevFitness $<$ nFitness) **Then**
    Accept configuration;
  Fi
  Else
    $deltaE = prevFitness - nFitness$;
    $threshold = e^{-k * \frac{deltaE}{temperature}}$;
    If (random(0,1) $<$ threshold) **Then**
      Accept configuration;
      /* even if worse than the previous one */
    Fi
    Else
      Reject changes;
    Esle
  Esle
End.

The induced dynamic is such that:

1. When the population is diverse $\Longrightarrow$ the fitnesses in the population are spread $\Longrightarrow$ temperature is low $\Longrightarrow$ only accept improving moves (EXPLOITATION).

2. When the population is converged on a local optimum the fitnesses are similar $\Longrightarrow$ temperature rises $\Longrightarrow$ accepts moves that deteriorate the objective value (EXPLORATION)

Point 1 leads to 2 leads to 1 leads to 2 ...

In [Kra02] Krasnogor shows that this **simple** and **cheap** mechanism improved:

- final fitness

- and was able to keep runs' diversity longer.

As the temperature is a function of first order statistics (e.g. average fitness) diversity was measured as the number of different fitness values cases.

(Caution show be used with the definition of diversity and its relation to evolutionary success [BGKK02])

Results were reported for a variety of Protein Structure Prediction Models and TSP.

The adaptive helper based on a fuzzy-logic local searcher is motivated by:

- Crisp criteria for the acceptance of new solutions are inadequate and/or myopic.

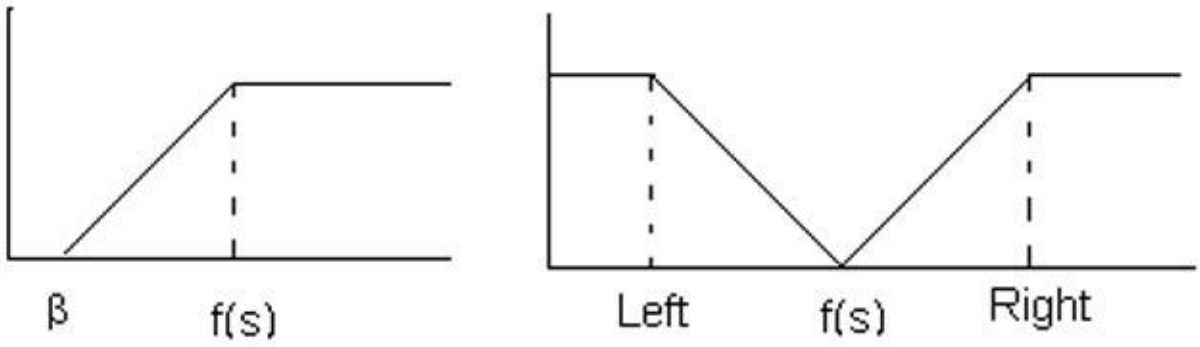
- fuzzy-logic provides easy to understand, hence to implement, robust intuitive criteria for exploring local neighborhoods.

- we can adapt the neighborhood explored via feedback of the state of the search.

In [KP02] we integrate Multimeme algorithms with Fuzzy-logic based helpers of the form:

Procedure FANS:
Begin
k:=maxK;
While (not-end) **Do**
/* The neighborhood scheduler, NS, searches */
/* the locality of the current solution */
/* via operator $\mathcal{O}^k$ */
S_{new} = NS($\mathcal{O}^k, \mu, S_{cur}$);
If (S_{new} is good enough in terms of $\mu()$) **Then**
$S_{cur} := S_{new}$;
adaptFuzzyValuation($\mu(), S_{cur}$);
Fi
Else
/* NS is not able to obtain a good */
/* enough solution. The operator will be */
/* changed by way of modifications to */
/* the parameter k */
If ((k=1)) **Then**
k:= maxK;;
Fi
Else
k := k-1;;
Esle
Esle
Od
End.

The *good enough* is defined in terms of the fuzzy-valuation function $\mu()$. Several reasonable $\mu()$ can be defined:



FANS based memes have:

- expression power of fuzzy-logic
- variable neighborhood search elements

With these helpers we discovered new global optima for lattice based instances of the Protein Structure Prediction.

Multimeme Algorithms

The majority of MA use a unique, complex and powerful local searcher, e.g., Lin-Kernigham or K-opt for TSP or Conjugate gradient for continuous Problems.

This is feasible *iif* a powerful LS is known, and even if it is known, is it reasonable?

The exploitation of several search neighborhoods (memes in this context) is the basic foundation of Variable Neighborhood Search [HM01]:

- several LS are *scheduled* to produce a more reliable search by jumping from one induced landscape to another one if/when the search is stagnated. See the DG graph(red lines).

Several other metaheuristics employ the same trick, among them, FANS, HyperHeuristics,VNS, etc.

In "Studies on the Theory and Design Space of Memetic Algorithms" Krasnogor investigates the *adaptive* use of *several* LS in MAs.

- In contrast with VNS,FANS or HH where just one solution is kept at all times, by the evolutionary nature of the MA, several solutions are used to search for global optima.

- Not all individuals (solutions) search with the same neighborhood (unless one neighborhood is clearly superior to the rest), so a parallel search is done on all the landscapes induced by the different memes. This again differs from VNS where just one neighborhood at a time is used.

- The approach automatically recognizes which is/are the most promising neighborhood(s) to explore at any given point in time and potentially includes both the fixed permutation of neighborhoods as VNS does and random permutations

- Memes that are considered harmful can be disable and re-introduced later for search

Protein Structure Prediction Example:

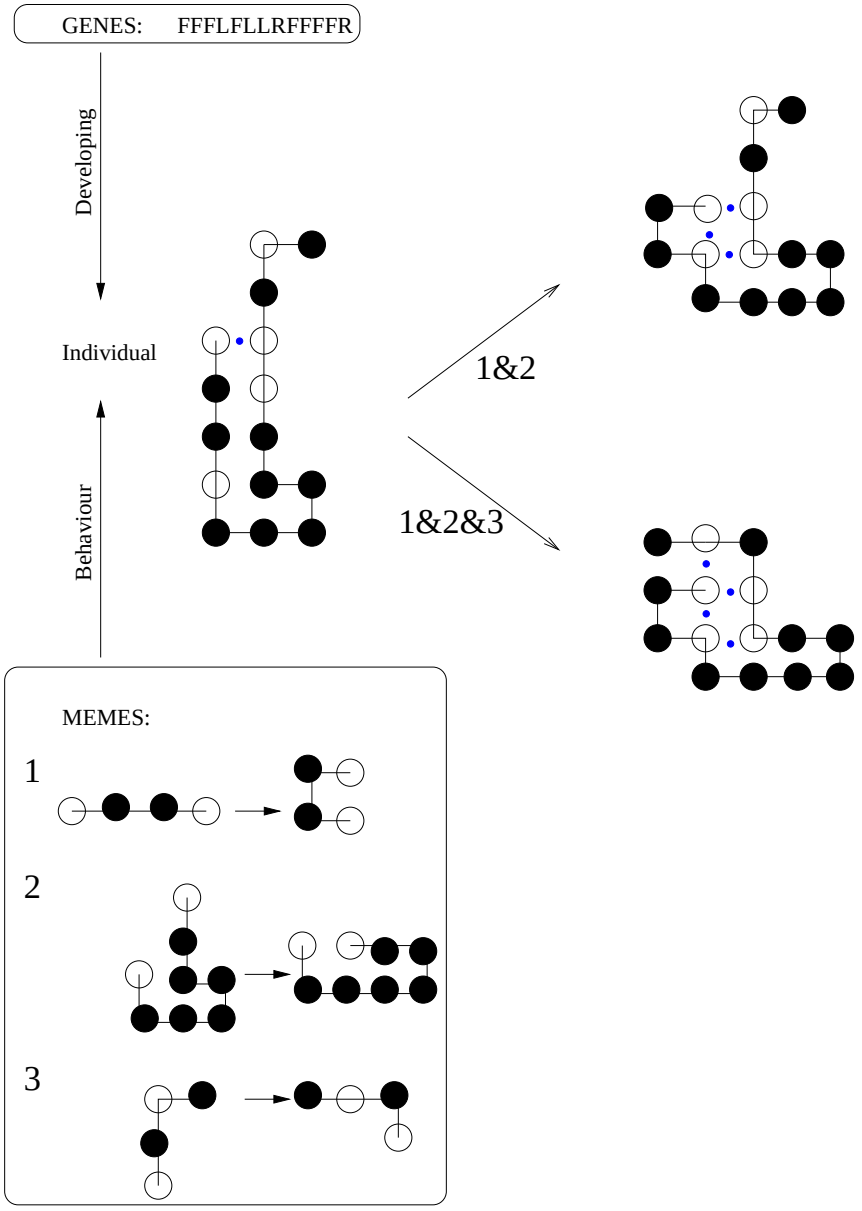
- an individual is $(genes, meme(plex))$ pair.
- the genes encode for solutions to the problem
- the meme(plex) represents one or more LS behavior.

The memepool contains:

- four types of LS: MM,R,S,P.
- MM,R,S come in 3 flavors.
- total number of LS = 9

From Krasnogor[KHSP99]

In gene space: Crossover,Mutation,Selection



In meme space: Replication,Mutation,Selection

Static Memes	O/R	MFHT
GA (no memes)	0/10	-
MA with Macro Mutation (r=4)	4/10	73.25
MA with Macro Mutation (r=8)	2/10	25.5
MA with Macro Mutation (r=16)	0/10	—-
MA with Reflect (r=4)	2/10	19
MA with Reflect (r=8)	0/10	—-
MA with Reflect (r=16)	2/10	31.5
MA with Stretch (r=4)	0/10	—-
MA with Stretch (r=8)	0/10	—-
MA with Stretch (r=16)	0/10	—-
MA with Pivot	6/10	20.33
MultiMeme (all local searchers)	7/10	23.71

Number of times and mean first hitting time (in generations) to achieve an optimal solution to instance 14. Different algorithms are compared based on 10 independent runs. These are Static Memes.

Static Memes	SO/R	MFHT
MA with Macro Mutation (r=4)	1/10	150
MA with Macro Mutation (r=8)	0/10	—-
MA with Macro Mutation (r=16)	0/10	—-
MA with Reflect (r=4)	0/10	—-
MA with Reflect (r=8)	0/10	—-
MA with Reflect (r=16)	0/10	—-
MA with Stretch (r=4)	0/10	—-
MA with Stretch (r=8)	0/10	—-
MA with Stretch (r=16)	0/10	—-
MA with Pivot	2/10	82
MultiMeme (all local searchers)	5/10	64.8

Number of times and mean first hitting time (in generations) to achieve an optimal solution to instance 19. Different algorithms are compared based on 10 independent runs. These are Adaptive Memes.

Static Memes	O/R	MFHT
GA (no memes)	0/10	-
Macro Mutation (r=4)	5/10	85.0
Macro Mutation (r=8)	2/10	100.0
Macro Mutation (r=16)	1/10	100.0
Reflect (r=4)	3/10	27.3
Reflect (r=8)	2/10	63.5
Reflect (r=16)	1/10	100.0
Stretch (r=4)	0/10	-
Stretch (r=8)	0/10	-
Stretch (r=16)	0/10	-
Pivot	5/10	67.4
MultiMeme(IR=0.2)	7/10	23.71

Relation between the number of times the single meme algorithm reached optima relative to the number of runs executed for different memes. Also, in the third column, the mean first hitting time is computed. The last row presents the associated values for the multimeme algorithm. Memes are static helpers. Instance 15

Many more examples on different models of Protein Structure Prediction, Protein Structure Comparisons, TSP, NK-Landscapes and dynamic OneMax showed similar trends: *The use of multiple Local Searchers produces more robust (and efficient) memetic algorithms*.

How does the Multimeme Algorithm decides which meme to use?

with a simple inheritance mechanism called SIM:

Formally a (potentially) poly-parental SI mechanism is

$$L^{t,i} = \begin{cases} L^{t-1,j} & \text{if } \forall k,j \in P(i), k \neq j, L^{t-1,j} == L^{t-1,k} \\ L^{t-1,j} & \text{if } F(I_j^{t-1}) > F(I_k^{t-1}) \forall k,j \in P(i), k \neq j \\ L^{t-1,k} \text{for } k \in |P(i)| & \text{otherwise} \end{cases} \tag{4}$$

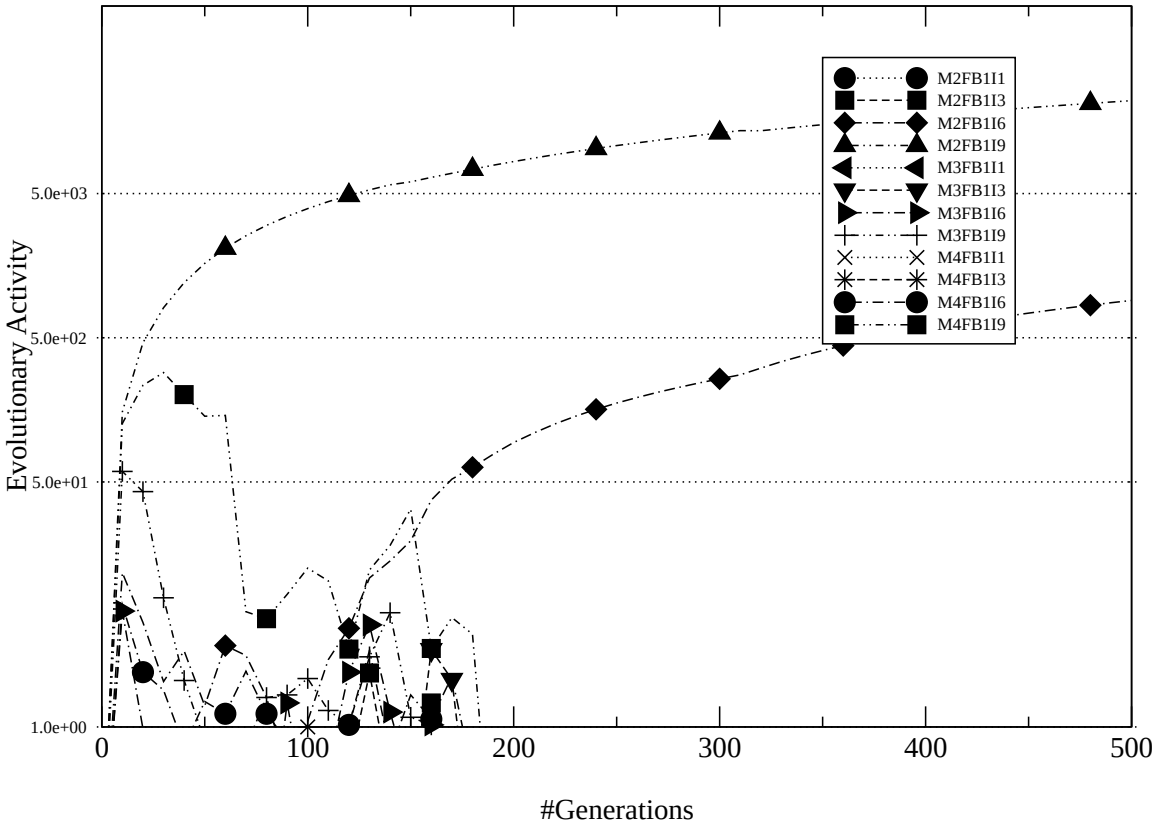
There is an *innovation rate* parameters that overrides the previous mechanism[Kra02] and ensures that memes are re-introduced in the population.

We can visualize how the Multimeme algorithm employs its memes by plotting[BN92]:

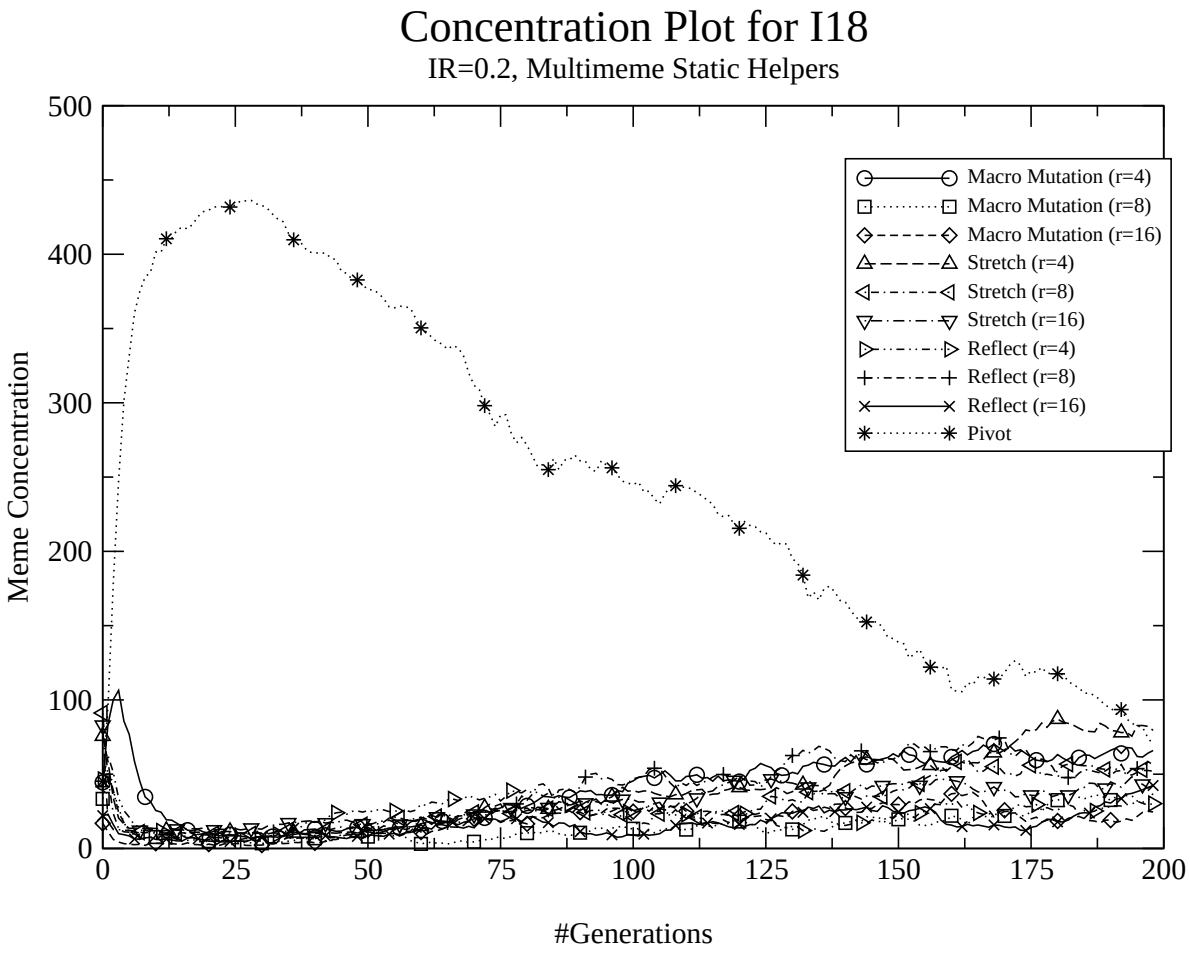
$$a_i(t) = \begin{cases} \int_0^t c_i(t)dt & \text{if } c_i(t) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

and by using this analyse/suggest ways to improve search.

Evolutionary Activity Vs #Generations(30 runs averaged)



(lin318.tsp):The evolutionary activity of a MultiMeme algorithm is shown. Several K-exchange LS are used as memes



(Instance 18): Meme concentration as a function of generation number for instance I18.

Toward Truly "memetic" Memetic Algorithms: discussion and proofs of concept

- The name "Memetic algorithm" is a very contested label that stirs critics and controversies among researchers and practitioners.

- It is important *not* to call these algorithms "hybrids" as we do not know hybrids of *what* we are talking about, e.g. a hybrid between a GA and Tony Blair? (a "third way" GA?)

- If we stick to the name Memetic Algorithm there is much to be learn

BUT we need to *put back* the memetics into memetic algorithms!

Memetic theory started as such with the definition given by R. Dawkins of a meme as a unit of cultural inheritance[Daw76][Daw82]:

> I think that a new kind of replicator has recently emerged on this very planet. It is staring us in the face. It is still in its infancy, still drifting clumsily about in its primeval soup, but already it is achieving evolutionary change at a rate that leaves the old gene panting far behind. The new soup is the soup of human culture. We need a name for the new replicator, a noun that conveys the idea of a unit of cultural transmission, or a unit of imitation. "Mimeme" comes from a suitable Greek root, but I want a monosyllable that sounds a bit like "gene". I hope my classicist friends will forgive me if I abbreviate mimeme to meme.(2) If it is any consolation, it could alternatively be thought of as being related to "memory", or to the French word "même". It should be pronounced to rhyme with "cream". Examples of memes are tunes, ideas, catch-phrases, clothes fashions, ways of making pots or of building arches. Just as genes propagate themselves in the gene pool by leaping from body to body via sperms or eggs, so memes propagate themselves in the meme pool by leaping from brain to brain via a process which, in the broad sense, can be called imitation.

Many other researchers and philosophers flirted with the idea that cultural phenomena can somehow be explained in evolutionary terms even before Dawkins introduction of a meme:

m-culture and i-culture, culture-types, etc.

The merit of Dawkins is that he defined a new signifier (meme) to the thing being signified (the unit of cultural transmission).

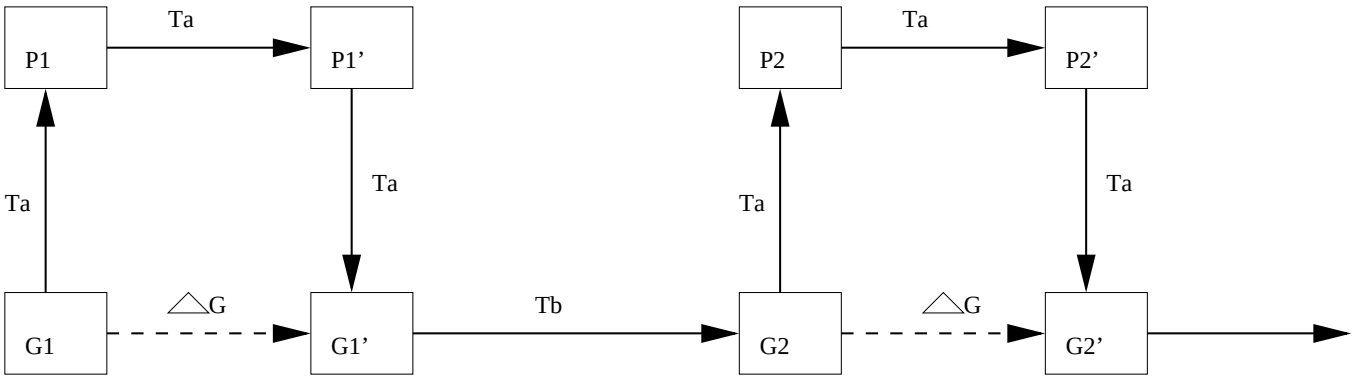
As a new word, it was not loaded with preconceptions and from a CS point of view it was appealing as it implies a discrete (albeit unknown) structure.

The fundamental innovation of memetic theory is the recognition that a *dual* system of inheritance, by the existence of two distinct replicators, mould human culture.

How this is related to optimization?

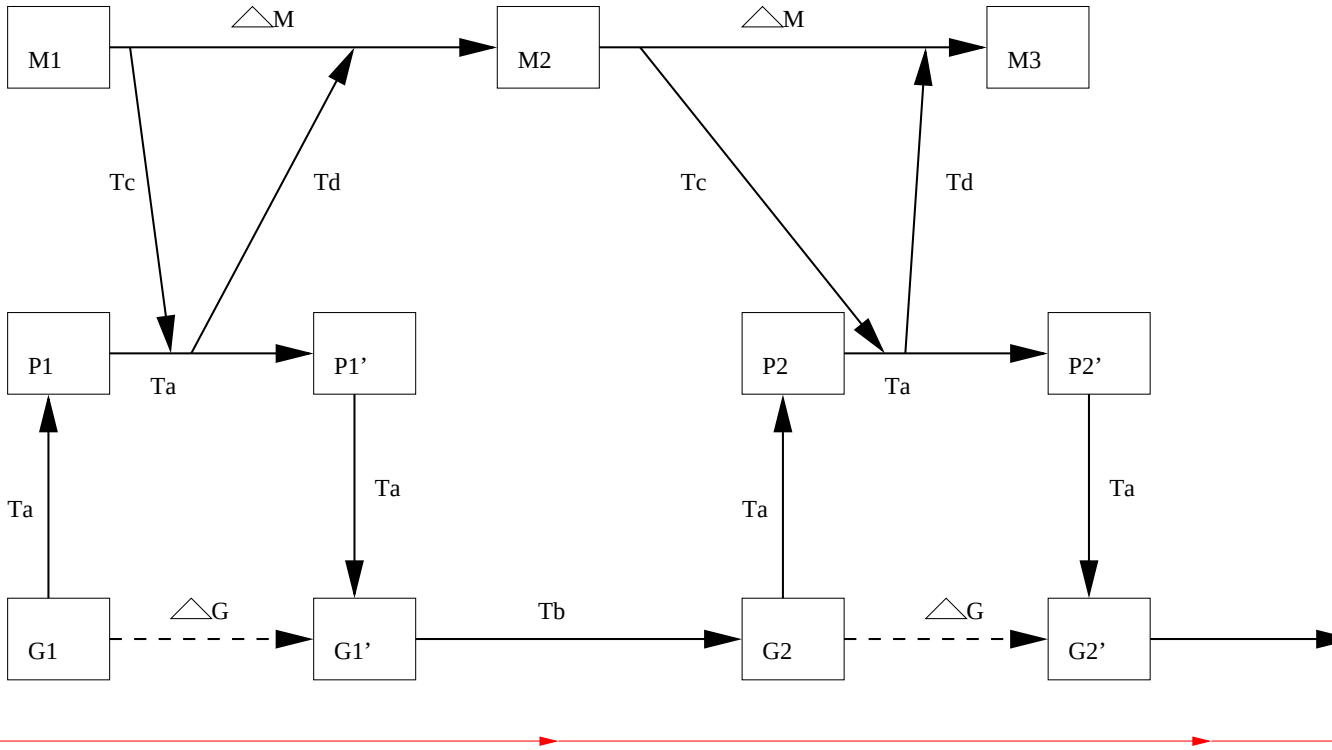
- different optimization methods are thought as different behaviors.
- behaviors are encoded as memes.
- memes evolve.
- hence optimization methods can be evolved (not *only* solutions but also the *way* to obtain solutions are evolved).

Memetic Theory in Evolutionary Computation



Evolutionary genetic cycle from [Dur91].

The "standard" evolutionary algorithm paradigm.



Coevolutionary memetic-genetic cycle from [Dur91].

What the "standard" memetic algorithm paradigm should look like if memetic was taken seriously.

In the first graph there are *epigenetic* phenomena, e.g., interactions with the environment, in-migrations, out-migrations, individual development, etc and also Mendelian principles which govern genetic inheritance that transform the distribution of genotypes. Only genes are taken into consideration $\Longrightarrow$ GEC has so far concentrated in implemented many facets of this graph.

In the second graph genes (solutions) and memes (improvement methods) co-exist. In a memetic system memes can potentially change and evolve using rules and time-scales other than the traditional genetic ones. Cavalli-Sforza[CSF81] argues that memetic evolution is driven by:

> ...the balance of several evolutionary forces: (1) *mutation*, which is both purposive (innovation) and random (copy error); (2) *transmision*, which is not as inert as in biology [i.e., conveyance may also be horizontal and oblique]; (3) *cultural drift* (sampling fluctuations); (4) *cultural selection)* (decisions by individuals); and (5) *natural selection* (the consequences at the level of Darwinian fitness) ...

Besides the traditional genetic transformations as those of the first graph we also have for memetics:

- learn memes
- adopt memes
- imitate memes
- modify memes

Also

- memes can be taught
- memes can be preached
- memes can be advertised

Memetic algorithms, as they were used so far, failed completely (or almost completely) to implement this dual inheritance system to any degree.

but why should we (optimizers) care?

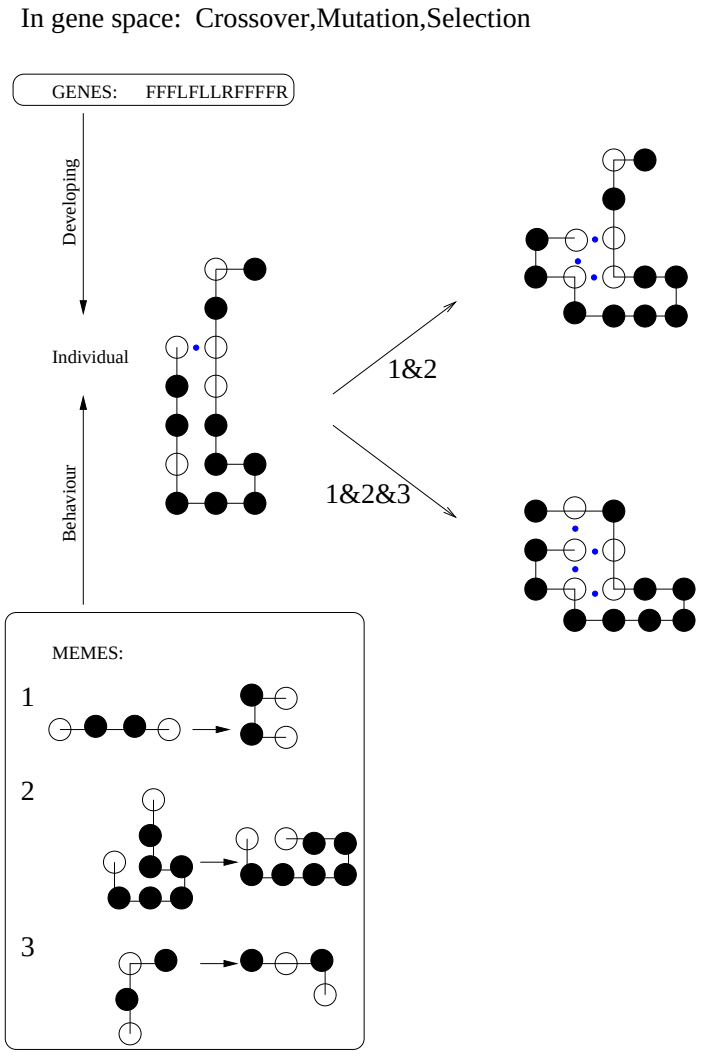
Because we know that different problems, different instances and different search states require different actions. Because not for every problem we have well-tested "winner" heuristics like a Lin-kernighan or K_opt.

Hence:

- we could **evolve those actions** on-the-fly $\Longrightarrow$
- Self-Generating Metaheuristics $\Longrightarrow$
- Self-Generating Memetic Algorithms [Kra02]

(a big opportunity (and challenge) for GPers)

In [KPMR98], [KdlCP+98] and [KHSP99] we initiated research on Self-Generating MAs of the form:



for models of Protein Structure Prediction.

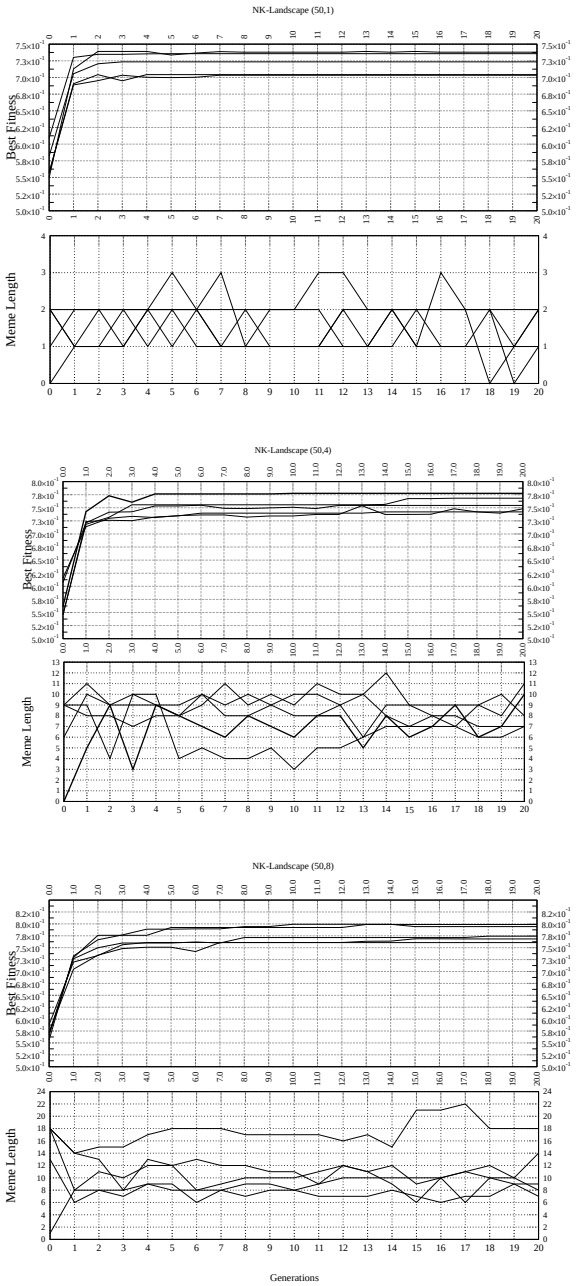
We also developed Self-Generating MAs for NK-Landscapes. We self-generated LS in the form of rules $initialPattern \rightarrow endPatter$.

The rules specify the radius of action of the local search, that is, the number of bits that should be condidered for improvement (similar to a variable K-Opt LS).

We tried the SGMA in 4 different regimes of NK-Landscapes:

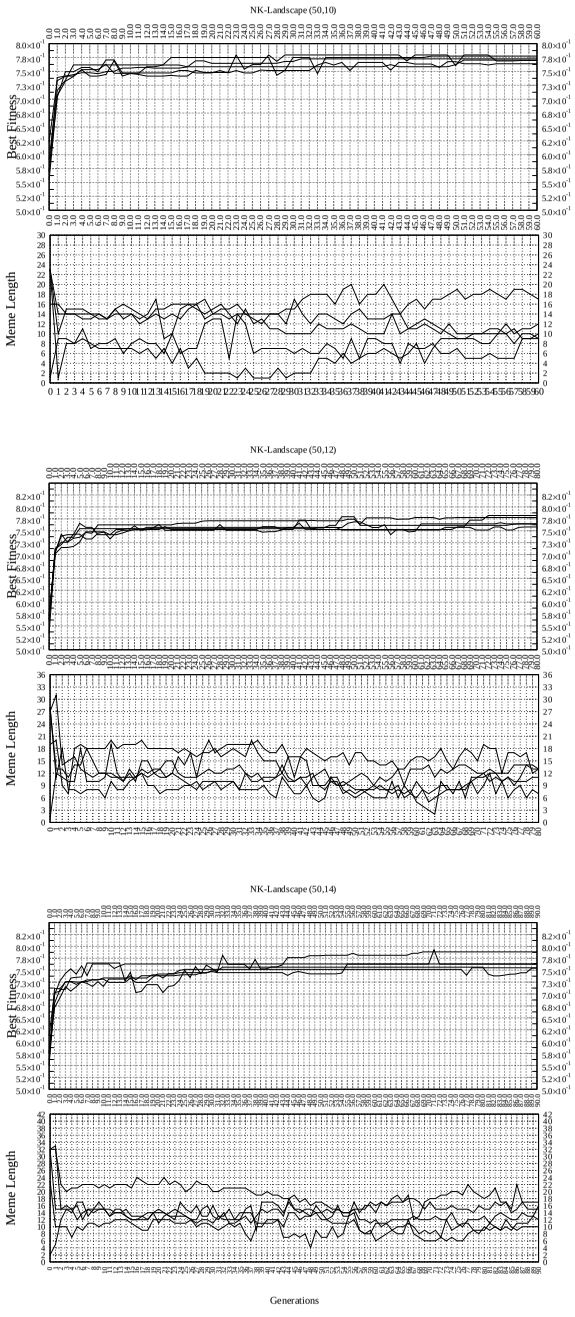
- low epistasis, poly-time solvable: $(50, 1), (50, 4)$ with adjacent neighbors.
- high epistasis, poly-time solvable: $(50, 8), (50, 10), (50, 12), (50, 14)$ with adjacent neighbors
- low epistasis, NP-hard: $(50, 1), (50, 4)$ with random neighbors.
- high epistasis, NP-hard: $(50, 8), (50, 10), (50, 12), (50, 14)$ with random neighbors.

Low epistasis, poly-time solvable:



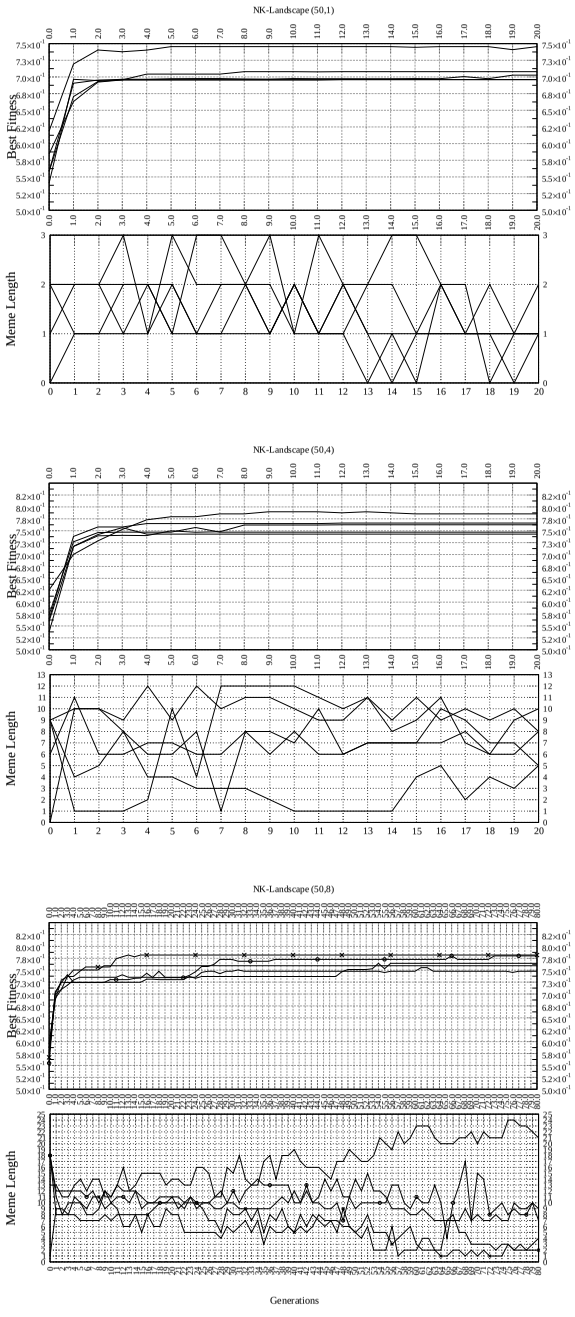
NK(50,1), NK(50,4) and NK(50,8). Adyacent neighbours.

High epistasis, poly-time solvable:



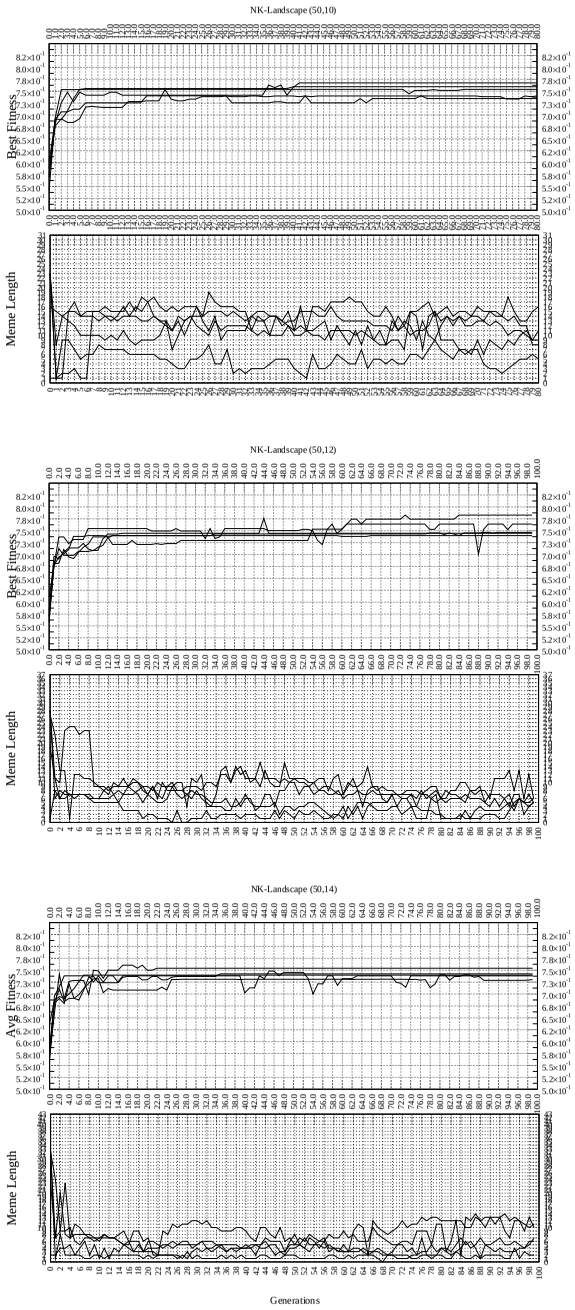
NK(50,10), NK(50,12) and NK(50,14). Adyacent neighbours.

Low epistasis, NP-hard:



NK(50,1), NK(50,4) and NK(50,8). Random neighbours.

High epistasis, NP-hard:



NK(50,10), NK(50,12) and NK(50,14). Random neighbours.

- The evolved rules *fit* different problem scenarios.

- For each problem regime, different types of rules are evolved.

- The range of regimes studied is significant as it spans from low epistasis problems to highly epistatic instances, and in an orthogonal dimension, from polynomial time solvable to NP-hard instances.

- The Self-Generating MA is clearly behaving differently for each regime *without* human intervention.

Invitation to Men of Theory

Memetic Algorithms are unique in the sense that:

- they exploit evolutionary metaphors and
- traditional local search insights.

Evolutionary Computation theory is obviously relevant but so it is local search theory!

Local search has been rigorously analyzed with the so called *Polynomial Local Search Complexity Theory*[Yan97][JPY88].

This has recently been investigated by Land[Lan98], Moscato[Mos01] and Krasnogor[Kra02] for MAs.

$\Longrightarrow$ much more must be done!

- Moscato[Mos01] presents a nice compendium of complexity results with a view to Memetic Algorithms (in Portuguese). He argues, following the paper *Parameterized Complexity: A Framework for Systematically Confronting Computational Intractability* by Downey,Fellows and Stege (in English), that parameterized complexity can be used to guide the design of genetic operators.

- Krasnogor[Kra02] shows the PLS-completeness of a family of MAs and argues that Kolmogorov Complexity Theory can be used to strengthen the bounds traditionally obtained by PLS and worst case complexity theory.

- Once Self-Generating Memetic Algorithms become more mature Evolutionary Game Theory will come into play.

Conclusions

- There is much more to MAs than simply putting a local search stage somewhere in the evolutionary cycle.

- Just a small fraction of the architectural design space has been explored and we do not know why a given architecture performs well or bad for a given problem yet.

- People usually use a unique LS. This is fine if that "silver bullet" is known. Otherwise use Multimeme algorithms: lots of simple local searchers can do the trick if your MMA is allowed to choose them on-the-fly.

- ADAPT: the search process is a dynamic process, your LS should detect and react to the state of the search (global/local).

- Not always small populations are better. Sometimes larger populations with infrequent local searchers are best.

- Be careful about the assumptions you make on how the genetic operators interact with LS(s).

- Investigate the usefulness of long vs short local searches.
- Investigate how you allocate LS trials in your problem.
- Understand that the fitness landscape explored by your MA is not a one-operator landscape but the super-position of several operators-induced landscape.
- Be cautious when you decide for a Lamarckian or Baldwinian MAs.
- Use more expressive acceptance criteria in your LS.
- Understand PLS, Kolmogorov and Complexity theories.
- If you have a problem for which you have no idea which memes to give to your multi operator MA then let it find them by its own! use Self-Generating MAs.
- Self-Generating Memetic Algorithms (and SG Metaheuristics in general) are a great niche for GP!

Put back the memetic into Memetic Algorithms!!

Thanks to P. Merz, W.E. Hart, M. Land for allowing me to reproduce material from their Ph.D thesis and papers.

Bibliography

KdlCP+98

E.H.L. Aarts and G.A. Verhoeven. Chapter g9.5: Genetic local search for the traveling salesman problem. In T. Back, D.B. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation*, pages G9.5:1–7. IOP publishing Ltd and Oxford Unviersity Press, 1997.

J.M. Baldwin. A new factor in evolution. *American Naturalist*, 30, 1896.

E.K. Burke, S. Gustafson, G. Kendall, and N. Krasnogor. Advance population diversity measures in genetic programming. In *Parallel Problem Solving from Nature VII. LNCS.* Springer-Verlag, 2002.

O.M. Becker and M. Karplus. The topology of multidimensional potential energy surfaces: Theory and application to peptide structure and kinetics. *J. Chemical Physics*, 106(4):1495–1517, 97.

M.A. Bedau and N.H.Packard. Measurement of evolutionary activity, teleology and life. In D. Farmer S. Rasmussen C.G. Langton, C. Taylor, editor, *Artificial Life II*, volume 98-03-023, pages 431–461. Addison-Wesley, 1992.

E.K. Burke and J.P. Newall. A multistage evolutionary algorithm for the timetable problem. *IEEE Transactions On Evolutionary Computation*, 3(1), April 1999.

K. Boese. *Models For Iterative Global Optimization*. Ph.D. Thesis, UCLA Computer Science Department, 1996.

J. Berger, M. Sassi, and M. Salois. A hybrid genetic algorithm for the vehicle routing problem with time windows and itinerary constraints. In W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakaiela, and R.E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*. Morgan Kaufamann, 1999.

D. Costa, A. Hertz, and O. Dubouis. Embedding of a sequential procedure within an evolutionary algorithm for coloring problems in graphs. *Journal of Heuristics*, 1:105–128, 1995.

P. Cowling, G. Kendall, and E. Soubeiga. A hyperheuristic approach to scheduling a sales summit. In E. Burke and W. Erben editors, editors, *Theory of Automated Timetabling PATAT 2000, Springer Lecture Notes in Computer Science,*, pages 176–190. Springer, 2001.

P. Cowling, G. Kendall, and E. Soubeiga. Hyperheuristics: a tool for rapid prototyping in scheduling and optimisation. In *Proceedings of the 2nd European Conference on Evolutionary Computation, EvoCop 2002. Lecture notes in computer science.* Springer, 2002.

L.L. Cavalli-Sforza and M.W. Feldman. *Cultural Transmission and Evolution: A Quatitative Approach*. Princeton University Press, Princeton, NJ., 1981.

J. Culberson. On the futility of blind search: An algorithmic view of "no free lunch". *Evolutionary Computation*, 6(2):109–128, 198.

B. Dengiz, F. Altiparmak, and A.E. Smith. Local search genetic algorithm for optimal design of reliable network. *IEEE Transactions On Evolutionary Computation*, 1(3), September 1997.

R. Dawkins. *The Selfish Gene*. Oxford University Press, New York, 1976.

R. Dawkins. The extended phenotype. 1982.

G. Dozier, J. Bowen, and A. Homaifar. Solving constraint satisfaction problems using hybrid evolutionary search. *IEEE Transactions On Evolutionary Computation*, 2(1), April 1998.

R. Dorne and J.K. Hao. A new genetic local search algorithm for graph coloring. In *Parallel Problem Solving From Nature V*, volume 1498, pages 745–754, Amsterdam, Holland, 1998.

W.H. Durham. *Coevolution: Genes, Culture and Human Diversity*. Stanford University Press, 1991.

C. Fleurent and J.A. Ferland. Genetic Hybrids for the Quadratic Assignment Problem. In

DIMACS Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, 1993.

C. Fleurent and J.A. Ferland. Genetic and hybrid algorithms for graph coloring. *Annals of Operations Research*, 63:437–461, 1997.

B. Freisleben and P. Merz. A Genetic Local Search Algorithm for Solving Symmetric and Asymmetric Traveling Salesman Problems. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 616–621. IEEE Press, 1996.

B. Freisleben and P. Merz. New Genetic Local Search Operators for the Traveling Salesman Problem. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Proceedings of the 4th Conference on Parallel Problem Solving from Nature - PPSN IV*, volume

1141 of *Lecture Notes in Computer Science*, pages 890–900. Springer, 1996.

D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, 1987.

D.E. Goldberg and S. Voessner. Optimizing global-local search hybrids. In W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakaiela, and R.E. Smith, editors, *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, 1999.

W. E. Hart. Adaptive global optimization with local search. *Ph.D. Thesis, University of California, San Diego*, 1994.

William E. Hart, Thomas E. Kammeyer, and Richard K. Belew. The role of development in genetic algorithms. In L. Darrell Whitley and Michael D. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 315–332, San Fransico, CA, 1995. Morgan Kaufmann Publishers, Inc.

D. Holstein and P. Moscato. Memetic algorithms using guided local search: A case study. In D. Corne, F. Glover, and M. Dorigo, editors, *New Ideas in Optimization*. McGraw-Hill, 1999.

P. Hansen and N. Mladenovic. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, (130):44
467, 2001.

G.E. Hinton and S.J. Nowlan. How learning can guide evolution. *Complex Systems*, 1:495–502, 1987.

D.S. Johnson, C.H. Papadimitriou, and M. Yannakakis. How easy is local search. *Journal of Computer And System Sciences*, 37:79–100, 1988.

P. Jog, J.Y. Suh, and D. Van Gucht. The Effects of Population Size, Heuristic Crossover and Local Improvement on a Genetic Algorithm for the Travelling Salesman Problem. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 110–115. Morgan Kaufman, 1989.

J.D. Knowles and D.W. Corne. M-paes: A memetic algorithm for multiobjective optimization. *Proceedings of the Congress on Evolutionary Computation (CEC2000)*, 1:325–332, 2000.

N. Krasnogor, E. de la Cananl, D.A. Pelta, D.H Marcos, and W.A. Risi. Encoding and

crossover mismatch in a molecular design problem. In P. Bentley, editor, *AID98: Proceedings of the Workshop on Artificial Intelligence in Design 1998*, 1998.

N. Krasnogor, W.E. Hart, J. Smith, and D.A. Pelta. Protein structure prediction with evolutionary algorithms. In W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakaiela, and R.E. Smith, editors, *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference*. Morgan Kaufman, 1999.

K.W.C. Ku and M.W. Mak. Empirical analysis of the factors that affect the baldwin effect. *PPSN-V: Parallel Problem Solving From Nature, Proceedings 1998. Lecture Notes in Computer Science*, 1998.

J.D. Knowles. *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization.*

PhD thesis. Department of Computer Science, University of Reading, UK., 2002.

N. Krasnogor and D.A. Pelta. Fuzzy memes in multimeme algorithms: a fuzzy-evolutionary hybrid. In J.L. Verdegay, editor, *Book chapter in "Fuzzy Sets based Heuristics for Optimization"*. Physica Verlag, 2002.

Natalio Krasnogor, David Pelta, Daniel H. Marcos, and Walter A. Risi. Protein structure prediction as a complex adaptive system. In *Proceedings of Frontiers in Evolutionary Algorithms 1998*, 1998.

N. Krasnogor. http://dirac.chem.nott.ac.uk/~natk/public/papers.html. In *Studies on the Theory and Design Space of Memetic Algorithms*. Ph.D. Thesis, University of the West of England, Bristol, United Kingdom., 2002.

N. Krasnogor and J.E. Smith. A memetic algorithm with self-adaptive local search: Tsp as a case study. In *Proceedings of the 2000 Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, 2000.

M.W.S. Land. Evolutionary algorithms with local search for combinatorial optimization. *Ph.D. Thesis, University of California, San Diego*, 1998.

P. Merz. *Memetic Algorithms for Combinatorial Optimization Problems:Fitness Landscapes and Effecitve Search Strategies*. Ph.D. Thesis, Parallel Systems Research Group. Department of Electrical Engineering and Computer Science. University of Siegen., 2000.

P. Merz and B. Freisleben. Genetic Local Search for the TSP: New Results. In *Proceedings of the 1997 IEEE International Conference on*

Evolutionary Computation, pages 159–164. IEEE Press, 1997.

P. Merz and B. Freisleben. On the Effectiveness of Evolutionary Search in High–Dimensional NK-Landscapes. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, pages 741–745. IEEE Press, 1998.

P. Merz and B. Freisleben. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In *Proceedings of the 1999 International Congress of Evolutionary Computation (CEC'99), Washington DC, USA*, 1999.

P. Merz and B. Freisleben. Genetic algorithms for binary quadratic programming. In *Proceedings of the 1999 International Genetic and Evolutionary Compuation Conference (GECCO'99), Orlando, USA*, 1999.

H. Muhlenbein, M. Gorges-Schleuter, and O. Kramer Evolution algorithms in combinatorial optimization. *Parallel Computing*, 7:65–85, 1988.

T. Murata, H. Ishibuchi, and M. Gen. Specification of local search directions in genetic local search algorithms for multi-objective optimization problems. In W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakaiela, and R.E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*. Morgan Kaufamann, 1999.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

C.L. Morgan. On modification and variation. *Science*, 4:733–740, 1896.

P. A. Moscato. On evolution, search, optimization, genetic algorihtms and martial arts:

Towards memetic algorithms. Technical Report Caltech Concurrent Computation Program Report 826, Caltech, Caltech, Pasadena, California, 1989.

P. Moscato. Memetic algorithms: A short introduction. In D. Corne, F. Glover, and M. Dorigo, editors, *New Ideas in Optimization*. McGraw-Hill, 1999.

P.A. Moscato. Problemas de otimizac$\tilde{a}$o np, aproximabilidade e computac$\tilde{a}$o evolutiva:da pr$\acute{a}$tica $\grave{a}$ teoria. *Ph.D Thesis, Universidade Estadual de Campinas,Brasil*, 2001.

E. Marchiori and C. Rossi. A flipping genetic algorithm for hard 3-sat problems. In W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakaiela, and R.E. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*. Morgan Kaufamann, 1999.

Y. Nagata and S. Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. *Proc. of the 7th Int'l Conf. on GAs*, 1997.

H.F. Osborn. Ontogenic and phylogenic variation. *Science*, 4:786–789, 1896.

Christopher D. Rosin, Scott Halliday, William E. Hart, and Richard K. Belew. A comparison of global and local search methods in drug docking. In Thomas Baeck, editor, *Proc 7th Intl Conf on Genetic Algorithms (ICGA97)*, pages 221–228, San Francisco, CA, 1997. Morgan Kaufmann.

R. Salomon. Evolutionary algorithms and gradient search: Similarities and differences. *IEEE Transactions On Evolutionary Computation*, 2(2), July 1998.

S. Tsutsui and A. Ghosh. Genetic algorithms with a robust solution searching scheme. *IEEE Transactions On Evolutionary Computation*, 1(3), September 1997.

P.D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2, 1995.

P.D. Turney. How to shift bias: Lessons from the baldwin effect. *Evolutionary Computation*, 4(3):271–295, 1996.

P.D. Turney. Myths and legends of the baldwin effect. *13th International Conference on Machine Learning (ICML96), Workshop on Evolutionary Computation and Machine Learning*, pages 135–142, 1996.

P.D. Turney, D. Whitley, and R.W Anderson. Evolution, learning, and instinct: 100 years of

the baldwin effect. *Evolutionary Computation*, 4(3):iv–viii, 1996.

F. Vavak, T.C. Fogarty, and K. Jukes. A genetic algorithm with variable range of local search for tracking changing environments. In H.M. Voigt, I. Rechenberg, and H.P. Schwefel, editors, *Lectures Notes in Computer Science - Vol 1141*. Springer-Verlag. Parallel Problem Solving From Nature - PPSN IV.

C.H. Waddington. *The Strategy of Genes*. Allen and Unwin, London, 1957.

A. Weismann. The germ-plasm: A theory of heredity. *New York: Scribners*, 1893.

D.H. Wolpert and W.G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, technical reports, 1995.

J. Watson, C. Ross, V. Eisele, J. Denton, J. Bins, C. Guerra, D. Withley, and A. Howe. The traveling salesrep problem, edge assembly crossover, and 2-opt. *PPSN-V: Parallel Problem Solving From Nature, Proceedings 1998. Lecture Notes in Computer Science*, 1998.

M. Yannakakis. Computational complexity. In E. Aarts and J.K. Lenstra, editors, *Local Search in Combinatorial Optimization*, pages 19–55. John Wiley & Sons Ltd., 1997.