Lecture L3:
Constraints & Theoretical Underpinnings

- Constraint handling
  - Penalty function approach
  - Penalty-parameter-less approach
  - Multi-objective approaches (later)
- Theoretical underpinnings
  - Schema theorem
  - Sizing of EA parameters
  - Better understanding of EA operators and their interactions

Constraint Handling in EAs

- Penalty function approach is common
  - Inequality constraints $(g_j(x) \geq 0)$ penalized for violation:

$$F(x) = f(x) + \sum_{j=1}^{J} R_j \langle g_j(x) \rangle^2$$

  - <a> = a if a is –ve, 0 otherwise
  - Equality constraints:
    - Simple: Replace a variable by using it
    - Complex: Convert $|h_k(\mathbf{x})| \leq \varepsilon_k$

Penalty Function Appoach (cont.)

- Optimized solution largely depends on R_j
- To reduce number of penalty parameters, use

$$F(x) = f(x) + R \sum_{j=1}^{J} \langle \hat{g}_j(x) \rangle^2$$

  - $\hat{g}(x)$ is a normalized constraint
- Penalized function distorts the original objective

Penalty Function Approach: An Example

- A welded-beam design problem:

Minimize $f_w(h,l,t,b) = 1.10471h^2l + 0.04811tb(14.0+l)$

Subject to
$g_1(x) \equiv 13{,}600 - \tau(x) \geq 0$
$g_2(x) \equiv 30{,}000 - \sigma(x) \geq 0$
$g_3(x) \equiv b - h \geq 0$
$g_4(x) \equiv P_c(x) - 6000 \geq 0$
$g_5(x) \equiv 0.25 - \delta(x) \geq 0$

Constraints are normalized

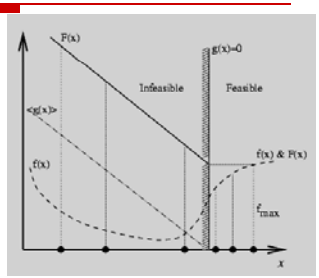
R	≤ 50%	Infeasible	Best	Median	Worst
10^0	12	13	2.41324	7.62465	483.50177
10^1	12	0	3.14206	4.33457	7.45453
10^3	1	0	3.38227	5.97060	10.65891
10^6	0	0	3.72929	5.87715	9.42353

Penalty Approaches

- Smaller R, lesser importance to constraint satisfaction
  - Final solution may be infeasible
- Larger R, more importance to constraint satisfaction
  - Final solution may be feasible, but likely to be away from optimum
- Dynamically changing R (with iteration)
  - Predefined R(t)
  - On the fly, depending on extent of constraint violation
- Multi-objective formulation (discuss later)

Biasing Approach: A Penalty-Parameter-less Method

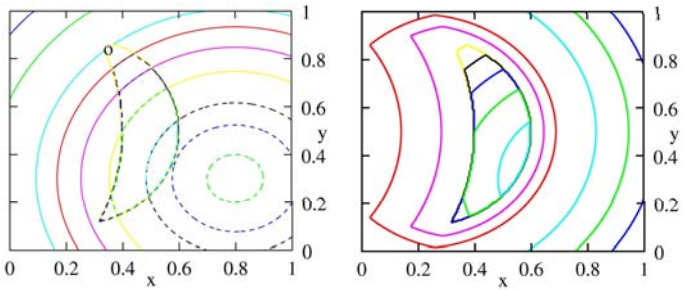
- Modify tournament sel.:
  - A feasible is better than an infeasible
  - For two feasibles, choose the one with better *f*
  - For two infeasibles, choose the one with smaller constraint violation $\left(\sum_j \langle g_j(x) \rangle\right)$



$$F(x) = \begin{cases} f(x), & \text{if } g_j(x) \geq 0, \forall j \in J \\ f_{\max} + \sum_{j=1}^{J} \langle g_j(x) \rangle, & \text{otherwise} \end{cases}$$

An Illustration of Fitness Function

Objective function *f* | Fitness function *F*

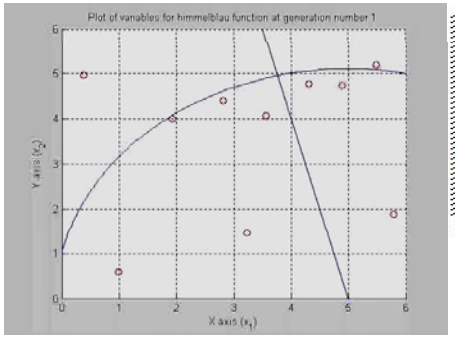


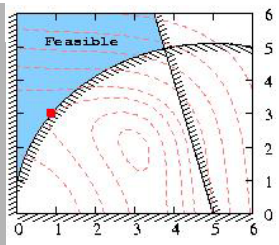
- No need for any penalty parameter

Practically Speaking

- Given a solution
- What would a designer do first?
- Probably, check if it is feasible
- If yes, compute its objective value (say, cost)
- If no, who cares how much it costs!
- Objective function is not computed if a solution is infeasible
  - Only find how *much* constraint is violated

A Computer Simulation





Global search ended up with constraint following

Test Problem 1

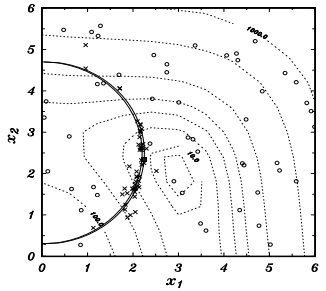
Minimize $f_0(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$
Subject to $g_1(x) \equiv 4.84 - x_1^2 - (x_2 - 2.5)^2 \geq 0$
$g_2(x) \equiv (x_1 - 0.05)^2 + (x_2 - 2.5)^2 - 4.84 \geq 0$
$0 \leq x_1 \leq 6, 0 \leq x_2 \leq 6$

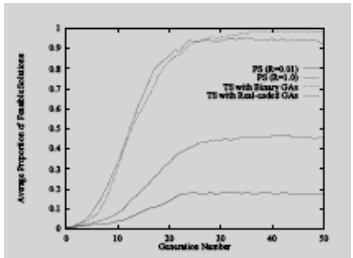
- Opt. Soln.: (2.246826, 2.381865), f=13.59085.
- 0.7% of the search space feasible

Method	ϵ ≤ 1%	≤ 2%	≤ 5%	≤ 10%	≤ 20%	≤ 50%	> 50%	Infeasible
PS ($R = 0.01$)	5	5	7	8	15	23	8	19
PS ($R = 1$)	17	19	20	20	24	32	7	11
TS-B	2	2	5	6	9	13	37	0
TS-R	29	31	31	32	33	39	11	0

Method	Best	Median	Worst
PS ($R = 0.01$)	13.58958	24.07437	114.69033
PS ($R = 1$)	13.59108	16.35284	172.81369
TS-B	13.59658	37.90495	244.11616
TS-R	13.59085	13.61673	117.02971

Results on Test Problem 1





Test Problem 2

Minimize $f_2(x) = 0.1365 - 5.843 \times 10^{-7} y_{17} + 1.17 \times 10^{-4} y_{14}$
$+2.358 \times 10^{-5} y_{13} + 1.502 \times 10^{-6} y_{16} + 0.0321 y_{12}$
$+0.004324 y_5 + 1.0 \times 10^{-4} c_{15}/c_{16} + 37.48 y_2/c_{12},$
Subject to $g_1(x) \equiv 1.5x_2 - x_3 \geq 0,$
$g_2(x) \equiv y_1(x) - 213.1 \geq 0,$
$g_3(x) \equiv 405.23 - y_1(x) \geq 0,$
$g_{j+2}(x) \equiv y_j(x) - a_j \geq 0, \quad j = 2, \ldots, 17,$
$g_{j+18}(x) \equiv b_j(x) - y_j(x) \geq 0, \quad j = 2, \ldots, 17,$
$g_{36}(x) \equiv y_4(x) - 0.28/0.72 y_5(x) \geq 0,$
$g_{37}(x) \equiv 21 - 3496 y_2(x)/c_{12}(x) \geq 0,$
$g_{38}(x) \equiv 62212/c_{17}(x) - 110.6 - y_1(x) \geq 0,$
$704.4148 \leq x_1 \leq 906.3855$
$68.6 \leq x_2 \leq 288.88,$
$0 \leq x_3 \leq 134.75,$
$193 \leq x_4 \leq 287.0966,$
$25 \leq x_5 \leq 84.1988.$

- Opt. function value : -1.90513

TS-R is the proposed approach

Method	Best	Median	Worst
PS ($R = 1$)	−1.86365	−1.69507	−1.35910
PS (R=10^2)	−1.898445	−1.651555	−1.009691
PS(R=10^6)	−1.91319	−1.65763	−1.11550
TS-B	−1.663156	−1.204836	−0.730438
TS-R	−1.91319	−1.65763	−1.11550
TS-R	−1.91410	−1.85504	−1.30643
TS-R	−1.91460	−1.91457	−1.91454

Welded-Beam Design Problem Revisited

Method	Muta-tion	shar ing	ε ≤ 1%	≤ 2%	≤ 5%	≤ 10%	≤ 20%	≤ 50%	> 50%
Maximum generations = 500									
TS-B	No	No	0	0	0	0	0	0	50
TS-B	No	Yes	0	0	0	0	0	0	50
TS-R	No	No	0	0	1	4	8	16	34
TS-R	No	Yes	28	36	44	48	50	50	0
Maximum generations = 4,000									
TS-R	No	Yes	28	37	44	48	50	50	0
TS-R	Yes	Yes	50	50	50	50	50	50	0

Case	Best	Median	Worst
1	4.939027	7.183082	12.084255
2	3.820984	8.899963	14.298933
3	2.442714	3.834121	7.444246
4	2.381191	2.392892	2.645833
5	2.381187	2.392025	2.645833
6	2.381447	2.382634	2.383554

Comparison on Test Problems

Summary of Results

Prob No.	True optimum	Best-known Best	Best-known Median	Best-known Worst	Results of this study Best	Results of this study Median	Results of this study Worst
2	−1.90511(*)	−1.90513	–	–	−1.915	−1.915	−1.915
3	−15.000	−15.000	−15.000	−15.000	−15.000	−15.000	−13.000
4	7049.331	7485.667	8271.292	8752.412	7060.221	7220.026	10230.834
5	680.630	680.642	680.718	680.955	680.634	680.642	680.651
6	30665.5	30005.7	–	–	30665.650	30665.648	29770.943
7	24.306	24.690	29.258	36.060	24.372	24.409	25.075
Weld	2.381	–	–	–	2.381	2.383	2.384

Equivalent or better than existing methods

In a Practical Set-Up

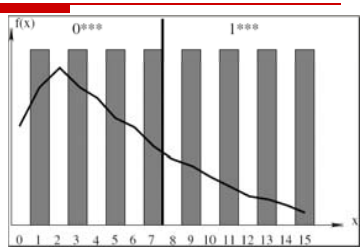
- Some constraints more important than others
  - Preferential addition of CV
  - Most important to least, one by one
  - Least important to most, one by one
  - Differentiate soft versus hard constraints
  - Soft constraints as objectives

Theoretical Underpinnings

- EAs are stochastic, multi-faceted, iterative procedures
- Not easy to do a perfect mathematical modeling and analysis
- Three approaches:
  - Functional decomposition approach: Engineering approach
  - Statistical dynamics approach: Dynamics of population statistics
  - Dynamical systems approach: Markov-chain approach
- Asymptotic convergence proof with elite-preservation and a finite mutation (Rudolph, 1996)
- Breadth of EA's application domain is the main reason for its popularity -> Does an airplane provably fly?

Theoretical Understanding Based on Concept of a Schema

- Schema: Similarity among strings
- Example: H_1=1*** Competitor H_0=0***
- Defining length, δ(H): Difference between outer-most defined positions
  - Example: δ(*1*0*)=2
- Order, o(H): Number of defined positions
  - Example: o(*1*0*)=2



John Holland (1975)

Schema Theorem

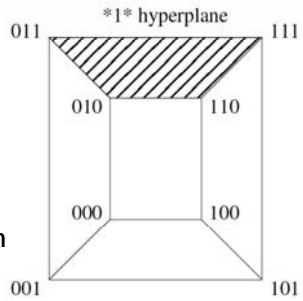
- Proportionate selection, 1-pt crossover, bit-wise mutation (Goldberg, 1989):

$$m(H,t+1) \geq m(H,t)\frac{f(H)}{\overline{f}}\left[1-p_c\frac{\delta(H)}{\ell-1}-p_m o(H)\right]$$

- Short, low-order schemata (building blocks) grow logistically
- More accurate schema theorem exists (see FOGA proceedings)

Holland's Schema Processing Argument

- Hyperplane sampling through string manipulation
- GA decides which hyper-plane to emphasize through iterations
- 2^l to $n2^l$ schemata get processed in a population of size n
- Parallel processing



A Hand Calculation

- Maximize $\sin(x)$ $\quad 0 \leq x \leq \pi$

Initial population						
String	DV[a]	x	$f(x)$	f_i/f_{avg}	AC[b]	Mating pool
01001	9	0.912	0.791	1.39	1	01001
10100	20	2.027	0.898	1.58	2	10100
00001	1	0.101	0.101	0.18	0	10100
11010	26	2.635	0.485	0.85	1	11010
	Average, f_{avg}		0.569			

Mating Pool	CS[c]	New population			
		String	DV[a]	x	$f(x)$
01001	3	01000	8	0.811	0.725
10100	3	10101	21	2.128	0.849
10100	2	10010	18	1.824	0.968
11010	2	11100	28	2.838	0.299
				Average, f_{avg}	0.710

[a] DV, decoded value of the string.

[b] AC, actual count of strings in the population.

[c] CS, cross site.

- Example: $H_1 = (1\ 0\ *\ *\ *)$

$$m(H_1, 1) \geq (1) \cdot \frac{0.898}{0.569}\left[1 - (1.0)\frac{1}{5-1} - (0.0)(2)\right],$$
$$= 1.184.$$

Functional Decomposition Approach

- Study one aspect of EAs at a time, keeping others fixed
- Wright brother's approach to first successful flight (Goldberg's approach)
- Approximate models, but provides useful insight:
  - GA operator selection
  - GA parameter interactions
  - Population sizing

Analyzing Selection Operator Alone

- Goldberg and Deb (1990) calculated take-over characteristic time
- Generations taken for the best string to occupy all but one population member and complexity (C)
- Proportionate
  - t^s_{prop}=n log n, C=O(n log n)
- Ranking
  - t^s_{rnk}=log n + log(ln n), C=O(n log n) + sl. time
- Tornament
  - t^s_{tour}= ln n/ln s, C=O(n)
- Tournament and ranking selections have better take-over time

Analyzing Crossover Operator Alone

- Random yet structured information exchange
- Single-point crossover:

  1 1 1|1 1 1 1 → 1 1 1|0 0 0 0
  0 0 0|0 0 0 0 → 0 0 0|1 1 1 1
- Two-point Crossover

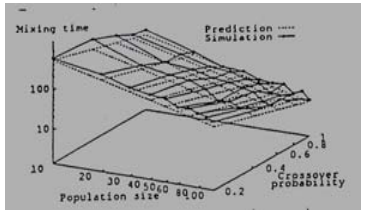
  1 1 1|1 1|1 1 → 1 1 1|0 0|1 1
  0 0 0|0 0|0 0 → 0 0 0|1 1|0 0
- Uniform crossover

  1 1 1 1 1 1 1 → 1 0 1 0 0 1 0
  0 0 0 0 0 0 0 → 0 1 0 1 1 0 1

Less structure ↓ More search

Mixing Time Study

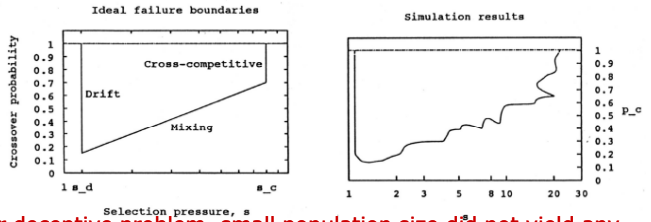
- Goldberg, Deb, Thierens (1993) and Thierens and Goldberg (1993) calculated mixing time
- Generations taken to get a particular string by recombination alone
- $t^m_{1\text{-pt}} = 1/(p_c n p^\ell)$, ℓ=string length



Putting Them Together

- For EAs to work, selection take-over time and mixing time must be of the same order:

$$p_c \geq A \ln s$$



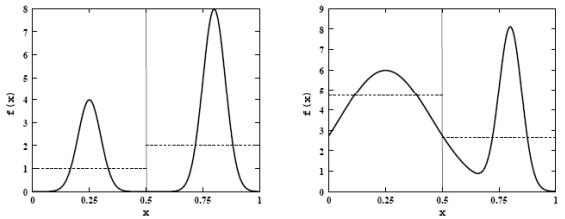
For deceptive problem, small population size did not yield any sweet-spot

Building Block Based EA Procedures

- Goldberg, Deb, and Clark (1989)
- Functional decomposition of two problems
  - Building block discovery
    - Initialization (deterministic or probabilistic)
  - Building block exchange
    - Messy cut and splice or linkage learning concept
- Plethora of theses at IlliGAL under Dave Goldberg
- Extended to ordering problems, multi-objective problems
- Hierarchical EAs (Watson and others)

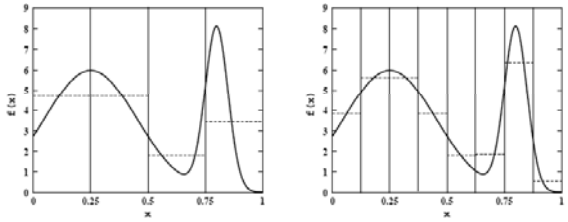
Population Sizing

- Statistical sampling theorem for ensuring a correct schema competition
- Population size of 2k is not enough for right problem



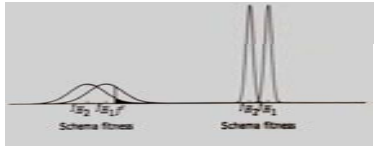
Population Sizing (cont.)

- A population size of 8k is required to start with correct schema competition
- Population size depends on complexity of the problem



Population Sizing (cont.)

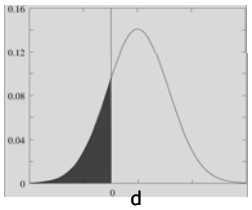
- Limit error in building block sampling
- Consider two BBs ($f_{H1} > f_{H2}$)



- Limit error in building block sampling
- Consider two BBs ($f_{H1} > f_{H2}$)
- Example: H: **f*ff***
- Assume normal approximation in variation of fitness
- Hope to choose a string containing H1 more often

Population Sizing (cont.)

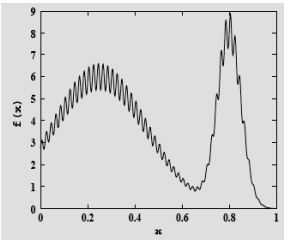
- Error: Shaded area
- Difference, $d = f_{H1} - f_{H2}$ is normal
- Equate error to desired limit α
- $n = 2c2^{o(H)}\sigma^2/d^2$
- Difficult to use in practice
- But some approximation can be made by sampling



Other Models exist:
Gambler's ruin model

Population Sizing (cont.)

- A large variability in function demands a large population size
- $N \approx 2c2^{o(H)}\ell/(SNR)^2$
- SNR: Signal to noise ratio
  - Signal: Diff. in global and local optima
  - Noise: Variance

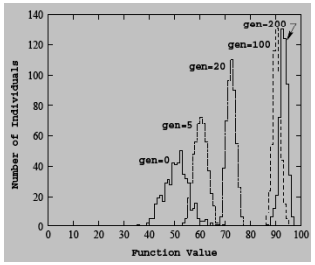


- For small o(H), $N = k\ell$

However, if started with a non-random population, much smaller population size is needed

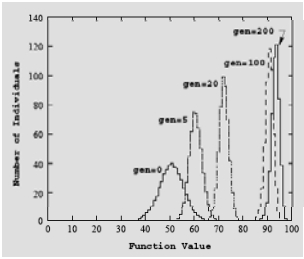
Statistical Dynamics Approach

- Starting from a random population how EAs reach the optima?
- Added difficulty: A population to track, not a point
- Population variance reduces with generation
- Rate of progress reduces with generation



Statistical Dynamics Approach (cont.)

- Model population mean, variance, kurtosis, skewness etc.
- Tracking of individual solution (best) requires more effort
- Modeling one population statistics is also difficult
- Model ensemble population statistics



Modeling Procedure

$$\begin{pmatrix} \overline{\mu}(t) \\ \overline{\sigma}(t) \\ \overline{\kappa_2}(t) \\ \overline{\kappa_3}(t) \end{pmatrix} \xrightarrow{\text{Selection}} \begin{pmatrix} \overline{\mu}^{(s)}(t) \\ \overline{\sigma}^{(s)}(t) \\ \overline{\kappa_2}^{(s)}(t) \\ \overline{\kappa_3}^{(s)}(t) \end{pmatrix} \xrightarrow{\text{Crossover}} \begin{pmatrix} \overline{\mu}^{(c)}(t) \\ \overline{\sigma}^{(c)}(t) \\ \overline{\kappa_2}^{(c)}(t) \\ \overline{\kappa_3}^{(c)}(t) \end{pmatrix}$$

$$\xrightarrow{\text{Mutation}} \begin{pmatrix} \overline{\mu}^{(m)}(t) \\ \overline{\sigma}^{(m)}(t) \\ \overline{\kappa_2}^{(m)}(t) \\ \overline{\kappa_3}^{(m)}(t) \end{pmatrix} = \begin{pmatrix} \overline{\mu}(t+1) \\ \overline{\sigma}(t+1) \\ \overline{\kappa_2}(t+1) \\ \overline{\kappa_3}(t+1) \end{pmatrix}.$$

- Such a model is problem dependent

Model for One-Max Problem

$$\begin{aligned}
\mu^{(s)}(t) &= \mu(t) + \frac{\sigma(t)}{\sqrt{\pi}}, \\
{\sigma^{(s)}}^2(t) &= \left(1 - \frac{1}{\pi}\right)\sigma^2(t), \\
\mu^{(m)}(t) &= p_m\ell + (1-2p_m)\mu^{(s)}(t), \\
{\sigma^{(m)}}^2(t) &= p_m(1-p_m)\ell + (1-2p_m)^2{\sigma^{(s)}}^2(t), \\
\mu(t+1) &= p_m\ell + (1-2p_m)\mu(t) + \frac{(1-2p_m)}{\sqrt{\pi}}\sigma(t), \\
\sigma^2(t+1) &= p_m(1-p_m)\ell + (1-2p_m)^2\left(1 - \frac{1}{\pi}\right)\sigma^2(t).
\end{aligned}$$

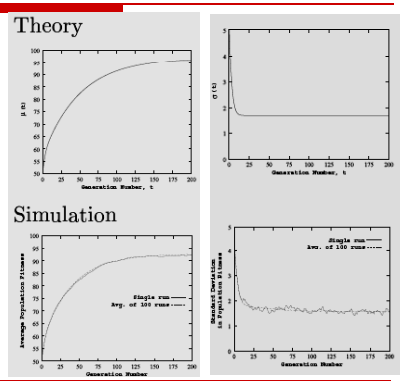
- $\mu(0) = \ell/2$ and $\sigma^2(0) = \ell/4$.

Results

- Attractor:

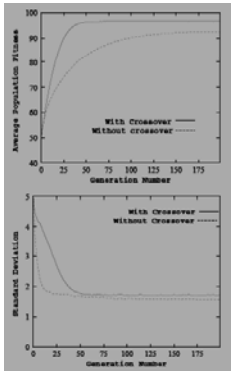
$$\mu^* = \ell/2 + \frac{1-2p_m}{2p_m\sqrt{\pi}}\sigma^*$$

$$\sigma^* = \sqrt{\frac{p_m(1-p_m)\ell}{1-(1-2p_m)^2(1-1/\pi)}}$$



Effect of Crossover

- Two phases of a selecto-mutation GA:
  - Fast reduction in variance due to strong selection pressure
  - Loss in variance due to selection is balanced by gain due to mutation
- Crossover introduces variability (σ increases)



Dynamical Systems Approach

- Markov chain approach, where every individual member is tracked
- Michael D. Vose (Simple GA: Foundation and Theory, MIT Press, 1999)
- Binary strings: $s=2^\ell$; a_i number of string i
- A population p is a s-dimensional vector with $p_i = a_i/N$
- Example: (00 10 10 11 10 10 10 11 11 10) -> $p=(0.1,0.0,0.6,0.3)^T$
- Three observations:
  - $p_i \geq 0$
  - $\sum_i p_i = 1$
  - $p_i \bmod (1/N) = 0$

Dynamical Systems Approach (cont.)

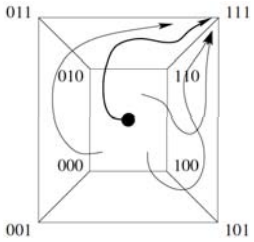
- Infinite versus finite population
- Model of a GA:

$$\frac{\mathcal{C}\mathcal{M}D}{\bar{f}(p(t))}p(t) = p(t+1)$$
$$\mathcal{C}\mathcal{M}Dp(t) = \bar{f}(p(t))p(t+1)$$

- At the fixed-point, $Ap = \lambda p$, an eigenvalue problem
- Perron-Frobenius theorem suggests for positive A
  - there exists only one positive eigenvector of A
  - That eigenvector corresponds to the largest eigenvalue
- Thus, there is only one infinite population fixed-point

Evolution Path

- Infinite population of random strings:
  - Starting point: centre
- Population with one string
  - One of the corners
- EA starts from centre and moves to one of the corners (hopefully, at opt.)
- Finite population: start away from centre
  - 5, 100 pop GAs are better than 1, 500 pop GA
  - Island with migration

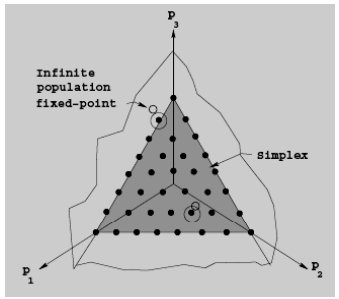


Finite Population Model

- Infinite population fixed point p* may not be a feasible finite population vector
- Example: $p^*=(0.2156, 0.1762, 0.5432, 0.0650)^T$ is not valid for N=10
- Assuming a force |p-p*|, all populations for which the force is small will attract GAs
- $p=(0.2, 0.2, 0.5, 0.1)^T$ is that population
- Population best is not easy to track
- Stability of attractors can be evaluated

Meta-Stable Attractors

- Other infeasible eigenvectors close to simplex may make some feasible populations more probable attractors
- Mutation ensures traversing every state in a finite time
- GAs may wander from one fixed-point to another in such cases
  - Fixed-point can be outside the simplex



Some Theoretical Results

- Population size (N) ∝ Number of variables (*n*)
- High crossover ($p_c \in [0.7,1.0]$), low mutation ($p_m \sim 1/n$)
- Loss in variance due to selection is balanced by gain due to crossover and mutation
- EA an act of exploitation versus exploration
- Mutation ensures traversing every state in a finite time
- Crossover useful in complex problems, but representation-dependent
- GAs wander from one fixed-point to another with finite populations
- Steady-state EAs are faster than generational EAs

GA Parameter Interactions

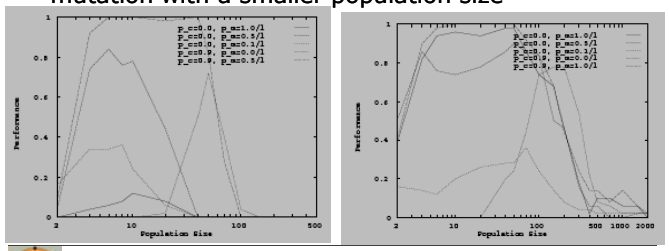
- Deb and Agrawal (FOGA, 1998)
- Coding, operators, and parameter setting are all inter-related
- Some GAs find good coding on the fly (messy GAs, linkage-learning GAs, etc.)
- Expensive, in general
- Theory lacks 'control maps' showing appropriate GA parameters for success
- Fixed total evaluations, vary population size

Problem Difficulties

- Problem difficulties causing trouble to a GA:
  - Multimodality, Deception, Isolation, Collateral noise, others
- Multimodality causes difficulty with many false attractors
- Deception causes difficulty with misleading low-order schema information
- Isolation causes difficulty with no useful low-order schema information
- Collateral noise causes difficulty by hiding good low-order schema information
- Chosen functions with first two difficulties here

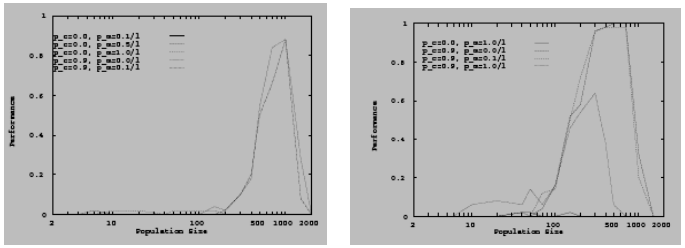
One-max and Himmelblau Problem

- Performance: proportion of successful runs
- Both crossover and mutation works well
- Crossover works with a larger population size, mutation with a smaller population size



Rastrigin and Deceptive Problems

- Only crossover works
- Requires an adequate population size



Important Results of the Study

- For unimodal or simple functions, mutation-based GAs performed better than crossover-based GAs with similar computational cost
- However, for fixed computational cost, mutation-based GAs require smaller population size than crossover-based GAs
- Too small or too large populations are detrimental
- GAs with both crossover and mutation operators perform better than either alone
- For more difficult problems, mutation-based GAs do not work well, crossover-based GAs are more reliable



Lack of a Rigorous Proof: What should we do?

- Need to show effect of GA parameter settings
  - Usually, fix # of overall evaluations (H)
  - Different population sizes (N)
  - Set maximum generations accordingly (H/N)
  - Different crossover and mutation probabilities
- Each case rerun from at least 10+ different initial populations
- Show best, median and worst performance
- Perform statistical tests to provide confidence to your results
- Will demonstrate in this course



Conclusions

- Population approach allows efficient constraint handling using EAs
- Different modes of theory based on rigor
- Asymptotic convergence proof provides confidence
- Computational complexity estimate exists for many classes of problems
- Lack of progress on rigorous theory did not thwart EA advancement and popularity

