Reading

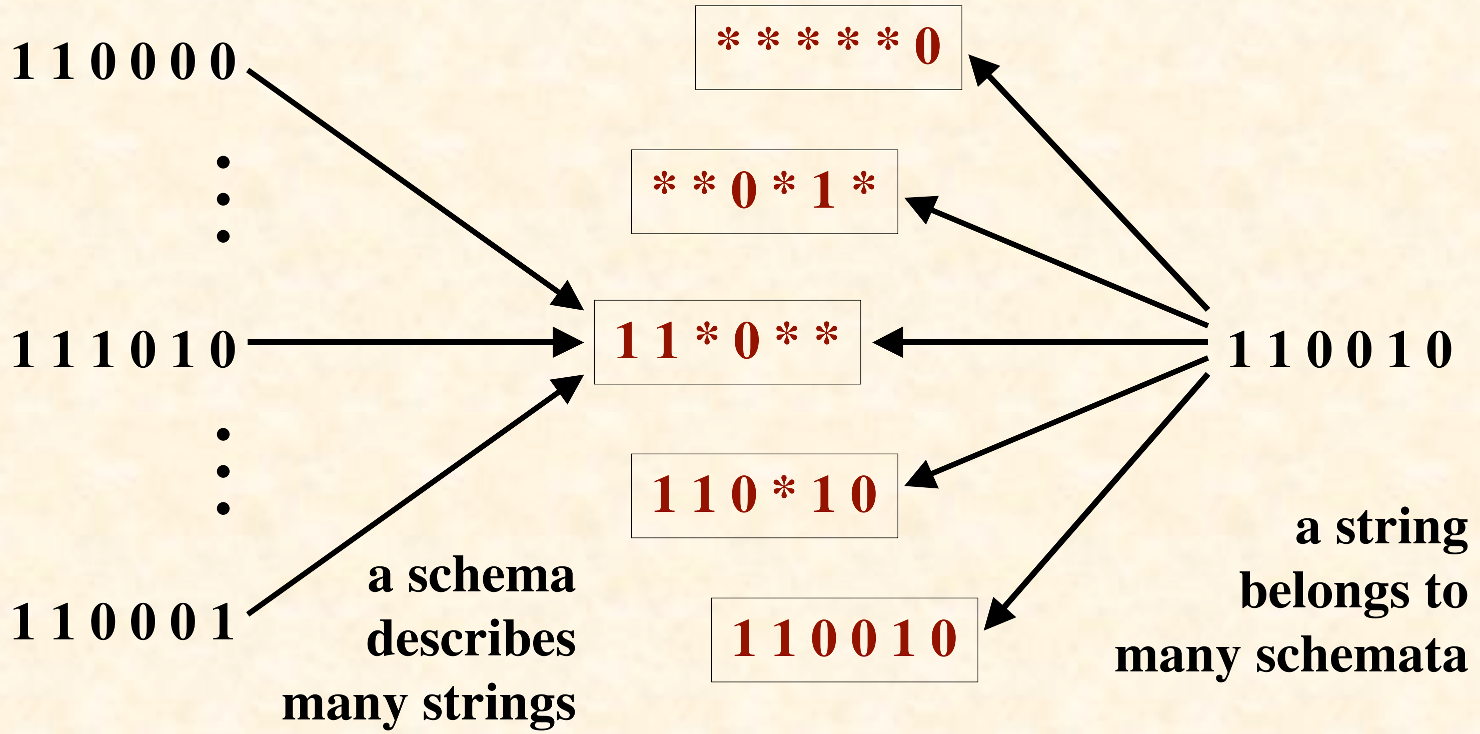
- CS 420/594: Read Flake, ch. 22 (Neural Networks and Learning)
- CS 594: Read Bar-Yam, sec. 2.3 (Feedforward Networks)

Why Does the GA Work?

The Schema Theorem

Schemata

A schema is a description of certain patterns of bits in a genetic string



110000

⋮

111010

⋮

110001

*****0

**0*1*

11*0**

110*10

110010

110010

a schema describes many strings

a string belongs to many schemata

The Fitness of Schemata

- The schemata are the building blocks of solutions
- We would like to know the average fitness of all possible strings belonging to a schema
- We cannot, but the strings in a population that belong to a schema give an estimate of the fitness of that schema
- Each string in a population is giving information about all the schemata to which it belongs (implicit parallelism)

Effect of Selection

Let n = size of population

Let $m(S,t)$ = number of instances of schema S at time t

String i gets picked with probability $\dfrac{f_i}{\sum_j f_j}$

Let $f(S)$ = avg fitness of instances of S at time t

So expected $m(S,t+1) = m(S,t) \cdot n \cdot \dfrac{f(S)}{\sum_j f_j}$

Since $f_{\text{av}} = \dfrac{\sum_j f_j}{n}$, $\quad m(S,t+1) = m(S,t)\dfrac{f(S)}{f_{\text{av}}}$

Exponential Growth

- We have discovered:
  $m(S, t+1) = m(S, t) \cdot f(S) / f_{\mathrm{av}}$
- Suppose $f(S) = f_{\mathrm{av}}\ (1 + c)$
- Then $m(S, t) = m(S, 0)\ (1 + c)^t$
- That is, exponential growth in above-average schemata

Effect of Crossover

1 … 0*

|□ □□|

- Let □ = length of genetic strings
- Let □(*S*) = defining length of schema *S*
- Probability {crossover destroys *S*}: $p_\mathrm{d} = □(S) / (□ – 1)$
- Let p_c = probability of crossover
- Probability schema survives:

$$p_\mathrm{s} \geq 1 □ \, p_\mathrm{c} \frac{□(S)}{□□1}$$

Selection & Crossover Together

$$m(S,t+1) \geq m(S,t)\frac{f(S)}{f_{\mathrm{av}}}\left[1 - p_{\mathrm{c}}\frac{\delta(S)}{\ell - 1}\right]$$

Effect of Mutation

- Let p_m = probability of mutation
- So $1 - p_\mathrm{m}$ = probability an allele survives
- Let $o(S)$ = number of fixed positions in S
- The probability they all survive is $(1 - p_\mathrm{m})^{o(S)}$
- If $p_\mathrm{m} << 1$, $(1 - p_\mathrm{m})^{o(S)} \approx 1 - o(S)\, p_\mathrm{m}$

Schema Theorem: “Fundamental Theorem of GAs”

$$m(S,t+1) \geq m(S,t)\frac{f(S)}{f_{\mathrm{av}}}\left[1 - p_{\mathrm{c}}\frac{\delta(S)}{\ell - 1} - o(S)\,p_{\mathrm{m}}\right]$$

The Bandit Problem

- Two-armed bandit:
  - random payoffs with (unknown) means m_1, m_2 and variances $\square_1$, $\square_2$
  - optimal strategy: allocate exponentially greater number of trials to apparently better lever
- *k*-armed bandit: similar analysis applies
- Analogous to allocation of population to schemata
- Suggests GA may allocate trials optimally

Goldberg’s Analysis of Competent & Efficient GAs

Paradox of GAs

- Individually uninteresting operators:
  - selection, recombination, mutation
- Selection + mutation □ continual improvement
- Selection + recombination □ innovation
  - generation vs.evaluation
- Fundamental intuition of GAs: the three work well together

Race Between Selection & Innovation: Takeover Time

- Takeover time t^* = average time for most fit to take over population
- Transaction selection: top $1/s$ replaced by s copies
  - s quantifies selective pressure
- Estimate $t^* \approx \ln n / \ln s$

Innovation Time

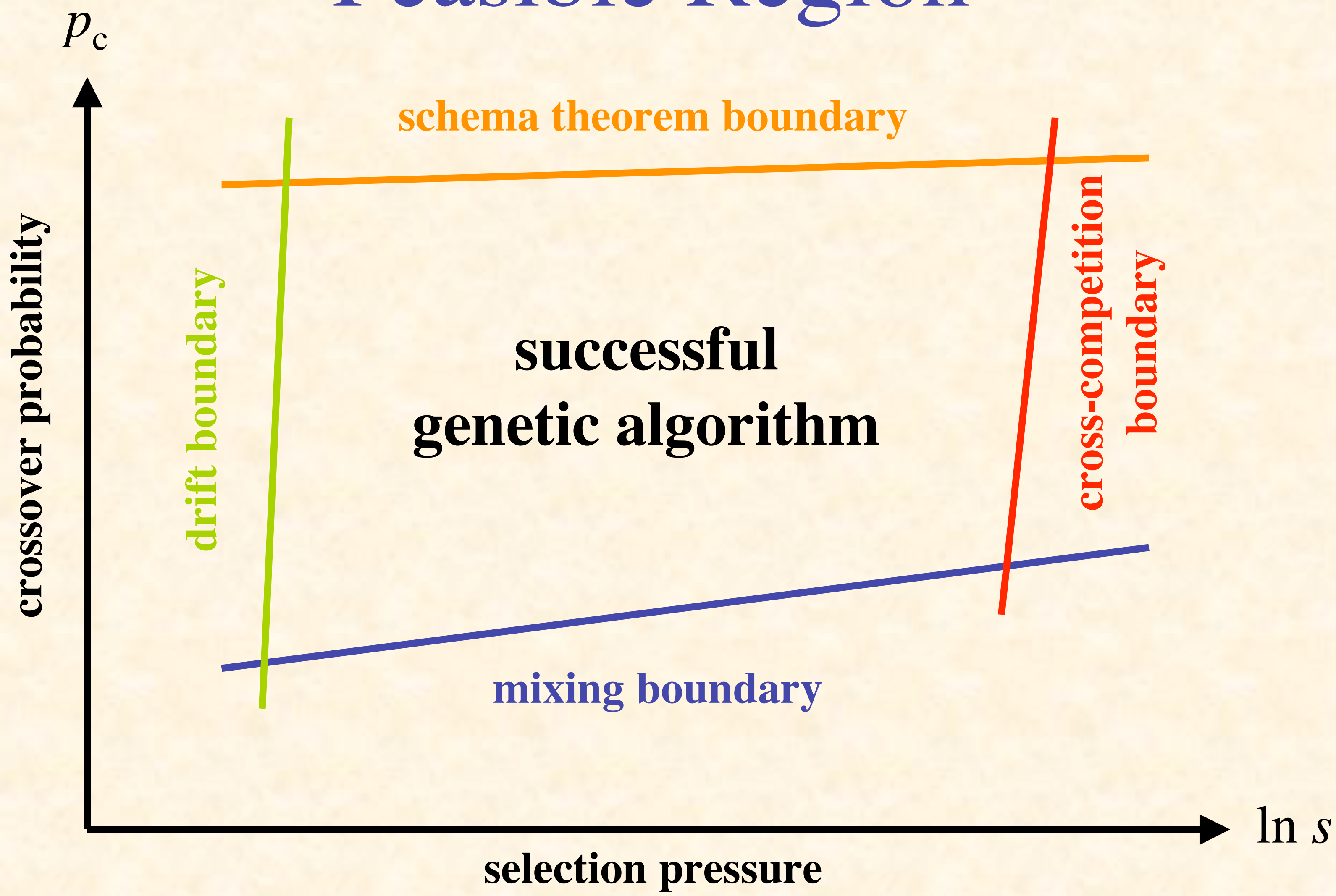
- Innovation time t_{i} = average time to get a better individual through crossover & mutation
- Let p_{i} = probability a single crossover produces a better individual
- Number of individuals undergoing crossover = $p_{\mathrm{c}}\, n$
- Probability of improvement = $p_{\mathrm{i}}\, p_{\mathrm{c}}\, n$
- Estimate: $t_{\mathrm{i}} \approx 1 / (p_{\mathrm{c}}\, p_{\mathrm{i}}\, n)$

Steady State Innovation

- Bad: $t^* < t_\text{i}$
  - because once you have takeover, crossover does no good
- Good: $t_\text{i} < t^*$
  - because each time a better individual is produced, the t^* clock resets
  - *steady state innovation*
- Innovation number:

$$\text{Iv} = \frac{t^*}{t_\text{i}} = p_\text{c} p_\text{i} \frac{n \ln n}{\ln s} > 1$$

Feasible Region



Other Algorithms Inspired by Genetics and Evolution

- Evolutionary Programming
  - natural representation, no crossover,time-varying continuous mutation
- Evolutionary Strategies
  - similar, but with a kind of recombination
- Genetic Programming
  - like GA, but program trees instead of strings
- Classifier Systems
  - GA + rules + bids/payments
- and many variants & combinations…

Additional Bibliography

1. Goldberg, D.E. *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*. Kluwer, 2002.
2. Milner, R. *The Encyclopedia of Evolution*. Facts on File, 1990.