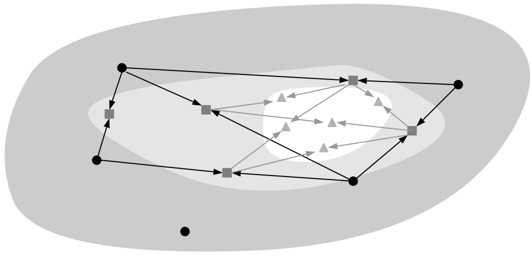
GA/GP Search

Often GAs and GP search like this:

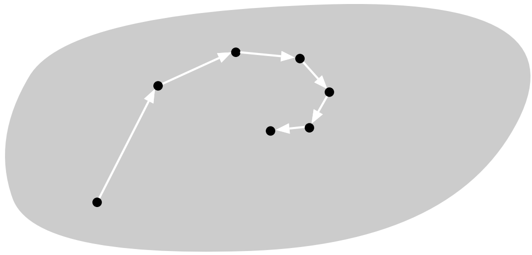


each point is an individual, pairs of arrows represent crossovers.

Question: How can we characterise/study/predict this search?

Microscopic Dynamical System Models

- *A possible approach:* We represent the population as a point in a multidimensional space, and we study the trajectory of this point (e.g. using Markov chain theory)



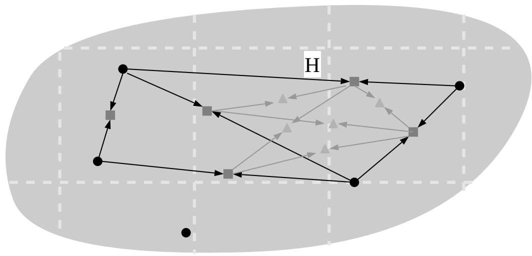
- Typically this leads to exact models with huge numbers of parameters (microscopic models).

Schema Theories

An alternative approach:

- We divide the search space into subspaces (*schemata*)
- We characterise the schemata using macroscopic quantities
- We study how and why the individuals in the population move from one subspace to another (*schema theorems*).

Example



- A reasonable search descriptor might be the proportion of individuals within a given schema H at generation t, $\Phi(H,t)$.
- Then a *schema theorem* would model mathematically how and why $\Phi(H,t)$ varies from one generation to the next.
- E.g. $\Phi(H,0) = 0$, $\Phi(H,1) = 2/5$, $\Phi(H,2) = 3/5$: why?

Classical (Pessimistic) GA Schema Theory

- In binary GAs, *syntactically* a schema H is a string of symbols from the alphabet {`0,1,#`}. E.g. H= `#10#1`
- *Semantically*, # is a "don't care" symbol and the schema is the set of binary strings matching the corresponding symbolic template. E.g. H={`01001,01011,11001,11011`}.
- Schema theorem for one-point crossover and mutation:

$$E[\Phi(H,t+1)] \geq \underbrace{p(H,t)}_{\textit{Selection}} \underbrace{(1-p_m)^{\mathcal{O}(H)}}_{\textit{Mutation}} \underbrace{\left[1 - p_{xo} \times \frac{\mathcal{L}(H)}{N-1} \times \sigma\right]}_{\textit{Crossover}}$$

where

	(Holland, 1975)	(Goldberg, 1989)	(Whitley, 1994)
σ	$1-\Phi(H,t)$	1	$1-p(H,t)$

$E[]$ = expectation operator

$\Phi(H,t)$ = proportion of individuals in H at generation t

$p(H,t)$ = selection probability for strings in H at generation t

p_m = mutation probability

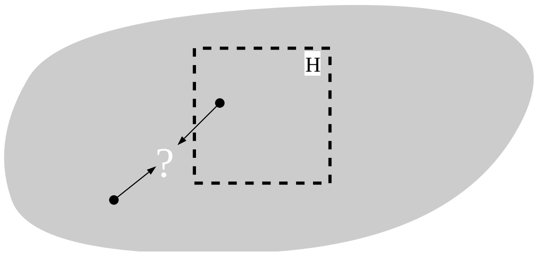
$\mathcal{O}(H)$ = schema order, i.e. number of defining bits

p_{xo} = crossover probability

$\mathcal{L}(H)$ = defining length, i.e. distance between the furthest defining bits in H

N = bit-string length

- Idea:



- Features:
  - The theorem includes an expected value (unless infinite populations are used)
  - It provides a lower bound (unless crossover and mutation are absent)

  So, it is difficult to use it to make accurate predictions.

Stephens and Waelbroeck's Exact GA Schema Theory (1997)

- For a binary GA with one point crossover applied with probability p_{xo} (and assuming $p_m = 0$)

$$E[\Phi(H,t+1)] = (1-p_{xo})p(H,t) + \frac{p_{xo}}{N-1}\sum_{i=1}^{N-1} p(L(H,i),t)p(R(H,i),t)$$

- $L(H,i)$ is the schema obtained by replacing the elements of H to the *right* of position i with "don't care" symbols
- $R(H,i)$ is the schema obtained by replacing the elements of H to the *left* of position $i+1$ with "don't care" symbols
- For example, if H =`1#111`, $L(H,1)$ =`1####`, $R(H,1)$ =`##111`, $L(H,3)$ =`1#1##`, $R(H,3)$ =`###11`.

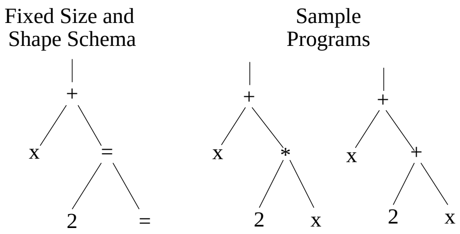
- For $N = 3$ and for the schema #11, the theorem gives:

$$\begin{aligned} E[\Phi(\texttt{\#11},t)] &= (1-p_{xo})p(\texttt{\#11},t) + \\ &\quad \frac{p_{xo}}{2}(p(\texttt{\#\#\#},t)p(\texttt{\#11},t) + p(\texttt{\#1\#},t)p(\texttt{\#\#1},t)) \\ &= (1-\frac{p_{xo}}{2})p(\texttt{\#11},t) + \frac{p_{xo}}{2}p(\texttt{\#1\#},t)p(\texttt{\#\#1},t), \end{aligned}$$

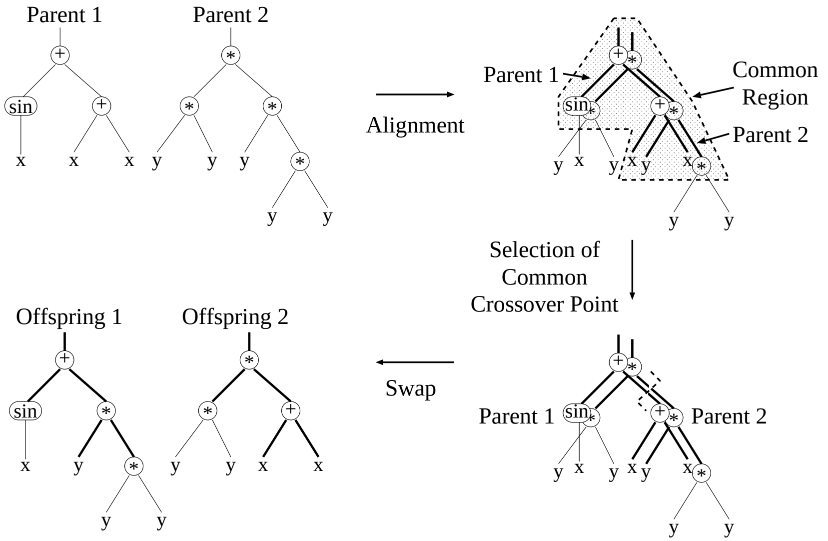
since $p(\texttt{\#\#\#},t) = 1$.

Poli and Langdon's GP Schemata (1997)

- A *GP schema* is a tree composed of functions from the set $\mathcal{F} \cup \{=\}$ and terminals from the set $\mathcal{T} \cup \{=\}$.
- The symbol $=$ is a "don't care" symbol which stands for a *single* terminal or function.
- For example, assuming that $\mathcal{F}$={+, -} and $\mathcal{T}$={x, y}:



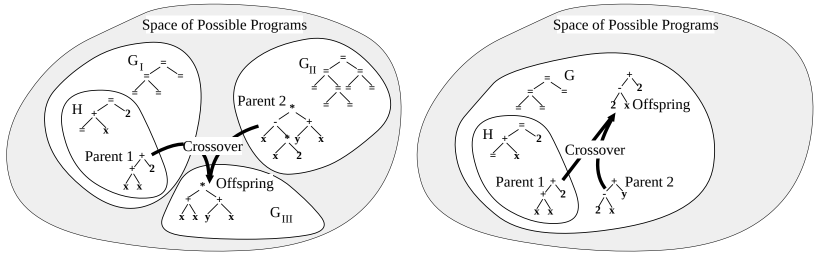
One-point Crossover in GP



Pessimistic GP S.T. for 1-pt XO (1997)

$$E[m(H,t+1)] \geq M \cdot \underbrace{p(H,t)}_{\text{Selection}} \cdot \underbrace{(1-p_m)^{\mathcal{O}(H)}}_{\text{Mutation}} \cdot$$

$$\left\{ 1 - p_{xo} \left[\underbrace{p_{\text{diff}}(t)\,(1-p(G(H),t))}_{\text{Crossover with parent } \notin G(H)} + \underbrace{\frac{\mathcal{L}(H)}{N(H)}\,(p(G(H),t)-p(H,t))}_{\text{Crossover with parent } \in G(H)} \right] \right\}$$



- $G(H)$ is the zero-th order schema with the same structure of H where all the defining nodes in H have been replaced with "don't care" symbols

- $p_{\text{diff}}(t)$ is the conditional probability that H is disrupted by crossover when the second parent does not sample $G(H)$ (normally $p_{\text{diff}}(t)$ is unknown and so it is replaced by 1).

- $p(H,t)$ is the probability of selection of the schema H

- $N(H)$ is the total number of nodes in the schema

- $\mathcal{O}(H)$ is the number of defining nodes in H

- $\mathcal{L}(H)$ is the number of links in the minimum tree fragment including all the non-$=$ symbols within a schema H

Exact GP schema theory for subtree-swapping crossovers (2001): backbone

- Let $\alpha(H,t)$ be the probability of generating new individuals in the schema H at generation t (*transmission probability*)

- Then an exact schema theorem has the form

$$E[\Phi(H,t+1)] = \alpha(H,t)$$

- Let us assume, for simplicity, that only reproduction and crossover are performed.

- Because these operators are mutually exclusive

$$\begin{aligned}\alpha(H,t) &= \Pr[\textit{An individual in } H \textit{ is obtained via reproduction}] \\ &+ \Pr[\textit{An offspring in } H \textit{ is produced by crossover}].\end{aligned}$$

- If reproduction is performed with probability p_r and crossover with probability p_{xo} (with $p_r + p_{xo} = 1$) then

$$\begin{aligned}\alpha(H,t) &= p_r \times \Pr\{\textit{An individual in } H \textit{ is cloned}\} \\ &+ p_{xo} \times \Pr\left\{\begin{array}{l}\textit{The parents and the crossover points} \\ \textit{are such that the offspring is in H}\end{array}\right\}\end{aligned}$$

- Clearly,

$$\Pr\{\textit{An individual in H is cloned}\} = p(H,t).$$

- We are now left with the hard part: calculating

$$\Pr\left\{\begin{array}{l}\textit{The parents and the crossover points} \\ \textit{are such that the offspring is in H}\end{array}\right\}$$

- **Observation**: in most crossovers the probability of choosing particular crossover points in the parents depends on the size and shape of such parents, *but not* on the actual primitives present in the two programs (*node invariance*).

- For example, the probability of choosing any crossover point in `(+ x (+ y x))` is often identical to the probability of choosing the corresponding crossover point in `(AND D1 (OR D1 D2))`.

- It makes sense to split the calculation by parent shape, i.e.

$$\Pr\left\{\begin{array}{l}\textit{The parents and the crossover points} \\ \textit{are such that the offspring is in } H\end{array}\right\}$$
$$= \sum_{\substack{\textit{For all} \\ \textit{pairs of} \\ \textit{parent} \\ \textit{shapes k,l}}} \Pr\left\{\begin{array}{l}\textit{The parents and the crossover points} \\ \textit{are such that the offspring is in } H \textbf{ and} \\ \textit{the parents have shapes } k \textit{ and } l\end{array}\right\}$$

- Then, using the node invariance property, we can write

$$\Pr\left\{\begin{array}{l}\textit{The parents and the crossover points}\\ \textit{are such that the offspring is in } H \textbf{ and}\\ \textit{the parents have shapes } k \textit{ and } l\end{array}\right\}$$

$$= \sum_{\substack{\textit{For all}\\ \textit{crossover}\\ \textit{points i,j}\\ \textit{in shapes}\\ \textit{k and l}}} \Pr\left\{\begin{array}{l}\textit{Choosing crossover points}\\ \textit{i and j in shapes k and l}\end{array}\right\}$$

$$\times \Pr\left\{\begin{array}{l}\textit{Selecting parents with shapes k and}\\ \textit{l, such that if crossed over at points}\\ \textit{i and j produce an offspring in H}\end{array}\right\}$$

- If we use a Koza-style subtree-swapping crossover where crossover points are selected with uniform probability, then

$$\Pr\left\{\begin{array}{l}\textit{Choosing crossover}\\ \textit{points i and j in}\\ \textit{shapes k and l}\end{array}\right\} = \frac{1}{\textit{(Nodes in shape k)}} \times \frac{1}{\textit{(Nodes in shape l)}}.$$

- So, we are left with the problem of calculating

$$\Pr\left\{\begin{array}{l}\textit{Selecting parents with shapes k and}\\ \textit{l, such that if crossed over at points}\\ \textit{i and j produce an offspring in H}\end{array}\right\}$$

- **Observation:** the offspring produced by crossover have the right shape and primitives to match H *if and only if*, after the excision of the chosen subtree, the first parent has shape and primitives *compatible* with H, and the subtree to be inserted has shape and primitives compatible with H.

- That is:

$$\Pr\left\{\begin{array}{l}\textit{Selecting parents with shapes k and l, such}\\ \textit{that if crossed over at points i and j produce}\\ \textit{an offspring in H}\end{array}\right\}$$

$$= \Pr\left\{\begin{array}{l}\textit{Selecting a root-donating parent}\\ \textit{with shape k such that its upper}\\ \textit{part w.r.t. crossover point i matches}\\ \textit{the upper part of H w.r.t. i}\end{array}\right\}$$

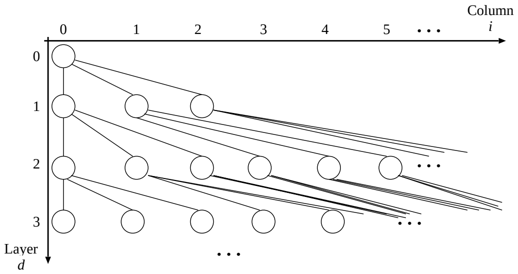
$$\times \Pr\left\{\begin{array}{l}\textit{Selecting a subtree-donating parent}\\ \textit{with shape l such that its lower part}\\ \textit{w.r.t. crossover point j matches the}\\ \textit{lower part of H w.r.t. i}\end{array}\right\}$$

- These two selection probabilities can be calculated exactly. However, the calculation requires the introduction of several other concepts and notation. These are also needed to generalise the theory to any subtree-swapping crossover.

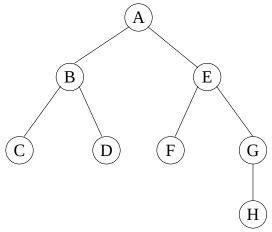
Theory for Subtree-swapping XO: nuts and bolts

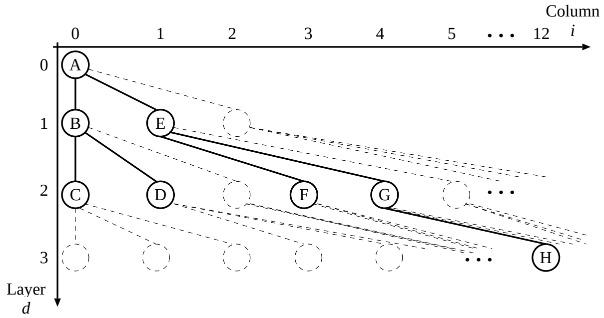
Node reference systems

- We need a method to indicate unambiguously the position of nodes in trees, i.e. we need a *node reference system*.
- Let us imagine the largest possible full tree that can be created with nodes of a given maximum arity, let us place the nodes in the tree into layers of increasing depth and align them to the left within a Cartesian coordinate system. E.g.



- Then we use the maximal system also to locate the nodes of non-maximal trees. E.g.





Functions over Node Reference Systems

- Given a node reference system it is possible to define functions over it.
- A program h can be seen as the function $N(d,i,h)$ which returns the label of the node in h stored at position (d,i). If (d,i) is not in h then $\emptyset$, is returned.

  E.g. if h=`(A (B C D) (E F (G H)))`, then $N(0,0,h) = \mathtt{A}$, $N(1,0,h) = \mathtt{B}$, $N(1,1,h) = \mathtt{E}$, $N(1,2,h) = \emptyset$, ...
- We can also define probability distributions over node reference systems, e.g.

  $$p(d,i|h) = \Pr\left\{\begin{array}{l}\text{A node at depth } d \text{ and column } i \text{ is selected}\\ \text{in program } h \text{ for some genetic operation}\end{array}\right\}$$

  where $p(d,i|h) = 0$ when (d,i) is not a node in h.

Modelling Subtree-swapping Crossovers

- It is possible to model crossover with

  $$p(d_1,i_1,d_2,i_2|h_1,h_2) = \Pr\left\{\begin{array}{l}\text{A node at coordinates } (d_1,i_1) \text{ is}\\ \text{selected in parent } h_1 \text{ and a node}\\ \text{at coordinates } (d_2,i_2) \text{ is selected}\\ \text{in parent } h_2\end{array}\right\},$$

  where $p(d_1,i_1,d_2,i_2|h_1,h_2) = 0$ if $N(d_1,i_1,h_1) = \emptyset$ or $N(d_2,i_2,h_2) = \emptyset$.
- If the crossover points are selected independently (*separable crossover*)

  $$p(d_1,i_1,d_2,i_2|h_1,h_2) = p(d_1,i_1|h_1) \cdot p(d_2,i_2|h_2)$$

- Standard crossover is separable and for uniform selection of XO points

$$p_{\text{Std}}(d_1, i_1, d_2, i_2 | h_1, h_2) = \frac{\delta(N(d_1, i_1, h_1) \neq \emptyset)\delta(N(d_2, i_2, h_2) \neq \emptyset)}{N(h_1)N(h_2)},$$

where $N(h)$ is the size of h

- One-point crossover is not separable and

$$p_{\text{1pt}}(d_1, i_1, d_2, i_2 | h_1, h_2) = \begin{cases} 1/\mathbf{NC}(h_1, h_2) & \text{if } (d_1, i_1) = (d_2, i_2) \text{ and} \\ & (d_1, i_1) \in C(h_1, h_2), \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathbf{NC}(h_1, h_2)$ is the number of nodes in the common region $C(h_1, h_2)$ between programs h_1 and h_2.

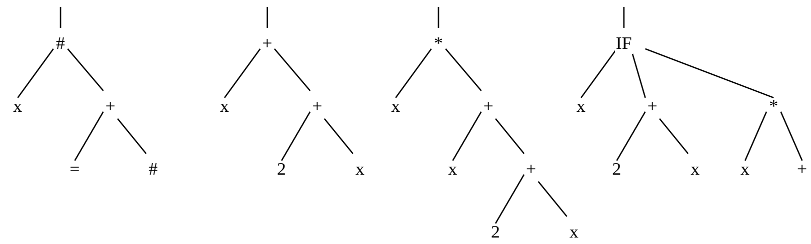
- Similarly, it is possible to model strongly typed GP crossover, context-preserving crossover, and most others.

Variable Arity Hyperschemata

- For each possible pair of XO points, we need to express the structure the parents must have to produce offspring in a particular schema. This can be done with VA hyperschemata.
- A *VA hyperschema* is a rooted tree composed of internal nodes from the set $\mathcal{F} \cup \{=, \#\}$ and leaves from $\mathcal{T} \cup \{=, \#\}$
- $\mathcal{F}$ and $\mathcal{T}$ are the function and terminal sets
- The symbol = is a "don't care" symbol which stands for exactly one node
- The terminal # stands for any valid subtree
- The function # stands for exactly one function of arity not smaller than the number of subtrees connected to it.

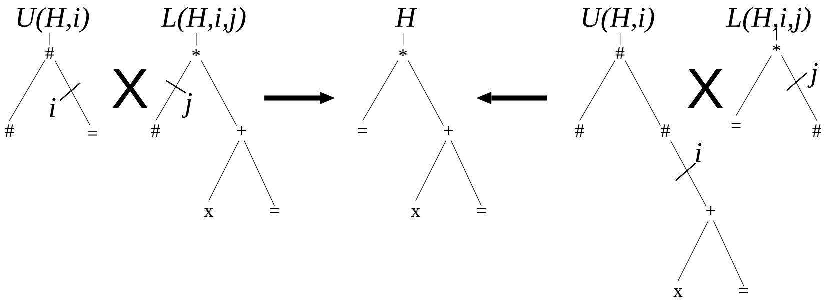
- Example: `(# x (+ = #))`

VA Hyperschema **Sample Instances**



Upper and Lower Building Blocks

- Using VA hyperschemata we can express which parents can produce instances of a schema of interest H:

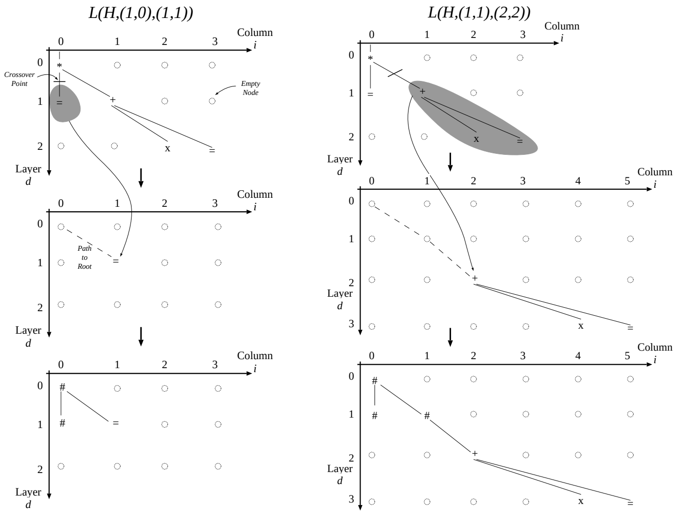


- If one crosses over at point j *any* individual matching $L(H, i, j)$ and at point i *any* individual matching $U(H, i)$, the resulting offspring is always an instance of H.
  Note, i and j are *coordinate pairs*.

Building the Building Blocks

- $U(H, i)$ is the VA hyperschema obtained from H by replacing the subtree below crossover point i with a # node.
- $L(H, i, j)$ is the VA hyperschema obtained by rooting at coordinate j in an empty reference system the subschema of H below crossover point i, then by labelling all the nodes on the path between node j and the root node with # function nodes, and labelling the arguments of those nodes which are to the left of such a path with # terminal nodes

- Examples:



Exact GP Schema Theorem for Subtree-swapping Crossover (2001)

- Under any node-invariant subtree-swapping crossover

$$\begin{aligned} E[\Phi(H, t+1)] &= (1 - p_{xo}) p(H, t) + \\ & \quad p_{xo} \sum_{k,l} \sum_{i \in H} \sum_{j} p(i, j | G_k, G_l) \times \\ & \quad p(U(H, i) \cap G_k, t) p(L(H, i, j) \cap G_l, t) \end{aligned}$$

where G_1, G_2, $\cdots$ are all the possible program shapes (i.e. schemata of order 0)

- Formally, a crossover operator is *node-invariant* if

$$p(i, j | h_1, h_2) = p(i, j | G(h_1), G(h_2))$$

Exact Schema Theorem for GP with Standard Crossover (2001)

- Standard crossover is node invariant so we can apply the previous theorem
- Under standard crossover with uniform selection of crossover points

$$\begin{aligned} E[\Phi(H, t+1)] &= (1 - p_{xo}) p(H, t) + \\ & p_{xo} \sum_{k,l} \frac{1}{N(G_k) N(G_l)} \\ & \sum_{i \in H \cap G_k} \sum_{j \in G_l} p(U(H, i) \cap G_k, t) p(L(H, i, j) \cap G_l, t). \end{aligned}$$

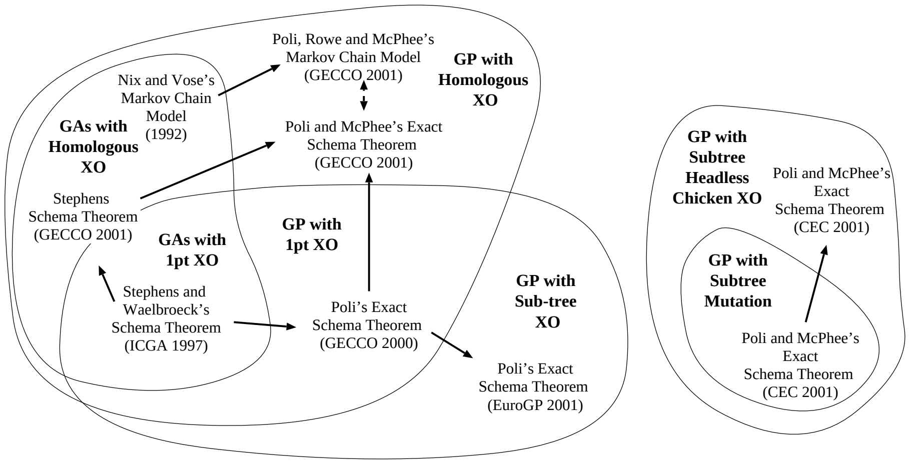
- Note, we have now filled the gaps in the schema theory backbone:

$$p(U(H,i)\cap G_k,t) \;=\; \Pr\left\{\begin{array}{l}\textit{Selecting a root-donating parent}\\ \textit{with shape } k \textit{ such that its upper}\\ \textit{part w.r.t. crossover point } i \textit{ matches}\\ \textit{the upper part of } H \textit{ w.r.t. } i\end{array}\right\}$$

$$p(L(H,i,j)\cap G_l,t) \;=\; \Pr\left\{\begin{array}{l}\textit{Selecting a subtree-donating parent}\\ \textit{with shape } l \textit{ such that its lower part}\\ \textit{w.r.t. crossover point } j \textit{ matches the}\\ \textit{lower part of } H \textit{ w.r.t. } i\end{array}\right\}$$

- The sets $U(H,i)\cap G_k$ and $L(H,i,j)\cap G_l$ are either the empty set or GP schemata.

Exact Schema Theories for Different Operators and Their Relations



Why Bother with Schema Theories?

- Typical objections/questions:
  - *Practitioners*: "OK, perhaps schema theories explain what happens and why in the next generation, but this does not help GP practitioners. The real question is: Which representation, operators, fitness function, search algorithm, population size, number of generations, number of runs, crossover probability, anti-bloat method, etc. should one use?"
  - *Theorists*: "What does this schema theory give me that another preexisting theory does not?"
- We have some answers.

Example: using the theory to predict/explain

- Assume $p_{xo} = 1$, $\mathcal{F} = \{A_f, B_f, C_f, D_f, E_f\}$ including only 1-arity functions, and $\mathcal{T} = \{A_t, B_t, C_t, D_t, E_t\}$.
- We can remove the subscripts without generating any ambiguity, obtaining the search space of variable-length strings over the alphabet $\{A, B, C, D, E\}$.
- Let us now consider the schema AB=. We want study its propagation in two populations, each one including 100 copies of the following strings (pop. size = 400):

Population 1	Fitness	Population 2	Fitness
AB	2	AB	2
BCD	2	BCD	2
ABC	4	ABC	4
ABCD	6	*BCDE*	6

- From the exact GP schema theorem for standard XO we get:

$$
\begin{aligned}
&E[\Phi(\texttt{AB=}, t+1)] = \\
&\quad \frac{1}{16}p(=\texttt{AB}=)p(====) + \frac{1}{16}p(==\texttt{B}=)p(\texttt{A}===) + \frac{1}{16}p(====)p(\texttt{AB}==) + \\
&\quad \frac{1}{12}p(=\texttt{B}=)p(\texttt{A}===) + \frac{1}{12}p(===)p(\texttt{AB}==) + \frac{1}{8}p(\texttt{B}=)p(\texttt{A}===) + \\
&\quad \frac{1}{8}p(==)p(\texttt{AB}==) + \frac{1}{4}p(=)p(\texttt{AB}==) + \frac{1}{12}p(=\texttt{AB}=)p(===) + \\
&\quad \frac{1}{12}p(==\texttt{B}=)p(\texttt{A}==) + \frac{1}{6}p(====)p(\texttt{AB}=) + \frac{1}{9}p(=\texttt{B}=)p(\texttt{A}==) + \\
&\quad \frac{2}{9}p(===)p(\texttt{AB}=) + \frac{1}{6}p(\texttt{B}=)p(\texttt{A}==) + \frac{1}{8}p(=\texttt{AB}=)p(==) + \\
&\quad \frac{1}{8}p(==\texttt{B}=)p(\texttt{A}=) + \frac{1}{3}p(\texttt{AB}=)p(==) + \frac{1}{6}p(=\texttt{B}=)p(\texttt{A}=) + \\
&\quad \frac{1}{4}p(\texttt{B}=)p(\texttt{A}=) + \frac{1}{4}p(=\texttt{AB}=)p(=) + \frac{2}{3}p(\texttt{AB}=)p(=)
\end{aligned}
$$

where $p(\cdot) = p(\cdot, t)$ for brevity.

- So, for population 1 $E[\Phi(\texttt{AB=}, t+1)] \approx 0.12$ and for population 2 $E[\Phi(\texttt{AB=}, t+1)] \approx 0.08$.

- Both are smaller than $\Phi(\texttt{AB=}, t) = 100/400 = 0.25$ despite the fact that the schema has above average fitness.

- The schema equation makes it clear why replacing `ABCD` with `BCDE` reduces the transmission probability α of `AB=`: because it zeroes the selection probabilities of some BBs (e.g. `AB==`).

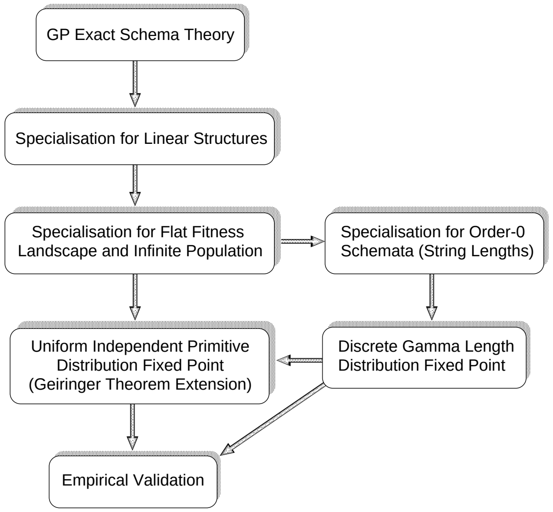
Operator Biases

- Exact schema theories can be used to understand the biases produced by the operators.

- For example, we can specialise and study the schema equations for the case where only one operator at the time is used.

- Selection is easy and well understood already. Crossover is hard, but can be studied by considering the case of a flat fitness landscape (no selection).

Studies of the biases of crossover in linear GP



Specialising the schema equations

- In linear GP schemata have the form $h_1 h_2 h_3 .... h_{N-1} h_N$ where some h_i can be "don't care" symbols (=)
- GP exact schema theorem for linear structures,

$$E[\Phi(h_1 ... h_N, t+1)] = (1 - p_{xo}) p(h_1 ... h_N, t) +$$
$$p_{xo} \sum_{k>0} \frac{1}{k} \sum_{i=0}^{\min(N,k)-1} p(h_1 ... h_i (=)^{k-i}, t)$$
$$\sum_{n=N-i}^{\infty} \frac{p((=)^{n-N+i} h_{i+1} ... h_N, t)}{n},$$

- Under the assumption of infinite populations the equation can be iterated numerically and/or its fixed points can be studied mathematically.

Crossover biases: Gamma length distribution

- Specialisation for flat landscape and infinite populations:

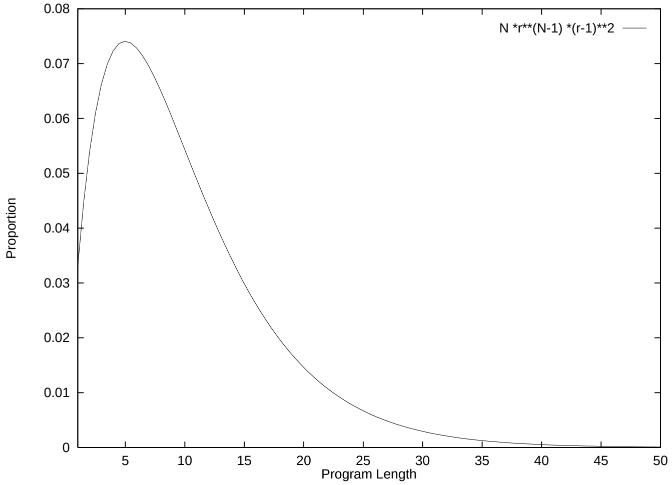
$$\Phi(h_1 ... h_N, t+1) = (1 - p_{xo}) \Phi(h_1 ... h_N, t) +$$
$$p_{xo} \sum_{k} \frac{1}{k} \sum_{i=0}^{\min(N,k)-1} \Phi(h_1 ... h_i (=)^{k-i}, t)$$
$$\sum_{n=N-i}^{\infty} \frac{\Phi((=)^{n-N+i} h_{i+1} ... h_N, t)}{n},$$

- In the case of a flat landscape evolution will continue until a fixed point length distribution is reached. This is a discrete Gamma distribution

$$\Phi((=)^N, \infty) = N r^{N-1} (r-1)^2,$$

where $r = (\mu - 1)/(\mu + 1)$, and μ is the initial average program length.

- For example the fixed-point length distribution for $\mu = 10$ is



Crossover biases: uneven search space sampling

- There are $|\mathcal{F}|^{x-1}|\mathcal{T}|$ different programs of length x in the search space, so the average probability that each of these will be sampled by standard crossover is

$$p_{\text{sample}}(x) = costant \times x r^{x-1} / |\mathcal{F}|^{x-1}$$

- For a flat landscape, standard GP will sample a particular short program much more often than it will sample a particular long one.
- E.g. if $|\mathcal{F}| = |\mathcal{T}| = 2$ and $\mu = 20$, on average GP will resample the same program of length 5 every 1054 crossovers, while it will resample the same program of length 50 every 3.33×10^{17} crossovers!

Crossover biases: allele diffusion

- The fixed point proportion of $h_1 h_2 \dots h_N$ is
$$\Phi(h_1 h_2 \dots h_N, \infty) = N r^{N-1} (r-1)^2 \times \prod_{i=1}^{N} c(h_i),$$
where $c(a)$ is the average number of (non-)terminal alleles of type a per (non-)terminal locus at generation 0.

- For non-terminal alleles of type a
$$c(a) = \frac{\sum_{n \geq 1} \Phi((\boxed{a} a)^n \; \boxed{a} =, 0)}{\mu(0) - 1},$$
for terminal alleles of type a
$$c(a) = \sum_{n \geq 0} \Phi((=)^n a, 0),$$
and in general $c(=) = 1$.

- Example if $\mathcal{C} = \{\sqrt{}, \mathrm{sin}, x, y, z\}$ and the population is initialised so that $c(\sqrt{}) = c(\mathrm{sin}) = 1/2$ and $c(x) = 1/3$, then $\Phi(\sqrt{\sqrt{\mathrm{sin}\, x}}, \infty) = (1/2)^2 \times (1/2) \times (1/3) = 1/24$.

- Note the similarity with Geiringer's theorem:
$$\Phi(h_1 h_2 \cdots h_N, \infty) = \prod_{i=1}^{N} \Phi(h_i, 0).$$

- Crossover attempts to push the population towards distributions of primitives where each primitive is equally likely to be found in any position in any individual.

- The primitives in a particular individual tend not just to be swapped with those of other individuals in the population, but also to diffuse *within* the representation of each individual.

- Experiments fully confirm the theory.

Summary of crossover biases

- Crossover pushes the population towards a specific (gamma) length distribution.

- Crossover generates both a "vertical" (between strings) primitive-mixing process and a "lateral" (within string) primitive-diffusion process.

- Vertical mixing behaviour is present in most crossover operators and is the reason why Geiringer's fixed point is a product.

- Lateral diffusion is a special feature of GP with subtree crossover. It pushes the population towards a state where the probability of finding a particular allele at a given locus is locus-independent.

Consequences for practitioners

- Because of primitive diffusion, the effects of any clever initialisation strategy can be quickly washed out.

- Lateral diffusion is somewhat analogous to mutation and may be a reason for the lack of convergence shown by standard GP.

- In some cases lateral diffusion can act as an automatic restart mechanism.

Multiple operators

- If different operators are used to create different members of the new population, the schema theorems for such operators can be combined very easily.
- For example, for GP using standard crossover with probability p_{xo}, full mutation with probability p_{FULL} and grow mutation with probability p_{GROW}, such that $p_{xo} + p_{\mathsf{FULL}} + p_{\mathsf{GROW}} = 1$, we have

  $$\alpha(H,t) = p_{xo} \times \alpha_{xo}(H,t) + p_{\mathsf{FULL}} \times \alpha_{\mathsf{FULL}}(H,t) + p_{\mathsf{GROW}} \times \alpha_{\mathsf{GROW}}(H,t)$$

  where the exact expressions for $\alpha_{xo}(H,t)$, $\alpha_{\mathsf{FULL}}(H,t)$ and $\alpha_{\mathsf{GROW}}(H,t)$ depend on the representation adopted.
- We can use this equation to study the interaction between operators

The one-then-zeros problem

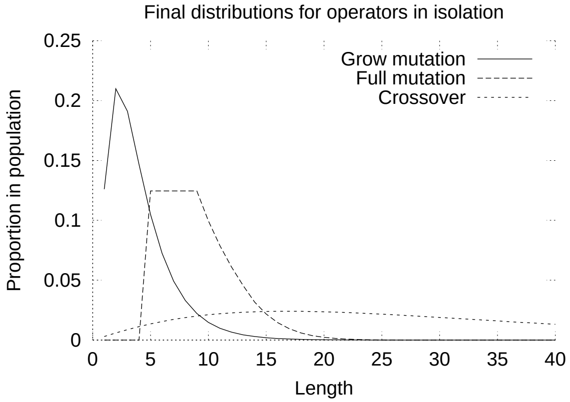
- In this problem we have $\mathcal{F} = \{0, 1\}$ and $\mathcal{T} = \{0\}$, where both 0 and 1 are taken to be unary operators, i.e. the representation is linear.
- Fitness in this problem will be 1 if the string starts with a 1 and has zeros elsewhere; fitness will be 0 otherwise.
- Under selection and crossover this problem induces bloat, while this does not happen under full and grow mutation.
- If we assume an infinite population, we can numerically iterate the schema equations to better understand the behavior of GP on this problem.

Experiments

- A depth limit $D = 5$ will be used for full mutation.
- Our initial population will consist of equal proportions (10% each) of the strings $1(0)^i$ for $1 \leq i \leq 10$.
- We iterated the schema equations multiple times using each of the legal combinations of proportions of (standard) crossover, grow mutation, and full mutation with values values from the set $\{0, 0.1, 0.2, 0.3, \ldots, 0.9, 1\}$.

Length Biases of the Operators

- The operators have strong biases when used alone.
- After 50 generations on the one-then-zeros problem:



Performance measures for the operators

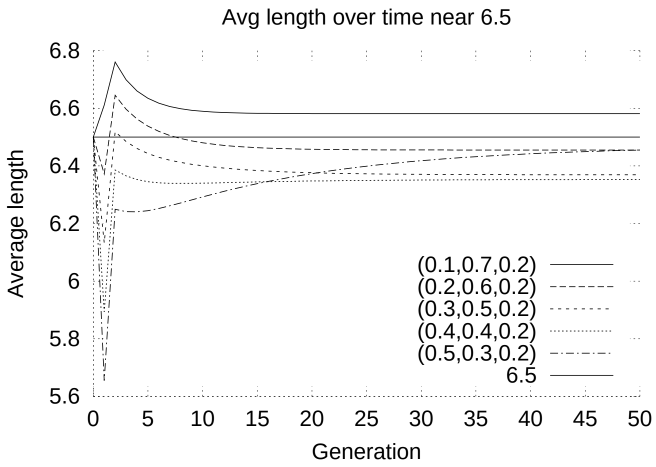
- We want to use the schema equations to guide the choices of the proportions of operators to help satisfy a variety of performance measures or goals:
  1. Avoid both bloating and shrinking, by having the average size after 50 generations be as close as possible to the initial average size.
  2. Avoid both bloating and shrinking (as above), but also maximize the number of correct individuals.
  3. Maximize the proportion of small solutions (as opposed to just short strings).
  4. Reach a state where the proportion of $1(0)^{30}$ exceeds 0.01 as early as possible.

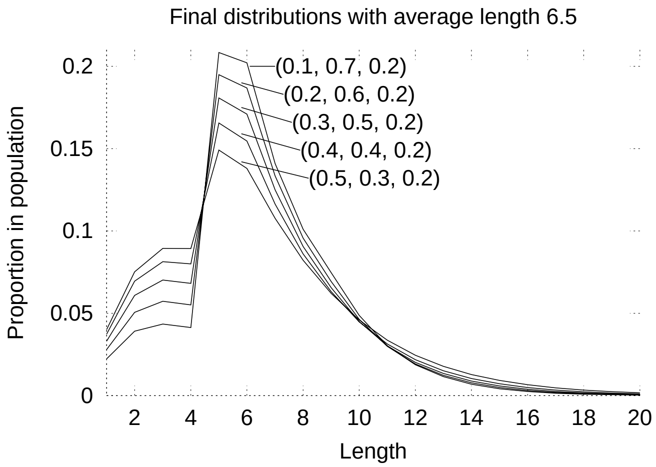
Avoid bloat and shrinkage

- Goal: avoid both bloating and shrinkage (i.e. average length after 50 generations is as close as possible to the initial average size).
- Five runs had a final average length that was less than 0.15 away from the initial average of 6.5:

XO	Full	Grow	Diff. from 6.5	Prop. fit
0.1	0.7	0.2	0.081	0.19
0.2	0.6	0.2	-0.045	0.26
0.3	0.5	0.2	-0.131	0.31
0.4	0.4	0.2	-0.147	0.37
0.5	0.3	0.2	-0.045	0.43

- In each case the proportion of grow mutation was 0.2.





Maximize proportion of small solutions

- Goal: minimize the average size of the *fit* individuals (i.e., those of the form $1(0)^a$ for $a > 0$).
- The three smallest are (0, 0, 1), (0.1, 0, 0.9), and (0.2, 0, 0.8) with final average sizes of fit individuals 3.02, 3.04, and 3.07 respectively
- So, on this problem the best way to make short correct solutions is to primarily use grow mutation, with small amounts of crossover being acceptable as well.

Find solutions of length 31 quickly

- Goal: find solutions of the form $1(0)^{30}$ as quickly as possible. We want a proportion of target individuals exceeding 0.01.
- Only four of our tested parameter settings ever obtain a proportion of at least 0.01 target individuals

XO	Full	Grow	First gen to 0.01
1.0	0.0	0.0	26
0.9	0.1	0.0	28
0.8	0.2	0.0	32
0.7	0.3	0.0	38

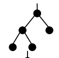
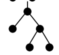
- Here crossover is the best operator.
- Grow mutation interferes with the process of finding a target string much more than full mutation.

Size-evolution equation for GP with subtree-swapping crossover

- The mean size of the programs at generation t is

$$\mu(t) = \sum_l N(G_l)\Phi(G_l, t)$$

E.g., population: `x`, `(+ x y)`, `(- y x)`, `(+ (+ x y) 3)`

l	G_l	$N(G_l)$	$\Phi(G_l, t)$
1		1	1/4
2		3	2/4
3		5	1/4
4		5	0
⋮	⋮	⋮	⋮

$$\mu(t) = 1 \times \frac{1}{4} + 3 \times \frac{2}{4} + 5 \times \frac{1}{4} = 3$$

- From the schema theory for GP with *symmetric* subtree-swapping crossover we have that

$$E[\mu(t+1)] = \sum_l N(G_l)p(G_l, t)$$

- We get exactly the same equation for the selection only case
- So, the mean program size evolves *as if* selection only was acting on the population.

Condition for bloat

- Growth can happen only if

$$E[\mu(t+1)-\mu(t)] = \sum_l N(G_l)(p(G_l,t)-\Phi(G_l,t)) > 0$$

- So, there must be some short G_l's for which $p(G_l,t) < \Phi(G_l,t)$ and also some longer G_l's for which $p(G_l,t) > \Phi(G_l,t)$ (at least on average).

- Let us define:

$$\begin{aligned} G_{\text{small}} &= \{G_l : N(G_l) < \mu(t)\} = \{\text{shapes smaller than average}\} \\ G_{\text{large}} &= \{G_l : N(G_l) > \mu(t)\} = \{\text{shapes larger than average}\} \end{aligned}$$

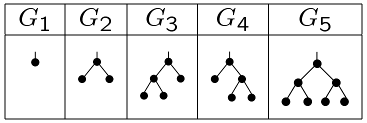
- Then

$$\begin{aligned} E[\mu(t+1)-\mu(t)] &= \sum_{G_l \in G_{\text{small}}} (N(G_l)-\mu(t))p(G_l,t) + \\ & \sum_{G_l \in G_{\text{large}}} (N(G_l)-\mu(t))p(G_l,t). \end{aligned}$$

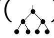
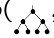
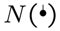
- Thus, the mean program size will be expected to grow if

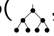
$$\sum_{G_l \in G_{\text{large}}} (N(G_l)-\mu(t))p(G_l,t) > \sum_{G_l \in G_{\text{small}}} (\mu(t)-N(G_l))p(G_l,t).$$

- Example:
  - $\mu(t) = 5$
  - Program shapes in population:



  - Then, growth is expected to happen if

$(N($$)-5)p($$,t) > (5-N($$))p($$,t) + (5-N($$))p($$,t)$

i.e. $2p($$,t) > 2p($$,t) + 4p($$,t)$

Lesson for bloat control

To prevent bloat, one must either decrease the selection probability for bigger than average programs or increase it for below-average-size ones, or both (for symmetric crossovers).

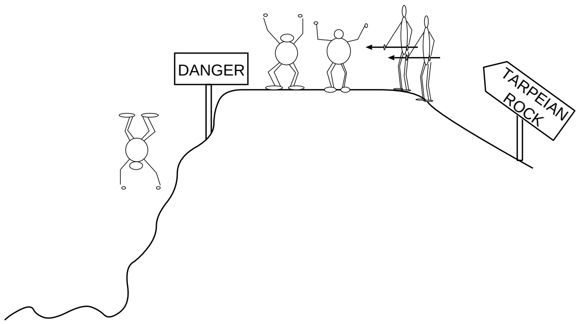
The Tarpeian method to control bloat

- Tarpeian fitness-wrapper

```
IF size(program) > average_pop_size AND random_int MOD n = 0
THEN
     return( very_low_fitness );
ELSE
     return( fitness(program) );
```

- The Tarpeian method drastically decreases the selection probability of longer-than-average programs creating a sort of *fitness hole* that discourages growth
- Note: it does it only for *some long random individuals* and for *some of the time*

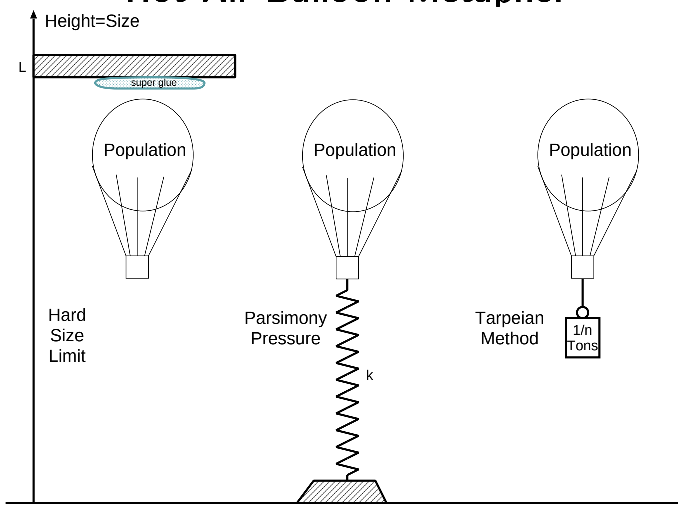
Why "Tarpeian"?



Features of the Tarpeian method

- The wrapper *does not* evaluate the individuals killed, so computation is saved not wasted.
- If enough better-than-average longer-than-average individuals are produced, the average program size grows, i.e. the algorithm allows growth as long as it is associated to fitness improvements.
- If the average program size increases, the fitness hole moves so as to discourage further growth.
- The parameter n determines the intensity of the repulsive force exerted by the fitness hole.
- The pressure to shrink does not change with program size.

Hot Air Balloon Metaphor



Lessons learnt from "Tarpeian" experiments

- Some care is needed in choosing the population size, the selective pressure and the intensity of the repulsive force exerted by the Tarpeian algorithm
- When the parameters are right the Tarpeian algorithm improves:
  - best fitness achieved
  - execution time
  - parsimony of the evolved solutions

Conclusions on the GP schema theory

- After years of stagnation and criticism, in the last six years schema theories have seen a formidable development.
- Today schema theories explain and predict the behaviour of GAs and GP.
- Thanks to the GP schema theories, we know how and why GP keeps, adds or removes individuals in/to/from a schema (our original goal).
- GP building blocks are now well understood.
- Different operators lead to different schema theorems. We have started integrating them into a single coherent theory, but the work is not finished.

- Schema-theory results have started clarifying the behaviour of the operators and the influence of their parameters on the search, and, more recently, they have also started helping design competent algorithms (like the Tarpeian method).
- In the future, the availability of exact schema theorems will allow formal performance comparisons between different types of operators and different problems.
- GP theory is more general than the corresponding GA theory (see schema theory and Markov chains for GP with homologous crossover).
- Lots of new results and practical applications of the theory are just around the corner waiting for *us* to discover them. Please, collaborate with us to extend and apply the theory.