Genetic Algorithms

mimic natural selection — nature-inspired — Ant Colony Optimization, Particle Swarm Opt

Individuals have genes
genes determine phenotype
phenotype contributes to fitness
fitness contributes to propagation

start with random population — collection of individuals w/ random genes (most basically, sequence of bits) — of one (or just a few)

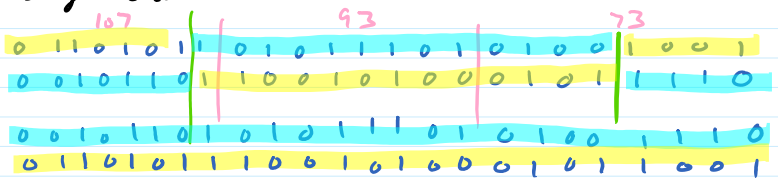
while not done (out of time, no improvement, good enough ...)

- evaluate each individual — genes → inputs to f → evaluate f → fitness
- select for crossover — select pairs of individuals, biased towards higher fitness
- crossover — randomly select genes from parents to produce offspring
- select for survival — replacement — fitness-based
- mutate — randomly change genes in some individuals (generally low rate)

evolution strategy: no crossover; produce several mutated offshoots

Representation: what is genetic code?

$f(x,y,z)$



Crossover:

bitstrings — ex: two-point crossover

0 1 1 1 0 1 0
1 1 0 1 1 0 0

permutation

C F B A E D
D C F A B E

C F F A B D
D C B A E E

not a valid input! also not a permutation! infeasible

offspring not enough like parents

3 5 2 1 2 1
4 3 4 1 1 1

C F E A B D ← 3 5 4 1 1 1
D C B A F E ← 4 3 2 1 2 1

solutions: 1) assign low fitness to infeasible solns
2) design genome to avoid infeasible offspring

C C F F C B B F A C C E A D
D C F C F B C B B E A D C F

C C F F F B C B B E A E A D

↳ phenotype: C F B E A D

	A	B	C	D	E	F
C F B A D E ←	50	70	100	20	10	80
D C F A B E ←	32	30	50	75	26	40
F D B C A E	50	70	50	75	26	80

0, 10, 10, 0, 5
0, 0, 0, 15, 10

0, 10, 10, 15, 10

Genetic Programming: genetic algorithms for programs

```
if seeds[0] > 5
{
    many++
}
```

```
if seeds[i] == 0 and seeds[i - 1] == 1
{
    return i-1
}
```

represent programs as syntax trees

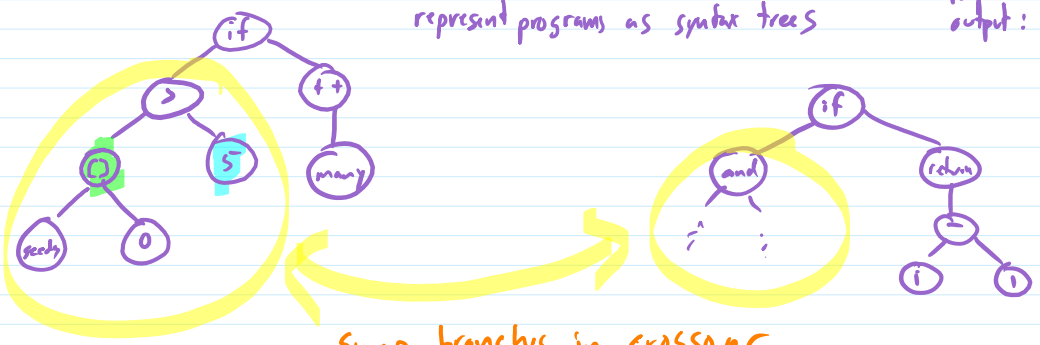
fitness function
input: heuristic function
output: performance when

```
        many++
    }
```

```
    return i-1
}
```

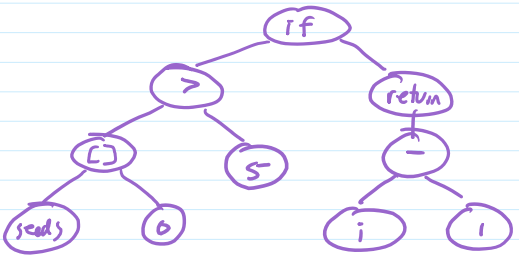
fitness function
input: heuristic function
output: performance when playing w/ that heuristic

represent programs as syntax trees



swap branches in crossover

```
if seeds[0] > 5
{
 return i - 1;
}
```



0. David et al

Chess Heuristic: for each player, count

9	for queen
5	for rook
3	for bishop
3	for knight
1	for pawn
+1	for rook mobility
+1	for king protection
+3	for breakaway pawn
	⋮

Find parameters to maximize Elo → numeric measure of strength of player
adjust rating after each game
larger adjustments for more surprising result

Problems for GA: what is fitness?

performance against existing agents
or population of agents
how well moves chosen match experts (or other good agents)

coevolution: play individuals against each other
fitness function is # wins in round-robin tournament

mentor: how often individual's choice matches mentor's

GAs in Noisy Environments

→ noise in fitness fcn

f(heuristic constants) = expected turns to complete game in Rule of 28

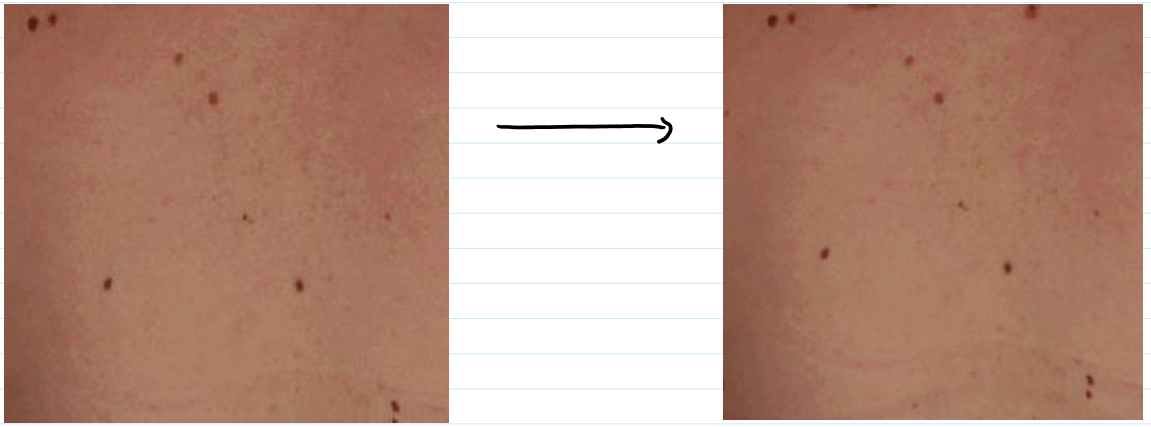
difficult to compute so estimate

play 10000 times

if σ for one game = 1

σ for estimate over 10000 games = $\frac{1}{\sqrt{10000}} = \frac{1}{100}$

Medical Image Registration



Optimization: Find parameters to maximize # matching pixels
↳ rotation angles, zoom, skew, translation, ...

Fitness function:

in general, GAs can deal with noisy fitness fcns
- don't need more samples per individual
randomness leads to exploration