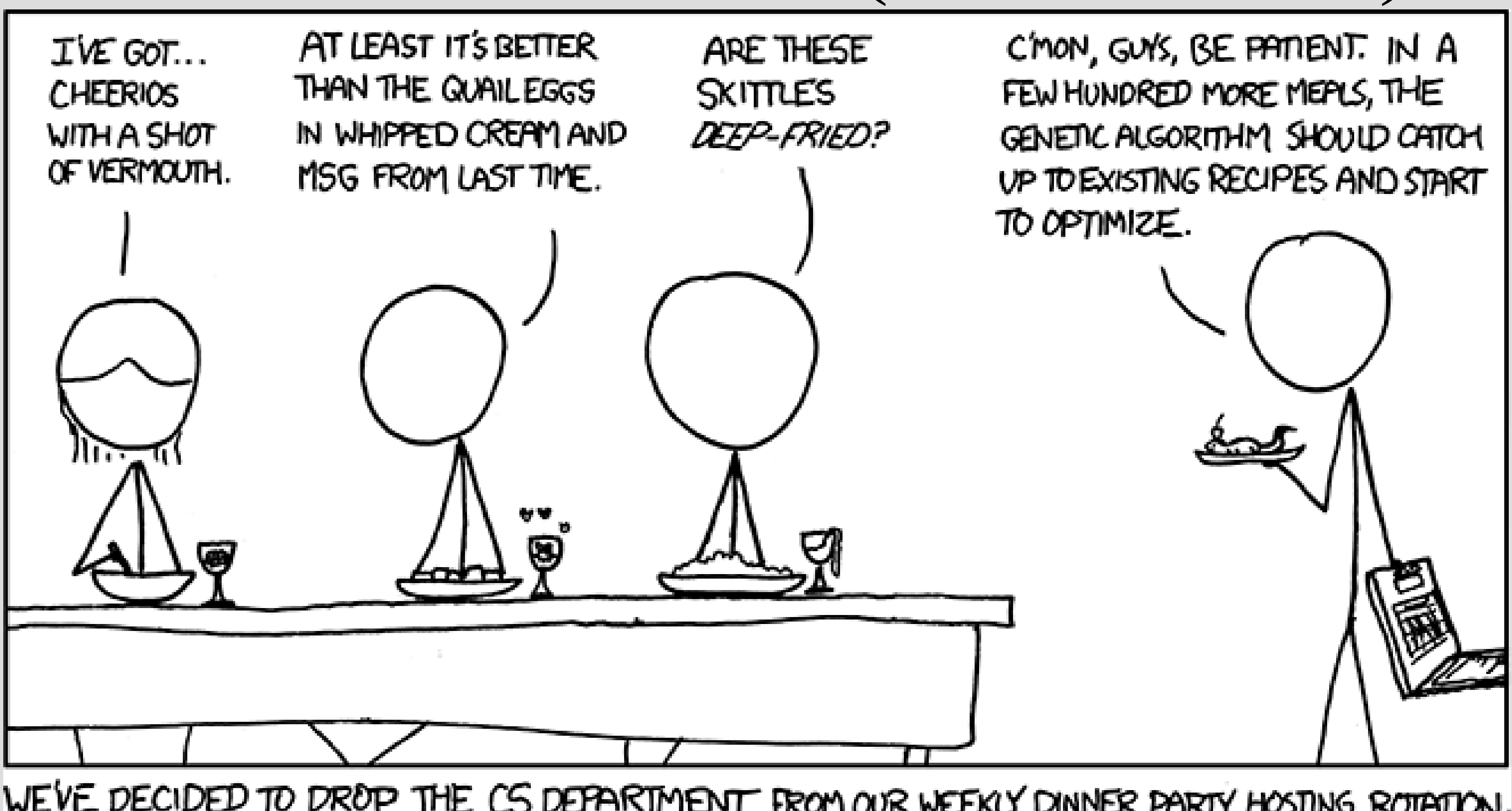
Local Search (Ch. 4-4.1)



Local search

We will discuss four optimization algorithms:

1. Hill climbing ✓
2. Simulated annealing ✓
3. Beam search
4. Genetic algorithms

All of these will only consider neighbors while looking for a goal

Local beam search

Beam search is similar to hill climbing, except we track multiple states simultaneously

Initialize: start with K random nodes

1. Find all children of the K nodes
2. Add children and K nodes to pool, pick best
3. Repeat...

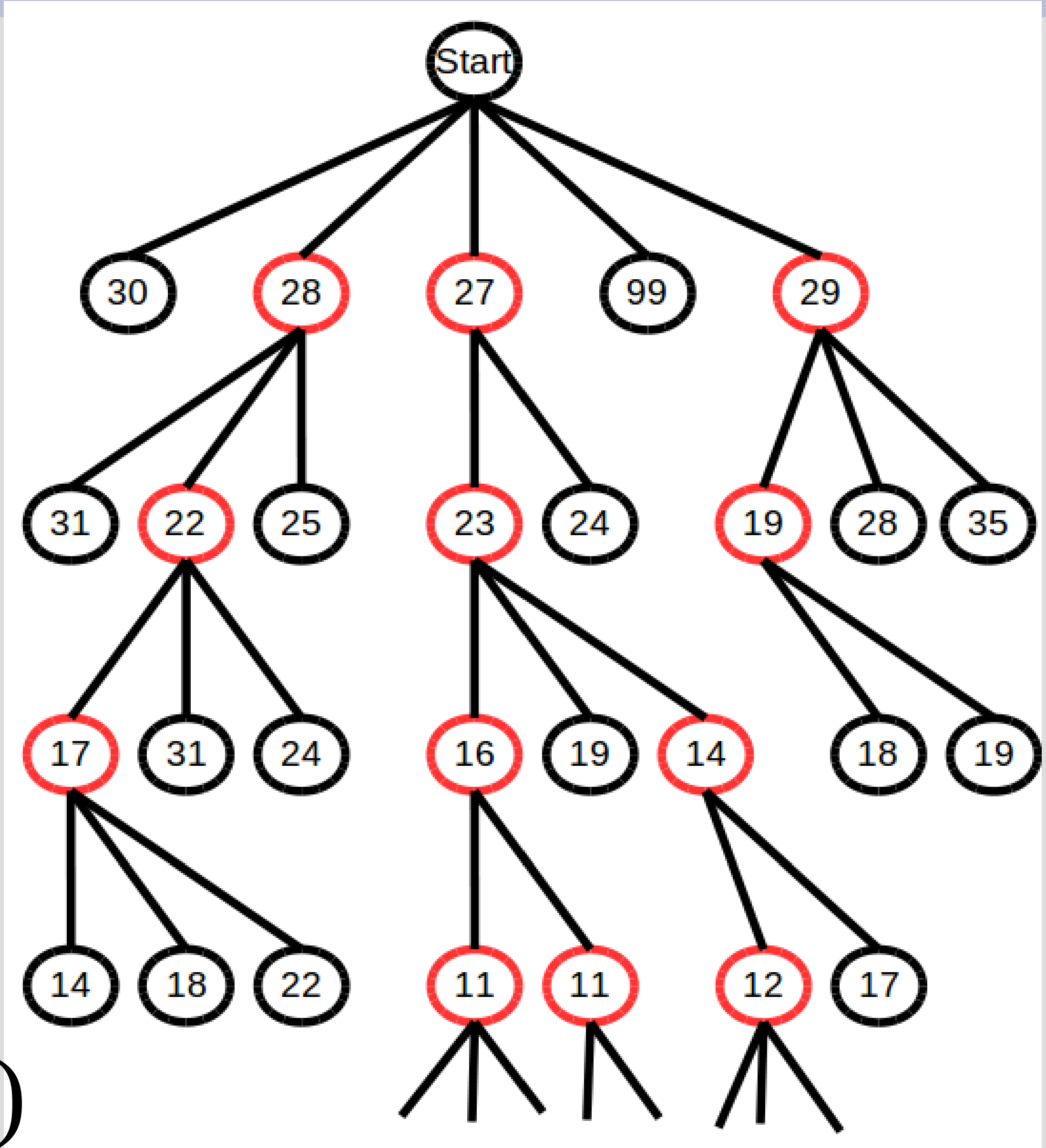
Unlike previous approaches, this uses more memory to better search “hopeful” options

Local beam search

Beam search with 3 beams

Pick best 3 options at each stage to expand

Stop like hill-climb (next pick is same or worse as last pick)



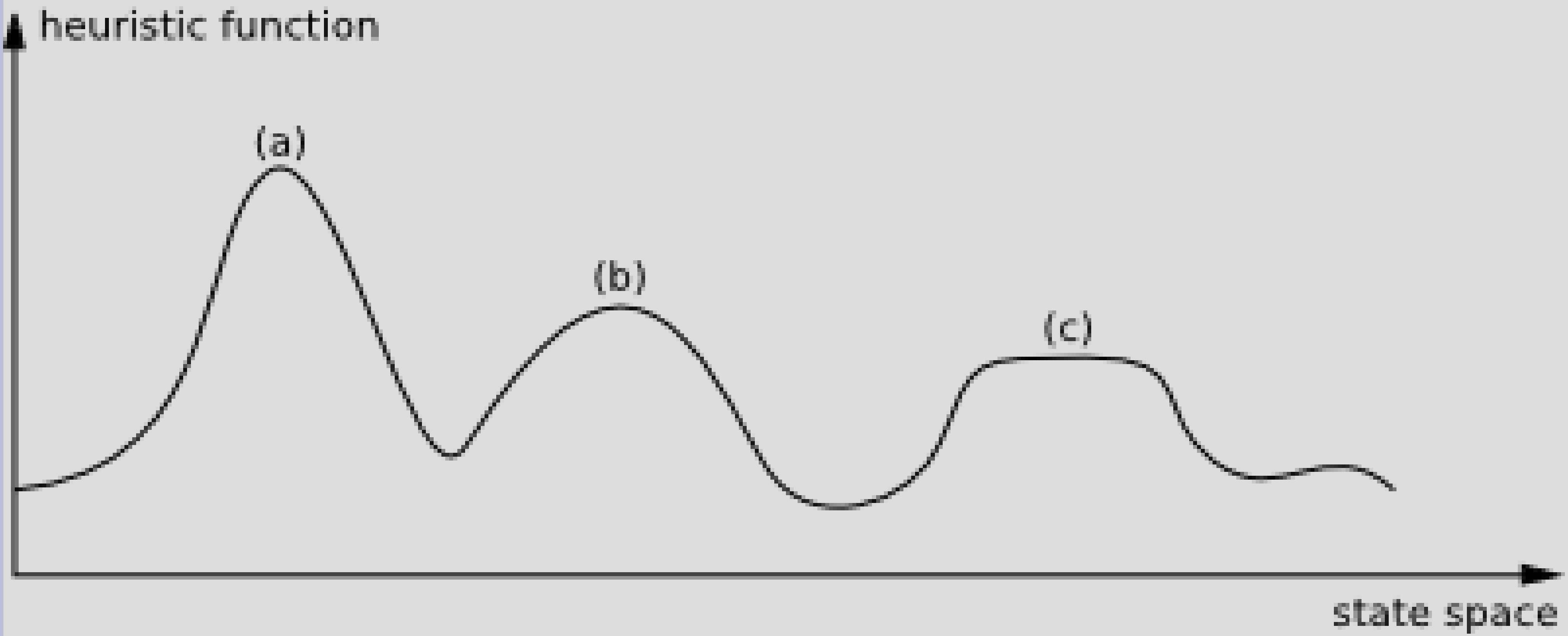
Local beam search

However, the basic version of beam search can get stuck in local maximum as well

To help avoid this, stochastic beam search picks children with probability relative to their values

This is different that hill climbing with K restarts as better options get more consideration than worse ones

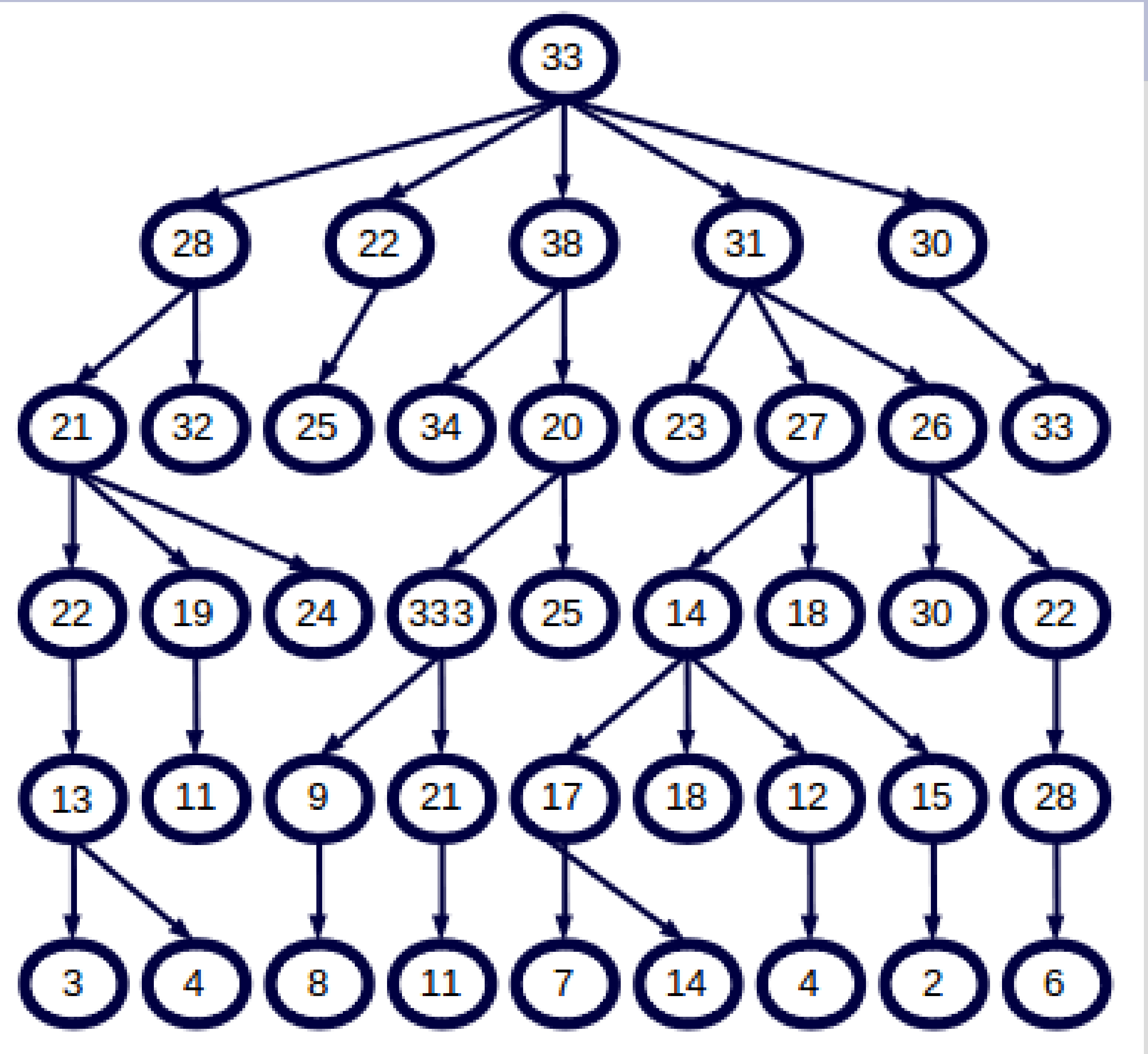
Local beam search



Local beam search

You try it!

Run local-beam search with k=4 on this tree



Genetic algorithms

Genetic algorithms are based on how life has evolved over time

They (in general) have 3 (or 5) parts:

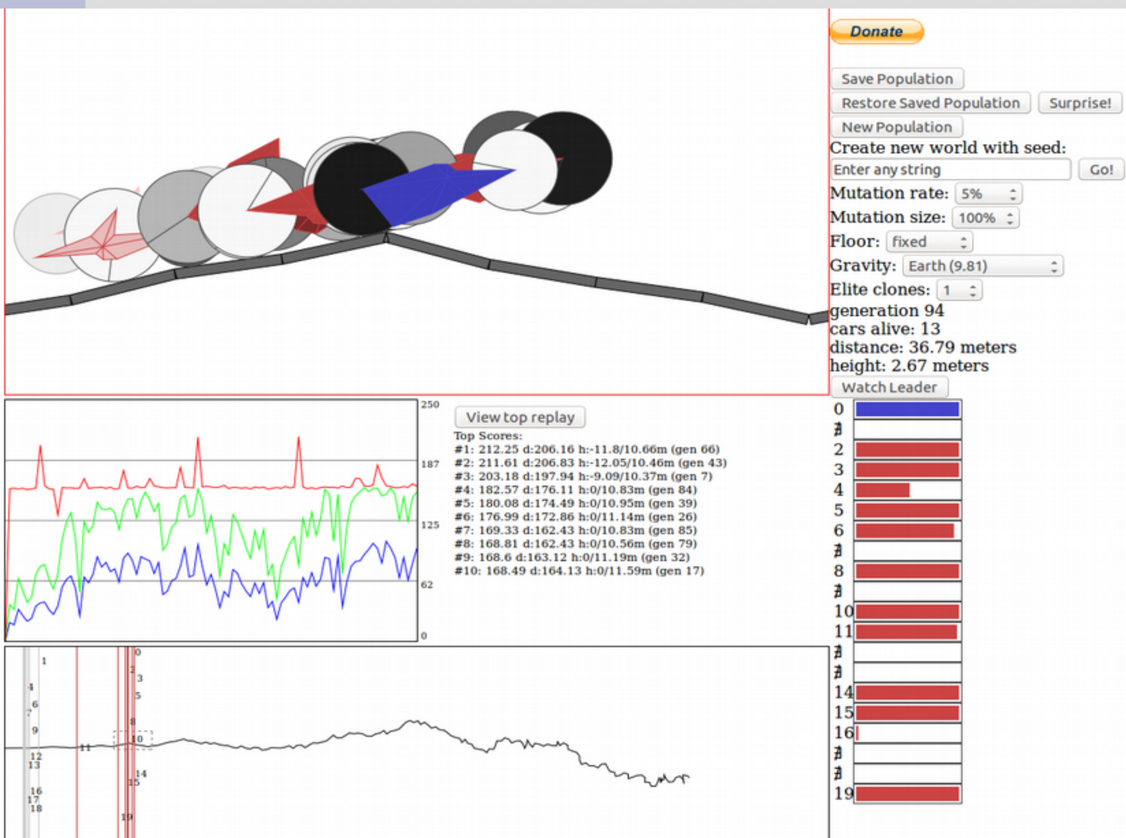
1. Select/generate children
   1a. Select 2 random parents
   1b. Mutate/crossover
2. Test fitness of children to see if they survive
3. Repeat until convergence

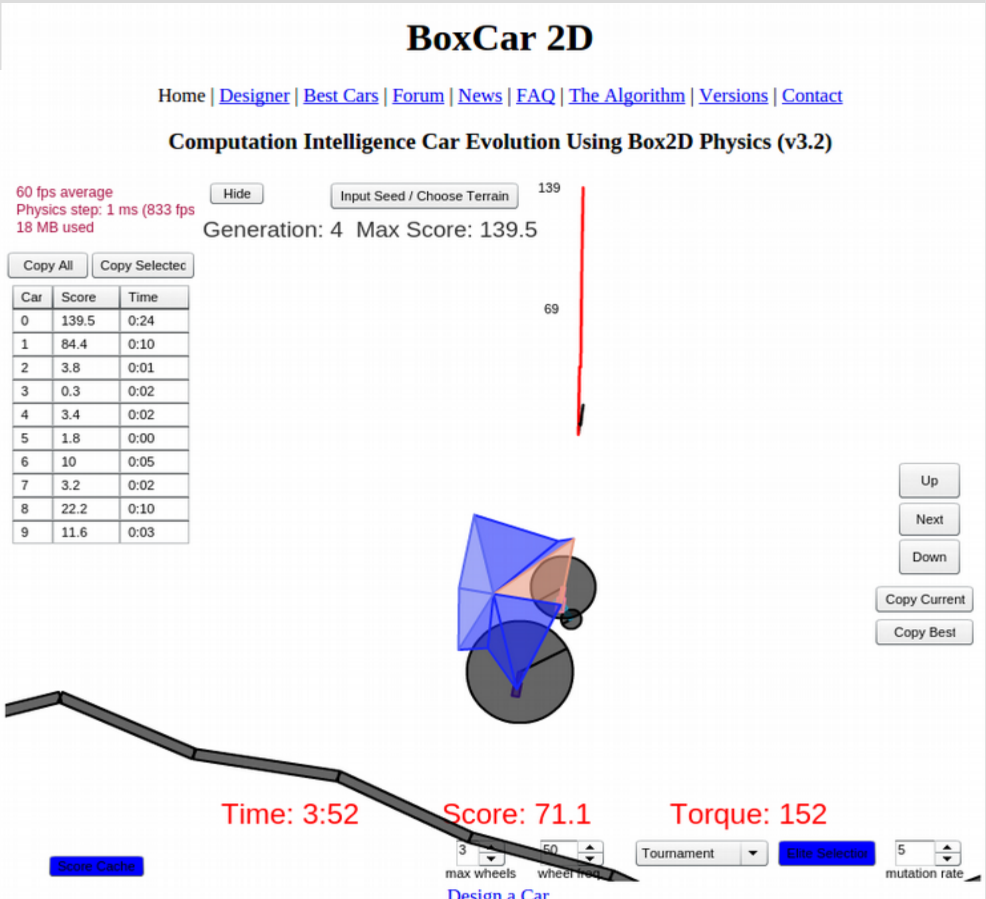
Genetic algorithms

Nice examples of GAs:

http://rednuht.org/genetic_cars_2/

http://boxcar2d.com/

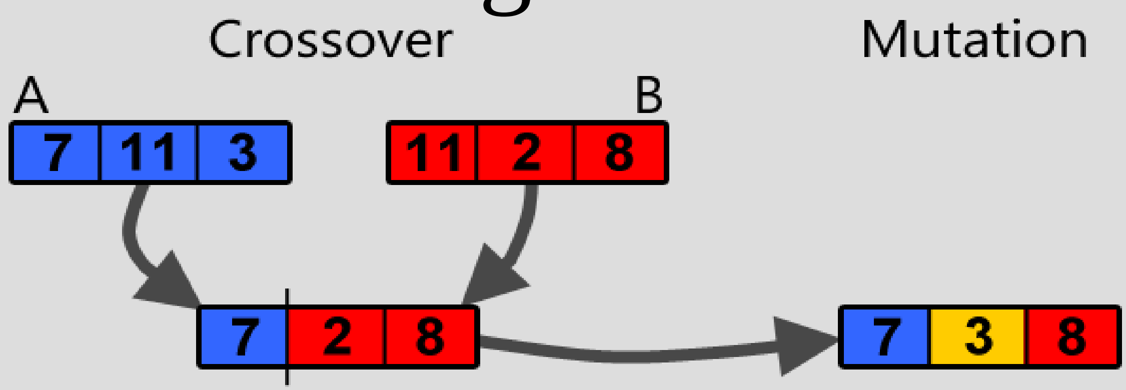




Genetic algorithms

Selection/survival:

Typically children have a probabilistic survival rate (randomness ensures genetic diversity)



Crossover:

Split the parent's information into two parts, then take part 1 from parent A and 2 from B

Mutation:

Change a random part to a random value

Genetic algorithms

Genetic algorithms are very good at optimizing the fitness evaluation function (assuming fitness fairly continuous)

While you have to choose parameters (i.e. mutation frequency, how often to take a gene, etc.), GAs tend to head to optimal

The downside is that often it takes many generations to converge to the optimal

Genetic algorithms

There are a wide range of options for selecting who to bring to the next generation:

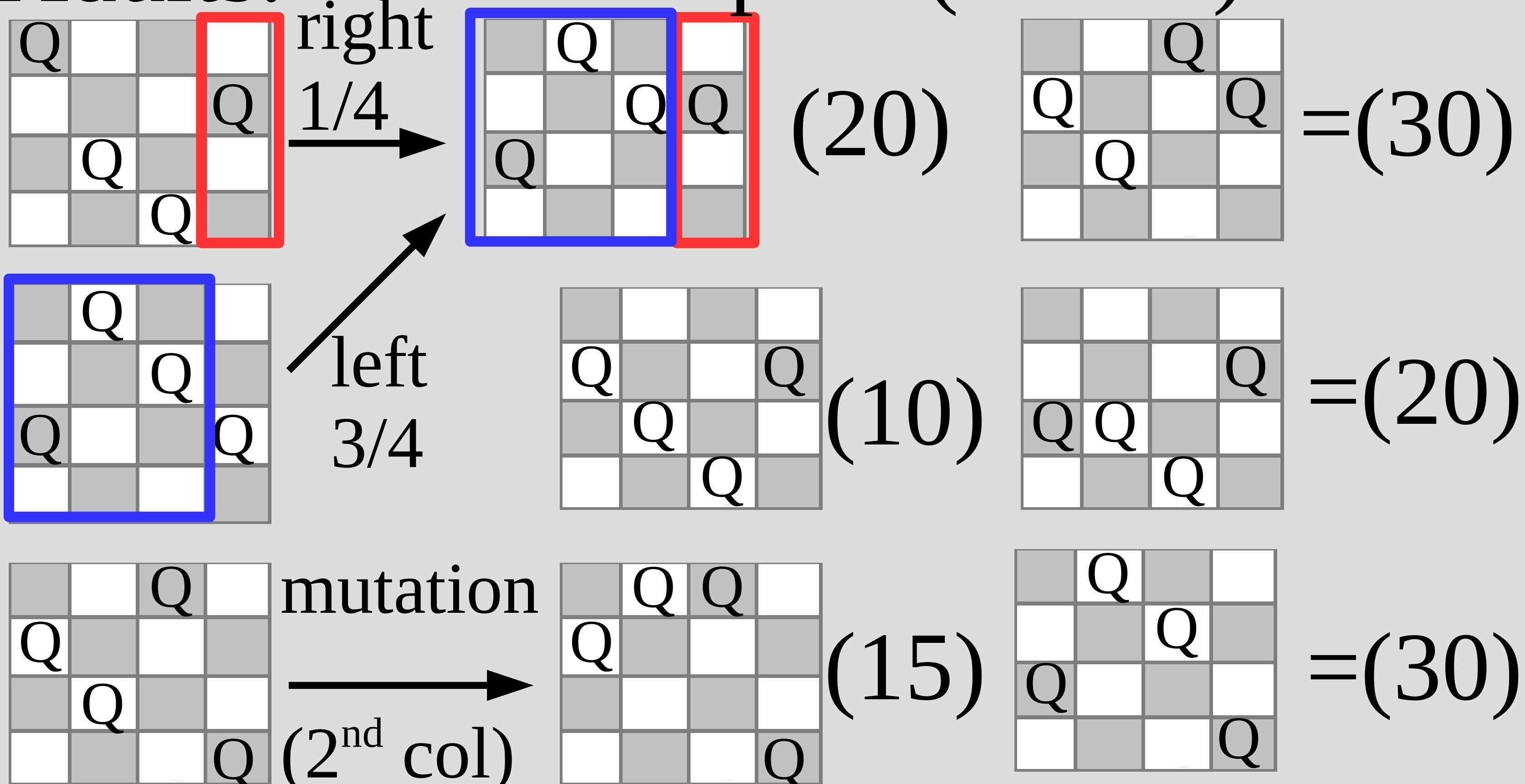
- always the top (similar to hill-climbing... gets stuck a lot)
- choose purely by weighted random (i.e. 4 fitness chosen twice as much as 2 fitness)
- choose the best and others weighted random

Can get stuck if pool's diversity becomes too little (hope for many random mutations)

Genetic algorithms

Let's make a small (fake) example with the 4-queens problem

Adults: Child pool (fitness):

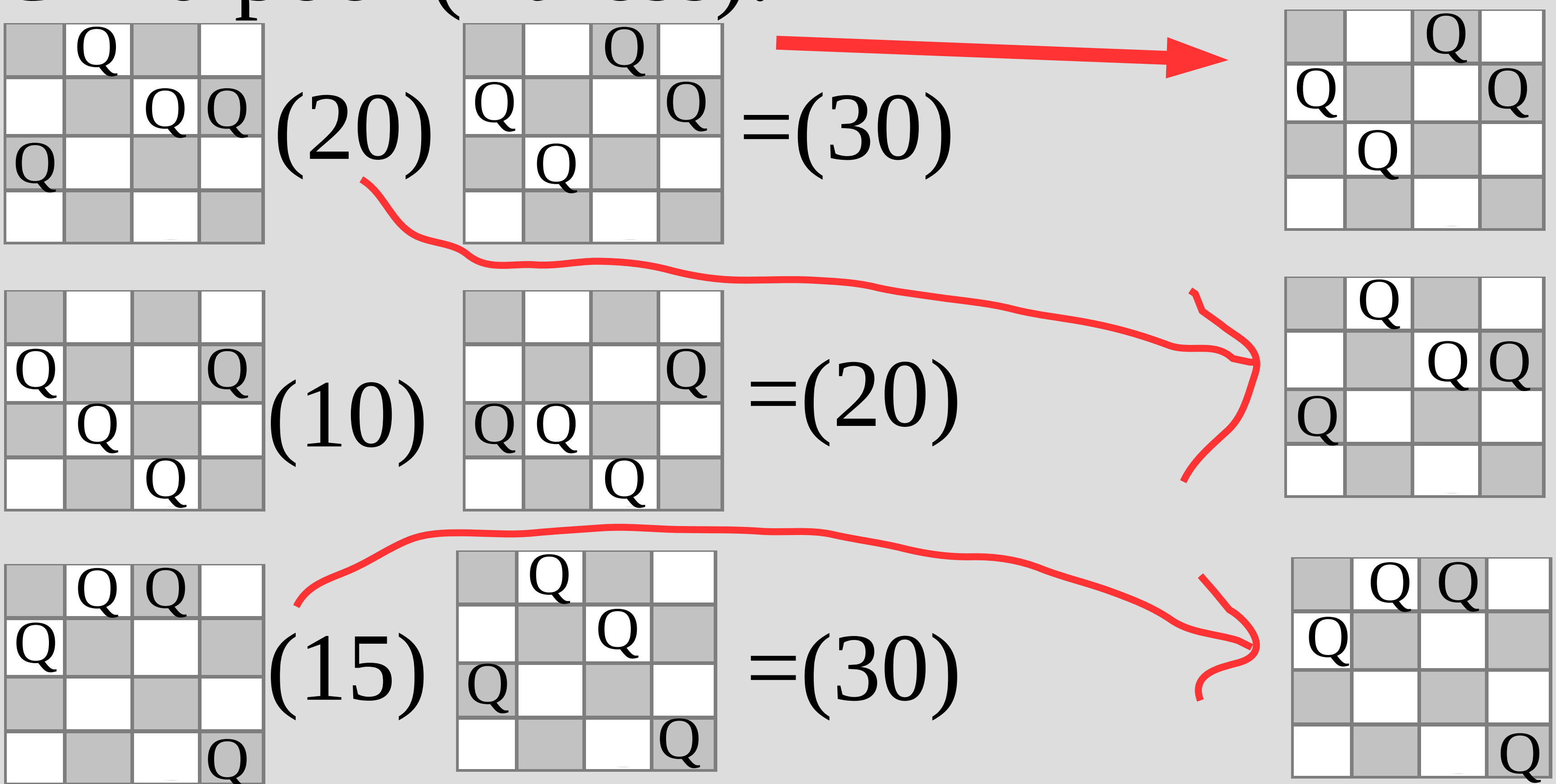


Genetic algorithms

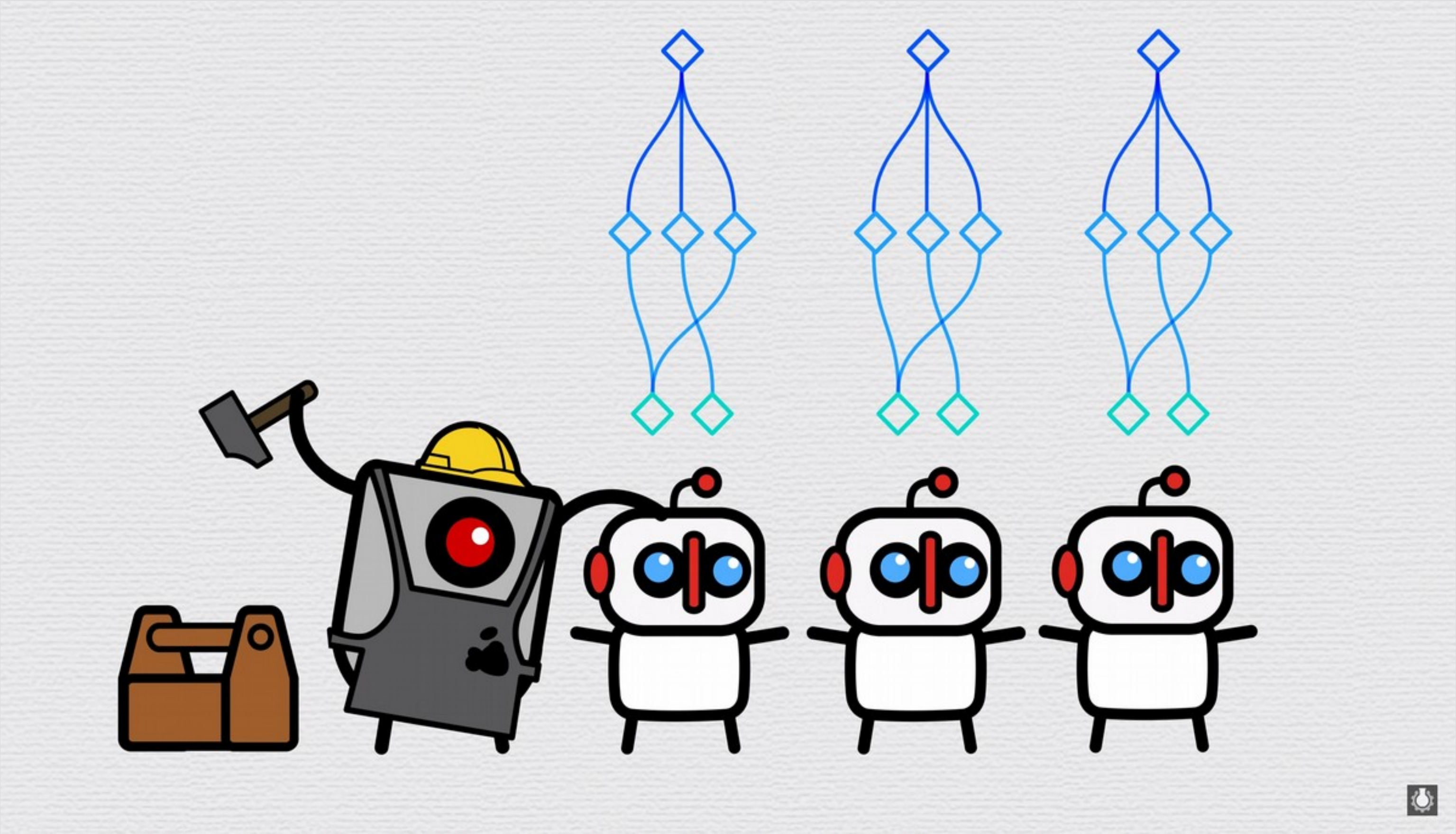
Let's make a small (fake) example with the 4-queens problem

Child pool (fitness):

Weighted random selection:



Genetic algorithms



https://www.youtube.com/watch?v=R9OHn5ZF4Uo