Special Topic: Genetic Algorithms in the Real World

CSCI 315: Artificial Intelligence

Simon D. Levy

Fall 2007

Outline

- Genetic Algorithms Intro
- Multi-Objective Optimization
- Elitism, Crowding, and the NSGA-II algorithm
    - Exercise: advantages of NSGA-II
- Neural networks intro
- Evolving neural networks
- The NEAT algorithm
    - Exercise: The NERO video game

# GA Intro:
AI = Search + Learning

- Much of what we do in AI involves searching for solutions
  - Tree search (Eight-Queens, Maze)
  - Logic-driven search (Wumpus World)
  - Searching for a parse (NLP/Prolog)
- We've also seen a bit of *machine learning*, where exposure to data helps us do better in the future
  - Decision-Tree Learning
- In the domains we've seen, there are very few acceptable solutions (all-or-nothing)

The Real World Isn't (Always) All-Or-Nothing

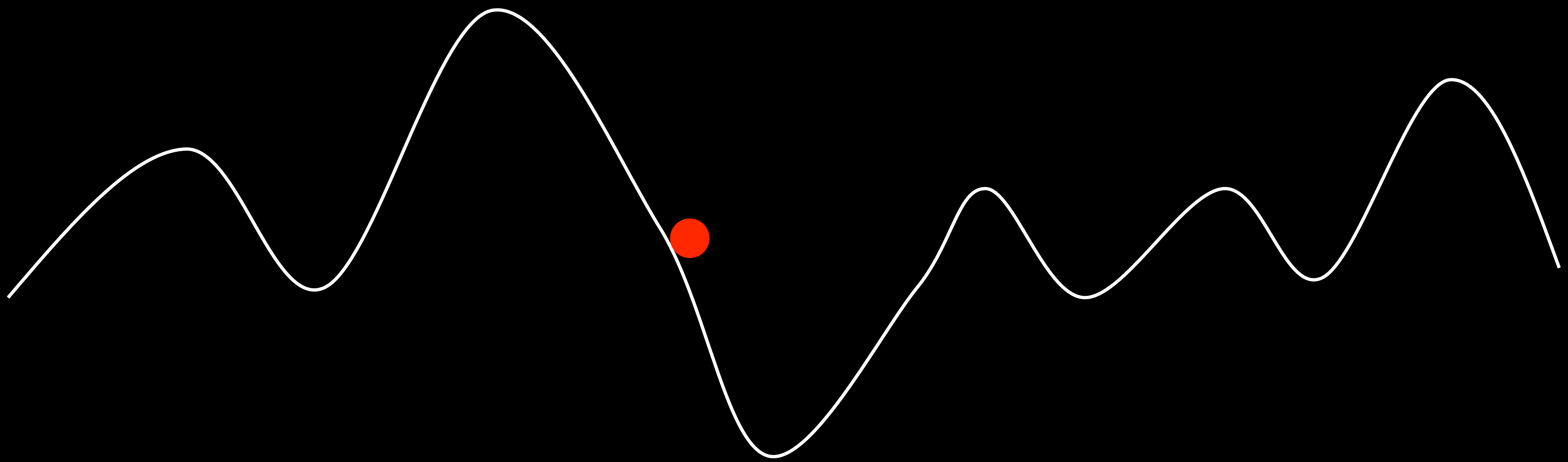
- Lots of real-world choices (profession, location, life partner) involve gradient values (from worst to best)
- Often there are many "locally optimal" solutions - not the absolute best, but good enough
- Then the search for a local optimum looks a lot like hiking a range hills....

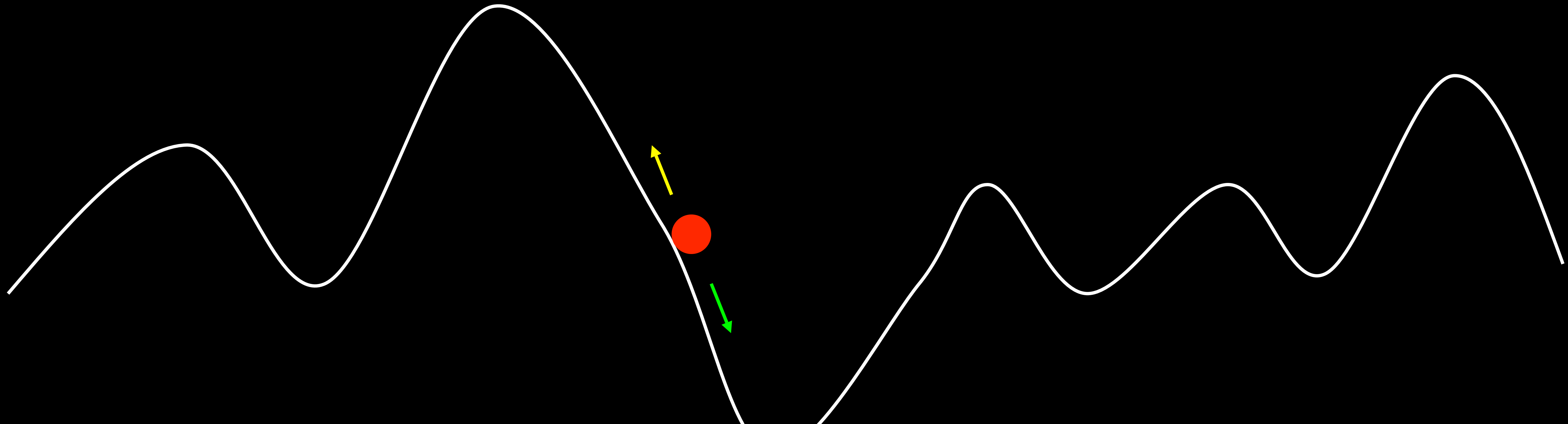
Search as Hill-Climbing

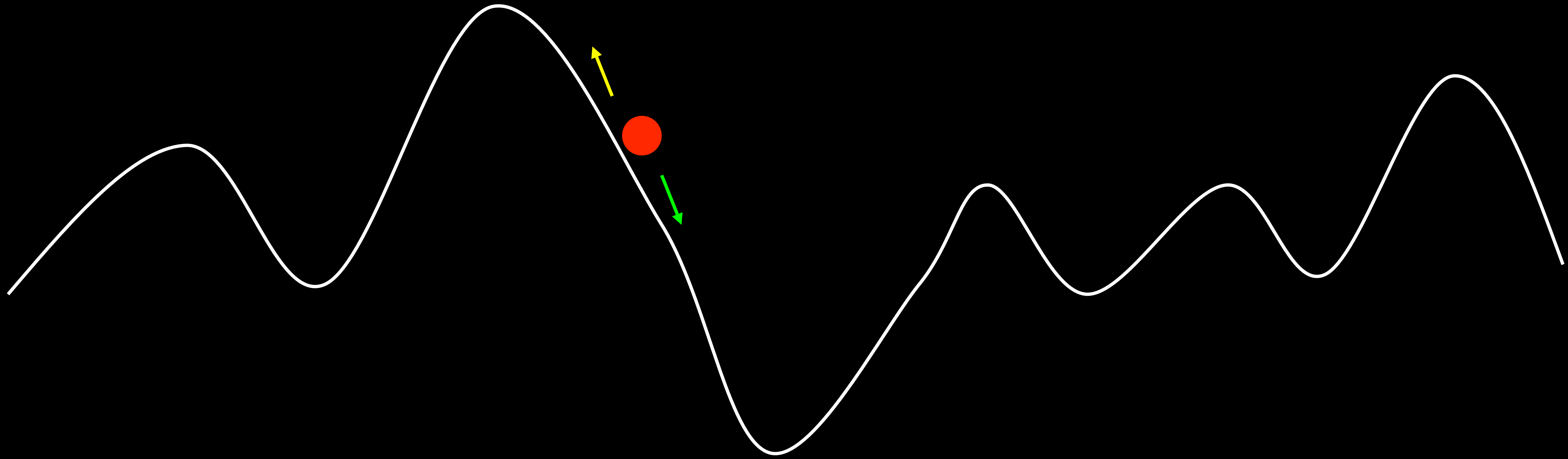
1. Start at some random position

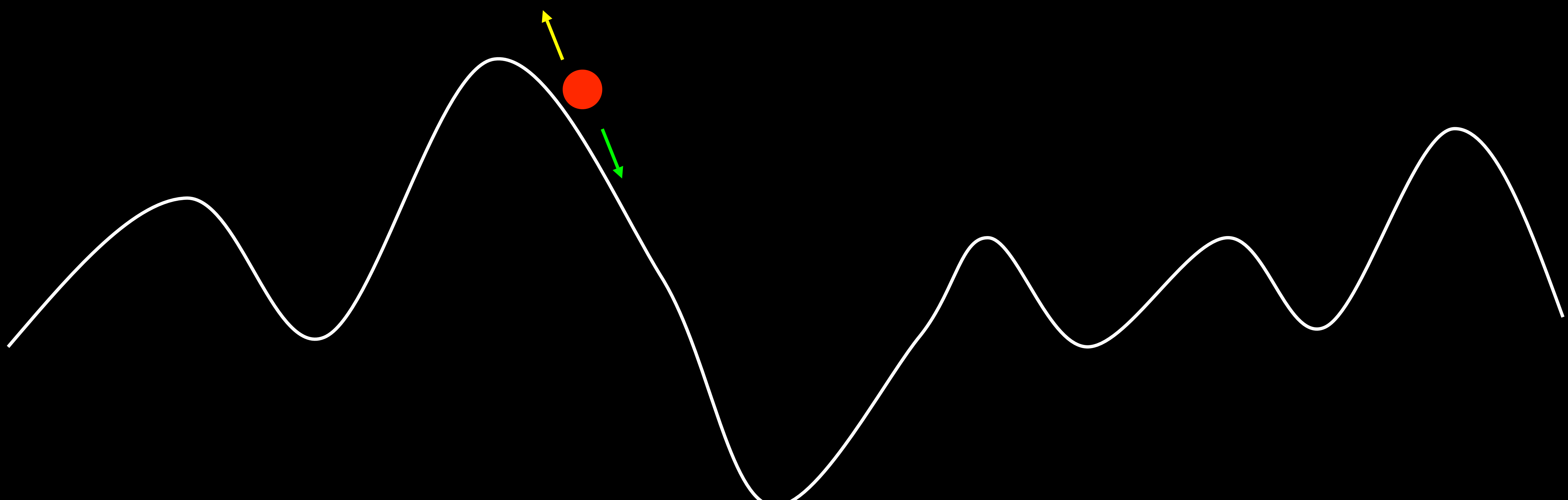
2. Explore your neighborhood to see which direction takes you higher – gives you a *fitter* (better) solution

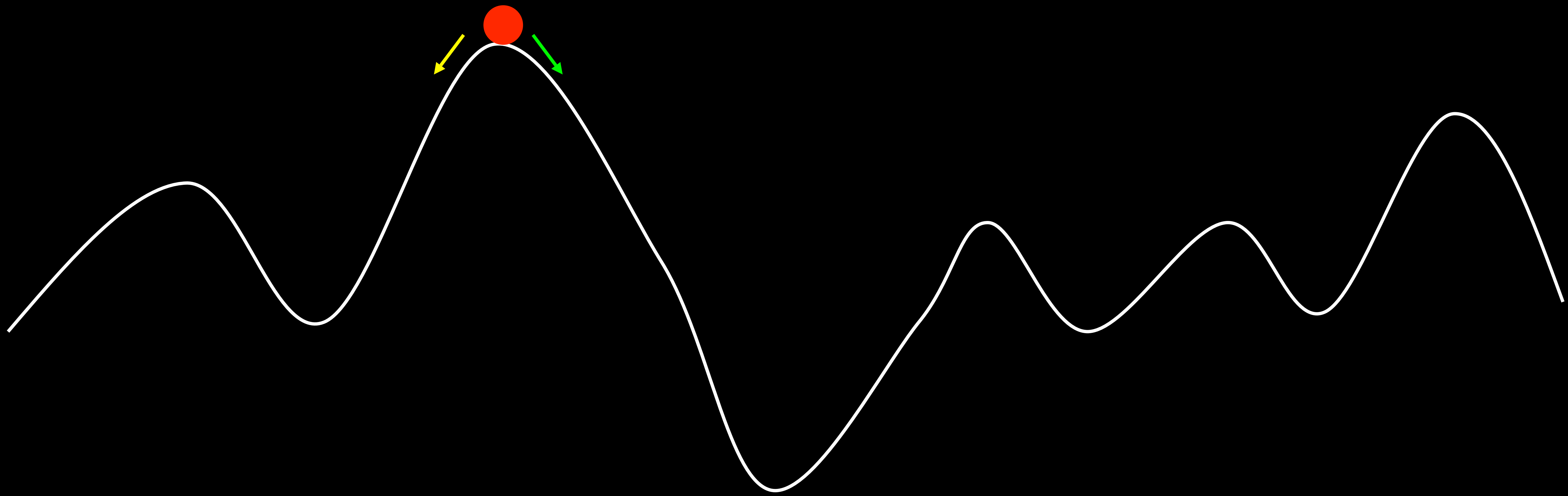
3. As long as your fitness is increasing, keep exploring; otherwise, stop.



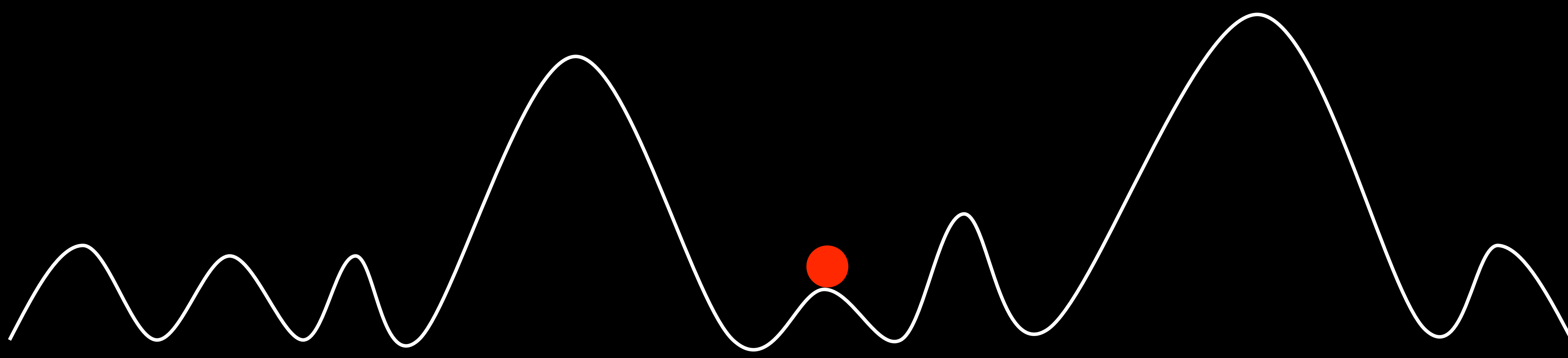




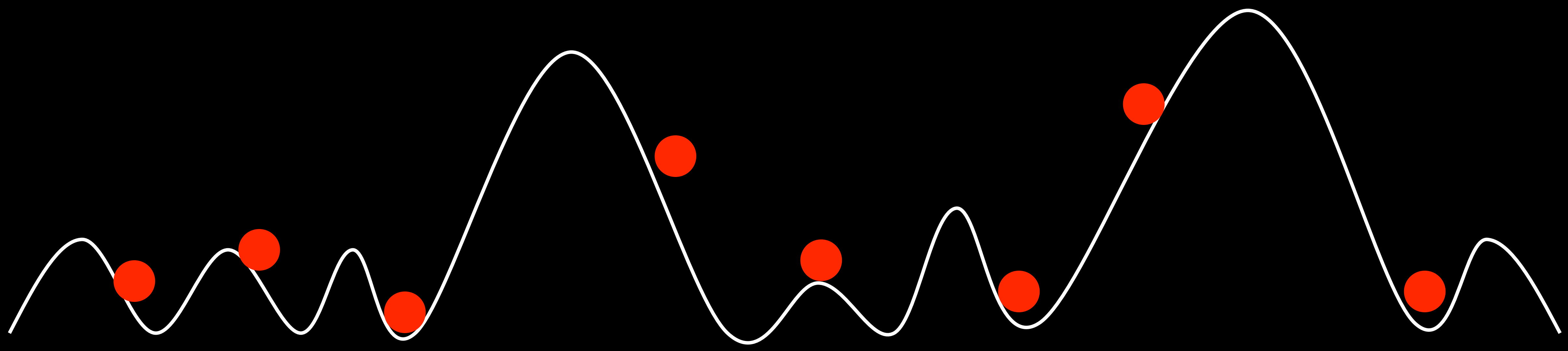




The Problem with Hill-Climbing

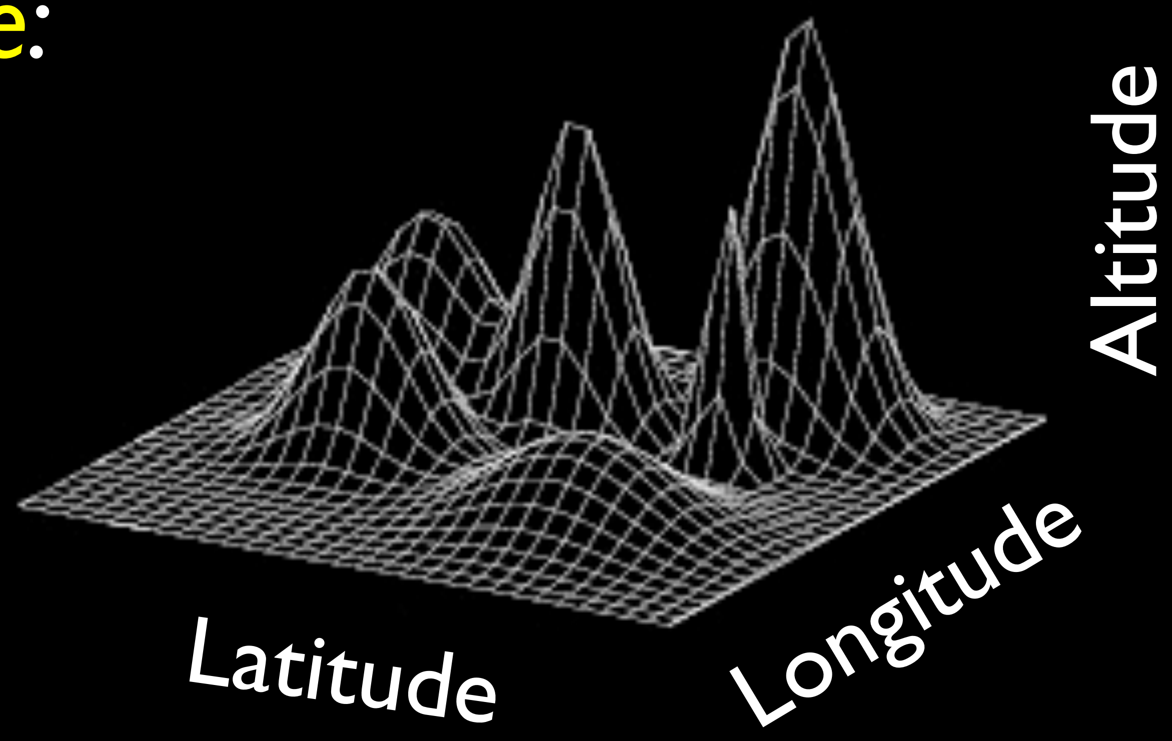


The Genetic Algorithm Insight: A *Population* of Candidate Solutions



Issue #1: Choice Isn't One-Dimensional

Fitness Landscape:



Issue #2: Fitness Isn't One-Dimensional

Multi-Objective *Fitness:*

Issue #3: Exploring the Landscape

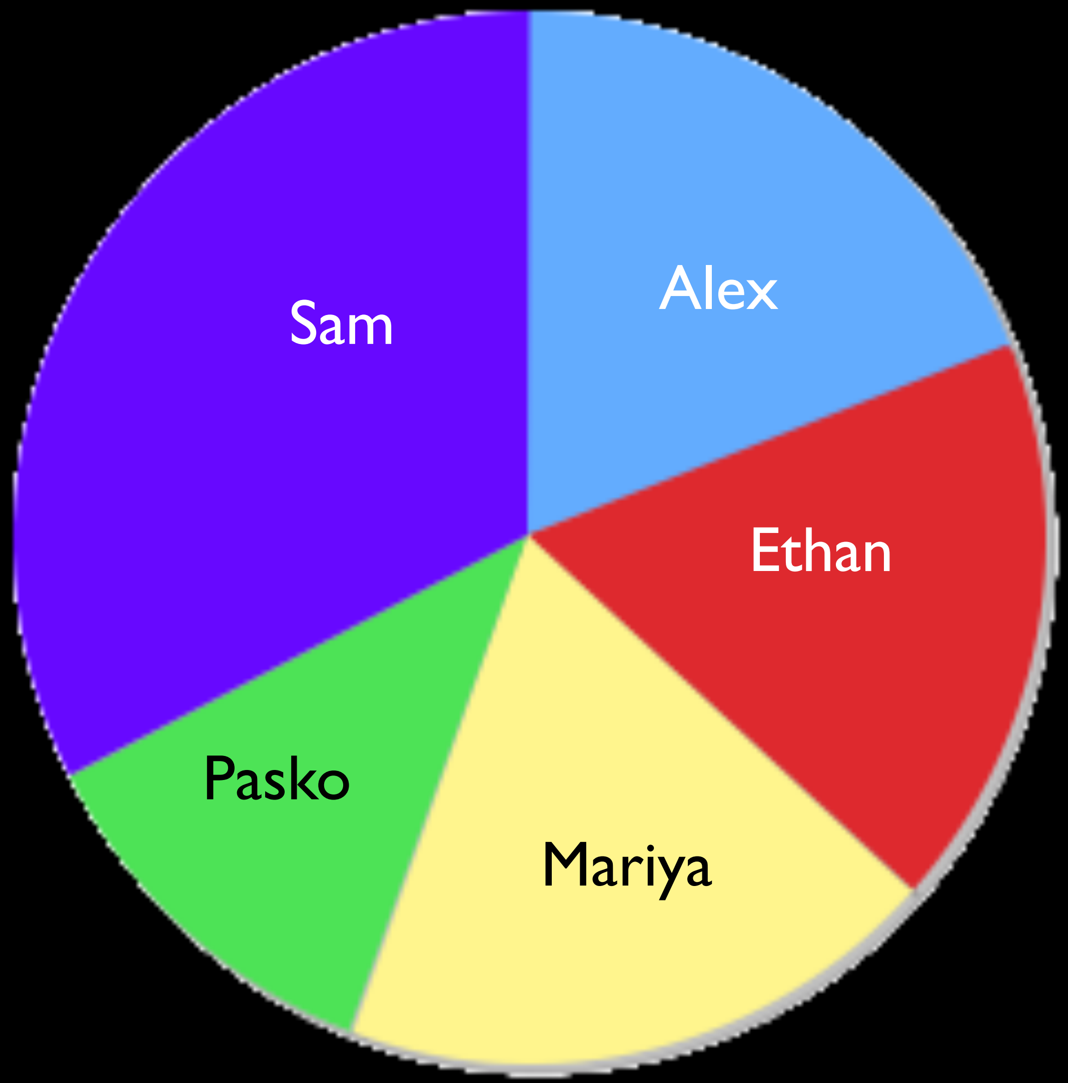
- Introduce a little variation into each member of the population
- "Explore a little bit in each direction" a.k.a. mutation
- Combine components of existing solutions to (we hope) get a better one: recombination a.k.a. crossover a.k.a. sex
- A good representation of our problem will allow us to exploit these operations; a bad one will make that very difficult (as in all AI).

Issue #4: Survival of the Fittest

- A balance between preserving only the fittest individuals (overbreeding) vs. preserving variation.
- Elitism: keep only the *N* fittest
- Fitness-proportionate selection (FPS): your *chance* of "surviving" till the next generation is proportionate to your fitness - a biased roulette wheel

Wheel! Of! Facebook!

Person	# Friends
Alex	184
Ethan	169
Mariya	181
Pasko	114
Sam	311
Simon	1



FPS Algorithm

Fitnesses:	[184,	169,	181,	114,	311]
	A	E	M	P	S
Normalize:	[0.192	0.176	0.189	0.119	0.324]
Accumulate:	[0.192	0.368	0.557	0.676	1.000]

Now roll wheel by picking a random # between 0 and 1 and returning first element whose cumulative value is greater than the number.

The Simple Genetic Algorithm (SGA)

Generate initial random population

Repeat until satisfied:

    Compute population fitnesses

    New population = { }

    While new population size < old population size:

        Pick two parent individuals from old pop. using FPS

        Cross parents to produce two offspring

        Mutate offspring

        Put offspring into new population

    Old population = new population

SGA Details: Mutation

- As in biology, most mutations are deleterious (harmful)
- So, we typically keep our mutation rate very low – e..g., each bit has a .01 probability of flipping (0 to 1; 1 to 0)

SGA Details: Crossover

If individuals are represented as strings of bits ("DNA"), we can cross them over at some randomly-chosen point, producing two offspring (children):

A = 0 1 1 0 1 1 0 1

B = 1 1 1 1 0 0 0 0

^

C1 = 0 1 1 1 0 0 0 0

C2 = 1 1 1 0 1 1 0 1

SGA Details: Crossover

Schema Theorem (Holland 1975): short, lower-order sub-sequences with high fitness increases in population over time

0 * * 1 0 1 * 1 = { 0 0 0 1 0 1 0 0,

0 0 0 1 0 1 0 0,

0 0 1 1 0 1 1 0,

. . .

0 1 1 1 0 1 1 1 }

Length = 8

Order = 5

Controversial Building Block Hypothesis (Goldberg 1989) says that GA's work by recursively combining schemas into bigger schemas.