Genetic Programming

Emil Lund Klinting

Aarhus University, Department of Chemistry

October 30, 2017

Outline

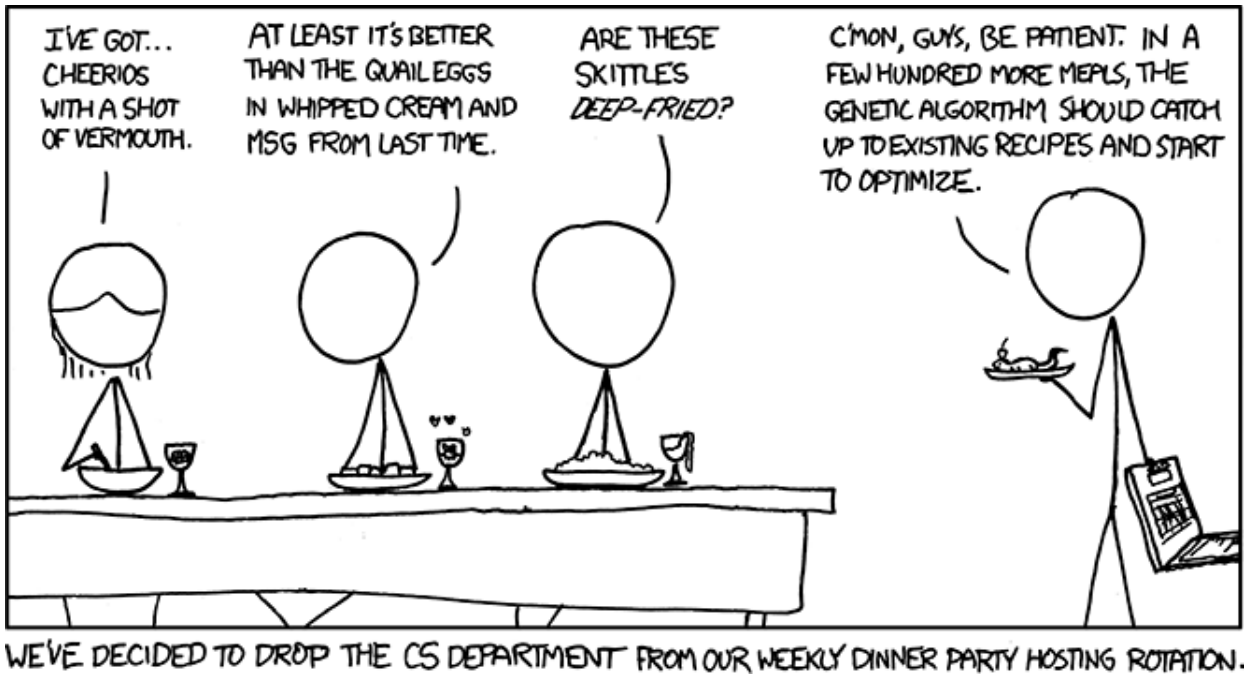
Definitions

- Genetic algorithms (GAs) belong to the large class of evolutionary algorithms (EAs), which are heavily inspired by biological process like natural selection.
- The aim of genetic algorithms is to get a computer to solve problems automatically without specifying the precise structure.
- Predominantly used for optimization problems, where a brute force method, i.e. a check for all known possibilities is not applicable.
- GAs evolve a population of individuals. The individuals comprise a list of information that can be strings, numbers, vectors or more complicated objects.
- Many different flavours, definitions and best practices.

Applications

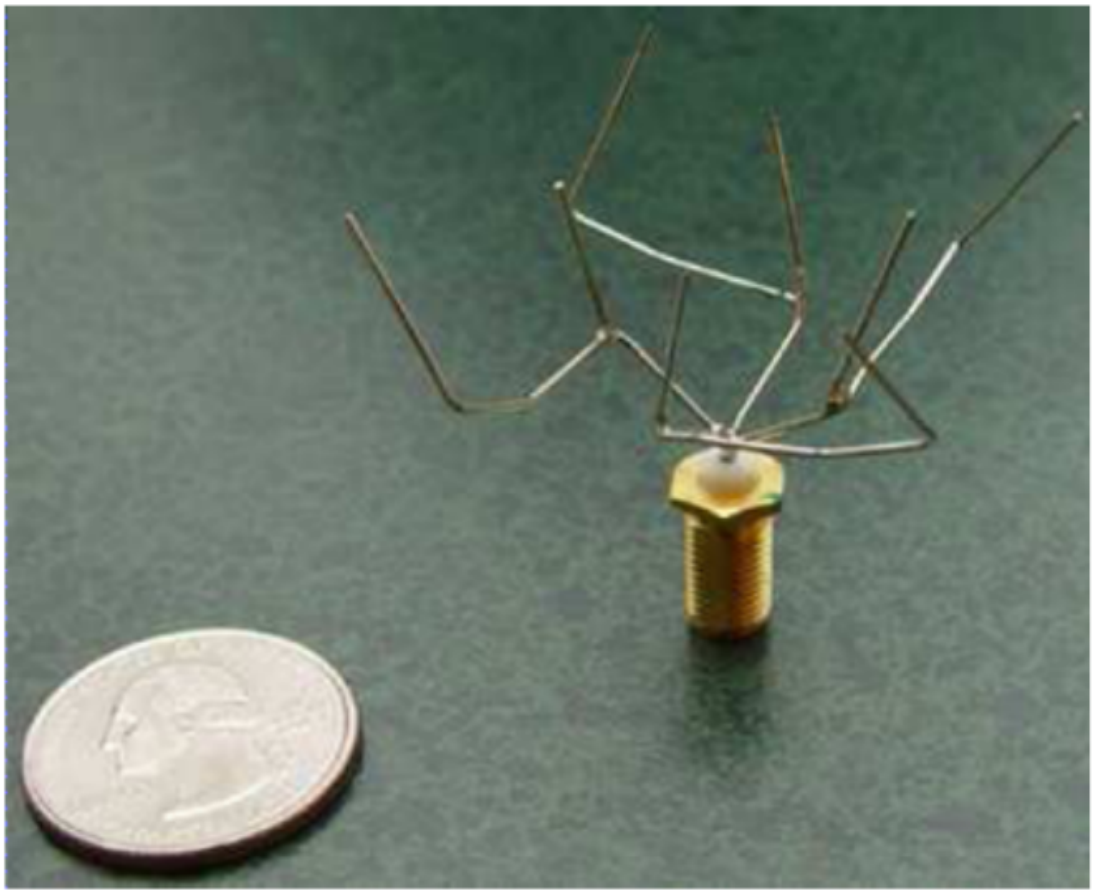
When can you expect GP to do well?

- If the relation between variables is not well understood.
- If the form of the final solution is not known.
- With that said it is quite common to let GAs find the approximate solution in a large search space and then further optimize the result via more conventional methods.



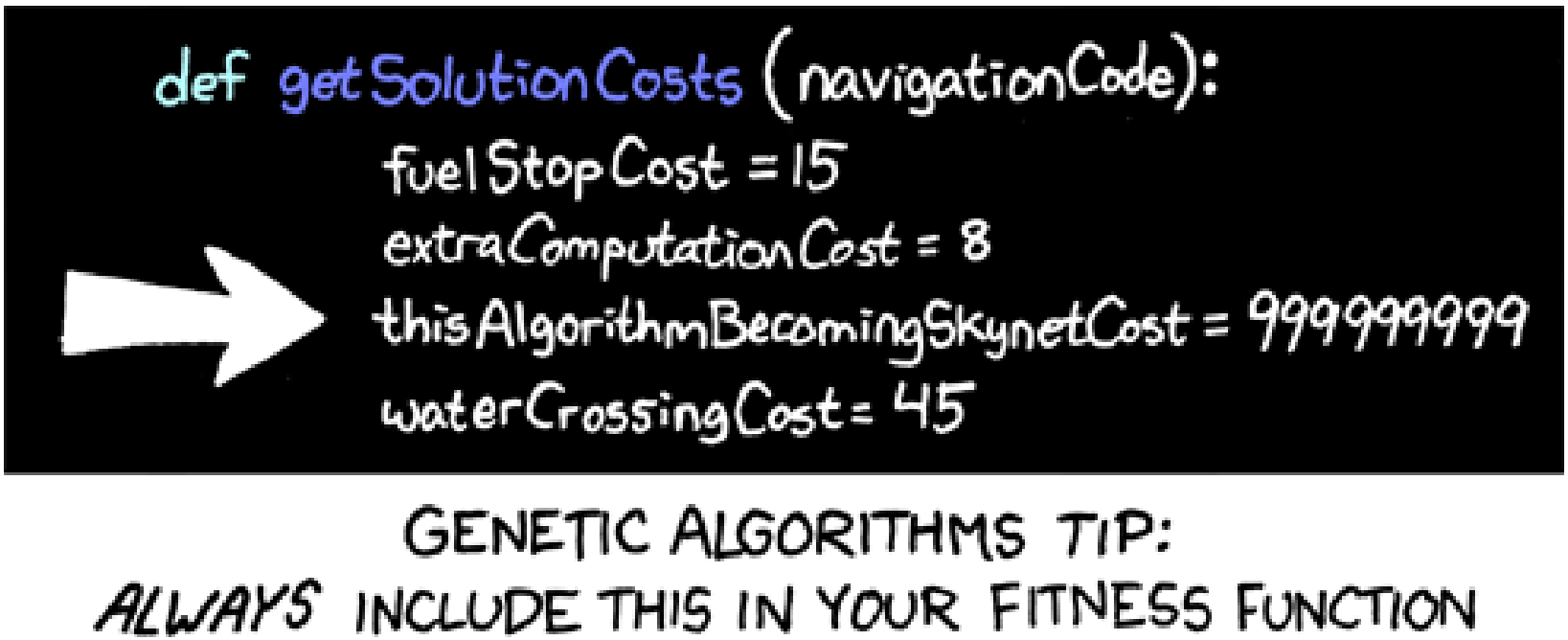
Applications: The NASA antenna

This antenna were designed via GAs for use with micro-satellites and deployed on NASA's Space Technology 5 Mission in 2006. The strange shape worked much better than any type of antenna that "human engineers" created.



Applications: Computer games

- Developers are trying to include better AI i games, most predominately in shooters, where the AI needs to be able to adapt to the players general skill.
- GAs are being applied to games in order to test the GAs themselves and their ability to solve problems of a general type.
  - Roller-coaster tycoon.



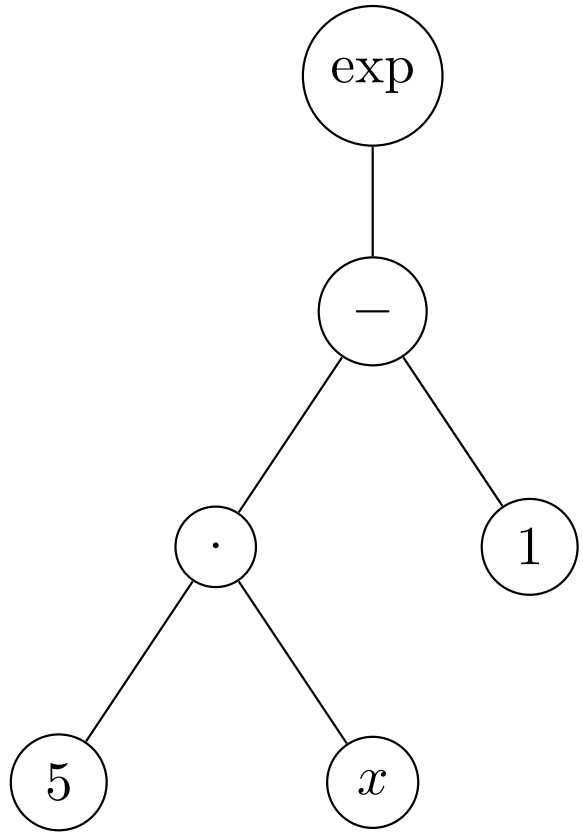
Applications: In chemistry

- GAs have been used in solving quantum chemical problems. All these examples performs faster and are more efficient than more standard procedures. Hartke observes that GAs might even be more reliable.
- Hartke and co-workers:
  - Geometry optimization of a Si_4 atomic cluster.
  - Optimization of basis sets.
  - Finding optimal protein folding arrangements.
- Sierka and co-workers:
  - Optimization of surface structures and gas-phase clusters.
  - Geometry optimization of Ti and V oxide atomic cluster.

Syntax trees: Nomenclature I

It is customary to express GP programs with (abstract) syntax trees, which only shows the structure of code and not the details.

- Terminal expressions are called leafs or simply terminals.
- Non-terminal expressions (internal nodes) are called functions.
- Depth of a tree is the depth of the deepest leaf.
- The root of the tree is at the top (depth zero).

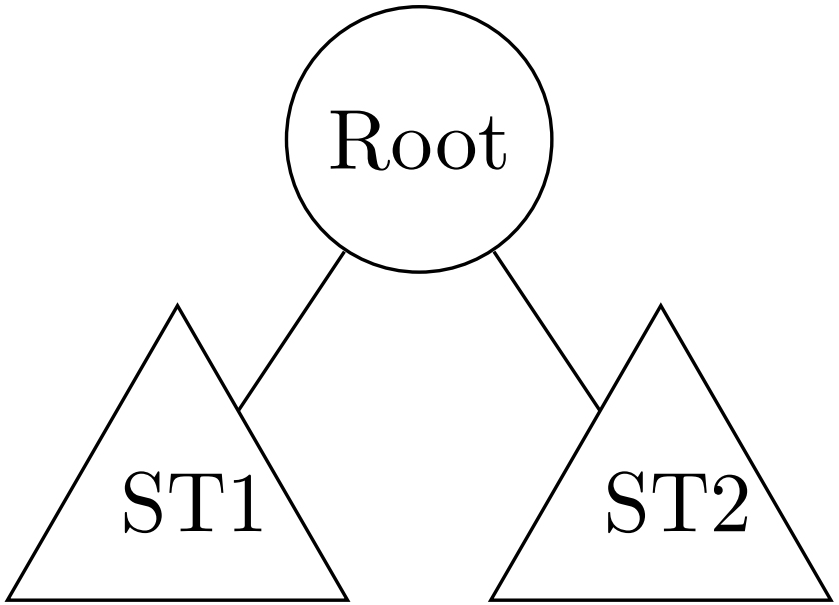


- Syntax tree example for the program 'exp $(5 \cdot x - 1)$' (infix notation) or '5 x $\cdot$ 1 $-$ exp ' (postfix notation) of depth 3 with 3 leafs and 3 functions.

Syntax trees: Nomenclature II

The set of available functions and terminals are part of the primitive set.

- Should a program have multiple components or subroutines, a "sub-tree" is spawned for each, called a branch.
- Evaluation of the components is an interpretation of the syntax tree.
- The architecture of the program is defined by the number of branches.



- Syntax tree example with 2 branches, where each is a syntax (sub-)tree in itself.

Syntax trees: A touch of biology

GP mimics the natural evolutionary process by introducing the following "cycles of evaluation", where selection and reproduction is repeated until a satisfactory result is obtained.

- Population: Randomly or non-randomly (seeding) individuals as the starting point for (each iteration of) the genetic algorithm.
  - A seed individual might do much better than randomly generated individuals and its descendants can take over quite quickly, thus resulting in loss of diversity.
- Selection: Evaluate the fitness of each individual in the population and establish their parent potential.
- Reproduction: Each individual with a non-zero parent potential are allowed to produce one or more children, which is added to a new population, which replaces the old population.
  - Sometimes the best individuals of the previous generation is transferred unaltered to the next generation (elitism).

Syntax trees: Selection

The method of selection depends on the problem at hand but it is hard to determine which method will do best before the calculation have been performed.

- Simple selection: Neighbouring individuals in the population are selected to produce a child, (lazy reproduction).
- Tournament selection: Individuals at selected at random from the population and compared to each other in terms of fitness in order to establish which is more suited to become a parent.
- Roulette selection: Individuals are being picked based on the actual value of the fitness function.

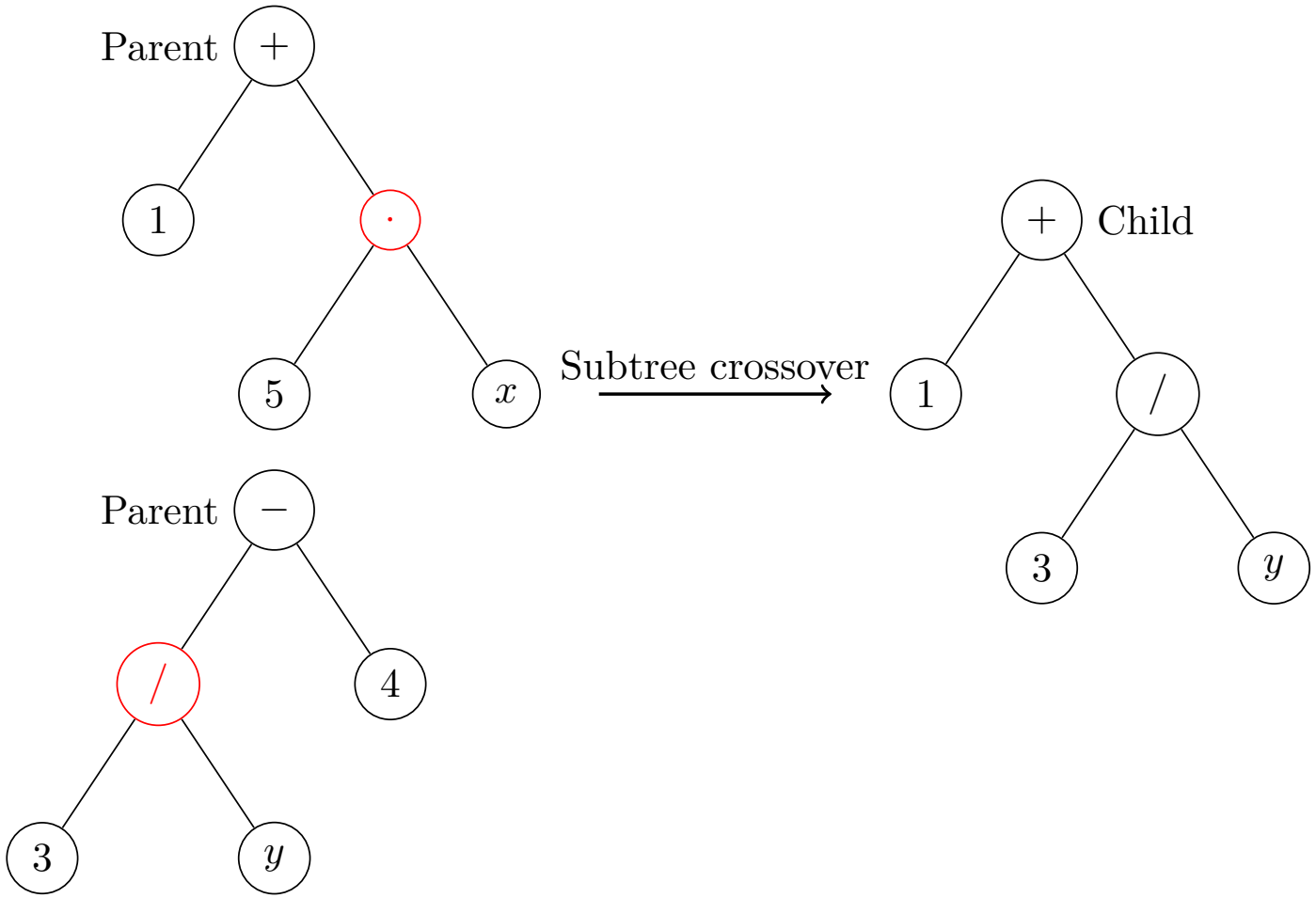
Syntax trees: Reproduction

There are two main operations in GP used to propagate towards a solution to the problem, whist keeping a reasonable amount of variation in the population.

- Crossover: Combine parts of two parents to produce a child.
  - A parent can have multiple children.
  - There can be more than one crossover point when exchanging genetic material.
  - It is possible to define a 3-parent crossover.
- Mutation: Each child have a chance of being randomly mutated before being committed to the new population.
  - Point mutation: Each node of a child have a chance of being randomly replaced by a different random primitive of the same size taken from the primitive set.
  - Subtree mutation: Randomly mutating a node in the child by introducing a randomly generated subtree.
  - The optimal probability of mutation depends on how much variation each generation needs, i.e. how similar conditions each generation encounter.

Syntax trees: Reproduction, Crossover

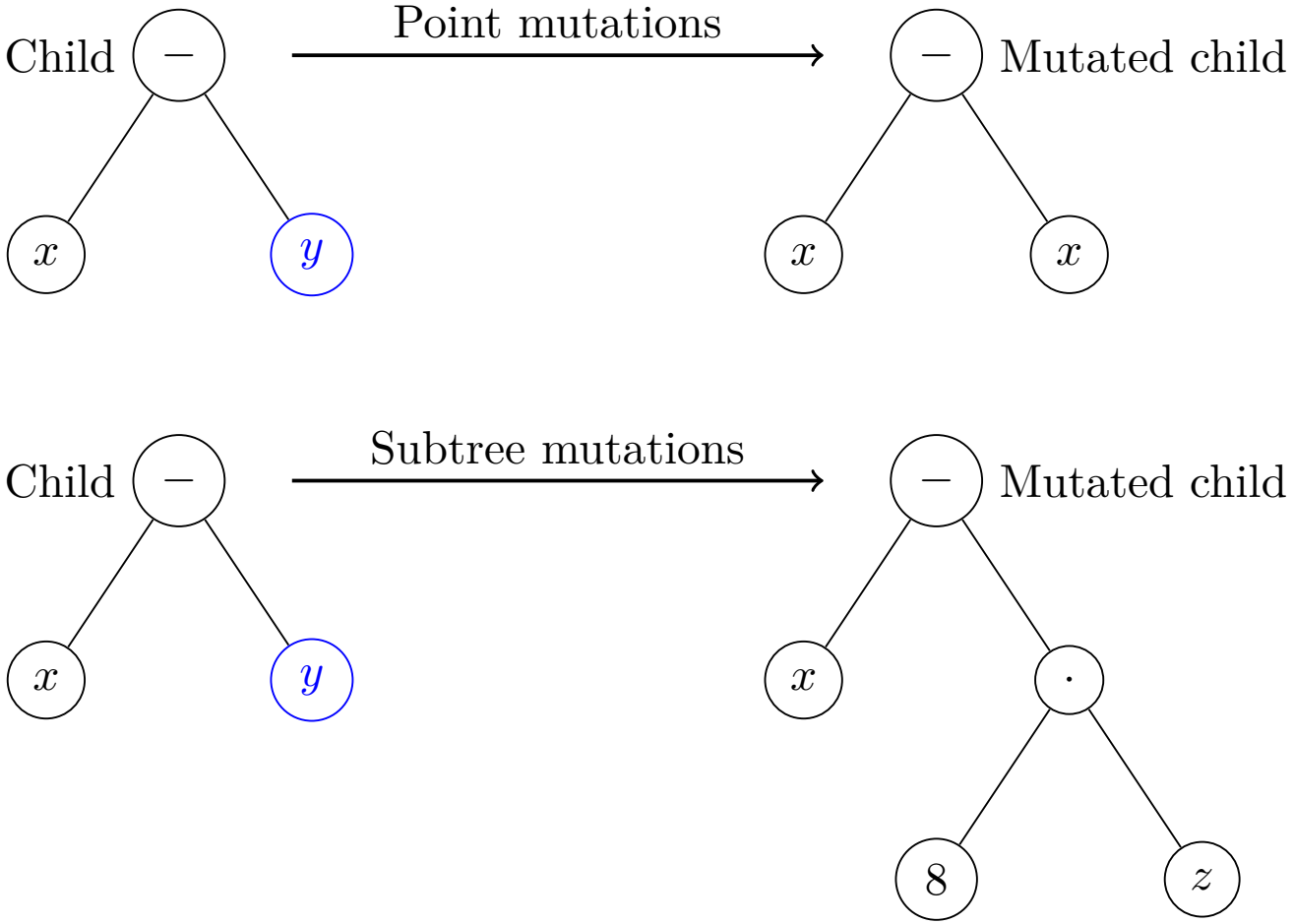
The subtree crossover is a single-point crossover, where a node on each parent tree is chosen randomly or with some probability as crossover points.



- The crossover points are indicated in red. Everything not "included" in the child dies.

Syntax trees: Reproduction, Mutation

Mutation can solve the problem of in-breeding in a population by introducing random variations in individuals.



- The mutation points are indicated in blue.

Preparatory steps

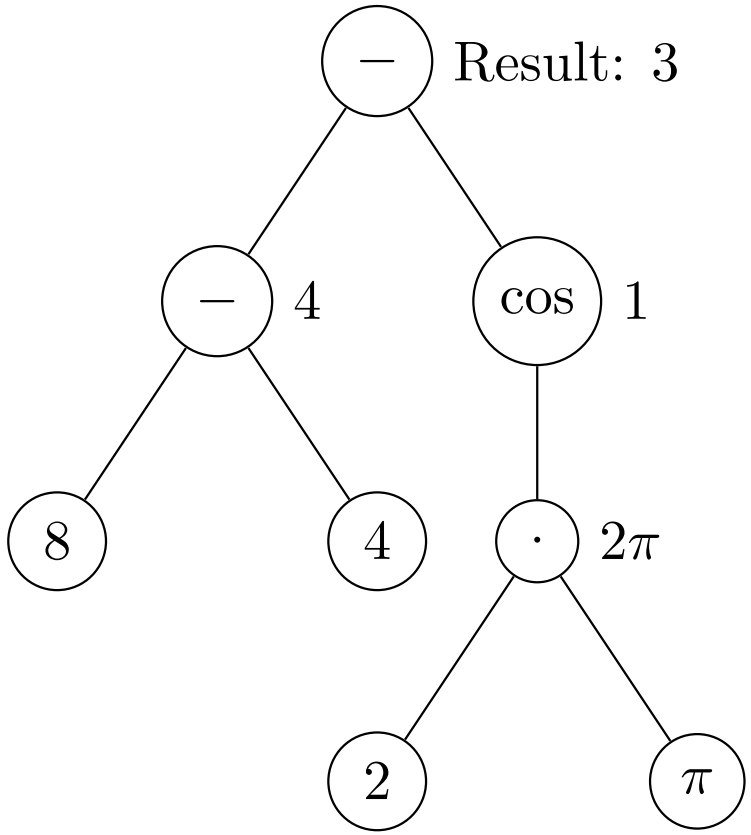
A couple of decisions should be taken before a GA is invoked:

- What is the primitive set?
  - Function set: Input to the GA, (arithmetic and/or mathematical functions, strings, boolean etc.), i.e. defines the search space.
  - Terminal set: Output of the GA, (mostly variables or constants).
  - Sufficiency: Genetic material present in the function set, should eventually be able to produce an individual that fulfil the termination criterion.
- Termination criterion: When should the GA stop?
  - After a maximum number of generations (iterations).
  - When an individual displays a satisfactory fitness.
- What is the measure of fitness?
  - Error to desired output, accuracy in pattern-recognition, scoreboard etc.
- What are the (internal) parameters?
  - Population size: Decreasing, increasing or constant.
  - Genetic operation probabilities: How often should a mutation or crossover be performed.
  - Maximum size of individuals: The depth of syntax trees for example.

The fitness function

The fitness function is used to evaluate individuals of the population. This is the most expensive part of the GA.

- All individuals of the population will need to be evaluated, i.e. what value is generated from evaluating the syntax tree.
- The is usually done by an interpreter, which travels from the root node to the deepest leaf and then evaluates each node from the bottom and up.



- The number to the right of each internal node is the evaluation result, which is then passed further "up" the tree.

Infinite monkey theorem: The short story

The infinite monkey theorem states that if we let a monkey hit a typewriter at random in an infinite amount of time, then the complete works of Shakespeare will be produced at some point.

- This amounts to a brute force search for the correct "phrase".
- Lets us instead try to write 'cat' as a limiting case
  - Initial population: dog, can, tan, ban, man
  - Fitness function: How many characters have the right position
  - Use roulette selection
  - Mutation rate: 25 %, (low variation in initial population)

Symbolic regression: Preparatory steps

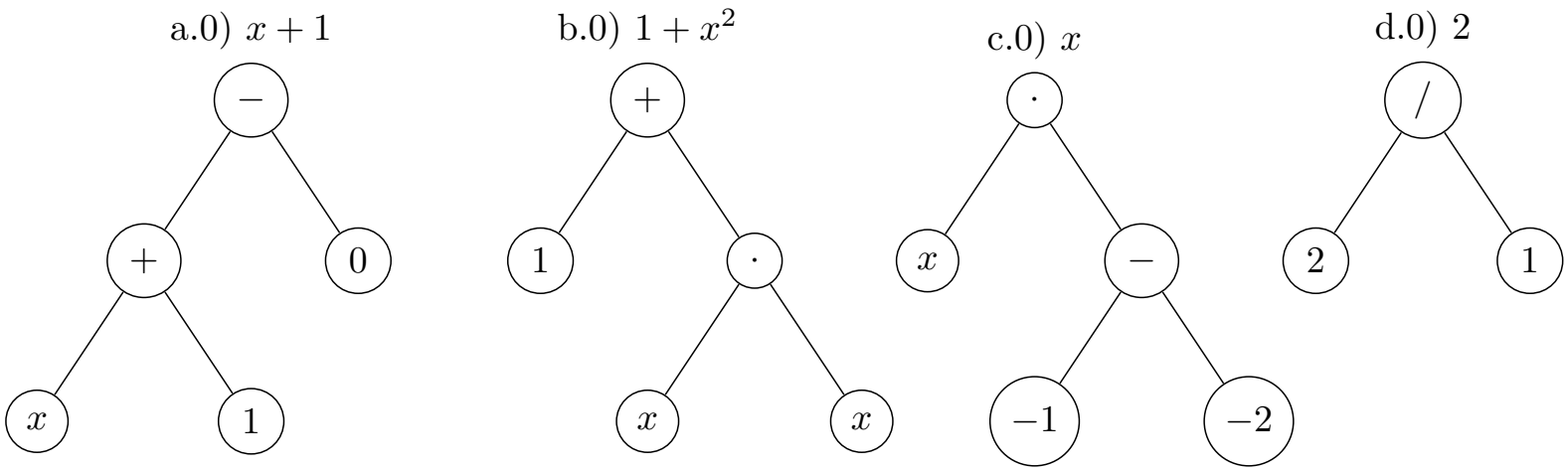
The objective is to evolve an expression that outputs similar value as the quadratic polynomial $x^2 + x + 1$ in the range $x \in [-1, 1]$.

- Function set: Numerical procedures, $(+, -, \cdot, /)$ and constants.
- Terminal set: x and constants
- Fitness: Sum of absolute errors for $x \in [-1, 1]$ in intervals of 0.1.
- Parameters: Population size should always be 4. We will use subtree crossover, elitism (reproduction) and allow 25% subtree mutation to evolve the population.
- Termination criterion: Whenever an individual achieves a fitness better than 0.1.

We are now ready to create the initial population evaluate their fitness before letting them evolve.

Symbolic regression: Initial population

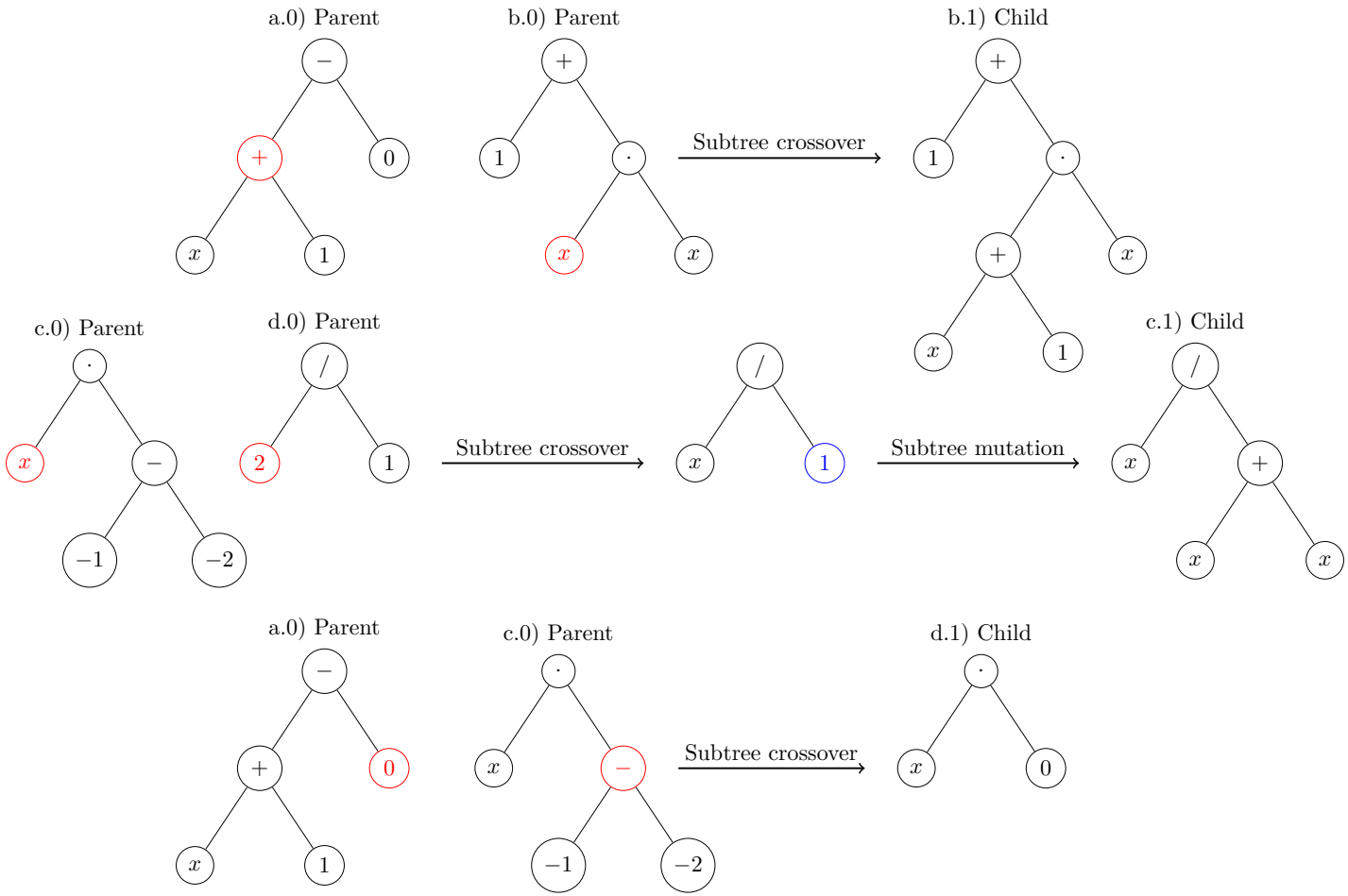
We obtain the following 4 randomly generated individuals.



- The selection process commences with evaluating the fitness of each individual.
- Fitness of the four individuals is calculated to be a.0) 9.40, b.0) 11.00, c.0) 28.70, and d.0) 17.98.
- The most fit individual is a.0) and this individual will be transferred directly to the next generation but will also feature in the selection process.

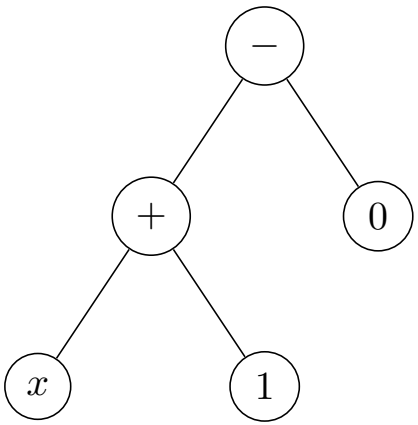
Symbolic regression: Reproduction

We choose 3 pairs of parents by roulette selection and ends up with the following members for the new generation of individuals

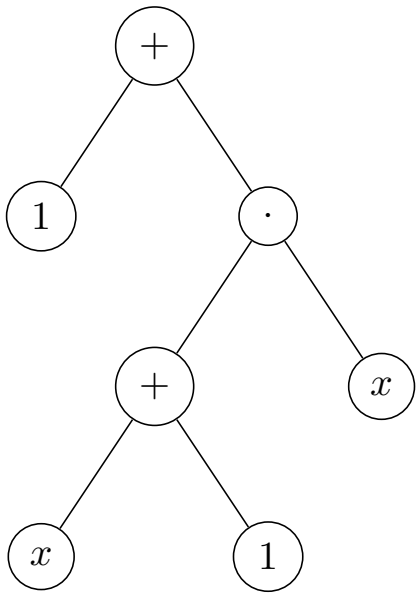


Symbolic regression: New generation

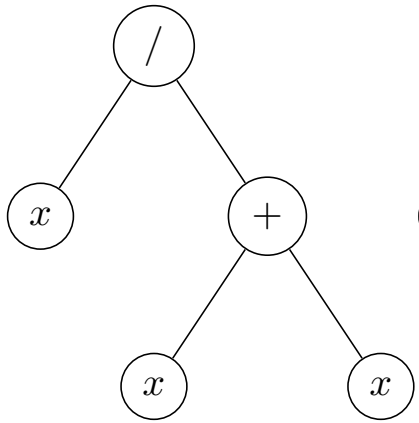
a.1) $x + 1$



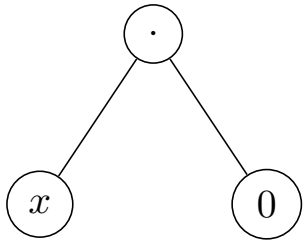
b.1) $1 + x^2 + x$



c.1) $\frac{1}{2}$



d.1) 0



- Fitness of the four new individuals is calculated to be a.1) 9.40, b.0) 0.0, c.0) 18.20, and d.0) 28.70.
- The termination criterion have thus been fulfilled for individual a.1). The genetic algorithm will thus offer this individual as the solution to the problem it was presented with.

Summary

- GAs are a great way to obtain unexpected solutions to problems, where more traditional methods have difficulties.
  - The solution is likely to be an approximation to the optimal solution.
  - Many arbitrary choice as part of the preparatory steps.
- Bloats, (significant program growth without significant improvement of the fitness) is a major concern and poorly understood.
- Parallelization is possible and comes in two main variants:
  - Without altering the selection process.
  - Having each node emulate different geographical environments for the same starting population, e.g probability of genetic operations or mating with individuals of "distant" populations.

Resources

- The field-guide to GP can be found at:
  `http://www.gp-field-guide.org.uk/`
- A nice GP presentation with examples:
  `https://www.youtube.com/watch?v=6l6b78Y4V7Y`
- Roller Coaster Tycoon hacking:
  `https://www.youtube.com/watch?v=KUBYTcVjp7I`
- Hartke papers:
  `http://pubs.acs.org/doi/pdf/10.1021/j100141a013`
  `http://dx.doi.org/10.1039/B412673D`
  `http://onlinelibrary.wiley.com/doi/10.1002/jcc.20254/abstract`
- Sierka papers:
  `http://www.sciencedirect.com/science/article/pii/S0079681610000237?via%3Dihub`
  `http://pubs.rsc.org/en/Content/ArticleLanding/2014/CP/c4cp00752b#\protect\kern-.1667em\relaxdivAbstract`