Artificial Intelligence
Università della Svizzera Italiana
Faculty of Informatics
December 20, 2019

Assignment 1: Genetic Algorithm & Alpha-Beta Pruning

Giorgia Adorni (giorgia.adorni@usi.ch)

1 Alpha-Beta Pruning

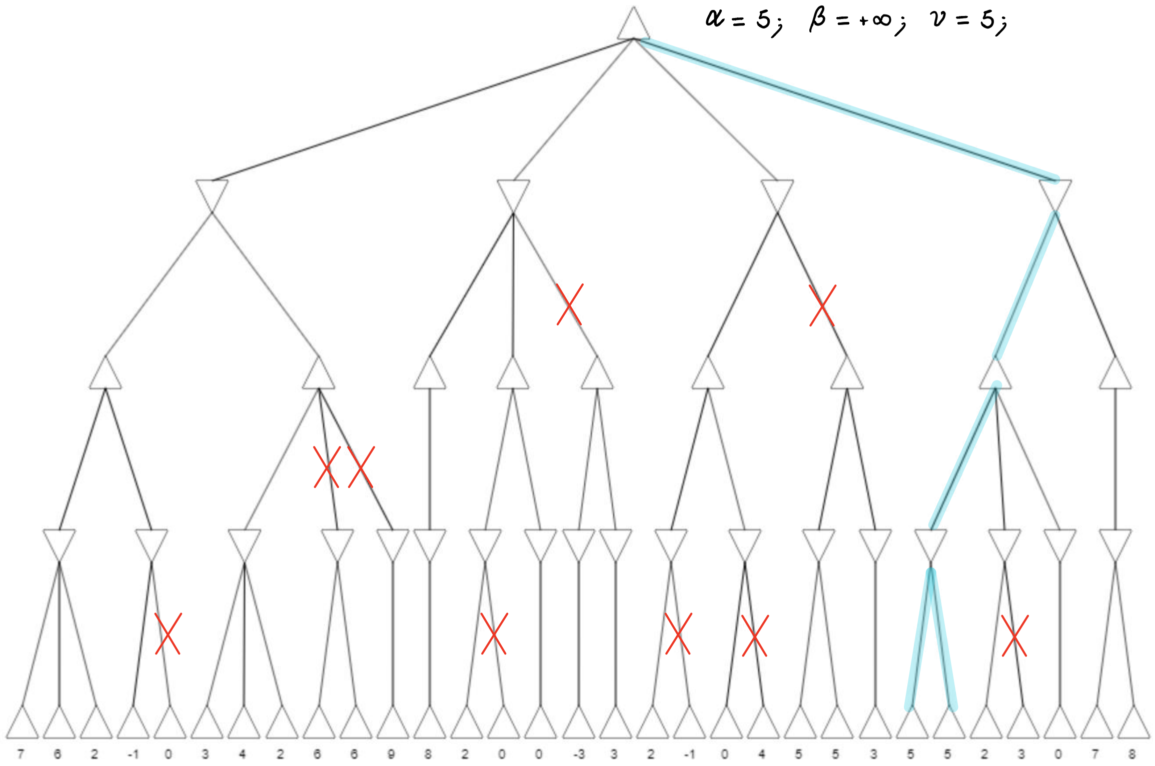

Figure 1: Alpha-Beta Pruning

2 Genetic Algorithm

Write high-level code for a genetic algorithm.

```
import numpy as np

# Each gene is a real number between 0 and 1 and corresponds to the
# probability to have 1 in the corresponding position.
n = 10
P = np.full(n, 0.5)  # initialisation

k = 100
generations = 10
m = 5
stop = False

while not stop :
    # Produce a given k number of traditional 0-1 individuals each of
    # them of length n by sampling gene by gene with the prob vector.
    solution_vector = np.random.binomial(1, P, size=[k, n])

    # Evaluate each individual computing the fitness of the
    # solution_vector
    fitness = evaluate(solution_vector)

    # Geerates new individuals
    new_solution_vector = np.empty([k, n])

    for i in range(k // 2) :
        # Select the best parents in the population for mating
        ind1, ind2 = select_parents(solution_vector, fitness)

        # Generate next generation using crossover
        ch1, ch2 = crossover(ind1, ind2)

        # Adding some variations to the offsrping using mutation.
        ch1, ch2 = mutate(ch1), mutate(ch2)

        new_solution_vector[2 * i, :] = ch1
        new_solution_vector[2 * i + 1, :] = ch2

    # At the end of each generation, the best m individuals are used to
    # update the probabilistic vector.
    fitness = evaluate(new_solution_vector)
    best_individuals = np.argsort(fitness)[::-1][:m]

    # For each individual of the best m chosen, the probabilistic vector
    # is updated using a learning rate LR parameter (real number between
    # 0 and 1)
    LR = 0.001

    for b in best_individuals :
    P = P * (1.0 - LR) + new_solution_vector[b] * LR

    # Test if the solution has converged according to some criterion
    if converged :
    stop = True
```