Algorithm 1: Evolutionary Algorithm for Path Optimization

Input: Population size *population_size*, Generations *generation_times*, Fitness limit $fitness\_limit$, Task *task*, Probability parameters p_shift, *p_swop*, *p_up*, *p_down*

Output: Optimal genome *best_genome*

```
Initialize population with genomes of random points ;
best_fitness_seen ← −∞;
for i ← 1 to generation_times do
    Sort population by fitness in descending order;
    fitness ← fitness of best genome;
    if fitness > best_fitness_seen then
        best_fitness_seen ← fitness;
    end
    if fitness ≥ fitness_limit then
        Break loop;
    end
    Select top n_top genomes as next_generation;
    for k ← 1 to n_perturb do
        Randomly select candidate from top genomes;
        if rand < p_shift then
            Apply shift_to_end to candidate;
        end
        if rand < p_swop then
            Apply swop to candidate;
        end
        if rand < p_up then
            Increment position in candidate if within bounds;
        end
        if rand < p_down then
            Decrement position in candidate if within bounds;
        end
        Add candidate to next_generation;
    end
    for j ← 1 to (population_size − n_top − n_perturb)/2 do
        Select parents using tournament selection;
        Perform partial map crossover with probability 0.9;
        Apply swop with probability 0.9 to offspring;
        Add offspring to next_generation;
    end
    Update population with next_generation;
end
Return best_genome from population;
```