Efficient CPU Implementation of a Genetic Algorithm for the Travelling Salesman Problem

By Alex Fillmore

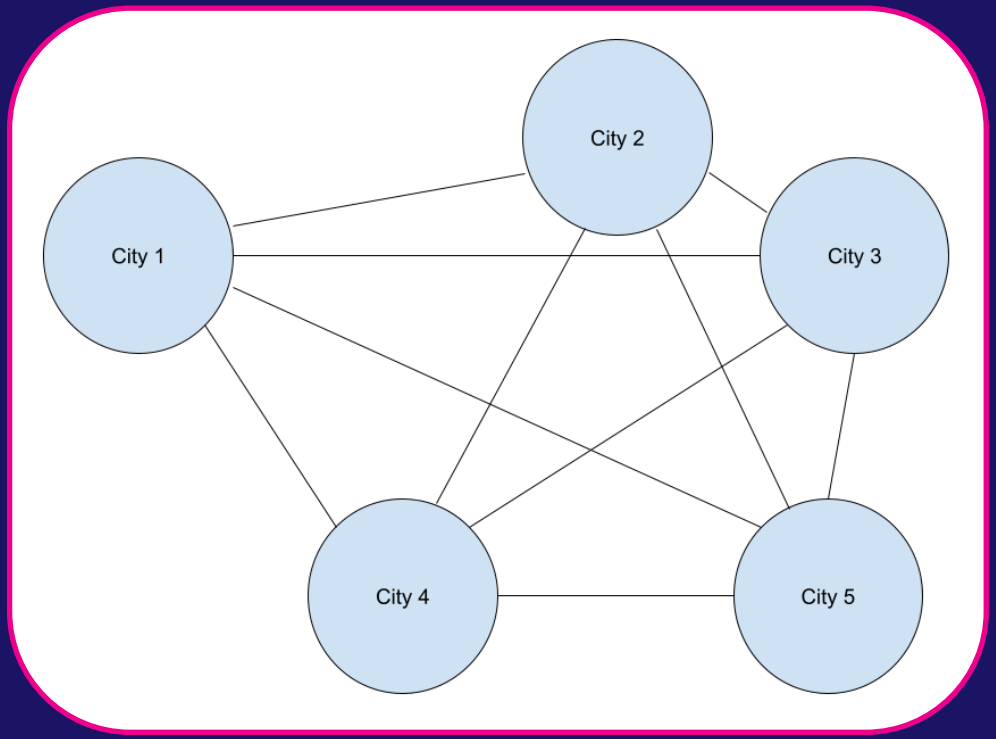
TABLE OF CONTENTS

01

Introduction

Travelling Salesman

A salesman wants to visit every city in the most efficient way



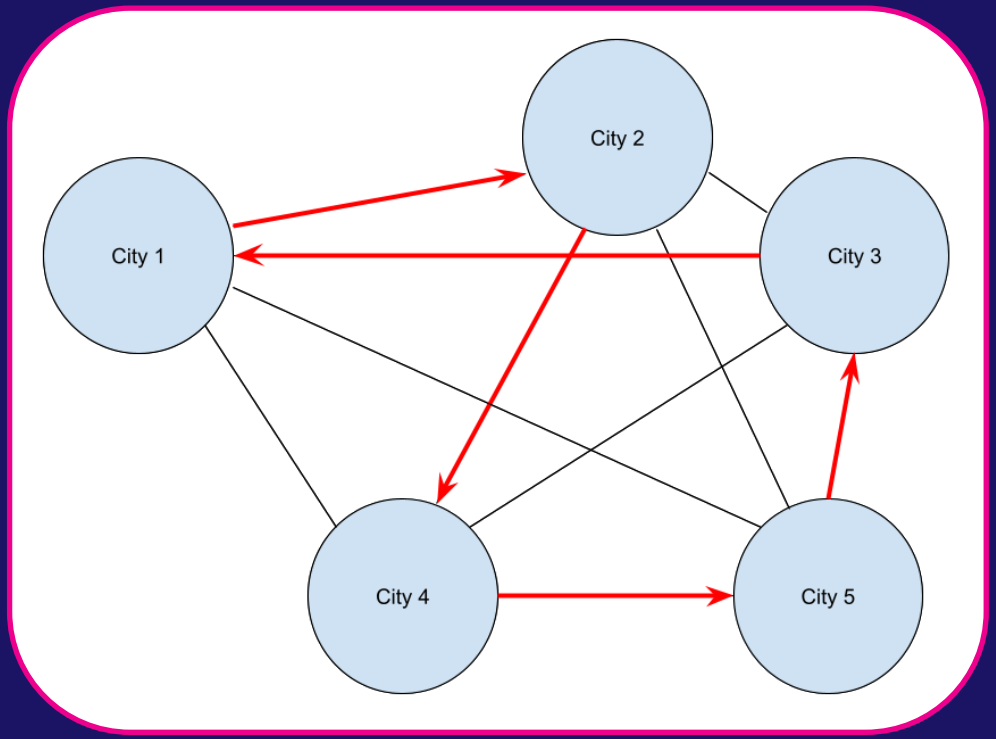
Travelling Salesman

One possible solution might look like this.

How do we know if it's the best solution?

NP-Hard Problem

O((N-1)!) Time Complexity

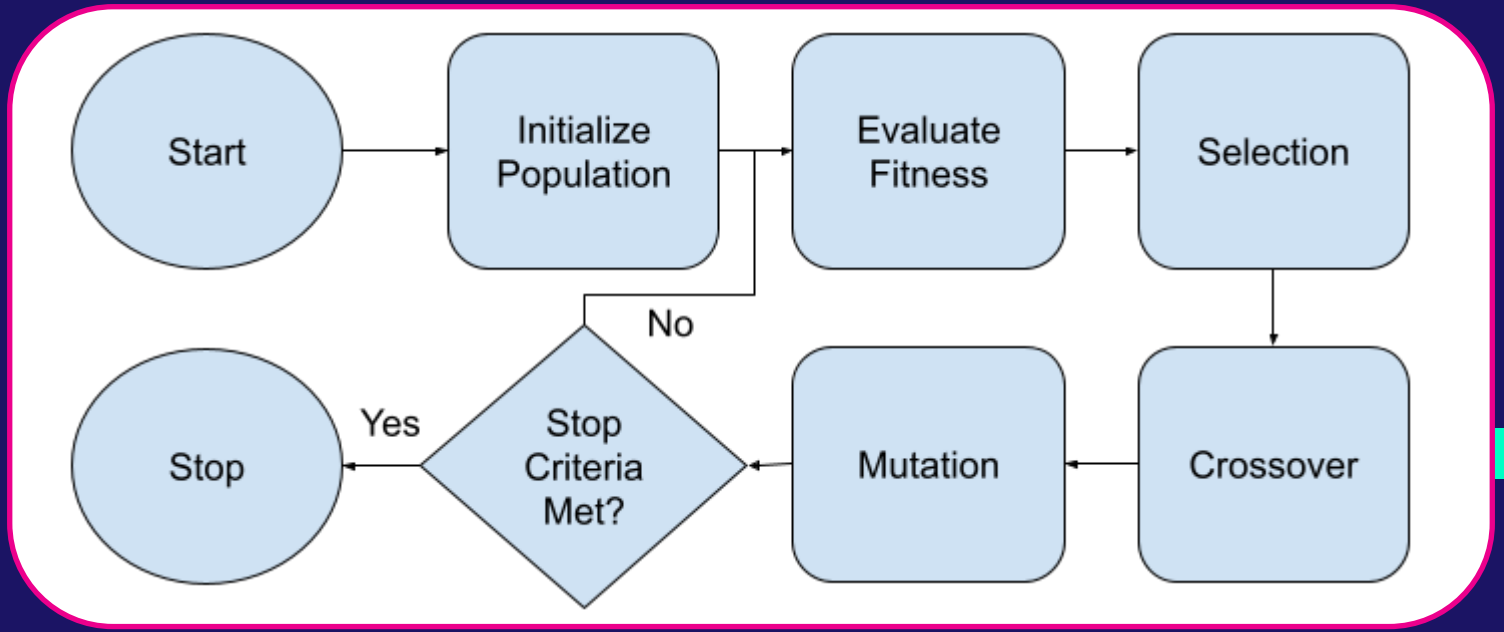


Genetic Algorithms

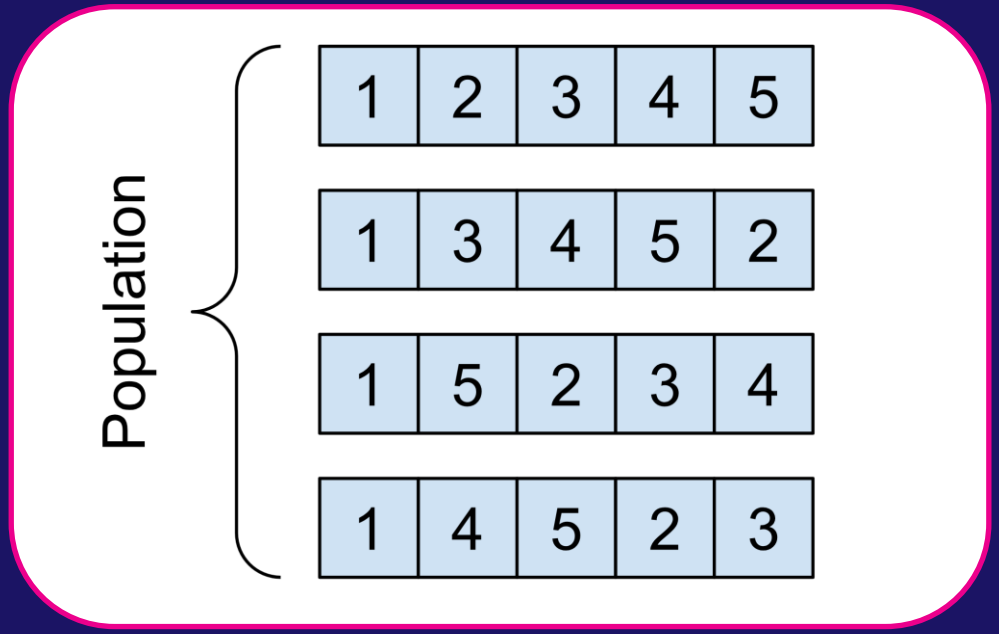
Metaheuristic Algorithm

Inspired by Biology

Highly Parallelizable

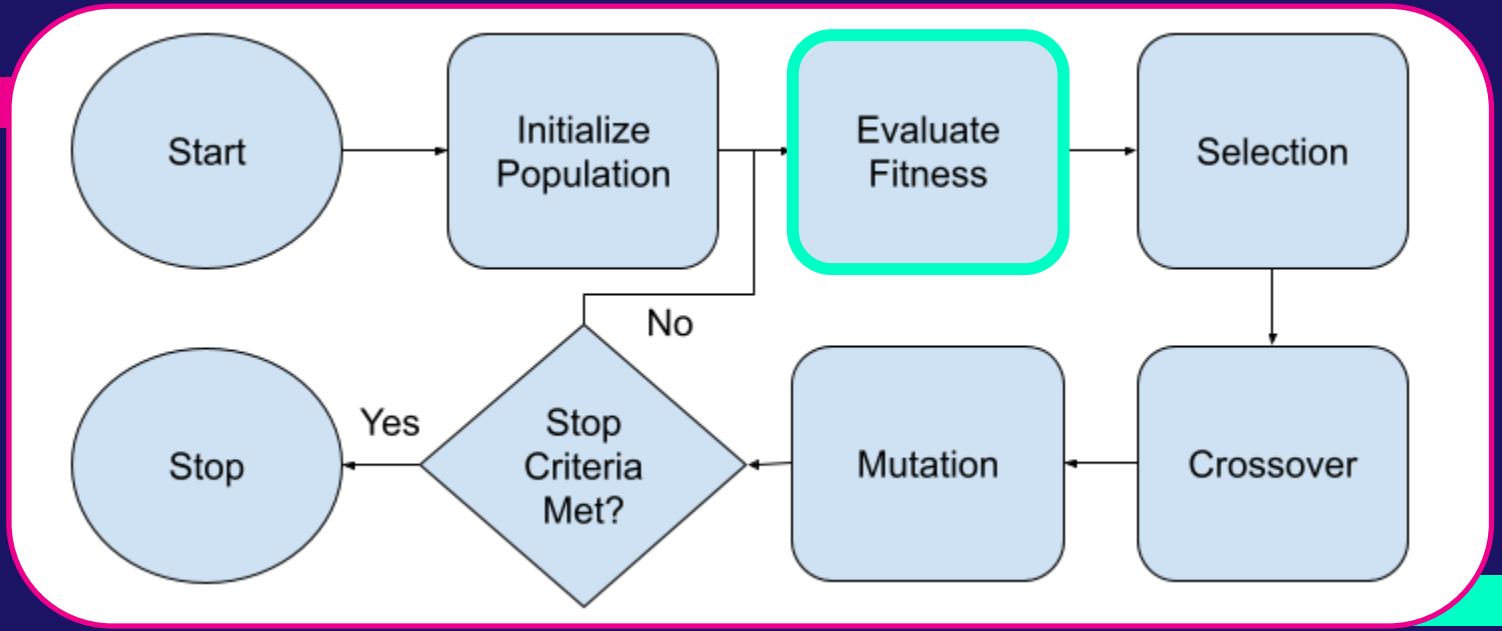


Genetic Algorithms



02

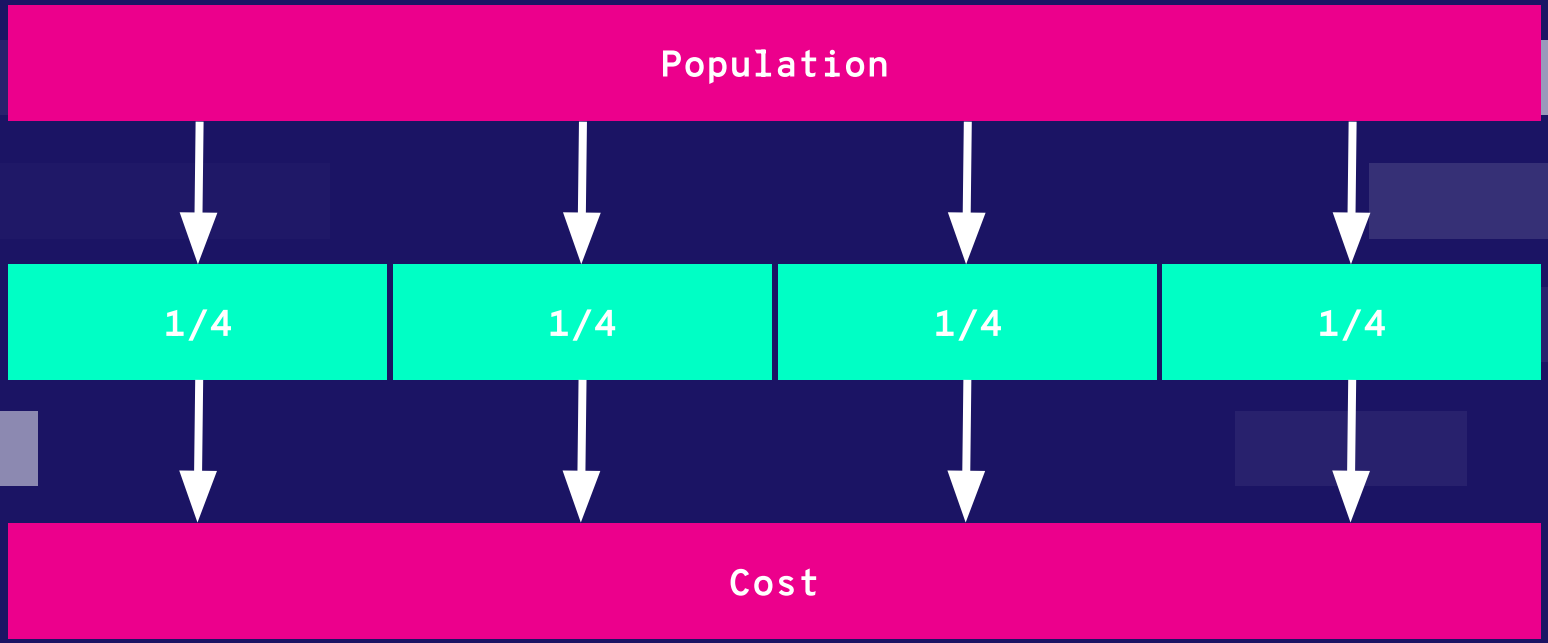
Algorithm


Start
Initialize Population
Evaluate Fitness
Selection
Crossover
Mutation
Stop Criteria Met?
No
Yes
Stop

Cost Function

```
// Loop through the population
for(int i=0; i<POP_SIZE; i++){
  float cost = 0.0;
  // Loop through solution
  for(int j=1; j<NUM_CITIES; j++){
    cost += dist[pop[i][j-1]][pop[i][j]];
  }
}
```

NUM_THREADS = 4

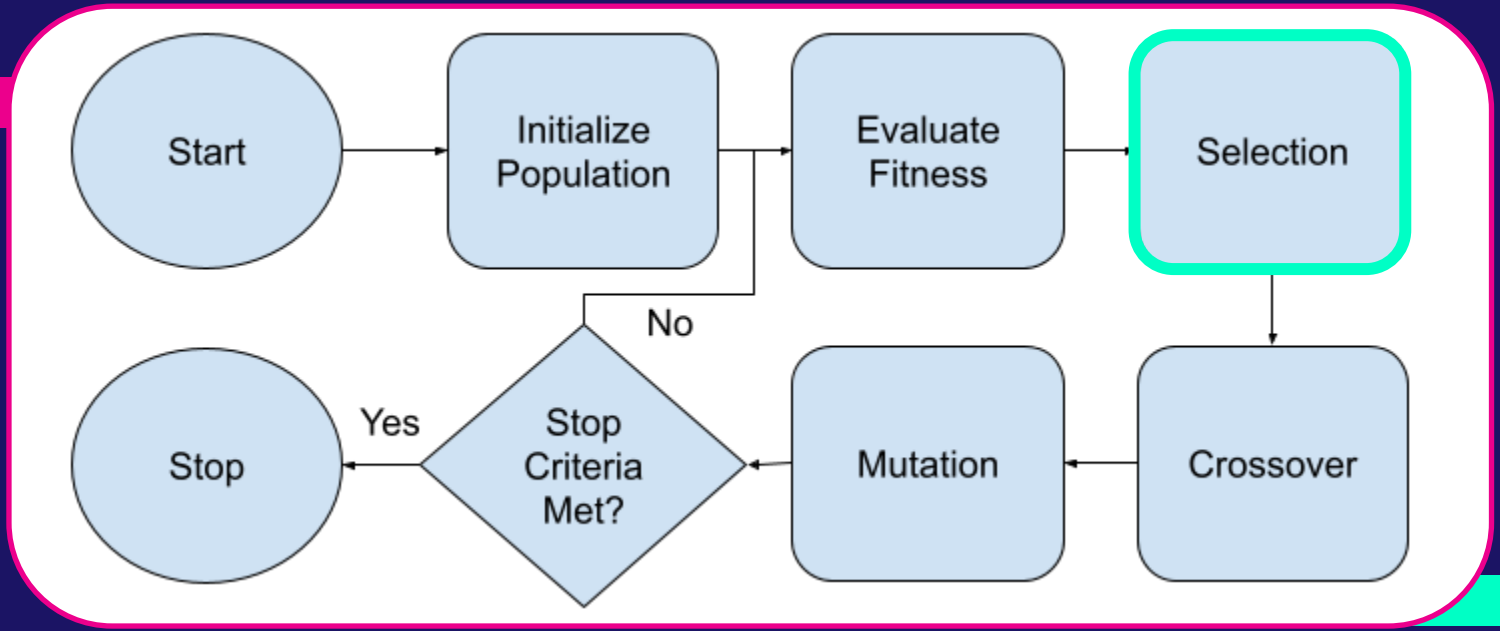


Cost Function

```
// Loop through the population
for(int i=start; i<end; i++){
  float cost = 0.0;
  // Loop through solution
  for(int j=1; j<NUM_CITIES; j++){
    cost += dist[pop[i][j-1]][pop[i][j]];
  }
}
```

Thread Argument Struct

```
// Structure for thread arguments
typedef struct {
  int **pop;
  float *cost;
  int *parents;
  float **cost_table;
  int start;
  int end;
  int seed;
  float *min;
  int thrdIdx;
} TH_args;
```

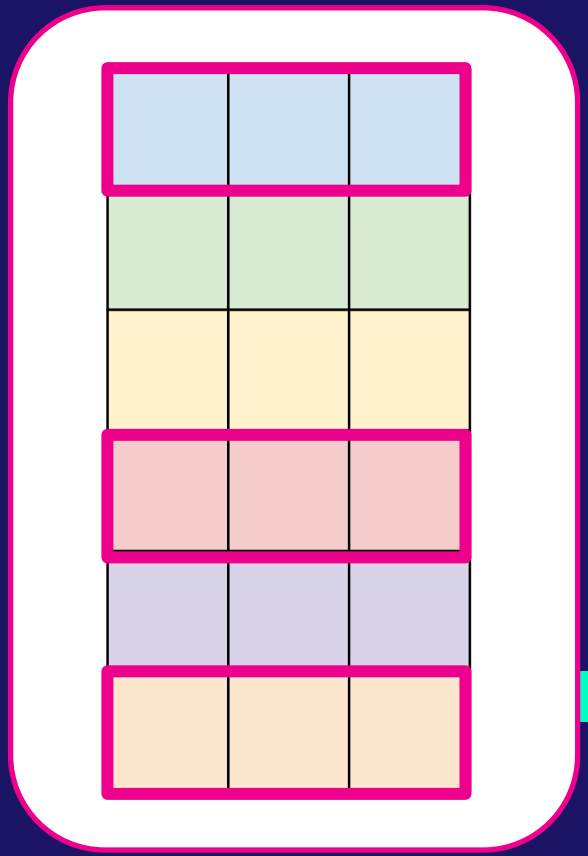

Start
Initialize Population
Evaluate Fitness
Selection
Crossover
Mutation
Stop Criteria Met?
Yes
No
Stop

Tournament Selection

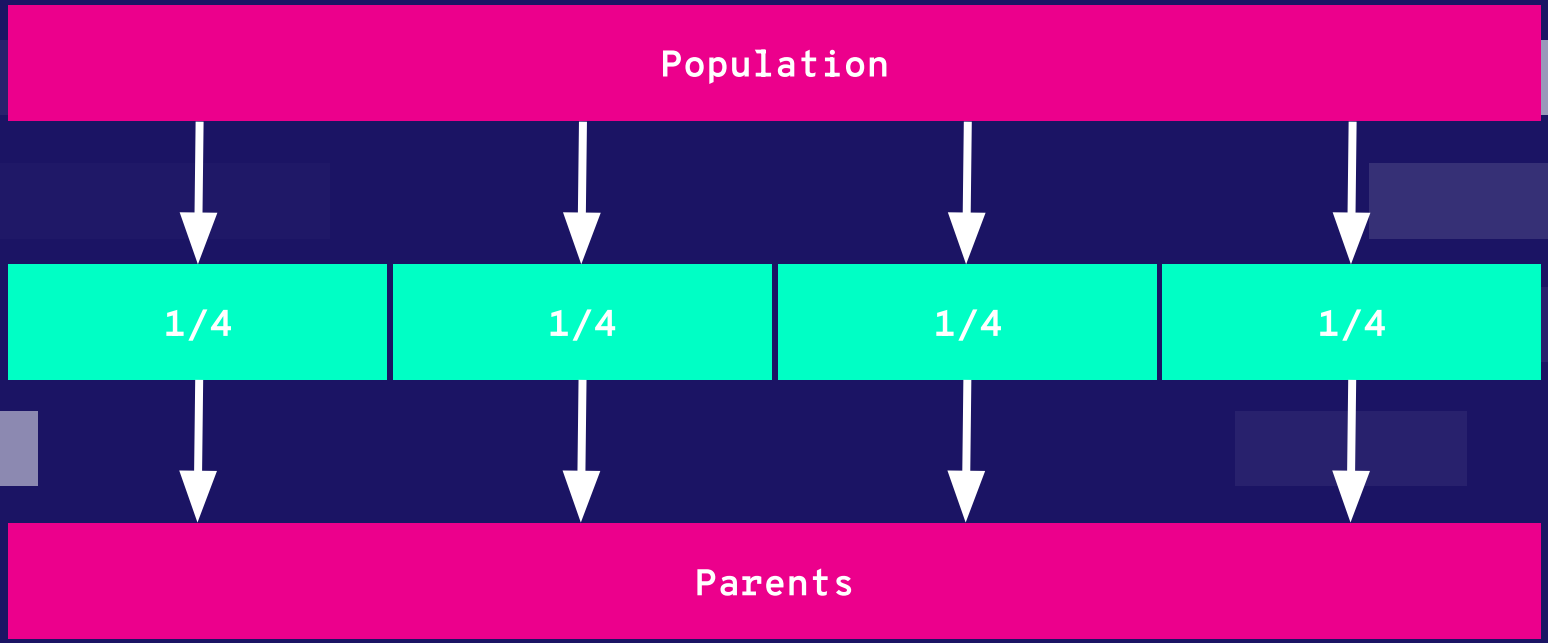
Make a subset of the population “compete” to reproduce

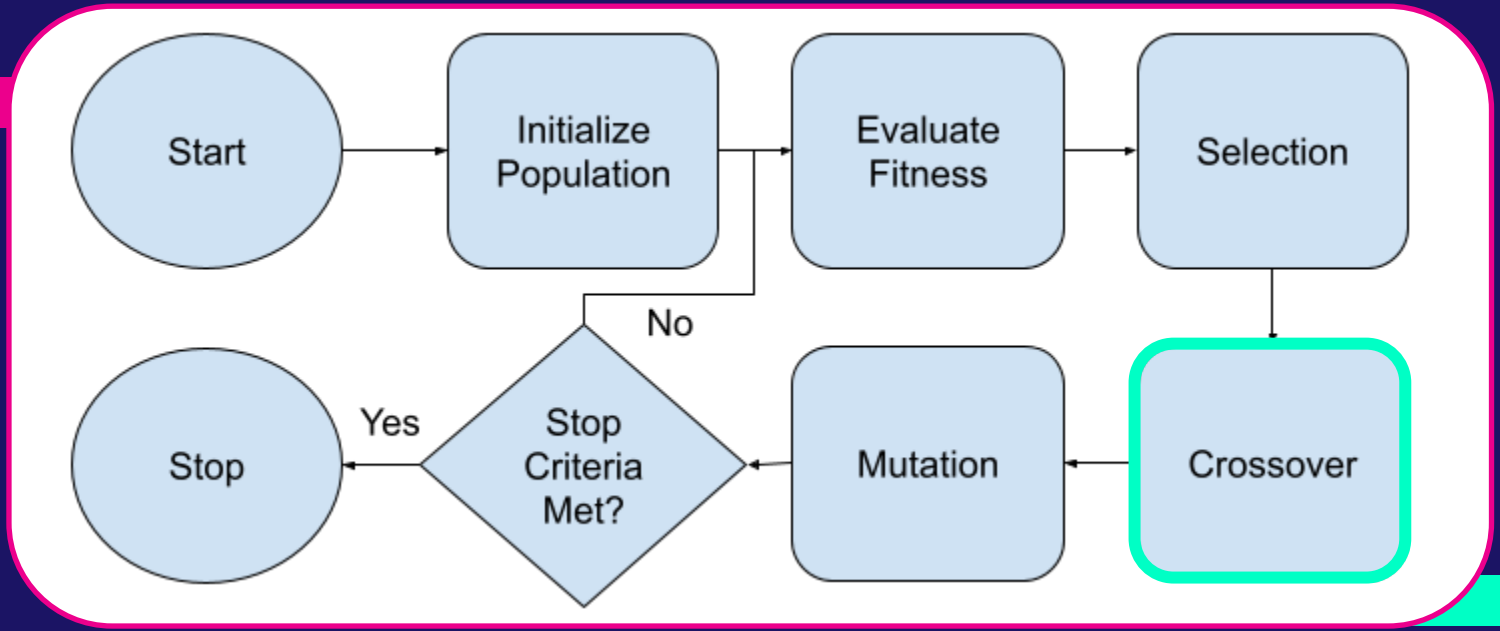
Compete via fitness/cost

The most fit wins!



NUM_THREADS = 4



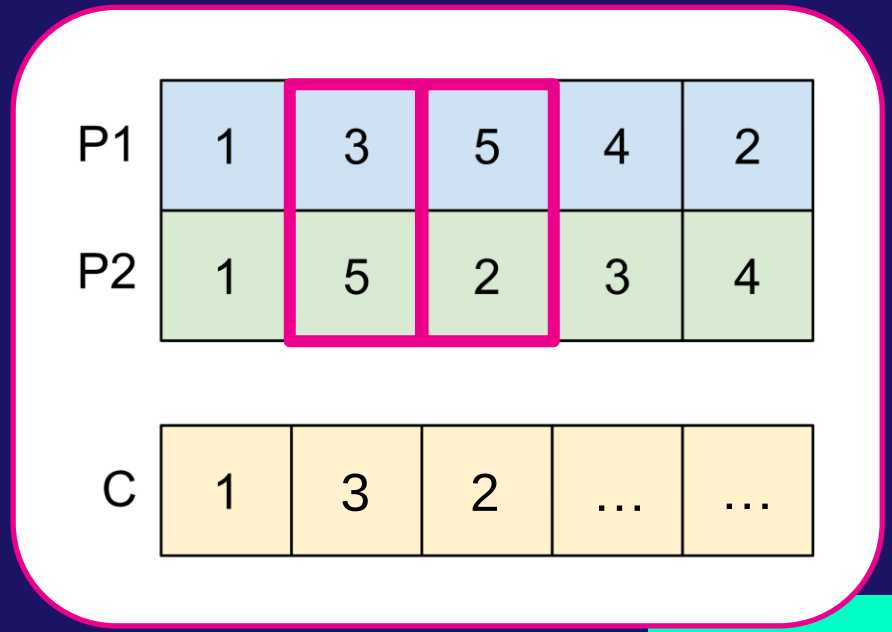

Start
Initialize Population
Evaluate Fitness
Selection
Crossover
Mutation
Stop Criteria Met?
No
Yes
Stop

Greedy Crossover

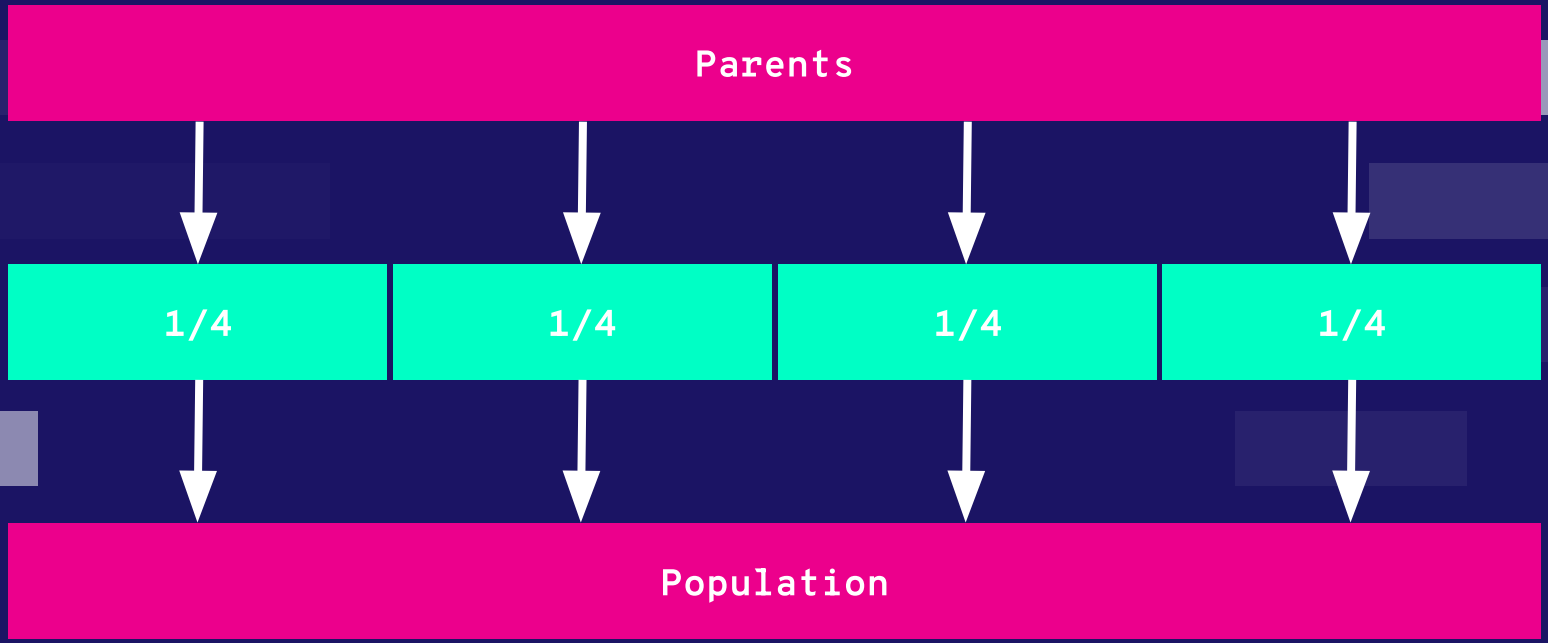
Given two parent solutions P1 & P2

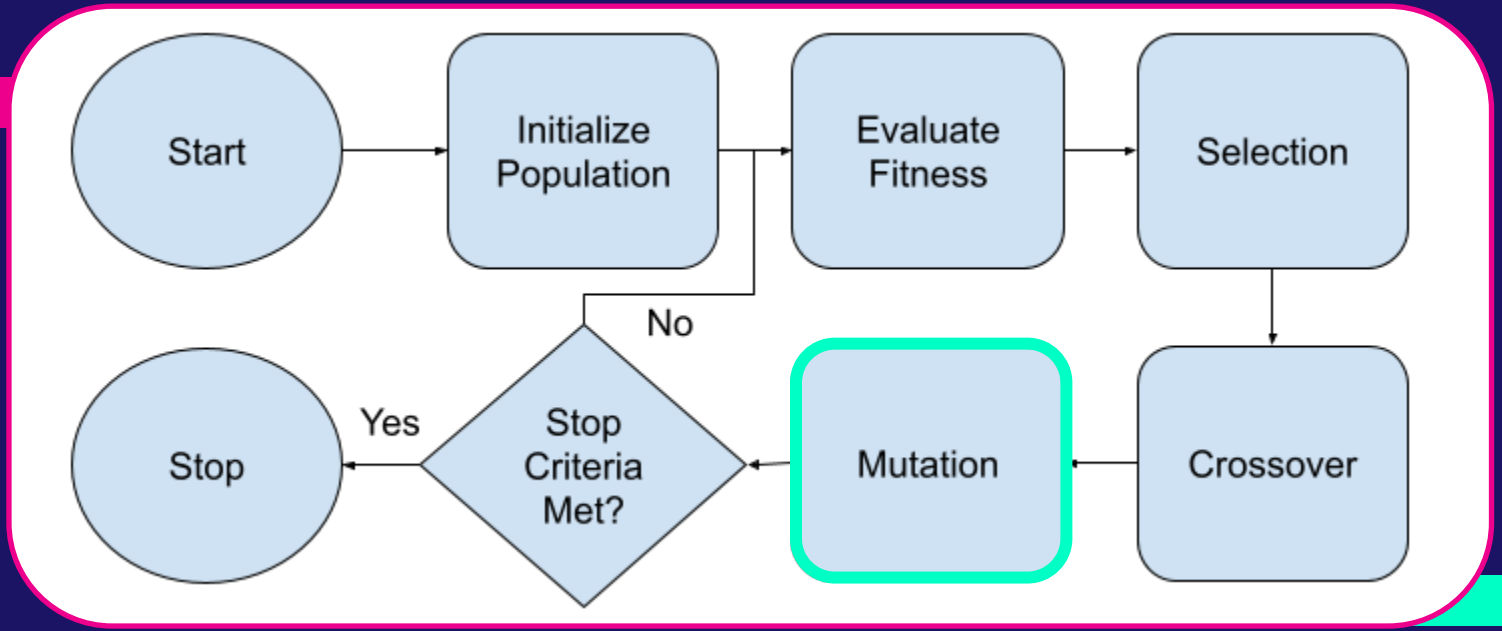
Create child C

Evaluate the next city in each parent and pick the best one

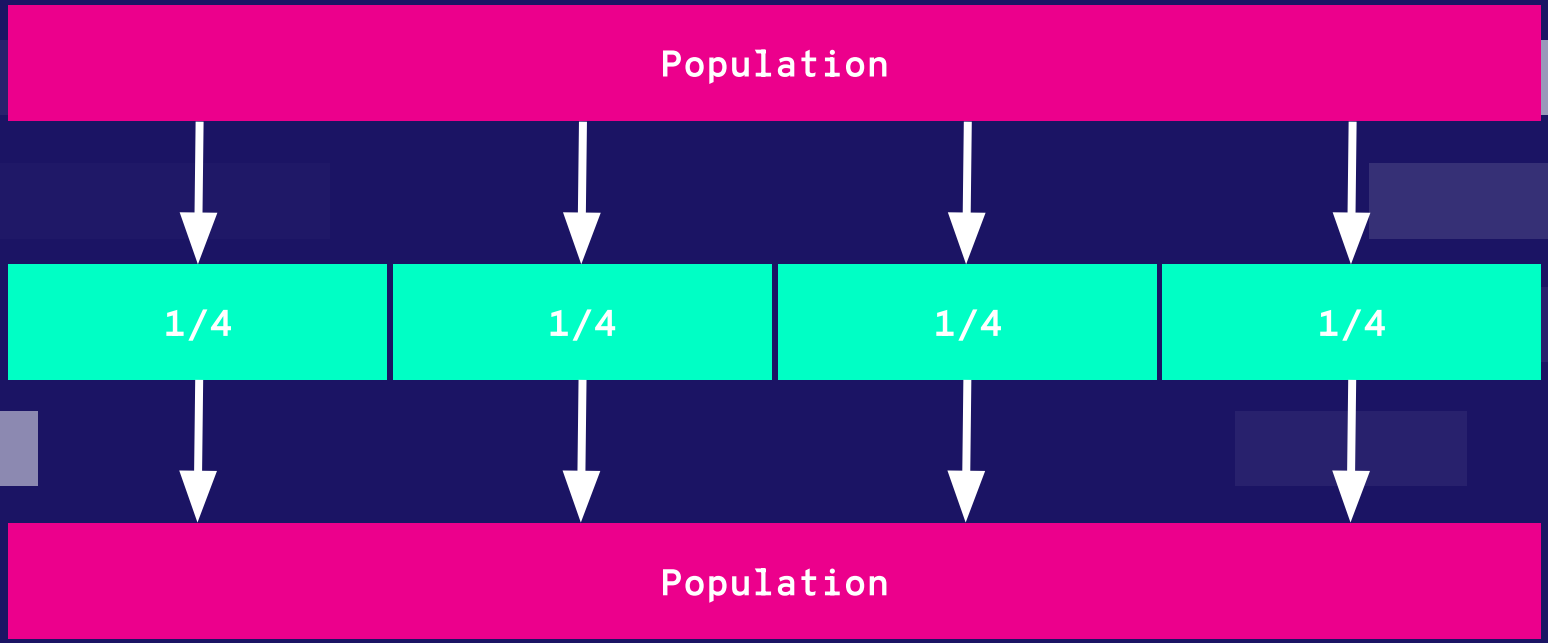


NUM_THREADS = 4




Start
Initialize Population
Evaluate Fitness
Selection
Crossover
Mutation
Stop Criteria Met?
No
Yes
Stop

NUM_THREADS = 4

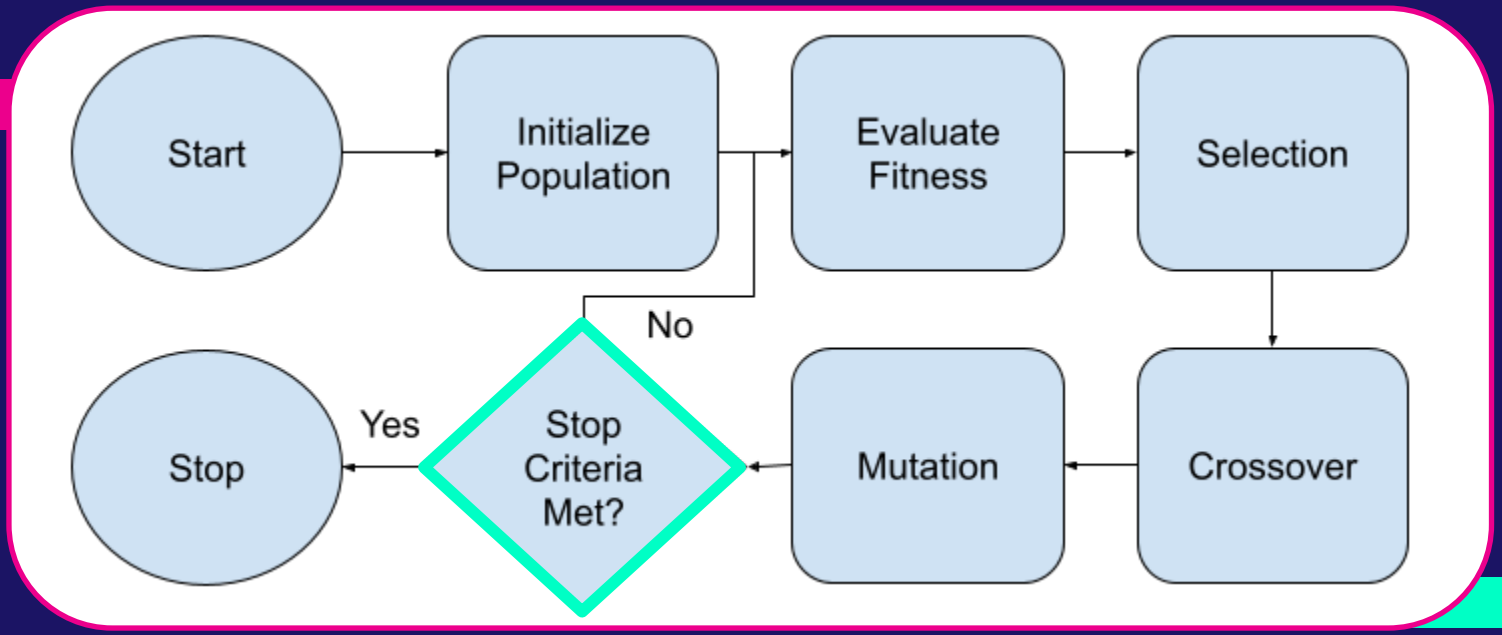


Sequential Mutation

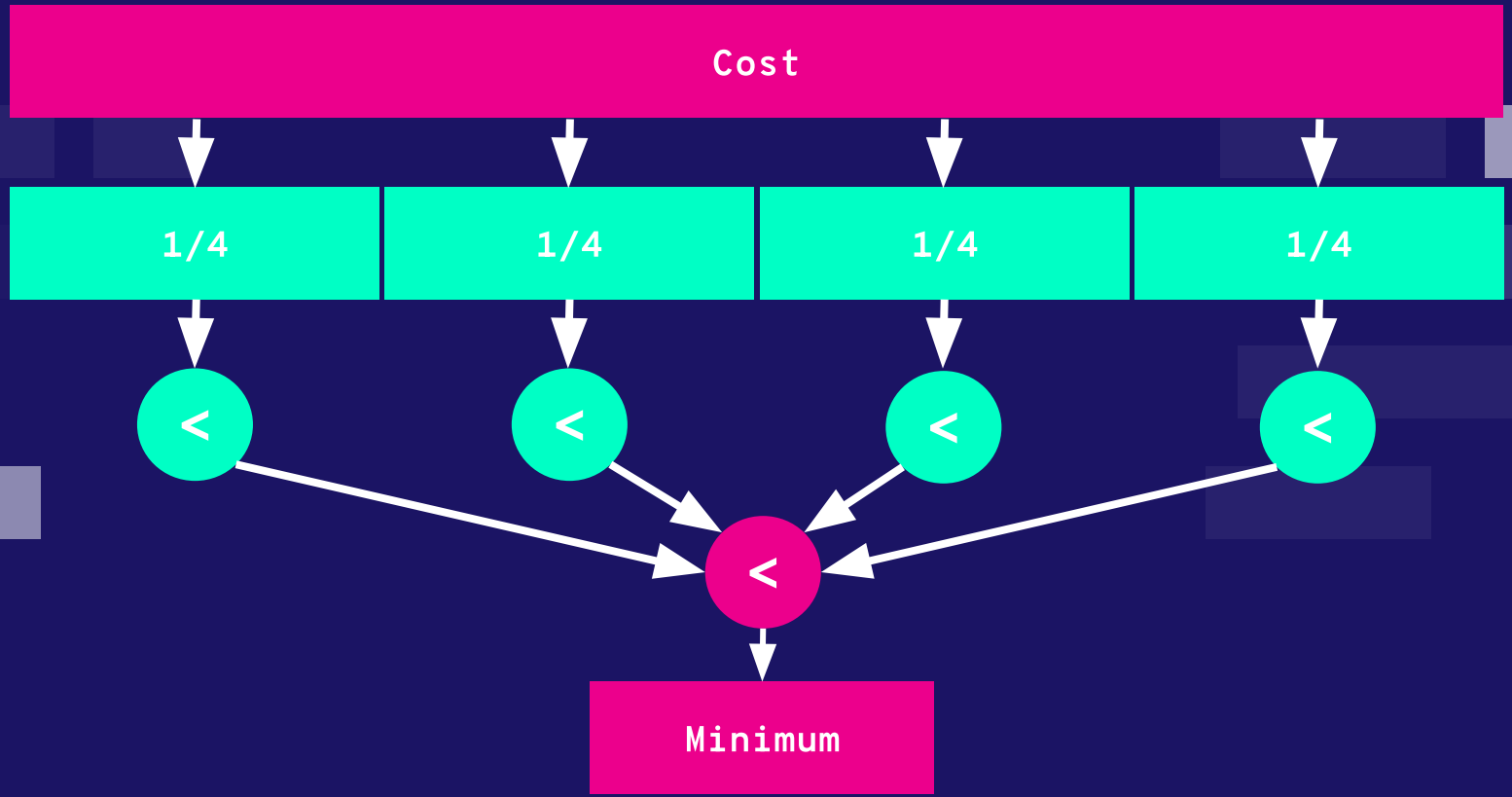
```
void mutation(int **pop){
  int i;
  // For the whole population
  for(i = 0; i < POPULATION_SIZE; i++){
    // If a random percent chance occurs
    if((rand() % 100) <= MUTATION_CHANCE){
      // Select two random idx and swap
      int index1 = rand() % NUM_CITIES;
      int index2 = rand() % NUM_CITIES;
      int temp = pop[i][index1];
      pop[i][index1] = pop[i][index2];
      pop[i][index2] = temp;
    }
  }
}
```

Parallel Mutation

```
void* mutation(void *slice){
…
// For a subset of the population
for(i = start; i != end; i++){
  // If a random percent chance occurs
  if((rand_r(&seed)%100) <=
      MUTATION_CHANCE){
    // Select two random idx and swap
    int index1 = rand_r(&seed)%NUM_CITIES;
    int index2 = rand_r(&seed)%NUM_CITIES;
      int temp = pop[i][index1];
      pop[i][index1] = pop[i][index2];
      pop[i][index2] = temp;
    }
  }
}
```

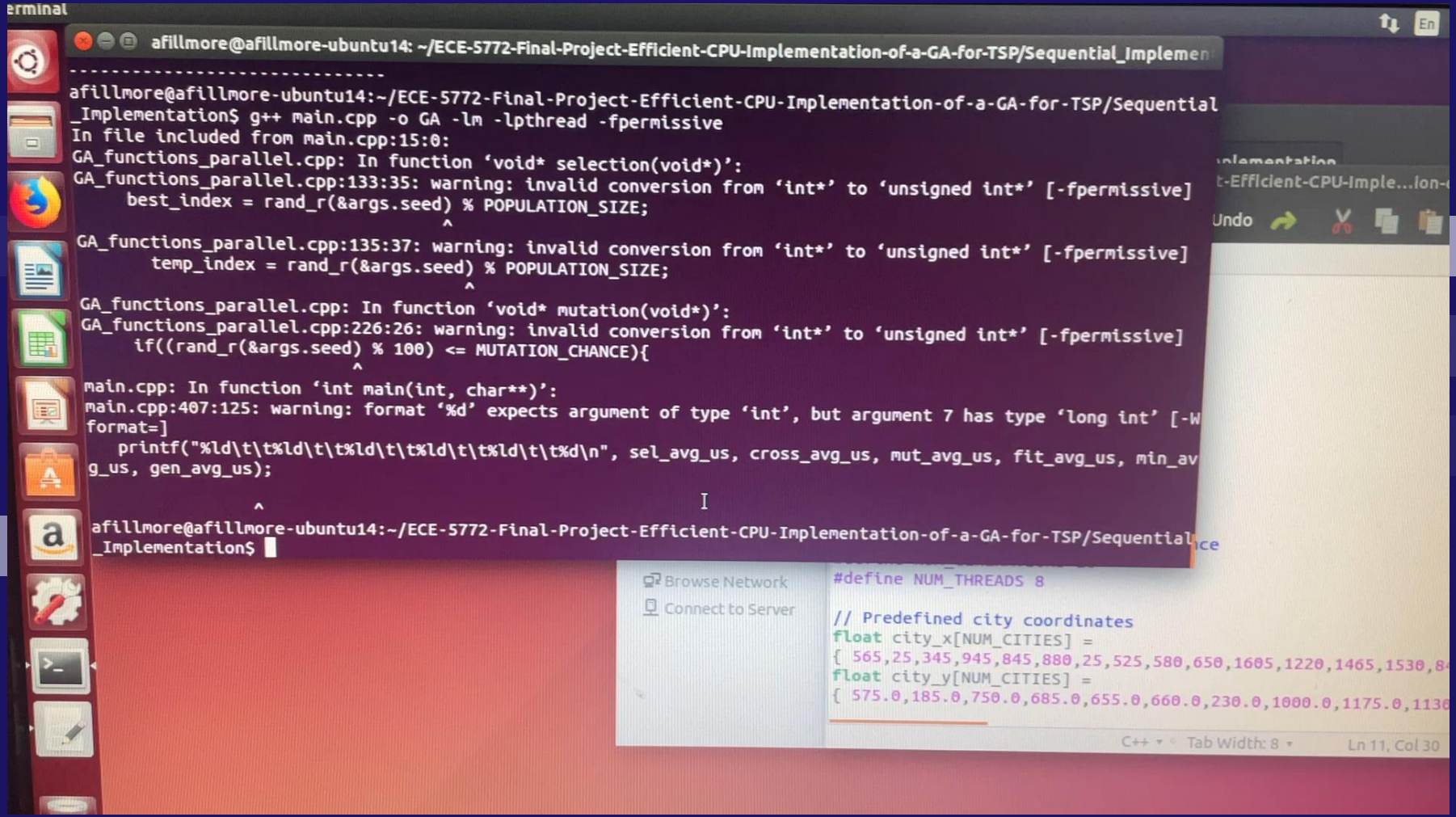

Start
Initialize Population
Evaluate Fitness
Selection
Crossover
Mutation
Stop Criteria Met?
No
Yes
Stop

NUM_THREADS = 4



03

Results

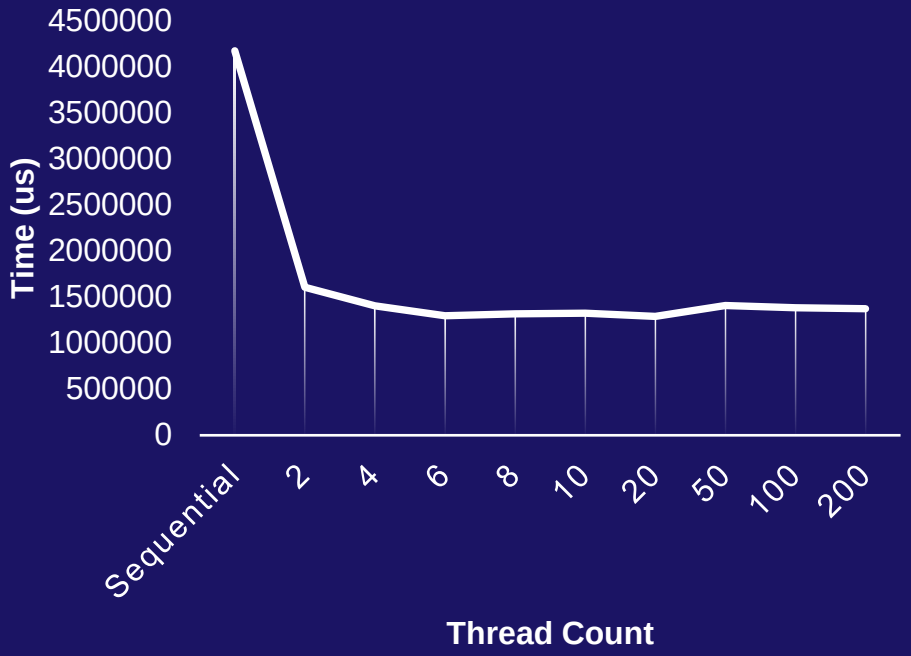

afillmore@afillmore-ubuntu14: ~/ECE-5772-Final-Project-Efficient-CPU-Implementation-of-a-GA-for-TSP/Sequential_Implemen
afillmore@afillmore-ubuntu14:~/ECE-5772-Final-Project-Efficient-CPU-Implementation-of-a-GA-for-TSP/Sequential_Implementation$ g++ main.cpp -o GA -lm -lpthread -fpermissive
In file included from main.cpp:15:0:
GA_functions_parallel.cpp: In function 'void* selection(void*)':
GA_functions_parallel.cpp:133:35: warning: invalid conversion from 'int*' to 'unsigned int*' [-fpermissive]
best_index = rand_r(&args.seed) % POPULATION_SIZE;
^
GA_functions_parallel.cpp:135:37: warning: invalid conversion from 'int*' to 'unsigned int*' [-fpermissive]
temp_index = rand_r(&args.seed) % POPULATION_SIZE;
^
GA_functions_parallel.cpp: In function 'void* mutation(void*)':
GA_functions_parallel.cpp:226:26: warning: invalid conversion from 'int*' to 'unsigned int*' [-fpermissive]
if((rand_r(&args.seed) % 100) <= MUTATION_CHANCE){
^
main.cpp: In function 'int main(int, char**)':
main.cpp:407:125: warning: format '%d' expects argument of type 'int', but argument 7 has type 'long int' [-Wformat=]
printf("%ld\t\t%ld\t\t%ld\t\t%ld\t\t%ld\t\t%d\n", sel_avg_us, cross_avg_us, mut_avg_us, fit_avg_us, min_avg_us, gen_avg_us);
^
afillmore@afillmore-ubuntu14:~/ECE-5772-Final-Project-Efficient-CPU-Implementation-of-a-GA-for-TSP/Sequential_Implementation$
Undo
Browse Network
Connect to Server
#define NUM_THREADS 8
// Predefined city coordinates
float city_x[NUM_CITIES] =
{ 565,25,345,945,845,880,25,525,580,650,1605,1220,1465,1530,
float city_y[NUM_CITIES] =
{ 575.0,185.0,750.0,685.0,655.0,660.0,230.0,1000.0,1175.0,
C++ Tab Width: 8 Ln 11, Col 30

Results

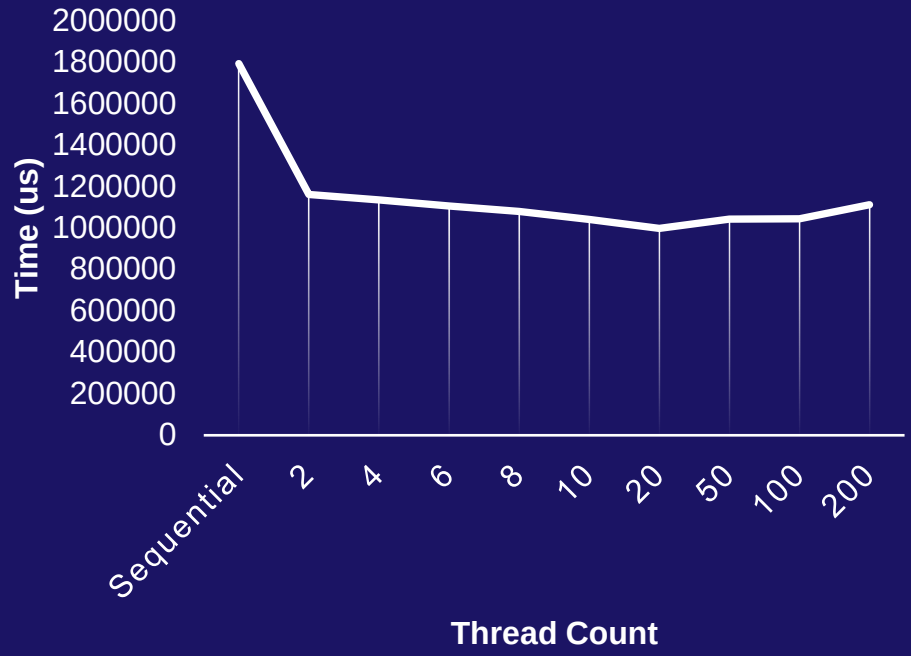
Fixed Parameters	
TOURNAMENT_SIZE	128
MUTATION_CHANCE	10%
NUM_GENERATIONS	10
POPULATION_SIZE	100000

Function	NUM_THREADS (Processing time in us, averaged across 10 generations)									
	Sequential	2	4	6	8	10	20	50	100	200
Selection	4169605	1605586	1402526	1296798	1315613	1324419	1288261	1406318	1382768	1371027
Crossover	1791997	1160138	1134223	1104464	1078718	1040005	996470	1042226	1043087	1111122
Mutation	16431	12886	12939	14440	15554	18645	18987	16613	28138	42932
Cost Update	284874	177117	162433	158543	159155	156719	167272	169064	174868	199817
Minimum Cost	1773	2777	4825	3704	5463	4356	8761	18059	23014	38039
Generation Total	6301773	2902777	2704825	2603704	2605463	2604356	2508761	2718059	2623014	2738039
Percent Speedup	N/A	54%	57%	59%	59%	59%	60%	57%	58%	57%

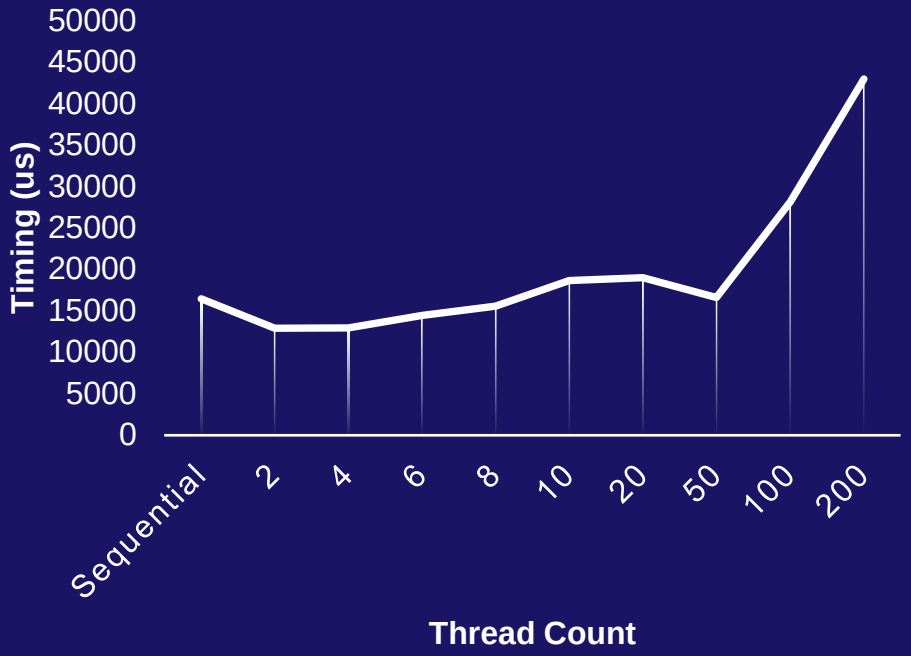
SELECTION TIMING



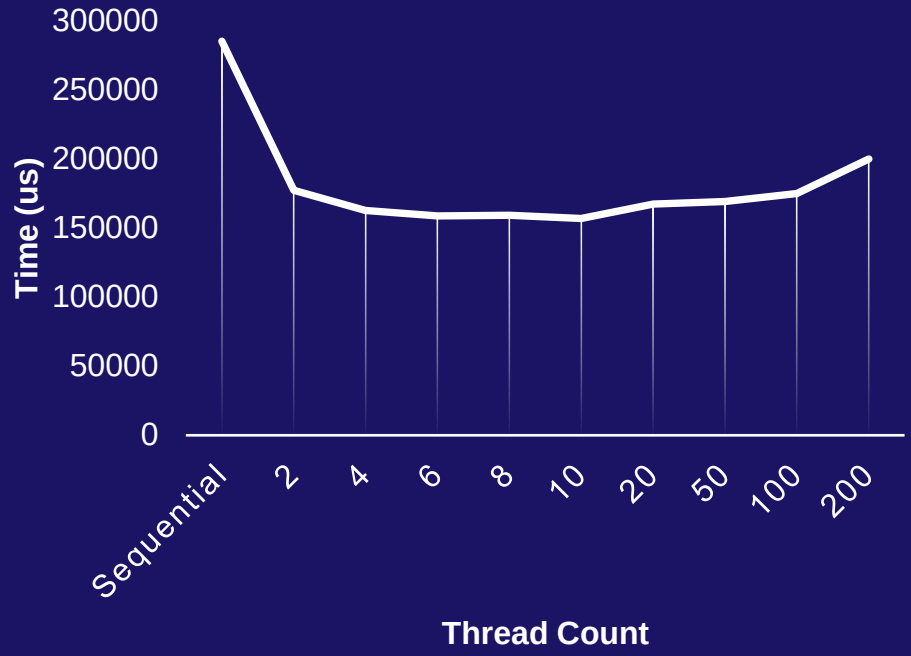
CROSSOVER TIMING



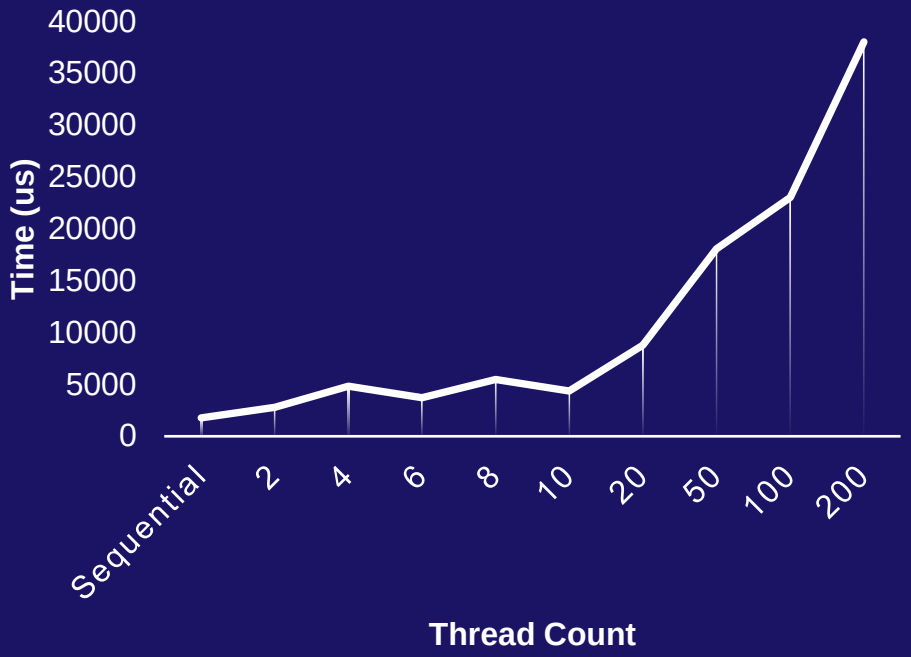
MUTATION TIMING



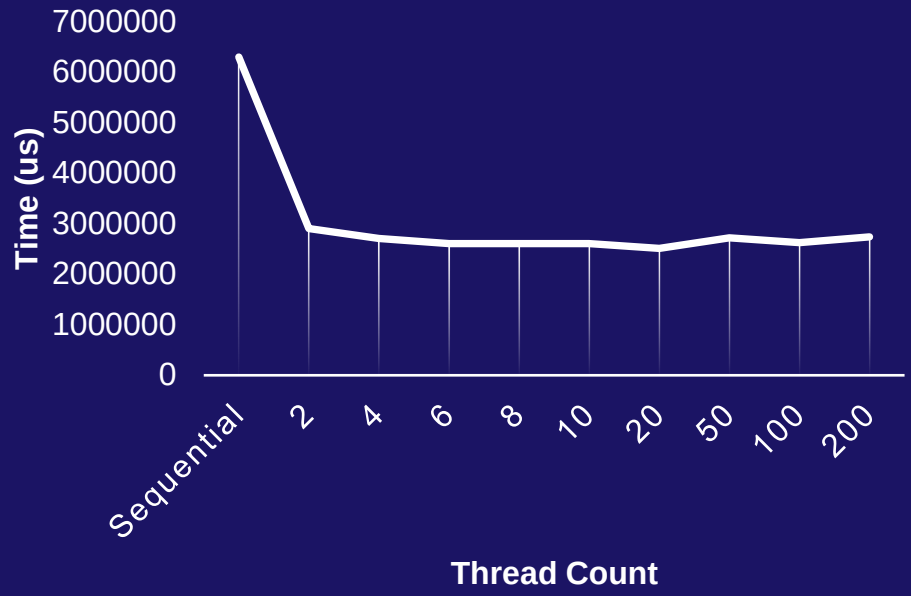
COST_UPDATE TIMING



MINIMUM COST



TOTAL GENERATION TIMING



04

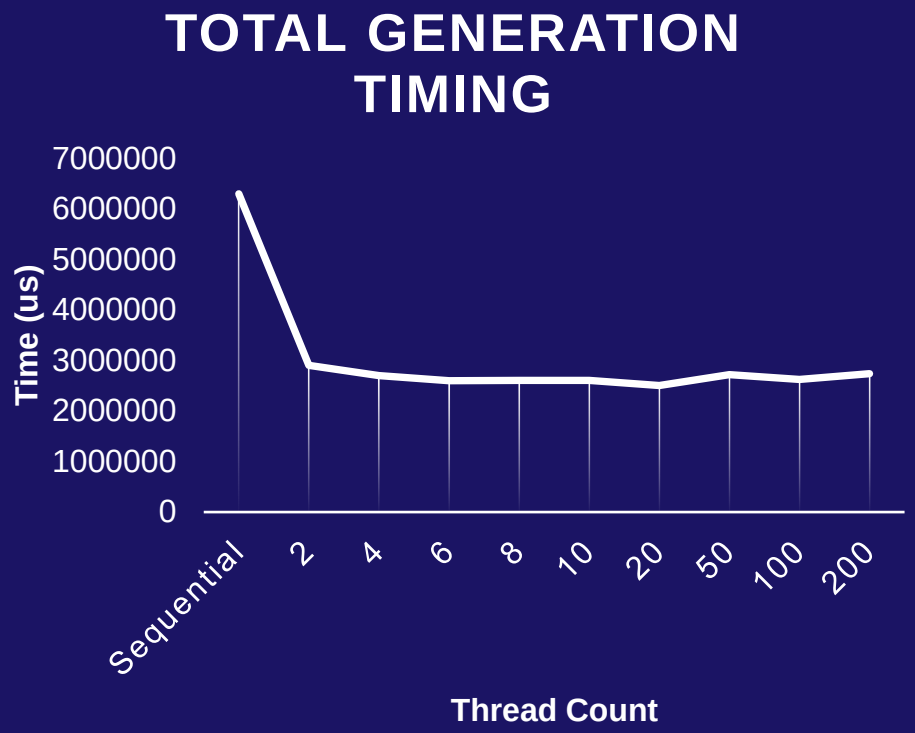
Conclusions

Conclusion

Just because you can parallelize, doesn't mean you should.

The parallelization strategy should consider the target hardware.

Ultimately, successfully sped up the algorithm with a 60% reduction in time.



THANKS!

Do you have any questions?