Saleh Afzoon

parallel algorithms -hw5
June 2020

GA optimization wit cuda

- ## Problem description

The goal it to find the target string (HELLO WOLRD) from possible character combinations.

- ## GA specification

  - **Representation:**

    Combination of capital English alphabet:

    chromosome = EQNDM UDGGD

  - **Fitness**

    String compare of chromosome string with target string.

  - **Xover**

    Uniform xOver : by 50% probability each gene choose from either first or second parent.

```
if (rand()%10 < 5){
child_chor1[i] = p1->chromosome[i];
child_chor2[i] = p2->chromosome[i];
}
else{
child_chor2[i] = p1->chromosome[i];
child_chor1[i] = p2->chromosome[i];
}
```

  - **Mutation**

    Switch mutation: one randomly chromosome position value replace with random alphabet.

  - **Parent selection**

    Tournament selection with size 5: best of randomly 5 chosen individual selected.

  - **Survivor selection**

    Elitism mode with 10%: 10% of current generation goes directly to next generation to improve GA memory history.

- ## Running and profiling command:

- Running

  Parameters: population size, parallel mode
  Like for serial mode with size 1000:

```
./ga_out 1000 0
```

- Profiling

```
nvprof ./ga_out 10000 1
```

- GA results:

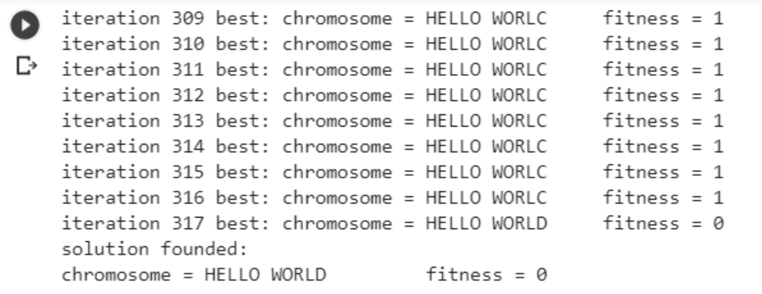
MUTATION_RATE = 0.1

MAX_GENERATION = 300

TOURNAMENT_SIZE = 5

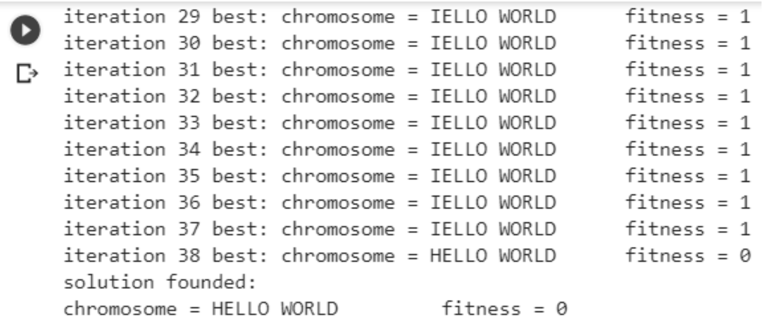
XOVER_METHOD = UNIFORM

- Population = 100



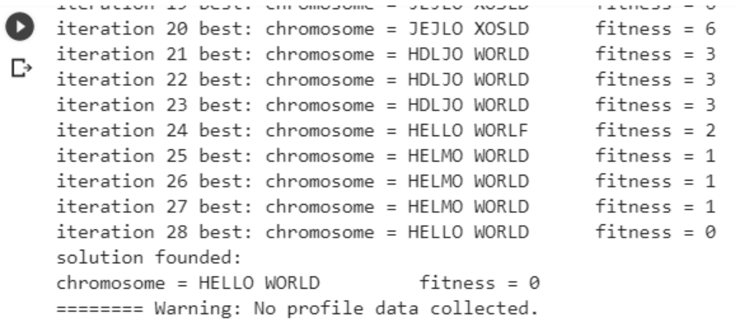
```
iteration 309 best: chromosome = HELLO WORLC     fitness = 1
iteration 310 best: chromosome = HELLO WORLC     fitness = 1
iteration 311 best: chromosome = HELLO WORLC     fitness = 1
iteration 312 best: chromosome = HELLO WORLC     fitness = 1
iteration 313 best: chromosome = HELLO WORLC     fitness = 1
iteration 314 best: chromosome = HELLO WORLC     fitness = 1
iteration 315 best: chromosome = HELLO WORLC     fitness = 1
iteration 316 best: chromosome = HELLO WORLC     fitness = 1
iteration 317 best: chromosome = HELLO WORLD     fitness = 0
solution founded:
chromosome = HELLO WORLD        fitness = 0
```

- Population = 500



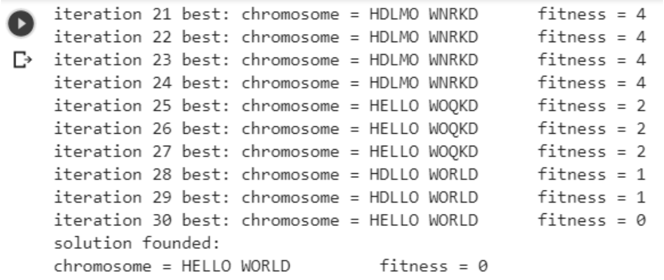
```
iteration 29 best: chromosome = IELLO WORLD      fitness = 1
iteration 30 best: chromosome = IELLO WORLD      fitness = 1
iteration 31 best: chromosome = IELLO WORLD      fitness = 1
iteration 32 best: chromosome = IELLO WORLD      fitness = 1
iteration 33 best: chromosome = IELLO WORLD      fitness = 1
iteration 34 best: chromosome = IELLO WORLD      fitness = 1
iteration 35 best: chromosome = IELLO WORLD      fitness = 1
iteration 36 best: chromosome = IELLO WORLD      fitness = 1
iteration 37 best: chromosome = IELLO WORLD      fitness = 1
iteration 38 best: chromosome = HELLO WORLD      fitness = 0
solution founded:
chromosome = HELLO WORLD        fitness = 0
```

- Population = 1000



```
iteration 19 best: chromosome = JEJLO XOSLD      fitness = 6
iteration 20 best: chromosome = JEJLO XOSLD      fitness = 6
iteration 21 best: chromosome = HDLJO WORLD      fitness = 3
iteration 22 best: chromosome = HDLJO WORLD      fitness = 3
iteration 23 best: chromosome = HDLJO WORLD      fitness = 3
iteration 24 best: chromosome = HELLO WORLF      fitness = 2
iteration 25 best: chromosome = HELMO WORLD      fitness = 1
iteration 26 best: chromosome = HELMO WORLD      fitness = 1
iteration 27 best: chromosome = HELMO WORLD      fitness = 1
iteration 28 best: chromosome = HELLO WORLD      fitness = 0
solution founded:
chromosome = HELLO WORLD         fitness = 0
======== Warning: No profile data collected.
```

- Population = 10000



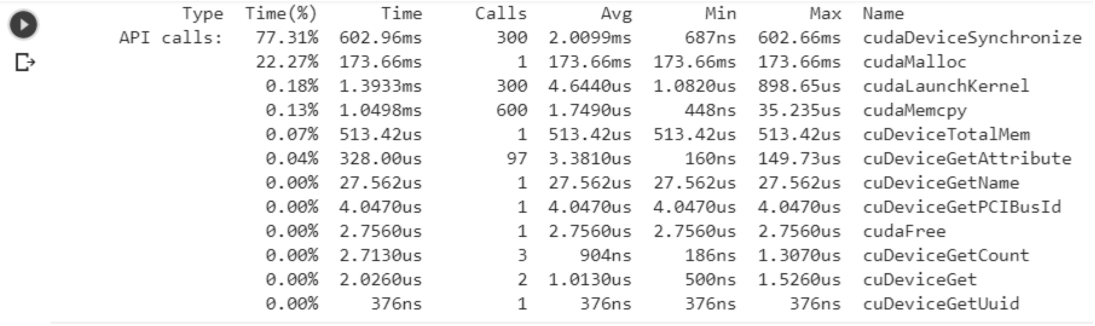
```
iteration 21 best: chromosome = HDLMO WNRKD      fitness = 4
iteration 22 best: chromosome = HDLMO WNRKD      fitness = 4
iteration 23 best: chromosome = HDLMO WNRKD      fitness = 4
iteration 24 best: chromosome = HDLMO WNRKD      fitness = 4
iteration 25 best: chromosome = HELLO WOQKD      fitness = 2
iteration 26 best: chromosome = HELLO WOQKD      fitness = 2
iteration 27 best: chromosome = HELLO WOQKD      fitness = 2
iteration 28 best: chromosome = HDLLO WORLD      fitness = 1
iteration 29 best: chromosome = HDLLO WORLD      fitness = 1
iteration 30 best: chromosome = HELLO WORLD      fitness = 0
solution founded:
chromosome = HELLO WORLD         fitness = 0
```

- GA parallelism:

For parent and survivor selection we need all population so it could not be parallel.
But fitness evaluation is a good choice of parallelism so we do as below:

```
if(parallel){

        cudaMemcpy( dev_pop, next_pop, pop_size * sizeof(Individual*),
        cudaMemcpyHostToDevice );

        parallel_eval<<< pop_size/numThread , numThread >>>(dev_pop,pop_size,
        target);

        cudaDeviceSynchronize();

        cudaMemcpy( next_pop, dev_pop, pop_size * sizeof(Individual*),
        cudaMemcpyDeviceToHost );

}
```

- Profiling data:



```
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 API calls:   77.31%  602.96ms       300  2.0099ms     687ns  602.66ms  cudaDeviceSynchronize
              22.27%  173.66ms         1  173.66ms  173.66ms  173.66ms  cudaMalloc
               0.18%  1.3933ms       300  4.6440us  1.0820us  898.65us  cudaLaunchKernel
               0.13%  1.0498ms       600  1.7490us     448ns  35.235us  cudaMemcpy
               0.07%  513.42us         1  513.42us  513.42us  513.42us  cuDeviceTotalMem
               0.04%  328.00us        97  3.3810us     160ns  149.73us  cuDeviceGetAttribute
               0.00%  27.562us         1  27.562us  27.562us  27.562us  cuDeviceGetName
               0.00%  4.0470us         1  4.0470us  4.0470us  4.0470us  cuDeviceGetPCIBusId
               0.00%  2.7560us         1  2.7560us  2.7560us  2.7560us  cudaFree
               0.00%  2.7130us         3     904ns     186ns  1.3070us  cuDeviceGetCount
               0.00%  2.0260us         2  1.0130us     500ns  1.5260us  cuDeviceGet
               0.00%     376ns         1     376ns     376ns     376ns  cuDeviceGetUuid
```

As the profiling data shows most of algorithm time spent for threads process synchronization (like before parent and survivor selection and population fitness sorting).