UHA **UFAZ (MDSAI)**

Mini project : A CUDA impelementation of differential evolution algorithm to optimize Rastrigin function

---

Continuous Optimization

Continuous optimization means finding the minimum or maximum value of a function of one or many real variables, subject to constraints. There are numerous well-known functions to be optimized such as Rastrigin, Rosenbrock and Griewank functions. In order to find the minimum value of these functions, we apply approximative techniques called metaheuristics, which are straightforward, easy to understand and to implement. The follower of the state-of-the-art of optimization field can find hunderds of metaheuristics being proposed. However, for this mini-project, you are asked to implement one well-known metaheuristic called : Differential evolution.

Differential evolution algorithm (DE)

DE is a population-based stochastic search algorithm well suited for continuous optimization. The search process of DE relies on three search operators : mutation, crossover and selection that are applied on the population (the population is a set of solutions). Supposing that the solution vector containing D variables, $x_i^G= [x_{i,1}^G,x_{i,2}^G, ..., x_{i,D}^G]$ represents the solution i at generation G, the procedure of DE can be represented in four phases :

1. Initialization : the mutation parameter F (real value), the crossover parameter CR (real value), the population size NP (integer value) and the number of generations G_{max} (integer value) are set. The starting population of individuals are randomly initialized.
2. Mutation : for each individual x_i^G in the parent population, a mutant vector v_i^{G+1} is calculated as follows :
$$v_i^{G+1} = x_{r1}^G + F.(x_{r2}^G - x_{r3}^G) \tag{1}$$
where $r1$, $r2$ and $r3$ are different randomly generated numbers within the range 1, ..., NP and different from the index i.
3. Crossover : binomial crossover is the basic search operator of DE, where for each individual x_i^G and for each dimension j, a trial vector u_i^{G+1} is generated as follows :
$$u_{i,j}^{G+1} = \begin{cases} v_{i,j}^{G+1} & \text{if } j = \sigma_j \quad \text{ or } R_j < CR \\ x_{i,j}^G & otherwise \end{cases} \tag{2}$$
where σ_j and R_j are randomly generated numbers within the ranges 1, ..., D and [0,1] respectively. It should be mentioned that other operators, such as the exponential crossover, may be also applied.

4. Evaluation : for each individual x_i^G, the trial vector is evaluated and it replaces the parent individual if it has a better fitness.
5. G is incremented and phases 2 to 5 are repeated while G is less than G_{max}.

A source code of differential evolution solving the Rastrigin function in C++ can be found on : https ://github.com/milsto/differential-evolution

Rastrigin function

This function is a minimization mathematical function used to test metaheuristics performance. It is written as follows :

$$f(x) = A * n + \sum_{i=1}^{n} [x_i^2 - A * cos(2 * \pi * x_i)] \tag{3}$$

where A = 10 and $x_i \in$ [-5.12, 5.12]. It has a global minimum value x = 0 where f(x) = 0.

CUDA implementation of differential evolution algorithm

Based on your knowledge about kernels and libraries such as : Thrust and cuRAND, and the brief description of DE and the rastigin function, implement a CUDA version of DE solving the function at hand.