Alpinist CellularDE: A Cellular based Optimization Algorithm for Dynamic Environments

Vahid Noroozi  
Dept. of Computer Eng. and IT  
Amirkabir University of Technology  
Tehran, Iran  
vnoroozi@aut.ac.ir

Ali B. Hashemi  
Dept. of Electrical and Computer Eng.  
University of Toronto  
Toronto, Canada  
alli.b.hashemi@utoronto.ca

Mohammad Reza Meybodi  
Dept. of Computer Eng. and IT  
Amirkabir University of Technology  
Tehran, Iran  
mmyebodi@aut.ac.ir

ABSTRACT
In this paper, we propose Alpinist CellularDE to address dynamic optimization problems. Alpinist CellularDE tries to detect different regions of the landscape and uses this information to perform more effective search and increase its performance. Moreover, in Alpinist CellularDE a directed local search is proposed to track local optima after detecting a change in the environment. The proposed algorithm is evaluated on various dynamic environments, modeled by Moving Peaks Benchmark. Experiments show superior performance of Alpinist CellularDE in all test cases in comparison with some of the best performing evolutionary algorithms for dynamic optimization problems.

Categories and Subject Descriptors
G.1.6 [Numerical Analysis]: Optimization – global optimization;  
I.2.8 [Artificial Intelligence]: Search – Heuristic methods

General Terms
Algorithms

Keywords
Dynamic Environments, Evolutionary Algorithms, Cellular Automata, Differential Evolution, Moving Peaks Benchmarks, CellularDE

1. INTRODUCTION
There are many optimization problems that are dynamic in which the fitness function of the problem changes over time [1]. One of the algorithms which are developed for DOPs is CellularDE that has shown a good performance in various tested dynamic environments [2].

In this paper, a modified version of CellularDE, named Alpinist CellularDE, is proposed. In the following, first CellularDE is introduced briefly in section 2. Then, the proposed approach is presented in Section 3. In Section 4, the settings of the experiments with the experimental results and relevant analysis are presented. Finally, section 5 concludes the paper.

2. CellularDE
CellularDE, inspired by Cellular PSO [3, 4], is a cellular based algorithm which utilizes Differential Evolution (DE) as the main search algorithm and uses Cellular Automata (CA) to control the search process. It utilizes CA to partition the search space into subspaces, known as cells, and to control the search process in each cell locally. The evolution of individuals in each cell is performed locally by DE. This helps the algorithm to maintain diversity meanwhile it can exploit the local optima appropriately.

3. PROPOSED ALGORITHM: ALPINIST CellularDE
In Alpinist CellularDE, each cell estimates the presence of a peak within its boundary or its neighbors’ and adopts its search strategy based on that. Moreover, to track the peaks more efficiently after detecting a change in the environment, Alpinist CellularDE utilizes a proposed directed local search instead of the random local search performed in CellularDE. The proposed algorithm works as follows.

First, for an environment with D dimensions, cellular automata with \( N_p \) cells are embedded into the search space. Then, \( M \) individuals are randomly initialized in the cells of the cellular automata. At each iteration, if there is at least one individual in cell \( i \), the cell checks to update the fittest location found in its boundary \((c_{Best})\). Then, local best memory of cell \( i \) \((N_{Best})\) is updated to contain the best position found in cell \( i \) and its neighbors. Afterwards, cell \( i \) determines its type based on which the search strategy of the individuals in cell \( i \) is selected and applied. Each cell determines its type as follows.

In Alpinist CellularDE a cell can be either of these types: PEAK, SLOPE, PLATUE, or UNKNOWN. Initially all type of cells are UNKNOWN. A PEAK cell is a cell which probably contains a local optimum. A cell \( i \) is a PEAK cell if the fitness of the best position found in cell \( i \) \((c_{Best})\) improves less than \( \phi \) during the last \( t_u \) updates, and at the same time the local best memory of cell \( i \) \((c_{Best})\) is located at cell \( i \), i.e. \( c_{Best} = N_{Best} \). A SLOPE cell is a cell adjacent to a PEAK cell. Hence, all neighbors of a PEAK cell are considered SLOPE cells. In PLATUE cells \( N_{Best} \) is located on a neighboring SLOPE cell.

Since PEAK cells are the most promising cells to contain local optima, in these cells the search process is performed through hill climbing around the \( N_{Best} \) instead of DE algorithm. This helps faster and more efficient convergence to the local optima.

Although it is possible that SLOPE and PLATUE cells contain a better local optimum that their neighbor PEAK cell, it is more efficient to perform the search process in these cells with less
individuals. Hence, in SLOPE and PLATUE cells, the capacity of cells, i.e. the maximum number of individuals allowed to search in a cell, is decreased to half.

Individuals in cell $i$, which is not a PEAK cell, search using DE algorithm. The mutant vector for an individual $k$ with the position vector $x_k$ which resides in cell $i$ is calculated by eq. (1).

$$v_k = NBest_i + \lambda_{rand}(x_{k1} - x_1) + \lambda_{rand}(x_{k2} - x_1)$$  (1)

where $x_1$ and $x_2$ are the position vectors of three random selected individuals in the cell $i$. $\lambda_{rand}$ and $\lambda_{rand}$ are uniform random variables in [0,1]. Moreover, to improve exploration capability of the individuals, after calculating the mutant vector, a normal noise with the probability of 0.25 is added to a randomly selected dimension.

3.1 Dealing with the Change

After a change is detected in the environment, in order to track previously found local optima more effectively, a directed local search, similar to a directed search introduced in [7], is proposed to substitute the simple random local search of CellularDE. In the proposed local search, the best memory of a cell is considered as the base point to start the local search as follows.

For cell $i$, a binary direction vector $dir_i^{(n)} = (dir_1^{(n)}, dir_2^{(n)}, ..., dir_D^{(n)})$, $dir_j \in [-1,1]$, is defined which specifies the direction of the search in each dimension. This vector is initialized randomly at the beginning of the local search, i.e. after detecting a change in the environment.

In each iteration, a random point around $cBest_i$ is considered such that it only differs from $cBest_i$ in the $j$th dimension. The location of the selected random point is calculated using eq. (2).

$$x_{ij} = cBest_i + h(a) \cdot \text{ndrs}_i \cdot dir_{ij}$$  (2)

where $a$ is the initial step of random search, $h$ is a constant in (0,1) which controls the speed of random search convergence. $\text{ndrs}_{ij}$ is the number of unsuccessful random search performed on dimension $j$ of cell $i$. If the fitness of the new position is better than the fitness of $cBest$, $cBest$ is replaced by the position. Otherwise, the search is performed on the opposite direction on dimension $j$. If the search in the new direction for dimension $j$ is not successful too, $\text{ndrs}_{ij}$ increases by one. The above procedure is performed for all dimensions until $\text{ndrs}_{ij}$ for all dimensions reaches its limit ($\text{ndrs}^*$).

4. EXPERIMENTS

For all the experiments, Moving Peaks Benchmark (MPB) [5] is adopted as the dynamic benchmark function [1]. Experiments are conducted on the well-known scenario II of MPB with the default settings described in [5, 3, 2]. Size of cellular automata, initial step size ($\alpha$), reduction factor ($\beta$), and $\text{ndrs}$ are empirically set to $10^3$, 10, 0.2, and 4, respectively. The size of history of best memory if cell ($Cr$) and fitness improvement threshold ($\phi$) is set to 6 and 0.1, respectively. Parameter $Cr$ of DE is set to 0.7. Other parameters are set to values reported in [2].

The proposed algorithm is compared with CellularDE [2] and Adaptive $m$QSO [6]. For all experiments, parameters of CellularDE and Adaptive $m$QSO are set to the values reported in [2] and [3], respectively. All algorithms are experimented for 500,000 fitness evaluations and the average offline errors of the algorithms in 100 runs with 95% confidence interval are reported in Table 1. Experiments are performed on dynamic environments with different frequency of change, $f^\prime$={1000, 2500, 5000}, and different number of peaks, $m$={10, 50, 100}. As depicted, Alpinist CellularDE outperforms other tested algorithms in all cases. This is due to the efficient use of fitness evaluations by increasing the exploitation power of the algorithm around the local optima by replacing DE with hill climbing in PEAK cells, and utilizing the proposed directed local search which results in faster tracking of the peaks after detecting a change in the environment.

<table>
<thead>
<tr>
<th>$m$</th>
<th>$f^\prime$</th>
<th>Offline error</th>
<th>Adaptive $m$QSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1000</td>
<td>3.43±0.04</td>
<td>4.11±0.08</td>
</tr>
<tr>
<td>50</td>
<td>2500</td>
<td>4.14±0.03</td>
<td>2.38±0.02</td>
</tr>
<tr>
<td>100</td>
<td>5000</td>
<td>2.04±0.04</td>
<td>2.28±0.02</td>
</tr>
</tbody>
</table>

5. CONCLUSION

In this paper we proposed Alpinist CellularDE to address dynamic optimization problems. In Alpinist CellularDE, a proposed DE scheme which is suitable for dynamic environments is used. Moreover, the search process in each cell is performed differently based on its location and estimated position of local optima. In addition, the random local search in CellularDE is replaced by a proposed directed local search to improve the efficiency of tracking local optima after detecting a change in the environment. Experiments on various dynamic environments modeled by Moving Peaks Benchmark shows that Alpinist CellularDE outperforms other tested algorithm in all cases.

6. REFERENCES


