International Journal of Advanced Science and Technology
Vol. 29, No. 7s, (2020), pp. 2765-2769

Optimized Resource Allocation Using Machine Learning Techniques

Tamil Selvi K, Aparna.K.S, Jayanthi.P

Department of Computer Science and Engineering, Kongu Engineering College.

Abstract

Resource allocation in any computing environment provides scope for optimization. To allocate hardware in a particular organization (or) institution in an efficient way, refined grasshopper optimization algorithm is used. In grasshopper optimization algorithm, randomness is not achieved properly. To achieve the randomness, Levy search algorithm is used. Levy search algorithm is the random search algorithm where the probability distribution of a random search walk and it is used to provide random factor efficiently. This result is compared with the results obtained from cuckoo search algorithm over the requests. The experimental result shows that Improved Grasshopper optimization algorithm gives the best result.

1. Introduction

Hardware Request management is the process of allocating requested hardware to their respective employees. The employees in a particular organization may have a need for a particular hardware to complete their work efficiently and productively. In order to fulfill their hardware needs, the hardware handling department needs a process flow which satisfies all the requestors with the available hardware. The hardware handling department needs a greater effort to allot the requestors with the hardware and buying the hardware that are at a greater requirement. To request a particular hardware, the requestor needs to submit the form with the hardware details needed and the process begins.

Efficient scheduling of the resources is a must in allocating hardware to the requestors, so achieve that we need a proper algorithm. To incorporate an efficient algorithm, swarm intelligence would be a best idea to implement it.

Swarm intelligence (SI) is the behavior of any natural or artificial system in a group which would help us in solving the real-time problems. The various natural behaviors is observed and made into algorithm that can be used further. Examples of swarm behavior are searching food by ants (ant colony optimization), bird flocking (bird flocking algorithm), animal herding (animal herding algorithm) and many more.

Here in order to allot hardware to the requestors, we use the various swarm behavior like grasshopper optimization algorithm. To make it as efficient one, we can implement the refined grasshopper optimization algorithm. Grasshopper optimization algorithm (GOA) is the process of finding the best solution of each grasshopper in the swarm within certain limits. The grasshopper optimization algorithm can be used in solving various applications in the field of resource management, task scheduling, etc. Using GOA, initial population is done with the help of general random function. The regular random function would place the search agents in a determinate position that it will not be so efficient.

To improve the efficiency of creating the randomness, the randomization of the requests is done using Levy flight pattern. This pattern is a random walk search strategy used by a wide variety of organisms when searching for heterogeneously distributed food. And also to avoid obtaining local optimum in a rapid manner which is a disadvantage in GOA. This strategy is of getting an optimal new position better than the initial one.

In the process of allotting the hardware to the requestors, action team can use the refined grasshopper optimization algorithm where the requests are randomly mapped with the available resources. Then the

ISSN: 2005-4238 IJAST

fitness function is calculated with the current allocation and if the fitness function gives a better result then the mapping is changed. This process takes place many times and finally a convergence value is obtained. This mapping would give us an efficient allocation of hardware than doing it manually.

2. Grasshopper Optimization Algorithm:

This simulates the nature behavior of the grasshopper swarm, which migrates to a long distance for finding their food. The social interaction among grasshoppers, wind force and gravity outside the swarm influences the trajectory of the grasshoppers. In the exploration stage, the swarm are moving rapidly and abruptly to find more potential targets, whereas in the exploitation stage, they tend to move locally to reach the better target.

The position of each grasshopper in a swarm is formulated as shown in equation 2.1 :

$$X_i = S_i + G_i + A_i \tag{2.1}$$

Where the position of the grasshopper i is X_i with the social interaction S_i and the gravity force on the grasshopper is G_i, and Ai defines the wind factor. Social interaction for the individual grasshopper can be calculated as in equation 2.2.

$$S_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} s(d_{ij})\,\widehat{d_{ij}} \tag{2.2}$$

where dij =| xj - xi | which gives the distance between the i-th and j-th grasshopper in the swarm. The distance between the grasshopper i and j is given by

$$\widehat{d_{ij}} = \frac{x_j - x_i}{d_{ij}} \tag{2.3}$$

In equation 2.2, the factor s is gives the social factors within the swarm

$$s(r) = fe^{\frac{-r}{l}} - e^{-r} \tag{2.4}$$

where f is the intensity of attraction and l is the length of the attraction scale. Here, values 1.5 and 0.5 are chosen for l and f respectively. The gravitational force Gi is

(2.5)

$$G_i = -g\widehat{e_g}$$

where g is the constant and the value of e_g shows the value towards the center of the earth. The attraction force is defined as

(2.6)

$$A_i = u\widehat{e_w}$$

where u is a drift constant and e_w shows the direction of wind.

Substituting S, G and A into the equation (2.1)

$$X_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} s\left(\left|x_j - x_i\right|\right)\frac{x_j - x_i}{d_{ij}} - g\widehat{e_g} + u\widehat{e_w} \tag{2.7}$$

where $s(r) = fe^{\frac{-r}{l}} - e^{-r}$ and N is the number of grasshoppers.

ISSN: 2005-4238 IJAST

The equation 2.7 cannot be directly used for solving optimization problems, since the swarm will attain local minima/maxima quickly with vanishing gradient. The parameters Gi and Ai which can be replaced by the parameter food target. The modified equation is,

$$X_i^d = c\left(\sum_{\substack{j=1 \\ j\neq i}}^{N} c\frac{ub_d - lb_d}{2} s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right) + \widehat{T_d} \tag{2.8}$$

where the upper bound value, lower bound value, the target from the best solution found so far for the D^{th} dimension is given by u_{bd} , $l_{bd,}$ and $T_{\underline{d}}$ respectively an c is a swarm constant random value.

Exploration : It is the process of searching new individuals far from the current individual grasshopper in a swarm behaviour in the search space.

Exploitation : It is the process of searching the individuals within closer distance, something like local search. The coefficient 'c' is the value of the reduction in comfort zone and is given by

$$c = cmax - l\frac{cmax - cmin}{L} \tag{2.9}$$

where cmax is the upper bound value of the comfort zone, cmin is the lower bound value of comfort zone, current iteration, and the maximum number of iterations are l and L. Here, values used are cmax =1, cmin=0.00001, L=100 iterations.

In each iteration, the best fit individual grasshopper is assumed to the best solution and needs all other grasshopper to move towards it.

The Grasshopper Optimization algorithm is given as below:

1) Initialize the particles of grasshopper, Xi
2) Initialize the values for maximum value (Cmax), minimum value (Cmin) and Maximum no. of iterations (L).
3) Calculate the target fitness value for each grasshopper
4) T= the best search agent
5) While (l<L)
    c = c + 1
      for each search agent
            calculate the distances between grasshoppers and normalize it
            Update the position of current search agent
            If agent is outside the target, adjust the distance appropriately
      end for
    if current solution is better,
     the update it as better solution
    increment l
  End while
6) Return Target

ISSN: 2005-4238 IJAST

3. Refined Grasshopper Optimization Algorithm:

The linear decreasing comfort zone used in the above GOA is not useful to make full utilization of every iterations. It has the variability because of the lack of random factors. It leads to lack of creativity during iterations, and every search agent could only search determinate position. It is easy to fall into the local optimum. Hence the local search can be improved by Levy Flight mechanism, which is a random search walk and is used to provide random factor efficiently. It is represented as,

$$\text{Levy}(d) = 0.01 * (a * \sigma) / (|b|^{(1/\beta)}) \quad (3.1)$$

where d – represents the dimension of the problem, a & b represents random numbers in the range (0, 1) and β is set to a value 1.5 in this application. The value of σ is found by,

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times 2\left(\frac{\beta-1}{2}\right)} \right)^{\frac{1}{\beta}} \quad (3.2)$$

where Γ(x) =(x-1)!.

4. Cuckoo search algorithm:

Cuckoo search algorithm is the searching mechanism where the cuckoo lays its egg in the nest of other birds. The behavior of cuckoo is to choose a random nest where the quantity of nest is fixed. Here the other bird which lives in the nest is referred to as host bird. The probability of the host bird finding the cuckoo egg is in the Ph ∈ [0, 1]. This decides that the host moves to other nest or throws cuckoo's egg from its nest. Here the solution to this problem is by finding the number of eggs in the nest and checking whether the cuckoo's egg can be placed in it or not. By solving, efficient solution is replaced with the existing old one. Here to find the global optimum Levy search is used.

$$X_i^{t+1} = X_i + \alpha \oplus Lévy(\lambda) \quad (4.1)$$

Where $X^{t+1}{}_i$ is the solution obtained at t+1 iteration, X_i is the current solution and α is the parameter

The cuckoo search algorithm is as follows:

1) Initialize position of cuckoo, max_population, max_iteration
2) While (max_iteration < max_population)
    Find used parameters of swarm particles
    Evaluate the parameters obtained for the particle
    Evaluate P_{best}
    Assign P_{best} to L_{best}
    Evaluate the best nest
    Update the best solution
3) Evaluate the new solution
4) Calculate the P_{best} and L_{best}
5) Assign Lbest to Gbest
6) Display ET, τw, PAR

5. Implementation:

Here N number of requests and M number of resources in a batch are assumed. Request is under any one of K types. One resource is allocated to a requestor at a time. The makespan of a resource is calculated by summing time utilized by all the requests using that particular resource. The total makespan is the longest time taken among all the resources for all the requests. The fitness is found by

ISSN: 2005-4238 IJAST

the total makespan value. Other constraints like one requestor needs various resources simultaneously, etc are ignored because of simplicity of this application. 30 search agents and 500 iterations are assumed. This algorithm is tested for 30 times and the average value is found. To know the performance of RGOA, it is compared with grasshopper optimization algorithm and cuckoo search algorithm.

6. Result:

This application was implemented to allocate the resources to the requestors in an efficient manner. For this purpose, swarm intelligence mechanism is adapted. The nature of grasshopper swarm for searching food has been used. To improve the performance of GOA, levy flight random search walk has been added in the proposed method. To know the performance of the proposed algorithm, results received in both methods are compared and shown in the following table. Finally it is obtained that comparison with the results, refined grasshopper optimization algorithm works efficient. The results of the algorithm is shown in the Table 6.1.

Average fitness: The average fitness value of all grasshoppers in a particular swarm in each iteration.

Algorithm	**Average**	**Standard Deviation**	**Best**	**Worst**
Refined GOA	13.50	0.62	12.63	14.62
GOA	14.95	0.82	13.5	16.67
CSA	11.02	0.86	15.84	19.43

Table 6.I. Results using both methods

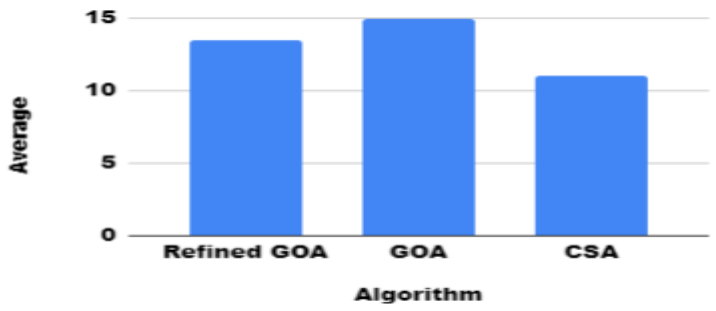

Figure 6.1 Graph based on various machine learning algorithms

References:

[1] https://en.wikipedia.org/wiki/Swarm_intelligence
[2] Shahzad Saremi, Syed Mirjalili, Andrew Lewis .;"Grasshopper Optimization Algorithm: Theory and application"
[3] Aleksei Chechkin, Ralf Metzler, Joseph Klafter, Vsevolod Yurievich Gonchar.;"Introduction to the Theory of Levy Flights"
[4] A.S. Joshi, Omar Kulkarni, G.M.Kakandikar, V.M.Nandedkar, "Cuckoo search optimization – A review"

ISSN: 2005-4238 IJAST