Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 15 (2011) 2880 – 2884

Procedia Engineering

www.elsevier.com/locate/procedia

Advanced in Control Engineering Information Science

Hybrid Artificial Glowworm Swarm Optimization Algorithm for Solving Multi-dimensional Knapsack Problem

Qiaoqiao Gong, Yongquan Zhou a*, Qifang Luo

College of Mathematics and Computer Science Guangxi University for Nationalities Nanning, Guangxi 530006, China

Abstract

In this paper, a hybrid artificially glowworm swarm optimization algorithm to solve multidimensional 0-1 knapsack problem is proposed. The algorithm utilizes two important strategies, how to select the item based on its unit volume value and the binary glowworm swarm optimization algorithm, 37 multidimensional 0-1 knapsack test instances are tested by the produced algorithm, The integrated performance of the produced algorithm is rather satisfied, and the hybrid glowworm swarm optimization algorithm is rather efficient for solving multidimensional 0-1 knapsack problem.

Keywords: *glowworm swarm, optimization algorithm, knapsack problem; greedy Algorithms; binary code*

1. Introduction

Knapsack problem is typical of combinatorial optimization problem, many complex optimization problems through a series of the knapsack sub-problem [1-2]. In recent years, some scholars at home and abroad put forward some special methods for 0-1 knapsack problems particularity. Several metaheuristics have been developed for MKP, including tabu search by Hanfi S etc in 1998.In 2002, A Plateau combined interior-point method with super metaheuristics and got a hybrid search algorithm for solving 0-1 linear programming [3].In 2005, Vasquez M, etc combined linear programming with tabu search method and got the LP + TS algorithm. Form the foregoing method, fixed some variable values and tried to get a heuristic algorithm fix + LP + TS [4] better than LP + TS. In 2005, Du Haifeng etc [5] was put forward artificial immune antibody adjusting clonally algorithm based on the cell clonally selection of the doctrine. The proposed algorithm can effectively avoid precocious problem and the rate of convergence is faster. In 2006,Chen Duanbing proposed a quasi-human heuristic algorithm and the integrated

* Corresponding author. Tel.: +86-771-3260264.
E-mail : yongquanzhou@126.com.

1877-7058 © 2011 Published by Elsevier Ltd.
doi:10.1016/j.proeng.2011.08.542

performance of the produced algorithm are rather satisfied, and the runtime is short. Some other scholars also put forward the ant colony algorithm, and also achieved good effect [6].

In 2005, Krishnanand and Ghose put forward glowworm swarm optimization (GSO) algorithm[7].The GSO algorithm is a swarm intelligence bionic algorithm and it is originated from simulating the nature glowworm swarm foraging and seeking spouses. The process of the research shows GSO is more superior than particle swarm optimization algorithm in multi-modal functions optimization. The paper puts forward a GSO algorithm adding to greedy ideal for MKP. This very effective local search method makes use of heuristic information and information so that the HGSO can accelerate convergence speed. Using the proposed algorithm, the large number of examples for practical numerical tests achieved good results.

2. Multidimensional Knapsack Problems Mathematical Model

A set of m items and a set of g resources are given. Each item j $(j=1,2,...m)$ has assigned a profit c_j and a resource consumption value a_{ij} for each resource i $(i=1,2,...g)$. The problem is to identify a subset of all items that leads to the highest total profit and does not exceed the resource upper bound b_i. The MKP can be formulated as follow:

$$\begin{cases} \max\ f(x_1, x_2, ..., x_m) = \sum_{j=1}^{n} x_j c_j \\ s.t \quad \sum_{j=1}^{n} x_j a_{ij} \le b_i, i = 1,2,.., g. \\ \quad x_j \in \{0,1\},\ j = 1,2,..., m. \end{cases} \tag{1}$$

The variable x_i is an indicator of item j, if x_i is set to 1, it means item j is selected, or 0 means item j is not selected for $j=1,2,...m$. Equation represents the total profit of selection items and the g resource constraints.

3. Glowworm Swarm Optimization and Greedy Algorithm

3.1. Glowworm Swarm Optimization Algorithm

The glowworm swarm optimization algorithm (GSO) is a kind of new method of swarm intelligence. The agents in AGSO are thought of as glowworms that carry a luminescent quantity called luciferin. Each glowworm uses an artificial equivalent to luciferin, which we will call luciferin for short in the following, to broadcast the fitness of its current location, evaluated using the objective function, to its neighbors. That is, glowworms are attracted to neighbors that glow brighter. These movement that are based only on local information and selective neighbor interactions, enable the swarm to partition into disjoint subgroups that converge to multiple optima of a given multimodal function. According to the above principles, the AGSO have used for multi-modal functions optimization, multi-modal functions with collective robotics applications [7], etc.

In the GSO algorithm, each solution correspond to a glowworm in the search space, and every glowworm has a fitness value decided by function of the optimization problem .Each glowworm encodes the objective function at the current location into a luciferin value. The glowworms depend on a variable neighborhood, which is bounded above by a radial sensor range, to identify their neighbors and compute their movements. Each glowworm, using a probabilistic mechanism, selects a neighbor that has a luciferin value higher than its own and moves toward it.

A set of n glowworms are randomly deployed in a m-dimensional workspace. According to the similarity of luciferin value, divide the swarm into nei neighbors and each glowworm i selects a neighbour j with a probability P_{ij} and moves toward it within its decision domains range R_d $(0 < R_d^i < R_s)$, where R_s is a circular sensor range of glowworm i. The position of glowworm i is x_i ($x_i \in R^m, i = 1,2,\cdots,n$), which is a potential solution. Put x_i into the objective function and gain the fitness function value $J(x_i)$ and luciferin value l_i. Estimate the solution with luciferin value.

The algorithm can gain the optimal value of functions. The equations that modeled theluciferin-update, probability distribution used to select a neighbour, movement update and local-decision range update are given below:

$$l_i(t) = \max\{(0,(1-\rho) * l_i(t-1) + \gamma * J(x_i(t)))\} \tag{2}$$

$$P_j(t) = \frac{l_j(t)}{\sum_{k \in N_i(t)} l_k(t)} \tag{3}$$

$$x_i(t+1) = x_i(t) + s\left(\frac{x_j(t) - x_i(t)}{\left\|x_j(t) - x_i(t)\right\|}\right) \tag{4}$$

$$r_d^i(t+1) = \frac{r_s}{1 + \beta * D_i(t)} \tag{5}$$

Where,

$$N_i(t) = \{j : \left\|x_j(t) - x_i(t)\right\| < r_d^i(t); l_i(t) < l_j(t)\} \tag{6}$$

is neighbours of glowworm i consisting of those glowworms that have a relatively higher luciferin value and that are located within a dynamic decision domain. If the luciferin value of glowworm i is greater than j's and the distance between the glowworm i and j is less than the dynamic decision domain, divide glowworm j into the neighbours of glowworm i. Where

$$D_i(t) = \frac{N_i(t)}{\pi * r_s^2} \tag{7}$$

is the neighbour-density of glowworm i at iteration t and β is a constant parameter. The constant parameter β affects the rate of change of the neighbourhood range. The constant parameter ρ decides whether algorithm has memory. A value $\rho = 0$ renders the algorithm memoryless where the luciferin value of each glowworm depends only on the fitness value of its current position. However, $\rho \in (0,1]$ leads to the reflection of the cumulative goodness of the path followed by the glowworms in their current luciferin values. The constant parameter γ can scale the function fitness values. The value of step-size s influences the range of objective function.

3.2. Greedy Algorithm

Introducing local search mechanism - greedy algorithm and using heuristic correction operator are used to ensure solution to feasibility, making the search be in the feasible solution space. At the same time the guarantee on the feasibility of solution, try to increase its fitness value. If the solution is illegal, it changes

$x_j = 1$ into $x_j = 0$ in according with the article j the unit volume value $c_j' = c_j / \sum_{i=1}^{m} a_{ij} (j = 1, 2, ...m)$ ascending sequence until change illegal solution into legal solutions; if the solution is legal, ensure the feasibility of solution and change $x_j = 0$ into $x_j = 1$ in the light of c_j' descending order. It tries to increase function fitness. Although computing cost is increased, it makes search be certain guidance and no longer blind. So it costs are worth it [8].

4. Hybrid artificial glowworm swarm optimization (HGSO) algorithm for Solving MKP

Encoding is the key of solving the problem of the AGSO for knapsack problem and the process follows as: x_i^t is the location of glowworm i at time t and $x_i^t = [x_{i1}^t, x_{i2}^t, ..., x_{id}^t]$ (d , number of the dimensions and quantity of objects for loading backpacks) corresponds to a solution of the problem. When $x_{ij}^t = 1$, the i th glowworm put the j -th object into the bag. Otherwise, not in. The step-by-step algorithm of the HGSO for solving MKP is as follows:

Where, n indicate the number of the glowworm; l_0 indicate the luciferin initial value; r_0 indicate the adaptive decision-range initial value; β indicate the update rate of dynamic decision-range; n_t indicate the neighbourhood threshold; s indicate the step-size; ρ indicate the dissipation rate of luciferin value ; $iter - \max$, indicate the maximum iteration number; $t = 1$, initialize iteration number.

Step 1: Initialize the glowworms location, neighbours range and other parameters.Take a random 0/1 matrix as each glowworm initial position.

$$X_{ij} = \begin{cases} 1 & rand > 0.5 \\ 0 & rand < 0.5 \end{cases} \tag{7}$$

use of 0.5 as the boundary to discrete function value, because the glowworms have only two states (0, 1).

Step 2: Execute greedy strategy.

Step 3: Calculate the fitness value.

$$f(x_1, x_2, ...x_m) = \sum_{j=1}^{n} x_j c_j \quad s.t \sum_{j=1}^{n} x_j a_{ij} \le b_i, i = 1, 2, ...g \tag{8}$$

Step 4: Update luciferin value, neighbourhood threshold, adaptive decision-range value and the glowworms location using formula (2)-(6).

Step 5: If the maximum iteration number is satisfied, termination, otherwise, return step 4.

5. Simulation Experiments

In order to explain the executive effect of the algorithm, the program of this article is compiled by MATLAB. This program is running in the computer of Intel Dual T3200,2.0 GHz , 1G of memory. The Simulation instances come from http://people.brunel.ac.uk/mastjjb/jeb/orlib/files/mknap1.txt (mknap4.txt).

Example 1: Seven test instances from mknap1.txt.

The table 1 lists each problem scale, the announced optimal solution and using this algorithm get the best solution.

Table 1. Seven test instances from mknap1.txt

	Problem scale(n/m)	The optimal solution	HGSO for the best
T1	6/10	3800	3800
T2	10/10	8706.1	8706.1
T3	15/10	4015	4015
T4	20/10	6120	6120
T5	28/10	12400	123700
T6	39/5	10618	10539
T7	50/5	16537	16421

Above seven examples using HGSO, the best solution into bag items set binary coding respectively is as follow:

T1：011001

T2：0101100101

T3：110101101100011

T4：10000000010001111111

T5：11100000000001101111111111111

T6：10010101001011111110101111111111111011111

T7：00010111011011111101001001111111011100111111101110

From Table 1, it is observes that the HGSO algorithm can find seven example of optimal solution or approximate optimal solution.

6. Conclusion

This paper utilizes two important strategies, how to select the item based on the unit volume value and the binary glowworm swarm optimization algorithm and puts forward a kind of solving multi-dimensional 0-1 knapsack problems hybrid artificially glowworm swarm optimization algorithm, which aim to improve overall optimize performance and effect of solving of the basic GSO algorithm, expanding further the range of applications of GSO.

References

[1] Balas E, Zemel E.An algorithm for large zero-one knapsack problems.Operations Research, 1985, 28(5):32-40.

[2] Dembo R S, Hammer P L .A reduction algorithm for knapsack problem. Methods of Operations Researeh, 1980, 36(1):49-60.

[3] Plateeau A, Tachat D, Tolla P. A hybrid search combing interior point methods and metaheuristics for 0-1 programming. International Transactions on Operational Research,2009, (9):731 -746.

[4] Vasquezm, V Imont Y. Improved results on the 0 - 1 multi-dimensional knapsack problems. European Journal of Operational Research, 2005, 165: 70 -81.

[5] Du Haifeng, Liu Ruocheng, Jiao Licheng.et.Artificial immune antibody adjusting clonal algorithm for the 0-1 knapsack problem. Control Theory &Applications, (in Chinese), 2005, 22 (3): 348 -352.

[6] Chen Duanbing, Huang Wenqi.A Quasi-human Heuristic Algorithm for Multidimensional 0-1 Knapsack Problem.Computer Engineering and Applications,2006, 42 (2): 17 -19.

[7]Krishnanand, K.N. Ghose, D. Glowworm swarm optimization: a new method for optimizing multi-modal functions. Computational Intelligence Studies, 2009, 1(1):93-119.

[8] Elbaz D, Elkihel M. Load balancing methods and parallel dynamic programming algorithm using dominance technique applied to the 0 - 1 knapsack problems. Journal of parall and Distributed Computing, 2005, 65: 74- 84