IOSR Journal of Mobile Computing & Application (IOSR-JMCA)
e- ISSN: 2394-0050, P-ISSN: 2394-0042.Volume 7, Issue 2 (Mar - Apr 2020), PP 46-55
www.iosrjournals.org

Hybrid Chaotic Chemical Reaction Optimization for Multiple-Choice Knapsack Problem

Nguyen Hoai Phương[1], Dang Gia Dung[2] ,
Dao Thi My Linh[2] ,Nguyen Thi Bao Thu[2] , Tran Minh Hai[2]
[1]*Department of Computer Science, University of Labour and Social Affairs, Vietnam*
[2]*College of Electrical and Electronic, Thai Binh University, Thai Binh, Viet Nam.*

This paper proposes an algorithm based on metaheuristic chemical reaction optimization and chaotic map for multiple-choice knapsack problem (MCKP). MCKP is a well-known NP-hard problem and it has a lot of applications in the real-world and theory. Chemical reaction optimization (CRO) is a new optimization approach mimic the chemical reaction process. In short, the CRO has been shown to be superior to many other approaches in both continues and discrete domain. The chaotic have strong points in generate random sequence when compare with traditional random function.The advantage of CRO for MCKP is that CRO used four types of search operator, chaotic map, and a penalty function to help the algorithm find optima solution fast and accuracy. The proposed method has shown better performance than the genetic algorithm in a large test set in solving MCKP.

Keywords: *0-1 knapsack problem, Chemical reaction optimization, multiple-choice knapsack problem.*

Date of Submission: 01-06-2020 Date of Acceptance: 16-06-2020

I. Introduction

Given m classes $N_i = 1, \dots n_i, i = 1, \dots, m$ of items to pack in some knapsack of capacity W. Each item $j \in N_i$ has a cost c_{ij} and a size w_{ij}, and the problem is to choose one item from each class such that the total cost is minimized without having the total size to exceed W. The multiple-choice knapsack problem (MCKP) may thus be formulated as:

$$minimize \sum_{i=1}^{m} \sum_{j=1}^{n_i} c_{ij} x_{ij} \quad (1.1)$$

$$subject\ to \sum_{i=1}^{m} \sum_{j=1}^{n_i} w_{ij} x_{ij} \leq W, \quad (1.2)$$

$$\sum_{j=1}^{n_i} x_{ij} = 1, \forall i \in \{1,2, \dots, m\}, \quad (1.3)$$

$$x_{ij} \in \{0,1\}, \forall i = \{1, \dots, m\}, j \in N_i. \quad (1.4)$$

All coefficients c_{ij}, w_{ij}, and W are positive numbers, and the classes $N_1, \dots, N_m$ are mutually disjoint.

MCKP is known as an NP-hard (Non-deterministic Polynomial-time difficult) issue [5]. The issue has a huge scope of utilizations: Capital Budgeting [1], Menu Planning [2], transportation programming [3], nonlinear backpack issues [1], deals asset designation [2], structure of data frameworks [4], and so forth. According to Marshall L. Fisher (2004), the MCKP additionally show up by Lagrange unwinding of a few number programming issues [18].

Over the span of the latest four decades, researchers have proposed various approaches to manage unwind MCKP. Author can describe the methods for this issue into two classes to be explicit, right figurings and derived estimations.

For exactly method, the branch-bound methodology is an including approach, which diminishes its request space by excepting unimaginable plans. According to M.E. Dyer et al., 1984, the branch-and-bound count and its varieties have been proposed [19], [1], [2]. Dynamic programming count are proposed by (Krzysztof Dudzinski and Stanislaw Walukiewicz, 1987, David Pisinger, 1995) [20], [21]. According to M.E.Dyer et al., (1995), two mutt figurings that join dynamic programming and branch-and-bound are proposed in [21], [22].

Since MCKP is a NP-Hard issue. The precisely calculations have unpredictability time in exponential capacities. The heuristic calculation has leverage in finding surmised ideal in polynomial time. One of the notable heuristic calculations is GA [6]. In spite of the fact that, GA is pioneer in explaining MCKP however it is still met a disadvantage that it gets stuck in nearby optima.

The CRO, by outflanking many existing transformative calculations, it has effectively tackled numerous issues as of late. CRO has been effectively connected to the quadratic task issue [7], asset compelled venture booking issue [7], direct task issue in remote work systems [8], populace change issue in distributed

gushing [9], subjective radio range assignment issue [10], framework planning issue [11, 12], standard consistent benchmark work [13], stock portfolio choice issue [14], counterfeit neural system preparing [15], organize coding advancement [16], booking on Heterogeneous Computing Environments [17], 0-1 Knapsack issue what's more, to numerous different issues.

In this examination, the CRO is made to enlighten MCKP. The proposed computation shows up GA while using a comparable presentation game plan, also, half breed and change managers in GA are moreover used as neighbor looks for in CRO. However, CRO and GA are absolutely particular in the way in which they attempting to find perfect game plan. GA is imitated from transformative procedure using three directors: change, cross breed, and region; while the CRO is reflect from engineered reaction process using four simple substance reactions: On-Wall Ineffective Collision, breaking down, Inter-nuclear insufficient, and mix. The investigation of MCKP test set demonstrates that CRO superior to GA.

The rest of the paper is organized in sections: Section I briefly gives the original framework of CRO. Section II explains the modification of the original CRO to adapt it to the MCKP problem. Author survey the behavior of CRO and compare the simulated results of the CRO with GA in Section V. Author conclude this paper and suggest potential future work in Section VI.

II. Basic Chemical Reaction Optimization

CRO [7] is a metaheristic energized by the engineered reaction process. In CRO, one molecule (M) that has the sub-nuclear structure ω, potential essentialness (PE), Kinetic imperativeness (KE), least structure (minStruct), hits number (numHit), number of hit that minStruct is gotten (minHit) and various characteristics address a potential course of action. It replicates four sorts of substance reactions including on-divider lacking effect, disintegration, between nuclear inadequate accident and blend.

In the reaction technique, the PE goes toward the unimportant state, similar to target work in improvement issues. PE is regularly used as the wellbeing of the objective work.

Author mean PE_ω and KE_ω are PE and KE of an atom with particle structure ω, individually. An atom with particle structure ω is likewise called atom ω.

CRO includes three phases: initial phase, iteration phase and final phase. The initial phase, assigning initial values for parameters *PopSize, KElossRate, InitialKE, MoleColl, α, β and buffer*. An initial population (Pop) including $PopSize$ molecules is generated. Their initial KEs are assigned by $InitialKE$, and their initial PEs are assigned by its objective function values.

In each iteration, if $r > MoleColl$, where r is a randomly number distributed in [0,1], and $MoleColl \in [0,1]$ is a presetting system parameter then uni-molecular reaction is chosen. Otherwise, an inter-molecular reaction is trigged. In an uni-molecular reaction, one molecule ω is randomly chosen and check if it satisfies the decomposition condition: $(numHit_\omega - minHit_\omega) > \alpha$, where α is a presetting system parameter. If so, decomposition reaction is trigged, else the On-Wall ineffective collision is trigged. In inter-molecular, two molecules ω_1, ω_2 are randomly selected from the Pop and check if synthesis condition is satisfied:($KE_{\omega_1} \leq \beta$ and $KE_{\omega_2} \leq \beta$), where β is a presetting system parameter, it consider as the minimum KE a molecule should have. Otherwise, an inter-molecular ineffective collision is executed. The pseudocode of CRO is described in Algorithm 1.

Algorithm 1: CRO algorithm

Input: Problem-specific information (the objective function f, constraints, and the dimensions of the problem

Output: The overall minimum solution and its function value

```
Assign parameter values to PopSize, KELossRate, MoleColl, buffer and InitialKE
Let Pop be the set of molecules {1, 2,..., PopSize}
Evaluate the molecules in Pop
while <the termination criteria not met> do
    Get r randomly in interval [0,1]
    if (r > MoleColl) then
        Select a molecule ω from Pop randomly
        if <decomposition criterion met> then
            Execute decomposition reaction
        else
            Execute On – Wall Ineffective Collision reaction
    else
        Select two molecules ω1 and ω2 from Pop randomly
        if <synthesis criterion met> then
            Execute Synthesis reaction
        else
            Execute Inter – molecular ineffective collision reaction
    Check for any new minimum solution
return
```

2.1 Elementary reactions

On-Wall Ineffective Collision

When this reaction is trigged, one molecule ω is randomly chosen from the Pop; a neighborhood operator is used to generate a new one ω'. If the (1.5) holds.

$$PE_\omega + KE_\omega \geq PE_{\omega'} \qquad (1.5)$$

In this case, $KE_{\omega'}$ is obtained by

$$KE_{\omega'} = (PE_\omega + KE_\omega - KE_{\omega'}) \times q$$

where$q$ is a random number in $[KElossRate, 1]$, $KElossRate \in [0,1]$ is a presetting system parameter of CRO. The molecule ω is replaced by the new one ω'. The remaining $1 - q$ portion of energy is transferred to a central energy buffer ($buffer$).

If (6.5) is not satisfied, the $numHit_\omega$ is increased by 1, ω' is discarded.

Decomposition

In the decomposition, a neighborhood search is used to generate two new molecules ω'_1, ω'_2 from one molecule ω. There are two cases, the change will be accepted.

The first case, (1.6) is held.

$$PE_\omega + KE_\omega \geq PE_{\omega'_1} + PE_{\omega'_2} \qquad (1.6)$$

Let $E_{temp\,1} = PE_\omega + KE_\omega - PE_{\omega'_1} - PE_{\omega'_2}$. The $KE_{\omega'_1}$, and $KE_{\omega'_2}$ are updated as follows.

$$KE_{\omega'_1} = E_{temp\,1} \times q$$

$$KE_{\omega'_2} = E_{temp\,1} \times (1 - q)$$

where q is a random number in [0; 1].

The second case, the reaction is also executed with the support of central energy $buffer$ when (1.7) is hold.

$$PE_\omega + KE_\omega + q_1 \times q_2 \times buffer \geq PE_{\omega'_1} + PE_{\omega'_2} \qquad (1.7)$$

whereq_1 and q_2 are randomly uniform number in [0,1]. In this case, let

$$E_{temp\,2} = PE_\omega + KE_\omega + q_1 \times q_2 \times buffer - PE_{\omega'_1} - PE_{\omega'_2}$$

The $KE_{\omega'_1}, KE_{\omega'_2}$ are updated as follows.

$$KE_{\omega'_1} = E_{temp\,2} \times q_3$$

$$KE_{\omega_2'} = E_{temp\,2} \times (1 - q_3)$$

whereq_3 is a random number in [0; 1]. The q_1 andq_2 are used to restrict the $KE_{\omega_1'}$ and $KE_{\omega_2'}$ from being too large, for the buffer is normally large. The buffer is then updated by

$$buffer = (1 - q_1 \times q_2)buffer$$

If either (1.6) or (1.7) is satisfied, the molecule ω is removed from the Pop, two new ones ω'_1 and ω'_2 are added. Otherwise, the change is forbidden, only $numHit_\omega$ is increased by 1.

Inter-molecular ineffective collision

A neighborhood operator is used to generate two new molecules ω'_1, ω'_2 from two molecules ω_1, ω_2.
The change is accepted if (1.8) holds

$$PE_{\omega_1} + PE_{\omega_2} + KE_{\omega_1} + KE_{\omega_2} \geq PE_{\omega_1'} + PE_{\omega_2'} \qquad (1.8)$$

Let $E_{temp} = PE_{\omega_1} + PE_{\omega_2} + KE_{\omega_1} + KE_{\omega_2} - PE_{\omega_1'} - PE_{\omega_2'}$.
$KE_{\omega_1'}$and$KE_{\omega_2'}$ are updated respectively by

$$KE_{\omega_1'} = E_{temp} \times q$$

$$KE_{\omega_2'} = E_{temp} \times (1 - q),$$

where$q$ is a random number in [0; 1]. $numHit_{\omega_1'}$, and $numHit_{\omega_2'}$ are increased by 1. $minHit_{\omega_1'}$and$minHit_{\omega_2'}$ equal to $numHit_{\omega_1'}$ and $minHit_{\omega_2'}$, respectively. If is not satisfied, only $numHit_{\omega_1}$ and $numHit_{\omega_2}$ are increased by 1.

Synthesis

Firstly, one combining operator is used to combined two molecules ω_1, ω_2 become one molecule ω'. If (1.9) is satisfied.

$$PE_{\omega_1} + PE_{\omega_2} + KE_{\omega_1} + KE_{\omega_2} \geq PE_{\omega'} \qquad (1.9)$$

Then $KE_{\omega'}$ is assigned by $KE_{\omega'} = PE_{\omega_1} + PE_{\omega_2} + KE_{\omega_1} + KE_{\omega_2} - PE_{\omega'}$
The new molecule ω' is added to the Pop, and two old molecules ω_1, ω_2 are discarded.

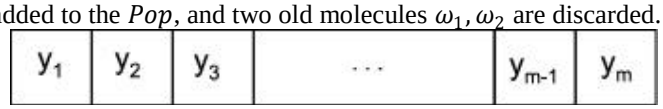


Figure 1.1: Solution presentation

If (1.9) is not satisfied, the $numHit_{\omega_1}$, and $numHit_{\omega_2}$ are increased by 1, ω' is discarded. The ideal of this operator is that a molecule with bigger KE has more chance to explore another place in the search space.

III. Logistic chaotic map

According to Geoff Boeing(2016), the logistic map is a polynomial mapping (equivalently, recurrence relation) of degree 2, often cited as an archetypal example of how complex, chaotic behaviour can arise from very simple non-linear dynamical equations[23]. The map was popularized in a 1976 paper by the biologist Robert May, in part as a discrete-time demographic model analogous to the logistic equation first created by Pierre François Verhulst by Cheng Zhang (2010) [24].
Mathematically, the logistic map is written

$$x_{n+1} = rx_n(1 - x_n)$$

where x_n is a number between zero and one that represents the ratio of existing population to the maximum possible population. The values of interest for the parameter r (sometimes also denoted μ) are those in the interval [0,4].
The chaotic have strong points in generate random sequence when compare with traditional random function.

IV. Solving MCKP by Hybrid Chemical reaction optimization

4.1 Solution Representation

According to M Gen and R Cheng (2000), similar to GA in represent a solution [25]. An integer string is used to represent a solution. The y_i receives an integer in N_i, it represents$y_i \in N_i$ is chosen. The string length is m corresponding to a solution in MCKP. The solution presentation is depicted in Fig. 6.1. By defining an indicator variable, y_i is as follows:

$$y_i = j \;\; if \; x_{ij} = 1, \quad j \in N_i, \quad i = 1,2, \ldots, m$$

4.2 Objective function

The objective PE is calculated as formula:

$$PE = \sum_{i=1}^{m} \sum_{i=1}^{n_i} c_{ij} x_{ij} + g(x) \qquad (1.10)$$

where $g(x)$ is penalty function as following:

$$g(k) = \begin{cases} 0 & if\ (1.2)\ is\ hold \\ \Omega_0 + \left(\sum_{i=1}^{m} \sum_{i=1}^{n_i} w_{ij} x_{ij} - W\right) & if\ otherwise. \end{cases}$$

where Ω_0 is a given positive constant. The ideal here is that for violate solution will have a larger PE. It forces the algorithm search both sides of search space that is feasible and infeasible domains.

4.3 Elementary operators

On-wall operator

This operator use for On-Wall Ineffective Collision reaction. One position i^{th} is randomly selected from $1, \dots, m$, and value of y_i is replaced by a random number in $1, \dots, n_i$.

Inter-molecular ineffective collision operator

Two solutions $\omega_1^{'}$ and $\omega_2^{'}$ are obtained from two solutions ω_1 and ω_2. The two points crossover operator commonly used in GA is adopted. In the procedure, two points k_1, k_2 will be chosen to separate each of the solutions ω_1 and ω_2 to three parts. The solution $\omega_1^{'}$ is created from the even parts of ω_1 combined with the odd parts of ω_2. The solution $\omega_2^{'}$ is created from the even parts of ω_2 combined with the odd parts of ω_1.

Decomposition operator

This process produces two solutions from one original solution. This operator affects diversification and makes the algorithm explorer the search space. The decomposition operator is designed inspiring from the "half-total-exchange" operator that is used to solve the channel assignment problem [7]. The operator creates two solutions $\omega_1^{'}$ and $\omega_2^{'}$ from solution ω. Firstly, ω is duplicated to generate $\omega_1^{'}$ and, $\omega_2^{'}$. After that, perturbations for $n/2$ positions in solutions $\omega_1^{'}$ and $\omega_2^{'}$ are made randomly. The pseudocode of the decomposition operator is described in Algorithm 10.

Algorithm 10:*decomposition (ω)*

Input: Solution ω
Output:$\omega_1^{'}$and $\omega_2^{'}$
1 Duplicate ω to produce $\omega_1^{'} and \omega_2^{'}$**for** change ← 1 to $n/2$**do**
2 Get *i* and *j* randomly in the set of $\{1, \dots, m\}$.
3$\omega_1^{'}(i) and \omega_2^{'}(j)$are assigned by two random integers in $N_i\ and\ N_j$, respectively
4 **return**

Synthesis operator

In this algorithm, the synthesis operator is used [13].The operator combines two molecules with solutions ω_1 and ω_2 into one molecule with solution $\omega^{'}$.For each $\omega^{'}$ (i) is randomly selected from ω_1(i) or ω_2(i).The repair function is also used to ensure the constraint is met. The pseudocode of the synthesis operator is described in Algorithm 11.

Algorithm 11:*synthesis*(ω_1, ω_2)

Input: Solution $\omega_1 and \omega_2$
Output: Solution$\omega^{'}$
1for$i$ ← 1 to n **do**
2 Get *r* randomly in [0,1]
3if $(r > 0.5)$**then**
4 $\omega^{'}(i) \leftarrow \omega_1(i)$
5 **else**
6 $\omega^{'}(i) \leftarrow \omega_2(i)$
7 **return**

V. Experiment and analysis

5.1 Data test set

All the algorithms were implemented in Matlab R2016b. The test environment is set up on a personal computer with Intel core i5 CPU at 1.6 GHz CPU, 2G RAM, running on Windows 10.

Author will consider how the algorithm behaves for different problem sizes, test instances, and data-ranges.

Two types of randomly generated data instances are considered, each instance tested with data-range $R = 1000$ for different number of classes m and sizes n_i:

• Strongly correlated data instances (SC): In knapsack problem w_j is randomly generated in [1, R] and $c_j = w_j + 10$. For each class i generate n_i items (w'_j, c'_j) as for knapsack problem, and order these by increasing weight. The data instance for MCKP is then $w_{ij} = \sum_{h=1}^{j} w'_h$ and $c_{ij} = \sum_{h=1}^{j} c'_h$, $j = 1,2,\dots,n_i$. Such instances have no dominated items, and form an upper convex set.

• Subset-sum data instances (SS): Each w_{ij} is randomly chosen in [1, R] and $w_{ij} = c_{ij}$. These instances are hard because its upper bound will yield $u_{ij} = W$[105].

For each instance, the C is calculated as follows.

$$W = \frac{1}{2}\left(\sum_{i=1}^{m} min_{j \in N_i}(w_{ij}) + max_{j \in N_i}(w_{ij})\right)$$

5.2 Parameter setting

GA's parameters are set: $Popsize = 20$, $Pc = 0.8$, $Pm = 0.1$[10].

For the CRO, parameter setting affects its performance. Our goal is to assign parameter values to CRO with relatively good performance for the test instances. The parameters are assigned for CRO1 and CRO2 as follows: $\text{KElossRate} = 0.8, \text{InitialKE} = 1000, \text{PopSize} = 20, \text{MoleColl} = 0.2, \text{buffer} = 0$, $\alpha = 10000$ and $\beta = 10$. For CRO2, the random generation is replaced by logistic chaotic map.

5.3 Experiment results

Author observe the convergence curves of three test instances in the strongly correlated test set. The three instances with $(m = 10, n = 10)$, $(m = 100, n = 100)$, and $(m = 1000, n = 100)$ are used in this experiment. Figure 1.3 shows the evolution of the mean of the best total costs of CRO1 and CRO 2 over 30 runs in the three instances. It indicates the global search ability and the convergence ability of CRO. There are several observations and they are given as follows:

In the case $(m = 10, n = 10)$, as depict in Fig. 1.2(a) the convergence curve of GA is tied with CRO's, but CRO is still better. For instance $(m = 100, n = 100)$, the Fig. 1.2(b) shows that CRO have a much more quick convergence compares with GA, and PSO. For larger instance $(m = 1000, n = 100)$, the Fig. 1.2(c) show that CRO still have a good convergence, while GA, and PSO shows a very slow convergence. From the Fig. 1.2 shows that CRO much more better GA, and PSO in convergent rate and solution quality when solving large MCKP.

Author adopt the same stopping criterion, that the function evaluation limit is set to 100000, for all the test.

Table 1.1 shows the experimental results of the strongly correlated instances. For all the proposed instances, CRO yields superior results compared with GA, and PSO. The series of experimental results demonstrate the superiority and effectiveness of CRO. In comparison with GA, and PSO; CRO can get better results in a shorter time. The smaller standard deviation (StdDev) shows that the new algorithm is more robust than GA, and PSO.

Table 1.2 shows the experimental results of the subset-sum instances. In all the instances, CRO shows much better than GA, and PSO in the solution quality.

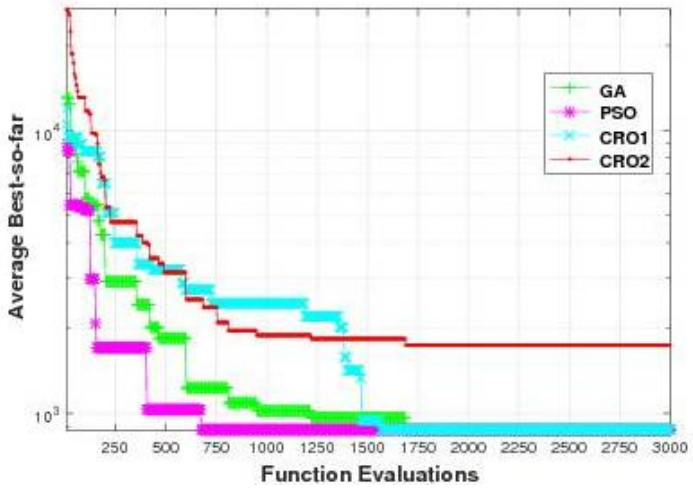
Instances	Algorithm	mean	worst	best	stdDev	time
m=10, n=10	GA	873.20	870.00	893.20	15.98	7.076
	PSO	878.79	873.51	898.92	0.00	7.355
	CRO1	887.10	881.91	907.00	15.98	7.076
	CRO2	893.75	883.67	907.29	0.00	7.355
m=10, n=100	GA	584.40	243.80	684.40	294.60	13.9
	PSO	591.23	246.36	687.45	0.00	7.384
	CRO1	593.64	250.25	689.46	294.60	13.9
	CRO2	594.02	259.69	696.06	0.00	7.384
m=10, n=1000	GA	3065.00	2340.80	3065.00	1807.00	20.74
	PSO	3070.34	2348.00	3073.42	0.28	7.655
	CRO1	3072.82	2352.00	3080.65	1807.00	20.74
	CRO2	3081.18	2353.02	3087.80	0.28	7.655
m=10, n=10	GA	30910.05	25289.44	30910.34	2978.00	29.66
	PSO	30918.21	25293.01	30915.24	0.00	8.969

	CRO1	30923.18	25297.21	30919.25	2978.00	29.66
	CRO2	30927.13	25303.29	30926.10	2978.00	25.66
m=100, n=10	GA	200000.00	100000.00	200000.00	24365.00	39.12
	PSO	200002.93	100000.95	200005.24	22.20	8.47
	CRO1	200005.18	100005.30	200010.24	24365.00	39.12
	CRO2	200007.84	100006.14	200011.83	24365.00	34.12
m=100, n=100	GA	2000230.00	1700000.00	2000230.00	300000.00	49.94
	PSO	2000230.99	1700005.80	2000234.16	31.04	9.075
	CRO1	2000235.89	1700015.15	2000239.95	300000.00	49.94
	CRO2	2000238.94	1700021.73	2000246.53	300000.00	49.4
m=1000, n=10	GA	10170210.00	10000000.00	10170210.00	31502.00	95.15
	PSO	10170218.34	10000004.60	10170211.14	133.22	17.89
	CRO1	10170226.44	10000009.73	10170213.23	31502.00	95.15
	CRO2	10170228.11	10000011.93	10170222.34	31502.00	99.15

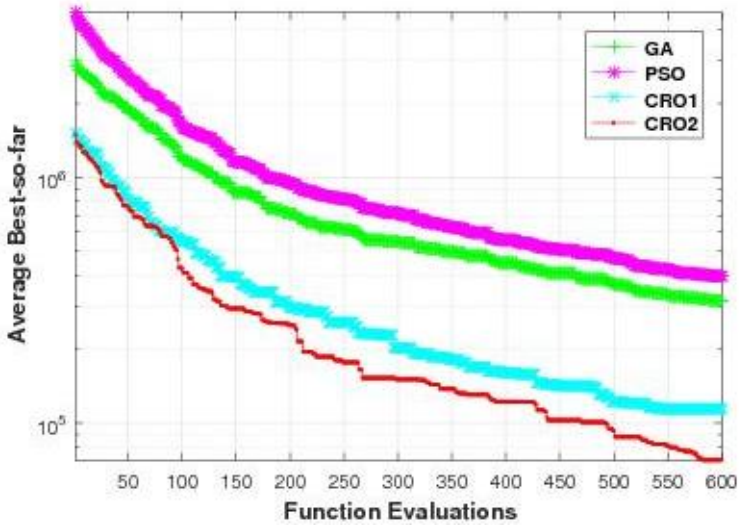
Table 1.1: Experimental results for strongly correlated instances

Instances	Algorithm	mean	worst	best	stdDev	time
m=10, n=10	GA	833.20	820.04	893.20	17.98	6.08
	PSO	834.48	823.89	893.78	0.00	7.36
	CRO1	836.16	829.27	893.99	15.98	7.08
	CRO2	844.45	834.44	903.71	0.00	7.36
m=10, n=100	GA	582.40	443.80	683.40	24.60	13.90
	PSO	590.45	448.32	692.14	0.00	6.38
	CRO1	599.73	454.17	700.07	294.60	13.90
	CRO2	607.66	455.19	707.02	0.00	7.38
m=10, n=1000	GA	3061.00	2740.80	3165.00	1807.00	20.74
	PSO	3061.62	2750.60	3165.99	0.28	7.66
	CRO1	3064.40	2757.72	3171.80	1807.00	20.74
	CRO2	3069.86	2764.10	3176.99	0.28	7.66
m=10, n=10	GA	30910.05	25289.44	30910.34	2978.00	29.66
	PSO	30917.24	25295.45	30911.79	0.00	8.97
	CRO1	30917.59	25305.40	30917.96	2978.00	9.66
	CRO2	30920.90	25312.24	30922.88	2978.00	25.66
m=100, n=10	GA	201230.00	201122.00	200000.00	24365.00	39.12
	PSO	201232.37	201124.45	200007.91	22.20	8.47
	CRO1	201240.67	201125.29	200010.70	24365.00	39.12
	CRO2	201248.73	201131.73	200018.87	24365.00	34.12
m=100, n=100	GA	000230.00	1700022.00	2000230.00	300000.00	49.94
	PSO	2000236.52	1700031.23	2000234.26	31.04	9.08
	CRO1	2000246.04	1700032.54	2000235.50	300000.00	49.94
	CRO2	2000246.43	1700038.89	2000239.93	300000.00	49.40
m=1000, n=10	GA	9170210.00	9000000.00	9170210.00	1502.00	95.15
	PSO	9170217.10	9000001.48	9170216.60	133.22	17.89
	CRO1	9170220.38	9000005.70	9170221.31	31502.00	95.15
	CRO2	9170220.79	9000010.49	9170223.73	31502.00	99.15

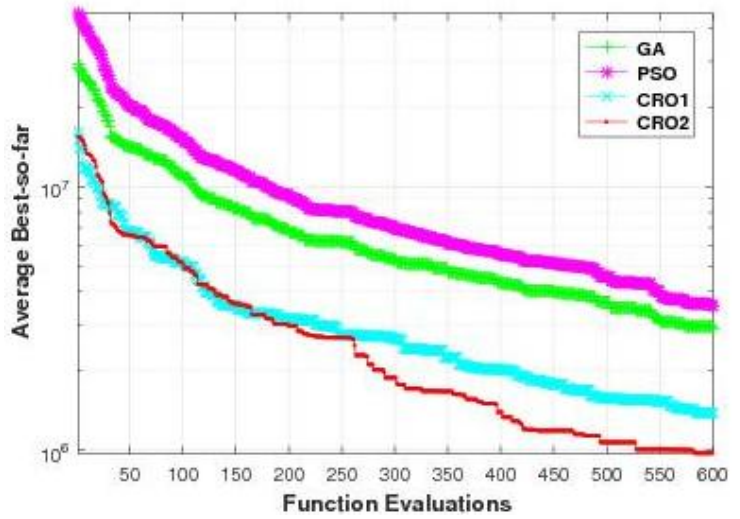
Table 1.2: Experimental results for subset-sum instances



(a)m=10, n=10

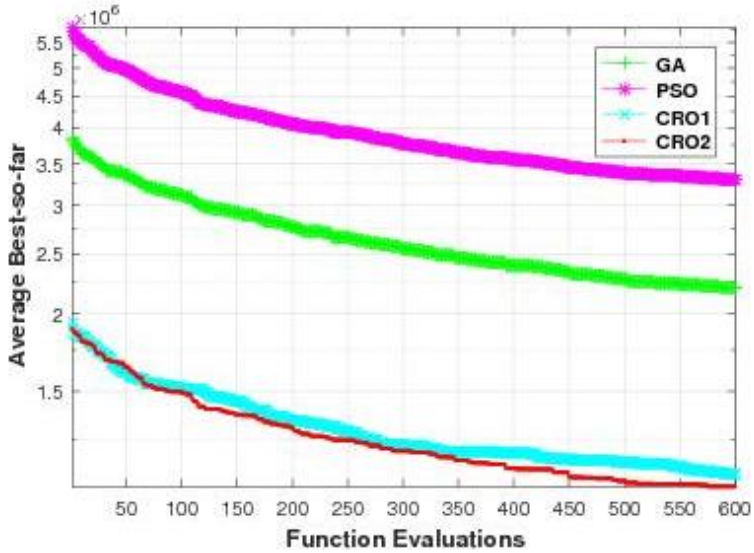


(b)m=100, n=100

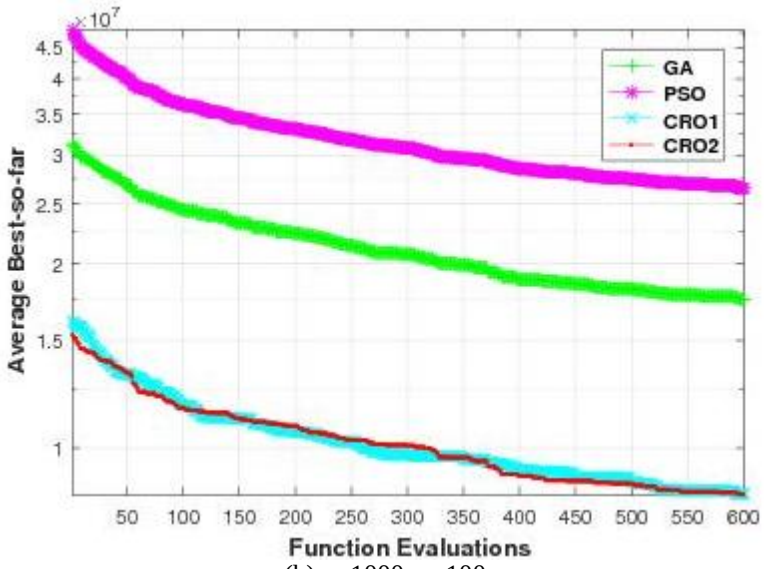


(c)m=1000, n=100

Figure 1.2: Comparison of GA, PSO, CRO1, and CRO2 on the MCKP. The objective function value ere averaged over 25 runs



(a)m=1000, n=10



(b)m=1000, n=100

Figure 1.3: Comparison of GA, PSO, CRO1, and CRO2 on the MCKP. The objective function value was averaged over 30 runs

VI. Conclusion

Multiple-choice knapsack problem is a well-known NP-hard problem. It has a large wide range of applications in real-world problems and mathematic theory. Chemical reaction optimization is a new metaheuristic mimic from a chemical reaction process that has shown excellent outperform many state-of-the-art approaches. The new algorithm in this chapter based on chemical reaction optimization metaheuristic is proposed to solve this problem. The experiment on a large range of data set demonstrates that the proposed method has superior performance when compared with GA, and PSO. In the future, author will develop a parallel version of this algorithm that improved efficiency.

References

[1]. Robert M. Nauss. The 01 knapsack problem with multiple choice constraints. European Journal of Operational Research, 1978, 2(2): 125-131
[2]. P. Sinha and A.A.Zoltners. The multiple-choice knapsack problem.Operational Research, 1979, 27(3): 503
[3]. Tao Zhong and Rhonda Young. Multiple choice knapsack problem: Example of planning choice in transportation. Evaluation and Program Planning, 2010, 33(2): 128- 137
[4]. P.C.Yue and C.K.Wong. Storage cost considerations in secondary index selection. International Journal of Parallel Programming, 1975, 4: 307-327
[5]. Michael R. Garey and David S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, America, 1979
[6]. M R Gen, R Cheng, M Sasaki, and Y Jin. Multiple-choice knapsack problem using genetic algorithms. Integrated Technology Systems, Maui, HI, 1998
[7]. A.Y.S.Lam and V.O.K.Li. Chemical-reaction-inspired metaheuristic for optimization. Evolutionary Computation, IEEE Transactions on, 2010, 14(3): 381-399
[8]. A.Y.S.Lam and V.O.K.Li. Chemical reaction optimization: a tutorial. Memetic Computing, 2012, 4: 3-17
[9]. A.Y.S.Lam, Jialing Xu, and V.O.K.Li. Chemical reaction optimization for population transition in peer-to-peer live streaming. In Evolutionary Computation (CEC), 2010 IEEE Congress on, 2010: 1-8
[10]. A.Y.S.Lam and V.O.K.Li. Chemical reaction optimization for cognitive radio spectrum allocation. In Global Telecommunications Conference(GLOBECOM 2010), 2010 IEEE, 2010: 1-5
[11]. Jin Xu, A.Y.S.Lam, and V.O.K.Li. Chemical reaction optimization for the grid scheduling problem. In Communications (ICC), 2010 IEEE International Conference on, 2010: 1-5
[12]. Jin Xu, A.Y.S.Lam, and V.O.K.Li. Chemical reaction optimization for task scheduling in grid computing. Parallel and Distributed Systems, IEEE Transactions on, 2011, 22(10): 1624-1631
[13]. A.Y.S.Lam, V.O.K.Li, and J.J.Q.Yu. Real-coded chemical reaction optimization. Evolutionary Computation, IEEE Transactions on, 2011, 16(3): 339-353
[14]. Jin Xu, Albert Y.S.Lam, and Victor O.K.Li. Stock portfolio selection using chemical reaction optimization. In Proceedings of International Conference on Operations Research and Financial Engineering (ICORFE 2011), Paris, France, 2011: 458-463
[15]. J.J.Q.Yu, A.Y.S.Lam, and V.O.K.Li. Evolutionary artificial neural network based on chemical reaction optimization. In Evolutionary Computation (CEC), 2011 IEEE Congress on, 2011: 2083-2090
[16]. Bo Pan, A.Y.S.Lam, and V.O.K.Li. Network coding optimization based on chemical reaction optimization. Proceedings of the Global Communications Conference (GLOBECOM 2011), 2011: 1-5
[17]. Kenli Li, Zhimin Zhang, Yuming Xu, Bo Gao, and Ligang He. Chemical reaction optimization for heterogeneous computing environments. InParallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on, 2012: 17-23

[18]. Marshall L. Fisher. The lagrangian relaxation method for solving integer programming problems. Manage. Sci.,2004, 50(12 Supplement): 1861-1871
[19]. M.E. Dyer, N. Kayal, and J. Walker. A branch and bound algorithm for solving the multiple-choice knapsack problem. Journal of Computational and Applied Mathematics, 1984, 11(2): 231-249
[20]. Krzysztof Dudzinski and Stanislaw Walukiewicz. Exact methods for the knapsack problem and its generalizations. European Journal of Operational Research, 1987, 28(1): 3-21
[21]. David Pisinger. A minimal algorithm for the multiple-choice knapsack problem. European Journal of Operational Research, 1995, 83(2): 394-410
[22]. M.E.Dyer, W.O.Riha, and J.Walker. A hybrid dynamic programming/ branch-and-bound algorithm for the multiple-choice knapsack problem. Journal of Computational and Applied Mathematics, 1995, 58(1): 43-54
[23]. Geoff Boeing. Visual Analysis of Nonlinear Dynamical Systems: Chaos, Fractals, Self-Similarity and the Limits of Prediction. Systems., 2016, 4 (4): 37
[24]. Cheng Zhang. Period three begins. Mathematics Magazine, 2010, 83: 295-297
[25]. M Gen and R Cheng. Genetic algorithms and engineering optimization. John Wiley and Sons, New York, 2000