Newton's Law of Gravity-Based Search Algorithms

*[1] S. Raja Balachandar, [2]K. Kannan,

Department of Mathematics, SASTRA University, Thanjavur, India,613001
[1]srbala09@gmail.com, [2] kkannan@maths.sastra.edu

Abstract

Many heuristic optimization methods have been developed in recent years that are derived from Nature. These methods take inspiration from physics, biology, social sciences, and use of repeated trials, randomization, and specific operators to solve NP-hard combinatorial optimization problems. In this paper we try to describe the main characteristics of heuristics derived from "Newton's law of gravitation", namely a gravitational emulation local search algorithm and a gravitational search algorithm. We also present the detailed survey of distinguishing properties, parameters and applications of these two algorithms.

Keywords: Meta -Heuristic algorithms, gravitation, Newton's law of Gravity, Combinatorial optimization problems, NP-Hard.

1. Introduction

Heuristic techniques are developed for solving combinatorial optimization problems (COP) with exponential search spaces because exact techniques are not practical for these cases. Using heuristics designed from natural and social systems is one of the most important and promising research fields of late. These non-deterministic heuristic methods are used to obtain very good results for solving NP-hard combinatorial optimization problems. In the literature various heuristic approachesderived from nature have been developed by researchers, namelyGenetic Algorithms (GA) (Tang et al., 1996), Simulated Annealing(SA)(Kirkpatrick et al., 1983),Ant Search Algorithm(ASA)(Dorigo et al., 1996), Particle Swarm Optimization(PSO)(Kennedy and Eberhart, 1995), Neural Nets(NN)(Hopfield, 1982; Righini,1992),Tabu Search (TS)(Glover,1989; Glover, 1990), etc. These algorithms are analyzed and applied in many areas (Badrand Fahmy, 2004; Berghand Engelbrecht2006; Ellabib et al., 2007; Hamzaçebi, 2008; Lozano et al., 2008; Tripathi and Bandyopadhyay, 2007). Performances of these algorithms differ from problem to problem, and there is no single approach to achieve the best solution for all COP.

These heuristics are obtained by using a fixed number of repeated iterations, employing one or more agents (neurons/particles/chromosomes/ants/etc.), operating with a mechanism of competition-cooperation, and/or embedding procedures of self-modifications.Nature has two operations: first, it selects the stronger individuals and penalizes the weaker ones, and second it introduces elements of randomness and permits the generation of new individuals. This same idea is used in the heuristic: selection is the basic idea for optimization, and mutation is the basic idea for non-deterministic search. The characteristics of these heuristics aremodeling a phenomenon that exists in nature, non-determinism, parallel structure and adaptability.

Recently, the Gravitational Emulation Local Search Algorithm (GELS) and Gravitational Search Algorithm (GSA) were developed by researchers for solving COP by using Newton's law of gravitation: "Every particle in the universe attracts every other particlewith a force that is directly proportional to the product of their masses and inversely proportional to the square of the distancebetween them".Many applications of these two gravitational algorithms are available in the literature (Abbas Bahrololoumet al., 2012; Barzegaret al., 2009; Behranget al.,2011;Binod Shaw et al., 2012; Chaoshun Li, and Jianzhong Zhou, 2011;Minghao Yin et al., 2011; Raja Balachandar and Kannan,2007; 2009; 2010; Rashedi et al. 2009;Rashediet al., 2010; Sarafraziet al., 2011; Rashediet al., 2011; SoroorSarafrazi,2013; Webster, 2004).

This paper is organized as follows:Section 2 provides background on the law of gravitation. GELS and the application of GELS are presented in Sections 3 and 4, respectively. Sections 5 and 6, respectively, are dedicated to GSA and its application. Convergence analysis and concludingremarks are discussed in Section 7 and Section 8, respectively.

2. Gravitation.

Gravitation is one of the four fundamental interactions (including the electromagnetic force, the weak nuclear force, and the strong nuclear force) innature (Schutz, 2003). It isthe tendency of masses to accelerate toward each other; everyparticle inthe universe attracts every other particle. The inescapability of gravity makes it different from allother natural forces.The way Newton's gravitational force behaves is called "action at a distance", which means gravity acts between separatedparticles without any intermediary and without any delay. Newton's law of gravity states that each particle attracts every otherparticle with a "gravitational force" (GF)(Schutz, 2003; Hollidayet al., 1993).

The GF between two particles is directly proportional to the productof their masses and inversely proportional to the square of the distance between them (Hollidayet al., 1993):

$$F = \frac{GM_1M_2}{R^2} \quad (1)$$

In (1),F, G, M_1, M_2 and R represent the magnitude of the gravitational force, the gravitational constant, the mass of the first and secondparticles, and the distance between the two particles, respectively. Newton's second law says that when a force, F, is appliedto a particle, its acceleration, a, depends only on the force applied and the mass, M, of the particle:

$$a = \frac{F}{M} \quad (2)$$

From(1) and (2), we conclude that there is an attracting gravity force among all particles of the universe, where the effect becomes greater with increasing mass of the particles and decreasing distance between them.In the literature, two different gravitational algorithms are available and based on equation (1), namely GELS and GSA. Researchers have designed these two approaches for solving various COP. In the next section we discuss the GELS approach, parameters, properties and applications.

3. Gravitational Emulation Local Search Algorithm (GELS)

Preliminary work done with GELS

For the study of GELS, a conceptual framework was first developed by Webster (2004) called the Gravitational Local Search Algorithm (GLSA). As there are a number of places in the litera-ture where the abbreviation"GLS" is used to refer to guided local search (Voudourisand Tsang,1995), this name was later changed to GELS. Two separate versions of GLSA were implemented us-ing the C computer programming language. They differ in two key areas: the first version, dubbed GLSA1, used as its heuristic Newton's equation for gravitational force between two objects, and included a"pointer" object that moved through the search space one position at a time, iteratively examining solutions to the COP. The second version, dubbed GLSA2, used as its heu-ristic Newton's method for gravitational field calculation, and the pointer object was allowed to move multiple positions at a time (i.e. to a different area of the search space). These issues are pre-cisely elaborated in (Webster, 2004).

Both versions of the original algorithm had operational parameters that the user could set to tune its performance. These were density(DENS), drag (DRAG), friction(FRIC), gravity(GRAV), initial velocity (IVEL), iteration limit(ITER), mass(MASS), radius(RADI), silhouette(SILH), and threshold(THRE). Of these, GELS uses only four – velocity, iteration limit, radius and direction of movement. Webster (2004) has beautifully designed this robust algorithm (GELS) which had overcome the following difficulties:

(i) increased number of iterations caused by characteristics of the search space

(ii) poor solutions caused by determination of objective function values of neighboring solutions.

Webster(2004) calculated objective function values of neighboring solutions as a function of the quality of those solutions as compared to earlier solutions examined by the algorithm. Because of this, objective function values of invalid solutions to the problem were tactically avoided from the beginning. He has also avoided the sensitive and redundant parameters of GLSA, because with these parameters finding the points of equilibrium was extremely difficult.

3.1 GELS Algorithm

Parameters used in the GELS algorithm

(a) Max velocity:

Defines the maximum value that any element within the pointer object velocity vector can have – used to prevent velocities that became too large to use.

(b) Radius:

Sets the radius value in the gravitational force formula – used to determine how quickly the gravitational force can increase or decrease.

(c) Iterations:

» Defines a number of iterations of the algorithm that will be allowed to complete before it is automatically terminated – used to ensure that the algorithm will terminate.

(d) Pointer:

Used to identify the current location and direction of movement of the pointer object within the search space.

Gravitational Force (GF)

The GELS algorithm uses the formula for the gravitational force between the two solutions as

$$F = \frac{G\ (CU - CA)}{R^2}$$

where,

G = 6.672 (Universal constant of gravitation)
CU = objective function value of the current solution
CA = objective function value of the candidate solution.
R = value of radius parameter

Webster's algorithm

Webster designed the GELS algorithm by using two methods and two stepping modes, which are:

GELS11:

» Computes the GF between a given single solution and the current solution

» Pointer object moves sequentially within the current local searchneighborhood

GELS12:

» Computes the GF between a given single solution and the current solution

» Pointer object can move to areas outside of the neighborhood

GELS21:

» Computes the GF among all solutions within the neighborhood

» Pointer object moves sequentially within the neighborhood

GELS22:

» Computes the GF among all solutions within the neighborhood

» Pointer object can move to areas outside of the neighborhood

During the development of GELS the default settings of the parameters was arrived at through trial and error. Some settings caused repetitive visits to the same solution within a neighborhood, and others generated large numbers, causing the algorithm to behave erratically. Webster, after a number of tests, settled the default values of the algorithm at 10 for maximum velocity, 4 for radius and 10000 for the maximum number of iterations. After a number of tests, we have settled the RGES algorithm at 7 for maximum velocity,7 for radius, and 1000 for iterations. The RGES algorithm and its explanation are given in section 5.

Distinguishing properties of GELS algorithm

The versions of the GELS algorithm have various distinguishing features from other algorithms like GA, SA, and HC in terms of search space, number of iterations, etc. In particular,

» Introduction of the velocity vector and relative gravitational force in each direction of movement emulated the acceleration effect of the process towards its goal.

» Multiple stepping procedures helped the solution pointer to show the next direction and amount of movement within the search space, leading to the other solutions.

» The algorithm is designed in such a way that it terminates on either of two conditions: either all the elements in the pointer object velocity vector have gone to zero, or the maximum allowable number of iterations has been completed.

Webster (2004) tested four different versions of the algorithms, and one of the items that differentiated the versions was the "stepping factor". As GELS executes, there is a pointer that is "moving" through the solution space, at a certain rate of "speed" and in a particular direction. The stepping factor related to the rate at which that speed could be altered. In two of the versions, if the pointer was affected by the gravitational influence of a nearby solution, then its speed could increase or decrease by only one unit at a time. In the remaining two versions, the speed could be increased or decreased by more than one unit at a time, depending on the strength of the gravitational influence(relative quality) of a nearby solution. In both cases, the algorithm stopped when the speed of its pointer dropped to zero(or after a specified maximum number of iterations, in order to prevent the possibility of a non-terminating execution).

4. Applications of GELS

Webster (2004)tested thesefour versions of GELS on three famous NP-hard problems, namelythe Traveling Salesman problem, the Bin Packing problem, and the File Assignment problem. Instances of these problems were randomly generated using several different problem sizes, then solved using three well-known comparison algorithms, namely Hill Climbing, Genetic Algorithm, and Simulated Annealing. The experimental results were rigorously analyzed using a variety of statistical techniques. Finally, Webster (2004) stated that with respect to the overall composite cases, Simulated Annealing and GELS 21 (method two with single stepping) were virtually tied in terms of performance.

Raja Balachandar (2007) has designed a new version of GELS called the Randomized Gravitational Emulation Search algorithm (RGES) for solving symmetric travelling salesman problems (STSP). This algorithm introducesa randomization concept along with the two of the four primary parameters, "velocity"and "gravity", in physics through swapping in terms of groups by using random numbers in the existing local search algorithm GELS in order to avoid local minima and thus can yield global minimum for STSP. Benchmark problems have been taken from OR-Library (Beasley,1990) to validate the performance of RGES and they havecompared with other heuristics,namely Genetic Algorithm (GA), Simulated Annealing (SA), and Hill Climbing(HC), in addition to the four variants of GELS. The overall comparative computational study of chosen benchmark problems shows that this RGES algorithm is an effectivetool for solving STSP.

RGES Procedure

1: Set the parameters(radius, velocity, direction of movement)
2: Set the current solution
3: While(termination condition are not met) do
-Construct neighborhood solutions based on direction of movement
-Compute gravitational force between current solution and
-best(candidate) solution in the neighborhood
-Update the velocity and direction of movement by using gravitational Force
End

The next application of GELS is in Grid Computing systems. Barzegar (2009) had used GELS to solve the scheduling and advance reservation of resources. The name of the algorithm offered for this problem is Gravitational Emulation Local Search Advanced Reservation algorithm (GELSAR) and was compared with GA to analyze the performance. The final results show that the accepted jobs by applying GELSAR has been increased 7.5

Pseudo-code for GELS

```
CurrentSolution = BestSolution = MonteCarloSolution
SolutionsExamined = 0
Iterationsremaining = Max Iterations Parameter
Velocity Sum = 0
For each Index in Velocity Vector
Velocity Vector [Index] = random integer between 1 and
Max Velocity Parameter
Velocity Sum = Velocity Sum  + Velocity Vector [Index]
end for
Direction = Maximum ValueIn (Velocity Vector)
while (Velocity Sum > 0 and Iterations Remaining > 0)
Generate Neighborhood (Current Solution)
if Method One selected
Candidate Solution = Neighborhood (Direction)
if Objective function (Candidate Solution) < Objective
function (Best Solution)
Best Solution = Candidate Solution
end if
Solutions Examined = Solutions Examined + 1
Force = Integer (6.672 * (Objective Function (Current
Solution ) – Objective solution     (Candidate solution)) /
Radius Parameter **2)
Velocity Vector [Direction] = Velocity Vector [Direction] +
Force
if Velocity Vector [Direction] < 0
Velocity Vector [Direction] = 0
end if
if Velocity Vector [Direction] > Max Velocity Parameter
Velocity Vector [Direction] = Max Velocity Parameter
end if
Velocity Sum = 0
for each index in Velocity Vector
Velocity Sum = Velocity Sum + Velocity Vector [Index]
end for
Direction = Maximum ValueIn (Velocity Vector)
else if Method Two Select
for each index in Neighborhood
Candidate Solution = Neighborhood (Index)
if Objective function (Candidate Solution) < Objective
function (Best Solution)
Best Solution = Candidate Solution
end if
Solutions Examined = Solutions Examined + 1
Force = Integer (6.672 * (Objective Function (Current
Solution ) – Objective solution     (Candidate solution)) /
Radius Parameter **2)
Velocity Vector [Index] = Velocity Vector [Index] + Force
if Velocity Vector [Index] < 0
Velocity Vector [Index] = 0
end if
if Velocity Vector [Index] > Max Velocity Parameter
Velocity Vector [Index] = Max Velocity Parameter
end if
end for
Velocity Sum = 0
for each index in Velocity Vector
Velocity Sum = Velocity Sum + Velocity Vector [Index]
end for
Direction = Maximum ValueIn (Velocity Vector)
end if
if SingleSteppingSelected
if TSPProblemBeingSolved
CurrentSolution = Neighborhood [random]
Else
CurrentSolution = Neighborhood [Direction]
End if

If ObjectiveFunction (CurrentSolution) <ObjectiveFunction
(BestSolution)
BestSolution = CurrentSolution
End if
SolutionsExamined = SolutionsExamined + 1
ElseifMultipleSteppingSelected
For 1 to VelocityVector[Direction]
If TSPProblemBeingSolved
CurrentSolution = Neighborhood [random]
Else
CurrentSolution = Neighborhood[Direction]
Endif
If ObjectiveFunction (CurrentSolution) <ObjectiveFunction
(BestSolution)
BestSolution = CurrentSolution
End if
SolutionsExamined = SolutionsExamined + 1
GenerateNeighborhood (CurrentSolution)
Endfor
Endif

IterationsRemaining = IterationsRemaining – 1

endWhile

return BestSolution, ObjectiveFunction (BestSolution), Solution-
sExamined
```

percent and the computation time of the new algorithm lowered down to 50 percent when compared to GA.

Procedure GELSAR

```
Number of solution dimensions is predefined number 'n'
 dim;   --- Dimension iteration counter
cnt   --- Loop iteration counter
begin
for dim =  1 to ndo
    --- Assign a predefined starting solution component as the current
solution component
        for
    --- dimension "dim" and as the best solution component seen thus
for that dimension
     --- Randomly  assign an initial velocity in the dimension "dim"
within the bounds of
       IVEL
end;
cnt = 0;
    --- Calculate an initial vector velocity sum, based on the random
initial velocity
      components
   ---- assigned in the previous step
while ( the velocity sum <> 0 ) and  ( cnt< ITER) do
   ---- Reset the velocity sum to 0
for dim=1 to ndo
   ---- calculate the solutions adjacent to the current solution adjacent
to the
       current solution and their respective RF values
   ---- If any of these is better than the best solution seen thus for, then
make
        that solution and
   ---- the new best solution
   ---- calculate the net difference in gravitational "force" between the
adjacent solutions  and
    ---- the current solution for the current dimension "dim", using the
Newtonian
         equation for
    ---- gravitational attraction
       ---- calculate change in acceleration for the current dimension
"dim"
     ----  Calculate change in velocity for the current dimension "dim"
      ---- Calculate new current solution component for the dimension
"dim", which
          will be the next
      --- adjacent node in the dimension "dim" in the current direction
of movement
         as indicated
     ---- by the velocity component for the dimension "dim"
 end;
     --- calculate the new velocity  sum
cnt = cnt +1;
end;
return best solution found, its RF value, and the iteration count ( cnt)
end
```

5. Gravitational Search Algorithm (GSA)

In this section, we employthe second version of a gravitational algorithm, namely the gravitational search algorithm (Rashedi, 2007) based on (1).

In physics, agents' performance is measured by their masses, and they are considered as objects. Based on gravitational force, the agents are attracted to each other, and their tendency is towards heavier masses. As a result, the heavy masses move in a manner much slower than the lighter ones, and this induces the exploitation.

Each and every mass (agent) has four specifications: position, inertial mass, active gravitational mass, and passive gravitational mass in GSA. The algorithm updates gravitational and inertia masses with the help of heavy masses and finds the optimum. This artificial world of masses obeys Newton's law of gravity and motion (Rashedi, 2007).Next, we present the GSA approach.

Now, consider a system with N agents (masses). We define the position of the ith agent by:

$$X_i = (x_i^1, ..., x_i^d, ..., x_i^n) \quad for\, i = 1, 2, 3 ....., N, \qquad (3)$$

wherex^d_i represents the position of i^{th} agent in the d^{th} dimension. At a specific time "t", we define the force acting on mass "i" from mass "j" as follows:

$$Fij^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}\left(xj^d(t) - xi^d(t)\right), \qquad (4)$$

where $M_{aj}$is the active gravitational mass related to agent j, $M_{pi}$is the passive gravitational mass related to agent i, G(t) is the gravitational constant at time t, εis a small constant, and R_{ij}(t) is the Euclidian distance between two agents iand j:

$$R_{ij}(t) = \left\| X_i(t), X_j(t) \right\|_2 \qquad (5)$$

To give a stochastic characteristic to our algorithm, we suppose that the total force that acts on agent iin a dimension d be arandomly weighted sum of the d_{th} components of the forces exerted by other agents:

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_i F_{ij}^{\ d}(t), \qquad (6)$$

where$rand_j$is a random number in the interval [0, 1]. Hence, by the law of motion the acceleration of the agent iat time t, and in direction d, $a^d_i$is given as follows:

$$a_i^{\ d}(t) = \frac{F_i^{\ d}(t)}{M_{ii}(t)}, \qquad (7)$$

where$M_{ii}$is the inertial mass of i^{th} agent.Furthermore, the next velocity of an agent is considered as a fraction of its current velocity added to its acceleration.Therefore, its position and its velocity could be calculated as follows:

$$v_i^{\ d}(t+1) = rand_i \times v_i^{\ d}(t) + a_i^{\ d}(t) \qquad (8)$$

$$x_i^{\ d}(t+1) = x_i^{\ d}(t) + v_i^{\ d}(t+1) \qquad (9)$$

where$rand_i$is a uniform random variable in the interval [0, 1]. We use this random number to give a randomized characteristicto the search.The gravitational constant, G, is initialized at the be-

ginning and will be reduced with time to control the search accuracy.In other words, G is a function of the initial value (G0) and time (t):

$$G(t) = G(G_0, t) \quad (10)$$

Gravitational and inertia masses are simply calculated by the fitness evaluation. A heavier mass means a more efficientagent. This means that better agents have higher attractions and walk more slowly. Assuming the equality of the gravitationaland inertia masses, the values of masses are calculated using the map of fitness. We update the gravitational and inertialmasses by the following equations:

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \quad (11)$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}, \quad (12)$$

$$M_{ai} = M_{pi} = M_{ii}\ M_i,\ i = 1,2,3.....N, \quad (13)$$

where$fit_i(t)$ represent the fitness value of the agent iat time t, and, worst(t) and best(t) are defined as follows for a maximization problem,

$$best(t) = \max_{j \in \{1,...,N\}} fit_j(t), \quad (14)$$

$$worst(t) = \min_{j \in \{1,...,N\}} fit_j(t), \quad (15)$$

One way to perform a good compromise between exploration and exploitation is to reduce the number of agents with lapseof time in Eq. (6). Hence, we propose only a set of agents with bigger mass apply their force to the other. However, we shouldbe careful of using this policy because it may reduce the exploration power and increase the exploitation capability.

In order to avoid local optima trapping the algorithm, we must use the exploration at beginning. Bylapse of iterations, exploration must fade out and exploitation must fade in. To improve the performance of GSA by controllingexploration and exploitation only the Kbestagents will attract the others. Kbestis a function of time, with the initial valueK_0 at the beginning and decreasing with time. In such a way, at the beginning all agents apply the force, and as timepassesKbestis decreased linearly. At the end, there will be just one agent applying force to the others. Therefore, Eq. (6) could be modified as:

$$F_i^d(t) = \sum_{j \in Kbest,\ j \neq i} rand_i F_{ij}^d(t), \quad (16)$$

whereKbestis the set of the first K agents with the best fitness value and biggest mass.

Distinguishing properties of GSA

To see how the proposed algorithm is efficient some remarks are noted:

- Since each agent can observe the performance of the others, the gravitational force is an information-transferring tool.
- Due to the force that acts on an agent from its neighborhood agents, it can see space around itself.
- A heavy mass has a large effective attraction radius and hence a great intensity of attraction. Therefore, agents with ahigher performance have a greater gravitational mass. As a result, the agents tend to move toward the best agent.
- The inertia mass is against the motion and make the mass movement slow. Hence, agents with heavy inertia mass moveslowly and hence search the space more locally. So, it can be considered as an adaptive learning rate.
- The gravitational constant adjusts the accuracy of the search, so it decreases with time (similar to the temperature in a Simulated Annealing algorithm).
- GSA is a memory-less algorithm. However, it works efficiently like the algorithms with memory. Our experimental results show a good convergence rate of the GSA.

Here, we assume that the gravitational and inertia masses are the same. However, for some applications different values for each can be used. A bigger inertia mass provides a slower motion of agents in the search space and hence a moreprecise search. Conversely, a bigger gravitational mass causes a higher attraction of agents, which permits a fasterconvergence.

6. Application of GSA

Rashedi(2009) first tested the GSA with 23nonlinear benchmark problems (Yao et al., 1999).The details of all the benchmark problems are also presented in(Rashedi,2009), alongwith their optimum solutions. They have also compared the performance of GSA with other heuristics, namely Particle Swarm Optimization (PSO), Central Force Optimization (CFO), and Real Genetic Algorithm (RGA). They have proven that GSA reached superior results in most cases and in all cases are comparable with PSO, RGA, and CFO.

GSA Procedure

a)Search space identification.
b)Randomized initialization.
c)Fitness evaluation of agents.
d)Update G(t), best(t), worst(t) and Mi(t) for i= 1,2,. . .,N.
e)Calculation of the total force in different directions.
f)Calculation of acceleration and velocity.
g)Updating agents' position.
h)Repeat steps c to h until the stop criteria is reached.
i)End.

Recently, Raja Balachandar(2009,2010) designed a binary version of GSA for solving the set covering problem(SCP) and the vertex covering problem(VCP). These two problems are 0-1 integer programming problems. These problems are already proved as NP-hard COP. To convert the infeasible solutions generated by GSA a problem-specific operator (called repair operator)has been designed for each problem. Benchmark problems have been taken from the OR-Library (Beasley, 1990) to check the performance of GSA, and to validate the computational efficiency of GSA comparisons have been made with other recent heuristics that are available in the literature and report the efficiency of their proposed algorithm.

Revised GSA Algorithm

a)Search space identification.
b)Randomized initialization.
c)Repair operator design.
d)Fitness evaluation of agents.
e)Update G(t), best(t), worst(t) and Mi(t) for i= 1,2,. . .,N.
f)Calculation of the total force in different directions.
g)Calculation of acceleration and velocity.
h)Updating agents' position.
i)Repeat steps c to h until the stop criteria is reached.
j)End.

Sarafrazi(2011) developed a new operator-based GSA called Improved Gravitational Search Algorithm(IGSA) to improve the exploration and exploitation abilities of the standard Gravitational SearchAlgorithm (GSA). A novel operator called "Disruption", originating from astrophysics, was proposed. Thedisruption operator is inspired by nature and, with the least computation, has improved the ability ofGSA to further explore and exploit the search space. The proposed improved GSA has been evaluated on23 nonlinear benchmark functions(Yao et al., 1999)and compared with standard GSA, the genetic algorithm and particleswarm optimization. It was proved that theproposed method confirmed the high performance insolving various nonlinear functions from obtained results, Another version of GSA is known asthe Binary Gravitational Search Algorithm (BGSA), which was designed by Rashedi(2010). This proposed algorithm was also tested on 23 nonlinear benchmark problems(Yao et al., 1999), and results have been compared with GA and Binary Particle Swarm Optimization (BPSO) to prove the efficiency of BGSA.

Binod Shaw(2012) presented an opposition-based gravitational search algorithm for combined economic and emission dispatch problems of power systems. The novelty of this algorithm is to accelerate the performance ofthe GSA. The proposed opposition-based GSA (OGSA) of the present work employs opposition-basedlearning for population initialization and for generation jumping to improve the convergence rate of the GSA. The author has tested the proposed algorithm on a comprehensive set of 23 complex benchmark test functions(Yao et al., 1999). In addition, four standard power systems problems of combined economicand emission dispatches (CEED) are solved by the OGSA to establish the optimizing efficacy of theproposed algorithm. The results obtained confirm the potential and effectiveness of the proposed algorithmcompared to some other algorithms that have surfaced in the recent state-of-the art literature.

IGSA Procedure

a)Search space identification.
b)Randomized initialization.
c)Fitness evaluation of agents.
d)Update G(t), best(t), worst(t) and Mi(t) for i= 1,2,. . .,N.
e)Calculation of acceleration and velocity.
f)Updating agents' position.
g)Disruption operator application.
h)Repeat steps c to h until the stop criteria is reached.
i)End.

Clustering is used to group data objects into sets of disjoint classes called clusters so that objects withinthe same class are highly similar to each other and dissimilar from the objects in other classes. K-harmonicmeans (KHM) is one of the most popular clustering techniques, and has been applied widelyand works well in many fields. But, this method usually runs into local optima easily. Minghao Yin(2011)proposed a hybrid data clusteringalgorithm based on an improved version of Gravitational Search Algorithmand KHM, calledIGSAKHM. The author proved that IGSAKHM helps theKHM clustering escape from local optima, but also overcomes the slow convergence speed of the IGSA. The proposed method was compared with some existing algorithms on seven data sets, and the obtainedresults indicate that IGSAKHM is superior to KHM and PSOKHM in most cases.

Abbas (2012)used GSA for the classification of instances in multi-class data sets. The author's proposed method employs GSA as a global searcher to find the best positions of the representatives (prototypes). The proposed GSA-based classifier was used to test for data classification of some of the well-known benchmark sets. Its performance was compared with the artificial bee colony (ABC), PSO, and nine other classifiers from the literature. The experimental results of twelve data sets from the UCI machine learning repository confirmedthe effectiveness and efficiency of the proposed method, and that the GSA can successfully be applied as to classification problems.

Rashedi(2011)discussedanewlinearandnonlinearfiltermodelingbasedonGSA. In this paper, the unknownfilterparameterswereconsideredasavector to beoptimized.Examples ofinfiniteimpulseresponse(IIR)filter design,aswellasrational nonlinear filter,weregiven.TheeffectivenessoftheproposedGSA-basedfiltermodeling was tested on differentsetsof initial populationswiththepresenceofdifferentmeasurablenoises. GAandPSOwerealsousedtomodelthesame examplesandsomes-

imulationresultsarecompared.Obtainedresultsconfirmedtheefficiencyofthe proposedmethod and that it is well suited to solve complex problems of this nature.

Chaoshun Li (2011)developed a GSA-based new optimization algorithm for parameters identification. The parameter identification of hydraulic turbine governing systems (HTGS) is crucial in precise modeling of hydropower plants and provides support for the analysis of the stability of power systems. In this paper, the authorintroduced GSA,showed its application to parameter identification of HTGS, and improved this version of GSA throughcombining it with the search strategy of particle swarm optimization. Furthermore, a new weighted objective function was proposed in the identification frame. The improved gravitationalsearchalgorithm (IGSA), together with GA, PSO and GSA, was employed in parameter identification experiments, and the procedure was validated by comparing experimental and simulated results. Consequently, IGSA was shown to locate more precise parameter values than the compared methods, with higher efficiency.

Behrang (2011) presented a novel approach for oil consumption modeling to forecast future oil demand in Iran. Growing energy demands in the world required the major oil and gas exporting countries to play a critical role in the energy supply. The geostrategic situation of Iran and its access to huge hydrocarbon resources placed the country among important areas and resulted in the development of the oil and gas industries. Three demand estimation models were developed to forecast oil consumption based on socio-economic indicators, using GSA.In the first model (PGIE), oil consumption was estimated based on population, gross domestic product (GDP), and import and export. In the second model (PGML) population, GDP, export minus import, and number of LDVs (light-duty vehicles) were used to forecast oil consumption. Lastly, in the third model (PGMH) population, GDP, export minus import, and number of HDVs (heavy-duty vehicles) were used to estimate oil consumption. Linear and nonlinear forms of equations were developed for each model.To show the accuracy of the algorithm, the author made a comparison with the GA andPSO estimation models which were developed for the same problem.Finally, oil demand in Iran was forecasted up to year 2030.

SoroorSarafrazi(2013)hybridized the GSA with support vector machine (SVM), and made a novel GSA-SVM hybrid system to improve classification accuracy with an appropriate feature subset in binary problems. To optimize the input feature subset selection and the SVM parameter setting, a discrete GSA was combined with a continuous-valued GSA in this system. The author evaluated the proposed hybrid system on several UCI machine learning benchmark examples, and the results showed that the proposed approach is able to select the discriminating input features correctly, and it is also able to achieve high classification accuracy which is comparable to or better than well-known similar classifier systems.

7. Convergence analysis

In this section we present that RGES converges with probability 1 to the globaloptimal solution. In order to describe the RGES, we need the following definitions:

- A neighborhood structure is a mapping N from M into total (M is afinite set, i.e. for each solution x it defines a neighborhood N(x) of x and each$y \in N(x)$ is called a neighbour of x. In the case of the STSP, the neighborhoodof a given tour x can be defined as the set of tours which can be generated by theRGES.
- A generation mechanism is a rule of selecting a solution y from the neighborhood N(x) of given solution x. In the context of RGES such a generationrule is usually called an updation rule. The generation mechanism can be described by probability matrix R such that

$$R(x, y) = P\{X_{t+1} = y \mid X_t = x\} \quad (17)$$

  - where$X_t$denotes the state of the system at time(iteration) t. Clearly $R(x, y) > 0$if and only if $y \in N(x)$. By(17) a Markov chain is defined over the set M offeasible solutions. However, in order to solve STSP this Markov chain has to bemodified by some acceptance criterion so that "good solutions" are selected moreoften (i.e. with higher probability) than bad ones.
- A local optimal solution is an $x \in M$ such that $F(x) \leq F(y)$ for all $y \in N(x)$.
- The global optimal solution is defined by $x \in M$ such thatF(x) ≤ F(y) for all $y \in M$. For the RGES not to get stuck in a local optimum (whichis not globally optimal) it is necessary to accept also deteriorations of the objectivefunction with probability. A state y is reachable from state x if there exists $z_1, z_2, ..., z_m \in M$ such that $z_1 \in N(x)$, $z_2 \in N(z_1)$,..., $y \in N(z_m)$.

The algorithm starts from an initial candidate solution set $X_0 = \{x^1_0, ..., x^p_0\}$.In each iteration t, from the current parent candidate solutionset X_t a group ofcandidate solutions X_t of c candidate solutions is generated by updation, where atleast one child is generated by an updation. Then the following selection process isapplied:

(a) Select x as best of all the p+c solutions $x^1_t, ..., x^p_t, x_t^{1'}, ..., x_t^{c'}$.

(b) Select x^2_{t+1} arbitrarily among all the candidate solutions $x_t^{1'}, ..., x_t^{c'}$which have not already been selected in step (a).

(c) Select $x^3_t, ..., x^p_{t+1}$ if $p > 2$ by any selection rule.

(c1) Select those candidate solutions (not already selected) with thebest values of the objective function.

(c2) Select those solutions (candidate solutions not already selected)with the best value of the objective function.

(c3) Select p-2 solutions arbitrary among the candidate solutions (notalready selected).

(c4) Select p-2 solutions arbitrary among the candidate solutions (notalready selected).

From the classical literature of Markov processes we can now formulate the following global convergence result:

Theorem 7.1

Let the set of feasible solutions M be finite and assume, that forall x, y ∈ M the state y is reachable from x by the updations considered. Thenthe RGES with selection rule (a) to (c) in any of the variants (c1)-(c4) has the followingproperty:

limP{At least one solution in Xt is globally optimal} = 1.

The proof Theorem 7.1 is trivial.

8. Conclusion

In this paper, we have shown the main characteristics of two algorithms based on the law of gravity andapplied them to the solution of some well-known NP-hard COPs. In future, we will analyze the practical results of various problems or new NP-hard problems and theoretical framework, and make a commercial package with these algorithms and study of computational complexity and related properties. In addition, we will extend our work to the design of "hybrid" algorithms or parallelism that will contain combinations of the ideas of the methods presented here.

9. References:

1. Abbas Bahrololoum, HosseinNezamabadi-pour, Hamid Bahrololoum, Masoud Saeed, (2012).A prototype classifier based on gravitational search algorithm. Applied Soft Computing.12(2),819-825.
2. Badr, A. and Fahmy, A.(2004).A proof of convergence for ant algorithms. Information Sciences 160, 267-279.
3. Barzegar, B., Rahmani,A.M. and Far Kz(2009). Gravitational emulation local search algorithm for advanced reservation and scheduling in grid computing systems.fourth international conference on computer sciences and convergence information technology, 1240-1245.
4. Beasley, J.E.(1990). OR-Library:Distributing Test Problems by Electronic Mail. Journal of Operational Research Society 41, 1069-1072.
5. Behrang, M.A., Assareh, E., Ghalambaz, M., Assari, M.R, andNoghrehabadi, A.R (2011). Forecasting future oil demand in Iran using GSA (Gravitational Search Algorithm). Energy, 36(9), 5649-5654.
6. Bergh, F.V.D. andEngelbrecht,A.P., (2006).A study of particle swarm optimization particle trajectories.Information Sciences, 176, 937-971.
7. Binod Shaw, Mukherjee, V. and Ghoshal, S.P.(2012). A novel opposition-based gravitational search algorithm for combined economic and emission dispatch problems of power systems. Electrical Power and Energy Systems, 35(1), 21-33.
8. Chaoshun Li and Jianzhong Zhou (2011).Parameters identification of hydraulic turbine governing system using improved gravitational search algorithm. Energy Conversion and Management, 52(1), 374-381.
9. Dorigo, M., Maniezzo, V.and Colorni, A. (1996).The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics – Part B, 26(1),29-41.
10. Ellabib, I., Calamai, P. and Basir, O.(2007).Exchange strategies for multiple ant colony system. Information Sciences, 177, 1248-1264.
11. Glover,F.(1989).Tabu search, part I. ORSA, Journal on Computing, 1(3), 190-206.
12. Glover, F.(1990).Tabu search, part II. ORSA, Journal on Computing, 2, 4-32.
13. Hamzaçebi, C.(2008).Improving genetic algorithms' performance by local search for continuous function optimization. Applied Mathematics and Computation, 196(1), 309-317.
14. Holliday, D, Resnick, R. and Walker, J.(1993). Fundamentals of physics. John Wiley and Sons.
15. Hopfield,J. (1982).Neutral networks and physical systems with emergent collective computational abilities.proc.Natl. Acad.Sci.USA,Vol.81,3088-3092.
16. Kirkpatrick, S., Gelatto, C.D. andVecchi, M.P.(1983). Optimization by simulated annealing. Science, 220, 671-680.
17. Kennedy, J.and Eberhart, R.C.(1995). Particle swarm optimization.in: Proceedings of IEEE International Conference on Neural Networks, 4, 1942-1948.
18. Lozano, M., Herrera, F. and Cano, J.R.(2008). Replacement strategies to preserve useful diversity in steady-state genetic algorithms. Information Sciences, 178, 4421-4433.
19. Minghao Yin, Yanmei Hu, Fengqin Yang, Xiangtao Li, and WenxiangGu(2011). A novel hybrid K-harmonic means and gravitational search algorithm approach for clustering. Expert Systems with Applications, 38, 9319-9324.
20. Raja,Balachandar S. andKannan, K.(2007) Randomized gravitational emulation search algorithm for symmetric traveling salesman problem. Applied Mathematics and Computation, 192,413-421.
21. Raja,Balachandar S.and Kannan, K.(2009). A Meta-heuristic algorithm for Vertex covering problem Based on Gravity. International Journal of Computational and Mathematical Sciences, 3(7), 332-336.
22. Raja,Balachandar S. and Kannan, K.(2010). A Meta-heuristic algorithm for Set covering problem Based on Gravity. International Journal of Computational and Mathematical Sciences. 4(5), 223-228.

23. Rashedi, E.(2007). Gravitational Search Algorithm, M.Sc. Thesis. ShahidBahonar University of Kerman, Kerman, Iran.

24. Rashedi,E.et al.(2009). GSA: A Gravitational Search Algorithm. Information Sciences, 179, 2232-2248.

25. Rashedi, E., HosseinNezamabadi-pour and SaeidSaryazdi (2010).BGSA: binary gravitational search algorithm. Natural Computing, 9, 727-745.

26. Rashedi, E., HossienNezamabadi-pour and SaeidSaryazdi(2011). Filter modeling using gravitational search algorithm. Engineering applications of Artificial Intelligence, 24, 117-122.

27. Righini,G. (1992).Modelli di retineurali per otimisizzazione combination.RicercaOperativa, 62, 29-67.

28. Sarafrazi, S., Nezamabadi-pour, H.and Saryazdi, S.(2011). Disruption: A new operator in gravitational search algorithm.ScientiaIranica, Transactions D: Computer Science & Engineering and Electrical Engineering, 18, 539-548.

29. Schutz, B.(2003). Gravity from the Ground Up, Cambridge University Press.

30. Sears,Francis W., Mark W.Zemansky and Hugh D. Young(1987). University Physics,7th ed.Reading, MA. Addison-Wesley.

31. SoroorSarafrazi andHosseinNezamabadi-pour (2013). Facing the classification of binary problems with a GSA-SVM hybrid system. Mathematical and Computer Modeling, 57(1–2), 270-278.

32. Tang, K.S., Man, K.F., Kwong, S. and He, Q. (1996). Genetic algorithms and their applications. IEEE Signal Processing Magazine, 13(6), 22-37.

33. Tripathi, P.K., Bandyopadhyay, S. and Pal, S.K. (2007). Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients. Information Sciences, 177, 5033-5049.

34. Voudouris,Chris and Edward Tsang(1995), "Guided Local Search". Technical Report CSM-247, Department of Computer Science,University of Essex,UK.

35. Webster, B. L.(2004).Solving combinatorial optimization problems using a new algorithm based on gravitational attraction, Ph.D. thesis, Florida Institute of Technology, Melbourne, FL.

36. Yao, X., Liu, Y. and Lin, G.(1999).Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation, 3, 82-102