

# A Meta-Heuristic algorithm for Set covering problem Based on Gravity

S. Raja Balachandar and K.Kannan

*Abstract*—A new Meta heuristic approach called "Randomized gravitational emulation search algorithm (RGES)" for solving large size set covering problems has been designed. This algorithm is found upon introducing randomization concept along with the two of the four primary parameters 'velocity' and 'gravity' in physics. A new heuristic operator is introduced in the domain of RGES to maintain feasibility specifically for the set covering problem to yield best solutions. The performance of this algorithm has been evaluated on a large set of benchmark problems from OR-library. Computational results showed that the randomized gravitational emulation search algorithm - based heuristic is capable of producing high quality solutions. The performance of this heuristic when compared with other existing heuristic algorithms is found to be excellent in terms of solution quality.

*Keywords*—Set covering Problem, Velocity, Gravitational Force, Newton's Law, Meta Heuristic, Combinatorial optimization.

## I. INTRODUCTION

There is a class of problems, whose exponential complexities have been established theoretically are known as NP problems. Designing polynomial time algorithms for such a class of problems is still open. Due to the demand for solving such problems, Researchers are constantly attempting to provide heuristic solutions one after the other focusing the optimality by introducing several operators with salient features such as (i) reducing the computational complexity, (ii) randomization etc.,

Some NP problems are Set covering problem, Traveling salesman problem, Problem of Hamiltonian paths, Knapsack problem, Problem of optimal graph coloring. If a polynomial time solution can be found for any of these problems, then all of the NP problems would have polynomial solutions. NP complete problems are described more detail in [15].

The set covering problem (SCP) is a main and fundamental model for several important applications. Crew scheduling problem in bus, railway and mass-transit transportations companies where a given set of trips has to be covered by a minimum - cost set of pairings, a pairing being a sequence of trips that can be prepared by a single crew [9] are worth mentioning. Though both exact (optimal) and heuristic approaches have been presented in the literature, this problem is still a difficult NP-complete problem.

The set covering problem (SCP) is the problem of covering the rows of this m-row, n-column, zero-one matrix  $(a_{ij})$  by

a subset of the columns at minimal cost. Defining  $x_j = 1$  if column j (with cost  $c_j > 0$ ) is the solution and  $x_j = 0$  otherwise.

It can be formulated as a binary integer program as follows: minimize

$$\sum_{j=1}^n c_j x_j \quad (1)$$

subject to

$$\sum_{j=1}^n a_{ij} x_j \geq b_j, i = 1, 2, \dots, m \quad (2)$$

$$x_j \in \{0, 1\}, j = 1, 2, 3, \dots, n \quad (3)$$

Equation (2) ensures that each row is covered by at least one column and (3) is the integral of constraint. The cost coefficients  $c_j$  are equal to 1 the problem is referred to as the unicast SCP, otherwise, the problem is called the weighted or non-unicast SCP. The SCP has been proved to be NP - complete [15].

In this paper, a new optimization algorithm based on the law of gravity, namely Randomized gravitational emulation search algorithm (RGES) is proposed. This algorithm is based on the Newtonian gravity: "Every particle in the universe attracts every other particle with a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between them".

This article demonstrates that RGES technique is capable of producing better quality results for the large size set covering problem than other heuristic approaches.

This paper is organized as follows: A brief survey of various approaches pertaining to this problem is elucidated in section II. In section III, we introduce the basic concepts of our algorithm. The proposed RGES is presented in section IV. The algorithm's utility is illustrated with help of benchmark problems in section V and we include the extensive comparative study of result of our heuristic with existing state-of-art heuristics. Salient features of this algorithm are enumerated in section VI, finally concluding remarks are given in section VII.

## II. PREVIOUS WORK

The SCP with arbitrary positive costs is NP-hard [15]. Several exact and heuristic approaches to solve SCPs have been reported to literature.

S.Raja Balachandar is with the Department of Mathematics, SASTRA University, Thanjavur, INDIA, e-mail: srbala@maths.sastra.edu

K.Kannan is with the Department of Mathematics, SASTRA University, Thanjavur, INDIA, e-mail: kkannan@maths.sastra.edu

Existing exact algorithms are essentially based on branch-and-bound and branch-and-cut. Fisher and Kedia [14] proposed an exact branch-and-bound algorithm based on a dual heuristic and able to solve instances with 200 constraints upto 2000 variables. Beasley combined a Lagrangian-based heuristic, feasible solution exclusion constraints, Gomory's f-cuts, to improve the branching strategy for strengthening his algorithm [4]. This algorithm could solve instances with constraint matrices upto the order of 400 X 4000 [6]. Harche and Thompson [19] developed an exact algorithm, called "column subtraction", which is capable of solving sparse instances of set covering problems. These optimal algorithms are based on tree-search algorithm. Since exact methods have limitations such as 'suffering for optimality', very heavy computational efforts, very large search space etc. Researchers started desiring approximation algorithms to meet the needs of less computation with high quality. The heuristic approaches can be divided into two main categories.

The first one exploits problem characteristics and specific features of each instance. Examples include Lagrangian relaxation-based procedures, sub gradient optimization methods[5], the relaxed dual model exploitation[10], and surrogate optimization [25]. Greedy algorithms is found to be the first natural heuristic approach for solving large size combinatorial problems. As for the SCP, the simplest approach is the greedy algorithms [12]. Although simple, fast and easy to code; Greedy algorithms could rarely generate solution of good quality as a result of their myopic and deterministic nature. Researchers have attempted to improve greedy algorithms by introducing randomization concept. These randomized or probabilistic greedy algorithms [32, 13, 18] often generate better results than pure greedy ones.

The second category includes local search procedures and the adaptation of meta heuristics to the SCP, such as genetic algorithm[7,31,1], ant colony algorithms[24], Simulated annealing algorithms[21,22], Neural Network algorithms[27], as well as specifically tailored local search procedure[34],but the quality of meta heuristic approaches using some features from the first category of heuristics, and the late appearances of a highly effective local search procedure make this category a competitive approach. Due to the unicost version specific characteristics, some specific heuristics have been developed for the unicost case, and some general heuristics have been adapted to the unicost case viz . Alminana and Poster adaptation [2], of a Lops and Lorena proposal [25] heuristics based on Lagrangian relaxations and the Surrogate problems solutions have been tested for solving 60 newly generated random instances and 5 literature based instances. Grossman and Wool [16], has designed neural network architecture (ANN) to solve unicost SCP and shown the superiority of ANN over the other heuristic algorithms available in literature. In this paper, we have designed a meta heuristic algorithm based on gravity and we enhanced the performance of RGS through feasibility operator to obtain best solutions at less computational cost.

### III. THE LAW OF GRAVITY

The gravitation is the tendency of masses to accelerate toward each other. It is one of the four fundamental inter-

actions in nature [29] (the others are: the electromagnetic force, the weak nuclear force, and the strong nuclear force). Every particle in the universe attracts every other particle. Gravity is everywhere. The inescapability of gravity makes it different from all other natural forces. The way Newton's gravitational force behaves is called "action at a distance". This means gravity acts between separated particles without any intermediary and without any delay. In the Newton law of gravity, each particle attracts every other particle with a 'gravitational force' [3,20,28,29,30,33]. The gravitational force between two particles is directly proportional to the product of their masses and inversely proportional to the square of the distance between them [20]:

$$F = \frac{GM_1M_2}{R^2} \quad (4)$$

where F is the magnitude of the gravitational force, G is gravitational constant,  $M_1$  and  $M_2$  are the mass of the first and second particles respectively, and R is the distance between the two particles. Newton's second law says that when a force, F, is applied to a particle, its acceleration, a, depends only on the force and its mass, M [20]:

$$a = \frac{F}{M} \quad (5)$$

Based on (4) and (5), there is an attracting gravity force among all particles of the universe where the effect of bigger and the closer particle is higher. An increase in the distance between two particles means decreasing the gravity force between them. In addition, due to the effect of decreasing gravity, the actual value of the "gravitational constant" depends on the actual age of the universe. Eq. (6) gives the decrease of the gravitational constant, G, with the age [26]:

$$G(t) = G(t_o) \times \left(\frac{t_o}{t}\right)^\beta, \beta < 1, \quad (6)$$

where G(t) is the value of the gravitational constant at time t.  $G(t_o)$  is the value of the gravitational constant at the first cosmic quantum-interval of time  $t_o$  [26]. Three kinds of masses are defined in theoretical physics:

Active gravitational mass,  $M_a$ , is a measure of the strength of the gravitational field due to a particular object. Gravitational field of an object with small active gravitational mass is weaker than the object with more active gravitational mass.

Passive gravitational mass,  $M_p$ , is a measure of the strength of an object's interaction with the gravitational field. Within the same gravitational field, an object with a smaller passive gravitational mass experiences a smaller force than an object with a larger passive gravitational mass.

Inertial mass,  $M_i$ , is a measure of an object resistance to changing its state of motion when a force is applied. An object with large inertial mass changes its motion more slowly, and an object with small inertial mass changes it rapidly. Now, considering the above-mentioned aspects, we rewrite Newton's laws. The gravitational force,  $F_{ij}$ , that acts on mass i by mass j, is proportional to the product of the active gravitational of mass j and passive gravitational of mass i, and inversely proportional to the square distance between them.  $a_i$  is proportional to  $F_{ij}$  and inversely proportional to inertia

mass of i. More precisely, one can rewrite Eqs. (4) and (5) as follows:

$$F_{ij} = \frac{GM_{aj}M_{pi}}{R^2}, \quad (7)$$

$$a_i = \frac{F_{ij}}{M_{ii}}, \quad (8)$$

where  $M_{aj}$  and  $M_{pi}$  represent the active gravitational mass of particle i and passive gravitational mass of particle j, respectively, and  $M_{ii}$  represents the inertia mass of particle i.

Although inertial mass, passive gravitational mass, and active gravitational mass are conceptually distinct, no experiment has ever unambiguously demonstrated any difference between them. The theory of general relativity rests on the assumption that inertial and passive gravitational mass are equivalent. This is known as the weak equivalence principle [23]. Standard general relativity also assumes the equivalence of inertial mass and active gravitational mass; this equivalence is sometimes called the strong equivalent principle [23].

#### IV. RANDOMIZED GRAVITATIONAL EMULATION SEARCH ALGORITHM(RGES)

In this section, we introduce our optimization algorithm based on the law of gravity [28]. In the proposed algorithm, agents are considered as objects and their performance is measured by their masses. All these objects attract each other by the gravity force, and this force causes a global movement of all objects towards the objects with heavier masses. Hence, masses cooperate using a direct form of communication, through gravitational force. The heavy masses - which correspond to good solutions - move more slowly than lighter ones, this guarantees the exploitation step of the algorithm. In RGES, each mass (agent) has four specifications: position, inertial mass, active gravitational mass, and passive gravitational mass. The position of the mass corresponds to a solution of the problem, and its gravitational and inertial masses are determined using a fitness function. In other words, each mass presents a solution, and the algorithm is navigated by properly adjusting the gravitational and inertia masses. By lapse of time, we expect that masses be attracted by the heaviest mass. This mass will present an optimum solution in the search space. The RGES could be considered as an isolated system of masses. It is like a small artificial world of masses obeying the Newtonian laws of gravitation and motion. More precisely, masses obey the following laws:

Law of gravity: each particle attracts every other particle and the gravitational force between two particles is directly proportional to the product of their masses and inversely proportional to the distance between them, R. We use here R instead of  $R^2$ , because according to our experiment results, R provides better results than  $R^2$  in all experimental cases.

Law of motion: the current velocity of any mass is equal to the sum of the fraction of its previous velocity and the variation in the velocity. Variation in the velocity or acceleration of any mass is equal to the force acted on the system divided by mass of inertia.

#### A. Initiation

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

$$X_i = (x_i^1, x_i^2, \dots, x_i^d, \dots, x_i^n) \text{ for } i = 1, 2, 3, \dots, N, \quad (9)$$

where  $x_i^d$  presents the position of ith agent in the dth dimension. At a specific time 't', we define the force acting on mass 'i' from mass 'j' as following:

$$F_{ij}^d(t) = G(t) \frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \epsilon} (x_i^d(t) - x_j^d(t)), \quad (10)$$

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 gravitational constant at time t,  $\epsilon$  is a small constant, and  $R_{ij}(t)$  is the Euclidian distance between two agents i and j:

$$R_{ij} = \|X_i(t), X_j(t)\|_2, \quad (11)$$

To give a stochastic characteristic to our algorithm, we suppose that the total force that acts on agent i in a dimension d be a randomly weighted sum of dth components of the forces exerted from other agents:

$$F_i^d(t) = \sum_{j=1, j \neq i}^N rand_j F_{ij}^d(t), \quad (12)$$

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

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}, \quad (13)$$

where  $M_{ii}$  is the inertial mass of ith 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:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t), \quad (14)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), \quad (15)$$

where  $rand_i$  is a uniform random variable in the interval [0, 1]. We use this random number to give a randomized characteristic to the search. The gravitational constant, G, is initialized at the beginning and will be reduced with time to control the search accuracy. In other words, G is a function of the initial value ( $G_o$ ) and time (t):

$$G(t) = G(G_o, t), \quad (16)$$

### B. Evaluation of fitness and updating

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

$$M_{ai} = M_{pi} = M_{ii} = M_i, i = 1, 2, 3, \dots, N, \quad (17)$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \quad (18)$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^N m_j(t)}, \quad (19)$$

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

$$best(t) = \min_{j \in \{1, \dots, N\}} fit_j(t), \quad (20)$$

$$worst(t) = \max_{j \in \{1, \dots, N\}} fit_j(t), \quad (21)$$

One way to perform a good compromise between exploration and exploitation is to reduce the number of agents with lapse of time in Eq. (12). Hence, we propose only a set of agents with bigger mass apply their force to the other. However, we should be careful of using this policy because it may reduce the exploration power and increase the exploitation capability. We remind that in order to avoid trapping in a local optimum the algorithm must use the exploration at beginning. By lapse of iterations, exploration must fade out and exploitation must fade in. To improve the performance of RGENS by controlling exploration and exploitation only the Kbest agents will attract the others. Kbest is a function of time, with the initial value  $K_o$  at the beginning and decreasing with time. In such a way, at the beginning, all agents apply the force, and as time passes, Kbest is decreased linearly and at the end there will be just one agent applying force to the others. Therefore, Eq.(12) could be modified as:

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i} rand_j F_{ij}^d(t), \quad (22)$$

where Kbest is the set of first K agents with the best fitness value and biggest mass.

### C. Repair operator

The solutions(agents) may violate constraints. To make all the solutions feasible an additional operator is needed. Here a proposed heuristic operator consists of two phases namely ADD phase and DROP phase that maintains the feasibility of the solutions in the neighborhood being generated. The steps required to make each solution feasible involve the identification of all uncovered rows and the addition of

columns such that all rows are covered. This is done by the ADD phase. Once columns are added, a solution becomes feasible. DROP phase (a local optimization procedure) is applied to remove any redundant column such that by removing it from the solution, the solution still remains feasible. In the algorithm, steps (i) and (ii) identify the uncovered rows and add the least cost column to the solution vector. Steps (iii) and (iv) identify the redundant column with high cost and dropped from the solution. The time complexity of this repair operator is  $O(mn)$ .

Different steps of the repair operator are the followings

- $S_{1 \times n}$  = solution vector
- $B_{n \times m}$  = transpose of the constraint matrix
- $D_{1 \times n}$  = temporary solution vector
- $C_{1 \times m}$  = counter vector ( 0 entry of any position is used to identify the uncovered rows)
- (i)  $C = S \times B$  ( matrix multiplication)
- (ii) ADD Phase
  - (a) For each 0 entry in  $C$  , find the first column  $j$ ( cost of  $j$  is in increasing order)
  - (b) Add  $j$  to  $S$  ie.,  $S(j) = 1$ .
  - (c)  $D = S$  ( temporary )
- (iii)DROP Phase
  - (a) Identify the column  $j$  ( cost in the decreasing order)
  - (b) Remove  $j$  from  $D$ , if  $C = D \times B$  have no zero entry, ie.,  $D(j) = 0$ .
  - (c)  $S=D$  is a feasible solution for SCP that contains no redundant columns.

The different steps of the proposed RGENS algorithm are the followings:

- (a) Search space identification.
- (b) Randomized initialization.
- (c) Repair operator.
- (d) Fitness evaluation of agents.
- (e) Update  $G(t)$ ,  $best(t)$ ,  $worst(t)$  and  $M_i(t)$  for  $i = 1, 2, \dots, N$ .
- (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.

## V. EXPERIMENTAL RESULTS AND ANALYSIS

This heuristic is tested on 65 SCP test instances from Beasley's OR library. The instances are divided into 11 sets as in Table I, in which 'Density' is the percentage of non zero entries in the SCP matrix. Each of types 4 and 5 has 10 instances, each of types 6 and A-H has five instances. Problem sets 4-6 and A-D are the ones for which optimal solution values are known. Problem sets E, F, G and H are large size SCPs for which optimal solution values are not known.

In our experimental study, 10 trials RGENS heuristic were made for each of the test problems. In all the cases, population size and dimension are set to  $n$  and maximum iteration is 1000. This algorithm was implemented in C and tested in P-IV,

TABLE I  
TEST PROBLEM DETAILS

| Problem set | Number of rows | Number of columns | Density | Number of problems |
|-------------|----------------|-------------------|---------|--------------------|
| 4           | 200            | 1000              | 2       | 10                 |
| 5           | 200            | 2000              | 2       | 10                 |
| 6           | 200            | 1000              | 5       | 5                  |
| A           | 300            | 3000              | 2       | 5                  |
| B           | 300            | 3000              | 5       | 5                  |
| C           | 400            | 4000              | 2       | 5                  |
| D           | 400            | 4000              | 5       | 5                  |
| E           | 500            | 5000              | 10      | 5                  |
| F           | 500            | 5000              | 20      | 5                  |
| G           | 1000           | 10000             | 2       | 5                  |
| H           | 1000           | 10000             | 5       | 5                  |

3.2GHz processor and 512 MB RAM running under Windows XP. Table II exhibits computational results with the following details: Instance: The name of the test problem appearing in the Beasley's ORLIB the first digit/alphabet indicating the name of the problem set and floating digit representing the problem number.

Opt: The number of trials out of 10 in which the RGEs has found the optimum solution /best known value.

Best: The number of trails out of 10 in which the RGEs found its best solution. Here it is worth mentioning that for problems for which optimal solutions are available in literature, the best solutions of proposed algorithm are equal to optimum solutions.

Average Execution Time: Average execution time of RGEs algorithm for 10 trials.

Mean, Min, Max: The mean, min and maximum objective values returned in the 10 trials (the value column) and the respective percentages above the optimal value (in pct column)

We can observe that the RGEs found optimal solutions for all the instances. For 55 of the problems the RGEs found the optimal solution/best known solution in every trial. The heuristic to return consistent solutions for smaller size problems, for large size problems it returns solutions that vary a little bit best higher (but close to each other in objective value). The average gap between RGEs solution and optimum / best known is 0.015.

< place table2 about here >

In order to bring out the efficiency of the proposed RGEs algorithm the solutions of the same set of test instances have been compared with the other heuristic and Meta heuristic algorithms (Simulated annealing, Genetic algorithm, Lagrangian heuristic, Greedy, 3 flip neighborhood). Table III provides a summary of the solution quality for these different heuristics namely average gap (average = (solution - BKS)/BKS x 100), number of optimum solutions and best solutions. RGEs found the optimal / best-known solutions for all the 65 test instances. From this table, we can observe that RGEs, CFT, and Meta-RaPS have zero deviation from the best-known or optimal solutions for these test problems.

The abbreviations mentioned in Table 3 stands for: BJT: Simulated annealing by Brusco, Jacobs and Thom-

TABLE II  
PERFORMANCE OF RGEs ALGORITHM

| Ins  | opt | best | Avg Exec Time | Mean  |       | Min |     | Max |      |
|------|-----|------|---------------|-------|-------|-----|-----|-----|------|
|      |     |      |               | Val   | pct   | Val | pct | Val | pct  |
| 4.1  | 10  | 10   | 189.5         | 429   | 0.0   | 429 | 0.0 | 429 | 0.0  |
| 4.2  | 10  | 10   | 182.0         | 512   | 0.0   | 512 | 0.0 | 512 | 0.0  |
| 4.3  | 10  | 10   | 179.6         | 516   | 0.0   | 516 | 0.0 | 516 | 0.0  |
| 4.4  | 10  | 10   | 188.2         | 494   | 0.0   | 494 | 0.0 | 494 | 0.0  |
| 4.5  | 10  | 10   | 183.8         | 512   | 0.0   | 512 | 0.0 | 512 | 0.0  |
| 4.6  | 10  | 10   | 185.0         | 560   | 0.0   | 560 | 0.0 | 560 | 0.0  |
| 4.7  | 10  | 10   | 185.9         | 430   | 0.0   | 430 | 0.0 | 430 | 0.0  |
| 4.8  | 10  | 10   | 181.1         | 492   | 0.0   | 492 | 0.0 | 492 | 0.0  |
| 4.9  | 10  | 10   | 189.3         | 641   | 0.0   | 641 | 0.0 | 641 | 0.0  |
| 4.10 | 10  | 10   | 184.6         | 514   | 0.0   | 514 | 0.0 | 514 | 0.0  |
| 5.1  | 10  | 10   | 195.7         | 253   | 0.0   | 253 | 0.0 | 253 | 0.0  |
| 5.2  | 10  | 10   | 194.0         | 302   | 0.0   | 302 | 0.0 | 302 | 0.0  |
| 5.3  | 10  | 10   | 198.3         | 226   | 0.0   | 226 | 0.0 | 226 | 0.0  |
| 5.4  | 10  | 10   | 192.0         | 242   | 0.0   | 242 | 0.0 | 242 | 0.0  |
| 5.5  | 10  | 10   | 199.2         | 211   | 0.0   | 211 | 0.0 | 211 | 0.0  |
| 5.6  | 10  | 10   | 193.8         | 213   | 0.0   | 213 | 0.0 | 213 | 0.0  |
| 5.7  | 10  | 10   | 194.7         | 293   | 0.0   | 293 | 0.0 | 293 | 0.0  |
| 5.8  | 10  | 10   | 197.9         | 288   | 0.0   | 288 | 0.0 | 288 | 0.0  |
| 5.9  | 10  | 10   | 198.0         | 279   | 0.0   | 279 | 0.0 | 279 | 0.0  |
| 5.10 | 10  | 10   | 191.6         | 265   | 0.0   | 265 | 0.0 | 265 | 0.0  |
| 6.1  | 10  | 10   | 190.4         | 138   | 0.0   | 138 | 0.0 | 138 | 0.0  |
| 6.2  | 10  | 10   | 187.5         | 146   | 0.0   | 146 | 0.0 | 146 | 0.0  |
| 6.3  | 10  | 10   | 193.7         | 145   | 0.0   | 145 | 0.0 | 145 | 0.0  |
| 6.4  | 10  | 10   | 194.0         | 131   | 0.0   | 131 | 0.0 | 131 | 0.0  |
| 6.5  | 10  | 10   | 188.8         | 161   | 0.0   | 161 | 0.0 | 161 | 0.0  |
| A1   | 10  | 10   | 207.8         | 253   | 0.0   | 253 | 0.0 | 253 | 0.0  |
| A2   | 10  | 10   | 210.0         | 252   | 0.0   | 252 | 0.0 | 252 | 0.0  |
| A3   | 10  | 10   | 204.1         | 232   | 0.0   | 232 | 0.0 | 232 | 0.0  |
| A4   | 10  | 10   | 208.9         | 234   | 0.0   | 234 | 0.0 | 234 | 0.0  |
| A5   | 10  | 10   | 206.6         | 236   | 0.0   | 236 | 0.0 | 236 | 0.0  |
| B1   | 10  | 10   | 211.1         | 69    | 0.0   | 69  | 0.0 | 69  | 0.0  |
| B2   | 10  | 10   | 207.2         | 76    | 0.0   | 76  | 0.0 | 76  | 0.0  |
| B3   | 10  | 10   | 209.8         | 80    | 0.0   | 80  | 0.0 | 80  | 0.0  |
| B4   | 10  | 10   | 213.0         | 79    | 0.0   | 79  | 0.0 | 79  | 0.0  |
| B5   | 10  | 10   | 205.4         | 72    | 0.0   | 72  | 0.0 | 72  | 0.0  |
| C1   | 10  | 10   | 222.2         | 227   | 0.0   | 227 | 0.0 | 227 | 0.0  |
| C2   | 10  | 10   | 226.0         | 219   | 0.0   | 219 | 0.0 | 219 | 0.0  |
| C3   | 10  | 10   | 215.9         | 243   | 0.0   | 243 | 0.0 | 243 | 0.0  |
| C4   | 10  | 10   | 228.6         | 219   | 0.0   | 219 | 0.0 | 219 | 0.0  |
| C5   | 10  | 10   | 224.8         | 215   | 0.0   | 215 | 0.0 | 215 | 0.0  |
| D1   | 10  | 10   | 219.4         | 60    | 0.0   | 60  | 0.0 | 60  | 0.0  |
| D2   | 10  | 10   | 225.0         | 66    | 0.0   | 66  | 0.0 | 66  | 0.0  |
| D3   | 10  | 10   | 227.7         | 72    | 0.0   | 72  | 0.0 | 72  | 0.0  |
| D4   | 10  | 10   | 224.1         | 62    | 0.0   | 62  | 0.0 | 62  | 0.0  |
| D5   | 10  | 10   | 228.0         | 61    | 0.0   | 61  | 0.0 | 61  | 0.0  |
| E1   | 10  | 10   | 229.9         | 29    | 0.0   | 29  | 0.0 | 29  | 0.0  |
| E2   | 10  | 10   | 237.9         | 30    | 0.0   | 30  | 0.0 | 30  | 0.0  |
| E3   | 10  | 10   | 228.5         | 27    | 0.0   | 27  | 0.0 | 27  | 0.0  |
| E4   | 10  | 10   | 231.4         | 28    | 0.0   | 28  | 0.0 | 28  | 0.0  |
| E5   | 10  | 10   | 234.8         | 28    | 0.0   | 28  | 0.0 | 28  | 0.0  |
| F1   | 10  | 10   | 230.6         | 14    | 0.0   | 14  | 0.0 | 14  | 0.0  |
| F2   | 10  | 10   | 234.1         | 15    | 0.0   | 15  | 0.0 | 15  | 0.0  |
| F3   | 10  | 10   | 235.8         | 14    | 0.0   | 14  | 0.0 | 14  | 0.0  |
| F4   | 10  | 10   | 231.5         | 14    | 0.0   | 14  | 0.0 | 14  | 0.0  |
| F5   | 10  | 10   | 236.0         | 13    | 0.0   | 13  | 0.0 | 13  | 0.0  |
| G1   | 8   | 10   | 268.7         | 176.4 | 0.004 | 176 | 0.0 | 179 | 0.03 |
| G2   | 9   | 10   | 255.3         | 154.1 | 0.001 | 154 | 0.0 | 155 | 0.01 |
| G3   | 9   | 10   | 264.9         | 166.2 | 0.002 | 166 | 0.0 | 168 | 0.02 |
| G4   | 10  | 10   | 268.5         | 168   | 0.0   | 168 | 0.0 | 168 | 0.0  |
| G5   | 9   | 10   | 272.4         | 168.1 | 0.001 | 168 | 0.0 | 169 | 0.01 |
| H1   | 10  | 10   | 266.0         | 63    | 0.0   | 63  | 0.0 | 63  | 0.0  |
| H2   | 10  | 10   | 259.6         | 63    | 0.0   | 63  | 0.0 | 63  | 0.0  |
| H3   | 8   | 10   | 261.8         | 59.2  | 0.002 | 59  | 0.0 | 60  | 0.01 |
| H4   | 9   | 10   | 267.4         | 58.1  | 0.001 | 58  | 0.0 | 59  | 0.01 |
| H5   | 10  | 10   | 273.0         | 55    | 0.0   | 55  | 0.0 | 55  | 0.0  |

son[8],BC: genetic algorithm by Beasley and Chu[7], Be: the Lagrangian heuristic by Beasley [5], Grdy: Greedy heuristic for set-covering problem[12], CNS: Lagrangian-based heuristic by Ceria, Nobili and Sassano [11], CFT: Lagrangian-based heuristic by caprara, Fischetti and Toth [10], MMT: 3-flip neighborhood local search by Mutsunori Yagiura, Masahiro Kishida and Toshihide Ibaraki [34], Meta-RaPS - effective and simple heuristic approach by Guanghai, Gail and Gary [17].

VI. FEATURES OF ALGORITHM

To see how the proposed algorithm is efficient some remarks are noted: Since each agent could 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

TABLE III  
SUMMARIZED RESULTS FOR THE SOLUTION QUALITY

| Prob set                       | BJT  | BC   | Be   | Gry   | CFT  | Meta-RaPS | RGES |
|--------------------------------|------|------|------|-------|------|-----------|------|
| 4                              | 0.00 | 0.00 | 0.06 | 3.78  | 0.00 | 0.00      | 0.00 |
| 5                              | 0.00 | 0.09 | 0.18 | 5.51  | 0.00 | 0.00      | 0.00 |
| 6                              | 0.00 | 0.00 | 0.56 | 7.72  | 0.00 | 0.00      | 0.00 |
| A                              | 0.00 | 0.00 | 0.82 | 5.61  | 0.00 | 0.00      | 0.00 |
| B                              | 0.00 | 0.00 | 0.81 | 5.57  | 0.00 | 0.00      | 0.00 |
| C                              | 0.00 | 0.00 | 1.93 | 6.88  | 0.00 | 0.00      | 0.00 |
| D                              | 0.00 | 0.00 | 2.75 | 9.79  | 0.00 | 0.00      | 0.00 |
| E                              | 0.00 | 0.00 | 3.5  | 12.75 | 0.00 | 0.00      | 0.00 |
| F                              | 0.00 | 0.00 | 7.16 | 12.98 | 0.00 | 0.00      | 0.00 |
| G                              | 0.13 | 0.13 | 4.83 | 8.49  | 0.00 | 0.00      | 0.00 |
| H                              | 0.32 | 0.63 | 8.12 | 11.78 | 0.00 | 0.00      | 0.00 |
| Over all gap                   | 0.04 | 0.07 | 2.36 | 8.21  | 0.00 | 0.00      | 0.00 |
| Total                          | 65   | 65   | 65   | 65    | 65   | 65        | 65   |
| Opt /best in atleast one trial | 65   | 61   | 22   | 0     | 65   | 65        | 65   |

a large effective attraction radius and hence a great intensity of attraction. Therefore, agents with a higher 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 move slowly and hence search the space more locally. So, it can be considered as an adaptive learning rate. Gravitational constant adjusts the accuracy of the search, so it decreases with time (similar to the temperature in a Simulated Annealing algorithm). RGES is a memory-less algorithm. However, it works efficiently like the algorithms with memory. Our experimental results show the good convergence rate of the RGES. Here, we assume that the gravitational and the inertia masses are the same. However, for some applications different values for them can be used. A bigger inertia mass provides a slower motion of agents in the search space and hence a more precise search. Conversely, a bigger gravitational mass causes a higher attraction of agents. This permits a faster convergence.

## VII. CONCLUSION

A feasibility operator based heuristic for the large size set covering problem based on RGES has been developed. Randomization enables the algorithm to escape from the local search and pave a way leading to find optimal solutions. Computational results indicate that our heuristic is able to generate optimal solutions for small size problems in less time. For large size problems the deviation from the optimal solutions are very less and are much below the deviations obtained by other existing algorithms.

## REFERENCES

- [1] Aickelin, U., An indirect genetic algorithm for set covering problems, *Journal of the Operational Research Society* 53, 1118-1126,2002.
- [2] Almiana M, Pastor JT, An adaptation of SH heuristic to the location set covering problem, *European Journal of Operational Research*; 100(3):586-93,1997
- [3] Barry Lynn Webster, Solving combinatorial Optimization Problems using a new algorithm based on gravitational attraction, Ph.D., thesis, Melbourne, Florida Institute of Technology, May 2004.
- [4] Beasley J.E, An algorithm for set covering problems, *European Journal of Operational Research* 31 85-93,1990
- [5] Beasley J.E, A Lagrangean heuristic for set covering problems, *Naval Research Logistics* ; 37(1):151-64,1990.
- [6] Beasley J.E., Jornsten. K, Enhancing an algorithm for set covering problems, *European Journal of Operational Research* 58, 293-300,1992.
- [7] Beasley J.E, Chu PC, A genetic algorithm for the set covering problem, *European Journal of Operational Research*; 94(2):392-404,1996.
- [8] Brusco M.J.,Jacobs L.W.,Thomson G.M, A morphing procedure to supplement a simulated annealing heuristic for cost-and-coverage-correlated set-covering problem, *Annals of Operations Research* 86, 611-627,1999.
- [9] Caprara A. , Fischetti M. Toth P, D.vigo and Guida P.L., Algorithms for railway crew management ,*Mathematical programming* 79 , 125 - 141,1997.
- [10] Caprara A, Fischetti M, Toth P, A heuristic method for the set covering problem, *Operational Research*; 47(5):730-743,1999.
- [11] Ceria S., Nobili P., Sassano A, A Lagrangian-based heuristic for large-scale set covering problems, *Mathematical Programming* 81, 215-228,1998.
- [12] Chvatal. V, A greedy heuristic for the set-covering problem,*Mathematics of Operations Research* 4, 233-235,1979.
- [13] Feo, T., Resende, M.G.C, A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters* 8, 67-71,1989.
- [14] Fisher M.L., Kedia P, Optimal solutions of set covering/partitioning problems using dual heuristics, *Management Science* 36, 674-688,1990.
- [15] Garey M.R. and Johnson D.S, *Computers and Intractability: A Guide to the theory of NP-completeness* , W.H.Freeman ,San Francisco,1979.
- [16] Grossman T,Wool A, Computational experience with approximation algorithms for the set covering problem. *European Journal of Operational Research*; 101(1):81-92,1997.
- [17] Guanghui Lan, Gail W. DePuy, Gary E. Whitehouse, An effective and simple heuristic for the set covering problem, *European Journal of Operational Research* 176, 1387-1403,2007.
- [18] Haouari, M., Chaouachi, J.S., A probabilistic greedy search algorithm for combinatorial optimization with application to the setcovering problem, *Journal of the Operational Research Society* 53, 792-799,2002.
- [19] Harche E and Thompson G.L., The column subtraction algorithm: An exact method for solving weighted setcovering, packing and partitioning problems, *Computers and Operations Research* 21, 689-705,1994.
- [20] D. Holliday, R. Resnick, J. Walker, *Fundamentals of physics*, John Wiley and Sons, 1993.
- [21] Jacobs L.W. and Brusco M.J.,A simulated annealing based heuristic for the set-covering problem, Working paper, Operations Management and Information Systems Department, Northern Illinois University, DeKalb, IL, 1993.
- [22] Jacobs, L., Brusco, M., Note: A local-search heuristic for large set-covering problems, *Naval Research Logistics* 42, 1129-1140,22,1995.
- [23] I.R. Kenyon, *General Relativity*, Oxford University Press, 1990.
- [24] Lessing L, Dumitrescu I, Stzle T, A comparison between ACO algorithms for the set covering problem, *Lecture Notes in Computer Science*; 3172;1-12,2004.
- [25] Lopes FB, Lorena LA, Surrogate heuristic for set covering problems, *European Journal of Operational Research*; 79(1):138-150,1994.
- [26] R. Mansouri, F. Nasser, M. Khorrami, Effective time variation of G in a model universe with variable space dimension, *Physics Letters* 259,194-200,1999.
- [27] Ohlsson, M., Peterson, C., Soderberg, B., An efficient mean field approach to the set covering problem, *European Journal of Operational Research* 133, 583-595,2001.
- [28] E. Rashedi, *Gravitational Search Algorithm*, M.Sc. Thesis, Shahid Bahonar University of Kerman, Kerman, Iran, 2007.
- [29] B. Schutz, *Gravity from the Ground Up*, Cambridge University Press, 2003.
- [30] Sears,Francis W., Mark W.Zemansky and Hugh D. Young, *University Physics*, 7th ed.Reading,MA.Addison - Wesley,1987.
- [31] Solar M, Parada V, Urrutia R., A parallel genetic algorithm to solve the set-covering problem, *Computer Operational Research*; 29(9):1221-35,2002.
- [32] Vasko, F.J., Wilson, G.R., An efficient heuristic for large set covering problems, *Naval Research Logistics Quarterly* 31, 163-171,1984.
- [33] Voudouris,chris and Edward Tsang, *Guided Local Search*, Technical Report,CSM-247,Department of Computer Science,University of Essex, UK,1995.
- [34] Yagiura M, Kishida M, Ibaraki T, A 3-flip neighborhood local search for the set covering problem, Technical Report 2004-001. Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University,2004