ADDIS ABABA UNIVERSITY

COMPUTATIONAL SCIENCE PROGRAM

MASTER THESIS

Systematic Global Optimization Algorithm for a general Multilevel Multi-follower Stackelberg Nash Problems with bounded decision Range

Supervisor: Mengistu Goa Sangago (PhD)

Associate Professor of Mathematics

Author: First Name : Yonas

Family Name : Dagne Yigletu

In partial fulfillment of the requirements
for the degree of Master of science
in Computational Science

June 12, 2018

Addis Ababa , Ethiopia

Table of Contents

Declaration

1. Candidate's Declaration

I, Yonas, hereby certify that the work presented in this thesis is, to the best of My knowledge and belief that it is original and in addition to as acknowledged in the text that the material has not been submitted either in whole or in part for a degree at this or any other university.

I was admitted as a graduate student in 2014/15 and as a candidate for the degree of master of science in computational science; the higher study which this is a record carried out in the Addis Ababa University.

I acknowledge that I have read and understood the University's rules, requirements, procedures and policy relating to my higher degree research award and to my thesis.

I certify that I have complied with the rules, requirements, procedures and policy of the University.

Yonas Dagne Yigletu

Name of Candidate Signature Date

2. Supervisor's Declaration

I hereby certify that the candidate has fulfilled the condition of the Resolution and Regulations appropriate for the degree of graduate study in Addis Ababa University and that the candidate is qualified to submit this thesis in application for that degree.

Name of supervisor	signature	Date
1.		
2.		
3.		

Acknowledgement

Scale up process processes in non cooperative environment is very challenging specially in academics. Hence, I would like to thank to those who had been participating and interfering in both explicit and implicit manner to the development of this thesis.

So, I would like to thank again for all of you, who are being cooperative rather than non cooperative, and also to the creator of the universe.

Abstract

Today, many decision makers decide a decision which is hard to accept and could affect or leave to unpleasant situation to those who follows them in structural arrangements. This is because of challenge of solving Structural multi level interdependent set of objects. And most of computational model has challenge for finding the optimal result of each object that considerer all the constraints and limitations of each decision maker in both computational time and decision support quality. This thesis develops a computational optimization computerized model based on root finding technique for a bounded decision range in a non-cooperative environment to reduce computational time and improve decision support quality.

Scope

The scopes of this research thesis are;

- Bi-level optimization model
- Operational analysis on optimization model in both sequential and parallel computational operation
- Analysis to other optimization models which had been developed from the point of view of decision support system
- Root finding technique in optimization model development
- Global optimization technique

Objective

The objective of this thesis is to show the global optimization algorithm provide a platform for solving Stackelberg Nash structural optimization problem in parallel computation. And create access for the development of optimization models that are focused on the root fining technique to locate the optimal result of multi level multi follower structural problems for both Stackelberg and Nash concept problems. In addition to those points, this thesis elaborates the usage of this optimization model for decision support systems that are either constrained or non constrained optimization multi level problems.

List of Tables

List of Figures

Introduction

Operations of a set of duties or responsibilities are recorded in a data set. Those data sets of results are generated from a decision which is taken among the alternative set of choices or strategies and those decisions are considered as it is optimal which considers the limitations and constraints of the objective function.

Many organizational decisions are made in a multi- level hierarchical structure, and each decision level is considered that it has no direct control over the decisions made by the others. The actions taken at one-decision level could affect those taken at all other levels as interest rate affects the price of products which uses an imported product by that interest rate. This hierarchical decision process arises in many fields, including decentralized resource planning, highway pricing, the power market, logistics, economics, manufacturing, and road network management and so on.

A decentralized non cooperative multilevel decision system is model in which one leader and several followers of equal status are involved. the assumptions that, the leader and followers have their own decision variables and objective functions with constraint function, and the leader can only influence (rather than dictate) the reactions of followers through his own decision variables, while the followers have full authority to decide how to optimize their objective functions in a consideration of the decisions of the leader and other followers below its level.

Decision entities of multilevel multi-follower are characterized by the three

hierarchical levels which are respectively termed as the top-level leader, the middle level follower and the bottom-level follower. The decision entities make their individual decisions in sequence, from the top level to the middle level and then to the bottom level with the aim of optimizing their respective objectives. Specifically, the leader gives priority to making a decision; however, this decision is implicitly determined by the actions of the followers. The middle-level follower then reacts to the decision made by the leader and optimizes its own objective function while taking into account the implicit reactions of the bottom-level follower. Lastly, in response of the given decisions from the bottom-level and middle levels, the top-level decision makers decide to optimize its own objective function.

This thesis design a systematic global optimization algorithm for solving Stackelberg-Nash equilibrium of general multilevel programming with multiple followers. And numerical example are provided to illustrate the effectiveness of the proposed algorithm.

Chapter 1 Literature Review on Optimization Techniques

1.1. Background

The techniques that have been mentioned for unconstrained optimization are golden search method[5], quadratic approximation method[5], Nelder-Mead Method[5], Steepest Descent Method[5], Newton Method[5], Conjugate Gradient Method[5],Simulated Annealing Method[5] and Genetic Algorithm[5].

In the constrained optimization, Lagrange Multiplier Method and Penalty Function Method [5], Karush–Kuhn–Tucker (KKT) conditions[5], Integer Programming[5],simplex[5] are the known optimization techniques.

So far research publication have showed algorithms that solves two level, three level, and multi-level structured optimization problems. Mostly and wildly known bi-level optimal searching techniques are simplex,. For Tri-level optimization, *K*th-best algorithm for linear tri-level programming [4], Systematic evolutionary algorithm for general multi-level optimization.[3]

In the above proposed algorithms none of them have tried to give description of algorithms to solve multilevel programming problems explicitly for general non linear functions, but all of them are done for a special types of problems specified on their assumptions and no one have proposed for a general multilevel multi follower with parallel computation. The algorithm in this thesis is a systematic Global optimization Strategy (in which the decision space of each level is divided to some number of regions and the global optimization selects the one which is the best fit

for region and sends those points of each region to followers to be optimized.)

First, the leader's level is solved alone for all the variables in the problem under all the constraint set, as if it controls all the variables and assumes all the other level decision makers are on their minimum decision range. Then by fixing the leader's decision, the second level problem is solved alone for all the variables at bottom level and finally the bottom level computed. This process continues until optimal values at top level decision maker selects its best optimal decision.

1.2. Mathematical formulation of Multilevel Multi follower Stackelberg Nash Problem

Stackelberg problems are a concept of the leader firm moves first and the follower firms moves next. If there are more than one leader or followers with equal status in each level, the Nash equilibrium concept that each decision maker is assumed to know the equilibrium strategies of the other decision maker, and no decision maker has anything to gain by changing only their own strategy. So this optimizes the equilibrium states of each firm or decision makers. The combined effect of Stacklberg and Nash equilibrium concept provide the multi level multi follower structured optimization problem.

In Multi level structured object optimization, assume that in a decentralized multi-level decision system there are one leader and m followers in k number of levels. In each level, there are leaders for respective followers. Let X be a set of control variable of both a leader and each level decision maker.

$$X = x_1, x_2, x_3, \ldots\ldots\ldots\ldots\ldots, x_k \qquad (1)$$

Where k is the representation of each level and x is a value of decision maker

And let a function F be set of objective function of a leader and follower

which are interrelated by the control variable X. the optimization result of a function F could be either maximization or minimization over the decision space S that is the Union set of each decision space domain of decision makers.

Therefore $X \subset S$ (2)

The optimization minimization function F is

$F(X) \leq F(x_i)$ For all combination of $x_i \in S$ (3)

Subjected to constraint/s $g(x) \leq 0$

The optimization maximization function F is

$F(X) \geq F(x_i)$ For all combination of $x_i \in S$ (4)

Subjected to constraint/s $g(x) \leq 0$

Where F is a vector-valued function of decision vector x, Then for each decision x chosen by the leader, the feasible set Y of control array $(y_1, y_2, y_3, \ldots\ldots\ldots, y_m)$ of followers m is dependent on x.

Assuming that the leader first chooses his control vector x of S, and the followers determine their control array $(y_1, y_2, y_3, \ldots\ldots\ldots, y_m)$. Then a general type of multi level programming formulated as follows,

The general this multi-follower multi-level optimization model is represented

$$F_1(x_1, y_1, y_2, \ldots\ldots\ldots, y_k)$$

$$F_2(x_1, y_1, y_2, \ldots\ldots\ldots, y_k)\, y_1 = x_2$$

$$F_3(x_1, y_1, y_2, \ldots\ldots\ldots, y_k)\, y_2 = x_3$$

$$F_{n-1}(x_1, y_1, y_2, \ldots\ldots\ldots, y_k)\, y_3 = x_{n-1}$$

$$F_n(x_1, y_1, y_2, \ldots\ldots\ldots, y_k)\, y_k = y_{n-1} \qquad (5)$$

Where k is last level for each function F there are be an equity or inequality constrained function g(x)

The top level function send control variable x to next function which trigger the next level function to make an optimal response while considering the other lower level functions are in their optimal value state. This process continues to the last level of functions and return back to its respective level to make an optimal change and then reach to first level function which maximizes or minimizes its function. In this process the whole system of equation are being optimized to its new state for the first function either maximization or minimization (implicit cooperative) While other level functions are their optimal best response level.

For functions more than one at each level, each function compute their best response simultaneously either for maximization or minimization while knowing the strategy of other functions in its group to reach to a stationary point which all functions agree implicitly. The last optimization result will be a non cooperative interaction of functions of a level. So the agreement of each function domain value will be the Representative optimal value for the next level function/s for example summation or results (non cooperative).

Chapter 2 Stackglburg and Nash Multi level multi-follower optimization Model

This optimization model is a deterministic model and generates access for top level decision maker to decide on decision options for either maximization or minimization optimization problem. For this purpose, the model accepts input values or parameters and returns the optimal decision of the leader function.

The proposed optimization model is designed for efficiently searching an approximate stackelberg-Nash equilibrium solution of having a common constraint set S, that contains the minimum and maximum values of the decision variables. The model searches stackelberg Nash solutions of a given problem in a hierarchical way. Within a computational iteration, the leader observes the decision of others by passing different values of the variable which is under its control. It is called those values, lookout of the leader. Similarly for each value passed by the Dms(decision makers) above their level, the other Dm observe the decisions of those below their level by passing their own lookout. For each value obtained from above, by recording the results gained from each lookout of their own, the Dms in the interior of the model selects a best lookout value of their interest. The leader Dm records the feedback of its entire lookout and selects the best of all based on its criteria.

2.1 The characteristic natures of the computational model

1. The feasible range of the decision space is categorized into a set of points of odd and even interval for decision range represented by integer number system.
2. Each group of data set, that is either odd or even, are grouped to a number of sub regions that compare to each other to find optimal representative of all region.
3. The optimal stackglburg Nash model solution of decision space for multi level

single decision maker at each level is a combination of choices or lookout in a feasible region of decision space of each level that maximizes or minimizes the outcome value of the leader function.

4. At each iteration the stackglburg Nash model optimal solution converges to the ideal solution. And the reaction tolerance between leader and follower becomes to an acceptable range.
5. Each iteration, which is different from the 1st iteration, the payoff value of the leader function is bounded to a range that satisfies the given tolerance. At each iteration, that starts from the second iteration processes the algorithm within bounded converged payoff range with updated lower boundary lookout values
6. Each sub region of decision maker function representative optimal lookout is the result of a continuous recursive iterative operation that selects the most optimal lookout value within the region. when this lookout is optimal to its region and can be seen as the only representative of its region and also for global region, then the lookout is high probable candidate for optimal decision
7. Whenever a stackglburg Nash model solution of odd and even distribution becomes in acceptable tolerance, then the model returns the final stackglburg nash model optimal solution.
8. When a number of Dms are greater than one in a single level, the Nash approximate equilibrium solution is a continuous recursive optimal stackglburg multi-level which becomes constant to each level function at n^{th} or final iteration.
9. If the representation of a data point set or population is represented by more than one co-ordinate points, the optimal lookout value is the result of stackglburg model which has an equal number of level to the number of co-ordinate points.

2. 2 Systematic and Fair distribution of lookouts on decision space S

Since the number of lookouts that is used to look for optimal solution affect the computation time and fair representation of number of data points in a decision space, a systematic and a fair distribution of lookouts technique is necessary to

justify the number of lookouts of each level to represent data points efficiently.

Out of the total given number of lookouts, the system distributes the total number of candidate lookouts to each decision makers on each level. So the maximum number of lookouts on a single space of X_j ,where j is function level, are selected as follows:- To be fair on distributions of lookout in a single iteration, the proposed algorithm consider the following k levels Where k is the total number of hierarchical level. The total number of lookouts path is less or equal to the Cartesian product of each level number of lookouts

$$\prod_{j=1}^{k} \leq \alpha \qquad (6)$$

Where α total lookout path

The difference of decision space density between the leader number of representative lookouts and all followers of leader representative lookouts must be approximately less or equal to zero.

For the fair distribution of lookouts to each level and follower/s region, the number of lookouts of each decision maker is directly proportional to the decision space they have that is d which is decision range of each Dms. Hence, the number representative lookouts on the leader level or j=1 region is calculated as follows

$$\frac{a_1}{d_1} - \frac{a_j}{d_j} \leq 0 \qquad \text{j=2, 3,.....,k} \qquad (7)$$

The decision levels number of representative lookouts of the leader is calculated as

$$a_1 \leq \left(\frac{\alpha * {d_1}^{k-1}}{d_{tot} - d_1} \right)^{\frac{1}{k}} \qquad (8)$$

Where α is the total number of lookout paths

The decision levels number of representative lookouts below the leader is calculated as

$$a_j \leq d_j \left[\frac{a_1}{d_1} \right] \tag{9}$$

For level that has more than one follower, the number of representative lookouts of the region is divided to each region based on the ratio they contributed to the decision space of that region.

The optimal value of the decision maker is confirmed from two perspectives that the optimal point must come from both odd and even distribution within a given reaction tolerance. so if the value is located from two different and opposite ends to common converged point, the solution is approximate optimal result. For this purpose, the number of lookouts of each level decision maker range is divided into two that is odd and even categories.

$$b_j = \left[\frac{a_j}{2} \right] \tag{10}$$

For j = 1, 2,3..... k, the step-size of odd and even distributions; on each decision maker dimension space interval $\triangle$ of either odd or even range is calculated as follows:-

$$\triangle_j = \frac{d_j}{2 b_j} \tag{11}$$

where the range of the bounded decision space is divided in to b_j intervals of step-size j and each interval contains two subintervals containing an odd and even distributed lookouts on X_j. sub interval or the number of populations exists in each

odd and even region is calculated by dividing the interval region of each to the given number of population to be in a single region or the maximum iterations of the search.

$$\partial_{subinterval} = \frac{\triangle_j}{maxiter} \tag{12}$$

maxiter is the number of population in each set of region.

2. 3 Global Optimization

It is an optimization technique that differs from local optimization by its focus on finding the maximum or minimum value from over all input values, as opposed to finding local minimum or maximum.

let a set of cluster be $C \subset c_1, c_2, c_3, \ldots\ldots\ldots\ldots\ldots c_{n-1}, c_n$

And global optimization is represented by Gopt. The global optimization result X is represented by

$$G_{opt}(X) = ._{max/min}\left(C_1(x), C_2(x), C_3(x), ,\ldots\ldots\ldots, C_{n-1}(x), C_n(x)\right) \tag{13}$$

where x is a result of domain range of a function of cluster C and yields either maximum or minimum of optimization function by global optimization technique among the set of cluster

The main objective of the model is to find a distributed most min value of each region of the leader function and to send those lookouts to the followers and observe the reaction of the follower. Finally to select the one that maximizes or minimizes the leader payoff value. But having a distributed most min value of each region function needs a systematic Global optimization to find a point in a region that is a candidate for number of population in each region and also a unique

representative for the region of other region representatives. At the first iteration of optimization, the random cluster id generates global points and the model assumes each representative lookouts of each region are optimal lookouts until other lookout within the region comes. Then selects the most min of all global representatives and check the validity of this lookout in local region. if the local region most min point does not match to the global point, the local most min lookout id becomes global id. This process continuous until all region representative lookouts becomes the same to all global and local points. If the iteration is greater than one the most min value will be within the given expected value payoff of the leader function.

2.4 Global optimization algorithm

1. Let N be the number of decision making steps and n_i is an i^{th} function of a level

2. each hierarchy is categorizes into two; namely odd and even number line points.

3. Each decision space of either odd or even will grouped into C number of clusters

3.1 Each cluster will have a data point list of a sub interval

4. Select random point of odd and even data point from each cluster C_i

5. Evaluate the data point payoff of each cluster and select the most either minimum or maximum of the list of payoff while considering the other level functions are on their lower boundary

6. Consider the most min/max value of all clusters as global optimum

7. Search for a point which is better than current global optimum point in a

data set interval that includes the current global optimal point.

8. Compare the new data set point of each cluster with the previous global points of each cluster

9. Search for a point better than the previous distribution point to locate global optimal

10. This process from 5 to 9 continues until the iterative convergence tolerance is satisfied for both odd and even distribution

11. The final outcome will be a list of global optimal point of each cluster in both odd and even distribution

12. Send this parameters for next level function response

2.5 Process of Approximate Stackglburg Nash equilibrium Model

The stackglburg nash equilibrium process starts by accepting parameters from the user and then processes computational operations for a given either data set or list of functions.

1. Maximum number of hierarchical lookouts over S or decision space (alpha)

2. Number of total levels (k)

3. Objective functions of each level (f)

4. Constraint function of each level (fc)

5. Lower boundary of each objective functions (lb)

6. Upper boundary of each objective functions (u)

7. Number of population in each region (maxiter)

8. Number of candidate spies in each region (b)

9. Tolerance of successive reaction (tol)

10. Convergence result between odd and even categories (rxntol)

11. Each objective function acceptable loss values (PlAcloss)

12. Execution direction (dir)

13. Decision range (range)

14. Number of levels

15. Number of lookouts

16. Cluster points or clusters (Fn.clu)

2.5.1 Leader Function Optimization

The leader function selects or sends the most minimum value of each region lookout variable value to the middle level respective follower by considering the followers are at lower boundary value of their own of the current iteration. This subroutine uses global optimization subroutine to get the most minimum value of each region. Hence, It sends the following parameters.

1. Cluster points of either odd or even categories

2. Random selection of cluster id for starting optimization process in condition of the first iteration. If the iteration is not first iteration, the system uses suggested cluster id instead of random cluster id.

3. Objective function of the leader

4. Initial optimal value that fixes the other level decision maker is at the lower boundary.

5. Start of global optimization loop iteration

6. Number of global representative lookout variables

7. Current level

8. Number of populations or candidate lookouts of the current category.

9. List of visited clusters

10. Number of visited clusters

11. Number of iteration or current iteration

12. Expected number of leader function payoff range

13. Execution direction

The global optimization sub-routines returns

1. List of optimal representative lookouts value of each region

2. List of visited clusters

2.5.2 Forward Direction Computation

The forward direction computation section of the model accepts the lookouts of the leader and optimize its reaction value by considering followers bellow the current level are at their lower boundary value of their own. In this section of computation,

the last level of the model decision maker gets or sends its the last and optimal response to the above level decision maker. When the number of decision makers at this middle level is greater than one (1) and have simultaneous decision variable, the computation becomes Nash equilibrium or approximate Nash equilibrium. So the reaction value of the level will be the sum or agreement form of each decision maker optimal choice with respect to the other decision maker choice. The forward direction operation subroutine evaluates the response of each decision maker for each leader lookouts. The followers use global optimization to find out the most minimum point of the follower or current decision maker. And then, evaluate those most minimum value of each region to get the most minimum point among the distribution which is obtained from global optimization sub-routine. This section of operation sub-routine passes the following parameters for the current level decision maker optimal response.

1. Cluster points of all level objective function
2. Representatives lookouts values of leader function
3. Objective function
4. Number of levels
5. Lower boundary of either odd or even categories
6. Number of population in each region
7. Number of representative lookouts on the decision range
8. Even or odd categorizes identification
9. Constraint functions
10. List of visited clusters

11. Number of visited clusters

12. Number of iteration or current iteration

13. Expected number of leader function payoff range

14. Execution direction

15. Sub-interval

16. Random or suggested cluster id

The forward execution sub-routine returns the follower reaction value. And it's visited cluster id at back execution which is used for next cluster id selection. After all decision makers perform their best response for the given leader function lookout, the back execution forms stackglburg Nash approximate solution. So that the leader function choose a lookouts value that maximizes or minimizes its payoff value.

Back execution section of the model starts from the last level of the follower reaction value and compute the approximate stackglburg-Nash equilibrium value of the middle level. And then compute the reaction value of the top level decision maker. In this computation, the reaction value of each level value must be in the feasible region of decision space. To check the feasibility of representative lookouts value that is returned from global optimization, the model uses constraint check subroutine. Some representative lookouts could not be feasible as it is obtained from global optimization. Therefore the application of mutation operator that checks candidate lookouts values one or two step back and forward of the representative lookout would form a chance to select most min lookouts after mutation. Therefore the back execution selects the most min optimal value which is in a feasible region. This process is applied to all representative lookouts of the above level decision maker.

2.5.3 Leader Decision

The equsoln sub-routine decides the leader function choice that maximizes or minimizes its payoff. For this purpose, it accepts the following parameters

1. The best response of each follower and the adjusted leader lookout value which forms equilibrium.

2. Leader function

Based on the above two inputs, the equ-solution sub-routine returns lookout value that maximizes its payoff. This process is applied for odd and even categories to make sure the selected lookout is the optimal solution.

The successive or the next iteration selected lookouts value tolerance to the first selected lookouts must be less or equal to the reaction tolerance. And the sum of this reaction tolerance of odd and even category must be less than or equal to tolerance that satisfies the tolerance condition and the model brakes the execution and return an approximate stackglburg nash equilibrium solution.

2.6 Clusters Optimization Computation Operations in either sequential or parallel

The main computational operations of the computer are fetching, load, execute, unload and operate. Computational optimization models are either boundary condition or initial condition problems. The thesis global optimization model is a boundary condition problem with the given either dataset or objective functions and constraint functions and boundary values. This global optimization model operational activity has four sections.

1. Initialization
2. Definition of data point interval and subinterval
3. Populating the data point to a number of cluster and regions

4. Approximate stackglburg nash equilibrium multi-level operation

1. Initialization

This section of the operation allocates variable space in RAM and assigns value. The model initial variables like number of levels, functions, constraints functions, boundary values, reaction tolerance etc is expressed. Based on this, the total number of operations that a computer do with a processor that supports parallel computation are

i. Number of functions
ii. Number of constraint functions
iii. Number of assignment operations for model parameters

For bi-level model			
		Sequential	Parallel
i	fn	2	0
ii	fc	2	0
iii	Asgn	17	0
Total		21	0

Table 2.1 Initialization; total number of operations on bi level example

2. Definition of data point interval and subinterval

In the boundary condition problems; data point definition is required before computation begins. The number of operations required to be performed to identify the data point gap is equal to the number of levels. But each level can be performed in parallel. Therefore; the number of operations becomes one as if there are equal numbers of threads as the levels on the computer. In each operation there are two assignment operations.

i. Number of levels
ii. Assignment operations

For bi-level model			
		Serial	Parallel
i	level	2	1
ii	Asgn	2	2
Total		4	2

Table 2.2 Definition of data point interval and subinterval for bi level example

3. Populating the data point to a number of cluster and regions

The number of operations required to define the data point of each region of each cluster can be performed in parallel. This kind of nested loop operation requires a decision to choose to which loop operation needs to be computed in parallel. On this bi-level model, region data definition for each category has more number of operations than the loop of level of functions. Therefore; the total number of operations of data point definition is

- ✓ $Number\ of\ level * Number\ of\ data\ point * number\ of\ region$

If there are equal number of threads as the number of data points in region; there will be number of level times number of operations. In each operation there are seven sub operations to be completed.

i. Number of levels
ii. Number of data point
iii. Computational operations

For bi-level model			
		Serial	Parallel
i	level	2	2
ii	data point	50	1
iii	opert	7	7
Total		100	10

Table 2.3 populating the data point to a number of cluster and regions for bi level example

4. Approximate stackglburg and nash equilibrium multi-level operation

The whole stackglburg and nash optimization model is inside the while loop. This while loop process operations iteratively. The maximum number of iteration is equal to the number of population or data point at each region assuming that

each data point has a full probability to be global point of its region.

The objective of this parallel computation model is to reduce the number of iterations or to determine the global point systematically without operating all candidate points one by one. Then the total number of operation is multiplied by number of iterations.

If the number of iteration is represented by iter, the total number operation is

$$operation^{total} = iter * (Stackglburge\ Nash\ number\ of\ operations)$$

The list of operations performed at this operation phase is:

4.1. Categorizing the data points of each level

The model performs operations on the data points categorically. In this computation there are two categories which are odd and even. This operation can be performed in parallel. The total number of operations for categorical computation is

$$operation^{total} = iter * (catgori * (Stackglburge\ Nash number\ of\ operations))$$

If there are two threads which work in parallel, the total number of operation become

$$operation^{total} = iter * (1 * (Stackglburge\ Nash number\ of\ operations))$$

Computation process of the model inside each category is performed iteratively. So the numbers of operations of the model are the sum of each operation inside the categories. The list of operations in the categories are

4.1.1. Random point generator or suggested cluster id definition/ identification (RPG)

4.1.2. Selecting initial condition (SIC)

4.1.3. Computing global optimization (CGO)

4.1.4. Computing forward execution (CFE)
4.1.5. Computing equilibrium solution (CES)
4.1.6. Tolerance calculation (TC)
4.1.7. Computing braking condition (CBC)

The RPG and SIC operations are performed iteratively and it is one to each operation. Then the total number of operation for RPG and SIC is two (2) operations. The CGO or computing global optimization has sub operation to be performed. Thus sub operations are

Initialization

This section performs assignment operation for initial condition, leader lookouts and loop iterations. There are three operations in total before moving to next section of computation.

Global point lookout computation

Each lookouts of the function is computed. But this can be performed in parallel. The total number of operations to be processed by global point's lookout computation is equal to one if there are equal numbers of threads that operate in parallel. With this global points searching, there are operations which are performed iteratively. The numbers of operations processed iteratively are three (3). And the total operations become four (4) operations.

Local optimization

Local optimization operation is related to the number of data points located in each region. But this operation can be performed in parallel if there are equal numbers of threads to the number of data points. So the total number of operation is one. In this section, there are nine operation need to be performed iteratively. The total operations become ten (10).

Recursive operation

Global optimization operation is the recursive operation. It processes the above operation iteratively. And it is expected to be completed to maximum of five to three operations. The total number of operation perform at this global optimization is ;

$$number\ of\ opertion = recusive * (local\ opt + global\ opt + initalization)$$

$$number\ of\ operation = (|5) * (10+4+3)$$

that is equal to either maximum eighty five (85) or minimum fifty one (51) operations.

Computing Forward process operation

The forward operation computation processes two operations iteratively. One is computing from head or start of operation to end level of operation. And then operate from the last level of operation to the first level for decision. For this; there is assignment of variables values to be passed as parameters. The total number of assignment operation is two. And then each lookout is performed in parallel. In each lookout there are eight plus number of lookout operations and global optimization (CGO) and forward fixation which have five numbers of operations. Function result calculation which is equal to the number of lookout is performed before back direction computation starts.

$$number\ operation^{forward} = 2 + lookout * (assign + CGO + frfix + b * 2)$$

In reverse operation that is from last level function to the first level function, the lookout is performed iteratively. Because nested loop operation both in parallel is not supported. Hence, the total number of operation during back direction operation is

$$number\ operation^{backward} = 2 + total\ level * (CGO + lookout * (constcheck + 6 + 5 * (2)))$$

Therefore, the total stacklburg nash model number of operation(SNO) is equal to

$$SNO = RPG + SIC + CGO + CFE + CES + TC + CBC$$

And the total number of computation operation become

$operation^{total} = iter * (1 * (SNO))$ for parallel operation and

$operation^{total} = iter * (2 * (SNO))$ for sequential operation

Chapter 3 Computer Model Simulation and Analysis

In this bi-level optimization problem, if the first function represent the rate change of the amount of water over the pool volume and the second function represent the amount of fish to be produced in the pool, then the optimal result of the bi-level gives the volume of water level in the pool and number of fish production.

100	101	104	109	116	125	136	149	164	181	200	221	244	269	296	325
81	82	85	90	97	106	117	130	145	162	181	202	225	250	277	306
64	65	68	73	80	89	100	113	128	145	164	185	208	233	260	289
49	50	53	58	65	74	85	98	113	130	149	170	193	218	245	274
36	37	40	45	52	61	72	85	100	117	136	157	180	205	232	261
25	26	29	34	41	50	61	74	89	106	125	146	169	194	221	250
16	17	20	25	32	41	52	65	80	97	116	137	160	185	212	241
9	10	13	18	25	34	45	58	73	90	109	130	153	178	205	234
4	5	8	13	20	29	40	53	68	85	104	125	148	173	200	229
1	2	5	10	17	26	37	50	65	82	101	122	145	170	197	226
0	1	4	9	16	25	36	49	64	81	100	121	144	169	196	225
1	2	5	10	17	26	37	50	65	82	101	122	145	170	197	226
4	5	8	13	20	29	40	53	68	85	104	125	148	173	200	229
9	10	13	18	25	34	45	58	73	90	109	130	153	178	205	234
16	17	20	25	32	41	52	65	80	97	116	137	160	185	212	241
25	26	29	34	41	50	61	74	89	106	125	146	169	194	221	250
36	37	40	45	52	61	72	85	100	117	136	157	180	205	232	261
49	50	53	58	65	74	85	98	113	130	149	170	193	218	245	274
64	65	68	73	80	89	100	113	128	145	164	185	208	233	260	289
81	82	85	90	97	106	117	130	145	162	181	202	225	250	277	306
100	101	104	109	116	125	136	149	164	181	200	221	244	269	296	325

Table 3.1 Data set of the first decision maker

The table 3.1 and 3.2 are data sets of results which are assumed to be the variation analysis result of the bi level problem. From this variation analysis table data, it is possible to develop functions of the bi-level which can be representative of this both table data information. The results out of this are objective functions and decision ranges.

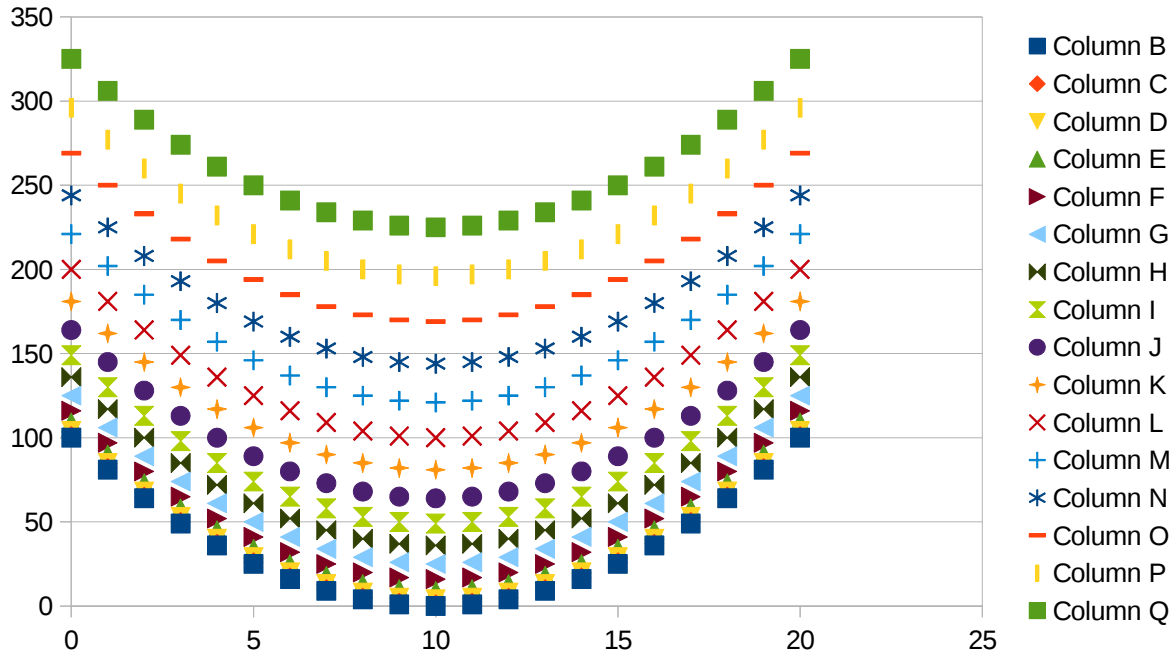

Figure 3.1 Variation result data of first decision maker

The following activities are the given data analysis to form behavioral study function

1. the minimum value of water level is the first row data of Table 3.1 when fish production is zero
2. identify the growing pattern of the first function dependent on the lower function variable variation of the water level
   $\{(0,100),(1,1),(2,4),(3,9),(4,16),(5,25),(6,36),(7,49),(8,64),(9,81)\}$
   $\{(10,100),(11,121),(12,144),(13,169),(14,196),(15,225)\}$
3. identify the serious of the development data pattern; and it has exponential nature that $100+(factor)^2$ where factor $X=\{1,2,3,\ldots,15\}$ for a constant value of $y=0$
4. perform step 2 and 3 for all dimensions; and the result is exponential pattern for all trials except the increment of dependent variable value y as the value varies, the constant function relation becomes $(10-y)^2$
5. Categorize the given data to the set of data which has same pattern; And the result is all have a common exponential data serious pattern

6. find common or general equation for all categorizes of given data; pattern $x^2+(y-10)^2$
7. organize each pattern equations; **function** $f=x^2+(y-10)^2$
8. test the equation to given data; that is valid for all set of data
9. perform from step 1 to 8 for each level function
10. finally; proceed to optimization problem solving

Based on the above steps, the first bi-level problem required data are;

- ✓ The function $f=x^2+(y-10)^2$
- ✓ Decision range from (0 to 15) for variable x
- ✓ Decision range from (0 to 20) for variable y

900	841	784	729	676	625	576	529	484	441	400	361	324	289	256	225
784	729	676	625	576	529	484	441	400	361	324	289	256	225	196	169
676	625	576	529	484	441	400	361	324	289	256	225	196	169	144	121
576	529	484	441	400	361	324	289	256	225	196	169	144	121	100	81
484	441	400	361	324	289	256	225	196	169	144	121	100	81	64	49
400	361	324	289	256	225	196	169	144	121	100	81	64	49	36	25
324	289	256	225	196	169	144	121	100	81	64	49	36	25	16	9
256	225	196	169	144	121	100	81	64	49	36	25	16	9	4	1
196	169	144	121	100	81	64	49	36	25	16	9	4	1	0	1
144	121	100	81	64	49	36	25	16	9	4	1	0	1	4	9
100	81	64	49	36	25	16	9	4	1	0	1	4	9	16	25
64	49	36	25	16	9	4	1	0	1	4	9	16	25	36	49
36	25	16	9	4	1	0	1	4	9	16	25	36	49	64	81
16	9	4	1	0	1	4	9	16	25	36	49	64	81	100	121
4	1	0	1	4	9	16	25	36	49	64	81	100	121	144	169
0	1	4	9	16	25	36	49	64	81	100	121	144	169	196	225
4	9	16	25	36	49	64	81	100	121	144	169	196	225	256	289
16	25	36	49	64	81	100	121	144	169	196	225	256	289	324	361
36	49	64	81	100	121	144	169	196	225	256	289	324	361	400	441
64	81	100	121	144	169	196	225	256	289	324	361	400	441	484	529
100	121	144	169	196	225	256	289	324	361	400	441	484	529	576	625

Table 3.2 Data set of the second decision maker

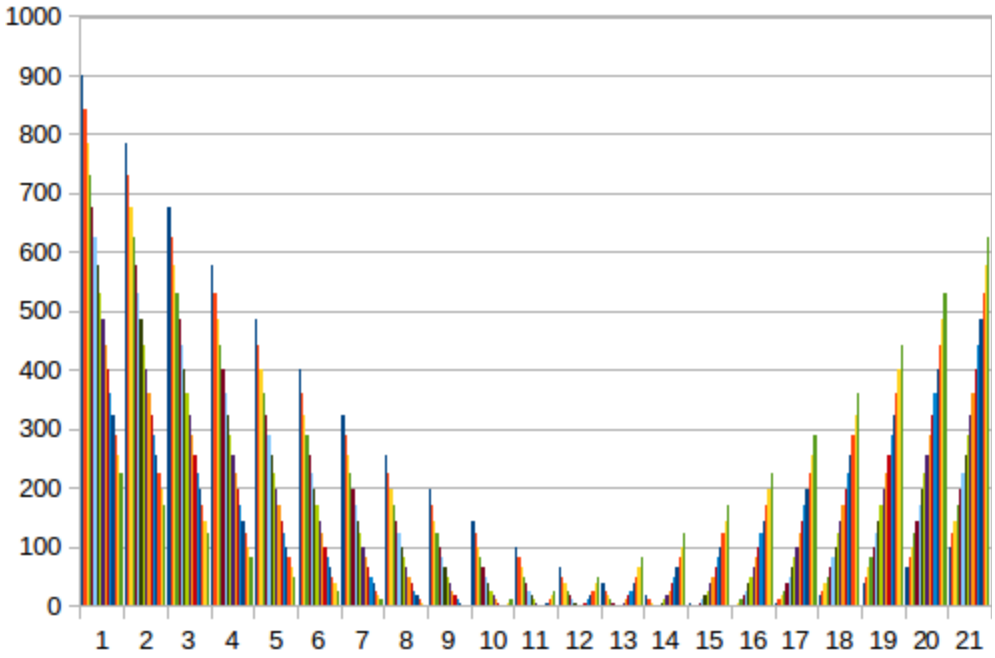

Figure 3.2 variation of result data set of second decision maker

The based on the above activity from 1 to 10 done for the first decision maker on table 3.2, the second level function becomes the following;

$$f_2(x,y)=(x+2y-30)^2$$

$$x+y\le 20$$

$$where\ 0\le y\le 20$$

In general the bi-level function optimization problem becomes

$$f_1(x,y)=x^2+(y-10)^2$$

$$subjected-x+y\le 0$$

$$where\ 0\le x\le 15$$

$$f_2(x,y)=(x+2y-30)^2$$

$$x+y\le 20$$

$$where\ 0\le y\le 20$$

The following example 3.1.1 depends on the following given table data information for computerized multi level multi follower optimization model.

No	Initialization	Symbol	value
1	Given lookouts	alpha	10,000
2	Max iteration	Maxiter	50
3	Number of levels	K	2
4	Odd/even categories	Od	2
5	lookouts of each level	$b=b_{11},b_{21}$	43,57
6	Reaction tolerance	Rxntol	0.005
7	Convergence between odd and even	Ee	0.01
8	Interval	Inval	0.1744,0.1754
9	Sub interval	Subinval	0.0035,0.0035
10	Recursive iteration	Loop	3

Table 3.3 Given table data information for example 1

Example 3.1.1 A given bi-level function elaboration; if the first function represents the amount of water volume needed to pour to second function that represents the size of pool which is used for mass production of fishes, then relationship between the volume of water and the volume of pool is the bi-level optimization problem.

- After a Matlab code process sequentially for about 2.56 seconds on dual core Pentium 4 processor, the result is as shown on Table 3.4.

For bi-level model result						
iteration		odd distribution		even distribution		
	level	root value	decision result	root value	decision result	tolerance
1	1	10	100	10	100	0.01
	2	9.9982	0	10	0	0.01
2	1	10	100	10	100	0.01
	2	9.9982	0	10	0	0.01

Table 3.4 Computational solution of Example 3.1

Solution Interpretation: the model performs bi-level minimization optimization. For ten (10.0000) units of water volume which costs 100 birr would result zero amount of sails in birr with 10 number of fishes in the pool. Therefore the bi-level model is the relation between water volume and number of fishes that are in cost function. The computational Code of this problem is attached to this document.

Note:- the above Interpretation is for the conceptual understanding of the model, its is not to refer to the actual event scenario.

3.2 Stackglburg Nash Multi level multi-follower optimization Model Analysis

So far, Competitive structured stackglburg and Nash problems has been challenging to solve due to the iterative nature of other suggested optimization multi level multi follower cooperative and non cooperative objective function computation techniques.

This model is a deterministic optimization model. it is a multi-follower multi-level optimization computational model that optimizes most kind of function within given decision range. This model works for both constrained and unconstrained optimization problem. It optimizes both homogeneous and non homogeneous set of functions. It is more efficient and less computational time taking plus more accurate

in optimization result. It is advantageous for usage of parallel computation to locate the optimal value of each level of decision maker. It solves for both linear and Non linear decision maker objective functions.

3.3 Summary

The optimization model optimizes a list of functions which are arranged in hierarchical structure. The whole multi level hierarchic is modeled into three levels namely top, middle and bottom level. The top and bottom level functions have one layer and the middle level may have more than one level.

Each level of the hierarchic has a number of lookouts. Each lookouts are compare to each other to be global as well as local representative of the all region. And the global optimization algorithm locates the representative lookout of each region in form of parallel computation. Each parallel computation optimization model locates the optimal value of set of functions by looking optimal pair value of each level that causes the function result to be either maximum or minimum.

The model uses two side perspective convergence to reach to expected range of tolerance between two consequent number that are labeled odd and even. This model starts from the given variational relation representation functions and compute the optimal variation as the given parameters of the function varies.

And finally, the computational model provides a list of decision alternative to top level decision maker for decision.

Chapter 4 Conclusion and Recommendations

This a systematic global optimization algorithm works for structured competitive functions that are in either cooperative or non cooperative agreement of each level. And it is a decision support model in design that each leader functions has a free will to choose the bast strategy alternatives. As the result of this, the computation logic works for all types of functions. It is a model that searches optimal result in parallel over different and grouped region in a given decision space.

since it create an environment for parallel computation over greed data set, This model is advisable to be used by research institutions for the development of other models which support different structural arrangement of decision makers.

In general, it is a multi-follower multi-level optimization computational model that optimizes any most widely known functions within given decision range. It is more efficient and less computational time taking plus more accurate in result.

Chapter 5 Reference

[1] R.Duncan Luce and Howard Raiffa. Games and Decisions Inroduction and Critical Survey. Dover Publications, 1985.

[2] BAODING LIU Department of Applied Mathematics Tsinghua University Bei- jing 100084 China. Stackelberg Nash Equilibrium for Multilevel Programming with Multiple Followers Using Genetic Algorithms. 1998 Elsevier Science Ltd, Received and accepted February 1998.

[3] Ashenafi Teklay Woldemariam. Systematic Evolutionary Algorithm for a general Multilevel Stackelberg Problems with bounded decision variables SEAMSP. Addis Ababa Univercity Mathematics Departement, July 1 2013.

[4] J. Montero J. Zhang G. Lu and Zeng Y. Model solution concept, and Kth-best algorithm for linear trilevel programming Information Sciences. 2010.

[5] Won Y.Yang Wenwu Cao Tae-Sang Chung John Morris Applied Numerical Methods Using MATLAB

Matlab Code

1. Mat lab Global Optimization for multi level bi level code

```
function leaderfollsysnew()
tic
f{1}=@(x) x(1)^2+(x(2)-10)^2;
fc{1}=@(x) -x(1)+x(2);
f{2}=@(x) (x(1)+ 2*x(2)-30)^2;
fc{2}=@(x) x(1)+x(2)-20;
alpha=10000; % Maximum number of hierarchical Spies
maxiter=50;  % number of candidate spies in each region
rxntol=0.005;% Tol of successive reaction from spies
eee=0.01;     % convergence result between odd and even catagories
l=[0 0];      % Define lower boundaries
u=[15 20];    % Define upper boundaries
d=u-l;
k=length(f);
a(1)=((alpha*d(1)^(k-1))/(sum(d)-d(1)))^(1/k);
den=a(1)/d(1);
a(2:k)=d(2:k)*den;
b=floor(a/2);% The total number of representative spies in each
categories
inv1=(d(:)/(2*b(:)))';
inv=inv1(2,:);
subinv=inv(1,:)/maxiter;
```

```
%cluster points
for cl=1:(length(f))
    evco=0;
    odco=0;
    page=2;
        for rg=1:(maxiter)

            if (rem(rg,2)~=0)
            evco=evco+1;
            Fn(cl).clu(evco,:,page)=linspace((rg-1)*inv(cl),d(cl)-
(rg-1)*subinv(cl),(b(cl)));
            else
            odco=odco+1;
            Fn(cl).clu(odco,:,(page-1))=linspace((rg-
1)*inv(cl),d(cl)-rg*subinv(cl),(b(cl)));
            % zero or first value of zero points
            end

        end
end
iter=1;
id=randperm(maxiter/2);
lb(1,:)=l;lb(2,:)=l;
%staglburg Equilibrium calculation
while(iter<=maxiter)
    for i=1:2
        if iter==1
```

```
        [x0]=frfix(1,k,lb(i,:));
        %fixing the values of followers
       [dist(i,:)]=GlobalOpt(Fn(1,1).clu(:,:,i),id,f,x0,0,b(1),1,
(maxiter/2));
        end
        %leader optimization
        [Rxnval(:,:,i)]=ForwardExe(Fn,dist(i,:),f,k,l,
(maxiter/2),b,i,fc);

[StagEqu(1,:,i),idx(i),Lpayoff(i)]=equsoln(Rxnval(:,:,i),f(1));
        Lv2(iter,i)=Lpayoff(i)-rxntol*(1+abs(Lpayoff(i)));
        Lv1(iter,i)=Lpayoff(i);
        if iter>1
        tol(1,i)=((Lv1(iter-1,i)- Lv1(iter,i))/(1+abs(Lv1(iter-
1,i))));
        end
        lb(i,:)=StagEqu(1,:,i);
        diff= Lpayoff(i)- Lv2(iter,i);
        range(i,:)=[(Lpayoff(i)+diff) abs(Lpayoff(i)- diff)];
         %value range definition
        Equsolution(iter,:,i)=StagEqu(1,:,i);
        %value range definition
    end
         if iter>=1 & iter <=maxiter
                if iter>=2
                      if sum(tol)<=0.01
                      Equsolution
                      iter=maxiter+1;
```

```
                    end
                    %braking condition
                end
            for odd=1:2
                fi=lb(odd,:);
                cotemp=0;
                for subint=1:(maxiter/2)
                    for spy=1:b(1)
                      fi(1,1)=Fn(1,1).clu(subint,spy,odd);
                      distemp=feval(f{1},fi);
                       if distemp <= max(range(odd,:))& distemp >=
min(range(odd,:))
                        cotemp=cotemp+1;

dist(odd,cotemp)=Fn(1,1).clu(subint,spy,odd);
                       end
                    end
                end
            end
            %looking for point or candidate spies that give a
objective value result within
            %the range of current iter value and next prediction
value
            %that satisfies tollerance
         iter=iter+1;
         end
end
toc
```

```
end
function [dis]=
GlobalOpt(Clus,Cid,Obfn,fi,Loop,spy,Lev,Maxiter,ObvalG,Vc,node)
Loop=Loop+1;
x=Clus(Cid,:);
node=x;
%global points based on Cluster id
 for i=1:spy
      fi(1,Lev)=x(1,i);
      ObvalG(Loop,i)=feval(Obfn{Lev},fi);
 end
%evaluate the Global points to the leader function
 if Loop~=1
 ObvalG(Loop,Vc)=inf;
 end
 %exclude the visited cluster
[vals,ids]=min(ObvalG(Loop,:));
%evaluate the most min value
for i=1:Maxiter
    fi(1,Lev)=Clus(i,ids);
    ObvalL(1,i)=feval(Obfn{Lev},fi);
end
%evaluate the local points to the leader function
[vals,idsl]=min(ObvalL);
ObvalG(Loop,ids)=vals;node(1,ids)=Clus(idsl,ids);
%find the most min of the local points of ids
[valG,idsG]=min(ObvalG);
```

```
%find the most min out of local and global points
 Vc(1,Loop)=ids;


%Record the visited Cluster
 if (Loop)~=(idsG) & Loop >1 & idsl==Cid,
     dis=node;
 else
     Cid=idsl;
     [dis]=
GlobalOpt(Clus,Cid,Obfn,fi,Loop,spy,Lev,Maxiter,ObvalG,Vc,node);
 end
end
function [Rxn]=ForwardExe(Clus,spi,Obfn,lev,lower,Maxiter,b,od,fc)
id=10;
 ex=1;
for i=1:length(spi)
    exist=1;
    x(1,:)=spi(i);
    for j=2:(lev)
    fi(j-1)=spi(i);
   fi(j:lev)=frfix(j,lev,lower);

[Fdist]=GlobalOpt(Clus(1,j).clu(:,:,od),id,Obfn,fi,0,b(j),j,Maxite
r);
   %Reaction calculation for follower
   for sp=1:b(j)
      fi(1,j)=Fdist(1,sp);
      ObvalG(1,sp)=feval(Obfn{j},fi);
```

```
   end
   % evaluate the function with value
   [val,ix]=min(ObvalG);
    x(1,j)=Fdist(1,ix);
    % evaluate most min value



    end
 %--------forward execution-----------------
      fi(:)=x;
    for jb=(lev-1):-1:1

[Fdist1]=GlobalOpt(Clus(1,jb).clu(:,:,od),id,Obfn,fi,0,b(jb),jb,Ma
xiter);


    for sp=1:length(Fdist1)
      fi(1,jb)=Fdist1(1,sp);
      ch=constcheck(jb,fc,fi,lev);
      if ch==1
      ObvalGB(1,sp)=feval(Obfn{jb},fi);
      else
       ObvalGB(1,sp)=inf;
      end
       %check constraint
       % evaluate the function with value
    end
```

```
    [val,ix]=min(ObvalGB);
    if val==inf
    jb=0;
    exist=0;
    else
    x(1,jb)=Fdist1(1,ix);
    end

    end
    if exist==1
    Rxn(ex,:)=x;
    ex=ex+1;
    end
 %----back operation-------------------------------------
end
end
function [fr]=frfix(lev,Nfn,lower)

          if (lev)<Nfn,%dir direction forward
             fr(1,(lev+1):Nfn)=lower((lev+1):Nfn); %fixed value
             if (lev-1)>=1
             fr(1,(1:(lev-1)))=lower((lev+1):Nfn);
             end
         else
         fr(1)=lower(lev);
         end
end
```

```
function [val]=constcheck(cl,fnc,fi,k)
fl=0;
     for j=cl:k
      fcx=feval(fnc{j},fi);
      if fcx<=0
      fl=fl+1;
      end
     end
    if fl==(k-cl)+1,
        val=1;
    else
        val=0;
    end
end
function [x,ix,val]=equsoln(xodd,fn)
   for j=1:length(xodd)
     fx(1,j)=feval(fn{1},xodd(j,:));
        end
   [va,ix]=max(fx);
   val(1)=va;
   x(1,:)=xodd(ix,:);
   for i=2:length(fn)
   val(i)=feval(fn{i},x(1,:));
   end
   % return the index for accessing range and cluster index
end
%Global Opt :- Global Optimization
```

```
%ForwardExe :- Forward Execuation

%frfix      :- Forward Fixation

%constcheck :- constraint Checking

%equsoln    :- Leader Decision
```