ISSN No. 0976-5697

Volume 2, No. 4, July-August 2011

International Journal of Advanced Research in Computer Science

CASE STUDY AND REPORT

Available Online at www.ijarcs.info

Non-Linear Great Deluge algorithm for Tanzanian High Schools Timetabling

Allen Rangia Mushi
Department of Mathematics
University of Dar es salaam
Box 35062, DSM, Tanzania
amushi66@yahoo.co.uk

Abstract: High school timetabling problem involves allocation of students, lessons, and teachers into timeslots while respecting constraints, both on students, teachers and other available resources. It is one of the Combinatorial Optimization Problems which are known to be NP-Hard and therefore no optimal algorithm is known for its solution. The problems differ from one institution to another depending on the educational system and administrative structures. In this paper, a Great Deluge Algorithm is developed based on an adaptation which employs a non-linear decay rate in the reduction of 'water level'. This is a case study in the application of the algorithm to Tanzanian high schools. The algorithm is tested on three high school systems in Tanzania. Since no such work has been previously done in Tanzania, the algorithm is compared with the manually generated timetables for the same schools. It has been shown that, the algorithm performs very well and can be used to greatly improve timetabling at Tanzanian high schools.

Keywords: High School Timetabling, Great Deluge, Combinatorial Optimization.

I. INTRODUCTION

High school timetabling problems consists of assigning teachers, lessons and students timeslots subject to some constraints on teachers, students and other important resources. It is one of the academic timetabling problems; others include university course timetabling and examinations timetabling which have significantly different features [1]. The constraints are normally divided into hard and soft, where hard constraints must be satisfied, while soft constraints have to be minimized as much as possible.

Most of the timetabling research has concentrated on university course and examination timetabling. High school timetabling problems have not been extensively studied despite of their importance in optimizing school resources [2]. Mathematical programming approaches have been attempted with some success [4]. However the problem is known to be NP-Hard [5, 6, 7], and therefore it is not possible to get optimal solutions for all instances within reasonable time.

Heuristic approaches are the most prominent in the literature. Zang, Liu, Halla and Leung [8] presented an algorithm for solving the high school timetabling problem using simulated annealing with a new neighborhood structure. The new neighborhood structure involves a successive set of swaps instead of a single swap of timeslots. The algorithm is applied to some benchmark test data with promising results.

Local Search heuristics have been presented by Schaef [9] where various types of moves were implemented with the use of adaptive relaxation on constraints. The work was tested on some large high school problems with good results. Schaef [10] implemented Tabu Search techniques as an improvement over the use of local search techniques. Same test data as in [9] were used which showed some improvement over local search.

Some work on Genetic Algorithms and its variants is also found in literature. Kimo, Nurmi, Jari Kyngas [2] presented a framework for optimization of highly constrained Finnish high school timetabling problem. They modified a previous work on Genetic algorithms [3] and tested their results on some real-life and artificial problems. Their high school model however considers many constraints which are not applicable to Tanzanian environment, including the flexibility of room use and the possibility of more than one teacher teaching the same class at the same time. Random non Ascent (RNA) search coupled with Genetic Algorithms and parallel processing [11, 15] have been applied on synthetic scenarios and real cases in Spanish system with positive results.

Graph-based methods using graph coloring on bipartite graphs have also been reported [12, 13]. Each node on the left of the bipartite graph represents a teacher and each node to the right represents a student group; there is one edge connecting left and right nodes for each timeslot. The task is to find a coloring of edges in such a way that no two edges adjacent to any vertex have the same color. The methods work very well in removing infeasibilities on hard constraints but do not help in dealing with vast number of soft constraints.

Other approaches includes evolutionary algorithms and their variants [14, 15, 16], cyclic transfers on neighborhood structure [17] and four phased approach [18] which combines several heuristics in phases. Pupeikiene and Mockus [19] presented an analysis of some optimization algorithms for high schools in commercial package and compared with their algorithm using perturbation methods on Monte Carlo, Simulated Annealing, and Bayesian heuristics. They unveiled many drawbacks of commercial packages especially on the fact that they cannot work for all school requirements. This motivates the need to develop and test algorithms for specific school environments.

This paper considers the use of Great Deluge Algorithm [20] for solving high school timetabling problems in Tanzanian environment. The algorithm has been applied to other types of timetabling problems but to our knowledge, no work has applied it to the high school timetabling problems. The algorithm is modified by changing the decay rate into a nonlinear function. We apply the resulting algorithm in three high school problems in Tanzania and compare results with manually generated solutions.

The rest of this paper is organized as follows; Section II defines the high school timetabling problem and presents a 0-1 integer programming formulation. Section III presents the

Great Deluge algorithm and discusses the adaptation to Tanzanian high school timetabling. Section IV analyses the performance of the algorithm, and section V provides a conclusion and suggests future research directions.

II. PROBLEM DEFINITION

Tanzanian high schools have been expanding rapidly due to the demand for higher education in the country. This has provided tremendous challenge in improving performance in the use of educational resources. The high school in Tanzania is a two years program (Form V and Form VI), based on specializations where students have to be selected and register for a combination of three subjects. For instance, a group of students may register for Physics, Chemistry and Mathematics (PCM), while another group may register for Physics, Chemistry and Biology (PCB). The combinations are predetermined by the education authority and students are assigned to them according to their previous performances. The following features apply;

a. Students from different combinations may share common subjects.
b. Not all schools have all combinations; schools offer combinations according to interest and availability of resources.
c. Each subject has a fixed number of lessons to be offered to a group of students per week and these may differ between subjects.
d. A high school teacher may teach at most two subjects and teachers may share lessons of the same subject where they may be teaching different student combinations or years of study.
e. Each combination of a particular year uses the same pre-determined room throughout the academic year.
f. Students have to take common courses which cuts across the year of study.
g. There are compulsory courses depending on the student combination. For instance, a student doing science or business related combinations which do not include mathematics have to take a Basic Applied Mathematics (BAM) course.
h. There are periods which need to be set aside every week for religious studies where clergies from different religions are normally using the time to teach religion in schools.
i. Students of the same combinations may be allocated into streams depending on their number and the available facilities. Thus PCMVA and PCMVB could stand for PCM combination for form V students in streams A and B respectively.

Making use of the features above, we need to introduce the following definitions;

Class – a group of students taking the same combination in the same year of study and in the same stream and sharing the same room.

Subject group – a subject taught to a group of students who belong to the same class. For example, subject group MTPCMVA stands for mathematics subject as offered to PCM in form V of stream A.

Lessons – a specific set of instructions associated with a particular subject group. A subject group is made up of one or more lessons which have to be spread across the timetabling week.

Period – a timeslot that can be assigned to a lesson.

The set of *hard constraints* is summarized as follows;

i. A teacher cannot be assigned to more than one lesson at the same time
ii. A class cannot be assigned more than one lesson at the same time
iii. Lessons of the same subject group cannot be assigned to the same timeslot
iv. The religion timeslots must be respected and must be common to all students
v. A timetable must be complete i.e. all lessons must be assigned to timeslots

Soft constraints considered in this implementation are as follows;

a. Lessons of the same subject must be spread as much as possible throughout the week.
b. Minimize the use of early morning timeslots as some teachers needs time to commute to work due to traffic jams; minimize the use of late afternoon hours for similar reasons.
c. Some teachers prefers to have free slots at some times of the week due to other commitments
d. Some lessons have preferences for free periods during the week e.g. Biology classes needs some time to collect samples in the field for dissection

a. Mathematical Programming Formulation:

Suppose there are *n* lessons, *y* subjects, *z* classes and *m* timeslots;

Let $H = \{l_1, l_2, \ldots l_n\}$: set of all lessons

$K = \{k_1, k_2, \ldots k_m\}$: set of all timeslots (periods)

$J = \{j_1, j_2 \ldots j_y\}$: set of all subjects groups

$C = \{c_1, c_2, \ldots c_z\}$: set of all classes

L_j = Number of lessons of subject *j*

T_j = A teacher who teaches subject group *j*

G_j = *A* class of a subject group *j*

S_M = A set of early morning timeslots

S_A = A set of late afternoon timeslots

S_R = A set of religion timeslots

S_P = A set of slots which have restrictions due to other preferences

$$x_{ijk} = \begin{cases} 1 & \text{if lesson } i \text{ of subject } j \text{ is assigned timeslot } k \\ 0 & \text{Otherwise} \end{cases}$$

b. Hard constraints:

i. A teacher of any pair of subjects (*p*, *q)* cannot be assigned lessons of these subjects in the same timeslot
ii. $x_{ipk} + x_{jqk} \leq 1$ for all $(i,j) \in H$, $k \in K$, $(p,q) \in J \ni T_p = T_q$.
iii. A class *c*, can be assigned at most one lesson at the same timeslot *k*.
iv. $\sum_{i \in H} \sum_{j \in J \ni G_j = c} x_{ijk} \leq 1$ for all classes $c \in C$, $k \in K$.
v. Lessons of the same subject group cannot be slotted into the same timeslot
vi. $x_{ujk} + x_{rjk} \leq 1$ for all $j \in J$, $k \in K$, $(u,r) \in H \ni u \neq r$
vii. The religion timeslots must be respected. This is simply done by excluding the religion timeslots in the whole timetabling process. Thus we re-define the new

set of timeslots available for timetabling as $K' = K \setminus S_R$.

viii. All lessons of a subject must be slotted in the timetable (completeness)

$$\sum_{i \in H} \sum_{k \in K} x_{ijk} = L_j \quad \text{for all subjects } j \in J$$

c. Soft constraints:

Soft constraints are used in the construction of the objective function as follows;

i. Spread all lessons of subjects as far as possible by maximizing the distance between lessons of the same subject. This is achieved by minimizing the inverse of the distances i.e. minimize

$$\sum_{(i_1,i_2)\in H} \sum_{(k_1,k_2)\in K' \ni k_1 \neq k_2} \sum_{j \in J \ni x_{i_1 j k_1} = x_{i_2 j k_2} = 1} \frac{1}{(k_1 - k_2)^2}$$

ii. Minimize the use of late morning, late afternoon hours and other restrictions. This is achieved by counting the number of times where these odd hours have been used i.e. $\sum_{k\in S_M} x_{ijk}$, $\sum_{k\in S_A} x_{ijk}$ and $\sum_{k\in S_P} x_{ijk}$

In Mathematical Programming formulation, objective function is modeled as a linear combination of the soft constraints, where weights are supplied to each objective component depending on the priority. Given weights λ_1, λ_2, λ_3 we have the following model;

Minimize

$$f(x) =$$

$$\lambda_1 \sum_{(i_1,i_2)\in H} \sum_{(k_1,k_2)\in K' \ni k_1 \neq k_2} \sum_{j \in J \ni x_{i_1 j k_1} = x_{i_2 j k_2} = 1} \frac{1}{(k_1 - k_2)^2} + \lambda_2 \sum_{k\in S_M} x_{ijk} + \lambda_3 \sum_{k\in S_A} x_{ijk} + \lambda_4 \sum_{k\in S_P} x_{ijk}$$

Subject to:

$x_{ipk} + x_{jqk} \le 1$ for all $(i,j)\in H$, $k\in K'$, $(p,q)\in J \ni T_p = T_q$

$\sum_{i\in H} \sum_{j\in J \ni G_j = c} x_{ijk} \le 1$ for all classes $c\in C$, $k\in K'$.

$x_{ujk} + x_{rjk} \le 1$ for all $j\in J$, $k\in K'$, $(u,r)\in H \ni u \neq r$.

$\sum_{i\in H} \sum_{k\in K} x_{ijk} = L_j$ for all subjects $j\in J$

$x \in \{0,1\}$, and $K' = K \setminus S_R$

III. SOLUTION APPROACH

Since the problem is NP-Hard, mathematical programming approach for exact solution is not expected to give results within reasonable time. The approach mostly applied is to use heuristic algorithms which have been shown to provide good results within reasonable time, although there is no guarantee that the solutions are optimal. The proposed solution approach is based on Great Deluge (GD) heuristic algorithm. The basic great deluge algorithm is designed for a maximization problem. The name is inspired by an analogy of a great deluge where a person climbing a hill will try to move in any direction that does not get the feet wet in the hope of finding a way up as the water level rises. Therefore, the solution is found by searching randomly on the solution space. A 'water level' L is designed in such a way that a solution can only be searched above the water level [20, 21]. Solution S is accepted only if $S>L$, and as time goes on the level is raised up to a point where the solution is forced to the peak and stops. The approach is easily mapped to a minimization problem by starting the 'water leve'l at the top and accepts solutions only when they are below the 'water level'. In this case the water level is lowered to a value of zero (minimum) and stops.

The basic Great Deluge algorithm is as shown in Fig.1. Initial solution is generated and 'water level' is set to the objective function value of this initial solution. A solution S_0 from neighborhood of S is accepted only if either $f(S_0) < f(S)$ or $f(S_0)<L$. This means that a worse new solution can be accepted as long as it is below the water level L. Decay rate (ΔL) is the only input parameter in this algorithm which determines the speed of level reduction (decay).

```
Great Deluge Algorithm {
Specify initial solution S_o;
Initial level L = f(S_o), where f is the objective function;
Input decay rate ΔL;
while further improvement is possible {
    Define Neighbourhood N(S_o);
    Randomly select a candidate solution S∈N(S_o))
    Calculate f(S);
    if f(S) ≤ f(S_o){
        Accept S (S_o = S);
    else if f(S) ≤ L{
        Accept S (S_o = S);
    Lower the level ( L = L - ΔL)
    }
    return S_o as the best solution;
}
```

Figure 1. Basic Great Deluge Algorithm

The basic Great Deluge has been applied to several problems with success including the works by Burke, Bykov, Newall and Petrovic [20]. However, this linear function is restrictive on the search space and may skip better solutions on the way to lowest level. This prompted researchers to consider the use of non-linear decay functions so as to further relax restrictions on the search space. Landa-Silva and Joe orbit [22] presents a new nonlinear decay rate function as applied to course timetabling problem in Universities. The same authors presented a follow up work which combines evolutionary algorithm and non-linear Great Deluge for course timetabling problem [23]. Computational studies on nonlinear Great Deluge are also reported by Orbit and Landa-Silva which shows the strength of the algorithm by tackling complex course timetabling problems [24]. The proposed nonlinear function is given as follows;

$$\boxed{L = L \times (e^{-\delta(random(Min,Max))} + \beta}.$$

Parameters δ, *Min* and *Max* controls the speed of decay, while β controls the shape of the decay function. The higher the values of *Min* and *Max*, the faster the 'water level' goes down, and consequently, the search quickly achieves improvement, but at the expense of getting stuck early in the local optima. Parameter β represents the minimum expected penalty corresponding to the best solution, and in our case this is β=0. This approach is introducing extra parameters which may require tuning for better results. However, the range of parameters proposed by Landa-Silva and Orbit [23] seem to work well in our high school model. The best parameters found in our model computations are; $\delta=5\times10^{-8}$, *Min*=100,000, *Max*=300,000 and β=0.

a. Adapting Non-Linear Great Deluge (NLGD) to High School Timetabling Problem:

Applying NLGD requires defining a solution configuration, identifying the neighborhood of the current solution and setting the parameters. For this problem, a solution is represented by a three-dimensional 0-1 matrix X of dimensions $n \times |L| \times m$, where each $x_{ijk} \in X$ has a value of 1 if lesson i of subject group j is slotted in timeslot k and 0 otherwise. The number of periods per day differs from one school to another; a typical school has ten 40-minute periods per day. These are numbered consecutively from 1 on Monday to 50 on Friday as shown in Table I.

Table I. Numbering of timeslots from Monday to Friday

	Timeslot numbers									
Mon	1	2	3	4	5	6	7	8	9	10
Tue	11	12	13	14	15	16	17	18	19	20
Wed	21	22	23	24	25	26	27	28	29	30
Thu	31	32	33	34	35	36	37	38	39	40
Fri	41	42	43	44	45	46	47	48	49	50

Teachers are pre-assigned to subject groups and therefore identified through subject-group number. An array T is defined such that $t_j \in T$ is a teacher of the subject group j.

A **neighbor** of the current solution S is obtained by swapping timeslots of S. A pair of subject-group lesson assignments is identified randomly and the corresponding timeslots are swapped to obtain a new neighbor solution. This arrangement does not guarantee feasibility of the resulting neighbor. These are taken care of by penalties in the objective function.

Objective function is a linear combination of all constraints, both hard and soft; more weights are assigned to the hard constraints so as to discourage infeasibilities.

Thus, objective function is of the form

$$f(x) = \lambda_1 f_1(x) + \lambda_2 f_2(x) + \lambda_3 f_3(x) + \lambda_4 f_4(x)$$
$$+ \lambda_5 f_5(x) + \lambda_6 f_6(x) + \lambda_7 f_7(x) + \lambda_8 f_8(x)$$

Where λ_i= weight given to constraint i.

The functions f_1 to f_8 are defined as follows;

a. $f_1(x)$ spread lessons of a subject group throughout the week i.e.

$$f_1(x) =$$
$$\lambda_1 \sum_{(i_1,i_2)\in H} \; \sum_{(k_1,k_2)\in K' \ni k_1 \neq k_2} \; \sum_{j \in J \ni x_{i_1 j k_1} = x_{i_2 j k_2} = 1} \frac{1}{(k_1 - k_2)^2}.$$

b. $f_2(x)$ minimizes the use of early morning hours; specifically $f_2(x) = \lambda_2 \sum_{k \in S_M} x_{ijk}$.

c. $f_3(x)$ minimizes the use of late afternoon hours; that is to say, $f_3(x) = \lambda_3 \sum_{k \in S_A} x_{ijk}$.

d. $f_4(x)$ minimizes the use of other non-preference timeslots,

$$f_4(x) = \lambda_3 \sum_{k \in S_P} x_{ijk}$$

- $f_5(x)$ sums up the cases where assigned lessons of a subject are incomplete i.e.

$$f_5(x) = \lambda_4 \left| \sum_{j \in J} \left(\sum_{i \in H} \sum_{k \in K'} x_{ijk} - L_j \right) \right|.$$

- $f_6(x)$ counts the number of times w a class has been assigned more than one lesson in the same timeslot i.e.

$f_6(x) = \lambda_6 w$,

$$\text{Where } w = \begin{cases} 1 \text{ if } \sum_{k \in K'} \sum_{i \in H} \sum_{j \in J \ni G_j = c} x_{ijk} > 1 \\ 0 \quad \text{Otherwise} \end{cases}$$

- $f_7(x)$ counts the number of lesson collisions u, explicitly;

$$f_7(x) = \lambda_7 u.$$

$$\text{Where } u = \begin{cases} 1 \text{ if } \sum_{j \in J, k \in K'} \; \sum_{(u,r) \ni u \neq r} (x_{ujk} + x_{rjk} > 1) \\ 0 \quad \text{Otherwise} \end{cases}$$

- $f_8(x)$ counts the number of teacher collisions v, that is $f_8(x) = \lambda_8 v$.

$$\text{Where } v = \begin{cases} 1 \text{ if } \sum_{(i,j) \in H, k \in K'} \; \sum_{(p,q) \ni T_q = T_q} (x_{ipk} + x_{jqk} > 1) \\ 0 \quad \text{Otherwise} \end{cases}$$

A solution is feasible only if the sum of objective values of the hard constraints is zero.

Initial solution used in this case study is a simple assignment of lessons to timeslots. Going through each subject-group, all lessons of the subject-group are assigned to the timeslots from 1 to 50. Once at the end, the next set of lessons are assigned from left to right (from 1 to 50) in the same fashion until all lessons have been allocated. This approach is simple and it minimizes the possibilities of lesson collisions by allocating all lessons of a subject-group into different timeslots as much as possible. However, the resulting solution is most likely infeasible and does not spread lessons of a subject-group as far as possible. The simplicity implies that an initial solution is obtained rather quickly and infeasibilities can be pruned down by penalties in the objective function. The details of the NLGD are shown in Fig. 2.

```
Non-Linear Decay Rate Great Deluge Algorithm {
Specify initial parameters (δ, Min, Max, β);
Specify initial solution So;
Initial level L = f(So), where f is the objective function;
while L > 0 {
        Define Neighbourhood N(So);
        Randomly select a solution S∈N(So))
        Calculate f(S);
        if f(S) ≤ f(So){
                Accept S (So = S);
        else if f(S) ≤ L{
                Accept S (So = S);
        else Reject solution (S);
Lower the level L = L × (e^(−δ(random(Min,Max))) + β
        }
return So as the best solution;
}
```

Figure 2. Non-Linear Great Deluge Algorithm

IV. COMPUTATIONAL RESULTS

The algorithm was tested on three high school timetables in Tanzania whose timetables are currently prepared manually. It was written in C++ and tested on a 3GHz processor on a Windows platform. Table II shows properties of the tested problems.

Table II. : Properties of the tested problems

Problem	Subject groups	Lessons	Timeslots	Teachers
1(Azania)	62	462	45	30
2(Jangwani)	46	332	40	26
3(Tambaza)	118	919	50	50

Through experimentation, the following weights were determined and used in the objective function (Table III);

Table III. Weights in the objective function

Weight	Value	Description
λ_1	5	Lesson spread
λ_2	3	Early morning hours
λ_3	3	Late evening hours
λ_4	3	Non-preference timeslots
λ_5	10	Lesson completeness
λ_6	10	Class collision
λ_7	10	Lesson collision
λ_8	20	Teacher collision

The NLGD algorithm was tested on both problems with varying set of parameters; Table IV shows the results when tested on problem 1. Clearly, case 5 performed better than all other cases. Cases 1, 4 and 6 produced solution values which are close to number 5, however, the quality of solution in case 5 is still significantly better than the rest. Furthermore, it only took 239 seconds (less than 4 minutes) to get the solution which is tolerable. The same trend was observed for different problems, indicating that the best parameters for NLGD in the case study are; *Min* = 100,000, *Max* = 300,000 and $\delta = 5\times10^{-8}$.

Table IV. Performance by parameter selection (Problem 1-Azania)

Case	*Min*	*Max*	δ	Solution	Time
1	10000	20000	5×10^{-7}	309.005	298
2	10000	20000	5×10^{-8}	312.003	1,733
3	10000	20000	5×10^{-9}	312.003	16,008
4	100000	300000	5×10^{-7}	309.008	54
5	100000	30000	5×10^{-8}	306.002	239
6	100000	30000	5×10^{-9}	309.004	1015

Table V presents a comparison of performances between the basic Great Deluge and NLGD using the best tuned parameters for each algorithm. The best N_{iters} for basic Great Deluge was found at N_{iters}=4000.

Table V. : Performances of NLGD compared to basic GD

Problem	Property	Basic GD	NLGD
1	Initial cost	630.67	630.67
	Final Cost	306.003	306.002
	Lesson Collision	0	0
	Teacher Collision	0	0
	Completeness violation	0	0
	Lesson spread	0.00313	0.00248
	Class Collision	0	0
	Morning violation	153	153
	Evening violation	153	153
	Time (Seconds)	193	169
2	Initial cost	500.67	500.67
	Final Cost	252.004	249.005
	Lesson Collision	0	0
	Teacher Collision	0	0
	Completeness	0	0
	Lesson spread	0.00365	0.00489
	Class Collisions	0	0
	Morning violation	126	126
	Evening violation	126	123
	Time (Seconds)	126	110
3	Initial cost	2190.67	2190.67
	Final Cost	632.005	549.006
	Lesson Collision	0	0
	Teacher Collision	80	0
	Completeness violation	0	0
	Lesson spread	0.00459	0.00571
	Class Collision	0	0
	Morning violation	276	276
	Evening violation	276	273
	Time (Seconds)	573	526

NLGD performed better than basic GD in both cases. Both algorithms managed to get feasible solutions, but the best solution found is better in NLGD than basic GD for all cases; NLGD managed to prune further on soft constraints and took shorter time. It is worth noting however that both methods managed to solve the problems within a few minutes (less than nine minutes), which is tolerable for timetabling problems.

An observation of the performance of the algorithm through iterations by level, decay rate and cost reduction is shown in Figure 5 for problem 3. Initially, at higher levels, we observe fast reduction of cost which is closely related to the decay rate. At lower levels however, as decay follows a smooth curve, the cost function shows some long periods of stationary moves without improvement, followed by sharp drops to another level. This is an indication that stopping criteria must be very carefully designed to ensure exhaustive search. Otherwise the algorithm might stop prematurely after some long periods of stationary moves, while further improvement could be realized in the longer searching periods.

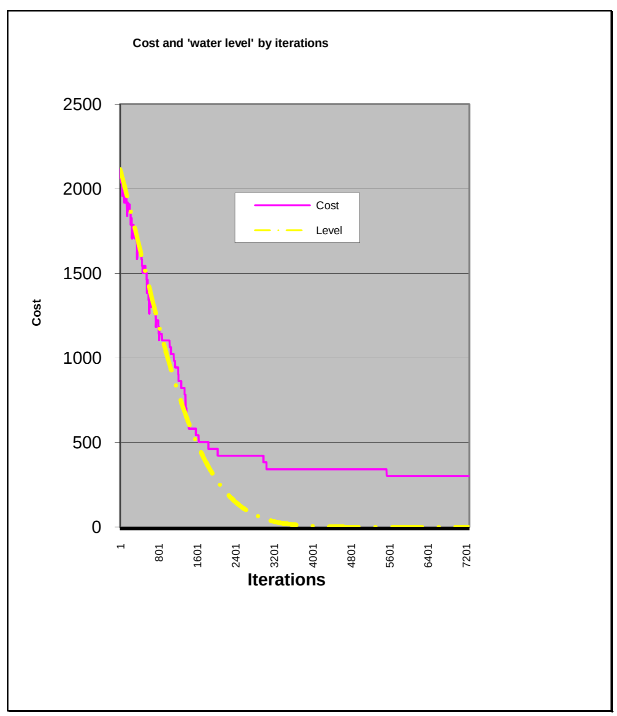

Figure 3. Performance improvement by iterations

Furthermore, it was worth comparing the performance of the algorithm to the manual system that is currently in place. The results are presented in Table VI for the three high school timetabling problems. Obviously, manual timetable is also feasible as shown in the table; otherwise it would not have been practical. However, the manual system provides tremendous disadvantages by failing to satisfy a lot of soft constraints. Both soft constraints have been addressed better by the automatic system despite of the fact that it took many days to prepare manual timetables. Clearly, NLGD provides tremendous advantages over manual system.

Table VI. : Manual versus NLGD Performances

Problem		**Manual**	**NLGD**
1	Solution cost	339.108	306.002
	Lesson Collision	0	0
	Teacher Collision	0	0
	Completeness violation	0	0
	Lesson spread	0.1083	0.00248
	Class Collision	0	0
	Morning violation	171	153
	Evening violation	168	153
2	Solution cost	311.014	249.005
	Lesson Collision	0	0
	Teacher Collision	0	0
	Completeness violation	0	0
	Lesson spread	0.01445	0.00489
	Class Collision	0	0
	Morning violation	154	126
	Evening violation	157	123
3	Solution cost	611.571	549.006
	Lesson Collision	0	0
	Teacher Collision	0	0
	Completeness violation	0	0
	Lesson spread	0.57111	0.00571
	Class Collision	0	0
	Morning violation	372	276
	Evening violation	239	273

V. CONCLUSION AND FURTHER RESEARCH

This paper presented a non-linear Great Deluge algorithm which incorporated an exponential decay function. This is a modification from the basic Great Deluge algorithm. The algorithm was adapted to the high school timetabling problems in Tanzanian schools and tested on three high schools. Moreover, solutions were compared to manually generated ones. It is shown that, the approach works well in the case study and performed better than manual system. The NLGD has also been shown to outperform the basic GD algorithm. This indicates that the non-linear function relates well to the 'water level' and cost reduction functions.

Since this is the first paper in the Tanzanian high schools to the best of our knowledge, it is clearly a good starting point for further research in the direction. It would be interesting to investigate other heuristic algorithms which have been well researched in similar problems. More data collection from varying schools and testing on the algorithm could bring more light into the properties and challenges of the problem. It is worth investigating further decay function which may come closer to the reduction function than the current situation.

It is concluded that, given careful tuning of parameters, the Non-Linear Great Deluge algorithm is a viable approach to high school timetabling problems in the Tanzanian environment.

VI. REFERENCES

[1] A. Schaerf, "A Survey of Automated Timetabling", Artificial Intelligence Review, Vol. 13, pp. 87-127, 1999.

[2] Kimo Nurmi, Jari Kyngas, "A Framework for school timetabling problem", Proceedings of the 5th Multidisciplinary International Scheduling Conference: Theory and Applications (MISTA), 28th – 31st August, 2007, Paris.

[3] K. Nurmi, "Genetic Algorithms for Timetabling and Traveling Salesman Problems", Ph.D. dissertation, 1998, University of Turku, Finland.

[4] T. Birbas, S. Daskalaki, E. Housos, "Timetabling for Greek High Schools", Journal of the Operational Research Society, 1997, Vol. 48, pp 1191-1200.

[5] D. de Werra, "An Introduction to Timetabling", European Journal of Operational Research, 1985, Vol. 19(2), pp. 151-162.

[6] A. I. S. Even, A. Shamir, "On the Complexity of timetable and multicomodity flow problems", SIAM Journal on Computing, Vol. 5, pp. 691-703, 1976.

[7] R. Willemen, "School timetabling construction: Algorithms and Complexity", PhD thesis, Technische Universiteit Eindhoven, The Netherlands, 2002.

[8] D. Zang, Y. Liu, R. Halla, S. Leung, "A Simulated Annealing with a new neighborhood structure based algorithm for high school timetabling problems", European Journal of Operational Research, 2010, Vo. 203, pp. 550-558.

[9] Andrea Schaerf, "Local Search Techniques for Large High-School Timetabling Problems", Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on Systems, Man and Cybernetics, 1999, Vol. 29(4), pp. 368 – 377.

[10] Andrea Schaeuf, "Tabu Search Techniques for Large High School Timetabling Problems", In Proceedings of the thirteenth national conference on Artificial intelligence – Vol. 1, (AAAI'96), 1996, Vol. 1. AAAI Press, pp. 363-368.

[11] Ana Cerdeira-Pena1 Luisa Carpente2 Antonio Fari˜na1 Diego Seco, "New approaches for the school timetabling problem", MICAI '08, Proceedings of the 2008 Seventh Mexican International Conference on Artificial Intelligence, IEEE Computer Society, doi>10.1109/MICAI.2008.19.

[12] David J. Abraham and Jeffrey H. Kingston**, "**Generalizing Bipartite Edge Coloring to Solve Real Instances of the Timetabling Problem", KanGAL Report No. 2006006. http://www.iitk.ac.in/kangal/papers/k2006006.pdf

[13] Edmund k Burke, Barry McCollum, Amnon Meisels, Sanja Petrovic, Rong Qu, "A Graph-Based Hyper-Heuristic for Educational Timetabling Problems", European Journal of Operational Research, Vol. 176(1), January 2007, pp. 177-192.

[14] Carlos Lara, Juan J. Flores and F´elix Calder´on, "Solving a School Timetabling Problem Using a Bee Algorithm", Lecture Notes in Computer Science, 2008, Vol. 5317/2008, pp. 664-674, DOI: 10.1007/978-3-540-88636-5_63.

[15] Hossam Faris, Alaa Sheta, "A Parallel Genetic Algorithm for Solving Timetable Problem", ICGST-AIML Journal, 2008, Vol. 8(II), pp. 43-52.

[16] Geraldo Ribeiro Filho and Luiz Antonio Nogueira Lorena, "A Constructive Evolutionary Approach to School Timetabling", In Proceedings of the EvoWorkshops on Applications of Evolutionary Computing, Egbert J. W. Boers, Jens Gottlieb, Pier Luca Lanzi, Robert E. Smith, Stefano Cagnoni, Emma Hart, G. R. Raidl, and H. Tijink (Eds.), 2001, Springer-Verlag, London, UK, pp. 130-139.

[17] Post, G., Ahmadi, S. and Geertsema, F. (2010) Cyclic transfers in school timetabling. OR Spectrum, Article in Press, online DOI: 10.1007/s00291-010-0227-y, 2010.

[18] de Haan, P. and Landman, R. and Post, G.F. and Ruizenaar, H.W.A., "A Four-phase Approach to a Timetabling Problem in Secondary Schools". In: Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling, 30 Aug - 01 Sept 2006, Brno, The Czech Republic.

[19] Pupeikiene P., Mockus J., High School Timetable Problem Solving and Comparison with Automatic Optimized Timetable, 2010, 16th International Conference on Information and Software Technologies IT2010, pp. 165 – 172.

[20] Dueck G. (1993): New Optimization Heuristics: The Great-Deluge Algorithm and the Record-to-record Travel, Journal of Computational Physics, Vol. 104, pp. 86-92.

[21] Burke E., Bykov, Y., Newall J., Petrovic S. (2003): A Time-Predefined Approach to Course Timetabling, Yugostlav Journal of Operational Research (YUDOR).

[22] Dario Landa-Silva, Joe H. Orbit, "Great Deluge with Non-linear Decay Rate for Solving Course Timetabling Problems", 2008, 4th International IEEE Conference "Intelligent Systems", pp. 8-18.

[23] Dario Landa-Silva, Joe H. Orbit, "Evolutionary Non-linear Great Deluge for University Course Timetabling", Hybrid Artificial Intelligent Systems, E. Corchado, X. Wu, E. Oja, A. Herrero, B. Baruque (eds.), Lecture Notes in Computer Science, Vol. 5572, pp. 269-276, Springer, 2009. DOI: 10.1007/978-3-642-02319-4

[24] Joe H. Orbit, Dario Landa-Silva, "Computational Study of Non-linear Great Deluge for University Course Timetabling", V. Sgurev, M. Hadjiski (eds). Intelligent Systems - From Theory to Practice. Post Conference IEEE-IS 2008 Volume, Series in Computational Intelligence, Springer Verlag, 2009.

