Digital Innovations
& Contemporary Research in SCIENCE, ENGINEERING & TECHNOLOGY
Vol. 6. No. 1, March, 2018

Article Citation Format
A.A. Ojugo & D. Allenotor (2018): Modeling the University Examination Schedule: A Comparative Study
Journal of Digital Innovations & Contemp Res. In Sc., Eng & Tech. Vol. 6, No. 1. Pp 193-210

Article Progress Time Stamps

Article Type: Research Article
Manuscript Received: 12th Dec, 2017
Review Type: Blind
Final Acceptance:: 29th March, 2018
DOI Prefix: 10.22624

Modeling the University Examination Schedule: A Comparative Study

[1]A.A. Ojugo and **[2]D. Allenotor**
[1,2]Department of Mathematics/Computer Science
Federal University of Petroleum Resources
Effurun, Delta State, Nigeria
ojugo.arnold@fupre.edu.ng, allenotor.david@fupre.edu.ng

ABSTRACT

Examination schedule is a hard NP-complete task with many challenges in the management/training of manpower. It persists as a herculean domain as it is rippled with chaotic and dynamic time-space-instructor-students_enrollment conflicts as input parameters to be satisfied. TTP studies aim to efficiently resolve conflict among the interacting constraints, and thus, yield a feasible, complete schedule. Examination TTP is more tedious in that same number of constraints must be resolved with shorter time continuum as its optimal outcome. Our study seeks computational intelligence harnessed by comparing accuracy of adopted models, convergence time and processing speed in generating such complete assignment scheduling using 2015/2016 academic session data of Federal University of Petroleum Resource Effurun Nigeria. Results obtained shows that GANN and PHMM are best for task. Hybrids have successfully proven to outperform single models as convergence time depends on parameter choice. Models yield a complete and valid schedule for examination scheduling at Federal University of Petroleum Resources Effurun.

Keywords : Examination, constraints, timetable, schedules, timeslots and hybrid models

1. INTRODUCTION

Examinations often culminates as the focal point of every University education as it drives home the idea that students have been taught and must be tested to ascertain if they adsorbed efficiently – what has been delivered by the instructors. In this guise, examination is often plagued by many activities that affect both students and instructors alike. Thus, examination timetabling seeks to assign courses that have been previously taught and delivered by instructors to a set of enrolled students (that undertakes the course or class) so that it yields an assignment of events (called lecture exams) into a limited set of specific timeslots and rooms subject to a a set of hard and soft constraints (Al-Milli, 2011; Alowwosile, 2016). The hard constraints must be satisfied or fulfilled under all circumstances; while the soft constraints may be fulfilled (if possible).

Scheduling has often been grouped into the field of problem solving – which seeks to resolve challenges that continues to plague the society. Previous studies and report shown argued that Government has no requisite funds to provide adequate infrastructure and the much needed laboratories to allow for effective learning – as all the needs of the University can and may never be met and yield as outcome, an improved quality of education received by Nigerian students. These constrains among others, continues to impede the effective planning and management of timetable schedules (lecture and examination).

Examination forms the crux of the education system as well as foundation on which certifications are awarded to deserving students having been judged worthy in character and learning to be bestowed any/all responsibilities deserving of certificate he/she possesses (NEEDs Report, 2015). Also, further survey recounts that these challenges are far from being resolved in nearest time as the Nigerian government surely perceives the Education sector as money-gulping – irrespective of its human capital developmental mantra. This consequently, necessitated the downward review of monetary allocation as clearly seen from the Federal Government's 2016 Budgetary Allocation – for which 8-percent of the total allocation goes to fund the educational sector (2016 Budget).

The main challenge of examination scheduling is that it aims at a complete, feasible and optimal assignment of events or a number of examinations for a set of students into room allocations and timeslots over a fixed period of time; while still seeking to satisfy a set of constraints on both invigilators and students (candidates). Thus, an examination overlaps must be avoided, while the schedule ensures a fair spread as much as possible of all exams. In Examination timetabling, there is usually a period range, not to mention: (a) shortage of rooms, (b) increase in number of students admitted on yearly basis, (c) not all admitted students graduate at proposed timeframe due to carryovers, (d) exam schedules occur within a fixed time period – unlike, timetable for normal classes extended over more flexible timeframe for which it must delivered. These and many other factors continue to mitigate against a successful schedule of the examination timetable (Ojugo et al, 2016; Alowosile et al, 2016).

Unlike regular timetable problems, examination timetable must meet students' and instructors preferences as much as possible – since rooms can be shared by more than one class or resource such that different courses can be examined in the same room at same timeslot. But, examination timetabling represents a difficult computational problem due to the strong inter-dependencies between exams caused by the many-to-many relationship between students and exams. Examinations are scheduled to meet room_type_capacity constraints and to avoid overlapping of times for student – making examination timetable scheduling a constraint satisfaction task.

1.1 Related Literature on TTP

Ojugo et al (2016) employed simulated annealing algorithm with 3-heating strategy (with reheating, adaptive cooling and exponential cooling methods) for academic scheduling at the University of Benin, and adopted a rule-based preprocessor model to yield initial solutions. Their results showed that SA with exponential cooling proved best with solution converging after 2.112seconds, and that convergence of SA is dependent on parameters like start population, initial/ final temperature as well as random swaps applied in the cooling strategies. While the model offers great insight into the task at hand, it does not provide us with the ability to discern from the experimentation if the results compares better to other methods. Also, there is no means of storing the schedules so that model can backtrack its previous schedules and pick the best for the semester at any single time.

Alowosile et al (2016) investigated a number of existing works for solving University examination timetabling problem with a view to resolving the conflict issues and inability to keep memory the best solution(s) encountered while using the automated design. They used the modified genetic algorithm and compared their results against standard genetic algorithm performance in generating possible timetables schedule. In their quest for better convergence of an optimal solution, their results of the modified GA used to schedule 2013/2014 rain semester exam at Ladoke Akintola University of Technology, Ogbomoso, Nigeria with a task involving 19,127students, 200-courses, 53-venues and 2-weeks (excluding Saturdays and Sundays) outperforms standard GA and maintains its accuracy level with increase in problem size; whereas standard GA lost its effectiveness as the problem size grows.

While, their work provides useful insight into further investigation of TTP, we observe these errors: (a) their study did has no sample output schedule, (b) did not account for course-room-time conflict (if schedule yields complete, consistent assignment of instructors to courses with venues and timeslots with no conflicts) – then there will be no need for the study since classes have rooms appropriately allocated to them, (c) instructor preferences out-weights those of student and thus, was not also accounted for, (d) instructors have ranks and thus, must be catered for since such senior ranked instructors are often times engaged in other administrative duties, (e) challenges with adjunct (part-time) instructors were not taken into consideration, (f) though, the study considered only examination – distance between venues for students having concurrent examination was not taken into account, and (h) their study initially set out to simulate and store results as generated with each iteration made so that they can easily detect through the stored results, the best schedule over time (this goal was not achieved).

1.2 Statement of Problem

1. As a non-Euclidean, complex, chaotic, dynamic, highly multi-dimensioned and multi-objective task with its range of applications as a scheduling task makes it **critical** as a determinant in many activities that resonates a University system. **Supervised** schedules have proven to be time-wasting especially with examination – where some models yield redundant and/or inconclusive, unsatisfying result with a lot of copy and paste work without recourse to instructor preference. And, results in many unresolved conflicts and swaps. To curb this, see Section III.
2. The study is often hampered with dynamic and chaotic feats such as instructors' ranks and preferences, student enrollment and preferences, distances between buildings, class type and capacity etc – yielding the unavailability of a precise/concise datasets that is often littered with noise, ambiguities and partial truth. This must be resolved via a robust search method as in Section III.
3. Processing speed challenges allows us to compare these models as in Section III. We need to resolve conflicts of statistical dependency imposed on the dataset in use and that of encoding dataset unto the model via the process of discretization. This will help the various models to avoid overtraining, overparameterization and over-fitting as in Section III/IV.

The **goal** is to compare various models to ascertain the best model suitable for the Examination timetable scheduling for Federal university of Petroleum Resources Effurun.

Dataset Used
Table 1 presents a summary of the dataset used.

Table 1: Size of Dataset for each Session

Items	First Semester	Second Semester
Invigilators	210	215
Students	10,917	11,012
Rooms	25	25
Courses Examined	360	344
Buildings	6	6
Time Frame	12-days	12-days

2. MATERIALS AND PROPOSED METHODOLOGIES

Soft computing is an inexact science that seeks to propagate observed data, as the model seeks the underlying probabilities of data feats of interest to yield an expected output, chosen from a set of possible solution space and is guaranteed of high quality – even with noise and ambiguity applied at its input. It exploits historic data to perform quantitative processing and explores a solution space to ensure qualitative knowledge as output, and experience as its new language (Ojugo et al, 2013). Our study samples: various hybrids and some known heuristics that 'claims' to have been successfully employed in similar or the same task; while, we employ the fuzzy model to act as a benchmark to measure their performance.

2.1 Rule-Based Preprocessor

Ojugo et al (2015; 2016) described in full the details of the adopted and adapted fuzzy logic system. It chooses between different control actions and transforms them into a fuzzy set value, (Nascimento, 1991; Ludmila, 2008; Ojugo et al, 2016) to yield recursive heuristic that assigns an event of time_space to lecture-class_student-enrollment slots suited to the problem domain. Also, the model's basics function has the data files of lecture-classes, room_space/buildings, department_buildings, distance_matrix, student_enrollment and inclusion data. Thus, using these structures – the system builds an internal database used to perform the scheduling. Its internal processes are such that checks distances between buildings, room_space_type and time allotted, and compares cum resolves all room_space and time_slot conflicts.

It also tracks and updates hours already scheduled. Thus: scheduling is done by department, so that each move is generated as loop over all the departments. The departments are chosen in order of size, with those having the most classes or lectures being scheduled first. The model first scans all currently unscheduled lectures. It then attempts to assign them to the first unoccupied rooms and timeslots that satisfy the rules governing constraints. Constraints for room capacity is difficult to satisfy, larger classes are scheduled first to result iteration and speed of processing as well as avoid conflicts in room exhaustion (Ojugo et al, 2015).

In most cases, only rooms and timeslots satisfying all rules will already by occupied by previously schedule; And in such case, the system attempts to move a lecture/class into a room and timeslot and allow the unscheduled lecture class to be scheduled. Next, system searches the schedule to select those with higher cost by checking all medium and soft constraints (e.g. how close a room match a class-size, how many students have lecture-time conflicts or clashes, which sporting activity conflicts with lecture-time, the student enrollment affected, if a lecture is in a preferred timeslot, room or building, and so on). If a poorly scheduled lecture class is identified, the model searches the space, maps or swaps it into a more comfortable timeslot – so that the hard constraint are still satisfied, but the overall cost of medium/soft constraints are reduced.

Room swapping process continues, provided all the rules are satisfied and no cycling (swapping of the same lecture) occurs. Once all the departments have been schedule, iteration cycle is complete. The model continues until complete iteration yields no further change in a schedule. There are many rules dealing with room_time conflict, room type, room_priority etc – many of which are complex. Our fuzzy model yields partial schedule as output (though, it is unable to assign all the given classes to room_time slots). Output is grouped into: list of unassigned classes from constraint conflict, and list of all assigned cum associated instructors/student_enrollment to room_time. The basic rules for implementing these constraints include (Ojugo et al, 2015):

a. IF room_size > student_enrollment AND {no conflict in room} THEN ASSIGN Room to the Lecture-Class.
b. IF Instructor = (Professor > Adjunct > Reader > Senior Lecturer > LecturerI > LecturerII) AND {NO room_time conflict} THEN (allot room to most senior instructor as preference).
c. IF (Time = allotted lecture) AND Student_has_Sports THEN Adjust Student OR Class_Time.

2.2 Genetic Algorithm Trained Neural Network (GANN)

Ojugo et al (2016) described GANN model. Adapting to task at hand, model is initialized with 30-solution rules. It fitness is computed to form new pool selected via tournament method, which determines individual solutions selected for mating. We apply a multi-point crossover to help net learn all dynamic and non-linear data feats in interest; while, mutation help to re-introduce diversity and chaos as well as ensure that the best fit schedules are chosen. Solutions are chosen by the model that corresponds to crossover points (since all genes are from a single parent initially). As new parents contribute to yield new solutions whose genes are combined, mutation will yield solutions that further undergoes mutation as they are allocated new random values that still conforms to our belief space via swaps. The number of mutation applied depends on how far CGA is progressed (how fit the fittest solution in pool), which equals fitness of fittest solution divided by 2. New solutions replace older ones of low fitness to create a new pool. Process continues until individual with a fitness value of 0 is found – indicating that the solution has been reached (Branke, 2001).

Initialization/selection via ANN ensures that first 3-beliefs are met; mutation ensures fourth belief is met. Its influence function influences how many mutations take place, and the knowledge of solution (how close its solution is) has direct impact on how algorithm is processed. Algorithm stops when best individual has fitness of 0 (Campolo et al, 1999; Dawson and Wilby 2001b).

Ojugo et al (2016) CGA belief rules are as thus:

a. Normative belief – each invigilator and students is bound to only an examination at a given timeslot).
b. Domain belief – sets boundaries for agents such time of exam set between 8am to 5pm, list of venues (buildings and room types) and their corresponding capacities, all courses/exams, all

invigilators, student enrollment for each exams, exclusion of adjunct instructors from exams but schedule their exams, Professors/Deans/Directors of units scheduled on supervisory role etc.

c. Temporal create data structure of semester-based courses, invigilator (exclude adjuncts) enrolled for each semester, invigilator preferences, student enrollment, no preference of students enrolled in semester-based exam taken into cognizance, etc

d. Spatial – creates the data structure of buildings and their distances, rooms allocation and distances between all the rooms in various building, laboratory, pavilions etc

Our Influence function mediates between belief space and the pool – to ensure and alter individuals in the pool to conform to belief space.

2.3 The Profile Hidden Markov Model

The Hidden Markov Model is a double embedded chain that models complex stochastic processes (Bhusai and Patil, 2011; Masoumeh, Seeja, and Afshar, 2012). The Markov process is a chain of states with probabilities associated to each transition between states. In n-order Markov, its transition probabilities depend on **current** and **n-1 previous** states. A Hidden Markov model determines state generated for each state observation in a series or sequence (solution space). In the exam scheduling, an event schedule not accepted by the trained HMM, yields high probability of being swapped (Srivastava et al., 2008; Dheepa and Dhanapal, 2009). Traditional HMM scores data via cluster method, which we have now adapted to depend on 3-value profiles that represent (hard, medium, soft) constraint. The probabilities of initial set of solutions are chosen; It then checks to see if an event has been scheduled using each profile created. HMM maintains in memory, an event timeslot to help reduce number of swaps made, which in turn – reduces the rate of false-swaps or moves to be made. HMM is initially trained with normal behaviour of each profile, and works on these constraints, which are classified into profiles (Tripathi and Pavaskar, 2012).

However, profile HMM as a variant of HMM, aims to deal with the fundamental problems of the HMM by: (a) it makes explicit use of positional (alignment) data contained in the observations or sequences, and (b) it allows null transitions, where necessary so that the model can match sequences that includes insertion and deletions as well as make way for good swaps to be made (Ojugo et al, 2014). In exam scheduling, O is each transaction rule, T is the time taken to make a swap, N is the number of assignments to yield a complete feasible and optimal solution in the (hidden) markov model, α is alphabet of model, M is the number of symbols in the alphabet, π is the initial schedule or state of the assignment database, A is state transition probability matrix, a_{ij} is probability of a transition from a state i to j, B contains the N probability distributions for the transactions in the knowledgebase (one assignment for each state of Markov process); while $HMM = (A, B, \pi)$. Note that though, the parameters for HMM details are incomplete as above; But, the general idea is still intact (Ojugo et al, 2014).

We align multiple assignment (data) rules as a sequence with significant relations. Output sequence helps us to determine if an unassigned course to be examined is related to the sequence belonging either properly assigned or unassigned class that comprise the Bayesian network. We then use the profile HMM to score assignments based on constraints (hard, medium and soft) as well as the necessary requisite decision. The circles are **delete** states that allows null and unclassified states which moves can be changed in the knowledgebase, **diamonds** are insert states that allow swaps in assignment upon which the knowledgebase is updated for classified false-moves; while **rectangles** are matched states that are accurately scheduled into a profile type in the HMM (Ojugo et al, 2014).

Match and insert are emission states in which observation(s) are made as PHMM passes via states. Emission probability corresponding to B is computed based on symbol frequency (assignment of exams) that can be emitted at a particular state – though, they are positional-dependent (in contrast to standard model). The emission probabilities are derived from Bayesian net (which essentially is our training phase). Finally, **delete** states allow model to pass via swaps or unassigned moves in the Bayesian net to reach other emission states. These swaps are necessary to prevent it from over-fitting of data as in fig 1 (Ojugo et al, 2014). To calculate probabilities for each possible swap, the model uses a forward algorithm that computes the desired probabilities recursively by reusing scores calculated for partial sequences as in Eq. 1-to-Eq. 3 respectively as thus:

$$F_j^M = Log\frac{eM_j(xi)}{qxi} + log(aM_{j-1}M_j exp\left(F_{j-1}^M(i-1)\right) + aI_{j-1}M_j exp\left(F_{j-1}^I(i-1)\right) + aD_{j-1}M_j exp\left(F_{j-1}^D(i-1)\right) \qquad (1)$$

$$F_j^I = Log\frac{eI_j(x_i)}{qx_i} + log(aM_jI_j exp\left(F_j^M(i-1)\right) + aI_jI_j exp\left(F_j^I(i-1)\right) + aD_jI_j exp\left(F_j^D(i-1)\right) \qquad (2)$$

$$F_j^D = log(aM_{j-1}D_j exp\left(F_{j-1}^M(i)\right) + aI_{j-1}D_j exp\left(F_{j-1}^I(i)\right) + aD_{j-1}D_j exp\left(F_{j-1}^D(i)\right) \qquad (3)$$

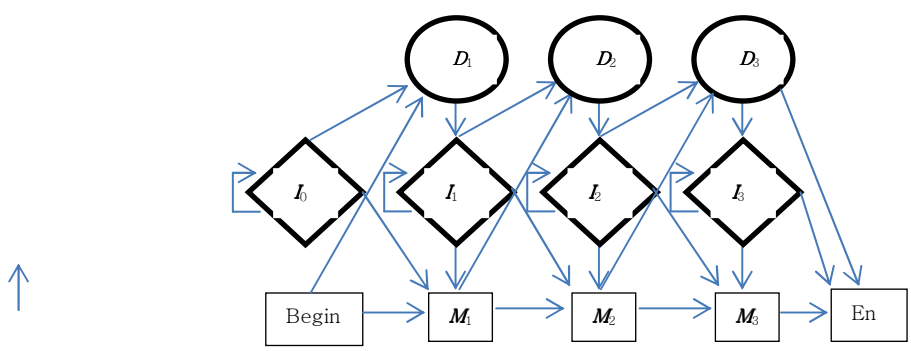

Fig 1: PHMM with 3-Match

2.4 Genetic Algorithm Trained Particle Swarm Optimizer

Ojugo et al (2015) PSO is a population model that attempt to simulate motion by investigating collective intelligence cum socio-cognitive of swarms – to specify a model of randomly initialized solutions propagated in space towards an optimal result, over a number of moves based on large amount of data about the domain, assimilated and shared by the swarm. It aims to generate particles (solution) adapted to an environ, using its task objectives and the corresponding constraints – so that in a number of moves, desirable traits (feats of interest) evolves and remain in the swarms' composition as the result set generated, over those of weaker desirable traits (Hassan et al., 2005). PSO is continuous.

Thus, to handle discrete design and variable(s), they are often modified (Kennedy, Eberhart and Shi, 2001) as thus:

a. **Particle Position/Velocity:** A particle is a point in space that changes its position from one move to another based on velocities updates. The positions X_i and velocities V_i of the initial particle swarm are randomly generated using lower and upper bounds of design variables values (X_{min} and X_{max}) as in Eq. 4 and Eq. 5 with initialization process that allows swarms to be randomly distributed in space as:

$$X_o^1 = X_{min} + rand(X_{max} - X_{min}) \quad (4)$$

$$V_o^1 = \frac{X_{min} + rand(X_{max} - X_{min})}{\nabla t} = \frac{Position}{Time} \quad (5)$$

b. **Velocities Update:** Each particle's velocity is updated in time t+1 via fitness values (which is function of particle's current position) in the solution space at t. Fitness value determines particles with best global-value (P_gi) in current swarm and the best position of each particle (P_i) in time (current and previous moves). Velocity update uses effect of the current motion V^i_t to yield search direction V^it+1 for the next iteration. To ensure good coverage and avoid local minima entrapment, it uses a uniformly distributed rand() along with 3-weight factors to effect a new search direction namely thus: (a) current motion/inertia factor ϖ, (b) particle's own memory (self-confidence) factor ϕ1 and (c) swarm influence (confidence) factor ϕ2 as in Eq. 6:

c.

$$V_{t+1}^1 = \omega + V_{(t)}^1 + \varphi_1\, rand() \frac{(P^1 - X_t^1)}{\nabla t} + \varphi_2\, rand() \frac{(P_t^2 - X_t^2)}{\nabla t} \quad (6)$$

Where
V^it+1 = Particle Velocity **i** at time t+1,
ϖ+V^it = The current motion
X^it = Particle position in time t and t is time
ϕ1 = self-confidence factor with value range 1.5 - 2
ϕ2 = swarm confidence with value range 2 - 2.5
ϕ1*rand() [(Pi-Xit)/∇t] = Particle's influence
ϕ2*rand() [(Pgt-Xit)/∇t] = Swarm's influence

d. **Position update:** Position update is in 3-steps: (i) velocity update, (ii) position update and (iii) fitness calculation – all repeated until a desired convergence criterion is reached – for which, the stop criterion is set (i.e., maximum change in best fitness is smaller than specified tolerance for a specified number of moves as in Eq. 7, which describes particle position update:

$$X_{t+1}^1 = X_t^1 + V_{t+1}^1 * \nabla t \left| f(P_t^g) - f(P_{t-g}^g) \right| \in \varepsilon \quad (7)$$

Design variables in PSO can take any value, even outside their bounds constraint, arising from their current position and updated velocity (a function of rapid growing vector velocity), which causes particles to diverge; rather than converge. To avoid such, variables that violate bounds are artificially brought back to its nearest side constraint via Eq. 8 (helps avoid particle velocity explosion and handles functional constraints using a linear exterior penalty).

$$f(x) = \varphi(X) + \sum_{i=1}^{N_{max}} X_i * max[0, g_i(x)] \quad (8)$$

Traditional PSO notes that each particle is a solution in space, while the data are encoded as string of positions representing a multi-agent task in multidimensional space. All dimensions are independent of each other – so that updates of velocities and particles are performed independently in each dimension (a merit of PSO). For permutation task like exam scheduling, exams are both independent and mutually inclusive of each other. Thus, 2-or-more positions can have same value after an update. This breaks permutation rule and implies that conflict in exams must be resolved (if they are prerequisite to each other, exams of same department or they have a common set of enrolled_students) must be resolved. Particle velocity is added on each dimension to update each particle (exam) so as to ensure distinctiveness.

If velocity is larger (exam has been scheduled in same room and at same time) with another exam, and the room capacity is exhausted or will not accommodate remaining students, such exam (particle) explores more distant areas via swapping and is thus, more likely to change to a new permutation series. A new velocity in such scenario signifies possibility of particle change (velocity update formula remains same). Also, update in velocity that causes particles (exams) to change can only be limited to absolute values – giving them distinct placement and assignment, which also represents the difference between particles (exams).

With PSO modified, its only shortcoming is that if an exam (particle) tries to follow same sequence as the neighborhood best (nbest), it gets trapped at local minima and may never get swapped into an optimal space. Thus, the particle stays in the current position forever (since it is identical to nbest) – so that only a new mutation factor introduced, will randomly swap such particle in a permutation task; Else, the mutation factor is ignored (Abarghouei et al, 2009; Kilic and Kaya, 2001).

2.4.1 Gravitational Search Neural Network Model

GSA is based on laws of gravity and motion. It considers a body of masses, where every mass represents a solution to a task. Law of gravity states that each particle attracts another as the gravitational force between them is directly proportional to the product of their masses and inversely proportional to their distances apart (Ojugo et al, 2013). Depending on their masses – agents of heavier masses attract those of lesser masses via the gravitational force (to measure their performance). Each N agents is initialized as: $Xi = (x_i^1 + x_i^2 + \ldots + x_i^d + x_i^n \quad (9)$

n is dimension of task, and also the position of the ith agent in dth dimension. Agents are randomly initialized so that at specific time t, a gravitational force is defined by Eq. 10:

$$Fij = G(t) = \frac{Mi(t) * Mj(t)}{Rij(t) + \varepsilon} \{Xj(t) - Xi(t)\} \quad (10)$$

M_i and **Mj** are object (i,j) masses, $R_{ij}(t)$ is Euclidean distance between (i,j), G(t) is gravitation constant at t and ε is a small constant. The randomly initialized gravitational constant G, decreases by time t to control the search's accuracy. Thus G is a function of initial value (G0) and time (t).

Total force acting on agent i in the dimension d is in Eq. 11 and fig 2:

$$F_i^d = \sum_{j \in kbest, j \neq 1} rand(i) * Fij \qquad (11)$$

rand – randomizes agents'initial states between intervals [0,1]. The acceleration of agent i at t in dth dimension is directly proportional to force acting on that agent, and inversely proportional to agent's mass by Eq. 12:

$$Aid(t) = \frac{Fid(t)}{Mij(t)} \qquad (12)$$

Next velocity of an agent, is a function of its current velocity plus its current acceleration calculated as in Eq. 13a,b:

$$V_i^d(t+1) = rand(i) * V_i^d(t) + A_i^d(t) \qquad (13a)$$

$$X_i^d(t+1) = X_i^d(t) * V_i^d(t+1) \qquad (13b)$$

Where $V_i^d(t)$ is the agent's velocity in dth dimension at time t, and rand is random number between [0,1]. Mass is calculated via fitness evaluation and are updated as Eq. 14:

$$Mi(t) = \frac{Fit(i) - worst(t)}{best(t) - worst(t)} \qquad (14)$$

Fit(t) is fitness value of an agent i at t. Best(t) and worst(t) indicates strongest and weakest agents based on to their fitness route. For a Min and Max task, they are defined by Eq. 15a,b:

$$worst(t) = \max_{j \in \{1,2 \ldots N\}} Fit(t) \qquad (15a)$$

$$best(t) = \min_{j \in \{1,2 \ldots N\}} Fit(t) \qquad (15b)$$

At start, agents are located as solution points in the search space such that with each cycle, the positions and velocities of agents are updated via Eq. 12 and 13. G and M as calculated are updated with each iteration or move, and stopped when an optimal solution is found. GSA use exploration (ability to navigate the space) and exploitation (ability to find optima around a good solution) in the shortest time. Exploration steps guarantee the choice of values or parameters of the random agents; while exploitation steps allows agents of heavier masses to move more slowly in order to attract those of lesser mass (Ojugo et al, 2013).

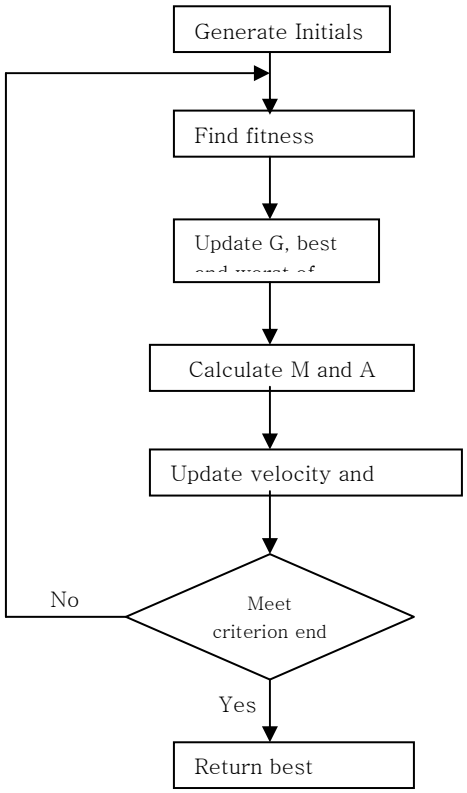

Fig. 2: Steps for Gravitational Search

2.4.2 Fuzzy-Based Simulated Annealing (FSA)

SA aims to strengthen glass or crystals by heating them till it liquefies and its molecules are allowed to slowly cool so that as they settle into states with lower energies, the model tracks and alters an individual solution state as well as constantly evaluating its energy via its energy function. Its optimal point is found by running series of Markov chain under various thermodynamic state [44]. The neighbouring state determined by randomly changing an individual's current state via a neighbourhood function.

If a state with lower energy is found, individual moves to it; else, if neighbourhood state has a higher energy, individual moves to that state only, if an acceptance probability condition is met. If not met, individual remains at current state (Ojugo et al, 2016). By heating the crystal or glass, its temperature(s) is initially set high, so that each individual is more inclined towards higher energy state – allowing individuals to explore a greater portion of the space and prevents it from being trapped at local optima. As model progresses – temperature reduces with cooling and individuals converge towards lowest energy states till an optimum point.

Ojugo et al (2016) described a rule-based SA algorithm as:

1. Select initial solution via preprocessor rule-based system
2. Select temperature change counter H = 0
3. Select a temperature cooling schedule,
4. Generate Initial schedule for individual solution state, energy & temperature S
5. Set initial best schedule S* = S.
6. Compute cost of S : C(S)
7. Compute initial temperature T_o
8. Set temperature T = T_o
9. Loop until temperature is at minimum
10. Loop until maximum number of iterations reached
11. While stop criterion is not satisfied or reached, Do:
12. Repeat Markov chain length (M) times
13. Select random neighbor S' to current schedule, (S' N_s)
14. Find neighbour state via neighbour function
15. If neighbourhood state has lower energy than current
16. Then change current state to neighbouring state
17. Else if the acceptance probability is fulfilled
18. Then move to the neighbouring state
19. Else retain the current state
20. Keep track of state with lowest energy
21. End inner loop: End outer loop

A major merit of SA over other methods is its ability to avoid being trapped in local minima via its uses of random search, that not only accepts changes that decrease a goal function f (for minimization task), but also allows some changes that increases it accepted with a probability as in Eq. 16:

$$P = \exp\left(\frac{-\delta f}{T}\right) \quad (16)$$

δf is increase in f, T is control parameter system temperature irrespective of the objective function involved. We implement SA with a structure of the following elements:

a. a representation of possible solutions
b. a generator of random changes in solutions
c. a means of evaluating the problem functions
d. annealing schedule - initial temperature and rules for lowering it as the search progresses.

This model uses exploratory search for multi-agent task, and is quite flexible in finding a better optimal point, even when a local minimum is present. The rule-based system helps us to initialize and yield candidates of low fitness. If a better solution is not found, best individual is chosen after a number of runs for a series of random walks until an optimal solution is found. SA is run on chosen 'fittest' candidates until solution is found on the neighbourhood size and function.

To randomly re-initialize the space, a temperature schedule is applied. After which, the neighbourhood function is then applied to randomly change individual (solution) energy states and compute best fitness with such individual tracked until a fitness of 0.51 is found. Model finds individuals of low energy to enter SA cycle early enough to apply temperature schedule as needed. Thus, a moderated Markov chain that accepts states with energies of lower or equal to current state's energy is used. The run continues till state of 0 energy is reached, to imply that solution is found (Ojugo et al, 2015).

We map SA to exam scheduling task via construct below:

1. A state is an exam timetable of the set: (a) I: a set of invigilator, (b) L: set of exams, (c) S: set of enrolled_students, (d) R: set of rooms, and (e) T: set of timeslot and intervals
2. A cost or energy E(I, C, S, R, T) such that:
   a. E(I) is cost of assigning more than maximum number of allowed class lectures in the same room space.
   b. E(L) is cost of assigning a certain exam within/at same timeslot in violation of the exclusion constraint etc
   c. E(S): cost of assigning a student to two or more exams that are in time conflict as well as cost of scheduling one or more exams that are not prerequisites of each other, exams timeslot requested by invigilator, or other requirements; plus the cost of exams evenly spread over a week.
   d. E(R): cost of assigning rooms of wrong size/type to exam.
   e. E(I): cost of assigning more time than required (being that most exams last 3hours for a 3unit course, exams may not start at exact time and overlap etc), and cost of imbalanced assignment etc.
3. A swap (move) is exchange of one or more of the following: lectures L_i with lecture L_j in a set L with respect to periods T_i and T_j, and/or with respect to classrooms R_i and R_j respectively – and is referred to as lecture class swapping.

We adopt SA with exponential cool with these constraints, and use rule-based preprocessor (output) in form of non-scheduled and scheduled events.

3. COMPARATIVE ANALYSIS OF PROPOSED APPROACHES

Performance evaluation of the above methods is based on 3-metrics namely: classification accuracy and processing time.

3.1 Classification Accuracy

For prediction accuracy, the table 1 is as thus:

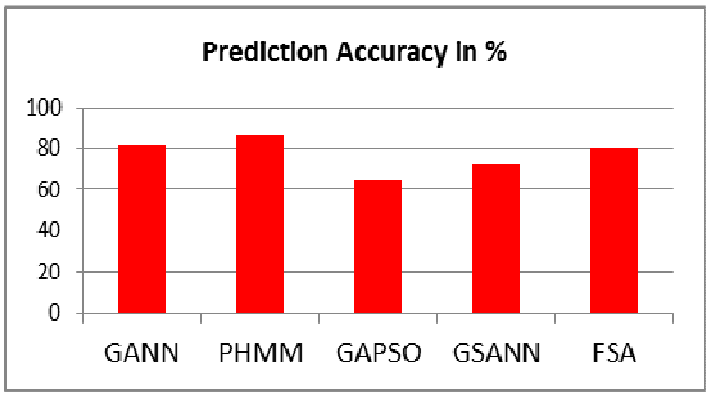

Fig. 5: Prediction Accuracy of Hybrids in %

Table 1: Prediction Accuracy of the Hybrid Models

Model	Accuracy
GANN	82%
Profile BHMM	87%
GAPSO	65%
GSANN	72
FSA	80

Table 1, shows prediction accuracy for hybrids and is further explained with graph representation as in fig. 5 respectively.

3.2 Processing Speed

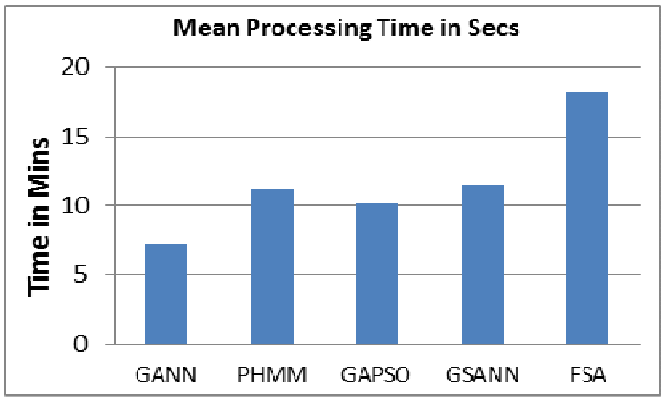

Fig. 6: Mean processing time in Seconds

3.3 Convergence Time

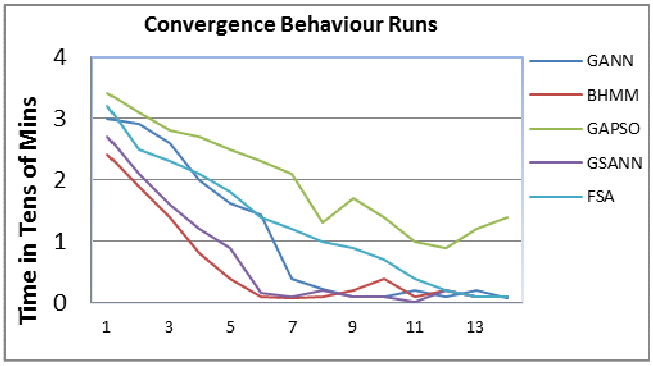

Fig. 7: Convergence time in Epochs

We observed from fig. 6, that GANN took about 101seconds (at best) to find its solution and converge after 130-iterations; PHMM took 156seconds after 187-iterations to converge; GAPSO took 141-seconds after 165-iterations to converge; GSANN took 213-seconds after 203-ietrations to converge; while, FSA took 168-seconds to converge after 198-iterations; Fig. 7 shows convergence comparison between all the various models. It is observed that hybrids converge in lesser time than Hidden Markov model; But, PBHMM model converges faster and better due to its use of profile clustering approach and its Bayesian learning method that allows training of individual profiles to ascertain good schedule and assignment. Hybrids have also been proven to converge in lesser time and iterations (Peter, 2014) when compared to other techniques. Thus, hybrid methods are quite significant when compared to other optimization approaches taken for consideration. Also, they all outperform the fuzzy system used as a preprocessor and benchmark model as a result of the parameter choice used.

3.4 Performance Measure

A cursory look at the data used for the comparison, we study these performance evaluations for the various models:

$$MSE = {}^{1}/_{n} \sum_{i=1}^{m} \{(Ypi - Yio)2\}^{1/2} \qquad (17)$$

$$MAE = {}^{1}/_{n} \sum_{i=1}^{m} |Ypi - Yio| \qquad (18)$$

$$MRE = {}^{1}/_{n} \sum_{i=1}^{m} \frac{|Ypi - Yio|}{Yio} \qquad (19)$$

Table 2. Model Performance for the Model in Use

Model	MSE	MRE	MAE	COE	COD
GANN	0.27	0.32	0.38	0.781	0.966
PHMM	0.28	0.31	0.35	0.753	0.921
GAPSO	0.45	0.53	0.76	0.588	0.812
GSANN	0.45	0.44	0.49	0.650	0.761
FSA	0.31	0.34	0.40	0.72	0.842

It is observed that GANN (memetic algorithm) performed reasonably in contrast with PHMM followed by FSA. This is attributed to the fact that PHMM and FSA uses clustering methods and random walks for Markov model. However, it is unfair to judge here, that GSANN and GAPSO are not suitable from results shown from the dataset used and results achieved. This is because, for the study – we used various parameters to decide which model is best suitable and to be adopted (in this case, for examination timetable scheduling for the Federal University of Petroleum Resources Effurun in which it is clear that our preferred choices are the GANN and PHMM). Details have been provided so that these experiments can be repeated and results verified.

3.5 Implementation Tradeoffs

Result trade-offs are as follows (Ojugo et al, 2013):

a. Researchers often display flawed and unfounded results, to validate their new or modified model rather than re-test limitations, insufficiency, bias and inabilities of existing ones – because, negative results (from their perception) are often less valuable. Also, most of such models aim to curb non-linearity and dynamism for the task they are predicting alongside discovering underlying properties of feats and in the historic datasets used to train models. We can do better if we consider these options as above. Reporting negative results often does not imply that such model cannot be used in a different scenario. Also, it can prove that such model may require tuning of parameters amongst many other reasons.
b. Researchers often use figures to show how well their model or forecast agrees with observed values (even with their limited dataset used for training model, that is often squeezed if made available). In some cases, observed versus predicted value/plots are not easily distinguishable – as such studies may not even provide numerical data to support their claim (though their model agrees well with the observed values). Some measure of goodness does not provide the relevant data.
c. Many studies suffer from inadequate dataset. If model aims to predict dynamic state, such ability should not be demonstrated with misleading results of limited dataset, inconclusive result and/or unclear contributions. Model must be adequately tested with methods laid bare so that process can be repeated to validate the usefulness and authenticity of such models.
d. Model validation is not an undertaking for a researcher or research group; but rather, a scientific dialogue. Improper model applications and ambiguous results often impede such dialogue. Study aims to minimize confusion in study of model as well as their corresponding implementation in examination timetable scheduling or any other scheduling task (where possible).

3.6 Notable Issues in the Hybrid Models

Some notable issues while experimenting are:

a. For hill-climbing methods, their speed often shrink as the solution approaches maxima; And thus, traps solution at local maxima. But GA, PSO, GSA and SA (proven to be best suited for tasks with a global optima) is effectively combined with ANN to speed up its last stages so that the convergence time of the hybrid depend on how close the initial population is to the solution as well as the number of recombination and mutation rates applied to the pool.
b. Also, PHMM with Bayesian Learning is a variant of the Markov process. Though, employs clustering technique, a difficulty experienced was issue of resolving statistical dependencies imposed on it with the adoption of various standalone models (combined hill-climbing and clustering techniques) as well as the data set used.

4. CONCLUSION AND RECOMMENDATIONS

Hybrids are tedious to implement along with the statistical dependencies imposed on it by individual adopted methods and its dataset. These must be appropriately encoded so that model can exploit numeric data and efficiently explore domain space to yield an optimal solution. Modelers must seek proper parameter selection and adjustment of weights/biases so as to avoid **over-fitting**, **over-training** and **over-parameterization** of the model.

Encoded via model's structured learning, it helps to address its issues of statistical dependencies between various heuristics used, highlight implications of such a multi-agent populated model as well as resolve conflicts in data feats of interest. Thus, as agents create/enforce their own behavioral rules on the dataset, hybridization helps to curb this (as CGA does in its belief space and operators as applied, and PHMM does in its states transition) to display probabilities of interest.

Models serve as new language to compile knowledge, help convey ideas and insight as well as to investigate dynamic and complex feats crucial to a task (Perez and Marwala, 2011). A detailed model helps us develop reasonably-applicable models even when not operationally applicable in a larger scale. Their implementation should seek feedback as more critical rather than seeking an accurate agreement with historic data. Thus, a model must balance its understandability and manageability, so that it can be fully explored (Ojugo et al, 2012).

REFERENCES

[1] Alowosile, O.Y., Oyeleye, C.A., Omidiora, E.O, Aborisade, D.O. & Odumosu, A.A. (2016): Comparative analysis of some selected cryptographic Algorithms, Computing, Information Systems, Development Informatics & Allied Research Journal. Vol 7 No 2. Pp 53-64, Available online at www.cisdijournal.met

[2] Bolton, R.J and Hand, D.J., (2002). Statistical fraud detection: a review, Statistical Science, 17(3), pp235-255.

[3] Chakraborty, R., (2010). Soft computing and fuzzy logic, Lecture notes, retrieved from http://www.myreaders.info/07_fuzzy_systems.pdf

[4] Delamaire, L and Abdou, H., (2009). Credit card fraud and detection techniques: a review, Banks and Bank Systems, 4(2), pp57

[5] Dheepa, V and Dhanapal, R., (2009). Analysis of Credit Card Fraud Detection Methods, Int. J. of Recent Trends in Engineering, 2(3), pp126

[6] Heppner, H and Grenander, U., (1990). Stochastic non-linear model for coordinated bird flocks, In Krasner, S (Ed.), The ubiquity of chaos pp.233–238. Washington: AAAS.

[7] Kennedy, C and Porter, A., (2013). Fraud detection and prevention, [online]: retrieved July 2015 from www.moss-adam.com/fraud_reviews

[8] Khashei, M., Eftekhari, S and Parvizian, J (2012). Diagnosing diabetes type-II using a soft intelligent binary classifier model, Review of Bioinformatics and Biometrics, 1(1), pp9-23.

[9] Kuan, C and White, H., (1994). Artificial neural network: econometric perspective, Econometric Reviews, Vol.13, Pp.1-91 and Pp.139-143.

[10] Mandic, D and Chambers, J., (2001). Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability, Wiley & Sons: New York, pp56-90.

[11] Michalewicz, Z., (1998). A survey of constraint handling techniques in Evolutionary computation methods, www.dhpc.adelaide.edu.au

[12] Minns, A., (1998).Artificial neural networks as sub-symbolic process descriptors, published PhD Thesis, Balkema, Rotterdam, Netherlands
[13] Ojugo, A., Eboka, A., Okonta, E., Yoro, R and Aghware, F., (2012). GA rule-based intrusion detection system, Journal of Computing and Information Systems, 3(8), pp 1182 - 1194.
[14] Ojugo, A.A., Emudianughe, J., Yoro, R.E., Okonta, E.O and Eboka, A., (2013). Hybrid neural network gravitational search algorithm for rainfall runoff modeling, Progress in Intelligence Computing and Application, 2(1), doi: 10.4156/pica.vol2.issue1.2, pp22–33.
[15] Ojugo, A.A., Ben-Iwhiwhu, E., Kekeje, D.O., Yerokun, M.O and Iyawa, I.J.B., (2014). Malware propagation on time varying network, Int. J. Modern Edu. Comp. Sci., 8, pp25 – 33.
[16] Ojugo, A.A., Ben-Iwhiwhu, E., R.E. Yoro., R.J. Ureigho., Yerokun, M.O and F.N. Efozia., (2014). Metamophic virus detection using profile hidden markov model, Technical Report: Centre for High Performance and Dynamic Computing, TRON-03-2013-01, Federal University of Petroleum Resources, Nigeria, p24-37.
[17] Ojugo, A.A., A.O. Eboka., R.E. Yoro., M.O. Yerokun and F.N. Efozia (2015a). Framework design for statistical fraud detection, Mathematics and Computers in Sciences and Engineering Series, 50: 176-182, ISBN: 976-1-61804-327-6.
[18] Ojugo, A.A., A.O. Eboka., R.E. Yoro., M.O. Yerokun and F.N. Efozia (2015b). Hybrid model for early diabetes diagnosis, Mathematics and Computers in Sciences and Engineering Series, 50: 207-217, ISBN: 976-1-61804-327-6
[19] Ojugo, A.A., D. A. Oyemade and D. Allenotor (2016): Solving For Computational Intelligence the Timetable-Problem, Advances in Multidisciplinary Research Journal. Vol 2, No. 3, Pp 67-84.
[20] Perez, M and Marwala, T., (2011). Stochastic optimization approaches for solving Sudoku, IEEE Transaction on Evol. Comp., pp.256–279.
[21] Peter, S., (2014). An Analytical Study on Early Diagnosis and Classification of Diabetes Mellitus, Bonfring International Journal of Data Mining, 4(2), pp7-13.
[22] Reynolds, R., (1994). Introduction to cultural algorithms, Transaction on Evolutionary Programming (IEEE), pp.131-139.
[23] Saleh Elmohamed, M.A, Fox. G and Coddington, P., (1998). A comparison of annealing techniques for academic course scheduling, Notes on Intelligence Computing, DHCP-045, pp 1-20. www.dhpc.adelaide.edu.au.
[24] Stolfo, S.J., Fan, D.W., Lee, W and Prodromidis, A.L., (2015). Credit card fraud detection using meta learning: issues and initial results, [online]: http://www.researchgate.net/publication/2282588
[25] Ursem, R., Krink, T., Jensen, M.and Michalewicz, Z., (2002). Analysis and modeling of controls in dynamic systems. IEEE Transaction on Memetic Systems and Evolutionary Computing, 6(4), pp.378-38