Australian Journal of Basic and Applied Sciences, 5(10): 1406-1413, 2011
ISSN 1991-8178

Particle Swarm Based Hyper-Heuristic For Tackling Real World Examinations Scheduling Problem

[1]Aftab Ahmed [2]Mazhar Ali, [3]Ahthasham Sajid, [4]Abdul Hussain Shah Bukhari

[1]Department of Computer Science, BUITEMS, Quetta.
[2]Department of Information Technology, BUITEMS, Quetta.
[3]Department of Computer Science, BUITEMS, Quetta.
[4]Faculy of Information and Communication Tech., BUITEMS, Quetta.

Abstract: Examinations scheduling is highly significant institutes to finalize the academic events of a semester. The is largely noticed by researchers of various domains as well. In this research work, PSO is involved to manage the overall hyper-heuristic solving method. The particle contains an ordered set of low level heuristics, each implement one by one on identical tentative solutions so the effectiveness of their order placement and selection mechanism can be measured by PSO. Consequently, each new generation gets converged set of particles. The partial but improved result promotes and furnishes to successive process whereas the substandard may be discarded. The research work has produced quite satisfactory and applicable results at end.

Key words: PSO, Scheduling Problem, Constraints.

INTRODUCTION

Universities scheduling can be classified under course (curricula based in most cases) and examinations timetables. Course scheduling instigates the academic activities while exam timetable finishes up the academic span. In spite of both scheduling share some basics characteristics however they can be differentiated by bespoke features.Academic examination scheduling is branch of combinatorial optimization of finite set of variables, their nonempty domain of values and certain number constraints. The solution demands to assign the appropriate venue (room) and human resources (Invigilator) on particular time-span (day and timeslot) for the group of students (Curricula).In recent decade, huge number of enrollment has been noticed in higher studies which exponentially have increased the need of academic automated systems specifically such as course scheduling, final and makeup exams in order to regulate educational cycle throughout the tenure. Examination scheduling is a set of exams $E = e_1, e_2, \ldots e_n$ over finite number of structured placements (timeslots) with respect to predefined time scale $T = t_1, t_2, \ldots t_n$, subject to a group of hard and soft constraints.

In general, the scheduling constraints can be classified as hard and soft. Almost in all problem instances, hard constraints are strictly followed and their violation cannot be tolerated under any circumstances. For example, two events cannot be placed on single venue or student(s) cannot avail two exams simultaneously. So the solution need to meet the basic criterion to remove all the hard constraints first, such partial solution is called feasible one. On other hand soft constraints, such as allocate enough time-gaps among invigilation duties for teachers or exams events, so that students can avail sufficient time to get prepared. Such constraints are required to be satisfied as much as possible but not mandatory at all.Due to size and complexity of the problem users have to compromise at minimum number of soft violations. The certain numbers of constraint are common in all academic institutes but bespoke version of hard/soft constraints also exists that makes a solution hard to apply various problem instances.

Scheduling the events is a tremendously vital job for regulating the educational cycle in all the academic institutes. It is a prominent NP hard problem in the domains of Operation Research and Artificial Intelligence. There is direct proportion between the time & space complexity and dimensions of problem. Classical solving techniques or manmade solutions need plenteous time and hard work to accomplish the task because of dependant and diverse parameters which are believed to be satisfied anyhow. More or less in the real world scheduling cases it is somewhat not possible to come up with entirely constraints free solution because one contented constraint may cause of violation to any other soft or hard constraint. However, each adopted solving technique has to negotiate over a small amount of unsolved violations. Solving Scheduling problem thus entail a concerted brainstorming, skillfulness and years of experience in the relevant field. Definitely this research field demands more study, investigation and classification of the precise techniques to formulate more effectual and proficient automated University scheduling. On the surface, scheduling is gridiron demonstration of academic resources containing faculty, curriculums, and enrolled groups of students situated together in crisscrossed slots and columns usually titled by session-time and venue. Scheduling provides an ordered chain of events where resources come to congregate in fixed time of intervals. The events container or scheduling

Corresponding Author: Aftab Ahmed, Department of Computer Science, BUITEMS, Quetta.
Cell# +92-333-7815354
E-mail: aftab.ahmed@buitms.edu.pk,

layout is very significant for detecting and solving all type of constraints. In fact, it provides a frame of reference from one event to other. The layout furnishes stand alone platforms or partial state-spaces for distinguished heuristics to shape up a complete search space. A number of vital objectives have been achieved by designing specific layout. Following sections are discussed for the mathematical and logical aspects of layout designing.

Related Work:

Carter(Carter, 1997) illustrated examination timetabling problem as *"The assigning of examinations to a limited number of available time periods in such a way that there are no conflicts or clashes"*. Examinations scheduling is supposed to be feasible if no any hard constraint is violated while the maximum elimination of soft constraints are highly desirable. A lower total penalty cost substantiates good quality solution. Plenty of research approaches have been studied for examining the automated course/exam scheduling in recent years. An inclusive survey of the classical scheduling solving methods was conducted by Burke *et al*. (Burke *et al*., 2004) S. Abdullah (S. Abdullah and H. Turabieh, 2008) effectively used hybrid GA along with sequential local search to resolve successfully timetabling problem. Aftab and Li (Aftab Ahmed and Li, 2010a) applied a novel approach of solving hard and soft constraints separately using two distinguished local heuristics.

First of all, the term hyper-heuristic was coined in 2000 around by Cowling *et al*. (P. Cowling *et al*., 2000) in which a heuristic had been chosen from a group of diverse heuristics using predefined selection mechanism. Cowling *et al*. (P. Cowling *et al*., 2002) introduced the term *`HyperGA'*, in their research genome of chromosomes were identified as sequences of local search heuristics implemented over distantly training staff. E. Burke *et al*. (E. Burke *et al*., 2005) also reported efficiency of using graph based hyper heuristics for solving exam timetabling problem.In (Schaerf, 1999) reported the PSO reaches the optimal solution significantly more faster than various optimization algorithms. In addition Shi(Shi and Eberhart, 1999) extended to describe rapid conversion rate and diverse dataset scalability of PSO. In (Kennedy and Spears, 1998) PSO shows remarkable performance and goal achievement ability in highly sophisticated continuous and binary problems. In the mean time(Shi and Eberhart, 1998) appreciated robustness of PSO for solving non-linear and dynamic problems. As for its scope for scheduling problems is concern, PSO is highly capable for combinatorial optimization problem especially tackling University scheduling. PSO is not only a substitute method but proved to be very effective for finer detection and solving competence over scheduling problem instances(Parsopoulos and Vrahatis, 2002a).

Problem Description:

The research work was investigated on real-world examination scheduling dataset of Faculty of Information and Communication Technology BUITEMS, Quetta. The Faculty comprises over five departments. The FICTis responsible to regulate spring and fall semester per annum respectively. More or less, FICT encompass 5 × 8 student groups, more often remedial/makeup exams also remain part of planning. Usually5 to 6 courses are used to offer to every group of credit hours. Although each department works as separate entity but exams are held collectively

Is illustrating the dataset specification of Faculty of ICT which comprises over Departments of Computer Science, Computer Engineering, Information Technology, Telecommunications and Electronics Engineering. It can be noticed that resources are somewhat insufficient against large number of events. Approximate eleven class rooms are for theory exams and total five labs for experiential exam (practical). Every department contributes with available number of invigilators. Plenty of courses are shared within faculty so it is tough task to accommodate students groups and invigilators without any conflict. Regular exams are required to be scheduled at beginning while remedial events and practical are planned to adjust at tailend. All working days are divided into three equal sessions of two hours each. Sessions are separated from each other by short interval. Not enough human and physical resources, plenty of exam events and highly complex constraints makethe problem extremely challenging.

Table 1: Dataset Specification.

Resources	Dept. of CS	Dept. of CE	Dept. of IT	Dept. of EE	Dept. of T.Com	Total
Rooms	02	02	02	03	02	11
Labs	01	01	-	02	01	05
Invigilators	05	05	03	06	04	23
exam per day	03	03	03	03	03	15
Working day						05
Exams Events	65	55	45	80	60	305

Terminology:

Exams	Indicates the total number of exams $E=\{e_1,e_2,e_3\ldots e_n\}$
Groups	Enrolled group of students
RoomCap	Depictsclass room capacity
Day_i	Working day where $i \in \{1, \cdots, 5\}$
Session	Session time length or number of periods, where *Session*= $\{Slot_1,Slot_2,Slot_3\ldots Slot_n\}$
Slot	Single exam placement
Faculty	Total number of invigilators
Rooms	Accumulated number of rooms in FICT, *Rooms*= $\{r_1,r_2\ldots r_n\}$
Equipments	Necessaryhardware equipments for specific course
S_{ij}	A complete day exams fixtures for a student's group where $i \in \{1, \cdots, \text{Gourps}\}$ and $j \in \{1, \cdots, D\}$
$LOAD_{ij}$	Invigilation Load of Faculty member, where $i \in \{1, \cdots, \mathit{Exams}\}$ and $j \in \{1, \cdots, \mathit{Day}\}$

Hard Constraints:

Almost in scheduling problem the group of hard constraints exceptionally should remain inviolate at any cost. The possibility of conflict could be caused by sharing of same room for two events or student(s) enrollment in two or more exams at time. In general the following hard constraints are largely recognized in academic scheduling problems.

i. *Exam Hard Constraint 1 (EHC_1):* Shows two or more simultaneous exams for same student(s) cannot be scheduled.

$$\sum_{i=1}^{\text{Exams}} \sum_{j=1}^{\text{Groups}} g_{ij}.\,\Delta(\text{Slot}_j\text{-Slot}_{j+1}) \leq 1 \forall\ g \in \{1 \cdots \mathit{Groups}\} \tag{1}$$

ii. *Exam Hard Constraint 2 (EHC2):*Defines that all the exams of semester courses must be scheduled.

$$\sum_{j=1}^{\text{Exams}} \alpha_{ij} \leq 1 \forall\ j \in \{0 \cdots \mathit{Session}\} \tag{2}$$

iii. *Exam Hard Constraint 3 (EHC3):* Invigilator must not be assigned for concurrent duties.

$$\sum_{i=1}^{\mathit{Session}\times\text{Days}} \sum_{j=1}^{\text{Load}} L_{ij} \leq 1 \tag{3}$$

iv. *Exam Hard Constraint (EHC4):* Class room capacity/resourcesmust be enough to accommodate/facilitate all the students of enrolled group.

$$\sum_{j=1}^{\text{Exams}} r_i\, e_j \leq (\mathit{RoomsCap}, \mathit{Equipments}) \quad \forall j \in \{1 \cdots \mathit{Rooms}\} \tag{4}$$

Soft Constraints:

Maximum Soft constrains are supposed to be wipe out from search-space but most of time such state cannot be achieved adequately.

i. *Exam Soft Constraint (ESC1):*The constraints violates if there is inadequate gap among exams.

$$\sum_{i=1}^{\text{Exams}} |\text{Slot}_i - \text{Slot}_{i+1}| \quad >= 1 \tag{1}$$

ii. *Exam Soft Constraint (ESC2):* Student(s) who has/have consecutive exams on the similar day must be assigned to the unchanged room.

$$\sum_{i=1}^{\mathit{Session}} \alpha_i \leq r \mathit{\ where\ } r \in \mathit{Rooms} \tag{2}$$

iii. *Exam Soft Constraint (ESC3):* Scheduling more than two consecutive exams for student (s) are supposed to be avoided.

$$\sum_{i=1}^{\text{Groups}} \alpha_i > 2 \tag{3}$$

iv. *Exam Soft Constraint (ESC3):* Number of investigation per teacher should not go over two per day.

$$\sum_{i=1}^{\mathit{Faculty}} L_i \leq 2 \tag{4}$$

Fitness Function

The fitness function(Aftab Ahmed and Li, 2010b)used in this research work is given as

$$\textit{Maximize } f(x) = \frac{1}{1+\sum_{i=1}^{n} \alpha EH_i + \sum_{j=1}^{n} \beta ES_j} \qquad (1)$$

The fitness function examines quality level of particles in each genome. Often outcome early elimination of hard constraints because they keep high penalty cost. The fitness values value between 0 and 1 by reason computing requirements. The coefficients α and β in expression are the penalty cost of particular constraint violation, on the other hand ES_i, and EH_jstand for the soft and hard constraints respectively.

Higher Level Mechanism:

Particle Swarm Optimization is population based solution for optimization problems. It was firstly introduced by Eberhart and J. Kennedy(Kennedy and Eberhart, 1995) the metaphor of this approach is taken from individual and collective behavior of bird flock, fish schooling, insects swarm and even dust particles swirl. The social activity performs these species usually for food search and impersonate itself as big entity to give stronger and integrated look to their enemies. Fundamental analogy is adopted from the inclination of birds to reach the swarm heart. Keeps each particle closer enough to its neighbor, avoiding any collision with each other and assembles a disciplined and ambitious cluster. Each particle gradually improves its current position keeping remember its best ever previous position and snooping position of swarm.

Algorithm 1 illustrates the standard Particle Swarm Optimization. First step contains initializing of each practical to random position. Assigning a diversely unique location over search space helps PSO not to trap into local optimum dilemma and it also speeds up the conversion and quality of performance (Parsopoulos and Vrahatis, 2002b). The other control parameters consist of population size (which is common in all population based methods), neighborhood size, group topology, weight of inertia cognitive as well as social coefficient. In most cases, these parameters may perform well particularlyin one problem and falls short for another, however their variant effects has been scrutinized on problem type, converging rate and overall performance (Carlisle and Dozier, 2001). In this research work the PSO has been used as top level heuristic to manage the performance quality of overall mechanism. Every particle in genome is partial solution and sequence of low level heuristics. At the end of each generation the *PBEST* and *GBEST*particles nominates on quality basis and that help to converge rest of particle optimal solution

Algorithm 1: Standard Particle Swarm Optimization.

```
1. Initialization:
2. Generate First Swarm by random position $\forall X_i$
3. Initialize the velocity $V_i$ with smallest random number
4. Evaluate Function: Calculate the fitness Level of Particles
5. Loop Until (Termination Criteria)
6. For i range (1,PopulationSize)
      a. $V_i^{k+1} \leftarrow \omega V_i^{k} + \alpha R_i (Pbest_i - X_i^k) + \beta R_i (Gbest_i - X_i^k)$
      b. $X_i^{k+1} \leftarrow X_i^{k} + V_i^{k+1}$
7. Revaluate Function:
      a. If $f(X_i^{k}) < f(Pbest_i)$ than $X_i^{k} \leftarrow Pbest_i$
      b. If $f(Pbest_i) < f(Gbest_i)$ than $Pbest_i \leftarrow Gbest_i$
8. End Loop
```

- x represent the current position of particle in search space.
- $x_{(k+1)}$ is calculated outcome of the equation.
- W represents weight of inertia factor, contains value between 0.4 to 1.4 usually.
- $V_{ik}$is current velocity of practical.
- C_1 is first constant, used as 'self confidence' can contain value between 1.5 to 2, the value is responsible to empower personal influence of the particle.
- P_{Best} is best ever position of particle in swarm until current iteration.
- C_2identifies as second constant, another self confidence value can have range between 2 to 2.5, accelerate the swarm influence.
- G_{Best} is swarm influential motivation attracts each particle to group focal point.

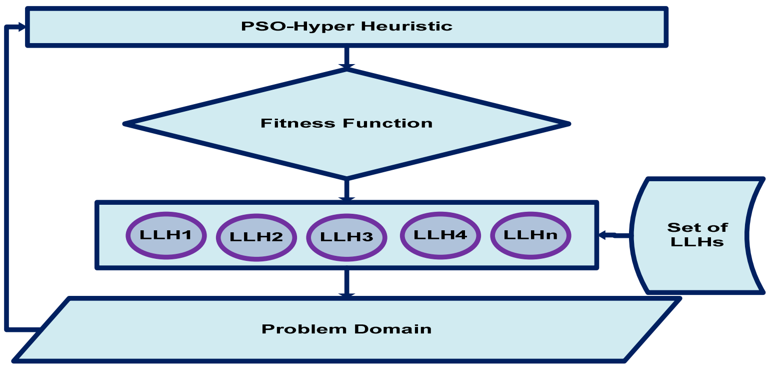


Fig. 1: Interaction with Low Level Heuristics.

Set of Low Level Heuristics:

The group of low level is the core of any hyper-heuristic methodology. Each heuristics fundamentally makes a specific alteration in partial solution either on the basis of random or pre-defined manner. The LLHs are classified in various distinguished categories as it is mentioned in Table 2. According to functionality the LLHs belong to *shift* or *swap*operation. The shifting requires anempty slot on other side interchanges two scattered events. The scope or applicability range of some LLHs is day level transaction and on the other hand, some can be workable within sections or columns. The random or low level heuristics used to spread up theevents in order to create spacesfor acceleratingthe events transition. Progressive LLHs are although expensive in computing cost but return a certain improvement or left previous status behind.

The gray-shaded Low Level heuristics:

The principal merit of constructing the hyper-heuristic is to apply it over other classes of scheduling problem with minor alteration or supplementary code; in addition many of previously developed components can be used directly. In fact, every problem instance may need its own version of evaluation function to diagnose violation.

Table 2: Shaded Heuristics demonstrate the gray-shade (Common solver for multiple classes or diverse instances of problems) low level heuristics.

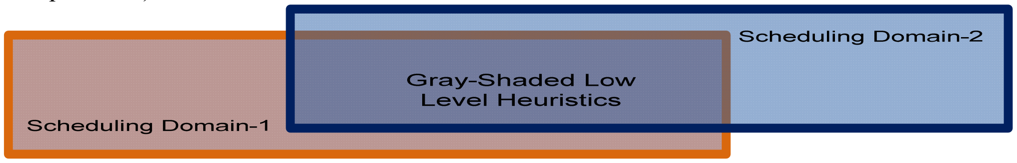


Fig 2: ***Gray-Shaded heuristics illustration:***

Table 2: Shaded Heuristics.

ID	Algorithm Name	Scope	Function	Interaction
LLH_2	Shift_MaxSiturated	Day	Shift	Semi –R
LLH_3	Shift_LessSiturated	Day	Shift	Semi –R
LLH_6	Swap_In	Session	Swap	Semi –R
LLH_7	Shift_RandDay	Day	Shift	Progressive
LLH_{10}	Swap_InRows	Session	Swap	Progressive
LLH_{11}	Shift_Dispersions	Day	Shift	Progressive
LLH_{12}	Swap_SingleDay	Session	Swap	Progressive

Low Level Heuristics (LLH_{11}): Shift_Dispersions_of _Exam.

```
DEFShift_ Dispersions_of _Exam
(t_1,t_2 ) = ScanWeek()
1.    IF|t_i − t_{i+1}| ≤ 1
Then
a.    Siturated_Day_key  = MaxOfDay(t_1,t_2)
2.    IF Less _Siturated_Day_key  NOTnone
Then
a.    Less_Shift_DayToEnd(Siturated_Day_key  ).
```

The operational and logical explanation of all the gray-shaded low level heuristics are given in details (Aftab Ahmed *et al.*, 2011). In this article only new exam specific heuristics are discussed. Algorithm LLH_{11} used to spreads up the exam events over layout. First it finds the difference between two timeslots, if there is no marginalgap than procedure moves the slot to less saturated day of the layout. The shifting process is based on progress instead of random pattern. So if there is any positive change found in solution than move is accepted otherwise it triggers the rollback action. Algorithm LLH_{12} represents the logic of exchanging the exam events within single day. The algorithm just makes the outcome finer to implement so it make slight improvements in fitness function.

Low Level Heuristics (LLH_{12}): Swap_SingleDayExam

```
DEF Swap_SingleDayExam:
1.    FOR k IN range (Sessions):
a.    IF  Layout[k] IS  None:
              key2 =  k
b.    ELIF  Layout[k]['Penlty'] != 0:
                            key2 =  k
c.    ELIF Layout[k]['PenType'] = = 'P':
                            key2 = (key1[0],key1[1],k)
d.    ELSE: key2 = None
2.    IF key1 AND key2:
a.    SwapSlots(key1,key2).
```

Experimental Results:

The computational approach is investigated the real world dataset of Faculty of ICT, BUITEMS, quality results have shown the perceptible potential and capability of adapted technique. The coding had been written on Intel® Core(TM) *i*3 CPU, M 350 @ 2.27 GHz, 2.0 GB RAM. The Python language version 2.6 was chosen to write project coding.The experimental results substantiate the correct research direction. The Hyper-Heuristics approach is examined over five departmental dataset. Shows the adopted parameters of PSO that were noticed particularly suitable for the case study. At the time of dataset initialization, it was tried at level best to scatter the events over the layout as much as possible.

Table 3: PSO parameters.

	Parameter	Value
1	topology	Ring topology
2	Neighborhood size	5
3	Max Evaluations	1000
4	Number of elites	1
5	Population Size	10
6	evaluator	Binary
7	Termination Criteria	Converging + Max iterations

is providing the precise details of outcome on each generation. The partial but improved solution is chosen from ten others of same genome. Later on the tentative results are promoted to next generation to make transaction with newly converged group of low level heuristics. Therefore, each step makes some finer changes over examination events which are scattered on scheduling layout. The Hyper-PSO stable and progressive state of tentative results after few generations. The each constraint processing history individually; gradually lessening number of conflict violations mostly in of hard constraints may be noticed. Rapid decline of hard constraints can be observed because they caught the immediate attention of LLHs due to higher penalty cost.Intermittent up-downs of penalty value for constraint SC_1 is obvious due to shifting of penalized timeslots from other slots. The identical situation is for SC_2 too. Whole penalty cost lessening under Hyper-PSO. Steadily increasing of saturation is reason of low rate of reduction penalty predominantly before finishing point, however very few violations remain unsettled. The fitness function progress throughout the program execution. The fitness scale almost is reached to*1*that shows the vast number of constraints have been eliminated from the search-space.

Conclusions:

The research work is illustrated in this article to investigate the Particle Swarm based hyper-heuristic approach to manage low level heuristics. PSO is proved to be a superbly accurate technique for tackling real-world exam problems. In Future, the approach is planned to implement on examinations and curriculum benchmark dataset together to evaluate the level of generality. In addition, research will be focusing on various efficient LLHs selection and move acceptance techniques in order to meet quality level solving methods.

Table 4: Outcome of ten partial solutions in each generation.

Gen	Worst	Gbest	Pbest	Average	Std. Dev
0	395	395	395		
50	420	375	386	399.9091	14.67961
100	400	360	376	385.6364	11.25409
150	357	330	344	348.5455	7.4211
200	337	302	313	321.5455	9.564138
250	316	294	304	310.6364	6.546338
300	308	280	293	297	7.334848
350	280	255	266	272.9091	7.217403
400	246	225	236	240.0909	6.122982
450	231	211	222	224.8182	5.58244
500	222	204	215	216.9091	5.088311
550	194	179	189	190.3636	4.201731
600	176	162	172	172.4545	3.856518
650	150	137	147	148	3.820995
700	125	113	123	123.3636	3.500649
750	111	100	110	109.4545	3.173756
800	90	80	85	86.81818	2.713602
850	68	60	65	65.81818	2.136267
900	45	39	41	41.81818	1.601136
950	33	28	29	30.90909	1.758098
1000	13	10	11	11.90909	1.136182

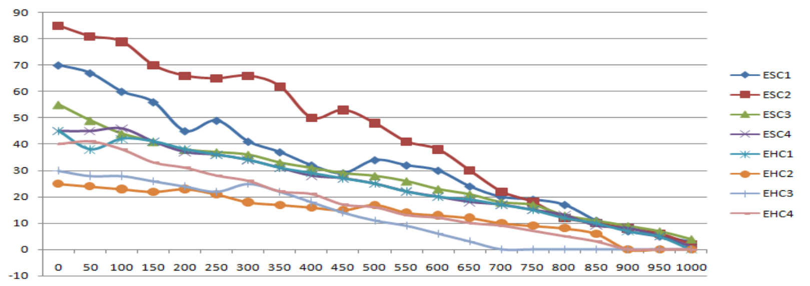


Fig. 3: Constraints elimination throughout the process.

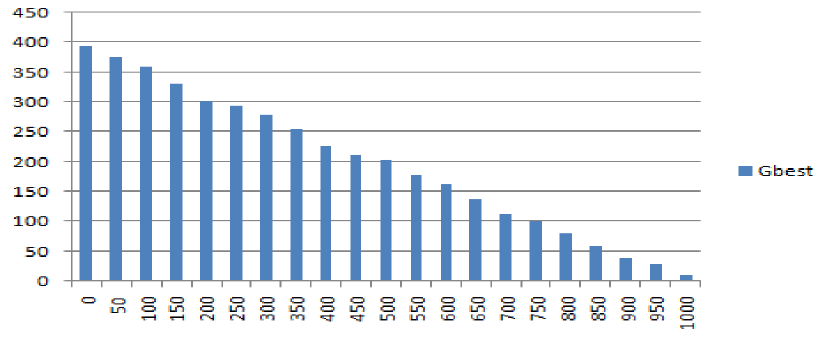
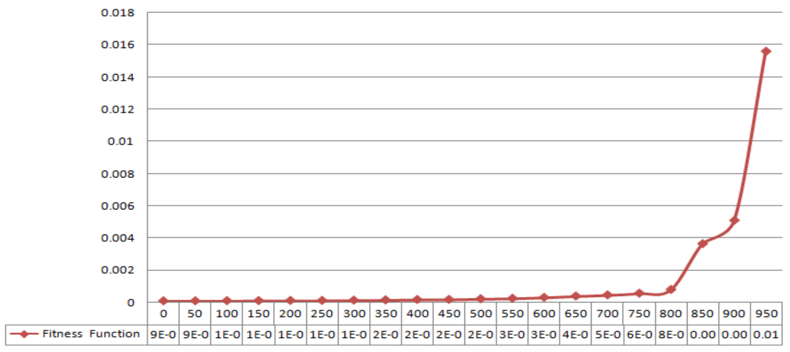


Fig. 4: Total Penalty Cost.



	0	50	100	150	200	250	300	350	400	450	500	550	600	650	700	750	800	850	900	950
Fitness Function	9E-0	9E-0	1E-0	1E-0	1E-0	1E-0	1E-0	2E-0	2E-0	2E-0	2E-0	3E-0	3E-0	4E-0	5E-0	6E-0	8E-0	0.00	0.00	0.01

Fig. 5: Fitness Function.

REFERENCES

Aftab Ahmed, Abdul Wahid Shaikh, Mazhar Ali, A.H.S. Bukhari, 2011. Hyper-GA for Solving Benchmark Scheduling Problems. Australian Journal of Basic and Applied Sciences, 5: 1657-1667.

Aftab Ahmed, Z. Li, 2010a. A Biphasic Approach for University Timetabling Problem. The 2nd International Conference on Computer Engineering and Technology (ICCET 2010), Chengdu, Sichuan, China, pp: 192-197.

Aftab Ahmed, Z. Li, 2010b. Solving Course Timetabling Problem Using Interrelated Approach. 2010 IEEE International Conference on Granular Computing IEEE, San Jose, California, USA, pp: 651-655.

Burke, E.K., Y. Bykov, J. Newall, S. Petrovic, 2004. A Time-Predefined Local Search Approach to Exam Timetabling Problems. IIE Transactions, 509-528.

Carlisle, A., G. Dozier, 2001. An off-the-shelf PSO. Proceedings of the workshop on particle swarm optimization, pp: 1-6.

Carter, M., 1997. EXAMINE:A General Examination Timetabling System. PATAT '97 Proceedings of the 2nd International Conference on the Practice and Theory of Automated Timetabling, pp: 363.

Burke, E., M. Dror, S. Petrovic, Qu., R., 2005. Hybrid graph heuristics in a hyper-heuristic approach to exam timetabling problems. The Next Wave in Computing, Optimization and Decision Technologies. Springer, pp: 79-92.

Kennedy, J., R.C. Eberhart, 1995. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks, pp: 1942-1948.

Kennedy, J., W.M. Spears, 1998. Matching Algorithms to Problems: An Experimental Test of the Particle Swarm and Some Genetic Algorithms on the Multimodal Problem Generator., Proc. 1998 IEEE World Congress on Computational Intelligence. IEEE, Alaska, pp: 74-77.

Cowling, P., G. Kendall, L. Han., 2002. An investigation of a hyperheuristic genetic algorithm applied to a trainer scheduling problem. Proceedings of the Congress on Evolutionary Computation, pp: 1185-1190.

Cowling, P., G. Kendall, E. Soubeiga, 2000. A hyperheuristic approach for scheduling a sales summit. In Selected Papers of the Third International Conference on the Practice and Theory of Automated Timetabling, PATAT 2000. Springer, Konstanz, Germany, pp: 176-190.

Parsopoulos, K.E., M.N. Vrahatis, 2002a. Initializing the Particle Swarm Optimizer Using the Nonlinear Simplex Method. Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, pp: 216-221.

Parsopoulos, K.E., M.N. Vrahatis, 2002b. Initializing the Particle Swarm Optimizer Using the Nonlinear Simplex Method. Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, 216-221.

Abdullah, S., H. Turabieh, 2008. Generating university course timetable using genetic algorithms and local search. Third 2008 International Conference on Convergence and Hybrid Information Technology ICCIT, pp: 254-260.

Schaerf, A., 1999. A Survey of Automated Timetabling. Artificial Intelligence Review, pp: 87-127.

Shi, Y., R.C. Eberhart, 1998. A Modified Particle Swarm Optimizer. Proceedings of the IEEE International Conference on Evolutionary Computation. NJ: IEEE Press, Piscataway, pp: 69-73.

Shi, Y., R.C. Eberhart, 1999. Empirical Study of Particle Swarm Optimization. Proceedings of the 1999 Congress on Evolutionary Computation. NJ: IEEE Service Center, Piscataway, pp: 1945-1950.