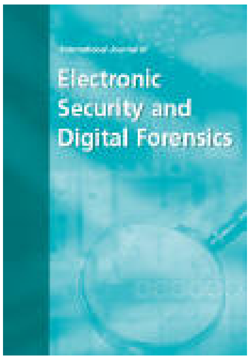


International Journal of Electronic Security and Digital Forensics

ISSN online: 1751-9128 - ISSN print: 1751-911X
https://www.inderscience.com/ijesdf

Management of electronic ledger: a constraint programming approach for solving curricula scheduling problems

Aftab Ahmed Shaikh, Abdullah Ayub Khan

DOI: 10.1504/IJESDF.2023.10045616

Article History:

Received:	17 November 2021
Accepted:	23 January 2022
Published online:	15 December 2022

Management of electronic ledger: a constraint programming approach for solving curricula scheduling problems

Aftab Ahmed Shaikh

Department of Computer Science,
Sindh Madressatul Islam University,
Karachi, 74000, Sindh, Pakistan
Email: aftab.ahmed@smiu.edu.pk

Abdullah Ayub Khan*

Department of Computer Science,
Sindh Madressatul Islam University,
Karachi, 74000, Sindh, Pakistan
and
Faculty of Computing Science and Information Technology,
Benazir Bhutto Shaheed University Lyari,
Karachi, 75660, Sindh, Pakistan
Email: abdullah.khan00763@gmail.com
*Corresponding author

Abstract: Curricula timetabling belongs to the scheduling and planning domain of artificial intelligence, the problem largely recognised by its key importance for initiating and afterward regulating the curricula events. In the literature the issue is reflected as a resources management job against puzzling constraints. The group of hard constraints requires the vital priority and must be removed, whereas the degree of solving of soft constraints upraises the quality scale and leads to optimal solution at the end. Constraint programming is one of the contemporary techniques that shape the research work presented in this article. The research investigates a constraint programming framework to examine over the various datasets. The study proposes and implements three incremental low-level heuristics operated by min-conflict algorithm approach for solving identical but unequal benchmark scheduling instances. The framework is designed in such way to provide fair chance of randomisation and incremental calculation to parameters in order to keep up the accuracy. The acquired prominent results validated the effectiveness and correctness of proposed methodology.

Keywords: heuristic scheduling; constraints programming; problem solving; electronic ledger management.

Reference to this paper should be made as follows: Shaikh, A.A. and Khan, A.A. (2023) 'Management of electronic ledger: a constraint programming approach for solving curricula scheduling problems', *Int. J. Electronic Security and Digital Forensics*, Vol. 15, No. 1, pp.88–99.

Biographical notes: Aftab Ahmed Shaikh earned his Doctorate degree in Computer Application and Technology from Beijing University of Aeronautics and Astronautics (BUAA) in 2010. He has more than 18 years of professional experience in teaching and research in different countries including Pakistan, China and Oman. He is associated with a number of reputable research communities and editorial boards. He is a well-published author of several research articles in reputable international journals and conference proceedings. He has supervised several dissertations and projects and leading a research group of computational intelligence (CI).

Abdullah Ayub Khan is currently pursuing his PhD at the Department of Computer Science, Sindh Madressatul Islam University Karachi. He has published around 20 research articles in well-reputed journals (such as IEEE Access, MDPI, Elsevier, Spring, Wiley) in the domain of digital forensics, cyber security, blockchain, hyperledger technology, and artificial intelligence.

1 Introduction

In curricula/exam scheduling, set of constraints is main design script for such problems. The constraints converge solution to shape onto positive adoptability and workability. In large, constraints are warped with each other on different layers and having correlative course of interaction. Their mutual and inversely proportional influence makes problem challenging enough to bring about the feasible task, particularly in real world cases. There are twofold groups of conditional rules recognised as hard and soft constraints. For all intents and purposes, the hard constraints are highly required not be violated, and reflected as an obligatory and basic measure of any solution. Soft constraints, on the other hand are also exceptionally important and preferred to be solved but as much as possible within finite computational resources. At various occasions, it is not likely to remove all soft violations. So, it may be established that for solving curricula and examination scheduling is meant to have relevant computational expertise and experience of the field. The obligatory academic problem continually demands state-of-the-art research and analysis techniques even for meager improvement. Curricula scheduling is a particularly challenging and estimable topic in combinatorial optimisation, which receive the attention of researchers from everywhere in the academia and a lot of novel research approaches have been surfaced to inspect the course/exam timetabling for last many years.

The research study in this article is inclined to constraint programming approach, curricula or any other scheduling is necessarily a job to fulfil requirements of numerous constraints, thus constraints logic programming (CLP) is a very natural and favourable option to illustrate and solve the problem. CLP is really a dynamic method that attain the focus of plenteous researchers (Barták, 1999b; Müller, 2002; Rudová, 2005) by reason of its likely companionably with the problem. Barták (1999a) defines, "constraint programming is an emergent software technology for declarative description and effective solving of large, particularly combinatorial, problems especially in areas of planning and scheduling". CLP included the three components, i.e., variables $V \leftarrow (v_1, v_2, v_3, \ldots, v_n)$, their related domain $D \leftarrow (d_1, d_2, d_3, \ldots, d_n)$ and set of designing constraints $C \leftarrow (c_1, c_2, c_3, \ldots, c_n)$ (Russell and Norvig, 200).

2 Literature review

The scheduling problem of different kinds has received tremendous attention of research community for plenteous time due to its certain importance for academia and NP-hard complexity. "Current research direction in scheduling problems is inclined to raise the level of generality by state-of-art techniques in order to address a broad range of problem instances" (Ahmed et al., 2011b). Two types of constraints are usually imposed to the task (hard-compulsory and soft-non-compulsory) (Mauritsius et al., 2017) and in planning and scheduling problem for solving the constraints by heuristic approach, the fitness or evolution function is very significant. An evaluation function (Adriaen et al., 2003) determines the degree of constraints violations and their relevant cost parameters. Simply, a permissible objective function is keen to acquire non-violated outcome preferably, however it is not likely in most of the timetabling problems and at the end it has to compromise on optimal or feasible outcome. In addition, Ghaemi et al. (2007) sketches an evaluation function that reads an group of constraints, each sum up by constant plenty followed by Boolean results. Though, occasionally evaluation function is supposed to perform a bit more when it has to maximise the resources. Choosing heuristic to execute from a group using predefined criteria. "The fitness of each heuristic combination is calculated as a function of the hard constraint and soft constraint costs of the timetable constructed using the combination" (Pillay, 2012). Moreover, Cowling explicated it, as handling technique "The choice of which lower-level heuristic method should be applied at any given time, depending on the characteristics of the region of the solution space currently under exploration". Fang et al. (1993, 1994) introduced the term `evolving heuristic choice'. They anticipated increasing the performance quality of genetic algorithms (GA). The method was implemented on benchmark job-shop scheduling and well-performed on numerous instances. A genetic algorithm-based hyper-heuristic by Cowling et al. (2002) set forth, which they named `HyperGA'. In their study, chromosome is denoted by a sequence of local search heuristics for geologically distributed training staff and curricula scheduling. Group of heuristics in shape of chromosomes were run over state space search and the improved solutions proceed as input into next generation. Whole genetic generation got evaluated according to the inclusive improvement obtained. However, parameters depend upon variance of CPU time. Largely, improvement in overall generation fitness is used to tune up the probability of mutation. The research study was implemented by four versions of the hyper-GA along with several simple heuristics. The technique examined over five test-data. E. Burke et al. also stated efficiency of using graph-based hyper-heuristics for addressing examination scheduling problem (Burke et al., 2005). In recent time various techniques and different dimensions of scheduling problems are addressed and discussed. SAT algorithm (Prachapipat and Khancome, 2018) is applied to meet the requirement under the limited of all related resources and factors.

3 Problem description and design

Usually, any sort of scheduling belongs to combinatorial problem, which includes numerous variables and constraints over resources, for example, teacher (resource person), room (venue), subject (course), time point (timeslot) and enrolled student group, so the timetable ← [I, T, C, S, G], where timeslot is a juncture point of time and any of other resource (site or group). The university curricula scheduling can be planned by describing and handling constraints (conditions), variables, their domains (range of values) and scores (violation penalties and reward for evaluation function). The optimal solution is to allocate the suitable venue and at most exploitation of all resources on particular timespan for the enrolled group of students.

1 Time slot ~ interconnected point or place, held between resources and time (session and day) to assign the string.

2 Period ~ length of time for executing multiple events in a row.

3 Working day ~ a day consist over multiple sessions.

4 Lecture ~ an activity exclusively organised on defined time span.

5 String ~ part of data includes over group, instructor and site.

6 Resource ~ class room, laboratory equipment, projector or any other desirable equipment to event.

7 Group ~ enrolled number of students for a specific course.

8 Teacher: a lecturer conducts the academic events.

9 Course ~ subject (topics or lectures) offered to students; each course is made up of a fixed number of lectures.

10 Curriculum ~ a classified group of interrelated courses encompassing remedial (corrective), elective, mandatory (compulsory) and prerequisites courses. Students may be registered in a number of elective and mandatory courses to shape up their curricula.

Curricula scheduling may be identified and characterised by constraints (procedures), variables (data-strings), their domains (timeslots) and fitness value (accumulated sum of penalties and rewards) (Ahmed et al., 2011a; Khan et al., 2021a, 2021d). With two groups of constraints schema, where hard constraints keep up their highest priority, the soft constraints play decisive role of quality outcome and research contribution. Table 1 illustrates all the well-known hard and soft constraints briefly.

Table 2, on the other hand, describes the details of three low level heuristics, first column shows their IDs and then their procedural name, scope of operational range, functionality and of course their outcome attitude, respectively.

Table 1 Set of hard and soft constraints

Var.	*Type*	*Constraint label*	*Description*
HC1	Hard constraints	Events-redundancy	Each group of students should not be arranged for more than one events at the same time.
HC2		Conflicts	It counts violation if an event iterates on two or more locations.
HC3		Room occupancy	Only one event can be assigned to a single venue.
HC4		Availability	Lecture must be avoided to schedule if teacher has shown unavailability.
HC5		Room suitability	Venue must not be allotted if it is not equipped with accordance of course contents.
SC1	Soft constraints	Room capacity	Each student more than seating arrangement in class may be counted violation.
SC2		Min working days	Some courses demand split up over various sessions and days, if not counted violation.
SC3		Isolated lectures	No, a single classes event may be fixed throughout the day
SC4		Windows	Gaps between consultative events for students
SC5		Room stability	Keeping same room throughout the day for single group.
SC6		Student min max load	Students prescribed number of lectures for day.
SC7		Travel distance	Moving from one venue (building) to other may be avoided
SC8		Double lectures	Some course demands bigger length of class-event
SC9		Teaching max load	Teacher may deliver quality lecture if daily lecture assignments are doable.

Source: Ahmed et al. (2011a)

Table 2 Set of local bespoke heuristics

No.	*ID*	*Heuristics name*	*Scope*	*Function*	*Interaction*
1	LBH_1	Move-with-Workday-Constraint-Improvements	Session	Move	Incremental
2	LBH_2	Move-Random-Workday-Improvements	Workday	Move	Incremental
3	LBH_3	Trade-off-in-Column	Session	Tradeoff	Incremental

Source: Ahmed et al. (2011a) and Khan et al. (2021b, 2021c).

4 Algorithm – low level heuristics 1 (Move-Slot-with-Day-Improvement)

This low-level heuristic named [MoveWithDayConstantImprovement] is exemplified in Figure 1, the Procedure LLH_1 returns with decrease in overall penalty cost of soft constraints. Alternatively, it executes the rollback process if earlier state of solution is got resorted or remains unchanged. The LLH1 operates exactly over workday constraints.

Low level heuristics (LLH$_1$): MoveWithDayConstImprovment

SUB-PROCEDURE MoveWithDayConstImprovment():

1 Day ← ConstrintsType(‘**Period**’)
2 **CASE** Day:
  a DaySum ← SumPenDay(Day)
  b EmpIndexList ← EmpHCFreeDayIndexs(Day)
3 **CASE NOT** len(EmpIndexList):
4   **RETURN**
5 **ELSE:**
6   **FOR** Day2 **IN** EmpIndexList:
    a MoveFrom(Day1, Day2)
    b NowDaySum ← SumPenDay(Day)

Figure 1 Move with day constraint improvement (see online version for colours)

5 Low level heuristics 2 (Move-in-Random-Day-Improvement)

Figure 2 and related procedure [MoveRandDayImprovement] reveals the task-oriented low-level heuristic that shapes reduction in any workday selected randomly. The LLH$_2$ functions operate within the limit of sub-search space. The evaluation function gets triggered before and after executing of LLH$_2$.

Low level heuristics (LLH$_2$): MoveRandDayImprovement

SUB-PROCEDURE MoveRandDayImprovment(self):

1 Index1 ← ConstrintsType(‘Day’)

2 **CASE** Index1:

  a DaySum ← SumPenDay(Day)

3 **CASE** len(EmpIndexList)

  a **FOR** Index2 **IN** EmpIndexList:

    1 ShCasetFrom(Index1, Index2)

    2 Evaluate (Day, Index1, Index2)

4 **CASE** NowDaySum IsGreaterThan ← DaySum:

  a Rollback(Index2, Index1)

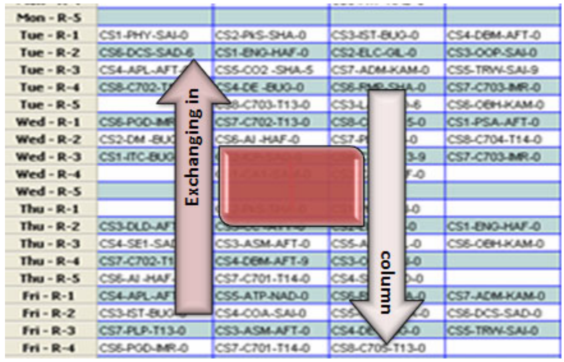
  b Evaluate(Day)

Figure 2 Improvements in any random day (see online version for colours)

6 Low level heuristics 3 (Tradeoff-in-Column)

LLH$_3$ procedure [Tradeoff-in-Column] explains a short-ranged heuristic search. The LLH$_3$ may work inside single column. The scope of LLH is planned for few prominent columns-based soft constraints such as the ‘room constraint’ is very noticeable objective of this technique. The LLH$_3$ is incremental technique, in case of better move the results are likely to be accepted otherwise the earlier status gets returned.

Figure 3 Exchanging between in single column (see online version for colours)



Low level heuristics 3: SwapInColumn

SUB-PROCEDURE SwapInColumn(Index1, 'Period'):

1 **FOR** k **IN** range (1, Rooms):
   a **CASE** Scheduler Events Container (SEC) [k] IS None:
      Index2 ← k
   b **ELSECASE** Scheduler Events Container (SEC) [k]['Penalty'] != 0:
      Index2 ← k
   c **ELSECASE** Scheduler Events Container (SEC) [k]['PenType'] = = 'P':
      Index2 ← (Index1[0], Index1[1], k)
   d **ELSE:** Index2 = None
2 **CASE** Index1 **AND** Index2:
   a SwapSlots(Index1,Index2)
   b ResetFitness(Day)

Figure 3, the algorithm establishes the shifting penalised data-slot to neighbouring session. It needs the unoccupied space in adjacent column. The technique detects mutual sides of the periods and passage the slot on suitable place.

7 Constraint satisfaction model

The research study inspects a min-conflicts algorithm established over three incremental low-level heuristics/operators for university scheduling problems (USTP). Min-conflicts algorithm belongs to CLP domain. Each low-level heuristic gradually evolves the partial solutions. The reason of operating min-conflicts algorithm (MA) over various operators is because of its maturity and credibility to produce promising results, as shown in Figure 2.

8 Min-conflicts model

A min-conflicts model is concisely illustrated in Algorithm 1. The framework includes the R procedure is random procedure which may accept or reject solution if no any noticeable improvement or somehow decline. A higher-level heuristic (min-conflicts) operate over a group of low level perturbative (incremental by alteration) heuristics. This practical model is embracing the whole idea. Initially, a tentative solution (S) produced with random initialisation than process move ahead on to applying all three low heuristics (LBH_i), followed by Evaluation Function Therefore, new candidate solution is shaped as (S_{new}). After each iteration of Evaluation Function (f (S_p)), decision point comes, If it is accepted, the new candidate solution (S_{new}) swaps the previous one (S_p), otherwise, rejection process goes under R-Criteria (set S: $\leftarrow S_p$), for giving a fair chance to slightly deteriorated candidate solution.

Algorithm 1: Min-Conflicts algorithm

1 Random Initialisation or Initial Solution S
2 set S^p: $\leftarrow$ S
3 **Do Loop**
   a Apply **Move WithDayConstImprovement**
   b Call Pro Min-Conflict
   c Apply **MoveRandDayImprovement**
   d Call Pro Min-Conflict
   e Apply **SwapInColumn**
   f Call Pro Min-Conflict
4 **While** (Termination Criteria)
5 **Procedure Min-Conflicts**
   a Apply *Evaluation function* f(S^p)
   b Produce Tentative solution S^{new}
      1 **If** S^{new} **Than** set S: $\leftarrow S^{new}$
      2 **ElseIf** f (S^{new}) < f(S^p) Then **R** (set S: $\leftarrow S^p$)
6 **End-Procedure**
7 **return** Solution (S)

9 Results

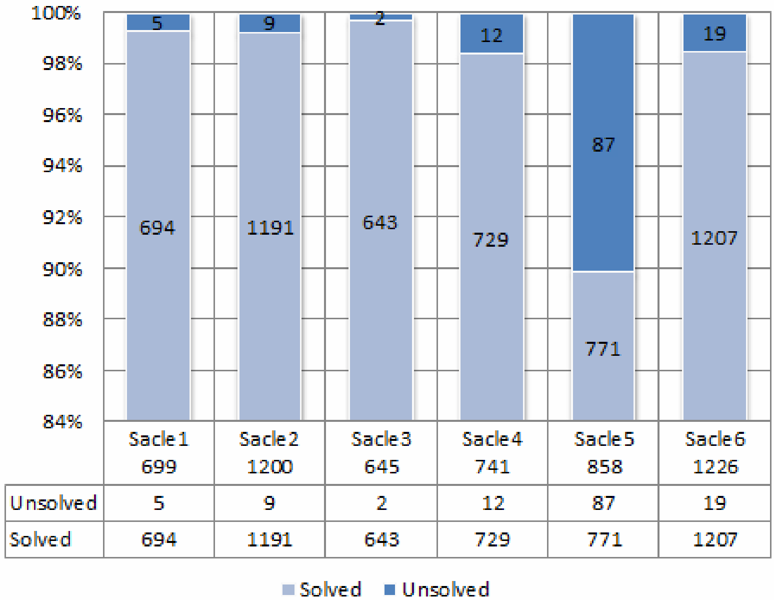
The investigational outcome authenticates the precise research path positively. The constraints programming approach is inspected over benchmark curricula datasets. The results revealed the inclusiveness of adapted computing approach. Table 1 and Figure 4, displays the dataset which is categorised over six (6) different and increasing complexity scales. In this research study, the solution for a single instance is depicted due to shortage of space, which however exemplifies and represents the capability, efficiency and effectiveness on the behalf of any data. Scale-1 comprises four (4) hard (compulsory) constraints (HC_1, HC_2, HC_3 and HC_4) and three (3) soft (discretionary) constraints in row (SC_1, SC_2 and SC_3) and number of constraints gradually increase with each complexity scale. HC_4 is perceived nullified (eliminated) at the starting and keep same status during the entire course of computation.

Table 3 Benchmark dataset

	Lectures	*Conflicts*	*R occupancy*	*Availability*	*R suitability*	*R capacity*	*MW days*	*Isolated*	*Windows*	*R stability*	*St min max load*	*Travel distance*	*Double lectures*	*Teaching max load*	*Total*
	HC1	*HC2*	*HC3*	*HC4*	*HC5*	*SC1*	*SC2*	*SC3*	*SC4*	*SC5*	*SC6*	*SC7*	*SC8*	*SC9*	
Scale1	52	156	0	138		121	87	145							699
Unsettled	0	0	0	0		1	1	3							5
Settled	52	156	0	138		120	86	142							694
Per. Sol.	100.00	100.00	0.00	100.00		99.17	98.85	97.93							
Scale2	52	156	0	138		121	87	145		501					1,200
Unsettled	0	0	0	0		1	1	6		1					9
Settled	52	156	0	138		120	86	139		500					1,191
Per. Sol.	100.00	100.00	0.00	100.00		99.17	98.85	95.86		99.80					
Scale3	52	156	0	138	48	121			43		87				645
Unsettled	0	0	0	0	0	1			1		0				2
Settled	52	156	0	138	48	120			42		87				643
Per. Sol.	100.00	100.00	0.00	100.00	100.00	99.17			97.67		100.00				
Scale4	52	156	0	138	48	121	87		43		87		9		741
Unsettled	0	0	0	0	0	0	4		4		4		0		12
Settled	52	156	0	138	48	121	83		39		83		9		729
Per. Sol.	100.00	100.00	0.00	100.00	100.00	100.00	95.40		90.70		95.40		100.00		
Scale5	52	156	0	138		121	87	145	43		87	29			858
Unsettled	0	0	0	0		0	14	15	17		21	20			87
Settled	52	156	0	138		121	73	130	26		66	9			771
Per. Sol.	100.00	100.00	0.00	100.00		100.00	83.91	89.66	60.47		75.86	31.03			
Scale6	52	156	0	138			87	145	43	501	87			17	1,226
Unsettled	0	0	0	0			4	0	5	1	7			2	19
Settled	52	156	0	138			83	145	38	500	80			15	1,207
Per. Sol.	100.00	100.00	0.00	100.00			95.40	100.00	88.37	99.80	91.95			88.24	
Settled	67	134	0	182			57	116	21	425	59			12	1, 073
Per. Sol.	100.00	100.00	0.00	100.00			87.69	96.67	65.63	98.38	90.77			92.31	

Source: Ahmed et al. (2011a)

Figure 4 Complexity scale 1–6 (see online version for colours)



10 Conclusions

The curricula or exam scheduling belongs to combinatorial optimisation of academic resources distribution and management, in which, hard constraints satisfaction is very basic requirement of solution whilst utmost soft constraints solution steep up the overall performance. The arduous and apparently tedious problem invites a broad range of algorithmic applicability and stat of the art techniques including machine learning, neural network and hyper-heuristics. Obviously, a well-finished scheduling may greatly be helpful to execute an educational session in an academia. This article tackle the problem with very effective and relevant techniques of constraints programming approach and the satisfactory outcome shows the potential and significant of 1research work. In future, the prototype may be extended by machine learning or neural net in order improve quality by keeping academia practices and interaction with problem.

References

Adriaen, M., De Causmaecker, P. and Berghe, G.V. (2003) 'Decentralised course timetabling in a large hierarchical organisation', *Proceedings of 1st Multidisciplinary International Conference on Scheduling: Practice and Theory, MISTA 2003*, Nottingham, pp.60–66.

Ahmed, A., Shaikh, A.W., Ali, M. and Bukhari, A.H.S. (2011a) 'Hyper-GA for solving benchmark scheduling problems', *Australian Journal of Basic and Applied Sciences*, June, Vol. 5, No. 6, pp.1657–1667, Research article.

Ahmed, A.A., Bukhari, H.S. and Ismaili, I.A. (2011b) 'A comparative study on three hyper-heuristic approaches for solving benchmark scheduling problems', *Sindh University Research Journal (Science Series)*, December, Vol. 43, No. 2, pp.161–168.

Barták, R. (1999a) 'Constraint programming: in pursuit of the holy grail', *Proceedings of the Week of Doctoral Students (WDS99)*, pp.555–564, MatFyzPress.

Barták, R. (1999b) 'Dynamic constraint models for planning and scheduling problems', *New Trends in Constraints*, LNAI 1865, pp.237–255, Springer.

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

Cowling, P., Kendall, G. and Han, L. (2002) 'An investigation of a hyperheuristic genetic algorithm applied to a trainer scheduling problem', *Proceedings of the Congress on Evolutionary Computation 2002*, pp.1185–1190.

Fang, H-L., Ross, P. and Corne, F. (1993) 'A promising genetic algorithm approach to job shop scheduling, rescheduling and open-shop scheduling problems', *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, pp.375–382.

Fang, H-L., Ross, P. and Corne, F. (1994) 'A promising hybrid ga/heuristic approach for open-shop scheduling problems', *Proceedings of ECAI 94: 11th European Conference on Articial Intelligence*, pp.590–594, John Wiley and Sons Ltd.

Ghaemi, S., Vakili, M.T. and Aghagolzadeh, A. (2007) 'Using a genetic algorithm optimizer tool to solve University timetable scheduling problem', *ISSPA 07*, IEEE.

Khan, A.A., Laghari, A.A., Liu, D-S., Shaikh, Ahmed, A., Ma, D-A., Wang, C-Y. and Wagan, A.A. (2021a) 'EPS-Ledger: blockchain hyperledger sawtooth-enabled distributed power systems chain of operation and control node privacy and security', *Electronics*, Vol. 10, No. 19, p.2395.

Khan, A.A., Laghari, A.A., Shaikh, A.A., Bourouis, S., Mamlouk, A.M. and Alshazly, H. (2021b) 'Educational blockchain: a secure degree attestation and verification traceability architecture for higher education commission', *Applied Sciences*, Vol. 11, No. 22, p.10917.

Khan, A.A., Shaikh, Z.A., Baitenova, L., Mutaliyeva, L., Moiseev, N., Mikhaylov, A., Laghari, A.A., Idris, S.A. and Alshazly, H. (2021c) 'QoS-ledger: smart contracts and metaheuristic for secure quality-of-service and cost-efficient scheduling of medical-data processing', *Electronics*, Vol. 10, No. 24, p.3083.

Khan, A.A., Shaikh, Z.A., Laghari, A.A., Bourouis, S., Wagan, A.A. and Ali, G.A.A.A. (2021d) 'Blockchain-aware distributed dynamic monitoring: a smart contract for fog-based drone management in land surface changes', *Atmosphere*, Vol. 12, No. 11, p.1525.

Mauritsius, T.A.N.F., Harisno and John, P. (2017) 'Novel local searches for finding feasible solutions in educational timetabling problem', *5th International Conference on Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME)*, pp.270–275.

Müller, T. (2002) 'Some novel approaches to lecture timetabling', *Proc. of the 4th Workshop on Constraints Programming for Decision and Control*, Gliwice, Poland, 2002, pp. 31–37.

Pillay, N. (2012) 'Evolving hyper-heuristics for the uncapacitated examination timetabling problem', *Journal of the Operational Research Society*, Vol. 63, pp.47–58.

Prachapipat, A.L.P. and Khancome, C. (2018) 'New examination timetabling algorithm using the superstar assignment technique', *MATTER: International Journal of Science and Technology*, Vol. 3, No. 3, pp.253–270.

Rudová, M.V.H. (2005) 'Multi-criteria soft constraints in timetabling', *Proc. MISTA*, pp.11–15.

Russell, S. and Norvig, P. (2003) *Artificial Intelligence A Modern Approach*, Prentice Hall.