Artificial Immune Algorithms for University Timetabling

Muhammad Rozi Malim[1], Ahamad Tajudin Khader[2], and Adli Mustafa[3]

[1] Faculty of Info. Technology & Quantitative Sciences, UiTM, 40450 Shah Alam, Malaysia
rozi@tmsk.uitm.edu.my
[2] School of Computer Science, USM, 11800 Minden, Penang, Malaysia
tajudin@cs.usm.my
[3] School of Mathematical Science, USM, 11800 Minden, Penang, Malaysia
adli@cs.usm.my

Abstract. The university timetabling, examination and course, are known to be highly constrained optimization problems. Metaheuristic approaches, and their hybrids, have successfully been applied to solve the problems. This paper presents three artificial immune algorithms, the algorithms inspired by the immune system, for university timetabling; clonal selection, immune network and negative selection. The main objective is to show that the algorithms may be tailored for educational timetabling. The experimental results have shown that all algorithms have effectively produced good quality timetables. The clonal selection and negative selection are more effective than immune network in producing good quality examination timetables; while for course timetabling, the immune network and negative selection are more effective than clonal selection. A comparison with other published results has significantly shown the effectiveness of these algorithms. The main operators in artificial immune algorithms are cloning and mutation. For future work, these algorithms will be improved by considering other cloning and mutation operators.

Keywords: Examination Timetabling; Course Timetabling; Artificial Immune Algorithms.

1 Introduction

The constructions of *examination* and *course* (lecture) timetables are common problems for all institutions of higher education. Usually it involves modifying the previous semester's timetable so it will work for the new semester. The examination and course timetabling are known to be highly constrained combinatorial optimization problems. Metaheuristic approaches such as simulated annealing (SA), tabu search (TS), evolutionary algorithms (EA), and their hybrids, have successfully been applied to solve the problems.

Artificial immune system (AIS), a new branch of Artificial Intelligence [4], is a new intelligent problem-solving technique that being used in optimization and scheduling. The AIS algorithms are more efficient than the classical heuristic scheduling algorithms such as SA, TS, and genetic algorithm (GA) [12]. AISs have

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 234–245. ISBN 80-210-3726-1.

been more successful than GA and other methods in applications of pattern recognition, computer and network security, and dynamic tasks scheduling due to the applicability features of natural immune systems. Furthermore, the solutions produced by the AIS are observed to be *robust* than solutions produced by a GA [13]. There are *three* algorithms that have widely been applied in AIS; *clonal selection algorithm* (CSA), *immune network algorithm* (INA), and *negative selection algorithm* (NSA).

This paper presents three artificial immune algorithms for examination and course timetabling. The main objective is to show that the algorithms may be tailored for educational timetabling, and also to compare the effectiveness of the three algorithms on examination and course datasets. Twelve *Carter* datasets (examination) and three *Schaerf* datasets (course) have been used in the implementation. The experimental results have significantly shown the effectiveness of the three algorithms; all algorithms have effectively produced good quality (low fitness) examination and course timetables in most of the datasets. The CSA and NSA are more effective than INA on examination datasets, and on course datasets, the INA and NSA are more effective than CSA. However, based on CPU time, INA runs faster than CSA and NSA on examination datasets, and CSA runs faster than INA and NSA on course datasets. A comparison with other published results have significantly shown that the three algorithms are capable of producing good quality examination and course timetables as good as other optimization algorithms.

The main operators in artificial immune algorithms are *cloning* and *mutation*. For future work, these algorithms will be improved by considering other cloning and mutation operators so that the fitness values may be further minimized. And also, especially for course datasets, a further study is required to solve timetabling problems with 100% occupancy by considering dummy timeslots and/or rooms.

2 University Timetabling Problems

University timetabling problems can be divided into *two* main categories: exam and course. The main difference is that in course timetabling there cannot be more than one course per room, but in exam timetabling there can be more than one exam.

Examination timetabling problem (ETP) is a specific case of the more general timetabling problem. The examination timetabling regards the scheduling for the exams of a set of university courses, avoiding overlap of exams of courses having common students, and spreading the exams for the students as much as possible [7]. Given is a set of exams, a set of timeslots, a set of students, and a set of student enrollments to exams, the problem is to assign exams to timeslots subject to a variety of *hard* and *soft* constraints. The ETP can be seen as consisting of *two* subproblems: (1) assigning exams to timeslots, and (2) assigning exams to rooms. For real-life situations, these two subproblems can be solved separately.

Course timetabling problem (CTP) is another specific case of the more general timetabling problem. At its simplest, course timetabling is the problem of scheduling a set of events (lectures, tutorials or labs) to a set of classrooms in a set of timeslots within a week, and taught by a set of teachers, such that no student or teacher is expected to be in more than one room at the same time and that there is enough space in each classroom for the number of students expected to be there. The CTP can be

seen as consisting of *three* subproblems; 'course-teacher assignment', 'event-timeslot assignment', and 'event-room assignment'. In 'course-teacher assignment', the teachers are scheduled to a number of events in all courses; in 'event-timeslot assignment', all events for all courses are scheduled into a fixed number of timeslots; and in 'event-room assignment', these events are assigned to a fixed number of rooms. Hence, an *assignment* is an ordered 4-tuple (*a, b, c, d*), and has the straightforward general interpretation: 'event *a* starts at timeslot *b* in room *c*, and is taught by teacher *d*'. For some institutions, the allocation of courses to teachers is carried out manually, and the allocation of events in a given timeslot to rooms is a secondary problem. These three subproblems can be solved separately.

Hard constraints must be satisfied in order to produce a *feasible* timetable. Any timetable fails to satisfy these constraints is deemed to be *infeasible*. Soft constraints are generally more numerous and varied, and far more dependent on the needs of the individual problem than the more obvious hard constraints. The violation of soft constraints should be minimized; it is the soft constraints which effectively define how good a given feasible solution is so that different solutions can be compared and improved via an objective (*fitness*) function.

3 Artificial Immune System and Artificial Immune Algorithms

The 'artificial immune system' is an approach which used the natural immune system as a metaphor for solving computational problems, *not* modeling the immune system [21]. The main application domains of AIS are anomaly detection [16], pattern recognition [23], computer security [14], fault tolerance [1], dynamic environments [18], robotics [19], data mining [20], optimization [22], and scheduling [12].

The 'immune system' (IS) can be considered to be a remarkably efficient and powerful information processing system which operates in a highly parallel and distributed manner [11]. It contains a number of features which potentially can be adapted in computer systems; recognition, feature extraction, diversity, learning, memory, distributed detection, self-regulation, threshold mechanism, co-stimulation, dynamic protection, and probabilistic detection. It is unnecessary to replicate *all* of these aspects of the IS in a computer model, rather they should be used as general guidelines in designing a system.

There are a number of different algorithms that can be applied to many domains, from data analysis to autonomous navigation [5]. These immune algorithms were inspired by works on theoretical immunology and several processes that occur within the IS. The AISs lead to the development of different techniques, each one mapping a different mechanism of the system. For examples, the *Artificial Immune Networks* as proposed by Farmer et al. [9], the *Clonal Selection Algorithm* proposed by de Castro and Von Zuben [6], and the *Negative Selection Algorithm* introduced by Forrest et al. [10]. Immune network models are suitable to deal with dynamic environments and optimization problems, algorithms based upon the clonal selection principle are adequate to solve optimization and scheduling problems, and the negative selection strategies are successfully applied to anomaly detection.

3.1 Clonal Selection Algorithms for University Timetabling

The clonal selection algorithm (CSA) is inspired by the immunological processes of *clonal selection* and *affinity maturation*. When an antigen is detected, those antibodies that best recognize this antigen will proliferate by cloning. This process is called *clonal selection principle* [6]. The clonal selection principle is used to explain how the IS 'fights' against an antigen. When a bacterium invades our organism, it starts multiplying and damaging our cells. One form the IS found to cope with this replicating antigen was by replicating the immune cells successful in recognizing and fighting against this disease-causing element. Those cells reproduce themselves asexually in a way proportional to their degree of recognition: the better the antigenic recognition, the higher the number of clones (offspring) generated. During the process of cell division (reproduction), individual cells suffer a mutation that allows them to become more adapted to the antigen recognized: the higher the affinity of the parent cell, the lower the mutation they suffer. Figure 1 shows the CSA for exam or course.

1. Initialization:	*initialize a population of antibodies (feasible timetables)*
	for each antibody (timetable)
	randomly select event (exam/course) one by one
	assign event to random selected timeslots and rooms (satisfying hard constraints)
	if no identical antibodies (duplicate timetables)
	add antibody (timetable) to the population
	else eliminate antibody
2. Population loop:	*for each generation of antibodies (feasible timetables)*
	for each antibody do
2.1 Affinity evaluation:	*determine the affinity of antibody via an affinity function (affinity = 1/fitness)*
2.2 Selection:	*calculate the selection probability (rate of cloning) using affinity*
	(selection probability = affinity/total affinities)
	randomly select an antibody (timetable) based on selection probability
	(using roulette wheel selection method)
2.3 Genetic variation:	*Cloning: clone copies of the selected antibody*
	(number of clones = population size × cumulative selection probability)
	Mutation: for each generated clone, do (mutation rate = 1 - selection probability)
	if a random probability <= mutation rate, mutation = failure
	while mutation = failure, select an event at random
	reassign event to random timeslot and room (satisfying all hard constraints)
	if all hard constraints are satisfied and no duplicate timetables
	determine the affinity of the new clone
	if the affinity (new clone) >= the affinity (original clone)
	mutation = success
2.4 Population update:	*if the affinity (new clone) > minimum affinity (population), say antibody X*
	then X = new clone
3. Cycle:	*repeat (Step 2) until stopping criteria are met.*

Fig. 1. Clonal Selection Algorithm for University Timetabling

The main operators in CSA are *selection*, *cloning*, and *mutation*. A timetable (high affinity) is randomly selected for cloning using Roulette Wheel selection method and, on average, a number of clones that equal to half of the population size are generated. Almost all clones will be mutated to produce new feasible timetables for the next generation since '1- selection probability' would give a high mutation rate for each clone. But only new timetables with high affinity will be selected to replace the low affinity timetables in the current population. The process (selection, cloning and mutation) will be repeated until the stopping criteria are met.

3.2 Immune Network Algorithms for University Timetabling

The immune network algorithm (INA) is based on *Jerne's idiotypic network theory* [15]. According to this theory, immune cells have portions of their receptor molecules that can be recognized by other immune cells in a way similar to the recognition of an invading antigen. This results in a network of recognition between immune cells. When an immune cell recognizes an antigen or another immune cell, it is stimulated. On the other hand, when an immune cell is recognized by another immune cell, it is suppressed. The sum of the stimulation and suppression received by the network cells, plus the stimulation by the recognition of an antigen corresponds to the stimulation level *S* of a cell. Figure 2 shows the INA for examination or course timetabling.

```
1. Initialization:        initialize a network (population) of immune cells (feasible timetables)
                          for each immune cell (timetable)
                             randomly select event one by one
                                assign event to random timeslot and room (satisfying all hard constraints)
                             if no identical immune cells (duplicate timetables)
                                add immune cell to the population
                             else eliminate immune cell
2. Population loop:       for each network (generation/population) of immune cells (feasible timetables)
2.1 Network interactions and Stimulation:
                             for each immune cell
                                determine the fitness of immune cell via a fitness function
                                calculate the stimulation level of immune cell (stimulation level = 1/fitness)
                             determine the total stimulation of the network (population)
                             calculate the stimulation probability for each immune cell
                             (stimulation probability = stimulation/total stimulation)
2.2 Metadynamics (Antigens and Genetic variations):
                             for each immune cell
                                cloning – generate a number of clones
                                (number of clones = population size ×stimulation probability)
                                for each clone
                                   determine the mutation rate (mutation rate = 1 – stimulation probability)
                                   generate a random probability
                                   if a random probability <= mutation rate
                                      mutation = failure
                                         while mutation = failure
                                            select an event at random
                                         reassign event to random timeslot and best room
                                         if all hard constraints are satisfied and no duplicate timetables
                                            mutation = success
                                            determine the fitness of the new clone
                                            if the fitness (new clone) > the fitness (original clone)
                                               mutation = failure, and reset the reassignment
                                   else (no mutation) assign a zero stimulation (large fitness) to immune cell
2.3 Network dynamics (immune cells and antigens interactions, and population update):
                             gather all immune cells (current population and cloned timetables)
                             sort immune cells according to stimulation level (descending order)
                             select the best (high stimulation) immune cells (feasible timetables)
                             (number of selected immune cells = network or population size)
                             update the network (population) of immune cells with the selected cells
3.  Cycle: repeat Step 2 until a given convergence or stopping criterion is met.
```

Fig. 2. Immune Network Algorithm for University Timetabling

The main operators in INA are *cloning* and *mutation*. All timetables are selected for cloning and, on average, one clone is generated for each timetable. Almost all clones will be mutated to produce new feasible timetables since '1- stimulation probability' would give a high mutation rate for each clone. All feasible timetables, current population and mutated clones, are gathered, but only the timetables with high stimulation will be selected to form a new population for the next generation. The process (cloning and mutation) will be repeated until the stopping criteria are met.

3.3 Negative Selection Algorithms for University Timetabling

The negative selection algorithm (NSA) is the most widely used techniques in AISs. The NSA is based on the principles of self-nonself discrimination [10]. The algorithm was inspired by the thymic negative selection process that intrinsic to natural immune systems, consisting of screening and deleting self-reactive T-cells, i.e. those T-cells that recognize self cells. Figure 3 shows the NSA for examination or course timetabling.

```
1.  Initialization:    initialize a population of candidate detectors (initial feasible timetables)
                       for each candidate detector (timetable)
                          randomly select event one by one
                             assign event to random timeslot and room (satisfying all hard constraints)
                          if no identical candidate detectors (duplicate timetables)
                             add candidate detector to the initial population
                          else eliminate candidate detector
2.  Population loop:   for each generation (population) of detectors (feasible timetables)
2.1 Censoring:            for each detector (timetable) in the current population
                             determine the fitness value via a fitness function (soft constraints)
                          determine the average fitness for the current population
                          for each detector
                             if the fitness >= average, eliminate the detector
                             if all fitness values are equal, eliminate only the second half of the detectors
2.2 Monitoring:        while the number of detectors (timetables) < population size
                          randomly select a detector according to fitness using roulette wheel
                          clone the detector, mutation = failure
                          while mutation = failure, randomly select an event
                             reassign event to random timeslot and best room
                             if all hard constraints are satisfied and no identical detectors
                                mutation = success
                                determine the fitness of new clone
                                if the fitness of the new clone > average fitness of the population
                                   mutation = failure
                                   eliminate the new clone, and reset the reassignment
                                else add the new clone to the new population
3.  Cycle: repeat population loop until a given convergence criterion is met.
```

Fig. 3. Negative Selection Algorithm for University Timetabling

The main operators in NSA are *negative deletion* (censoring), *cloning* and *mutation*. The timetables (current population) with fitness greater than or equal to average fitness are eliminated or deleted from the current population. A timetable is randomly selected from the remaining timetables for cloning and mutation using Roulette Wheel selection method (based on fitness). All clones will be mutated to produce new feasible timetables. For each new (mutated) timetable, if the fitness is less than or equal to average, the timetable will be added to the new population for the next generation; otherwise, it will be deleted. The monitoring process (cloning and mutation) will be repeated until the number of feasible timetables in the new population is equal to population size. Finally, the optimization process (censoring and monitoring) will be repeated until the stopping criteria are met.

4 Benchmark Datasets

The benchmark datasets (*Carter* and *Schaerf*) used in the implementation of the *three* immune algorithms are available from ftp://ftp.mie.utoronto.ca/pub/carter/testprob/ and http://www.diegm.uniud.it/schaerf/projects/coursett/, respectively. These datasets provide reasonable benchmark problems for comparison of the three different artificial immune algorithms. The datasets are shown in Table 1 and Table 2.

Table 1. Examination Datasets and Characteristics (Carter Datasets)

Code	University	No. of Exams	No. of Students	No. of Enrollments	Timeslot Capacity
car-f-92	Carleton University 1992	543	18419	55522	2000
car-s-91	Carleton University 1991	682	16925	56877	1550
ear-f-83	Earl Haig Collegiate 1983	190	1125	8109	350
hec-s-92	Ecole des Hautes Etudes Comm 92	81	2823	10632	650
kfu-s-93	King Fahd University 1993	461	5349	25113	1955
lse-f-91	London Sch. of Econ. 1991	381	2726	10918	635
rye-s-93	Ryerson University 1993	486	11483	45051	2055
sta-f-83	St. Andrews High 1983	139	611	5751	465
tre-s-92	Trent University 1992	261	4360	14901	655
uta-s-92	Uni. of Toronto, Arts & Science 92	622	21266	58979	2800
ute-s-92	Uni. of Toronto, Engineering 92	184	2750	11793	1240
yor-f-83	York Mills Collegiate 1983	181	941	6034	300

Each of the datasets come in two files, one file (*course data file*) contains the list of courses and the other (*student data file*) contains a list of student-course selections. The courses and student-course selections are sorted in ascending order.

Table 2. Course Datasets and Characteristics (Schaerf Datasets)

Instance	No. of Courses	No. of Rooms (R)	No. of Timeslots (T)	Timeslots per day	Total lectures (L)	No. of Teachers	Occupancy (L/(R×T))
1	46	12	20	4	207	39	86.25%
2	52	12	20	4	223	49	92.92%
3	56	13	20	4	252	51	96.92%
4	55	10	25	5	250	51	100%

Each of the datasets comes in *five* files; *course.dat* contains the information about the courses, *periods.dat* contains the list of timeslots of the timetabling horizon, *curricula.dat* contains the information about groups of courses that share common students, *constraint.dat* contains additional constraints about timeslot unavailabilities, and *room.dat* contains information about rooms. The 'occupancy' indicates the percentage of timeslot-room required to schedule all the lectures.

However, the dataset 'Instance 4' was not considered; 100% occupancy would make the mutation process impossible, and perhaps *dummy* timeslots and rooms would solve the problem. This requires more time and further study, and will be included in the future work.

5 Comparing Artificial Immune Algorithms on Exam Datasets

The three artificial immune algorithms (CSA, INA, NSA) have been implemented on the twelve examination datasets (Carter datasets). The main objective is to compare the effectiveness of the three algorithms on examination datasets.

Three hard constraints were considered for each of the datasets: (1) no students must be assigned to two different exams at the same timeslot, (2) timeslot capacity must not be exceeded, and (3) each exam must be assigned to exactly one timeslot. The *fitness value* (soft constraint violations) is the minimum number of students having two exams in adjacent (consecutive) timeslots.

The following (Table 3) are the experimental results for examination datasets using the three artificial immune algorithms. Each algorithm was run on each dataset for *five* trials, and the maximum number of generations '500' was used as the stopping criterion. The best fitness, the average fitness and the average CPU time (in seconds) for each algorithm on each of the datasets, based on five trials, have been recorded.

Table 3. Comparing Three Artificial Immune Algorithms on Examination Datasets

Institution	No. of Exams	No. of Timeslots	Fitness Values					
			CSA		INA		NSA	
			Best	Average Fitness (CPU time)	Best	Average Fitness (CPU time)	Best	Average Fitness (CPU time)
car-f-92	543	31	**285**	466.6 (310.6s)	406	455.2 **(249.8s)**	386	**432.8** (359.4s)
car-s-91	682	40	535	569.6 (512.6s)	554	582.8 **(399.6s)**	**439**	**486.2** (484s)
ear-f-83	190	24	**17**	**48** (75.4s)	65	112.8 **(34.8s)**	74	118.8 (88s)
hec-s-92	81	19	3	11 (17.2s)	**0** **(271)**	**9.8** **(7.8s)**	5	14.4 (11.4s)
kfu-s-93	461	20	35	69.4 **(172.4s)**	16	32.6 (202.2s)	**2**	**13.6** (240.2s)
lse-f-91	381	18	45	**68.8** (132.4s)	**34**	82.6 **(120.8s)**	115	167.2 (147s)
rye-s-93	486	24	**143**	**240.2** **(233.8s)**	217	309 (247s)	180	327.6 (336.4s)
sta-f-83	139	14	0 (196)	**0** (12.2s)	0 (185)	0.4 (11.4s)	**0** **(160)**	**0** **(9.6s)**
tre-s-92	261	25	**27**	**36.8** (110s)	58	70.2 **(57.4s)**	56	79.2 (134s)
uta-s-92	622	32	436	487.6 (343.2s)	374	436 **(307.4s)**	**165**	**244.6** (387s)
ute-s-92	184	10	**0** **(352)**	**0.4** (34.8s)	0 (454)	2.6 **(26.8s)**	1	9.8 (34.4s)
yor-f-83	181	22	3	8 (62.6s)	24	33.2 **(30.4s)**	**1**	**6.6** (63.2s)

The *number of timeslots* used for all datasets were imposed according to those given in Carter's results. However, the number of timeslots for all datasets may be further reduced if necessary. Based on five trials, for the *best fitness*, both CSA and NSA have achieved the first position in *five* datasets, while INA has achieved the first position in only *two* datasets. The best fitness values for CSA have converged to '0' in *two* datasets, INA in *three* datasets, and NSA in *one* dataset. For the *average fitness*, both CSA and NSA has achieved the first position in *six* datasets, and INA in only *one* dataset. Finally, for the *average CPU time*, INA has achieved the first position in *nine* datasets, only *two* for CSA and *one* for NSA.

Hence, it may be concluded that CSA and NSA are equally effective in producing good quality (low fitness) examination timetables, and both are more effective than NSA. Based on CPU time, INA runs faster than CSA and NSA. The results from 12 different examination datasets have significantly shown the effectiveness of the three algorithms. All algorithms have effectively produced good quality examination timetables in most of the datasets.

A comparison with other published results was also conducted. This is to access the effectiveness of the three algorithms against other optimization algorithms. Only *five* datasets were considered; car-f-92, car-s-91, kfu-s-93, tre-s-92, and uta-s-92. The following are the authors and metaheuristic approaches used in the published results:

(**A**) Burke et al. [2] – Memetic Algorithm.
(**B**) Di Gaspero and Schaerf [7] – Tabu Search.
(**C**) Caramia et al. [3] – A set of heuristics: Greedy Assignment, Spreading Heuristic.
(**D**) Merlot et al. [17] – Hybrid Algorithm: Constraint Programming, Simulated Annealing, Hill-climbing.

All authors had considered the same hard and soft constraints, hence a comparison based on the fitness values (number of students having two exams in adjacent timeslots) may be carried out. The main goal is to show that the immune algorithms can produce good quality examination timetables as good as other methods. The number of timeslots used for all datasets were imposed according to the papers of the published results. The maximum number of none-improvement generations '25' was considered as the stopping criterion. Table 4 summarizes the results.

Table 4. Comparison with Other Solution Methods

Code	Timeslots (Sessions)	Timeslot Capacity	Fitness Values						
			A	B	C	D	Average Fitness		
							CSA	INA	NSA
car-f-92	31	2000	331	424	268	158	**75.7**	97.3	154.3
car-s-91	51	1550	81	88	74	31	33.0	73.7	**21.7**
kfu-s-93	20	1995	974	512	912	247	5.3	12.0	**5.0**
tre-s-92	35	655	3	4	2	**0**	7.7	13.0	8.0
uta-s-92	38	2800	772	554	680	334	24.7	81.7	**12.7**

The results of other solution methods are available on Internet from http://www.or.ms.unimelb.edu.au/timetabling/ttframe.html?ttexp3.html. Hence, based on five trials, and five datasets, the three artificial immune algorithms have effectively produced good quality examination timetables, as good as other solution methods.

6 Comparing Artificial Immune Algorithms on Course Datasets

The three artificial immune algorithms (CSA, INA, NSA) have been implemented on the three course datasets (Schaerf datasets). The main objective is to compare the effectiveness of the algorithms on course datasets.

Five hard constraints were considered for all datasets: (1) all lectures of all courses must be scheduled, (2) two distinct lectures cannot take place in the same room in the same timeslot, (3) lectures of courses of the same group must be all scheduled at different timeslots, (4) lectures of courses taught by the same teacher must be scheduled at different timeslots, and (5) lectures of some courses must not be assigned to certain timeslots. The *fitness value* (soft constraint violations) is the number of students without a seat, plus the number of courses that assigned to less than the minimum number of days multiply by 5, and plus the number of gaps between lectures of the same group on the same day multiply by 2.

The following (Table 5) are the experimental results for course datasets using the three artificial immune algorithms. Each algorithm was run on each dataset for *five* trials, and the maximum number of generations 1000 was used as the stopping criterion. The best fitness, the average fitness, and the average CPU time (in seconds) for each algorithm on each of the datasets, based on five trials, have been recorded.

Table 5. Comparing Three Artificial Immune Algorithms on Course Datasets

Instance	No. of Courses	Fitness Values									Di Gaspero & Schaerf (2003)
		CSA			INA			NSA			
		Best	Ave	Ave CPU	Best	Ave	Ave CPU	Best	Ave	Ave CPU	
1	46	284	297	1560s	265	**296**	3976s	**263**	298	1583s	200
2	52	21	46	347s	**11**	**21**	1190s	21	36	525s	13
3	56	69	98	1617s	50	**72**	5873s	**48**	74	2254s	55

The results have significantly shown the effectiveness of the three algorithms. All algorithms have effectively produced good quality course timetables with low fitness values in all datasets. For the *best fitness*, NSA has achieved the first position in *two* datasets, while INA in *one* dataset. For the *average fitness*, INA has achieved the first position in *all* datasets. Finally, for the *average CPU time*, CSA has achieved the first position in *all* datasets. It may be concluded that, based on three datasets and five trials, NSA and INA are more effective than CSA in producing good quality course timetables; however, CSA runs faster than INA and NSA.

There is only one published result available on these datasets, by Di Gaspero and Schaerf [8], as shown on the right-hand side of Table 5; available from http://www.diegm.uniud/satt/projects/EduTT/. The artificial immune algorithms have achieved the first position in two datasets (Instances 2 and 3). However, no results have been produced by artificial immune algorithms for Instance 4; 100% occupancy in Instance 4 requires dummy timeslots and/or rooms for the mutation process. This requires more effort and time and will be considered in the future work.

7 Conclusion and Future Work

This paper has presented and compared three artificial immune algorithms for the university timetabling problems; CSA, INA and NSA. The experimental results using twelve Carter datasets (examination) and three Schaerf datasets (course) have significantly shown the effectiveness of these algorithms on university timetabling datasets. All algorithms have efficiently produced good quality examination and course timetables with low fitness values in most of the datasets. For examination datasets, CSA and NSA are both more effective than INA in producing good quality timetables; while for course datasets, NSA and INA are more effective than CSA. Based on CPU time, INA runs faster than CSA and NSA on examination datasets, and CSA runs faster than INA and NSA on course datasets.

All artificial immune algorithms show great promise in the area of educational timetabling, particularly in its ability to consider, solve, and optimize the wide variety of different examination and course timetabling problems. The algorithms can handle the hard constraints and soft constraints very well. The experimental results have shown that the algorithms can successfully be applied to solve various kinds of examination and course timetabling problems. These algorithms may be accepted as new members of evolutionary algorithms for solving timetabling problems.

The most important operators in artificial immune algorithms are cloning and mutation. For future work, these algorithms will be improved by considering other operators, especially mutation, so that the fitness values may be further minimized. However, different timetabling problems may require different operators. A good cloning or mutation operator for one problem is not necessarily a good operator for other problems. For the course timetabling, the three algorithms were not designed to handle timetabling problems with 100% occupancy as in Instance 4 of Schaerf datasets. Perhaps the use of dummy timeslots and/or rooms will solve the problems. This will be the first priority in our future research.

References

1. Bradley, D.W., and Tyrrell, A.M.: Immunotronics: Hardware Fault Tolerance Inspired by the Immune System. Proceedings of the International Conference on Evolvable Systems (ICES2000). April 17-19, Edinburgh, UK (2000) 11-20.
2. Burke, E.K., Newell, J.P., and Weare, R.F.: A memetic algorithm for university exam timetabling. Proceedings of the First International Conference on Practice and Theory of Automated Timetabling (ICPTAT 1995). August 30 – Sept 1, Edinburgh, UK (1995) 496-503.
3. Caramia, M., Dell'Olmo, P., and Italiano, G.F.: New algorithms for examination timetabling. Lecture Notes in Computer Science 1982, S. Nher and D. Wagner (eds), Springer-Verlag, Berlin-Heidelberg, Germany, (2001) 230.
4. Dasgupta, D., Ji, Z., and González, F.: Artificial Immune System (AIS) Research in the Last Five Years. Proceedings of the International Conference on Evolutionary Computation (CEC2003). December 8-12, Canberra, Australia (2003).

5. de Castro, Leandro Nunes: Immune, Swarm, and Evolutionary Algorithms, Part I: Basic Models. Proceeding of the ICONIP, Workshop on Artificial Immune Systems, Vol. 3. November 18-22, Singapore (2002) 1464-1468.
6. de Castro, L.N. and Von Zuben, F.J.: The Clonal Selection Algorithm with Engineering Applications. Proceedings of The Genetic and Evolutionary Computation Conference 2000 (GECCO'OO) - Workshop Proceedings. July 8-12, Las Vegas, USA (2000) 36-37.
7. Di Gaspero, L. and Schaerf, A.: Tabu search techniques for examination timetabling. In Practice and Theory of Automated Timetabling, Third International Conference, E.K. Burke and W. Erben (eds): Lecture Notes in Computer Science 2079, Springer-Verlag, Konstanz, Germany (2001) 104-117.
8. Di Gaspero, L. and Schaerf, A.: Multi Neighborhood Local Search with application to the Course Timetabling Problem. Proceedings of the 4th International Conference on Practice and Theory of Automated Timetabling (PATAT2002). Lecture Notes in Computer Science 2740. Springer-Verlag, Berlin-Heidelberg, Germany (2003).
9. Farmer, J.D., Packard, N.H. and Perelson, A.S.: The immune system, adaptation, and machine learning. Physica, 22D (1986) 187-204.
10. Forrest, S., Perelson, A.S., Allen, L. and Cherukuri, R.: Self-nonself discrimination in a computer. Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy Los Alamitos, CA: IEEE Computer Society Press (1994) 202-212.
11. Hart, Emma: Immunology as a Metaphor for Computational Information Processing: Fact or Fiction? PhD Thesis, University of Edinburgh (2002).
12. Hart, E. and Ross, P.: An Immune System Approach to Scheduling in Changing Environments. Proceedings of The Genetic and Evolutionary Computation Conference 1999 (GECCO'99). July 13-17, Florida, USA (1999) 1559-1566.
13. Hart, E., Ross, P., and Nelson, J.: Producing Robust Schedules Via An Artificial Immune System. Proceedings of the International Conference on Electronic Commerce 1998 (ICEC'98). April 6-9, Seoul, Korea (1998).
14. Hofmeyr, S.A. and Forrest, S.: Architecture for an Artificial Immune System. Research Notes. Department of Computer Science, University of New Mexico (2003).
15. Jerne, N.K.: Towards a Network Theory of the Immune System. Ann. Immunol. (Inst. Pasteur) 125C (1974) 373-389.
16. Kim, J., Ong, A., and Overill, R.: Design of an Artificial Immune System as a Novel Anomaly Detector for Combating Financial Fraud in Retail Sector. Congress on Evolutionary Computation 2003 (CEC2003), Dec 8-12, Canberra (2003).
17. Merlot, L.T.G., Boland, N., Hughes, B.D. and Stuckey, P.J.: A Hybrid Algorithm for the Examination Timetabling Problem. Lecture Notes in Computer Science 2740, E.K. Burke and P. De Causmaecker (eds), Springer-Verlag, Belgium (2003) 207.
18. Simões, A., and Costa, E.: An Immune System-Based Genetic Algorithm to Deal with Dynamic Environments: Diversity and Memory. Proceedings of the 6th International Conference on Neural Networks & Genetic Algorithms. April 23-25, France (2003).
19. Singh, S., and Thayer, S.: Immunology Directed Methods for Distributed Robotics: A Novel, Immunity-Based Architecture for Robust Control & Coordination. Proceedings of SPIE: Mobile Robots XVI, Vol. 4573 (2001).
20. Timmis, Jonathan: Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network Theory. Dissertation, University of Wales (2000).
21. Timmis, Jonathan: An Introduction to Artificial Immune Systems. Research Notes, ES2001, Cambridge, University of Kent at Canterbury, England, UK (2001).
22. Walker, J.H. and Garrett, S.M.: Dynamic Function Optimization: Comparing the Performance of Clonal Selection and Evolution Strategies. Proceeding of Second International Conference on Artificial Immune Systems (ICARIS2003). September 1-3, Napier University, Edinburgh, UK (2003).
23. White, J.A. and Garrett, S.M.: Improved Pattern Recognition with Artificial Clonal Selection? Proceeding of Second International Conference on Artificial Immune Systems (ICARIS2003). September 1-3, Napier University, Edinburgh, UK (2003).