Learning and Cooperating Multi-Agent Scheduling Repair Using a Provenance-Centred Approach

Terence Tan[†], Tele Tan[††], Geoff West[†††], Siow Yong Low[†]
[†]Department of Electrical and Computer Engineering
Curtin University Sarawak Campus
Miri, Malaysia
{terence.tan, siowyong}@curtin.edu.my

[††]Department of Computing,
[†††]School of Spatial Sciences
Curtin University
Perth, Australia
{t.tan, g.west}@curtin.edu.au

Abstract—**The timetabling problem is to find a timetable solution by assigning time and resources to sessions that satisfy a set of constraints. Traditionally, research has focused on optimization towards a final solution but this paper focuses on minimizing disturbance impact due to changing conditions. A Multi-Agent System (MAS) is proposed in which users are represented as autonomous agents negotiating with one another to repair a timetable. From repeated negotiations, agents learn to develop a model of other agent's preferences. The MAS is simulated on a factorial experiment set up and varying the cooperation level, learning model and selection strategy.A provenance-centred approach is adopted to improve the human aspect of timetabling to allow users to derive the steps towards a solution and make changes to influence the solution.**

Keywords-schedule repair; multi-agent system; learning; cooperating; provenance

I. INTRODUCTION

Timetabling problems are shown to be NP-hard problems [1], which means it is probably impossible to find an algorithm that will generate an optimal solution for a realistically large problem.

In a survey [2] on examination timetabling techniques, ten categories of techniques were listed including graph colouring, constraint based, tabu search, simulated annealing, local search based, evolutionary algorithms, ant colony optimisation algorithms and artificial immune algorithms, multi-criteria techniques, hyper-heuristics and decomposition approaches. Although the survey was about examination timetabling the techniques described are widely used in other types of timetabling problems.

The survey paper also stated a future research challenge whichis to close the gap between theory and practice. Most research is applied on simplified versions of the problem while "reformulations of problems will better reflect more constraints in real world environments".

Existing timetabling techniques do not adequately represent individual preferences. Individual preferences, which are a type of soft constraint, are either not included or are grouped togetherthus disregarding the individual characteristics. A Multi-Agent System (MAS) is proposed to represent individual humans as agents where the agents then learn and cooperate to repair a schedule after a change in conditions.

This paper explores five benefits of individual representation in solving timetabling problems that will be discussed in the following sections.

A. Learn individual preferences.

Individual preferences are not taken into consideration because it is difficult to capture and model these preferences. Users are able to articulate simple statements such as "I prefer morning classes to night classes" but may omit or fail to clearly articulate "I prefer night classes to three morning classes in a row." The human user might only realize their own preferences after given a timetable solution and only then is the user able to reason and articulate preferences.

This paper describes a method for agents to learn and create a model of other agent's individual constraints via iterated negotiations. By learning, better solutions can be generated by each iteration. In other timetabling approaches, artificial intelligence techniques like constraint-based and case-based reasoning are applied to generate timetable solutions but not to learn other individual constraints.

B. Cooperate by communication and negotiation.

Given a timetable, if there exists two stakeholders A and B, for which a better solution can be obtained by swapping their timeslots, a measure of the improvement needs to be quantified and is defined here as the utility value. It is necessary for the system to know the individual's preferences in order to calculate the utility value.

This paper suggests that agents use the Contract Net Protocol as a mechanism to 'offer a timeslot' and receive bids to swap. By using a distributed and not a centralized computation, each bidding agent can compute its own utility value and bid accordingly. With ubiquitous computing becoming a reality, it is feasible for each timetabling agent to reside and run computations on a human user's device. In traditional timetabling approaches, distributed mechanismshave been applied to find a solution more efficiently but not to model individual human users or map computation to a ubiquitous network of devices.

C. Minimise the human aspects of change.

Once a timetable is generated and given to stakeholders, what happens if conditions have changed, for example if a room is no longer available, a lecturer falls sick and a substitute with a set of new preferences enters the system or there is a sudden reshuffling of units amongst existing lecturers. From the human aspect, once a user has received a timetable, the user proceeds to make other time commitments. Hence they are reluctant to see changes to the original timetable.

Instead of computing the solution from the start, traditional timetabling approaches have been successfully combined with scheduling repair approaches to produce a hybrid algorithm [3]. However, the objective of hybridization in such cases is to generate a better coarse-grain timetable and not to consider the human aspect or reduce the disturbance impact. The human aspect is the objective of this paper.

D. Human users can understand how the solution is generated.

Given a timetable, it is difficult for the non-technical user to understand for example, how graph colouring techniques can be mapped to timetabling problems or how evolutionary, ant colony optimisation algorithms and artificial immune algorithms arrive at a timetable solution.

This paper proposes that agents represent human users that can negotiate with other agents for timeslots to increase their respective utility. Once a solution is reached, a human user can read the communication logs for each iteration, understand and trace through how the timetable solution is generated. The provenance or derivation of the timetable solution is useful to analyze the performance ("does my agent make the right decision during negotiation?") and provide an intelligible decision-making audit trail [4].

E. Human users can make changes to influence the solution generated.

Given a timetabling technique, if an individual stakeholder would like to influence the final timetable solution, it is difficult to know what to change and by how much. For evolutionary algorithms, even if each individual preference is completely captured and modelled, a human user is unable to meaningfully make a change to influence a certain outcome.

This paper proposes a system that generates logs which provide provenance. By analysing the logs, the human agent can determine which timeslot his agent offered and how it responded to another agent's offer. The human user can remove, add or modify preferences to make the agent a more effective representative in negotiations. Based on the offers from other agents, the human user can determine which timeslots are high or low in demand. The global demand for a timeslot is the accumulated demand from individual agents.

In traditional approaches, this global demand is unknown *a priori* because it is dependent on relative preference. Hence, this paper proposes a market-based approach in which global demand can be determined from the collective actions of individual agents.

II. Background

A. TimetablingProblem

A timetabling problem is a problem with four parameters: T, a finite set of times; R, a finite set of resources; M, a finite set of meetings; and C, a finite set of constraints. The problem is to assign time and resources to the meetings so as to satisfy the constraints as far as possible [5]. Constraints can be divided into two categories. Hard constraints are constraints that must be satisfied, such as a teacher cannot be in two different rooms at the same time. Soft constraints are 'nice to have' constraints, such as a teacher would prefer to teach in the morning. In this paper, soft constraints and preferences are used interchangeably.

In this paper, the school timetabling problem is used to test the multi-agent system schedule repair algorithm. This timetabling problem has the following characteristics:

- *Homogenous stakeholder types.* No categorization of lecturer or student groups. In the MAS, there is no subclass of agents that has a higher or lesser ability than another. Agents differ by the values of the personal constraints not in their ability.
- *Homogenous rooms*. No differentiation of rooms or resources. While such constraints are valid and are often real, we assume that all rooms have the same basic facility and capacity.
- *Homogenous units.* No differentiation of lecture or tutorial or lab. Differentiation of type has implication for rooms, the size (lecture hall or tutorial session) or facilities (ICT or lab equipment). This third characteristic is implied from the second characteristic of homogenous rooms.

Soft constraints that are not fulfilled are represented as a negative utility. The value of the utility, U, is calculated as a weighted sum of the constraints as follows:

$$U = \sum_{i=1}^{n} w_{ci} \times frequency(c_i) \qquad (1)$$

where c_i is the ith constraint and w_{ci} is the weight assigned for the ith constraint. Frequency(c_i) is the number of times the constraint has been violated.

In this paper, violated constraints produce a negative utility while satisfied constraints have zero value. Therefore, the maximum utility possible is zero.

B. Multi-Agent Systems

The agent can be defined as “anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators" [6]. A multi-agent system is simply an environment that contains more than one agent.

In a Multi-Agent System (MAS), groups of agents interact with each other to achieve some goals. The agent’s individual goal may or may not be aligned with the group’s goal.

The agents have a limited perception and knowledge of the world. Consider that if each agent was to have a complete knowledge of the world, then the behaviour of the system would be akin to a centralized decision maker working through the lower level actuators and sensors [7].

There are several applications that have demonstrated the benefits of using such multi-agent systems. Industrial applications were developed as early as 1987 [8] in areas including process control [9], manufacturing [10] and many other areas. An example of a framework is the Procedural Reasoning System (PRS) used for the Oasis system for air traffic management at Sydney Airport, the basis for Swarmm an agent-based simulation system for Australia’s Defence Science and Technology Organisation and also in general business software for running call centres or internet services [11].

The advantages and disadvantages of using MAS is similar to that of using Object-Oriented Design and Analysis (OODA) and Object-Oriented Programming (OOP) in software development. The use of agents allow a one to one mapping from real world user to software representative which can speed up the software design lifecycle.

A human user can configure their own agent's constraints or configuration without making it known to the public. In the real world, human users may have personal preferences that they would not disclose to other people. Not disclosing the true value of a preference while pursuing a specific goal is a human behaviour that previous timetabling techniques have not considered. In previous timetabling techniques, constraints have to be global because computation is done at a single point. MAS allows localized constraints that are only known to the human user and not to others.

The agents indirectly express the human user's preferences by offering timeslots for bidding and by bidding for other agent's timeslots. This process is done via the Contract Network Protocol mechanism.

C. Contract Network Protocol

Contract Net Protocol (CNP) is a coordination mechanismwhich employs a manager that announces bids for a contract, collects bids from contractor agents, announces the award of the bid to the agent(s) and repeats until the goal is achieved [12].The interaction protocol specifies four main steps for each contract between the manager/initiator and contractor/participant as shown in Fig 1.

1. The initiator sends out a Call for Proposals (CFP).
2. Each participant reviews the CFPs and bids on the feasible ones accordingly.
3. The initiator chooses the best bid and awards the contract to the respective participant.
4. The initiator rejects the other bids.

Figure 1: CNP Interaction Protocol

CNP has been applied successfully in a variety of domains including scheduling [13], cooperative pursuit [14], market based approach multi-team robot cooperation [15] and others.

In this paper, all agents have the ability to call for proposals and submit a bid. For each iteration, an agent is selected to take up the manager role. The call for proposals is a timeslot offer to swap with another agent. The other agents will take up the contractor role, review their utility and offer a timeslot to swap. The manager chooses the bid that maximizes their utility and the timetable is updated accordingly.

One noticeable problem with CNP is the high communication overhead required. In order to overcome this overhead, researchers have produced several extensions to the CNP [16].

In this paper, the communication overhead is reduced by agents learning and developing a model of other agents. A model allows a contractor agent to estimate the preferences of the manager and bid accordingly. This results in better quality bids. Given an accurate model, the manager can filter unlikely contractors and the contractors can choose not to bid for a contract that they will unlikely win. This results in less bids and a reduced communication overhead.

The Multi-Agent System is implemented using the Java Agent DEvelopment framework (JADE) [17] that has the Contract Net Protocol built in. The Foundation of Intelligent Physical Agents (FIPA) established standards for multi-agent systems. JADE is FIPA compliant that means it is possible for two independently developed multi-agent systems to communicate with one another.

JADE was selected instead of other frameworks because its agents and interaction protocols can be modified not just for agent-based modelling research but also for scheduling or optimization research.

III. METHODOLOGY

A. Algorithm

The design for the system is show in Fig. 2 below. The timetable solution is first randomized. This initial timetable solution simulates the scenario where the solution is no longer viable due to changing conditions. Because of this, a schedule repair is required. The schedule repair is implemented using a learning and cooperating multi-agent systems provenance-centred approach.

```
1.  Randomise timetable solution.
2.  Initialise all agents.
3.  Do for each iteration{
4.    All agents calculate utility.
5.    A *selection strategy* determines which agent will be
      the manager agent.
6.    Manager agent calls for proposals by offering a
      timeslot to swap.
7.    Contractor agents consider the proposed time by
      calculating their own utility. Using their own
      preferences, the *manager's preference model* and
      *cooperation level*, they will propose a time slot,
      otherwise they will reject the call.
8.    Manager considers all proposals and calculates the
      utility values.
9.    Manager picks the proposal that maximizes his
      utility, informs the result to all contractor agents
      and the timetable is updated. If there are no good
      proposals, the manager rejects all proposals.
10.   All agents, whether their proposals are accepted or
      rejected, *update the manager's preference model*.
11.   Update *communication logs*.
12. }Repeat steps 4 through 11 until the termination
    condition is reached.
```

Figure 2: Algorithm design

The loop terminates when the number of iteration has reached a predefined maximum. The selection strategy, agent preference model, cooperation level and communication logs are detailed in the following sections.

B. *Selection strategy*

The selection strategy is an influential parameter in determining both the quality and efficacy of the solution. This research implements and analyses five types of selection strategies.

- Round robin. All agents are selected in sequence. An agent can only be selected again if all other agents have been selected before.
- Least utility. The agent with a higher need, i.e. the lowest utility value, is selected first, then followed by a different agent with the next lowest value. An agent can be selected multiple times in the same round.
- Least utility round robin. The agent with a higher need, i.e. the lowest utility value, is selected first, then followed by a different agent with the next lowest value. An agent can only be selected again if all other agents have been selected before.
- Roulette selection. Agents with a higher need will have a higher probability of being selected. After each iteration, the utility for all agents is recalculated. An agent can be selected multiple times in the same round.
- Ranked tournament selection. Agents are ranked in order of their utility. Agents with the lowest utility will rank lowest. Unlike the roulette selection strategy where agents with no or significantly smaller needs will never be selected, the ranked tournament selection strategy gives every agent a chance of being selected based on its rank. Using the linear ranked tournament selection, the fitness value is determined by the rank as shown below:

$$Fitness(r) = 2 - SP + 2 \cdot (SP - 1) \cdot \frac{(r-1)}{(n-1)} \quad (2)$$

wherer is the rank of the agent, *SP* is the selection pressure [1.0, 2.0], and *n* is the number of agents. Once the fitness is determined, the probability of rank *r* being selected can be expressed as:

$$Probability(r) = \frac{Fitness(r)}{\sum_{k=1}^{n} Fitness(k)} \quad . \quad (3)$$

C. *Agent Preference Model*

In the first iteration, the agent has no information about the other agents' preferences. Therefore the agent perceives that every other agent is amenable to any timeslot(see Table I).

TABLE I. ORIGINAL PREFERENCE MODEL

Mon	Tue	Wed	Thu	Fri
100	100	100	100	100
100	100	100	100	100
100	100	100	100	100
100	100	100	100	100

At step 6 in the algorithm, the agent will propose a timeslot to maximize its utility after a swap if it satisfies the cooperation threshold. At step 9, there are two possibilities. If the agent's proposed timeslot was accepted, the agent perceives the manager agent has a preference for either that time slot or day. The agent updates the model accordingly by increase the weights. For example if agent accepts Tuesday 10-12, the model is updated for that timeslot and day as in Table II.

TABLE II. PREFERENCE MODEL AFTER ACCEPT

Mon	Tue	Wed	Thu	Fri
102	102	102	102	102
102	102	102	102	102
100	100	100	100	100
100	100	100	100	100

If the agent's proposed timeslot was rejected, the agent perceives the manager agent has a negative preference for either that timeslot or day. The agent updates the model by decreasing the weights for that timeslot and day, as in Table III.

TABLE III. PREFERENCE MODEL AFTER REJECT

Mon	Tue	Wed	Thu	Fri
98	98	98	98	98
98	98	98	98	98
100	100	100	100	100
100	100	100	100	100

In the next iterations, the agents have information about the other agents' preferences based on the acceptance and rejection of bids. At step 6, the agent will select a timeslot that maximizes its own utility and maximizes the perceived preferred timeslot of the manager. This change in utility must satisfy the cooperation level. At step 9, the model is updated whether the manager agent accepts or rejects the bid.

After a number of iterations, each agent will learn the preference of other agents which reduces the communication overhead of the CNP, increases the quality of the bids, and the human user can gain an understanding of the relationship between its agent and other agents.

D. *Cooperation level*

The cooperation level is the minimum change in utility, ΔU, in order for the contractor agent to bid for a proposal and for the manager agent to accept a proposal. The cooperation level can be positive, zero or negative.

A positive cooperation level means that the agent requires an increase in utility for a swap to occur. A high value means the agent is more demanding.A zero cooperation level means that the agent is neutral and is willing to swap even if there is no change in utility value. Anegative cooperation value means that the agent is altruistic and is willing to sacrifice its own utility in order to increase the utility of another agent and perhaps the overall utility of the system.

By using a cooperation level value, the MAS models the human aspect of timetabling. In this paper, the effects of zero and negative cooperation are explored.

E. *Communication logs*

One of the strong motivations for MAS over traditional timetabling techniques is provenance: the ability for human users to derive a timetable solution as a sequence of logical steps. Communication logs are generated for each agent to record the details of each iteration. This means a human user is able to view the logs of their own agent to answer:

a) How was the subject assigned to a specific timeslot?
b) Was there a better alternative in previous iterations?
c) What were the timeslots my agent offered?
d) What were the timeslots my agent bid for?
e) What timeslots were always offered?

In a provenance-aware multi-agent health care system [4], the system was able to assist in analysing performance and provide an audit trail to assess whether decisions and procedures were properly followed. This audit trail is particularly useful in timetabling for the schedule repair phase.

From the organizational perspective, provenance provides evidence towards the necessity of reassigning timeslots and why certain individuals were affected and others were not. By analysing logs, the organization has evidence for policies or plans for resource and planning.

From the individual's perspective, provenance provides data to change the individual agent's configuration to be better representative. The human user can determine the demand of certain timeslots and make corresponding adjustments in order to maximize the utility in future timetabling rounds.

IV. Experimental Results

A. *Experiment set up*

The timetable ranges from Monday to Friday, with slots 8-10, 10-12, 12-2 and 2-4. There are 40 units to be scheduled amongst 10 teachers in 3 rooms. The soft constraint is the preference for early morning classes or late afternoon classes. The timetable is small enough to reduce processing requirements but still allow analysis of the agent behaviours throughout the simulation.

The experiment is set up as a factorial 2×2 experiment by varying the learning model and cooperation level.

The learning model has two levels:

L_{yes}	-Apply learning model
L_{no}	- No learning model

The cooperation level has two levels.

$C_{neutral}$	- Neutral cooperation level
$C_{altruistic}$	- Altruistic cooperation level

The four possible simulation quadrants isshown in Table IV.

TABLE IV. SIMULATION QUADRANTS

	L_{yes}	L_{no}
$C_{neutral}$	A	B
$C_{altruistic}$	C	D

For each quadrant, five selection strategies are simulated.

RR	- round robin
LU	- least utility
LURR	-least utility round robin
ROU	- roulette selection
RT	- ranked tournament selection

Each quadrant was simulated for 500 runs. Hence, the entire simulation runs 2000 times.

B. *Utility*

As explained before, the maximum utility is zero. Table V below shows that amongst the five selection strategies the best is the Roulette selection strategy in all four quadrants. The

Round Robin is the least effective selection strategy in every quadrant because it does not consider the utility or needs of the agents but its inclusion is useful for benchmarking purposes.

In both cases of neutral and altruistic cooperation level, the use of a learning model results in a higher utility value. When the learning model is kept constant and the cooperation level is varied, the utility level is quite similar. This suggests that cooperation level has a minimal effect on the utility.

TABLE V. UTILITY

	Selection Strategy				
	RR	LU	LURR	ROU	RT
A (neutral, learning)	-0.26	-0.15	-0.21	**-0.13**	-0.22
B (neutral, not learning)	-0.33	-0.22	-0.26	-0.18	-0.28
C (altruistic, learning)	-0.25	-0.17	-0.19	**-0.13**	-0.20
D (altruistic, not learning)	-0.32	-0.20	-0.23	-0.18	-0.31

This experiment shows that the selection strategy, and learning models have an impact on the effectiveness or utility of the solution generated.

C. *Acceptable proposals*

At the end of the each iteration, the number of acceptable proposals is counted.An acceptable proposal is one in which the utility gained would be higher than the cooperation level. For this experiment, the maximum value is 900, which is the case when, for all iterations, all proposals are acceptable. A high value indicates that there are more choices for the contractor manager. On the other hand, it also means more processing to determine the best one amongst many more proposals. It is argued that a few good proposals are better than many mediocre proposals.

Based on Table VI, Quadrant A and C have less acceptable proposals than Quadrant B and D. This indicates that agents without a learning model produce more proposals.

TABLE VI. ACCUMULATED ACCEPTABLE PROPOSALS

	Selection Strategy				
	RR	LU	LURR	ROU	RT
A (neutral, learning)	562	459	628	543	618
B (neutral, not learning)	**709**	**667**	**773**	**727**	**764**
C(altruistic, learning)	609	555	640	540	621
D(altruistic,not learning)	**770**	**746**	**773**	**742**	**769**

When comparing Quadrant A against C or Quadrant B against D, the impact of an altruistic cooperation level seems minimal. In a few cases (e.g. LURR, ROU, RT), the results are similar. Therefore, cooperation level seems to play a minimal impact on the acceptable proposals generated.

From the table, it can be inferred that the learning model acts as a filter prior to broadcasting a proposal. Through multiple iterations, each agent builds and refines the Agent Preference Model of the other agents. A rejected proposal reduces the chance of a repeat or a similar proposal in future. If there are no bids worth proposing, the contractor agent can choose to refuse the call for a proposal or allow the deadline time to pass. This has the desirable effect of reducing network congestion.

D. *Variability*

The disturbance impact is measured using the variance. The variance between two timetable solutions is the difference in the initiator's proposed time and the subsequent accepted proposed time. This is calculated by taking the difference of the days and the timeslots. Two timetables with similar days and times would have a smaller difference and vice versa. The meanof the variance is shown in Table VII. The standard deviation of the variance is in Table VIII.

TABLE VII. MEAN OF THE VARIANCES

	Selection Strategy				
	RR	LU	LURR	ROU	RT
A (neutral, learning)	**5.62**	**4.85**	**5.88**	**4.89**	**5.67**
B (neutral, not learning)	6.45	6.37	6.71	6.39	6.41
C (altruistic, learning)	**5.62**	**4.77**	**5.80**	**4.89**	**5.68**
D (altruistic, not learning)	6.58	6.33	6.69	6.44	6.61

The agents that learn have a consistently lower mean of variance than agents that do not learn. This could be because once a model of other agents is determined, only likely bids are offered.

The mean of the Least Utility selection strategy is the lowest in all quadrants. The mean of the Least Utility Round Robin selection strategy is highest in all quadrants. This should indicate that when variation is desired the Least Utility Round Robin strategy is better than the Least Utility strategy. However, since the difference between the various strategies is small, this hypothesis requires further study.

TABLE VIII. STANDARD DEVIATION OF THE VARIANCES

	Selection Strategy				
	RR	LU	LURR	ROU	RT
A (neutral, learning)	1.62	1.66	1.63	1.94	1.31
B (neutral, not learning)	1.27	**1.11**	1.22	1.32	1.15
C (altruistic, learning)	1.52	1.49	1.80	1.91	1.30
D (altruistic, not learning)	1.45	1.40	1.26	1.34	1.20

Note that Least Utility selection has the lowest (1.11) standard deviation when it is altruistic and not learning. This result would be useful in designing a system where variation needs to be reduced from iteration to iteration.

From the scheduling repair perspective, a solution that reduces the variation of the timetable is better. The Ranked

Tournament selection strategy has the lowest standard deviation of variance in Quadrants A, C and D and the second lowest at Quadrant B. Therefore, if the least variation is required, the Ranked Tournament selection strategy is the preferred choice.

E. Provenance

As described in the introduction, a multi-agent system approachprovides a way for the human user to trace the decision making process and make improvements on the system. Fig. 3is an iteration log from a simulation sample.

```
Iteration number: 35
Agent 2 calls for proposal for [Monday, Room2, 8-10am]
Agent 0 proposes [Monday, Room2, 12-2pm], utility = 0
Agent 1 proposes [Monday, Room1, 10-12pm], utility = 0
Agent 3 proposes [Monday, Room1, 8-10am], utility = -5
Agent 5 proposes [Monday, Room1, 10-12am], utility = -5
Agent 2 accepts Agent 0 proposal [Monday, Room2, 12-2pm].
Timetable swap success.
```

Figure 3: Example of a communication log

From this example, a human user can trace the decision point when his agent (i.e.,Agent 2) was assigned to the day, room and time(i.e.,Monday, Room 2 from 12 to 2 pm). The human user can view all the bids or proposals from other agents and the corresponding utility to determine if the agent made the correct decision or whether the utility is representative of the human user's actualpreference. The user can then reconfigure the decision-making parameters such as the cooperation level or the weights for the Agent Preference Model to better reflect their preference.

At a global level, by compiling the logs, the real and actual scheduling demand is determined. Summary data for a simulation run is shown in Fig. 4 below:

```
SimulationRun: A02-13 081448
Call for proposals: 66% morning, 34% afternoon
Number of bids: 102 morning, 98 afternoon
```

Figure 4: Example of a summarydata

In the sample above, 66% of agents are calling for bids to swap a morning session. Therefore there is a lower demand for morning sessions compared to afternoon sessions. The number of bids is equally distributed between morning and afternoon. This global information is useful for the human user to reconfigure his agent or for the institution's policy makers or planners.This information is representative because it is an aggregate of individual lecturers' preferences evidenced by their call for proposals and bidding patterns. In this case, the actual preference of the individual user and aggregate users is determined based on market forces.

The user can also determine whether there were any similar bids and the frequency of such instances as shown in Figure 5 below:

```
SimulationRun: A02-27 151917
[Monday, Room2, 10-12pm], offered 85 times
[Tuesday, Room1, 10-12pm], offered 78 times
[Monday, Room1, 10-12pm], offered 78 times
[Tuesday, Room2, 12-2pm], offered 67 times
[Monday, Room1, 12-2pm], offered 63 times
```

Figure 5: Extract from detailed summary data

A human user is able to identify which bids are common and adjust the agent's constraint when required.

Since MAS have agents representing human users, it is also possible to have human-in-the-loop systems where a human user is able to interrupt the schedule repair to reconfigure the agent halfway through the iterations. From the communication logs and summary data, a human user is able to make changes to adapt to the global demand based on the bids the user sees. MAS allows a provenance centred approach that other timetabling techniques do not provide.

V. Discussion

In this experiment, the use of a learning model does not provide a clear benefit from utility or variability aspect. From a communication aspect, the application of a learning model reduces the number of proposals that in turn reduces network congestion. A future study would increase the number and type of preferences. A more complex problem would be a better test of the usefulness of a learning model. Future study would validate the learning model against real user data to determine whether the system accurately models the user's preferences.

This research prepares the groundwork for future work on adaptive cooperation in which the cooperation level can acts as a mechanism to relax or restrict the solution space during successive iterations.

In this research, the cooperation level is fixed to be the same for all agents. In later research, the cooperation level can be localized to explore the impact of cooperating/non-cooperating agents in developing a timetable solution.

The Contract Net Protocol is shown to be an effective communication mechanism for schedule repair. Future work would look into simultaneous bilateral or multilateral communication between subgroups of neighbouring agents. This would better reflect the real world negotiation process in scheduling repair.

The system described in this paper provides a schedule repair that considers the human aspects. Further study would look into combining traditional scheduling approaches with this system over a larger sample size.

VI. Conclusion

In this paper, a multi-agent system schedule repair is implemented with emphasis on the human aspect using a provenance-centred approach. The system is simulated using five selection strategies and a 2×2 factorial experiment for cooperation level and presence of learning model.

The choice of selection strategy has a stronger impact on utility than the cooperation level or learning model. The presence of a learning model reduces the number of proposals from agents that can result in a lower communication overhead. Using a learning model also reduces the variation for timetable solutions that can be helpful to minimise disturbance impact due to a changing conditions.

These results provide a foundation for future study on advanced learning algorithms, complex soft constraints and improved communication mechanisms.

References

[1] Oprea, M., 2007. MAS_UP-UCT: A multi-agent system for university course timetable scheduling. *With Emphasis on the Integration of Three Technologies*, 2(1), pp.94–102.

[2] Qu, R., Burke, E.K., McCollum, B., Merlot, L.T.G., Lee S.Y., 2009. A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling*, 12(1), pp.55–89.

[3] Marco, C., Mauro, B., Krzysztof, S., Olivia, R., 2006. An effective hybrid algorithm for university course timetabling. *Journal of Scheduling*, 9(5), pp. 403-432.

[4] Kifor, T., Varga, L.Z., Vazquez-Salceda, J., Alvarez, S., Wilmott, S., Miles, S., Moreau, L., Provenance in Agent Mediated Healthcare Systems. IEEE Intelligent Systems

[5] Burke, E.K., Kingston, J. & de Werra, D., 2004. Applications to timetabling. In J. Gross & J. Yellen, eds. *The Handbook of Graph Theory*. Chapman Hall/CRC Press, pp. 445-474.

[6] Russell, S. & Norvig, P., 2003. *Artificial Intelligence a Modern Approach* 2nd ed., Prentice Hall.

[7] Panait, L. & Luke, S., 2005. Cooperative multi-agent learning: The state of the art. *Autonomous Agents and Multi-Agent Systems*, 11(3), pp.387–434.

[8] Jennings, N. & Wooldridge, M.J., 1998. *Agent technology: foundations, applications, and markets*, Springer Verlag.

[9] Jennings, Corera, J.M. & Laresgoiti, I., 1995. Developing industrial multi-agent systems. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*. pp. 423-430.

[10] Parunak, Applications of distributed artificial intelligence in industry. In G. O'Hare & N. R. Jennings, eds. *Foundations of Distributed Artificial Intelligence*. Wiley, pp. 139-164.

[11] d'Inverno, M., Luck, M., Georgeff, M., Kinny, D., Wooldridge, M., 2004. The dmars architecture: A specification of the distributed multi-agent reasoning system. *Autonomous Agents and Multi-Agent Systems*, 9(1), pp.5–53.

[12] Smith, R.G., 1980. The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers, vol. 100, (12), pp. 1104–1113.

[13] Ouelhadj, D.,Hanachi, C., Bouzouia, B., Moualek, A., Farhi, A., 2002. A multi-contract net protocol for dynamic scheduling in flexible manufacturing systems (FMS). In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*. pp. 1114–1119.

[14] Zhou, P.C., Hong, B.R., Wang, Y.H., Zhou, T., 2005. Multi-agent cooperative pursuit based on extended contract net protocol. In *Machine Learning and Cybernetics, 2004. Proceedings 2004 International Conference on.* Pp. 169-173.

[15] Lim, C.S., Mamat, R. & Braunl, T., 2009. Market-based approach for multi-team robot cooperation. In *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on*. pp. 62–67.

[16] Bozdag, E., 2008. A Survey of Extensions to the Contract Net Protocol.

[17] Bellifemine, F., Bergenti, F., Caire, G., Poggi, A., JADE – a java agent development framework, 2005. *Multi-Agent Programming*, pp. 125-147.