A multi-agent system for automated timetabling with shared resources

João Pedro Pedroso

Faculdade de Ciências da Universidade do Porto
Departamento de Ciência de Computadores
Rua do Campo Alegre 823
4150-180 Porto, Portugal
e-mail: jpp@ncc.up.pt

We propose an automated timetabling system for a typical situation in universities, where agents (usually departments or faculties) compete for a set of resources (lecture rooms) on a given number of time slots.
Each agent uses its own algorithm (which might be unknown to the others). A central system decides whether some agent is granted a resource or not, based on a list of requests and on a certificate, obtained from each agent, asserting that it does not have requests with priority higher that a certain amount.
Priority is measured primarily by the number of attendees and some requirements for particular features on the resources, but other criteria are proposed for ties.
We describe a prototype implementation, in use at the Faculty of Sciences, University of Porto.

1 INTRODUCTION

In most universities, a subset of the rooms available for lectures is shared by several departments. On the other hand, each department may have its own rules for constructing the timetables: student preferences, lecturer preferences, breaks, etc. can be handled differently by different departments. For example, a department of Mathematics may want to ensure that no more than two hours of theoretical classes are given in a row, for students not to fall asleep, while a department of Physical Education may wish that all classes in the gymnasium are given in a row, so that the students do not have to shower too often, and a department of Politics may give a very high priority to the preferences of lecturers which are presently ministers.

Notice that the problem tackled in this paper does not exist if all the departments are self-sufficient in terms of room availability; is only makes sense if there is a lack of rooms in at least one of them. This is actually the current situation in the Faculty of Sciences at the University of Porto: the buildings of some departments are under construction, and hence they have no rooms under their direct control. The "standard" practice has been the following: the departments which have enough rooms make their timetables independently, in

a first stage; the other departments can, in the second stage, use the "holes" (unused time slots) on these rooms. This was a rather contested procedure, as for the departments which have no rooms it is virtually impossible to produce reasonable timetables, and gave rise to a new authority who decides who can use each departments rooms.

The aim of the current paper is, hence, to establish a set of rules that can be used by the authority that controls the rooms, so that it can allow/interdict the use of a room corresponding to each request that a department makes. This provides a way for each of the departments can construct its timetable independently of the others (with the exception of room occupation), while keeping a high degree of "fairness" on the room attribution.

The actual rules used by the room authority might be different from university to university, but the concepts developed here should be common to many universities which share resources.

2 RULES FOR ROOM ACCESS

The basis for the construction of the rules is mostly common sense. We will assume that there is a set of rooms which are not assigned to any departments (common rooms), as well as a set of rooms which belong to some department, but which might be used by other departments.

The concepts and rules are the following:

- Timetables are constructed independently, except for room attribution, by each of the departments.

- At every step in the construction procedure, each of the departments makes a request concerning the event that it wants to add to the solution; for this event, the department requests a room at a given time slot.

- The room authority dispatches one request on each step.

- The measures for the priority of a request are the following:

  1. the number of students that attend the corresponding event;
  2. the number of compatible rooms with the event (i.e., rooms large enough and with the some required features);
  3. if the department that makes a requirements to a room owns that room, it has priority over requests for that room by other departments;
  4. if at a given step if there are no requests for a room from the department that owns it, precedence is given to requests for common rooms, rather than requests for rooms on other departments;
  5. requests that lead to better (partial) solutions have priority over requests that lead to worse (partial) solutions.

- The preceding rules are goals; this means that the second rule is only used for deciding on ties of the first rule, the third is used on ties of the first and second, and so on.

- Room clashes, attribution of rooms which do not fulfil the required features, and student clashes, are only dispatched by the room authority if at a given step there are no non-clashing requests.

3 ALGORITHMIC BACKGROUND

From the exposed above, it becomes clear that we need a goal programming, multi-criteria approach for the solution of the timetabling problem of each of the departments, which we propose in this section, based on (Pedroso 2003). The problem considered for each department is that of finding a time slot and a room for each element of a set of events $\mathcal{E}$ of the responsibility of that department, such that a set of constraints $\mathcal{C}$ is respected "as much as

possible", and in a predefined order. We will assume that each department will pose its requests ordered by the number of students, as this is the main criterion that the room authority will use for dispatching.

Given the set of events $\mathcal{E}$, a set of time slots $\mathcal{T}$ and a set of rooms $\mathcal{R}$, the search space is $\mathcal{S} = (\mathcal{E} \times \mathcal{T} \times \mathcal{R})$. Any solution $s \in \mathcal{S}$ is considered feasible, i.e., all the constraints are considered soft. Each goal is defined by one or more constraints. Let $\mathcal{C}_i$ be the subset of constraints defining goal i, and $v_i(x)$ the amount of violation of constraint i of a solution $x \in \mathcal{S}$. Then, goal i concerns the minimization, for all the constraints in $\mathcal{C}_i$, of the sum of the weighted violation

$$g_i(x) = \sum_{k \in \mathcal{C}_i} w_k v_k(x).$$

We will denote $g(x)$ as the N-dimensional vector of goals, $g(x) = (g_1(x), \ldots, g_N(x))$.

Let the first goal be

$$g_1^* = \min\{g_1(x), \forall x \in \mathcal{S}\}.$$

The second goal is then

$$g_2^* = \min\{g_2(x), \forall x \in \mathcal{S} : g_1(x) = g_1^*\}.$$

If there are N goals, the objective of the problem is then to obtain:

$$g_N^* = \min\{g_N(x), \forall x \in \mathcal{S} : g_{N-1}(x) = g_{N-1}^*, \ldots, g_1(x) = g_1^*\}.$$

3.1 *Solution classification*

During the search, a department classifies solutions $x \in \mathcal{S}$ according to the corresponding value of the N goals. For two solutions $x, y \in \mathcal{S}$, the precedence relation between them is:

$g(x) \prec g(y) \Leftrightarrow$

$g_1(x) < g_1(y)$, or

$g_1(x) = g_1(y)$ and $g_2(x) < g_2(y)$, or

$\ldots$

$g_1(x) = g_1(y),\ g_2(x) = g_2(y),\ \ldots,$

$g_{N-1}(x) = g_{N-1}(y)$ and $g_N(x) < g_N(y)$.

This relation will be used to compare solutions; notice that it can be used both for the case of complete solutions and the case of partially constructed solutions. For the latter case, the condition for this relation to make sense is that the number of unplaced events on the two solutions being compared is the same.

3.2 *Greedy functions*

The algorithm proposed for each department is a simple greedy procedure. On this procedure (Algorithm 1) each department checks the best placement (slot and room) for each event that has not yet been assigned (lines 4 to 8), based on a current partial solution x. The best placement found is posed as a request to the room authority.

3.3 *Dispatch*

The room authority dispatches the events on a pool based on the criteria defined in section 2. Let $\mathcal{D}$ be the set of departments, $n(e)$ be the number of students attending event e, $c(e)$ be the number of rooms compatible with event e, $d(r)$ be the department that owns room r (or -1 if it is a common room, not assigned to any department). The dispatch algorithm is presented in Algorithm 2.

4 RESULTS

Some preliminary tests of this algorithm have been made with variants of the benchmark problems of the *International Timetabling Competition* (Paechter 2003) (more details on the benchmark test suite and on other meta-heuristics for this problem are presented in (Socha et al. 2003)).

The first goal for the original benchmarks is not to have unsuitable rooms for events, stu-

dent clashes, or room clashes (each of these objectives having weight 1). The second goal is to have no students with 3 or more events on a row, no students with a single events on a day, and no students with events on the last slot of the day.

In this section we use the following notation: for goal 1, F is the number of events features missing, S the number of student clashes and R the number of room clashes. For goal 2, T is the number of three or more events on a row, D the number of single-day events, and L the number events on the last slot of the day, on student schedules.

On the original benchmarks there is no division of the set of events and rooms into departments; the variants have been created by assigning a department to each of the events, and departments to each of the rooms. For this purposes of this test, we have decided that there are three departments (0,1, and 2), and that event with an order number i belongs to the department $i\%3$. Similarly, rooms have been assigned to departments; rooms with an order number i belongs to the department $(i\%3)-1$. When this number is -1, the corresponding room is common, i.e., does not belong to a particular department; this formula also implies that department 2 does not own any room.

We present results for three variants of these benchmarks:

Variant 1: The greedy procedure with no room authority (i.e., only the greedy function is used to decide on the order and placement of each event). This corresponds to the greedy function applied to the original benchmarks.

Variant 2: The room authority dispatches events; the goals of each department are identical: $g_1 = F + S + R$, $g_2 = T + D + L$.

Variant 2: The room authority dispatches events; the first goal of each department is the same, $g_1 = F + S + R$, but the second goals are different:

- $g_2 = T$ for department 1.
- $g_2 = D$ for department 2.
- $g_2 = L$ for department 3.

For presenting the results, we report four rows for each instance:

1. the value of the solution as if there was no division on departments (global);
2. the value of the solution for events of department 0 (without considering events of the other departments on its calculation);
3. the same for department 1
4. the same for department 2

The number of three events on a row T and single-event days D are therefore much higher for department's solutions than for the global solution. The number of last slot events on the global solution is the sum of last slot events for all the departments, as expected.

5 CONCLUSION

We have proposed a method for deciding how to share common resources—rooms—, on the construction of university timetables, based on an authority that dispatches requests that each user of the common resources—each department of the university—poses.

The method has been applied on three cases:

- The case where the dispatch is based only on a greedy function. In this case the events are fixed on the solution starting from the one with the best greedy function, independently of its number of students and of the department it belongs to. These results are reported as "variant 1". They are almost always worse that the results of the other strategies; this indicates that the greedy function could probably be improved, using ideas from the rules proposed for the dispatcher in section 2.

- The case where the dispatch is based on the rules proposed in section 2, and all the departments have similar goals. These results are reported as "variant 2". In this case the events are fixed on the solution starting from those with more students attending. The goals of each department are used for deciding where and when to place the event (i.e., for making the department's request at each step), and also for deciding on ties of the other rules. As expected, the results for this variant are, roughly speaking, similar for all the departments.

- Another case where the dispatch is based on the rules proposed in section 2, but now each of the departments has different goals. These results are reported as "variant 3". On this variant, department 1 has as second goal to have no three-in-a-row events, department to have no single-day events, and department 3 to have no last-slot-of-the-day events. These values on the solution tend to be smaller for the expected departments.

The computational results show that there is a large potential for using the dispatcher method for automated timetabling when there are common rooms shared by several departments. Results should be comparable if the type of share is different (for example, if the are no common rooms), as the extended benchmarks cover a rather general setup.

As extensions to the methods proposed, we envision the inclusion of a local search procedure (see, for instance, (Aarts and Lenstra 1997)), with the room authority deciding on the possibility of an exchange, after the greedy construction. The possibility of using some GRASP (Pitsoulis and Resende 2002) variations where solutions are partially destroyed and reconstructed is also a promising direction, making use of an equilibrium between intensification and diversification, as suggested in (Glover and Laguna 1997).

REFERENCES

Aarts, E. and J. K. Lenstra (1997). *Local Search in Combinatorial Optimization*. John Wiley & Sons.

Glover, F. and M. Laguna (1997). *Tabu Search*. Boston: Kluwer Academic Publishers.

Paechter, B. (2003). International timetabling competition. Internet repository. http://www.idsia.ch/Files/ttcomp2002.

Pedroso, J. P. (2003). A multi-objective/goal programming, approach for timetabling. Technical report, LIACC, Universidade do Porto.

Pitsoulis, L. S. and M. G. C. Resende (2002). Greedy randomized adaptive search procedures. In P. M. Pardalos and M. G. C. Resende (Eds.), *Handbook of Applied Optimization*, pp. 168–183. Oxford University Press.

Socha, K., M. Sampels, and M. Manfrin (2003). Ant algorithms for the university course timetabling problem with regard to the state-of-the-art. Technical report, IRIDIA, Université Libre de Bruxelles, Belgium.

Algorithm 1:
Greedy function, for each of the departments. This function differs for each department on the comparison operator ($\prec$).

GREEDY(x)
(1) $\mathcal{U} =$ set of events with no attributed room/slot on x
(2) **while** $\mathcal{U} \neq \{\}$
(3) **foreach** $e \in \mathcal{U}$
(4) **for** $s = 1$ **to** NS
(5) **for** $r = 1$ **to** NR
(6) $\bar{x} = x \cup \{(e, s, r)\}$
(7) **if** g^* not initialized **or** $g(\bar{x}) \prec g^*$
(8) $s^*[e] = s$; $r^*[e] = r$; $g^*[e] = g(\bar{x})$
(9) **return** e^*, s^*, r^*, g^*

Algorithm 2:
Construction: dispatch by the room authority for requests of each of the departments, until all events are in the solution.

CONSTRUCT()
(1) $x := \{\}$
(2) $\mathcal{U} =$ set of all events
(3) **while** $\mathcal{U} \neq \{\}$
(4) **foreach** $i \in \mathcal{D}$
(5) $(e_i, s_i, r_i, g_i) := \text{GREEDY}_i(x)$
(6) **if** n^* not initialized **or** $n(e_i) > n^*$
(7) $e^* = e_i; s^* = s_i; r^* = r_i$
(8) **else if** $n(e_i) = n^*$
(9) **if** c^* not initialized **or** $c(e_i) < c^*$
(10) $e^* = e_i; s^* = s_i; r^* = r_i$
(11) **else if** $c(e_i) = c^*$
(12) **if** $d(r_i) = i$ or $d(r_i) = -1$ and $d(e_j) \neq j,\ \forall j \in \mathcal{D}$
(13) $e^* = e_i; s^* = s_i; r^* = r_i$
(14) **else**
(15) **if** g^* not initialized or $g_i \prec g^*$
(16) $e^* = e_i; s^* = s_i; r^* = r_i$
(17) $\bar{x} = x \cup \{(e^*, s^*, r^*)\}$
(18) $\mathcal{U} = \mathcal{U} \backslash \{e^*\}$
(19) **return** x^*

Benchmark		Variant 1						Variant 2						Variant 3					
		F	S	R	T	D	L	F	S	R	T	D	L	F	S	R	T	D	L
01:	global	3	2	21	233	47	262	3	2	9	247	35	259	1	0	6	378	61	354
	dep. 0	3	0	8	12	373	85	0	2	1	13	396	56	0	0	1	0	390	211
	dep. 1	0	2	7	10	400	100	2	0	4	14	379	64	1	0	0	14	360	130
	dep. 2	0	0	6	3	414	77	1	0	4	10	373	140	0	0	5	15	374	13
02:	global	9	4	26	184	69	256	5	0	4	250	47	196	7	0	3	374	67	323
	dep. 0	0	2	11	2	390	56	2	0	0	12	413	74	3	0	1	0	405	190
	dep. 1	3	1	8	11	388	93	0	0	2	11	380	65	2	0	1	19	324	123
	dep. 2	6	1	7	12	403	107	3	0	2	8	383	57	2	0	1	25	361	10
03:	global	2	4	16	218	56	273	2	1	16	236	32	236	1	0	16	400	60	322
	dep. 0	1	1	2	14	370	88	0	0	5	12	359	80	0	0	9	1	402	204
	dep. 1	1	3	7	6	355	59	1	1	4	12	385	66	1	0	3	23	375	113
	dep. 2	0	0	7	12	374	126	1	0	7	11	373	90	0	0	4	22	402	5
04:	global	19	22	24	343	103	437	8	7	9	417	66	359	10	7	12	525	70	511
	dep. 0	5	3	6	13	571	112	5	3	2	11	550	103	3	6	5	2	561	214
	dep. 1	8	8	8	13	604	221	1	3	4	10	597	101	1	1	4	25	598	297
	dep. 2	6	11	10	17	541	105	2	1	3	18	565	155	6	0	3	81	457	0
05:	global	8	19	16	400	76	417	6	6	1	347	36	371	7	2	0	493	59	494
	dep. 0	4	3	6	6	601	129	1	3	0	11	563	50	2	1	0	1	591	262
	dep. 1	2	8	5	25	583	138	2	2	0	9	564	204	2	0	0	17	547	206
	dep. 2	2	8	5	17	594	151	3	1	1	16	607	117	3	1	0	45	533	27
06:	global	6	19	14	409	101	491	1	0	0	309	46	308	0	0	2	624	97	482
	dep. 0	2	9	5	17	600	127	0	0	0	8	630	62	0	0	1	0	617	332
	dep. 1	1	5	6	8	590	114	0	0	0	7	573	103	0	0	0	28	595	150
	dep. 2	3	5	3	15	550	252	1	0	0	11	543	143	0	0	1	64	534	0
07:	global	19	47	20	562	139	598	0	0	0	431	53	226	0	0	0	796	101	507
	dep. 0	6	18	5	29	640	122	0	0	0	29	639	124	0	0	0	3	684	328
	dep. 1	6	16	6	17	672	308	0	0	0	14	605	72	0	0	0	20	626	179
	dep. 2	7	13	9	15	629	174	0	0	0	18	639	30	0	0	0	58	517	0
08:	global	10	11	45	348	69	357	5	0	2	247	29	198	3	0	1	574	81	369
	dep. 0	4	2	12	21	455	116	4	0	0	5	462	82	2	0	0	0	484	229
	dep. 1	3	5	13	7	479	118	0	0	1	12	460	67	0	0	0	4	482	133
	dep. 2	3	4	20	20	508	124	1	0	1	3	495	49	1	0	1	30	434	7
09:	global	12	4	18	287	82	311	3	0	3	228	38	177	3	0	0	469	91	364
	dep. 0	3	2	6	5	416	94	2	0	2	9	432	64	2	0	0	0	438	197
	dep. 1	5	1	8	18	422	81	1	0	0	14	417	52	1	0	0	13	387	167
	dep. 2	4	1	4	10	406	137	0	0	1	12	406	61	0	0	0	22	362	0
10:	global	15	1	14	223	57	207	3	0	10	224	35	217	6	1	4	371	67	317
	dep. 0	6	1	3	3	377	53	1	0	6	8	394	83	2	0	0	0	373	201
	dep. 1	5	0	4	2	387	40	2	0	2	9	382	35	2	0	2	18	380	116
	dep. 2	4	0	7	10	355	114	0	0	2	7	367	99	2	1	2	19	361	0

Table 1: Results for the original benchmarks, for the three variants presented: Quality of the solutions for the set of departments (global), and for each individual departments (dep. 1 to 3). F is the number of events features missing, S the number of student clashes and R the number of room clashes; T is the number of three or more events on a row, D the number of single-day events, and L the number events on the last slot of the day, on student schedules.