Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)

Scheduling Meetings at Trade Events with Complex Preferences

Andreas Ernst and **Gaurav Singh** and **René Weiskircher**
CSIRO Mathematical and Information Sciences,
Postal Bag 33, South Clayton VIC 3169, Australia

Abstract

We present a complex scheduling problem where we want to plan meetings between customers and exhibitors at a trade event. One cause of the complexity of the problem is the general nature of the preferences expressed by the participants prior to the event. These map a set of requesters to a set of requested participants and express a preferred length of the meeting. A further source of complexity is that the quality of the solution not only depends on the realisation of the preferences in the solution but also on the distance covered by the customers. We present our solution method and the results of preliminary computational experiments.

Introduction

Tourism accounted for nearly $32 billion of total GDP in 2003-04 (Australian Bureau of Statistics 2005) and is thus one of the main industries in Australia. Tourism Australia (TA) is the official tourism marketing body of Australia and organises a number of trade events annually where providers of tourism services (like hotels, airlines and resorts) meet buyers of these services (for example travel agencies).

To increase the efficiency of the bigger trade events, TA schedules meetings of customers and exhibitors prior to the event. In the run-up to an event, each participant provides a list of the participants they want to meet and assign values to them, which reflect their interest in the other participants. The task is then to compute a timetable for each participant that maximises the value of the scheduled meetings.

The closest application we could find in the scientific literature is described by Bartholdi and McCroan (Bartholdi and McCroan 1990), where they consider the problem of scheduling meetings at a job fair between law students and law firms. But in this application, the set of meetings is fully determined by the preferences submitted by the students and the selection of students by the firms. The task is therfore to schedule the meetings without conflicts. In our application, determining the set of meetings is part of the problem.

Another related work is by Cangalovic et al. (Cangalovic et al. 1998). They study the problem of assigning students to examinations with many practical constraints. Their solution heuristics include a tabu search approach. Hertz and Roberts solve a constrained course scheduling problem by decomposing it into a series of assignment problems (A. Hertz 1998), which is also part of our approach.

If we disregard the constraints caused by the limited time between meetings and the maximum walking speed of a participant, we can simplify our problem to a planar three-dimensional assignment problem. This problem is NP-hard, even if the weights are binary (Frieze 1983). If the costs are invariant in one dimension, such a problem can be solved in polynomial time using a linear program followed by a number of assignment problems (Gilbert and Hofstra 1987). But since we also have the additional goal of minimising the distance walked by each participant (which makes it a variant of the vehicle routing problem), we cannot hope to solve the problem optimally for large instances and instead present a heuristic.

In 2003, CSIRO provided a first implementation of an algorithm for solving a reduced version of this problem (Ernst, Mills, and Welgama 2003). We have now implemented a new algorithm for TA which is able to deal with more complex preferences and multiple objectives.

In the previous version, a preference always consisted of one requesting participant and one requested participant. The new version accepts preferences where a set of participants requests a meeting with a set of requested participants. Such a preference is satisfied if one of the requesting participants meets one of the requested participants. If we do not want to schedule unnecessary meetings, we should schedule at most one of the meetings that satisfy a complex preference.

TA requested the ability to modify the behavior of the algorithm in a flexible way which added extra complexity to the problem. Examples include biasing the solution in favor of customers or exhibitors and the ability to assign an importance to optimisation goals. The goals of the optimisation include planning as many mutually preferred meetings as possible, reducing the distance covered by the customers and increasing the number of scheduled meetings.

Problem Description

There are two groups of participants in an event: The *exhibitors* who are stationed at *booths* with fixed locations and *customers* who visit exhibitor booths. Each customer can meet at most one exhibitor during a session while an exhibitor may be able to accommodate several customers in one session (due to large booths and a number of staff representing the exhibitor). Each participant is associated with a weight, the *importance factor*, between 0 and 1. This enables us to give preferential treatment to participants that are considered important by TA.

An event spans several days and each day has a number of *session blocks* that are separated by longer breaks like a coffee break or lunch. Each session block consists of a number of *sessions* separated by short breaks to enable customers to walk to the next exhibitor booth they want to visit. A *meeting* between a customer and an exhibitor usually lasts only one session but may span several consecutive session in the same session block (*jumbo meeting*).

Prior to the event, the participants receive lists of all participants of the event and can then provide a list of *preferences*. A standard preference consists of one requester, one requested participant, a minimum and maximum duration for the meeting expressed in numbers of sessions as well as a *rank* for the preference. A smaller rank reflects a higher desirability of the meeting.

A complex preference consists of a set of requesters and a set of requested participants. This is useful if organisations send several participants to the event and they want one of them to meet any one participant sent by a different organisation.

In a complex preference, all requesters must belong to the same group (customers or exhibitors) and the same holds for the requested schedules. Just as in standard preferences, complex preferences are also satisfied by scheduling *one* meeting. We call any preference a *customer preference*, when its set of requesters consists of customers and otherwise an *exhibitor preference*.

We do not allow two customer or two exhibitor preferences to *intersect*. We say two preferences intersect if they share at least one requester and one requested participant. It follows that each meeting can satisfy at most two preferences, a customer preference and an exhibitor preference. The values of prospective meetings are set in such a way that a top rank meeting has double the value of a medium rank meeting.

The optimisation has several goals, each associated with a weight between 0 and 1 to enable TA to stress different aspects of the solution.

1. Maximise the number of meetings scheduled that realise a mutual preference. A preference is mutual if it is requested by both participants. Meetings that are only requested by one prospective participant have lower priority.
2. Maximise the sum of the values of scheduled meetings. As already mentioned, a meeting can satisfy at most two preferences: one customer preference and one exhibitor preference. Let the meeting involve customer c and exhibitor e and let v_c be the value of the customer preference implied by its ranking, v_e the corresponding value for the exhibitor preference, w_c the importance factor of c and w_e the importance factor of e. Another factor that influences the value of a meeting is the customer-exhibitor preference balance factor f_c which gives TA the option of giving buyers or sellers preferential treatment. The value of a meeting between c and e is defined as:
$$val(c, e) = f_c w_c v_c + (1 - f_c) w_e v_e$$
3. The system should create meetings where the number of meetings for customers and exhibitors, who have given high preference to each other, is maximised. To achieve this, we assign an additional bonus value to meetings that satisfy two preferences with low rank (high value) that depends on the weight given by TA to this optimisation goal.
4. Another optimisation goal is the reduction of the customer's travel time in order to attend the meetings on their schedule. Each pair of booths is associated with a walking time in seconds. We would like to minimise the total travelling time, which is the sum of the travel times of all customers. We assume that there is no travel time between meetings at the end of one day and the start of the next. The minimization of walking distance was not an optimization goal in our previous version of the system (Ernst, Mills, and Welgama 2003).
5. Analogously to the previous point, the total discomfort experienced by all customer schedules should be minimised. A customer experiences different levels of discomfort when he has to walk to a neighboring booth, has to walk to a different building or has to take a bus to go to a different venue. Each pair of booths is associated with a discomfort level. We treat the discomfort analogously to the travel time.
6. There are several classes of tickets for events and for each of them, a different maximum number of meetings sessions is assigned to the participant by TA. We use the term *meeting session* here because a single meeting may span several sessions and what is limited by TA is the number of sessions where a certain participant can have meetings and not directly the number of meetings. We want to maximise the percentage of meeting sessions that are actually created for each participant with respect to his or her maximum number. When we schedule the meetings for a session, the value of a meeting also depends on how full the schedule of the participants is already. We apply a bonus to the value of the meetings that increases quadratically with the number of missing meeting sessions for a participant. This also increases the fairness because a solution where one participant has a full schedule while another participant has no meetings will have a smaller objective value than a meeting where both participants have a half full schedule.
7. The final component of the objective function should ensure that each participant's schedule contains a high percentage of his or her most preferred meetings. This works analogously to the maximisation of the fill degree of each participants schedule in the previous point. It just adds an extra bonus for meetings that give a highly preferred meeting to a participant who has only a small percentage of his or her high preferences satisfied.

We introduce a parameter for each of the goals that can be used to fine tune the structure of the solutions. These parameters represent the importance given to the goals. Thus, tuning the parameters can be used to strengthen or to diminish the influence of a goal on the solution.

Constraints

The constraints in the system include:

- Each exhibitor (customer) meets with at most a specified maximum number of customers (exhibitors) in a time slot. For a customer, this upper limit is one. For an exhibitor, it depends on the booth size and number of delegates and is part of the input.
- There is at most one meeting between a customer and exhibitor pairing.
- Jumbo meetings (meetings that span more than one consecutive session) have the following properties:
  - they are completely contained in one session block;
  - they are created only if both parties have requested a Jumbo meeting. Each preference gives a lower and an upper bound for the duration of the meeting as a number of sessions. If these values are c_l and c_h for the customer and e_l and e_h for the exhibitor, than the length of any meetings between the two is at least $\max\{c_l, e_l\}$ and at most $\min\{c_h, e_h\}$. This interval may be empty and in this case, we cannot schedule a meeting.
- Each customer has limited time to reach to the next booth between consecutive sessions and this time is the difference of the start date/time of the later meeting and the end date/time of the earlier meeting.
- No meetings are created where there is no corresponding preference in the input.
- Each participant is assigned a maximum number of meeting sessions that cannot be exceeded in a feasible solution.
- For each feasible solution there exists an assignment of the preferences to the scheduled meetings such that each preference is assigned to at most one meeting and each meeting is assigned to at least one preference. Thus, we schedule at most one meeting to satisfy a preference and each meeting realises a preference.

  This constraint means that there should be no superfluous meetings in a schedule. Assume for example there are two customers A and B and one exhibitor C. There are two preferences:

  $$\{A, B\} \to C \quad \text{and} \quad C \to B$$

  We further assume that we have scheduled the two meetings (A, C) and (B, C). During the construction of a schedule we may assign the first preference to the first meeting and the second preference to the second meeting. However, only scheduling the meeting (B, C) would have satisfied both preferences. We have implemented a post-processing algorithm that removes unnecessary meetings of this type in order to enforce this constraint. This constraint was not necessary in the previous version of the system (Ernst, Mills, and Welgama 2003), since there were no complex preferences then.
- No customer has to travel more than a certain amount of time each day.
- No customer suffers more than a certain total discomfort each day.

Re-optimisation

Re-optimisation becomes important when a major company cancels or there are late entries. This will usually occur after publishing the schedules and before the actual event. TA will in this case provide a list of already scheduled meetings.

Our algorithm takes existing meetings into account and does not schedule new meetings that are in conflict with them or satisfy the same preferences. Note that this is also a new feature of the system, since the solution described in (Ernst, Mills, and Welgama 2003) was not designed to deal with existing meetings.

Description of the Algorithm

Our new implementation follows the same principle as the one described in (Ernst, Mills, and Welgama 2003) but is complicated by allowing complex preferences, existing meetings and making the minimization of walking time an optimization goal. A high level overview of the algorithm is shown in Algorithm 1. First we deal with the meetings that are already part of the input. For each of the meetings, we update our two main data structures:

- The meeting data structure contains all scheduled meetings and can efficiently answer questions about the previous or next meeting for a participant given a session.
- The satisfaction data structure stores for each participant their satisfied and unsatisfied preferences and how full their schedules are.

The satisfaction data structure is responsible for computing the value of prospective meetings. This is not trivial because the value of a meeting is not static but changes with every new meeting planned. The reason is that the value of a meeting depends on how full the schedule of each participant is and also on how many of their highly valued preferences are already satisfied.

The meeting data structure is important for excluding meetings because participants are busy with other meetings or cannot fit the meeting into their schedule because of the distance between the two locations. When planning meetings, we also have to look forward in time. The reasons for this are already planned meetings in the input and that we plan the session blocks in non-increasing order of their weights rather than in temporal sequence. Note that we plan the meetings *inside* a session block in temporal sequence. Therefore, when we plan the meetings of a session, there may already be meetings planned for following sessions. So we have to look back *and* forward in time to decide if there is enough time for a customer to attend a meeting given the meetings that have already been planned and the walking distances between the booths.

Because the algorithm plans one session at a time, it tends to plan more and higher valued meetings in the sessions that are scheduled first. TA provides weights to the session blocks that signify how important it is to schedule many highly valued meetings in the block. By stepping through the session blocks in sequence of decreasing weight, we make sure that many highly valued meetings are scheduled in the blocks with high weight.

```
foreach fixed meeting $m$ do
    Update the meeting data structure for both participants;
    Update the satisfaction data structure for both participants;
end
for all session blocks $B$ sorted by decreasing weight do
    for all sessions in $B$ in chronological order do
        Set up the meeting value matrix, creating copies for exhibitors who can take more than one customer;
        Solve the matching problem using Lagrangian relaxation;
        foreach new meeting $m$ do
            Update the meeting data structure for both participants;
            Update the satisfaction data structure for both participants;
        end
    end
end
```

Algorithm 1: High level overview of the algorithm

For each session block, we step through the sessions in chronological order. We compute the *meeting matrix* M, which holds for each pair of customers and exhibitors a value representing the desirability of a meeting between the two during the current session. The entry of a pair has value zero, when a meeting is undesirable or impossible. For each exhibitor, e, who can accommodate k customers with $k > 1$, we introduce $k - 1$ copies that have the same booth location and the same preferences as e. On the other hand, each customer who has a preference for e has the same preference for each copy of e.

For each potential meeting between c and e, we first have to compute minimum and maximum length by looking at the corresponding preferences. A meeting is associated with at most two preferences, a customer preference and an exhibitor preference. The preferred meeting length is an interval determined by the upper and lower bounds given in these preferences as described in the earlier subsection.

There are three other factors that limit the length of a meeting:

1. the number of sessions left in the session block (a meeting can not span a break between session blocks);
2. meetings in the future that are already scheduled. The meeting must be finished by the time e has his next planned meeting and c must have enough time to walk to his or her next planned meeting;
3. the maximum number of meeting sessions c or e are allowed to have and the number planned already.

It is possible that the limits placed on the meeting length by these constraints enforce a maximum meeting length that is shorter than the minimum meeting length expressed by the participants in their preferences. This excludes the meeting for the current session.

As mentioned earlier, the entries in M are non-negative values that represent the desirability of a meeting between the corresponding customer and the exhibitor. An entry $m(c, e)$ is zero if a meeting will not be scheduled between customer c and exhibitor e in this session. Possible reasons are:

- there is no unsatisfied preference for a meeting between c and e;
- a meeting is already planned for customer c before or after the session and c cannot cover the distance between the two corresponding booths in the available time;
- either e or c is busy during the session. This could be caused by a jumbo meeting planned for a previous session or because of an existing meeting in the input;
- the maximum number of meeting sessions c or e are allowed to have is already scheduled;
- the maximum walking time per day for c would be exceeded by planning the meeting. We check this by tentatively inserting the meeting in the customer schedule and computing the walking time;
- the maximum discomfort per day for c would be exceeded by planning the meeting.

The desirability $m(c, e)$ of a meeting depends on the length of the meeting, the percentage of sessions that c and e are lacking in order to have the maximum allowed number of meeting sessions, the number of highly desirable meetings that are already planned for the two participants, the valuation of the preferences and the additional travel time and discomfort caused by scheduling the meeting. The weightings of the optimisation goals are multipliers in this process and thus influence the entries of M. The matrix entries are computed in such a way that the value of a meeting that is not excluded for some reason is always positive.

Complex preferences introduce the problem that scheduling one meeting in a session might make other meetings, that could potentially be scheduled in the same session, undesirable. We call a maximal set S of meetings with the property that scheduling one of the meetings will make all other meetings in S undesirable a *bundle*.

Assume that we have a complex preference $P_1 : \{A, B\} \rightarrow \{C, D, E\}$. As mentioned earlier, this preference is satisfied if either A or B meets one of C, D or E. In this case,

$$S = \{(A, C), (A, D), (A, E), (B, C), (B, D), (B, E)\}$$

is a bundle. It is desirable to schedule only one of the meetings from S, as it will be enough to satisfy the preference P_1. In addition, assume there are preferences $P_2 : D \rightarrow A$ and $P_3 : E \rightarrow B$. Then we would actually have to schedule at least two meetings in S to satisfy P_1, P_2 and P_3.

To simplify the algorithm, we do not allow two meetings in S to be scheduled until P_1 is satisfied. When P_1 is satisfied, S ceases to be a bundle and we can potentially schedule another meeting in S if a different preference that is not yet satisfied (and so also was not satisfied by the meeting that satisfied P_1) is still unsatisfied.

We use Lagrangian relaxation to exclude the scheduling of more than one meeting of a bundle in the same session. In each iteration of the Lagrangian relaxation, we use an implementation of the algorithm described in (Jonker and Volgenant 1987) for solving the corresponding linear assignment problem. Since the algorithm deals with square matrices, we extend M by introducing dummy customers or exhibitors. All entries associated with these dummy participants have value zero. The assignment computed by the algorithm designates exactly one entry in each row and column of M but we only pick the positive entries at these positions as the new meetings of this session.

If the result contains several meetings of a bundle, we update the Lagrangian multipliers for the corresponding matrix entries and re-optimise. This means that the value of a meeting is lower, if the previous solution contained another meeting of the same bundle. The values are updated using the violation of all bundle constraints until there either is at most one meeting chosen from each bundle or when the maximum number of iterations is reached or the lower bound is sufficiently close to the upper bound.

The new meetings force updates of the satisfaction data structure and the meeting data structure. Note that there might be jumbo meetings among the new meetings and so we have to take care to take this into account when we update the data structures.

The run-time of the assignment algorithm is $O(n^3)$ where n is the size of the square assignment matrix. In our case n is the maximum of the number of customers and the sum of all exhibitors and their copies. The Lagrangian relaxation has a constant limit on its iterations. Because the bookkeeping for each session is also in $O(n^3)$, the run-time of the complete system is $O(ln^3)$ where l is the number of sessions of the event.

Computational Results

The first computational results presented in this paper use the input data for ATE 2005. The Australian Tourism Expo (ATE) is a tourism trade event held annually in every Australian capital city. The data set consists of 99 sessions divided into 14 session blocks and four days. A session is 15 minutes and the break between two sessions in the same block is two minutes.

The event has 292 customers and 539 exhibitors. There are 80 meetings already given in the input. The number of preferences contained in the input is 16898 which is about 20 per participant. The data does not contain complex preferences or preferences for jumbo meetings. The maximum number of meetings for one participant is either 49 or 99, depending on the entry fee they paid. Our Java implementation needed 90 seconds CPU-time to solve this problem on a 2.4GHz Pentium.

In our first experiment, we tested the influence of the parameter that determines the balance between customer and exhibitor preferences. This parameter can be chosen between 0 (customer preferences are disregarded) and 1 (exhibitor preferences are disregarded). We kept all other parameters constant and measured for each participant the sum of the weights of unsatisfied preferences, which we call the *discontent* of the participant. For each value of the parameter, we measured the average discontent over all participants, the average discontent of the exhibitors and the average discontent of the customers.

Figure 1 shows the result of these experiments. Since any meeting involves a customer and an exhibitor and the value that both attach to the meeting are summed up to obtain the overall value of the meeting in the meeting matrix, the influence of the parameter is not very strong unless it is either 0 or 1 where the preferences from one of the two participant groups are completely disregarded. In between 0 and 1, the influence of the parameter is small but noticeable and shows the desired behavior: If the balance parameter is small, the customer discontent is higher than if the parameter is large. The exhibitor discontent shows the opposite behavior.

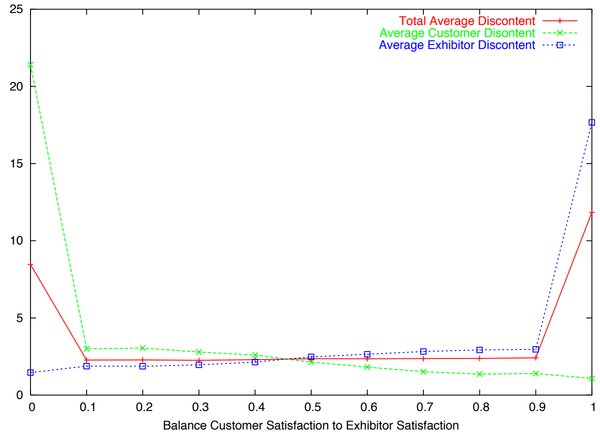

Figure 1: Influence of the balance factor between customers and exhibitors on the discontent of the participants

In the second experiment, we changed the weight given to the goal of reducing the walking time while keeping the parameters for all other goals low and constant. We increased the travel minimisation parameter and measured the average travel time for each customer as well as the average discontent over all participants. The results are shown in Figure 2.

The average walking time during the event (the y-axis on the left is the time walked in seconds) reduces rapidly until the parameter reaches 0.3, then more slowly until 0.5 and fluctuates for higher values of the parameter. This behavior is caused by the fact that the travel distance never makes the value of a prospective meeting negative but only makes meetings that cause less travel more attractive than meetings causing more travel. We set the system up in this way because the main purpose of a trade event is bringing people together and the reduction of travel time is a secondary objective.

For very high values of the travel time minimisation parameter, the heuristic works similar to a nearest insertion heuristic for the travelling salesman problem. This together with the fact that even high travel weights cannot exclude a meeting from being scheduled explains why the average travel time does not diminish smoothly when the weight is increased. The plot also shows that the average discontent is not influenced significantly by the reduction in walking time. Again, the reason is that the travel weight never excludes a meeting from being scheduled.

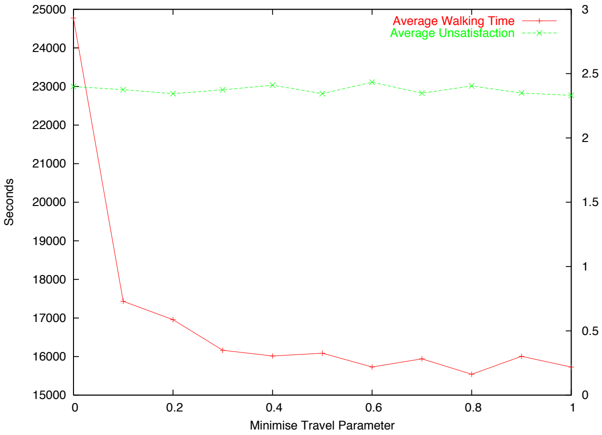

Figure 2: Influence of the travel minimisation factor on the average discontent

The last experiment we conducted tested the effects of the daily walking constraint used to limit the maximum time any participant spends in transit between booths on any day. The plot in Figure 3 shows that allowing more travel per day decreases the average discontent until a minimum discontent level is reached that cannot be diminished by allowing more travel.

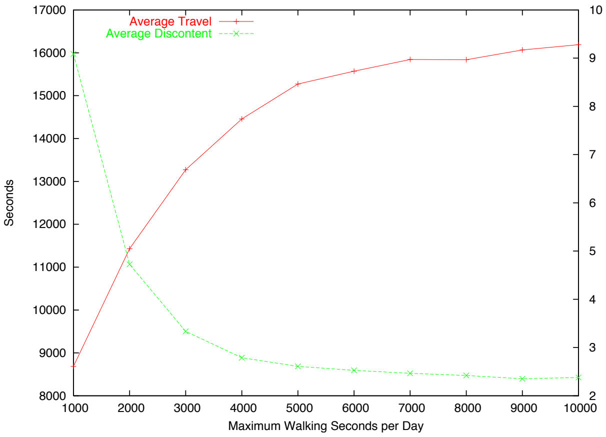

Figure 3: Influence of the daily travel limit on the average discontent

We also constructed artificial problems with complex preferences and jumbo meetings. One example has 100 customers and 100 exhibitors and 105 session blocks of 3 sessions each over 5 days (each session is 10 minutes). There are 20 fixed meetings, 19918 preferences of which 10 are complex. The Requested meeting lengths vary between 1 and 3. Our implementation solves the instance in 24 seconds CPU-time.

Further Enhancements

The meetings generated by the system for customers who can only have 30% or 50% of their total meeting slots filled, were not evenly spread over the whole event. It is important for Tourism Australia's main event ATE that meetings are evenly spread over the whole event and not clustered at certain parts of the event. We added this additional feature to our system which spreads the meetings somewhat evenly across the event. This was achieved by adding dummy exhibitors with a dynamic meeting value for each customer. This meeting value depends on the fraction of allowed meeting slots that the customer has already been allocated. So, if a customer has already a large fraction of his schedule filled then this customer is very likely to have a meeting with the dummy exhibitor. In this way we achieve an even spread of the meetings across the whole event. Figures 4 and 5 show the spread of the meetings without and with this additional feature of spreading the meetings evenly, respectively. Each line corresponds to the schedule of a participant with each column representing a meeting session. A star represents a meeting while a dash represents a session without a meeting. In Figure 4, the meetings are concentrated towards the beginning of the event, while they are spread much more evenly in Figure 5, where we applied our spread strategy described above.

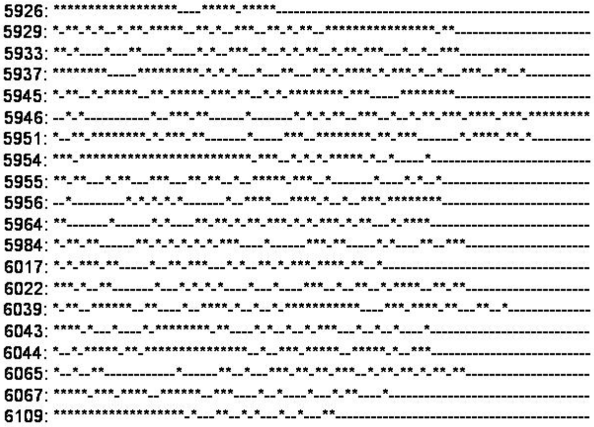

Figure 4: Spread of the meetings without the even spread option

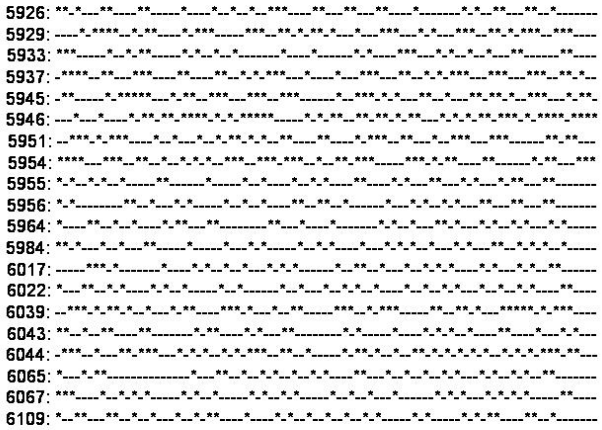

Figure 5: Spread of the meetings with the even spread option

Conclusion

We have presented a system for scheduling meetings at trade events. Since the system will be used to schedule real events, it has to cope with complex requirements. Our system is able to produce solutions that meet all requirements and the run time is short enough to enable the user to try several different combinations of solution parameters. Because the notion of an optimal schedule is not well defined, we expect this approach to be more successful than a system which takes a long time to optimise a fairly arbitrary objective function to near optimality. We have demonstrated empirically that the system is well behaved in trading off several different objective functions and thus gives Tourism Australia useful control in fine-tuning the schedule for their trade events.

References

A. Hertz, V. R. 1998. Constructing a course schedule by solving a series of assignment type problems. *European Journal of Operational ResearchVolume* 108(3):585–603.

Australian Bureau of Statistics. 2005. 5249.0 Australian National Accounts: Tourism Satellite Account. Technical report, Australian Bureau of statistics, `http://www.abs.gov.au`.

Bartholdi, J. J., and McCroan, K. L. 1990. Scheduling interviews for a job fair. *Operations Research* 38(6):951–960.

Cangalovic, M.; Kovacevic-Vujcic, V.; Ivanovic, L.; and Drazic, M. 1998. Modeling and solving a real-life assignment problem at universities. *European Journal of Operational Research* 110(2):223–233.

Ernst, A. T.; Mills, R. G. J.; and Welgama, P. 2003. Scheduling appointments at trade events for the australian tourist commission. *Interfaces* 33(3):12–23.

Frieze, A. M. 1983. Complexity of a 3-dimensional assignment problem. *European Journal for Operations Research* 13:161–164.

Gilbert, K. C., and Hofstra, R. B. 1987. An algorithm for a class of three-dimensional assignment problems arising in scheduling applications. *IIE Transactions* 19(1):29–33.

Jonker, R., and Volgenant, A. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing* 38(4):325–340.