TRACS – TRACTABLE CONFERENCE SCHEDULING

Malvika Gulati, Department of Operations and Decision Technologies, mgulati@indiana.edu
Arijit Sengupta, Department of Information Systems, asengupt@indiana.edu
Kelley School of Business, Indiana University, Bloomington, Indiana, USA

ABSTRACT

Conference scheduling requires presentations scheduled to minimize the conflict between sessions that participants are interested to attend. Sampson [1, 2] provides a solution wherein attendee preference information is used to maximize session attendance and minimize conflicts. We improve Sampson's PBCS by developing a bi-criterion approach to set up two objective functions. One augments Sampson's objective function by incorporating session times as well as reviewer evaluations. The second objective function incorporates only session time and reviewer's evaluations in the scheduling process to maximize session coverage and minimize conflicts. This produces good schedules even in the absence of attendee preference data.

Keywords: Scheduling, Heuristics, Linear Programming, Conflict Resolution

INTRODUCTION

Academic conferences are the premier means for researchers to present and receive feedback on their research from the community. In order for this process to be effective, however, conferences need to be properly scheduled to ensure that attendees are able to attend sessions that they are interested in, and also to ensure that merited papers get scheduled in time slots that are convenient for participants. Creating schedules that suit attendees and presenters alike is a difficult task, and often an unmanageable one. In this paper we provide a general method for conference scheduling, which can be applied to the scheduling of most conferences and workshops, and we show that for conferences with a good paper review process the process of generating good schedules can be quite manageable. We call this process TRACS (Tractable Conference Scheduling). The primary contribution of TRACS is the ability of generating quick, simple schedules without conflicts that maximize attendee participation.

Let us take into consideration the specifics for scheduling a conference and discuss the factors that come into play. Scheduling a conference depends on: (i) the number of days it will run, (ii) the time-blocks that have been allocated, (iii) the number of rooms in which different sessions are being held, and (iv) the presenters, co-presenters, and chairs for each session. Also, like any other scheduling problem when a schedule is designed for a conference it should be catered for the attendees, as the main aim is to maximize the number of attendees attending each session.

LITERATURE REVIEW

We come across some kind of scheduling in our lives on a daily basis; whether it is organizing tasks that need to be completed, or classes that need to be attended for a particular day, chores need to be done around the house, etc. Therefore, there has been a great deal of research done on scheduling and developing algorithms and heuristics to solve such problems on both large and small scales.

Scheduling and timetabling has been the topic of research in both Artificial Intelligence (AI) as well as the Operations community [3]. While research using AI techniques have used simulated annealing [4], Tabu search [5] and Graph Coloring techniques [6, 7], the operations community uses logic programming and its variations to find optimal time schedules [8-11]. Although not much can be found in literature regarding the specific problem of conference scheduling, the problem is quite similar to the problem of university course timetabling, which has been extensively researched. Here we summarize some of the work in the literature on both of these problems.

University timetabling has been a widely researched topic in scheduling literature. The typical problem of university scheduling involves assigning course sections to specific meeting times over the week. The problem of university course scheduling is a well-researched problem, and several solutions have been proposed [10-12]. Conference scheduling involves the placement of conference sessions in appropriate time blocks, ensuring that there are no conflicts for the presenters, when sessions are offered concurrently. The primary restrictions are the same – all sessions must be scheduled *(completeness)* and presenters cannot be scheduled in concurrent sessions *(conflict-free)*. The differences lie in the fewer number of hard restrictions and inter-dependencies between time blocks. The absence of hard constraints and complex inter-dependencies make conference scheduling more tractable, although in real life the problem is complicated by other factors such as time preferences requested by presenters and the presence of "special" sessions such as plenary sessions.

Thompson differentiates between scheduling conferences from a "Presenter-based Perspective (PBP)" to an "Attendee-based Perspective (ABP)", and posits that ABP scheduling can increase participants' ability to attend their favorite sessions [13]. Sampson [1, 2] provides a novel way of using the preferences indicated by users to minimize conflicts and maximize session coverage. In this research, he uses a preference matrix (PREF) for every presentation, aggregated to a preference matrix (SPREF) for every session, and provides a mathematical optimization theory for minimizing conflicts and at the same time maximizing session attendance. However, Sampson points out that a truly optimal solution would have an exponential complexity, and uses simulated annealing techniques to heuristically provide an optimal solution. Sampson shows that post-conference surveys indicate higher user satisfaction, although these could be biased by the fact that the users' preferences were actually used in the process.

Although conference scheduling techniques in the literature have used the shift towards the APB approach in order to increase participants' satisfaction, the process of collecting participants' preferences is a time consuming task. Moreover, the process requires knowing the potential group of attendees in advance. Typically the request for preferences is sent to members of the academic society that organizes the conference, many of whom actually do not end up attending the conference. This results in data that is often inadequate and potentially inaccurate.

FORMULATION

The preference-based conference scheduling (PBCS) [1] problem revolves around scheduling sessions for a conference based on the preferences sent in by the attendees as well as the presenters. We present a solution for this formulation using a bi-criterion approach with two separate but related objective functions, with slightly different goals. The first objective function is based on the one provided by Sampson, and augmented by adding a few new additional constraints to incorporate session times as well as reviewer's evaluation. The second objective function deals with only the session times and the reviewer's evaluation (without attendee preferences), with the assumption that attendee preference data cannot be obtained. For any conference, there are typically some very well-attended sessions, whereas some sessions are highly under-attended. For instance, the 2003 DSI annual meeting attendance record shows an average of 11.38, although the actual attendance of individual non-plenary sessions varied between as low as 4 to as high as 30, clearly showing attendance anomalies based on meeting times. The other aspect which has been incorporated into the first approach is the reviewer's evaluation. In this approach, we propose asking the reviewer one additional question as part of the review, in which he evaluates his view of the "popularity" of the paper on a 5 point Likhert scale.

Bi-Criterion Approach

In the first approach of the formulation we build on Sampson's PBCS model. We add to the reviewer's evaluation a question on the potential "popularity" of the paper, which will then be incorporated in the objective function as a popularity matrix (SPOP$_{p,s}$), which builds on the SPREF matrix by Sampson and incorporates the "popularity" score by the reviewer. Here, we define SPOP$_{p,s}$ to be a summation over all the discussion over all the sessions covered with respect to presenters as well as discussions. To compute SPOP$_{p,s}$ we first compute DPOP$_{p,d}$ which is a weighted sum of all available evaluations for a discussion *d*. The available evaluations include any acceptance or evaluation scale and award nominations. In addition to the standard reviewer comments, we propose that reviewers be asked to evaluate the perceived "popularity" of the presentation on a Likhert 5 point scale (1=lowest, 5=highest). Further, if attendee preferences are available, they are also included. So, we have,

$$DPOP_{p,d} = w_e.EVAL_d + w_b BP_d + DPREF_{p,d} ,$$

where *EVAL* is the available reviewer evaluations, $BP \in \{0,1\}$ indicates whether the discussion *d* was nominated for a best paper award, and *DPREF* is the cumulative discussion preference as in [1]. The weights w_e and w_b are constant weights given to these evaluations, based on perceived importance of reviewer evaluations over attendee data.

Given the above set up, we can now come up with the primary measure of "popularity" of a session (SPOP$_{p,s}$) defined as follows:

$$SPOP_{p,s} = \sum_{d \in DIS_S} DPOP_{p,d}$$

We now define our objective functions and the constraints to satisfy this CLP formulation:

$$\text{Max.} \quad \sum_{p,s,t} W_t (SPOP_{p,s}) E_{p,s,t} \qquad \text{[Eq. 1]}$$

Subject to

$$\sum_s E_{p,s,t} \le 1 \qquad \forall p,t \qquad \text{[Eq. 2]}$$

$$\sum_s E_{p,s,t} \le S_{s,t,r} CAP_r \qquad \forall c,t \qquad \text{[Eq. 3]}$$

$$\sum_{t,r} S_{s,t,r} = 1 \qquad \forall s \qquad \text{[Eq. 4]}$$

$$\sum_{x \in S(p)} S_{s,t,r} \le 1 \qquad \forall p,t \qquad \text{[Eq. 5]}$$

$$E_{p,s,t} \in \{0,1\} \qquad \forall p,s,t \qquad \text{[Eq. 6]}$$

$$S_{s,t,r} \in \{0,1\} \qquad \forall s,t,r \qquad \text{[Eq. 7]}$$

The main objective function aims at maximizing enrollment of a particular session as rated by the reviewer as well as encompassing the time frame. Time preference of attendees is incorporated as W_t which is a weighted matrix for the time-block allocated per session over all the days required for the whole conference. The different entities which we take into consideration are: **People (p)** consisting of presenters, attendees as well as track chairs and quazi presenters; **Discussions (d)** referring to actual presentations; **Sessions (s):** the sessions that are being scheduled (usually the number of discussions per session will vary but for the consideration of this proof we will assume that we have a uniform number of discussions per session); **Time-Blocks (t):** referring to the time period for which that session has been scheduled and **Rooms (r):** the set of rooms.

The given formulation utilizes two sets of variables:

1. **S** variables which are scheduling variables in regards to sessions (*s*), time (*t*) and the room (*r*) to which the session has been assigned.

2. **E** variables which are enrollment variables in regards to which presenter (*p*) is enrolled in which session (*s*) in which time-block (*t*).

The constraints are set up as follows:

Equation 2: Maximizes participant utility.

Equation 3: Allows participants to be enrolled in sessions and within capacity constraints.

Equation 4: Requires each session to be scheduled in exactly one place.

Equation 5: Prevents any presenter from being in more than one place at a time, where s(*p*) is the set of sessions where *p* is the presenter.

Equation 6 and Equation 7: Specify the binary decision variables.

The second approach to the problem looks only at the reviewer's "popularity" scores and the time of the sessions. In this case, we have $RPOP_s$ (a preference matrix) which is a summation over POP_d which incorporates the reviewer's "popularity" scores over discussions *d*. So,

$$RPOP_S = \sum\nolimits_{d \in DIS_S} POP_d \text{ , where } POP_d = w_e.EVAL_d + w_b BP_d$$

Given the above measure of session popularity, we can proceed with the CLP formulation as follows:

$$\text{Max.} \quad \sum\nolimits_{S,t} W_t (RPOP_S) S_{S,t} \qquad \text{[Eq. 8]}$$

Subject to

$$\sum\nolimits_{t,r} S_{S,t,r} = 1 \qquad \forall s \qquad \text{[Eq. 9]}$$

$$\sum\nolimits_{s \in S(p)} S_{s,t} \leq 1 \qquad \forall p, t \qquad \text{[Eq. 10]}$$

$$S_{s,t,r} \in \{0,1\} \qquad \forall s, t, r \qquad \text{[Eq. 11]}$$

In this setting, there is only one set of variables – the *S* variables as before. Description of the above constraint equations follow.

Equation 8: The objective function which incorporates the reviewer's "popularity" score is the matrix $RPOP_s$, W_t is a weighted matrix for the time-block allocated per session over all the days required for the whole conference and $S_{s,t}$ is the session variable for session *s* for a time-block *t*.

Equation 9 and 10: Similar to Eq. 4 and 5 before.

Equation 11: Specifies the binary decision variable *S*.

CLP SOLUTION

A direct solution to the formulation presented here is difficult because of the complexity of the problem. However, we can consider some of the constraints to be hard (i.e., cannot be violated at any cost), and some of the constraints to be soft (violation of such constraints should ideally be avoided) [14]. We have in fact, used this principle in our first formulation, where the constraints in [1] have been revisited for hardness and the soft constraints were removed. A complete CLP-based solution is still not possible for this set of constraints, but a similar simulated annealing technique can be used to create a good schedule. For the second approach, the problem is more tractable, and an algorithmic solution is possible, which is presented next.

Algorithmic solution and Implementation

In the first formulation where user preferences are considered can be solved using the same approach taken by [1] using a simulated annealing technique, where individual minima are computed and iteratively improved on. Sampson shows why the problem of conference scheduling is a conceptually hard problem and will need strategies in order to produce a solution. However, if preference data collection is not possible, then the Sampson PBCS solution cannot be used. The second formulation approach discussed above can still be used with fairly good results. Here we present an algorithm to solve the second objective function, where no preference data is collected, every session has a pre-computed popularity measure, and our objective is to maximize the overall popularity measure. Figure 1 shows the algorithm, which uses a greedy strategy [15] for maximizing the objective function and ensuring that there are no conflicts for presenters at each step. The objective is to maximize the product of the popularity measure for a time block and the popularity measure of the session scheduled at that time block. A greedy approach is highly suitable here, since both popularity measures are positive, and the sum of products of maximum values yields the maximum aggregate. In the case multiple sessions have same popularity, the algorithm tries to spread them across time blocks with high popularity instead of placing them in the same time block in multiple concurrent rooms. The algorithm ensures *completeness* by ensuring all sessions are scheduled, and is *conflict-free* because of the conflict-check before each session is scheduled.

CONTRIBUTION, CONCLUSION AND FUTURE WORK

TRACS improves on Sampson's PBCS method by including reviewer evaluations as well as session timings. The GreedyTracs algorithm presented here can be easily shown to be complete and conflict-free. However, whether or not it produces a good schedule should be empirically determined, although several arguments can be made towards the quality of schedules generated by TRACS. First, it can be argued that the reviewers are typically the only people to read papers completely before publication, and therefore can judge a paper in its entirety. In addition, we posit that papers that are recommended for a best paper award, or are given remarkably good evaluations from the reviewers deserve more attention from conference participants. Moreover, timing of sessions play a significant role in participant attendance, since late sessions and last days of conferences are often ill-attended. The above factors view the TRACS approach in a different light as compared to PBCS, and make the TRACS approach highly suitable for generating quick conference schedules inclusive of the reviewers "popularity" of the papers. We are in the process of implementing the algorithm and incorporating it into the new Conference Information System (CIS) currently under development for the Decision Sciences Institute.

```
Algorithm GreedyTracs (s: Array of Sessions, t: Array of TimeBlocks,
                       r: Array of Rooms, W: Array of popularity for each t,
                       RPOP: Array of combined popularity for each s) {
        Create Array Sch(s,t,r) = 0 for all s, t, r;
    /* Step 1 – Sort inputs with highest values up front – the greedy strategy */
        Sort W in descending order of popularity value;
        Sort RPOP in descending order of popularity value;
        Create Queue Q;     /* Queue for temporary unscheduled sessions */
    /* Step 2 – Main algorithm stage for matching
        While (All sessions are not scheduled) {
                TR = subset of RPOP with highest RPOP(s) values;
                TW = subset of W with highest W(t) values;
                for each ct in TW {
                  if TR is empty then break;
                  remove first st from TR;
                  if (conflicts(st, ct, Sch))
                  then add st to Q;
                  else {
                    Pick rt from r which is available at time ct with highest capacity;
                    Set Sch(st, ct, rt) = 1;
                    Remove st from TR and RPOP
                  }
                  Remove all s in Q and add back to TR;
                }
        }
      Return Sched as output.
}
```

Figure 1. The "Greedy" TRACS Algorithm. Note that all arrays in this algorithm are associative arrays – actual implementation can use Hashtables.

REFERENCES

1. Sampson, S. *DSI 2001 Preference-Based Scheduling Part I: PBCS Intruduction and Data Collection*. in *Decision Sciences Institute Annual Meeting (DSI '2002)*. 2002.
2. Sampson, S. *DSI 2001 Preference-Based Scheduling Part II: Session Composition and Timetabling*. in *Decision Science Institute Annual Meeting (DSI '2002)*. 2002.
3. Schaerf, A., *A survey of automated timetabling*. 1995, Computer Science Department of Software Technology.
4. Saleh Elmohamed, M.A., P. Coddington, and G. Fox, *A Comparison of Annealing Techniques for Academic Course Scheduling.* Lecture Notes in Computer Science, 1998. **1408**: p. 92-114.
5. Costa, D., *A Tabu Search Algorithm for Computing an Operational Timetable.* European Journal of Operational Research, 1994. **76**(1): p. 98-101.
6. Lotfi, V. and S. Sarin, *A Graph Coloring Algorithm for Large Scale Scheduling Problems.* Computers & Operations Research, 1986. **13**(1): p. 27-32.
7. Mehta, N.K., *The Application of a Graph Coloring Method to an Examination Scheduling Problem.* Interfaces, 1981. **11**(5): p. 57-64.
8. Frangouli, H., V. Harmandas, and P. Stamatopoulos. *UTSE: Construction of Optimum Timetables for University Courses - A CLP Based Approach*. in *3rd International Conference on the Practical Applications of Prolog (PAP'95)*. 1995. Paris.
9. Caseau, Y. and F. Laburthe. *Improved CLP Scheduling with Task Intervals*. in *11th International Conference on Logic Programming, ICLP '94*. 1994: The MIT Press.
10. Goltz, H. and D. Matzke, *University Timetabling Using Constraint Logic Programming.* Lecture Notes in Computer Science, 1999. **1551**: p. 320-334.
11. Rudova, H. and L. Matyska. *Constraint-based timetabling with student schedules*. in *PATAT 2000 - 3rd International Conference on the Practice and Theory of Automated Timetabling*. 2000.
12. Mooney, E., R.L. Rardin, and W.J. Parmenter, *Large-scale classroom scheduling.* IIE Transactions, 1996. **28**: p. 369-378.
13. Thompson, G.M., *Improving conferences through session scheduling.* Cornell Hotel and Restaurant Administration Quarterly, 2002. **43**(3): p. 71-76.
14. Schiex, T. *Possiblilistic Constraint Satisfaction Problems or "How to Handle Soft Constraints?"* in *Proceedings of the Eighth Internation Conference on Uncertainty in Artificial Intelligence*. 1992. Stanford, CA.
15. Curtis, S.A., *The classification of greedy algorithms.* Science of Computer Programming, 2003. **49**(1-3): p. 125-157.