Improving Conferences through Session Scheduling

A computer program that schedules a conference's concurrent sessions can enhance participants' conference experience—and gives hoteliers an opportunity to offer (and maybe charge for) a new service.

BY GARY M. THOMPSON

Meetings and conferences are an essential part of many hotels' business mix. The conferences on which this paper focuses are those with a variety of sessions occurring simultaneously, from which participants select the particular sessions they wish to attend. Examples are conferences organized for educational or informational purposes, including academic conferences. Anecdotal evidence from meeting planners suggests that most conferences have at least some component whereby participants can select their sessions of interest. Invariably, conference participants complain that at some times there are several sessions of particular interest scheduled simultaneously, while at other times there are no sessions scheduled in which they are interested.

Scheduling options. Historically, conferences have been scheduled from what one might call a presenter-based perspective (PBP). With a PBP, the conference is scheduled to avoid time conflicts and meet time preferences for the session presenters. Although a PBP uses no direct input from the conference participants, meeting planners try to improve satisfaction by scheduling sessions on related topics so that they do not conflict. At the same time, many conference participants exhibit eclectic tastes and choose to attend sessions on a variety of topics.

A different approach to conference scheduling is offered by an attender-based perspective (ABP). With an ABP, the meeting planner attempts to schedule the conference while considering the participants' session preferences. Indeed, the objective of scheduling the conference becomes maximizing the participants' satisfaction. For such an approach to be possible, however, the meeting planner must be able to obtain session-preference information from the registrants. I return to this issue later.

Conference scheduling is often performed just once by an individual (or a team) in the organization that's planning the conference, even when the conference occurs regularly (e.g., annually). For the next event, another person or team takes over the scheduling chore. The nature of conference scheduling limits the extent of institutional learning that occurs and certainly would seem to curtail the ability to implement an ABP. To the best of my knowledge, hospitality organizations simply provide the facilities for conferences—I am not aware of any conference centers or hotels that actually schedule the conference's sessions for their customers.

Conference participants spend substantial money and time attending the event, and their satisfaction is important in gauging the meeting's success. My aim in this paper is to show that conference scheduling that uses an ABP approach can increase participants' ability to attend their preferred sessions. Specifically, I present the results of two experiments in conference scheduling that were designed to compare the effectiveness of manual and computerized ABP conference-scheduling techniques to the effectiveness of conference scheduling using a PBP.

Scheduling Conferences Using a Computer

The literature on conference scheduling is scant. Eglese and Rand were the first to examine the problem.[1] They developed a computerized tool and focused on its application in a single conference. In contrast to their analysis, however, I consider meeting rooms to have finite capacities. Sampson and Weiss conducted an investigation into some of the conference-design issues affecting participant satisfaction, but they used limited assumptions about room availability and capacity.[2] Compared to their approach, however, I allow meeting rooms to have different capacities and anticipate that not all rooms are available at all times. I also require that presenters be available for all duplicate (repeated) offerings of the same session.

Because of the lack of existing tools that address the level of conference-scheduling complexity that I wanted to study, I developed a specialized computer algorithm, called a heuristic algorithm, to schedule conferences using an ABP.[3] A heuristic algorithm is a procedure that attempts to find acceptable solutions to a problem quickly. A heuristic algorithm may in fact find the best solution to a problem, although the algorithm will not be able to verify that solution as being the best. Heuristic algorithms are useful in complex combinatorial problems such as conference scheduling, where the complexity of the problem limits one's ability to find an optimal (best possible) solution within a reasonable amount of computational effort.

Experiments in Conference Scheduling

I conducted two experiments to investigate the effectiveness of various approaches to conference scheduling. I describe each below.

Experiment 1. I used actual data from a real conference to construct three artificial (hypothetical) conferences having similar attributes. I refer to my conference scenarios as 1A, 1B, and 1C. Each conference had 12 sessions, which were to be scheduled at three meeting times in four different rooms. Since the number of time-and-room combinations equaled the number of sessions, all rooms would be occupied during each meeting period. In other words, four sessions would be occurring simultaneously at each meeting time. Each artificial conference had 100 participants and 10 presenters. Eight of the presenters were assigned to present one session each, while two presenters were responsible for two sessions each. As such, the presenters averaged 1.2 sessions. The participants had specified at least three and as many as six preferred sessions, and averaged 5.3 preferred sessions. When evaluating the quality of schedules, I allocated six

[1] R.W. Eglese and G.K. Rand, "Conference Seminar Timetabling," *Journal of the Operational Research Society,* Vol. 38 (1987), pp. 591–598.

[2] S.E. Sampson and E.N. Weiss, "Designing Conferences to Improve Resource Utilization and Participant Satisfaction," *Journal of the Operational Research Society,* Vol. 47 (1995), pp. 297–314.

[3] Gary M. Thompson, "Conference Scheduling to Maximize Attendee Satisfaction," 2001 (unpublished paper).

points if the participant was able to attend her first choice, five points for her second choice, and so on up to the sixth choice (one point). With 100 participants, the maximum possible schedule score would be 1,500 points if all of the participants were able to attend their top-three choices.

The four available meeting rooms had capacities of 60, 60, 80, and 300 seats. Since even the smallest room could accommodate the session having the greatest number of participants desiring to attend it, the three experimental conferences in Experiment 1 were not constrained by room capacity. I expected that the absence of room-capacity constraints would make these conferences relatively easy to schedule manually, since the room assignments can be safely ignored when the schedule is being developed.

I scheduled the three conferences in Experiment 1 in three ways: manually, randomly, and with the conference-scheduling heuristic algorithm described earlier. My goal in having the schedules developed manually was to mimic the manual scheduling that frequently occurs for conferences. To do this, I asked 22 Cornell University students in a master's level course in service-operations management to develop the schedules. The students were assigned the task of scheduling one of the three conferences selected at random (seven students scheduled the first conference, seven the second, and eight the third). None of the students knew anything about the other students' assignments. Each student was asked to develop the best possible schedule, using a method of her or his own choosing. (They were not given access to the heuristic algorithm, however.) They were directed to identify the time and room assignments for each of the 12 conference sessions, but they were not required to identify which participants would attend which session(s) due to the effort involved in making the manual assignments. Since the students were provided with the participants' session-preference information, the students would be able to develop their schedule with regard to this information and so the manually developed schedules would implement an ABP.

In randomly developing the conference schedules, I arbitrarily selected a time and room for each session and then checked to ensure that there were no time conflicts among the presenters. In the event of such a time conflict, I repeated the process. Thus, my randomly generated schedules took a PBP approach to conference scheduling.

Since neither the manual approach to conference scheduling nor the randomly generated schedules actually assigned participants to sessions, I took those schedules and ran them through the portion of my heuristic algorithm that assigns participants to sessions (as described above).

The Heuristic Algorithm

It is beyond the scope of this article to present complete details on the heuristic algorithm that I developed for conference-scheduling problems, its general functioning is as follows. It begins with no sessions scheduled. It then repetitively adds one session at a time to the conference. Each time a session is added to the conference, participants who are not presenting at the time are assigned to the session, in order of descending interest in the session, until the room capacity is reached or all interested participants have been assigned to the session. If a conference participant is already assigned to a session that is scheduled at the same time as another session that is, in fact, of greater interest to that person, then she is assigned to the newly scheduled session. The algorithm then searches for a participant who would like to take the space that has been freed up in the old session.

Once an initial schedule has been constructed, the heuristic algorithm then repetitively perturbs the schedule by dropping and re-adding sessions and compares the new schedule to the previous one. The heuristic algorithm uses simulated annealing to determine whether the new schedule should be adopted.[1] If a heuristic only adopted solutions (schedules) that were better, then it eventually would arrive at a solution that could not be improved on using the rules of the heuristic. This schedule would typically not be the best possible schedule, however. The logic of simulated annealing, which mimics the annealing process that occurs when liquid compounds (such as metals) are cooled, sometimes adopts inferior schedules. This reduces the likelihood of getting "stuck" at an inferior schedule and, consequently, it tends to find very good, if not optimal, solutions. Simulated annealing has been successfully applied to a variety of complex optimization problems.—*G.M.T.*

[1] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220 (1983), pp. 671–683.

EXHIBIT 1

Experimental results

Conference	Experiment 1[a]			Experiment 2[b]
	1A	1B	1C	2
Random average[c]	1293.2	1305.8	1283.4	795.6
Random std. dev.[c]	10.33	19.79	21.07	12.07
Manual average	1295.86[d]	1289.57[d]	1273.88[e]	N/A
Manual std. dev.	20.49[d]	10.44[d]	27.06[e]	N/A
Heuristic average[a]	1350.0	1345.0	1322.2	922.4
Heuristic std. dev.[a]	11.62	0.00	4.38	9.34

Notes:

[a] As explained in the text, scores for Experiment 1 are obtained by assigning 6 points for each participant who is able to attend her highest ranked session, 5 points for each participant who is able to attend her second-ranked session, and so on down to 1 point for each participant who is able to attend her sixth-ranked session.

[b] Scores for Experiment 2 are obtained by assigning one point for each preferred session that a participant is able to attend.

[c] Sample size = 5 schedules

[d] Sample size = 7 schedules

[e] Sample size = 8 schedules

Finally, the third approach to developing schedules in Experiment 1 was to use the heuristic algorithm I developed, which uses an ABP model.

Experiment 2. Experiment 2 uses the actual data from a different conference. This conference had 64 sessions, which were to be scheduled at eight meeting times in eight distinct rooms. Of the sessions, 32 were offered only once, 13 were offered twice, and 2 were offered three times (for a total of 47 unduplicated sessions). The conference had a total of 175 presenters and participants. The presenters in this conference, who were presenting between one and five different sessions (averaging 2.03 sessions), wished to attend sessions when they were not presenting. The participants had each specified a desire to attend between zero and eight sessions and averaged 6.02 preferred sessions. Participants did not rank order their session preferences for this conference, so to score the schedules I allocated one point for each desired session that a participant was able to attend.

The available rooms for this conference ranged in size from 50 to 165 seats, averaging 81.25 seats. In contrast to Experiment 1, Experiment 2 experienced room-capacity constraints, since the smallest room was not large enough to hold the most-popular session.

I scheduled the Experiment-2 conference using two of the three methods used from Experiment 1: randomly and using the conference-scheduling heuristic algorithm. I implemented the heuristic algorithm in the same way as I did for Experiment 1. However, my initial method of generating random schedules developed 1,000,000 schedules without finding one that avoided time conflicts for the presenters. The time conflicts arose because a number of sessions were offered two or three times and because of the generally high number of presentations made by the presenters. I consequently modified the random-schedule generator to consider only times for sessions that would not result in presenter time conflicts.

Results

Exhibit 1 summarizes the results from both experiments. A number of the initial schedules developed by the students in Experiment 1 were not feasible, due to presenter or room-time conflicts. In those cases I asked the students to develop a second (and in one case a third) schedule.

I used pooled *t*-tests, assuming unequal variance in the samples, to compare the results from the different techniques within each experiment. To adjust for the differences among the three conferences in Experiment 1, from the raw satisfaction score for each case (i.e., Experiment 1A, 1B, and 1C) I subtracted the mean satisfaction score for that conference. That test indicated that the difference between manually generated and randomly generated schedules was not statistically significant. The tests also indicated that the difference between the randomly (or manually) generated schedules and the schedules generated using the heuristic algorithm was statistically significant at the 0.001 level for Experiment 1.

Exhibit 2 summarizes the improvement in participant satisfaction resulting from the schedules generated by the heuristic algorithm, as compared to the manually and randomly generated schedules. For Experiment 1, the schedules generated using the heuristic algorithm yielded between 3.0- and 4.4-percent higher participant satisfaction than randomly generated schedules, and between 3.8- and 4.3-percent higher participant satisfaction than manually generated schedules. For Experiment 2, the schedules generated using the heuristic algorithm increased participant satisfaction by 15.9 percent compared to randomly generated schedules.

EXHIBIT 2

Heuristic algorithm's improvement in participant satisfaction

	Experiment 1			Experiment 2
Conference	**1A**	**1B**	**1C**	**2**
Percentage improvement over randomly generated schedules	4.39	3.00	3.02	15.93
Percentage improvement over manually generated schedules	4.18	4.30	3.79	N/A

Implications for Hospitality Managers

My results have a number of implications for hospitality managers. First, the results have shown the difficulty associated with manually scheduling a conference. The heuristic software generated superior schedules in a few minutes on a computer, while the most-successful manual schedulers reported spending a number of hours on their scheduling efforts. Even for relatively small conferences with 100 participants and no room-capacity constraints, the quantitatively sophisticated individuals in this study (the master's degree students) generated schedules that were statistically indistinguishable from randomly generated schedules. One might argue that those students were inexperienced with conference scheduling. That is undeniable. However, their lack of experience mimics the situation in organizations where conference scheduling is often a one-time or an occasional task. Since I am unaware of any place where people are taught how to schedule conferences manually, the continued manual scheduling of conferences will perpetuate the less-than-ideal levels of satisfaction that participants often experience with conference schedules.

Second, the results show that some conferences are easier to schedule than others. Because of the difference in participant satisfaction for randomly generated schedules and those generated using the heuristic algorithm, I feel safe in saying that my conference in Experiment 2 was more difficult to schedule than the three conferences in Experiment 1. I believe the increased difficulty with the conference in Experiment 2 arose because of its room-capacity constraints and because its presenters—who were presenting in more sessions—had more time constraints. An investigation into the characteristics of conferences that make them "hard to schedule" would seem warranted. I expect that the relative advantage of the heuristic algorithm will be apparent as scheduling complexity increases.

Third, clearly there are benefits to managers of conference centers and hotels from improved conference scheduling. As far as I am aware, however, at present there are no hospitality organizations offering this service to their customers. As such, the conference centers and hotels that first offer this service may gain a competitive advantage, whether they offer the service on a fee-based or free basis. Historically, there were two barriers to offering such services—collecting the data and using them. Collecting data about conference participants' session preferences would have been problematic just a few years ago, but the internet now provides the ideal means of accomplishing this. For example, participant-preference information can be collected via a web-based interface, perhaps as part of the on-line conference registration process.[4] Having the data is not enough, however, as evidenced by the results of my manual schedulers, who had access to preference data but who were unable to make effective use of them. Sophisticated tools like the heuristic algorithm can take the participants' preferences and make good use of that information.

Fourth, although it is beyond the scope of this investigation, I see a potential link between conference scheduling and enhancing function-space revenue management.

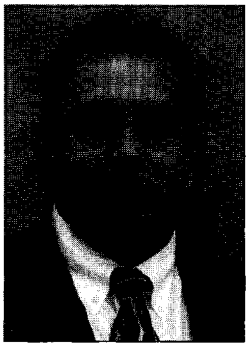
Given that it seems beneficial for conference centers and hotels to offer some conference-scheduling services, next up is the question of how to proceed. Should hotels develop their own proprietary software or should they make use of third-party scheduling services? At present, I do not know the answer to that question. However, organizations wishing to develop proprietary solutions may be aided by papers published on the topic.[5] I also note that a search on the internet[6] identified just one third-party provider of conference-scheduling services, namely, my own business venture called SchedulExpert.[7] ■

[4] Indeed, companies that provide on-line registration would be ideally suited to offer conference scheduling as an ancillary service.

[5] See: Eglese and Rand; and Sampson and Weiss.

[6] A search at google.com using the search term "conference scheduling service."

[7] That is, a software firm of which I am founder and CEO, SchedulExpert, Inc., at schedulexpert.com.



Gary M. Thompson, Ph.D., is an associate professor of operations management at the Cornell University School of Hotel Administration (gmt1@cornell.edu).

refereed article: submitted on January 2, 2002; accepted with minor revisions on February 14, 2002.