Scheduling school meetings

Franca Rinaldi and Paolo Serafini

University of Udine, Department of Mathematics and Computer Science
{rinaldi, serafini}@dimi.uniud.it

We address the following problem: on certain days of the school year parents can meet teachers to discuss about their children. Each parent tries to meet a selected subset of teachers and the meetings are individual. Since in most schools there is no advance planning, parents wait in lines for a long time (one line for each teacher), not only wasting time but also preventing the possibility to meet several teachers.

In this paper we propose a planning method in order to guarantee that each parent meets all required teachers and the wasted time is minimized. A prerequisite for the method to work is that all meetings must last a fixed amount of time, we call *time slot*. Practically, this constraint can be relaxed if a meeting is shorter.

We first consider the problem of minimizing the total time needed for the meetings. This problem can be modeled as a minimum makespan open-shop problem: there is a set I of teachers (machines), a set J of parents (jobs) and each parent j wants to meet a specified subset I_j of teachers (each meeting is an operation). The processing times are $a_{ij} = 1$ if parent j wants to meet teacher i and $a_{ij} = 0$ otherwise.

This particular instance of the open shop problem is polynomial and can be easily solved by means of the algorithm by Gonzalez and Sahni [2] for the open-shop problem with preemption. The minimum makespan can be computed as follows. For each $i \in I$, denote by $b_i := |\{j \in J : a_{ij} = 1\}|$ the number of parents that want to meet teacher i. Similarly, for each $j \in J$, let $c_j := |I_j|$ be the number of meetings required by parent j. It turns out that $\alpha := \max\{\max_{i\in I} b_i, \max_{j\in J} c_j\}$ is the minimum makespan, and the optimal schedule is computed by solving a sequence of α bipartite matching problems, one for each time slot.

Each matching problem assigns parents to teachers. Some teachers and/or some parents may not be assigned in a particular time slot, which, in this case, is called an *idle time* for the teacher and/or the parent. Although the concept of idleness refers in general to both teachers and parents, we consider only idle times of parents as a measure of wasted time. Furthermore, idle times of one particular parent occurring either before or after his/her meetings do not really count as a wasted time. Therefore, from now on we will use the term idle times *only* for the parent idle times in between meetings.

More in detail, the algorithm by Gonzalez and Sahni uses the quantities $\text{slack}(i) = \alpha - b_i$ and $\text{slack}(j) = \alpha - c_j$, which are dynamically updated at each iteration. Elements with zero slack are called *critical* and must be assigned. So

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 484–486. ISBN 80-210-3726-1.

an assignment of parents to teachers is computed under the only requirement that critical teachers and critical parents must be assigned.

The structure of the above algorithm leaves some room to take into account the lexicographically second objective function of the meeting problem, that is minimizing the total number of idle times. The problem of minimizing the number of idle times within a fixed makespan is NP-hard, as can be seen by transforming the no-wait open shop problem $O^g|\text{no-wait}, p_{ij} \in \{0,1\}|C_{\max}$ [1]. Being the problem NP-hard, we can approach its solution heuristically.

Our first strategy consists in solving, at each step, a max weight matching problem instead of a feasible matching problem. To this aim let $s_k(j) \in \{0,1,2\}$ be the state of parent j at time k, where the meaning is that $s_k(j) = 0$ if no meeting has been assigned to parent j up to time slot $(k-1)$ (included), $s_k(j) = 1$ if some, but not all, meetings have been assigned, and $s_k(j) = 2$ if all meetings have been assigned. Let $W_k(j)$ be the number of idle times assigned to parent j during time slots $\{1, \ldots, k-1\}$ by the first $(k-1)$ matching problems. Then each pair (i,j) receives the weight

$$w_{ij} := \begin{cases} 0 & \text{if } s_k(j) = 0 \\ (1 + W_k(j))^2 & \text{if } s_k(j) = 1 \end{cases} \tag{1}$$

in the k-th matching problem (note that if $s_k(j) = 2$ parent j is not considered in the matching problem). Hence, until a parent has not been assigned, his/her meetings receive a zero weight. As soon as a meeting for the parent is assigned, the weight rises to one and then increases quadratically with the number of assigned idle times.

With the cost function (1) the procedure tends to schedule the major part of the meetings in the last slots, when all parents and teachers become critic. This introduces inevitable idle times. In order to overcome this behaviour, we have observed that it is useful to modify the cost function by randomly generating the meeting costs so that parents still to be assigned might be encouraged to be assigned a meeting. So we redefine (1) as

$$w_{ij} := \begin{cases} \left.\begin{array}{ll} -1 & \text{with probability } p_1 \\ 0 & \text{with probability } p_2 \\ 1 & \text{with probability } 1 - p_1 - p_2 \end{array}\right\} & \text{if } s_k(j) = 0 \\ c\,(1 + W_k(j))^2 & \text{if } s_k(j) = 1 \end{cases} \tag{2}$$

where the probabilities p_1 and p_2 and the coefficient c must be properly tuned. A multirun use of this random algorithm has led to remarkable improvements.

However, the solution found this way may have some idle times. In order to improve it we adopt the following local search procedures that modify the schedule of a single teacher and a single parent, respectively. The first one considers only exchanges between meetings of a same teacher. Given teacher i, define the weighted directed graph $G(i)$ having α nodes and, for each $1 \le t_1, t_2 \le \alpha$, an arc (t_1, t_2). This arc is assigned a negative (positive) weight equal to the number

of idle times deleted (introduced) by moving the meeting of i currently scheduled at t_1 to time t_2. Nodes corresponding to idle times of teacher i are called *sink* nodes. Apparently, if there exists a directed cycle in $G(i)$ of negative cost, the corresponding sequence of switches in the meetings of i decreases the global wasted time by the cost of the cycle. Detecting a negative cycle is easy. If there are no negative cycles in $G(i)$ and there are negative paths ending in a sink node, then we switch the meetings of i according to the corresponding path. The search for such cycles and paths can be repeated, after the necessary updating, until no one exists.

With a symmetric approach, the second procedure tries to eliminate idle times of parent j by performing local exchanges only in the meetings of j. To this aim, define the graph $H(j)$ having α nodes and an arc (t_1, t_2) whenever the teacher i met by j at t_1 is idle at time t_2. In this case, the meeting (i, j) can be moved from t_1 to t_2. As a consequence, if there exists a directed path in $H(j)$ from either the first or the last active time of j to one of her/his idle times, the corresponding sequence of exchanges decreases the wasted time of j by 1.

Finally we may also consider to resequence the meetings in order to decrease the number of idle times. This procedure works effectively mainly for solutions with many idle times. We build a complete graph with nodes corresponding to matchings. Any Hamiltonian path corresponds to a sequence of matchings. If we assign to each pair of matchings a cost given by the number of adjacent time slots for the parents, the cost of an Hamiltonian path is equal to the number of all meetings minus the number of all parents minus the number of idle time blocks. An idle time block is a maximal set of adjacent idle times. Since the first two quantities are invariant, a maximum Hamiltonian path provides a matching sequence with the minimum number of idle time blocks. This is however different from minimizing the number of idle times.

References

1. Gonzalez, T. Unit Execution Time Shop Problems *Math. Oper. Res.* **7** (1982), 57–66
2. Gonzalez, T. and S. Sahni Open Shop Scheduling to Minimize Finish Time *Journal of the ACM*, **23** (1976), 665-679