LINEAR ORDER CONGRUENCES AND PARALLEL SHEDULING PROBLEMS WITH PRECEDENCE CONSTRAINTS

Szilvia Szilágyi
University of Miskolc
Hungary

Attila Körei
University of Miskolc
Hungary

Ingrida Vaičiulytė
Šiauliai State University of Applied Sciences
Lithuania

Audronė Rimkevičienė
Šiauliai State University of Applied Sciences
Lithuania

Annotation

Scheduling problems can be alternately approached by methods based on discrete mathematical tools. One obtains a model of a given production technology by representing the relations of the phases of production as partial ordered sets. To solve the scheduling problem in such a case one basically needs a linear order. As an important result of the theory of partial orders we can cite the Szpilrajn theorem, stating that each partial order can be extended to a linear order. Studying the order congruences of partially ordered sets it became clear that the minimal linear order congruences can be successfully applied in investigating scheduling problems. An algorithm for finding minimal linear order congruences is introduced, and a partition of jobs obtained which carries important information about the solution of the parallel scheduling problem.

Keywords: poset, topological sorting, order congruence, parallel scheduling.

1. Introduction

Planning and scheduling are forms of decision-making that play an important role in most manufacturing and service industries (Pinedo, 2005). Partially ordered sets occur widely in computation, in sorting and even in scheduling. For some years research on these themes has focused first on combinatorial optimization. Because of its special place in the landscape of the mathematical sciences order is especially sensitive to new trends and developments. The most important applications of partial orders come from theoretical computer science. Partial orders and linear orders also often occur in optimization problems. Scheduling, sorting and searching problems are among the most common instances of order. Typically, an order must be transformed to another, say a partial extension or a linear extension, which itself may represent a schedule or a sort. In a scheduling problem we have to find an optimal schedule of jobs. Depending on the number of the available machines and the given objective function, different approaches could be applied to find the optimal scheduling. In this paper we present the basic relationship between the notions originating from the theory of partial orders and scheduling problems with precedence constraints.

2. Preliminary definitions and results

We consider a *partially ordered set* or *poset* **P** $=(X,\leq_P)$, where X is a set and $\leq_P$ is a reflexive, antisymmetric and transitive binary relation on X. We call X the ground set and $\leq_P$ a partial order on X. Elements of the set X are also called *points*, and the poset is *finite* if the set X is finite. Let **P** $=(X,\leq_P)$ be a poset and take $x,y\in X$ with $x\neq y$. We say that x and y are *comparable* if either $x\leq_P y$ in or $y\leq_P x$. On the other hand, x and y are called *incomparable*, if neither $x\leq_P y$ nor $y\leq_P x$. If any two points of X are comparable in the poset **P** $=(X,\leq_P)$, we call $\leq_P$ a *linear* or *total order* on X.

A *chain* in the poset **P** $=(X,\leq_P)$, is a subset $C\subseteq X$, such that any two elements in C are comparable. The cardinality of a chain is called the *length* of the chain. A chain is said to be *maximal* if it is not a real subset of any other chain. If a poset has no infinite chains, then the length of the maximal chain is called the *height* of the poset. An *antichain* in the poset **P** $=(X, \leq_P)$, is a subset $A \subseteq X$, such that any two elements in A are incomparable.

We say that y *covers* x, if $x \leq_p y$ and there is no z such that $x \leq_p z \leq_p y$. We shall use the notation $x <_P y$ if $x \leq_P y$ and $x \neq y$. A point $x \in X$ is called a minimal point if there is no point $y \in X$ with the property $x >_P y$ in *P*. Similarly, a point $x \in X$ is called a maximal point if there is no point $y \in X$ with $x <_P y$ in *P*. It is easy to see, that a maximal chain must be contain one of the maximal points and one of the minimal points as well.

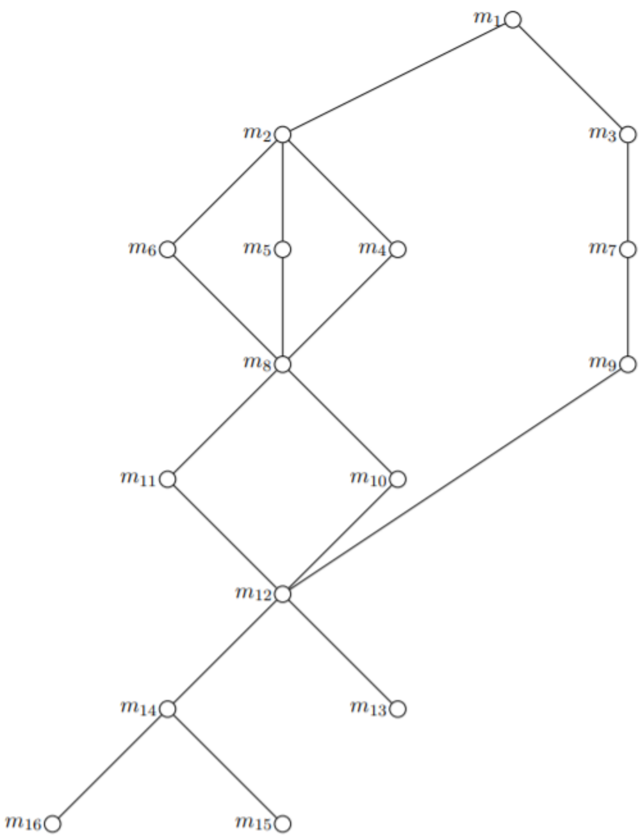
If **P** $=(X, \leq_P)$ and **Q** $=(Y, \leq_Q)$ are partially ordered sets, the function f is called *isotone* or *order-preserving* if $x_1 \leq_P x_2$ implies $f(x_1) \leq_Q f(x_2)$ for all $x_1, x_2 \in X$. For a detailed study of the theory of partially ordered sets we refer Davey and Priestley (2002) and Trotter (2002).

Theorem 2.1. (Szpilrajn) Any partial order has a linear extension (Szpilrajn, 1930).

This theorem says that any partial order can be extended in such a way that every two points becomes comparable, and it saves the original ordering relations. The linear extensions of a partially ordered set are often referred as the *topological sorts* of the poset. The two common topological sorting algorithms are Kahn's algorithm and a modified depth-first search, for other methods see Tella et al. (2014).

A finite partially ordered set $(M, \leq)$ can be represented by its Hasse diagram in the form of drawing of its transitive reduction. It means that each element of M is represented as a vertex in the plane and a line segment or a curve is drawn that goes upward from x to y whenever y covers x . So only the direct dependencies are presented, we omit the loops due to the reflexivity and the edges due to the transitivity.

Example 2.2. The Hasse diagram of a partially ordered set with 16 points.



The presented partially ordered set has 3 minimal and 1 maximal element. The height of the poset is 8, because the common cardinality of its maximal chains is 8.

Two of the linear extensions of the poset are:

$$L_1 = \{m_{16}, m_{15}, m_{14}, m_{13}, m_{12}, m_{11}, m_{10}, m_8, m_6, m_9, m_7, m_3, m_5, m_4, m_2, m_1\}$$

and

$$L_2 = \{m_{16}, m_{15}, m_{14}, m_{13}, m_{12}, m_9, m_{10}, m_{11}, m_7, m_8, m_6, m_5, m_4, m_2, m_3, m_1\}.$$

3. Scheduling problems with precedence constraints

Scheduling problems can be precisely described in terms of machines, constraints and objective functions. These problems are generally represented by a 3-field notation introduced in Graham et al. (1979). The first field of the triplet $\alpha|\beta|\gamma$ is about the machines, where 1 denotes the single and P denotes the parallel identical machines environment. In case of Pm the number of machines m is considered as a constant. In the β-field the constraints regarding the jobs are given, for example the precedence and preemption relations, the completion and processing times, deadlines etc. The γ-field describes the different types of the objective functions which may be the largest or the total completion time, the maximum lateness, the number of tardy jobs and so on. In this paper we mainly focus on the problem $P|prec;p_i = 1|C_{max}$ where we have jobs with unit processing times with precedence constraints, the jobs are to be scheduled on parallel identical machines such that the makespan (the total completion time) is minimized.

Suppose that $M = \{m_1, m_2, \ldots, m_n\}$ is a finite set of jobs or tasks. Let $m_i \leq m_j$, $(i, j \in \{1,2, \ldots, n\})$ denote the fact that job m_i precedes job m_j in time. Obviously, the pair $(M, \leq)$ is a partially ordered set and the direct precedencies of the jobs can be represented by a Hasse diagram. Consider again the poset given in Example 2.2, and suppose that its points correspond to jobs, and its relations define precedencies between the jobs. If job m_i precedes job m_j $(m_i < m_j)$ in time, then the node representing job m_i is located below the node representing job m_j.

First consider the single machine scheduling problem $1|prec;p_i = 1|C_{max}$. In this problem only one machine processes all the jobs, which should be scheduled regarding the precedence constraints. Since we have unit processing times, the minimal makespan's length equals n, and any linear extension of the partial order $\leq$ (including L_1 and L_2) provides an optimal scheduling

4. Order congruences of partially ordered sets

Before dealing with the parallel scheduling problems, we need an insight in the theory of order congruences. We have seen in the previous chapter, that the single machine scheduling problem can be solved by generating a linear extension of the partial order representing the problem. If more machines can work parallelly at the same time, we can solve the problem by dividing the set M of the jobs into disjoint subsets $M_1, M_2, \ldots, M_t$ where any set M_i contains only incomparable jobs which can be assigned to the parallelly working machines.

Let $(M, \leq)$ be a partially ordered set. A partition $\{M_1, M_2, \ldots, M_k\}$ of M is a covering of M by nonempty and pairwise disjoint subsets of M, that is $M = M_1 \cup M_2 \cup \ldots \cup M_k$ and $M_i \cap M_j = \emptyset$ if $i \neq j$. The subsets $M_i \subset M$ are called the *blocks* of the partition $(i = 1,2, \ldots, k)$. Since $\{M_1, M_2, \ldots, M_k\}$ is a partition of *M*, we can find an equivalence relation ρ on M such that

$$\frac{M}{\rho} = \{M_1, M_2, \ldots, M_k\}.$$

Therefore, the partition is a factor set of M with respect to ρ. Let

$$\phi: M \to \frac{M}{\rho},$$

denote the natural map, then

$$\phi(m_i) := M_j, \text{ if } m_i \in M_j.$$

Definition 4.1. Let $(M, \leq)$ be a poset and $\rho \subseteq M \times M$ be an equivalence relation on it. ρ is called an *order congruence* of $(M, \leq)$ if there exists an induced partial order $\leq_\rho$ on the factor set $\frac{M}{\rho} = \{M_1, M_2, \ldots, M_k\}$ such that the function $\phi: M \to \frac{M}{\rho}$ is order-preserving, that is $m_i \leq m_j$ implies $\phi(m_i) \leq_\rho \phi(m_j)$ for any $m_i, m_j \in M$.

The easiest way to give an order congruence of the poset $(M, \leq)$ is to cut into subintervals some of its linear extensions. More precisely, if $L = [x_1, x_2, \ldots, x_n]$ is a linear extension of $\leq$ on M, then form subsets of M in the following way:

$$M_1 = [x_1, x_{i_1}], \quad M_2 = [x_{i_1+1}, x_{i_2}] \;\ldots\; M_k = [x_{i_{k-1}+1}, x_n]. \quad (1)$$

The obtained partition $\{M_1, M_2, \ldots, M_k\}$ is the factor set of an order congruence of $(M, \leq)$.

Denote by $\vartheta(M)$ the set of all order congruences of a poset $(M, \leq)$. Clearly, $\vartheta(M)$ with the set-theoretical inclusion is a partially ordered set. To solve a parallel scheduling problem, we are interested in finding blocks where the induced partial order on the blocks is a linear order.

Definition 4.2. An order congruence $\rho \in \vartheta(M)$ is called a *linear order congruence* if the factor poset $\left(\frac{M}{\rho}, \leq_\rho\right)$ is a chain.

Suppose that we have an order congruence obtained by (1). It is a linear order congruence if any consecutive intervals M_j, M_{j+1} $(j \in \{1,2,\ldots,k-1\})$ have elements $x_s \in M_j$ and $x_t \in M_{j+1}$ such that $x_s \leq x_t$.

Obviously, the linear order congruence which is the solution of a parallel scheduling problem must be minimal in the following sense.

Definition 4.3. An order congruence $\rho \in \vartheta(M)$ is called a *minimal linear order congruence* of $(M, \leq)$ if there is no $\psi \in \vartheta(M)$, such that ψ is a linear order congruence and $\psi \subseteq \rho$ $(\psi \neq \rho)$.

If the intervals in (1) are antichains and for any two consecutive intervals M_j, M_{j+1} $(j \in \{1,2,\ldots,k-1\})$ there are elements $x_s \in M_j$ and $x_t \in M_{j+1}$ such that x_t covers x_s then we get a minimal linear order congruence (Körtesi et al. 2005).

Algorithm 4.4. The following method gives a partition which corresponds to a minimal linear order congruence in a partially ordered set.

Let $(M, \leq)$ be a poset and M_1 be the set of all minimal points in M. Delete all the points of M_1 from the set M and cancel all relations in which there is a point belonging to M_1. Then the process is continued for the set $(M\backslash M_1, \leq)$: let the set M_2 consist of the minimal points of $M\backslash M_1$, delete the points of M_2 from the set $M\backslash M_1$ and cancel all relations in which there is a point belonging to M_2. In general, the minimal points of $M\backslash(M_1 \cup M_2 \cup \ldots \cup M_{i-1})$ are collected in M_i, $i = 1,2,\ldots,t,$ where t denotes the height of the poset. To prove that the number of the sets obtained by the process equals exactly $t,$ consider the fact that each set M_i must be an antichain, because it contains incomparable points. For this reason, any two elements of the maximal chain cannot be in the same set M_i, so we need t different antichain to cover the poset $(M, \leq)$.

Consider the partially ordered set given in Example 2.2. Using Algorithm 4.4. we obtain the following partition of the set $M = \bigcup_{i=1}^{16} m_i$:

$$M_1 = \{m_{16}, m_{15}, m_{13}\}, \quad M_2 = \{m_{14}\}, \quad M_3 = \{m_{12}\}, \quad M_4 = \{m_{11}, m_{10}, m_9\}, \quad M_5 = \{m_8, m_7\},$$
$$M_6 = \{m_6, m_5, m_4, m_3\}, \quad M_7 = \{m_2\}, \quad M_8 = \{m_1\}.$$

The obtained sets are antichains, and in any two neighbor blocks one can find covering points, so this is the factor set of a minimal linear order congruence.

5. Parallel scheduling problems

In case of parallel machines scheduling problems, the aim is to assign the jobs to the machines where the number of machines can be a fixed number, or we can assume that arbitrary number of machines are available. Each job must be processed in one machine and each machine can only process one job at the same time. In general, if the processing times vary or we have more than two machines then to find an optimal scheduling is an NP-hard problem. If all of the jobs take unit time, and the number of machines is two, (that is, the problem is defined as $P2|prec; p_i = 1|C_{max}$), then a polynomial time algorithm exists to find the optimal solution. This method is the Coffman-Graham algorithm (Coffman and Graham, 1972), which can be executed in the following steps:

1. Make the transitive reduction of the precedence-graph, that is, remove all indirect dependencies (in case of the Hasse diagrams this step is completed, because the diagram contains only direct precedencies).
2. Label and sort the jobs by a lexicographical order which yields a special linear extension of the original partial order.
3. Apply Graham's list scheduling: whenever a machine has no work to do, assign to it he first unscheduled job from the list created in Step 2. The machine with the lower index is prior.

In Example 2.2, the jobs (the vertices of the graph) are labelled due to Step 2 of the algorithm, and after applying Step 3, we obtain the following optimal scheduling for two machines:

Machine 1	m_{16}	m_{14}	m_{12}	m_{11}	m_8	m_6	m_4	m_2	m_1
Machine 2	m_{15}	m_{13}	m_9	m_{10}	m_7	m_5	m_3	–	–

In a general $Pm|prec; p_i = 1|C_{max}$ problem, where $m > 2$, the Coffman-Graham algorithm usually does not find the optimal scheduling, but especially in small problems often provide a quasi-optimal solution. In our example the situation is even better in the case of $m = 3$. The Coffman-Graham algorithm gives the following solution to the problem $P3|prec; p_i = 1|C_{max}$:

Machine 1	m_{16}	m_{14}	m_{12}	m_{11}	m_8	m_6	m_3	m_1
Machine 2	m_{15}	–	–	m_{10}	m_7	m_5	m_2	–
Machine 3	m_{13}	–	–	m_9	–	m_4	–	–

This solution is optimal for our objective function, because we got a partition of 8 blocks, which means that the total completion time is 8 units which equals the length of the maximal chain in the set $(M, \leq)$. The chain with the maximal length is often referred as the critical path in scheduling theory because the jobs cannot be completed in steps less than the maximum chain length. The length of the critical path equals the number of the blocks in a minimal linear order congruence. Beside this, the partition corresponding to the minimal linear order congruence determines the maximum number of the machines worth setting to process all the jobs. In Example 2.2. the largest block of the minimal linear order congruence was M_6 having 4 elements. It means that there is no use to apply more than 4 machines, because there are only 4 jobs which can be completed at the same time. As a matter of fact, the blocks corresponding to the minimal linear order congruence also provide an optimal solution of the problem $P4|prec; p_i = 1|C_{max}$, because the makespan is minimal. On the other hand, this solution probably is not optimal with another objective function (for example total idling time).

6. Conclusion

In this paper some order theoretical notions and results were applied in solving special scheduling problems. If there are precedence relations between the jobs in a scheduling problem, then the set of the jobs can be considered as a partially ordered set where the order is defined by the precedencies. The single machine scheduling problem can be solved applying Szpilrajn's theorem directly, by giving any linear extension of the partial order determined by the precedencies. In case of parallel scheduling problems minimal linear order congruences can provide useful information in the solution process. A simple algorithm was presented to determine the partition corresponding to the minimal linear order congruence. The number of blocks in this partition is equal to the length of the critical path to the scheduling problem, and the number of items in the largest block indicates how many tasks can be completed at one time. Our results show that the theory of partially ordered sets is a promising approach to interpreting, modelling, and solving scheduling problems.

References

1. Coffman, E. G. and Graham, R. L. (1972). Optimal scheduling for two-processor systems. *Acta Informatica* 1, 200–213, https://doi.org/10.1007/BF00288685.
2. Davey, B. A. and Priestley, H. A. (2002). Introduction to Lattices and Order. Cambridge University Press.
3. Graham, R. L., Lawler, E. L., Lenstra, J. K. and. Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics* 5, 287-326, https://doi.org/10.1016/S0167-5060(08)70356-X.
4. Körtesi, P., Radeleczki, S. and Szilágyi, Sz. (2005). Congruences and isotone maps on partially ordered sets. *Mathematica Pannonica* 16/1, 39–55.
5. Pinedo, M. L. (2005). Planning and Scheduling in Manufacturing and Services. Springer-Verlag, https://doi.org/10.1007/978-1-4419-0910-7.
6. Szpilrajn, E. (1930). Sur l'extension de l'ordre partiel. *Fund. Math*. 16.1, 386-389, http://eudml.org/doc/212499.
7. Tella, Y., Singh, D. and Singh, J. N. (2014). Some aspects of topological sorting. *Covenant Journal of Informatics and Communication Technology* 2/1, 28–45.
8. Trotter, W. T. (2002). Combinatorics and Partially Ordered Sets, Dimension Theory. Johns Hopkins University Press.

Received: 1 April 2022
Accepted: 6 June 2022