Scheduling with Complete Multipartite Incompatibility Graph on Parallel Machines

Tytus Pikies [1], Krzysztof Turowski [2], and Marek Kubale [3]

[1,3] Department of Algorithms and System Modeling, Gdańsk University of Technology, Poland
[2] Theoretical Computer Science Department, Jagiellonian University, Poland

[1] tytpikie@pg.edu.pl, [2] krzysztof.szymon.turowski@gmail.com, [3] kubale@pg.edu.pl

1. The model considered

- There is a set of jobs $\mathcal{J}$ and a set of parallel machine $\mathcal{M}$.
- The jobs are in a binary relation forming a ***complete multipartite*** **graph**.
- On no machine there can be scheduled two jobs that are in the relation, i.e. that are connected by an edge in the graph.

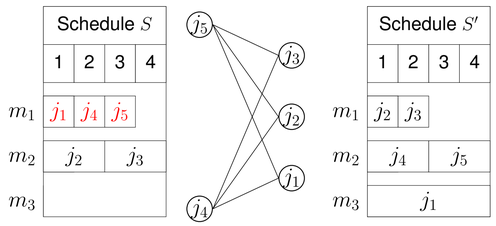

FIGURE 1: *An example instance: A sample set of jobs $\{j_1, \ldots, j_5\}$ and a set of machines $\{m_1, \ldots, m_3\}$ and the incompatibility relation; an unfeasible schedule S and a feasible schedule S'.*

For a broader overview of the considered model and a history of the research in this model see (Tytus Pikies, Krzysztof Turowski, Marek Kubale: Scheduling with Complete Multipartite Incompatibility Graph on Parallel Machines. ICAPS 2021: 262-270). See also (Klaus Jansen, Alexandra Lassota, Marten Maack, Tytus Pikies: Total Completion Time Minimization for Scheduling with Incompatibility Cliques. ICAPS 2021: 192-200) for results on a similar model.

2. A sample application

Imagine that we are treating some people ill with contagious diseases. There are quarantine units containing people ill with a particular disease waiting to receive some medical services. We also have a set of nurses, perhaps of different efficiency. We would like the nurses to perform the services in a way that no nurse will travel between different quarantine units, to avoid spreading of the diseases. Also, we would like to provide each patient with the required services, which correspond to the time to be spent by a nurse.
Consider two sample goals: The first might be to lift the quarantine in the general as fast as possible. The second might be to minimize the average time of patient treatment.
Notice, that such problems can be easily expressed in the presented scheduling model.

3. The results obtained

- An optimal polynomial time algorithm for $P|G = \textit{complete multipartite}|C_{\max}$.
- NP-hardness of $Q|G = \textit{complete multipartite}|C_{\max}$ and $Q|G = \textit{complete multipartite}|\sum C_j$.
- An optimal polynomial time algorithm for $Q|G = \textit{complete k-partite}, p_j = 1|C_{\max}$ and $Q|G = \textit{complete k-partite}, p_j = 1|\sum C_j$.
- 2-approximation algorithm for $Q|G = \textit{complete multipartite}, p_j = 1|C_{\max}$.
- 4-approximation algorithm for $Q|G = \textit{complete multipartite}, p_j = 1|\sum C_j$.
- 4-approximation algorithm for $Q|G = \textit{complete k-partite}|\sum C_j$.
- Inapproximability up to arbitrary constant of $R|G = \textit{complete 2-partite}|C_{\max}$ and $R|G = \textit{complete 2-partite}|\sum C_j$.

By *complete multipartite* we mean a graph with arbitrary number of parts. By *complete k-partite* we mean a graph with a number of parts bounded by a fixed value k.

4. The algorithms developed

- A optimal greedy method for $P|G = \textit{complete multipartite}|\sum C_j$:
  1. Assign to each part of the graph a single machine,
  2. Assign the remaining machines one by one to the parts in a way that decreases $\sum C_j$ in each step as much as possible.
- A dynamic programming method for $Q|G = \textit{complete k-partite}|C_{\max}$ and for $Q|G = \textit{complete k-partite}|\sum C_j$. For example for the latter problem it is as follows:
  1. Let initial state be $(|J_1|, \ldots, |J_k|; 0)$.
  2. Let S' be a set of all feasible sets of unscheduled jobs after considering the first i machines.
  3. Construct all feasible states of "unscheduled" jobs after considering the first $i+1$ machines as follows. For a given state $s_i \in S_i$ guess (exhaustive search) to which part m_i is to be assigned and how many jobs it has to do.
  4. Trim the states of for each $(a_1, \ldots, a_k)$ removing all the tuples $(a_1, \ldots, a_k; c)$ except the one where c is the smallest one.
- A greedy method with a dynamic programming twist for $Q|G = \textit{complete multipartite}|\sum C_j$.
  Let the first part gets the fastest l_1 machines, the next part gets the next fastest l_2 machines, etc. Let us call by *ordered assignment* any such assignment from machines to parts as long as $\sum_{i=1}^{k} l_i \leq m$. Always there exists an ordered assignment such that there is a corresponding schedule that is at most 4-approximate for $Q|G = \textit{complete k-partite}, p_j = 1|\sum C_j$. Moreover, we can find at least as good one by a dynamic programming.
- A suitably tailored rounding theorem, an exhaustive search, and a linear programming model for $Q|G = \textit{complete k-partite}|\sum C_j$:
  1. Round the speeds of the machines up to the nearest multiple of 2.
  2. For each part J_{pr}, guess the speed s'_{pr} and the number n'_{pr} of fastest machines assigned to it in an optimal schedule.
  3. Solve the following linear program with the variables:
     – $n_{pr,tp}$, where $pr \in \{1, \ldots, k\}$, $tp \in \{1, \ldots, l\}$,
     – $x_{jb,lr,tp}$, where $jb \in J_1 \cup \ldots, \cup J_k$, $lr \in \{1, \ldots, n\}$, $tp \in \{1, \ldots, l\}$.
     Let the LP conditions be:

$$\sum_{pr} n_{pr,tp} = |M_{tp}| \qquad \forall tp \qquad (1)$$

$$n_{pr,tp} = 0 \qquad \forall pr \forall tp > s'_{pr} \qquad (2)$$

$$n_{pr,tp} \leq |M_{tp}| - \sum_{i \in \{i | s'_i = tp\}} n'_i \qquad \forall pr \forall tp < s'_{pr} \qquad (3)$$

$$n_{pr,tp} = n'_{pr} \qquad \forall pr, tp = s'_{pr} \qquad (4)$$

$$\sum_{lr,tp} x_{jb,lr,tp} = 1 \qquad \forall jb \qquad (5)$$

$$\sum_{jb \in J_{pr}} x_{jb,lr,tp} \leq n_{pr,tp} \qquad \forall pr \forall tp \forall lr \qquad (6)$$

$$0 \leq x_{jb,lr,tp} \qquad \forall jb \forall lr \forall tp \qquad (7)$$

$$0 \leq n_{pr,tp} \qquad \forall pr \forall tp \qquad (8)$$

     Let the LP cost function be: $\sum_{jb,lr,tp} x_{jb,lr,tp} \cdot lr \cdot p(jb) \cdot \frac{1}{s(tp)}$, where $p(jb)$ is the processing requirement of job jb and $s(tp)$ is the speed factor of machine of type tp.
  4. Solve the jobs assignment for each part separately using the optimal solution of LP.

5. Summary & open problems

- The status of $P|G = \textit{complete multipartite}|\sum C_j$ has been settled.
- The complexity status of the problem $Q|G = \textit{complete k-partite}|\sum C_j$ and the best polynomial-time approximability of both $Q|G = \textit{complete multipartite}|\sum C_j$ and $Q|G = \textit{complete multipartite}|C_{\max}$ remain open problems. Also, the approximability status of $Q|G = \textit{complete multipartite}, p_j = 1|\sum C_j$ remains open.
- The statuses of $R|G = \textit{complete 2-partite}|C_{\max}(\sum C_j)$ have been settled. Are there some interesting subproblems of unrelated machines that admit exact or approximate polynomial time algorithms, assuming P $\neq$ NP?