Chapter 5
Concurrent Open Shops

5.1 Introduction

The *concurrent* open shop scheduling permits processing more than one operation of the same job at a time, which is the main difference from traditional open shop scheduling. We use the abbreviation *cncnt* of the word concurrent to denote concurrent open shop scheduling in the extended Graham et al. notation [14]. Thus the problem $O|\text{cncnt}|\sum C_i$ denotes the problem of minimization total completion time for concurrent open shops. Ahmadi et al. [1], Leung et al. [19], Wagneur and Sriskandarajah [26], and Cheng et al. [5] provide a broad list of real-life applications of the concurrent open shop scheduling. We list some of them below:

- *Product design*. A product design team whose members independently design modules for different products. A product design is only completed once all its modules have been designed, Ahmadi et al. [1];
- *Audit*. A team of accountants auditing various parts of different companies. The audited company receives a final report once all accountants completed their audit, Ahmadi et al. [1];
- *Assembly*. Assembly of a final product often needs to wait until all parts for the assembly are available, Ahmadi et al. [1] and Framinan et al. [11];
- *Lenses production*. Each type of plastic lenses is produced on a dedicated production line. The manufacturer produces lenses based on confirmed customer orders. Each order consists of different quantities of various lens types (order parts). After completion on different production lines the components of a customer order are packaged and shipped to the customer as a complete order, Ahmadi et al. [1];
- *A car repair shop*, Leung et al. [19];
- *A paper converting facility*. The facility produces paper products of different types and sizes from large rolls of paper. Any product is produced on a dedicated machine. Orders are received from customers; each order specifies quantities

W. Kubiak, *A Book of Open Shop Scheduling*, International Series in Operations Research & Management Science 325,
https://doi.org/10.1007/978-3-030-91025-9_5

of different paper products. The customer receives its entire order in a single shipment in order to reduce transportation and order handling costs, Leung et al. [19];
- *Airplane maintenance*, Wagneur and Sriskandarajah [26].

This broad range of applications has been almost matched by a long list of the terms used for concurrent open shops starting with the Coordinated Scheduling of Customer Orders Problem or the scheduling of customer orders used by Ahmadi and Bagchi [2], see also Ahmadi et al. [1], open shops with job overlaps used by Wagneur and Sriskandarajah [26], concurrent open shop used by Roemer [24], or the order scheduling used by Framinan and Perez-Gonzalez [10]. Roemer [24] points out that the literature on the problem of scheduling customer orders and on concurrent open shops evolved in parallel and in isolation from one another which may have lead to some redundancy of research efforts, see Roemer [24] for a detailed account of those concurrent research efforts. This book uses the term concurrent open shops which seems to best reflect the concept which is not limited to customer orders and manufacturing only.

In all those real-life situations described above the customer preference for the objective function seems to prevail. A customer prefers to receive a *complete* order fast, thus the minimization of flow time or equivalently the minimization of total completion time, may be preferred over other objectives. At the same time it may not be of much importance to the customer when exactly particular components of its order are completed as long as the complete order is received fast. Similarly, a client is interested in the arrival time of the final audit report rather than in the completion times of partial reports for various department, like in the audit example mentioned above. Ahmadi et al. [1] provide a comprehensive discussion of the rationale behind the choice of objective functions for the concurrent open shop scheduling. We focus on the following three objective functions: total completion time, number of tardy jobs, and total tardiness in this chapter. Those functions seem to be well aligned with the real-life applications of the concurrent open shop scheduling. In Chap. 6 we introduce a special class of concurrent open shops where some operations of a job are *required* to be processed simultaneously at any time.

5.2 Complexity of Concurrent Open Shop Scheduling

We begin with concurrent open shop scheduling with 0-1 operations. The solution to $O|\text{cncnt}, p_{ij} = 0, 1|C_{\max}$ is trivial. The optimal $C_{\max}$ equals the maximum machine workload L which occurs on machines with maximum number of operations. For any other regular objective function, optimal schedules exist among schedules with makespan on each machine being equal to the machine's workload. In other words among the schedules with no idle time on any machine. When the number of machines is part of the problem input the problem is NP-hard in the strong sense for total completion time and thus for total tardiness, and for the number of tardy jobs.

Somewhat surprisingly the proof of NP-hardness for $O|\text{cncnt}, p_{ij} = 0, 1|\sum C_i$ is much easier than for $O|p_{ij} = 0, 1|\sum C_i$, see Sect. 3.6.2. We have the following theorem.

Theorem 5.1 *The problems $O|cncnt, p_{ij} = 0, 1|\sum C_i, \sum U_i, \sum T_i$ are NP-hard in the strong sense.*

Proof The reduction is from the MAXIMUM INDEPENDENT SET problem, see Garey and Johnson [12]. Let a simple graph $G = (V, E)$ and a positive integer k be an instance of the MAXIMUM INDEPENDENT SET problem. We set the set of jobs $\mathcal{J} = V$ and the set of machines $\mathcal{M} = E$ in the corresponding instance of the $O|\text{cncnt}, p_{ij} = 0, 1|\sum C_i$ problem. The job $v \in V$ has a unit-time operation on each machine $e \in E$ such that $v \in e$, and it is missing on all other machines. The number of unit-time operations of job v equals the degree of vertex, $\deg(v)$, in G. We are looking for a concurrent open shop schedule with the total completion time $\sum C_i$ being at most $2|V| - k$ (the number of tardy jobs being at most $|V| - k$ for the objective $\sum U_i$, and the total tardiness being at most $|V| - k$ for the objective $\sum T_i$).

Without loss of generality we can limit ourselves to schedules with $C_{\max} = 2$. The key observation is that in any such schedule all jobs that complete at time 1 form an independent set in G, and all remaining jobs finish at time 2. Thus having larger independent set results in smaller total completion time (fewer tardy jobs, and smaller total tardiness assuming due date $d = 1$ for each job).

Suppose that schedule S has total completion time not exceeding $2|V| - k$. Without loss of generality we assume that each unit-time operation starts either at 0 or at 1 in S. Let J be the set of all jobs with *all* their unit-time operations starting at 0. We have $C_v = 1$ for each job $v \in J$. The jobs in $V \setminus J$ complete at 2 each, thus $C_v = 2$ for each job $v \in V \setminus J$. Therefore $\sum C_i = |J| + 2(|V| - |J|) = 2|V| - |J| \leq 2|V| - k$ for S, which implies $k \leq |J|$. The set J is independent, since otherwise there would be a machine $e = \{v, u\} \in E$ with $v, u \in J$, thus either v or u would need to complete at 2 in S which leads to contradiction since both v and u complete at 1 in S. Thus J is independent set in G of size at least k.

Now suppose U is an independent set of size at least k in G. Start each job $v \in U$ on each machines $e = \{v, u\} \in E$ for some $u \in V$ at 0. There is no conflict since the request to start two jobs u and v from U at 0 on the same machine e implies that $e = \{v, u\} \in E$ which is a contradiction since U is an independent set in G. Thus $C_v = 1$ for each $v \in U$. For each job $u \in J \setminus U$, start its unit-time operation at 1 on each machine $e = \{u, v\}$ for some $v \in U$ or either at 0 or at 1 on each machine $e = \{u, v\}$ for some $u \in J \setminus U$. Since there are exactly two unit-time operations of two different jobs on each machine e, a feasible concurrent schedule can easily be obtained, see Figs. 5.1 and 5.2 for an example. Observe that $C_u \leq 2$ for each $u \in J \setminus U$. Thus $\sum C_i \leq |U| + 2(|V| - |U|) = 2|V| - |U| \leq 2|V| - k$ since $|U| \geq k$. Hence we obtained a required schedule for an independent set of size at least k in G. The proof for the other two objective functions $\sum U_i$ and $\sum T_i$ is similar. □

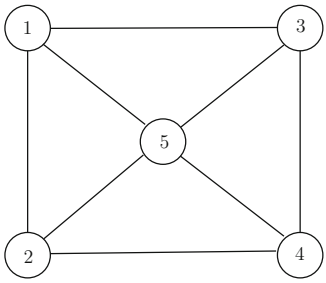

Fig. 5.1 Graph G

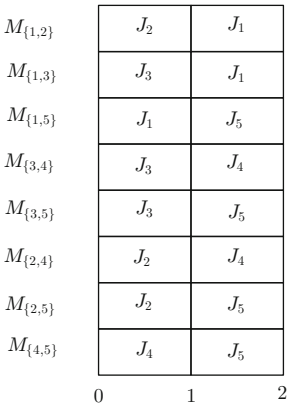

Fig. 5.2 A schedule for graph G and independent set $\{2, 3\}$

Roemer [24] settles the complexity status for the minimization of total completion time on two machines by proving the following theorem.

Theorem 5.2 *The problem* $O2|cncnt|\sum C_i$ *is NP-hard in the strong sense.*

5.3 Permutation Schedules and Minimization of Maximum Lateness

A schedule of concurrent open shop is a *permutation schedule* if all operations are scheduled without preemption, idle time, and in the same order on each machine. Wagneur and Sriskandarajah [26], see also Mastrolilli et al. [20], prove that optimal schedules for regular objective functions can be found among permutation schedules.

Theorem 5.3 *For each feasible schedule* S *of a concurrent open shop there is a permutation schedule* S_σ *that completes each job not later than in* S.

Proof Let S be a feasible schedule for an instance I of $O|\text{cncnt}|\sum C_i$. Let C_i be completion time of job J_i in S. For S, consider m instances $I_1, \ldots, I_m$, one for

each machine in $\mathcal{M}$, of the single machine maximum lateness minimization problem $1||L_{\max}$. The instance I_h for machine M_h is made up of n jobs $J_1, \ldots, J_n$ with processing times $p_{1,h}, \ldots, p_{n,h}$ and due dates $d_1 = C_1, \ldots, d_n = C_n$ respectively. The schedule S on machine M_h, denoted by S_h, completes job i at $C_{i,h} \leq C_i = d_i$ (we assume $C_{i,h} = 0$ if job J_i is missing on M_h). Thus S_h is a feasible schedule for I_h with zero maximum lateness. Without loss of generality we assume that the schedule is non-preemptive. Therefore the maximum lateness of an optimal solution for I_h is non-positive as well. The optimal solution S_h^* to $1||L_{\max}$ orders the jobs in the Earliest Due Date Order, σ, see Jackson [17]. The same argument works for each instance I_h, $h = 1, \ldots, m$. Thus the same EDD sequence for each I_h defines a permutation schedule S_σ for I. Since the maximum lateness for each S_h^* is non-positive, no job completes later in S_σ than it does in S. □

Leung et al. [19] take this result even further to prove that the Earliest Due Date permutation is optimal for the minimization of maximum lateness.

Theorem 5.4 *The Earliest Due Date permutation is optimal for* $O|cncnt|L_{\max}$.

Proof Consider an optimal schedule S for an instance I of $O|cncnt|L_{\max}$. By Theorem 5.3 we may assume that the schedule is a permutation schedule S_σ for some permutation of jobs σ. Suppose there exists position k, $k = 1, \ldots, n-1$, in σ such that $i = \sigma(k)$ and $j = \sigma(k+1)$ and $d_i > d_j$ so that σ is not EDD. We call the position k a *violator*. Let k be the violator in S_σ with the largest value of i. Without loss of generality assume that S_σ maximizes i, and among all permutation schedules with maximum i has the maximum k. The exchange of i and j on machine $\mathcal{M}_h$ results in permutation σ' such that

$$\max\{C_{i,h} - d_i, C_{j,h} - d_j\} > \max\{C'_{i,h} - d_i, C'_{j,h} - d_j\}$$

for $h = 1, \ldots, m$, where $C_{i,h}$ and $C'_{i,h}$ are completion times of job J_i on machine $\mathcal{M}_h$ in S_σ and $S_{\sigma'}$ respectively. Hence

$$\max\{\max_h\{C_{i,h} - d_i\}, \max_h\{C_{j,h} - d_j\}\} >$$
$$\max\{\max_h\{C'_{i,h} - d_i\}, \max_h\{C'_{j,h} - d_j\}\}$$

and the exchange reduces the maximum lateness for the pair i and j, and leaves the maximum lateness for the remaining jobs unchanged. Therefore we get an optimal permutation schedule $S_{\sigma'}$ for I with either smaller i or the same i but larger k which contradicts the choice of S_σ and proves the theorem. □

The EDD solution to $O|\text{cncnt}|L_{\max}$ is illustrated for an instance with $n = 7$ jobs and $m = 5$ machines given in Table 5.1. The EDD permutation is $J_2, J_5, J_1, J_4, J_7, J_6, J_3$ for the instance. The optimal schedule is given in Fig. 5.3 where $L_{\max} = L_3 = C_3 - d_3 = 18 - 11 = 7$.

Table 5.1 An instance of $O|\text{cncnt}|L_{\max}$ with $m = 5$ machines

Job (i)	M_1	M_2	M_3	M_4	M_5	d_i
1	3	2	0	0	1	7
2	1	0	4	6	0	5
3	2	0	3	3	1	11
4	0	4	4	0	3	8
5	0	0	5	1	2	6
6	4	3	0	0	3	10
7	3	0	2	0	1	9

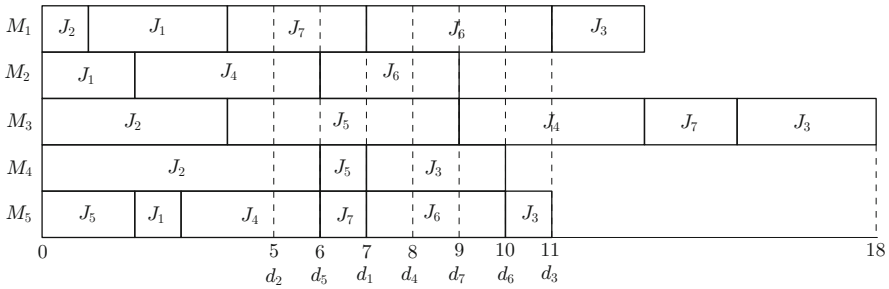

Fig. 5.3 Optimal schedule with the EDD permutation $J_2, J_5, J_1, J_4, J_7, J_6, J_3$ for the instance in Table 5.1

5.4 2-Approximation Algorithm for $O|\text{cncnt}|\sum w_i C_i$

5.4.1 The Algorithm

In this section we present a 2-approximation algorithm for the problem $O|\text{cncnt}|\sum C_i$ developed by Mastrolilli et al. [20]. By Theorem 5.3 the search for optimal or approximate schedules can be limited to permutation schedules. All such schedules have the same makespan $C_{\max}$ which equals maximum machine workload L. Thus one knows that the job in position n completes at $C_{\max}$ but one does not know which job that is, i.e., does not know $\sigma(n)$. The approximation algorithm considers all machines with the heaviest workload $C_{\max}$ and selects one of them, say M_h (possible ties are broken arbitrarily). The machine then is used to select a job $\sigma(n)$ for the position n. The selected job has the minimum job weight to operation-processing-time on M_h ratio $\frac{w_i}{p_{i,h}}$ (again possible ties are broken arbitrarily). The selection $\sigma(n)$ affects both the machines workloads, which are reduced by deleting job $\sigma(n)$ from each machine, and the weight w_i of each remaining job i, which is reduced by $\frac{w_{\sigma(n)}}{p_{\sigma(n),h}} p_{i,h}$, prior to the next iteration. The next iteration, in order to find the job $\sigma(n-1)$ in position $n-1$, begins with these updated machine workloads and job weights and proceeds exactly in the same way as for the job in position n. The algorithm stops after n iterations once the permutation $\sigma(1), \ldots, \sigma(n)$ has been found. Formally, the algorithm works with two main lists:

The list of machine workloads in iteration k, $k = n, \ldots, 1$,

$$L_1(k), \ldots, L_m(k),$$

and the list of job weights in iteration k

$$w_i(k) \qquad i \in J(k),$$

where $J(k)$ is the set of k jobs left to schedule in iteration k. The algorithm starts with the following machine-workload list:

$$L_1(n) = L_1, \ldots, L_m(n) = L_m,$$

the following job weight list:

$$w_1(n) = w_1, \ldots, w_n(n) = w_n, \tag{5.1}$$

and the set of jobs

$$J(n) = N = \mathcal{J}.$$

In iteration k, $k = n, \ldots, 1$, the algorithm computes: The index of a machine with the *heaviest* workload in that iteration

$$\mu(k) = \arg\max_h \{L_h(k)\};$$

the index of a job with the *minimum* job weight to operation-processing-time ratio on that machine

$$\sigma(k) = \arg\min_i \left\{ \frac{w_i(k)}{p_{i,\mu(k)}} \right\};$$

and the *minimum* job weight to operation-processing-time ratio on that machine

$$\theta(k) = \frac{w_{\sigma(k)}(k)}{p_{\sigma(k),\mu(k)}}. \tag{5.2}$$

The job $\sigma(k)$ is then scheduled in position k on each machine. To complete the iteration the algorithm reduces machine workloads by deleting job $\sigma(k)$ from each machine and sets

$$L_h(k-1) = L_h(k) - p_{\sigma(k),h} \qquad h = 1, \ldots, m,$$

and updates the job weights by setting

$$w_i(k-1) = w_i(k) - \theta(k)p_{i,\mu(k)} \qquad i \in J(k). \tag{5.3}$$

Finally

$$J(k-1) = J(k) \setminus \{\sigma(k)\},$$

and the next iteration starts if $k > 1$. Otherwise, the algorithm stops. The algorithm runs in $O(n(n+m))$ time.

5.4.2 The Proof

Following Mastrolilli et al. [20], we now show that the algorithm is a 2-approximation algorithm for $O|\text{cncnt}|\sum_i w_i C_i$. We begin with the following observation about the job weights produced by the algorithm:

Observation 5.5 *By (5.1) and (5.3) we have*

$$w_j(k-1) = w_j - \sum_{l=k}^{n} p_{j,\mu(l)}\theta(l)$$

and by (5.2) and (5.3)

$$w_{\sigma(k)}(k-1) = 0.$$

The observation implies the following observation about the weight $w_{\sigma(k)}$ of the job in position k.

Observation 5.6 *For any job $\sigma(k)$*

$$\begin{aligned}
&\sum_{h=1}^{m} p_{\sigma(k),h} \sum_{S\subseteq N:\sigma(k)\in S} y_{S,h} \\
&= p_{\sigma(k),\mu(k)} y_{J(k),\mu(k)} + \cdots + p_{\sigma(n),\mu(n)} y_{J(n),\mu(n)} \\
&= p_{\sigma(k),\mu(k)}\theta(k) + \cdots + p_{\sigma(n),\mu(n)}\theta(n) \\
&= w_{\sigma(k)} - w_{\sigma(k)}(k-1) \\
&= w_{\sigma(k)},
\end{aligned}$$

where

$$y_{S,h} = \begin{cases} \theta(k) & \text{if } h = \mu(k) \text{ and } S = J(k) \text{ for some } k = 1,\ldots,n, \\ 0 & \text{otherwise.} \end{cases} \tag{5.4}$$

Finally, we observe the following for the job completion times.

Observation 5.7

$$C_{\sigma(k)} = \sum_{i \in J(k)} p_{i,\mu(k)} = \sum_{i=1}^{k} p_{\sigma(i),\mu(k)} \quad k = 1, \ldots, n$$

and

$$C_{\sigma(1)} \leq \cdots \leq C_{\sigma(n)}.$$

We are now ready to prove the main theorem of this section.

Theorem 5.8 *The algorithm is a 2-approximation algorithm for* $O|cncnt|\sum w_i C_i$.

Proof The following LP- relaxation of $O|\text{cncnt}|\sum w_i C_i$

$$LP : \min \sum_{i=1}^{n} w_i C_i$$

$$\text{s. t.} \sum_{i \in S} p_{i,h} C_i \geq f_h(S) \quad \text{for all } h = 1, \ldots, m \text{ and } S \subseteq N,$$

where

$$f_h(S) = \frac{1}{2} \sum_{i \in S} p_{i,h}^2 + \frac{1}{2} \left(\sum_{i \in S} p_{i,h} \right)^2 \tag{5.5}$$

was given in Chen and Hall [4], see also Mastrolilli et al. [20]. Its dual was given in Mastrolilli et al. [20]

$$D : \max \sum_{h=1}^{m} \sum_{S \subseteq N} f_h(S) y_{S,h}$$

$$\text{s. t} \sum_{h=1}^{m} p_{i,h} \sum_{S \subseteq N : i \in S} y_{S,h} = w_i \quad \text{for all } i = 1, \ldots, n,$$

$$y_{S,h} \geq 0 \quad \text{for all } h = 1, \ldots, m \text{ and } S \subseteq N.$$

For the function $f_h(S)$ in both linear programs, Schulz [25] proves the following inequality:

$$\left(\sum_{i \in S} p_{i,h} \right)^2 \leq \left(2 - \frac{2}{n-1} \right) f_h(S) \quad \text{for any } h = 1, \ldots, m \text{ and } S \subseteq N. \tag{5.6}$$

We are now ready to complete the proof. We have

$$\begin{aligned}
\sum_{k=1}^{n} w_{\sigma(k)} C_{\sigma(k)} &=^{(1)} \sum_{k=1}^{n} \Big(\sum_{h=1}^{m} p_{\sigma(k),h} \sum_{S \subseteq N : \sigma(k) \in S} y_{S,h} \Big) C_{\sigma(k)} \\
&=^{(2)} \sum_{k=1}^{n} \Big(\sum_{h=1}^{m} p_{\sigma(k),h} \big(y_{J(k),h} + \cdots + y_{J(n),h} \big) \Big) C_{\sigma(k)} \\
&= \sum_{k=1}^{n} \Big(p_{\sigma(k),\mu(k)} y_{J(k),\mu(k)} + \cdots + p_{\sigma(k),\mu(n)} y_{J(n),\mu(n))} \Big) C_{\sigma(k)} \\
&= \sum_{k=1}^{n} y_{J(k),\mu(k)} \big(p_{\sigma(1),\mu(k)} C_{\sigma(1)} + \ldots p_{\sigma(k),\mu(k)} C_{\sigma(k)} \big) \\
&\leq^{(3)} \sum_{k=1}^{n} y_{J(k),\mu(k)} \Big(C_{\sigma(k)} \big(p_{\sigma(1),\mu(k)} + \ldots p_{\sigma(k),\mu(k)} \big) \Big) \\
&=^{(4)} \sum_{k=1}^{n} y_{J(k),\mu(k)} \big(p_{\sigma(1),\mu(k)} + \ldots p_{\sigma(k),\mu(k)} \big)^2 \\
&\leq^{(5)} \left(2 - \frac{2}{n+1} \right) \sum_{k=1}^{n} y_{J(k),\mu(k)} f_{\mu(k)}(J(k)) \\
&\leq^{(6)} \left(2 - \frac{2}{n+1} \right) \sum_{k=1}^{n} w_i C_i^{LP} \\
&\leq^{(7)} \left(2 - \frac{2}{n+1} \right) \sum_{k=1}^{n} w_i C_i^{*},
\end{aligned}$$

where the equality (1) follows from Observation 5.6, the equality (2) follows from (5.4), the inequalities (3) and (4) follow from Observation 5.7, the inequality (5) follows by (5.6), the inequality (6) holds since the solution $y_{S,h}$ is feasible for the dual D, and C_i^{LP} is an optimal solution to the primal LP, finally (7) holds since C_i^* is an optimal solution to $O|\text{cncnt}|\sum w_i C_i$. □

Mastrolilli et al. [20] prove that the performance guarantee of the algorithm cannot be better than $2 - \frac{2}{n+1}$.

5.4.3 An Example

We illustrate the algorithm run on an instance of $O|\text{cncnt}|\sum w_i C_i$ with $n = 7$ jobs and $m = 5$ machines specified in Table 5.2.

In iteration $k = 7$ we have the following list of machine workloads in Table 5.3 thus $\mu(7) = 3$, and the list of job weights is shown in Table 5.4 thus $\sigma(7) = 5$, and $\theta(7) = 0.4$. The iteration $k = 6$ starts with the following machine workloads in Table 5.5. Thus $\mu(6) = 3$, recall that the ties can be broken arbitrarily. The weights for the remaining jobs in $J(6)$ are given in Table 5.6. Thus $\sigma(6) = 2$ and $\theta(6) = \frac{3.4}{4} = 0.85$.

The iteration $k = 5$ starts with the machine workloads in Table 5.7. Thus $\mu(5) = 1$.

The weights for the remaining jobs in $J(5)$ are given in Table 5.8. Thus $\sigma(5) = 1$ and $\theta(5) = \frac{3}{3} = 1$.

The iteration $k = 4$ starts with the machine workloads in Table 5.9. Thus $\mu(4) = 3$, again ties can be broken arbitrarily.

The weights for the remaining jobs in $J(4)$ are given in Table 5.10. Thus $\sigma(4) = 4$ and $\theta(4) = \frac{3}{4} = 0.75$.

The iteration $k = 3$ starts with the machine workloads in Table 5.11. Thus $\mu(3) = 1$.

The weights for the remaining jobs in $J(3)$ are given in Table 5.12. Thus $\sigma(3) = 6$ and $\theta(3) = \frac{3}{4} = 0.75$.

Finally, the iteration $k = 2$ starts with the following machine workloads in Table 5.13. Thus $\mu(2) = 3$.

The weights for the remaining jobs in $J(2)$ are given in Table 5.14. Thus $\sigma(2) = 3$ and $\theta(2) = \frac{2.5}{3}$.

Table 5.2 An instance of $O|\text{cncnt}|\sum w_i C_i$ with $m = 5$ machines

Job (i)	M_1	M_2	M_3	M_4	M_5	w_i
1	3	2	0	0	1	3
2	1	0	4	6	0	5
3	2	0	3	3	1	10
4	0	4	4	0	3	8
5	0	0	5	1	2	2
6	4	3	0	0	3	7
7	3	0	2	0	1	11

Table 5.3 Machine workloads in iteration $k = 7$: $\mu(7) = 3$

	M_1	M_2	M_3	M_4	M_5
Load	13	9	18	10	11

Table 5.4 Job weights and operation processing times on machine $\mu(7) = 3$ in iteration $k = 7$

	J_1	J_2	J_3	J_4	J_5	J_6	J_7
Weight	3	5	10	8	2	7	11
M_3	0	4	3	4	5	0	2

Table 5.5 Machine workloads in iteration $k = 6$: $\mu(6) = 3$

	M_1	M_2	M_3	M_4	M_5
Load	13	9	13	9	9

Table 5.6 Job weights and operation processing times on machine $\mu(6) = 3$ in iteration $k = 6$

	J_1	J_2	J_3	J_4	J_6	J_7
Weight	3	3.4	8.8	6.4	7	10.2
M_3	0	4	3	4	0	2

Table 5.7 Machine workloads in iteration $k = 5$: $\mu(5) = 1$

	M_1	M_2	M_3	M_4	M_5
Load	12	9	9	3	9

Table 5.8 Job weights and operation processing times on machine $\mu(5) = 1$ in iteration $k = 5$

	J_1	J_3	J_4	J_6	J_7
Weight	3	6.25	3	7	8.5
M_1	3	2	0	4	3

Table 5.9 Machine workloads in iteration $k = 4$: $\mu(4) = 3$

	M_1	M_2	M_3	M_4	M_5
Load	9	7	9	3	8

Table 5.10 Job weights and operation processing times on machine $\mu(4) = 3$ in iteration $k = 4$

	J_3	J_4	J_6	J_7
Weight	4.25	3	3	5.5
M_3	3	4	0	2

Table 5.11 Machine workloads in iteration $k = 3$: $\mu(3) = 1$

	M_1	M_2	M_3	M_4	M_5
Load	9	3	5	3	5

Table 5.12 Job weights and operation processing times on machine $\mu(3) = 1$ in iteration $k = 3$

	J_3	J_6	J_7
Weight	4	3	4
M_1	2	4	3

Table 5.13 Machine workloads in iteration $k = 2$: $\mu(2) = 3$

	M_1	M_2	M_3	M_4	M_5
Load	5	0	5	3	2

Table 5.14 Job weights and operation processing times on machine $\mu(2) = 3$ in iteration $k = 2$

	J_3	J_7
Weight	2.5	1.75
M_3	3	2

Therefore the algorithm produces the following permutation: $J_7, J_3, J_6, J_4, J_1, J_2, J_5$ of jobs on each machine. The schedule for this permutation is shown in Fig. 5.4; the total weighted completion time for the schedule $\sum w_i C_i$ equals 355.

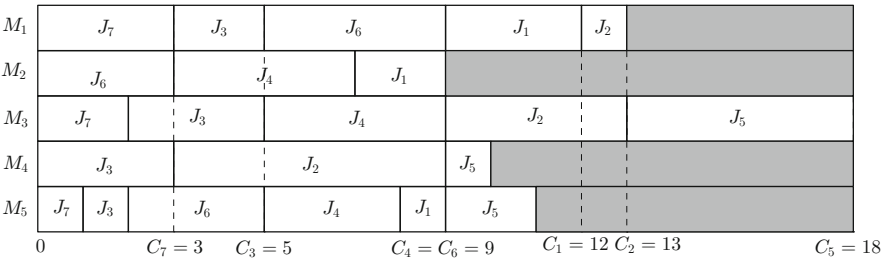

Fig. 5.4 2-approximation solution for the instance in Table 5.2

5.5 Hardness of Approximation

This section concentrates on the hardness of approximation for the problems of minimization of total completion time, $O|\text{cncnt}|\sum C_i$, total tardiness, $O|\text{cncnt}|\sum T_i$, and number of tardy jobs, $O|\text{cncnt}|\sum U_i$, for concurrent open shops. We begin by introducing a key inapproximability result for independent sets of uniform hypergraphs.

A pair $H = (N, E)$ where N is a finite set of vertices and E is a family of subsets (hyperedges) of N each of size exactly r is called an r-uniform hypergraph. An *independent set* of an r-uniform hypergraph $H = (N, E)$ is a subset I of vertices that does not completely include any of the hyperedges in E, i.e., $e \setminus I \neq \emptyset$ for each hyperedge $e \in E$. Dinur et al. [7] prove the following.

Theorem 5.9 *For any $\gamma \in (0, 1)$, $\delta \in (0, 1/2)$, and integer $r \geq 3$ the following problem is NP-hard. Given an r-uniform hypergraph $H = (N, E)$ decide whether*

(i) There exists an independent set of H of size at least $\left(1 - \frac{1}{r-1} - \delta\right)|N|$, or
(ii) Each independent set of H has size strictly less than $\gamma|N|$.

5.5.1 $O|cncnt|\sum C_i$

Mastrolilli et al. [20] make the decision problem in Theorem 5.9 a point of departure to prove the following limit on the approximation guarantee for polynomial-time algorithms for $O|\text{cncnt}|\sum C_i$.

Theorem 5.10 *The problem $O|cncnt|\sum C_i$ is NP-hard to approximate within a factor $\frac{6}{5} - \epsilon$ for any $\epsilon > 0$, unless $P = NP$.*

Proof By Theorem 5.9, for any $\gamma \in (0, 1)$, $\delta \in (0, \frac{1}{2})$, and integer $r \geq 3$, there is a class of r-uniform hypergraphs where the size of a maximum independent set is either greater or equal $\left(1 - \frac{1}{r-1} - \delta\right)|N|$ or less than $\gamma|N|$, and it is NP-hard to decide

for a given hypergraph $H = (N, E)$ in that class whether it falls in the former or in the latter category. If such decision could be made in polynomial time, then $P = NP$ which seems unlikely. In the corresponding instance $f(H)$ of $O|\text{cncnt}|\sum C_i$, we have $\mathcal{J} = N$ and $\mathcal{M} = E$. A job $v \in N$ has a unit-time operation on each machine $e \in E$ such that $v \in e$, and it is missing on any other machine.

Consider an independent set I of H. On each machine $e \in E$ schedule the jobs in $e \cap I$ in the interval $[0, |e \cap I|]$, and the jobs in $e \cap (N \setminus I)$ in the interval $[|e \cap I|, r]$. Since $e \setminus I \neq \emptyset$ for each $e \in E$, we have $|e \cap I| \leq r - 1$. Moreover, since each machine's e workload is exactly r for r-uniform hypergraphs, each job in $N \setminus I$ can be completed by r. Therefore, we can readily obtain a feasible schedule where each job in I completes by $r - 1$, and each job in $N \setminus I$ completes by r. Therefore, (i) in Theorem 5.9 can be used to calculate an upper bound U on the value $OPT(f(H))$ of minimum total completion time for the instance $f(H)$ with H in the category (i) as follows. If (i) holds, then

$$\begin{aligned} OPT(f(H)) &\leq (r-1)|I| + r(|N| - |I|) \\ &= r|N| - |I| \\ &\leq r|N| - \Big(1 - \frac{1}{r-1} - \delta\Big)|N| \\ &= \Big(r - 1 + \frac{1}{r-1} + \delta\Big)|N| \\ &= U. \end{aligned}$$

On the other hand, in an optimal schedule for $f(H)$ the set of all jobs that complete by $r - 1$ is an independent set I_O of H, and all jobs in $N \setminus I_O$ complete at r. Therefore, (ii) in Theorem 5.9 can be used to calculate a lower bound L on the value $OPT(f(H))$ of minimum total completion time for $f(H)$ with H in the category (ii) as follows. If (ii) holds, then

$$\begin{aligned} OPT(f(H)) &= r(|N| - |I_O|) + |I_O| \\ &= r|N| - (r-1)|I_O| \\ &> r|N| - (r-1)\gamma|N| \\ &= \Big(r - (r-1)\gamma)\Big)|N| \\ &= L. \end{aligned}$$

Suppose A is a $\left(\frac{6}{5} - \epsilon\right)$-approximation algorithm for $O|\text{cncnt}|\sum C_i$, for some $0 < \epsilon < 1$, which runs in polynomial time. Consider the class $\mathcal{C}$ of r-hypergraphs with $r = 3, \gamma = \frac{\epsilon}{4}, \delta = \frac{\epsilon}{2}$. We have $U = \frac{5+\epsilon}{2}$ and $L = 3 - \frac{\epsilon}{2}$. Run A on $f(H)$ where $H \in \mathcal{C}$, if $A(f(H)) \leq L$, then $OPT(f(H)) \leq A(f(H)) \leq L$ and the condition (ii) does not hold for H. Thus, the size of maximum independent set is greater or

equal $\left(1-\frac{1}{r-1}-\delta\right)|N|$. If $\frac{A(f(H))}{\frac{6}{5}-\epsilon} > U$, then $OPT(f(H)) \geq \frac{A(f(H))}{\frac{6}{5}-\epsilon} > U$ and (i) does not hold for H. Thus, the size of maximum independent set is less than $\gamma|N|$. Finally there is no instance $f(H)$ such that

$$L < A(f(H)) \leq \left(\frac{6}{5}-\epsilon\right)U \tag{5.7}$$

since

$$\frac{L}{U} = \frac{6-\epsilon}{5+\epsilon} > \frac{6}{5}-\epsilon \tag{5.8}$$

for $H \in C$. Therefore A could distinguish between the two categories of hypergraphs in the class C in polynomial time which leads by Theorem 5.9 to contradiction if $P \neq NP$, and proves the theorem for $O|\text{cncnt}|\sum C_i$. □

The factor $\frac{6}{5}-\epsilon$ in Theorem 5.10 can be strengthened to $2-\epsilon$ under the assumption that the Unique Games Conjecture holds, see Khot [18], and Bansal and Khot [3] for the conjecture. The following inapproximability result of Bansal and Khot [3] is key for the proof of the factor $2-\epsilon$.

Theorem 5.11 *Assuming the Unique Games Conjecture holds, for any $\delta \in (0,1)$, $\gamma \in (0,1)$, and integer $r \geq 2$ the following problem is NP-hard. Given an r-uniform hypergraph $H=(N,E)$ decide whether*

(iii) There exist disjoint subsets $N_1, \ldots, N_r \subseteq N$, satisfying $|N_i| \geq \frac{1-\delta}{r}|N|$ and such that $|e \cap N_i| \leq 1$ for $e \in E$ and $i = 1, \ldots, r$, or

(iv) Each independent set of H has size at most $\gamma|N|$.

This result is then used by Bansal and Khot [3] to prove the $2-\epsilon$ factor.

Theorem 5.12 *Assuming the Unique Games Conjecture, $O|\text{cncnt}|\sum C_i$ is hard to approximate within a factor $2-\epsilon$ for any $\epsilon > 0$, unless $P = NP$.*

Proof We use the same transformation $f(H)$ as in the proof of Theorem 5.10. If (iii) in Theorem 5.11 holds for H, then schedule job $v \in N_i$ in time slot $[i-1, i]$ of each machine e such that $v \in e \cap N_i$. The schedule thus obtained is feasible since the sets $N_1, \ldots, N_r$ are disjoint, and $|e \cap N_i| \leq 1$ for each $e \in E$ and $i = 1, \ldots, r$. Therefore at least $\frac{1-\delta}{r}|N|$ jobs complete at i for $i = 1, \ldots, r$ in the schedule. Complete the schedule by scheduling the remaining jobs from $N \setminus \left(N_1 \cup \cdots \cup N_r\right)$ in the available unit-time slots in the interval $[0, r]$ on each machine e. Each of those jobs completes by r. Hence we get the following upper bound on the total completion time $OPT(f(H))$ for each hypergraph H in the category (iii)

$$\begin{aligned} OPT(f(H)) &\leq \left(\frac{1-\delta}{r}(1+\cdots+r)+\delta r\right)|N| \\ &= \left(\frac{1-\delta}{2}(r+1)+\delta r\right)|N| \end{aligned}$$

$$= \Big(\frac{1+\delta}{2}r + \frac{1-\delta}{2}\Big)|N|$$
$$\leq \Big(\frac{1+\delta}{2}(r+1)\Big)|N|$$
$$= U.$$

If (iv) in Theorem 5.11 holds for H, the lower bound is calculated in a similar way as for (ii) in the proof of Theorem 5.10

$$OPT(f(H)) \geq \Big(r - (r-1)\gamma)\Big)|N|$$
$$= L.$$

Suppose there is a $(2-\epsilon)$-approximation polynomial-time algorithm A for $O|\text{cncnt}|\sum C_i$. Without loss of generality $0 < \epsilon < 1$. Consider the class C of r-hypergraphs with $\delta \leq \frac{1}{r+1}$, $\gamma \leq \frac{1}{r-1}$, and $r \geq \frac{6}{\epsilon} - 1$. Run A on an instance $f(H)$ with $H \in C$. If $A(f(H)) < L$, then $OPT(f(H)) \leq A(f(H)) < L$ and the condition (iv) does not hold for H. Thus, H falls in the category (iii). If $\frac{A(f(H))}{2-\epsilon} > U$, then $OPT(f(H)) \geq \frac{A(f(H))}{2-\epsilon} > U$ and (iii) does not hold for H. Thus, the size of maximum independent set is less than $\gamma|N|$ and H falls in the category (iv). Finally there is no $f(H)$ such that

$$L \leq A(f(H)) \leq (2-\epsilon)U \tag{5.9}$$

since

$$\frac{L}{U} = 2\frac{(r-(r-1)\gamma)}{(r+1)(1+\delta)} > 2-\epsilon \tag{5.10}$$

for the class C. Therefore A could distinguish between the two categories of hypergraphs from C in polynomial time. This, assuming the Unique Games Conjecture holds, leads by Theorem 5.11 to contradiction if $P \neq NP$ and proves the theorem.□

5.5.2 $O|cncnt|\sum T_i$, and $O|cncnt|\sum U_i$

The problems $O|\text{cncnt}|\sum T_i$ and $O|\text{cncnt}|\sum U_i$ are harder to approximate than $O|\text{cncnt}|\sum C_i$. Polynomial-time algorithms cannot guarantee approximations within a factor $(1-c)\ln m$ for any constant $c > 0$ for those two due date based problems.

To prove this we first recall a hard to approximate SET COVER problem which becomes a point of departure in the hardness proof for $O|\text{cncnt}|\sum U_i$ and $O|\text{cncnt}|\sum T_i$. An instance of the SET COVER problem is made up of a collection

$V_1, \ldots, V_\ell$ of subsets of a set $V = \{v_1, \ldots, v_n\}$. The SET COVER problem is the problem of selecting as few as possible subsets from the collection $V_1, \ldots, V_\ell$ such that every $v \in V$ is included into at least one of the selected subsets, Garey and Johnson [12]. Feige [9] shows that the problem cannot be approximated within a factor $(1-c)\ln n$ for any $c > 0$ in polynomial time, unless NP includes slightly superpolynomial problems. Dinur and Steurer [8] strengthen this result by proving that the problem cannot be approximated within a factor $(1-c)\ln n$ for any $c > 0$ in polynomial time, unless $P = NP$. The following result of Dinur and Steurer [8] is key to showing that both $O|\text{cncnt}|\sum U_i$ and $O|\text{cncnt}|\sum T_i$ are hard to approximate within a factor $(1-c)\ln m$ for any $c > 0$, unless $P = NP$.

Theorem 5.13 *Set cover is NP-hard to approximate within a factor* $(1-c)\ln n$ *for any* $c > 0$, *unless* $P = NP$.

We have the following inapproximability result for $O|\text{cncnt}|\sum U_i$ and $O|\text{cncnt}|\sum T_i$.

Theorem 5.14 $O|cncnt|\sum U_i$ *and* $O|cncnt|\sum T_i$ *are hard to approximate within a factor* $(1-c)\ln m$ *for any* $c > 0$, *unless* $P = NP$.

Proof Let collection $V_1, \ldots, V_\ell$ of subsets of V be an instance I of the SET COVER problem. For each $v \in V$ define the set $S_v = \{i : v \in V_i, i = 1, \ldots \ell\}$ and $S = \{1, \ldots, \ell\}$. We assume that all sets S_v have the same cardinality $r = \max_v\{|S_v|\}$. Otherwise, we can add a collection of $r - |S_v| < r$ copies of the subset $\{v\}$ to the collection $V_1, \ldots, V_\ell$, thus creating a regular instance I_r. The sizes of minimum set cover are the same in both I and I_r. Each S_v corresponds to machine S_v in the concurrent open shop. There are $m = n = |V|$ machines in the concurrent open shop instance I_O. Each $i \in S$ corresponds to a job with a unit-time operation on each machine S_v such that $i \in S_v$ and missing operations on any machine S_v such that $i \notin S_v$. There are ℓ jobs in I_O. Set $d_i = r - 1$ for each job in I_O.

Let the family $\mathcal{V}$ be a minimum set cover. Then $S_v \cap \mathcal{V} \neq \emptyset$ for each $v \in V$ so that each machine has a job from $\mathcal{V}$. Consider a schedule where all the jobs from $\mathcal{V}$ are scheduled at the end of the schedule on each machine. Thus each job in $\{1, \ldots, \ell\} \setminus \mathcal{V}$ completes by $r - 1$ in the schedule, and each job in $\mathcal{V}$ completes at r. Thus exactly $\ell - |\mathcal{V}|$ jobs completes by their due dates in the schedule and the number of tardy jobs equals the size of the subset cover $|\mathcal{V}|$, i.e., $\sum U_i = |\mathcal{V}|$.

On the other hand, for a schedule with the number of tardy jobs equal to $\sum U_i$ this number equals the number of jobs that complete at r in the schedule. Let S' be the set of those jobs. Thus for each machine S_v there exists job $i \in S'$, or in other words for each v there is V_i such that $v \in V_i$. Thus S' is a subset cover, and $\sum U_i = |S'|$.

Suppose there is $(1-c)\ln m$-approximation polynomial-time algorithm A for $O|\text{cncnt}|\sum U_i$ for some $c > 0$, then we have

$$\frac{\sum U_i^A(I_O)}{\sum U_i^{OPT}(I_O)} \leq (1-c)\ln m, \tag{5.11}$$

where $\sum U_i^A(I_O)$ and $\sum U_i^{OPT}(I_O)$ are the numbers of tardy jobs in a schedule produced by A and in an optimal schedule, respectively. We have $\sum U_i^A(I_O) = |S^A(I_r)|$ and $\sum U_i^{OPT}(I_O) = |S^{OPT}(I_r)|$ for some set covers $S^A(I_r)$ and $S^{OPT}(I_r)$ for I_r. However, $|S^{OPT}(I_r)|$ is the cardinality of minimum set cover $C^*(I_r)$ for I_r and thus the cardinality of minimum set cover $C^*(I)$ for I. Finally, there is set cover $S(I)$ for I such that $|S^A(I_r)| \geq |S(I)|$. Therefore

$$\frac{|S(I)|}{|C^*(I)|} \leq \frac{\sum U_i^A(I_O)}{|C^*(I)|} \leq (1-c)\ln m.$$

Since the number of machines m in the concurrent open shop instance I_O equals $n = |V|$ we have

$$\frac{|S(I)|}{|C^*(I)|} \leq (1-c)\ln n,$$

which proves that A is a $(1-c)\ln n$-approximation algorithm for the set cover problem which runs in polynomial time. This however contradicts Theorem 5.13. □

A similar inapproximability result was obtained by Ng et al. [22] under a stronger than the $P \neq NP$ assumption. The result was based on the inapproximability result for set cover obtained by Feige [9]. Garg et al. [13] present further complexity results for $O|\text{cncnt}|\sum w_i C_i$ and its special cases.

5.6 Fixed Number of Machines and Special Cases

Cheng et al. [5] give a PTAS for the problem $Om|\text{cncnt}|\sum w_i C_i$ where the number of machines m is not part of the input. Ahmadi et al. [1] report a $\frac{\sqrt{5}+3}{\sqrt{5}+1}$-approximation for the two-machine problem $O2|\text{cncnt}|\sum w_i Ci$, see also Roemer [23]. Ahmadi et al. [1] propose heuristics and report computational experiments with the heuristics for $O|\text{cncnt}|\sum w_i C_i$. Cheng and Wang [6] give a pseudopolynomial-time algorithm for the problem $Om|\text{cncnt}|\sum w_i U_i$ where the number of machines m is not part of the input. This implies that $Om|\text{cncnt}, p_{ij} = 0,1|\sum U_i$ is polynomial. Leung et al. [19] give a polynomial-time algorithm $Om|\text{cncnt}|\sum U_i$ for job-ordered open shops, see Sect. 7.4 for definition of job-ordered open shops. We observe that $O1|\text{cncnt}|\sum U_i$ is the same as the single machine problem $1||\sum U_i$ which is solved by Hodgson–Moore algorithm, see Moore [21], in $O(n \log n)$ time. Besides, we observe that operation processing times follow the same order on each machine in job-ordered open shops. These two observations suffice to prove that $Om|\text{cncnt}|\sum U_i$ for job-ordered open shops, see Problem 5.5. Leung et al. [19] give an exact algorithm based on constraint propagation and bounding approach for $O|\text{cncnt}|\sum U_i$ and report on computational experiments with the algorithm. Framinan and Perez-Gonzalez [10] propose

heuristics for $O|\text{cncnt}|\sum T_i$ and report on computational experiments with the heuristics.

5.7 Conflict Graphs and the Classification of Open Shops

The open shop scheduling problems can be defined and classified by using the concept of operation conflict graphs introduced in Chap. 2. The operation conflict graph determines which two operations cannot be done in parallel in any feasible schedule. That is, if (o, o') is an edge in the graph, then operations o and o' can never be processed in parallel in a feasible schedule. The classification of open shops can then be done according the characteristic of the set of edges of the conflict graph. To be more precise, let $\mathcal{O}$ be the set of all operations $O_{i,h}$, where $J_i \in \mathcal{J}$ and $M_h \in \mathcal{M}$. An operation conflict graph is a simple graph $\mathcal{C} = (\mathcal{O}, \mathcal{E})$ with the set of vertices $\mathcal{O}$ and the set of edges in $\mathcal{E}$ linking operations in $\mathcal{O}$. We suggest the following classification depending on $\mathcal{E}$. The classification can be easily extended to other classes of $\mathcal{E}$.

- Let $\mathcal{O}_h = \{O_{1,h}, \ldots, O_{n,h}\}$, and let $\mathcal{K}_h$ be a clique of size n on $\mathcal{O}_h$, $h = 1, \ldots, m$. The union $\mathcal{E} = \mathcal{K}_1 \cup \cdots \cup \mathcal{K}_m$ of m disjoint cliques is a conflict graph of concurrent open shop.
- Let $\mathcal{J}_i = \{O_{i,1}, \ldots, O_{i,m}\}$, and let $\mathcal{G}_i$ be a clique of size m on $\mathcal{J}_i$, $i = 1, \ldots, n$. The union $\mathcal{E} = \mathcal{K}_1 \cup \cdots \cup \mathcal{K}_m \cup \mathcal{G}_1 \cup \cdots \cup \mathcal{G}_n$ of $n + m$ cliques is a conflict graph of open shop.
- Let $\mathcal{R}$ be any non-empty set of edges $(O_{i,h}, O_{j,\ell})$ such that $i \neq j$ and $h \neq \ell$. The union $\mathcal{E} = \mathcal{K}_1 \cup \cdots \cup \mathcal{K}_m \cup \mathcal{G}_1 \cup \cdots \cup \mathcal{G}_n \cup (\mathcal{O}, \mathcal{R})$ is a conflict graph of open shop with additional resources in Chap. 2.
- Let $\mathcal{P}$ be any non-empty set of edges $(O_{i,h}, O_{j,\ell})$ such that $h \neq \ell$. The union $\mathcal{E} = \mathcal{K}_1 \cup \cdots \cup \mathcal{K}_m \cup (\mathcal{O}, \mathcal{P})$ is a conflict graph of partially concurrent open shop. Partially concurrent open shops are studied by Ilani et al. [16] and Grinshpoun et al. [15].

Problems

5.1 Prove Theorem 5.1 for $\sum_i U_i$ and $\sum_i T_i$.

5.2 Show that the performance guarantee of the 2-approximation algorithm in Sect. 5.4 cannot be better than $2 - \frac{2}{n+1}$.

5.3 Find an optimal schedule for an instance in Table 5.2.

5.4 Proof Theorem 5.14 for $O|cncnt|\sum T_i$.

5.5 Show that the problem $Om|\text{cncnt}|\sum U_i$ is polynomial for job-ordered open shops.

References

1. R.H. Ahmadi, U. Bagchi, T.A. Roemer, Coordinated scheduling of customer orders for quick response. Naval Res. Logist. **52**, 483–512 (2005)
2. R.H. Ahmadi, U. Bagchi, *Scheduling of Multi-Job Customer Orders in Multimachine Environments* (ORSA/TIMS, Philadelphia, 1990)
3. N. Bansal, S. Khot, Inapproximability of hypergraph vertex cover and applications to scheduling problems, in *Automata, Languages and Programming. ICALP 2010*, ed. by S. Abramsky, C. Gavoille, C. Kirchner, F. Meyer auf der Heide, P.G. Spirakis. Lecture Notes in Computer Science, vol. 6198 (Springer, Berlin, 2010), pp. 250–261
4. Z.L. Chen, N.G. Hall, Supply chain scheduling: assembly systems. Working paper, Department of Systems Engineering, University of Pennsylvania, 2001
5. T.C.E. Cheng, Q. Nong, C.T. Ng, Polynomial-time approximation scheme for concurrent open shop scheduling with a fixed number of machines to minimize the total weighted completion time. Naval Res. Logist. **58**, 763–770 (2011)
6. T.C.E. Cheng, G. Wang, Customer order scheduling on multiple facilities. Working paper no. 11/98-9, Faculty of Business and Information Systems, The Hong Kong Polytechnic University, 1999
7. I. Dinur, V. Guruswami, S. Khot, O. Regev, A new multilayered PCP and the hardness of hypergraph vertex cover. SIAM J. Comput. **34**, 1129–1146 (2005)
8. I. Dinur, D. Steurer, Analytical approach to parallel repetition, in *Proceedings of the 2014 ACM symposium on Theory of Computing. STOC'14* (ACM, Berlin, 2014), pp. 624–633
9. U. Feige, A threshold of $\ln n$ for aproximating set cover. J. ACM **45**, 634–652 (1998)
10. J.M. Framinan, P. Perez-Gonzalez, Order scheduling with tardiness objective: improved approximate solutions. Eur. J. Oper. Res. **266**, 840–850 (2018)
11. J.M. Framinan, P. Perez-Gonzalez, V. Fernandez-Viagas, Deterministic assembly scheduling problems: a review and classification of concurrent-type scheduling models and solution procedures. Eur. J. Oper. Res. **273**, 401–417 (2019)
12. M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W. H. Freeman, San Francisco, 1979)
13. N. Garg, A. Kumar, V. Pandit, Order scheduling models: hardness and algorithms, in *Lecture Notes in Computer Science 4855* (Springer, Berlin, 2007), pp. 96–107
14. R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discr. Math. **5**, 287–326 (1979)
15. T. Grinshpoun, H. Ilani, E. Shufan, The representation of partially-concurrent open shop problems. Ann. Oper. Res. **252**, 455–469 (2017)
16. H. Ilani, E. Shufan, T. Grinshpoun, Partially concurrent open shop scheduling with integral preemtions. Ann. Oper. Res. **259**, 157–171 (2017)
17. J.R. Jackson, Scheduling a production line to minimize maximum tardiness. Management science research project, research report 43, UCLA, 1955
18. S. Khot, On the power of unique 2-prover 1-round games, in *Proceedings pf the 34th Annual ACM Symposium on Theory of Computing* (2002), pp. 767–775
19. J.Y.-T. Leung, H. Li, M. Pinedo, Scheduling orders for multiple product types with due date related objectives. Eur. J. Oper. Res. **168**, 370–389 (2006)
20. M. Mastrolilli, M. Queyranne, A.S. Schulz, O. Svensson, N.A. Uhan, Minimizing the sum of weighted completion times in a concurrent open shop. Oper. Res. Lett. **38**, 390–395 (2010)

21. J.M. Moore, An *n* job, one machine sequencing algorithm for minimizing the number of late jobs. Manag. Sci. **15**, 102–109 (1968)
22. C.T. Ng, T.C.E. Cheng, J.J. Yuan, Concurrent open shop scheduling to minimize the weighted number of tardy jobs. J. Sched. **6**, 405–412 (2003)
23. T.A. Roemer, A note on establishing heuristic bounds by instance construction. Technical report, Sloan School at MIT, Cambridge, MA, 2004
24. T.A. Roemer, A note on the complexity of the concurrent open shop scheduling problem. J. Sched. **9**, 389–396 (2006)
25. A.S. Schulz, Scheduling to minimize total weighted completion time: performance guarantees of lp-based heuristics and lower bounds, in *Integer Programming and Combinatorial Optimization, IPCO 1996*, ed. by W.H. Cunningham, S.T. McCormick, M. Queyranne. Lecture Notes in Computer Science (Springer, Berlin, 1996), pp. 301–315
26. E. Wagneur, C. Sriskandarajah, Open shops with jobs overlap. Eur. J. Oper. Res. **71**, 366–378 (1993)