J Comb Optim (2013) 26:345–371
DOI 10.1007/s10878-012-9466-y

Optimal job insertion in the no-wait job shop

Reinhard Bürgy · Heinz Gröflin

Published online: 16 March 2012

Abstract The no-wait job shop (NWJS) considered here is a version of the job shop scheduling problem where, for any two operations of a job, a fixed time lag between their starting times is given. Also, sequence-dependent set-up times between consecutive operations on a machine can be present. The NWJS problem consists in finding a schedule that minimizes the makespan.

We address here the so-called optimal job insertion problem (OJI) in the NWJS. While the OJI is NP-hard in the classical job shop, it was shown by Gröflin & Klinkert to be solvable in polynomial time in the NWJS. We present a highly efficient algorithm with running time $\mathcal{O}(n^2 \cdot \max\{n, m\})$ for this problem. The algorithm is based on a compact formulation of the NWJS problem and a characterization of all feasible insertions as the stable sets (of prescribed cardinality) in a derived comparability graph.

As an application of our algorithm, we propose a heuristic for the NWJS problem based on optimal job insertion and present numerical results that compare favorably with current benchmarks.

Keywords No-wait job shop · Optimal job insertion · Stable sets · Comparability graph

1 Introduction

Job insertion in job shop scheduling problems is the process of inserting a job in a given “schedule” of other jobs specified by their sequences on the machines. The

R. Bürgy (✉) · H. Gröflin
Department of Informatics, University of Fribourg, Boulevard de Pérolles 90, 1700 Fribourg, Switzerland
e-mail: reinhard.buergy@unifr.ch

H. Gröflin
e-mail: heinz.groeflin@unifr.ch

optimal job insertion problem (OJI) consists in finding a job insertion minimizing some objective, usually makespan or maximum tardiness.

Job insertion is of interest for applications as well as for method development and scheduling theory. In some environments, jobs arrive successively and scheduling is done in a rolling fashion, inserting a job at its arrival into the current schedule.

Methodologically, job insertion has been used by several authors as a tool in devising scheduling heuristics. These methods build up a schedule by successive job insertion or improve it by repeatedly extracting and reinserting a job. As examples, we mention in the classical job shop the work of Werner and Winkler (1995), in the blocking job shop Gröflin and Klinkert (2009) and in the no-wait job shop Schuster (2006). More references will be given in Sect. 5.

In these methods, a job is usually not inserted optimally. Indeed, finding an optimal job insertion is in general a nontrivial problem in scheduling theory. A pioneering contribution was the work by Kis (2001) and Kis and Hertz (2003) on the OJI-JS, the optimal job insertion problem in the classical job shop. They gave a polyhedral characterization of the family of feasible job insertions as well as a procedure yielding lower and upper bounds for the OJI-JS, which is a NP-hard problem.

In a different approach, we defined in Gröflin and Klinkert (2007) more generally "insertions" (of a set of operations) as selections in so-called (disjunctive) insertion graphs. We showed that if these graphs have certain properties, the feasible insertions can be characterized as stable sets (of prescribed cardinality) in a bipartite graph and devised an algorithm for deriving lower and upper bounds for the optimal insertion problem. These general results apply in particular to *job* insertion, and not only in the classical job shop. As examples, we examined job insertion in the multi-processor task job shop, the blocking job shop and the no-wait job shop. Moreover we showed that the algorithm for deriving lower and upper bounds solves *optimally* the OJI-NWJS, the optimal job insertion problem in the no-wait job shop.

This last result established that the OJI-NWJS is solvable in polynomial time. It provided the motivation for this work to study the OJI-NWJS in more detail and to find a specialized algorithm for this problem that is more efficient from both a theoretical and practical point of view. Indeed, the algorithm of Gröflin and Klinkert (2007) is weakly polynomial, due to a binary search component. Also, its computation time is relatively high, precluding its frequent use as a subroutine in good heuristics for the NWJS problem.

We propose in this paper an algorithm for the OJI-NWJS that is strongly polynomial and highly efficient, so that optimal job insertion can be implemented with reasonable computational effort in constructive and improving heuristics for the NWJS problem. The algorithm necessitates some structural developments which seem to be also of interest for themselves and which are sketched below.

The paper is organized as follows. The next section describes the NWJS problem and gives a classical disjunctive graph formulation. In Sect. 3, a new compact formulation of the NWJS problem is derived that captures two key properties of the NWJS. In Sect. 4, the OJI-NWJS is first formulated in a compact insertion graph. It is then shown that, for any bound ρ, the family of all job insertions yielding a makespan smaller than ρ is in 1-to-1 correspondence with the stable sets (of prescribed cardinality) of a derived graph H^ρ which turns out to be a comparability graph. These results and some additional properties (of antichains in a poset) form the main ingredients of the OJI-NWJS algorithm. In Sect. 5, as an application, a new heuristic for the NWJS problem based on the OJI-NWJS algorithm is proposed and numerical results are presented showing that the method is competitive. The Appendix provides a detailed implementation of the algorithm and a complexity analysis.

We conclude this introduction with some notation and terminology. All graphs will be directed and the following standard notation will be used. In the graph $G = (V, E)$, an arc $e \in E$ has a *tail* (node) $t(e)$ and a *head* $h(e)$. For any disjoint sets $M, N \subseteq V$, $\delta(M, N) = \{e \in E : t(e) \in M \text{ and } h(e) \in N\}$, and for any $N \subseteq V$, $\delta(N) = \delta(N, V - N) \cup \delta(V - N, N)$. If an arc length vector $c \in R^E$ is given, G will be denoted by the triplet $G = (V, E, c)$. Sometimes a triplet alone is used to identify a graph, usually a subgraph of a given graph. In $G = (V, E, c)$, a cycle is called *positive* if its length is positive. Finally, some concepts defined for undirected graphs, such as clique, stable set and comparability graph, are used with directed graphs, with the meaning that they apply to the corresponding undirected graphs obtained by ignoring arc orientation.

2 The no-wait job shop

2.1 Formulation, notation and data

The NWJS can be described as follows. Let I be a set of operations $i \in I$ and $\mathscr{J} \subseteq 2^I$ a set of jobs such that $\mathscr{J}$ forms a partition of I, i.e. a job $J \in \mathscr{J}$ is a set of operations $\{i : i \in J\}$ and any operation $i \in I$ is in exactly one job $J \in \mathscr{J}$. We assume that the set of operations of a job is ordered in a sequence and denote sometimes $\{i : i \in J\}$ as the ordered set $\{J_1, J_2, \ldots, J_{|J|}\}$, J_r denoting the r-th operation of job J. Two operations i, j of job J are consecutive if $i = J_r$ and $j = J_{r+1}$ for some r, $1 \le r < |J|$. Furthermore, let M be a set of machines. Each operation $i \in I$ needs a specific machine, say $m_i \in M$, for its execution of duration $p_i > 0$.

In the NWJS considered here, for each job $J \in \mathscr{J}$ and any two operations i and $j \in J$, a *fixed time lag* γ_{ij} of arbitrary sign is imposed between the starting times of i and j. Without loss of generality, we may assume that the order of operations $J_1, J_2, \ldots, J_{|J|}$ of job J is such that $\gamma_{J_r, J_{r+1}} \ge 0$, $1 \le r < |J|$. Also, only time lags between consecutive operations need to be specified since for any $i = J_s$ and $j = J_t$ with $s < t$, $\gamma_{ij} = \sum_{s \le r < t} \gamma_{J_r, J_{r+1}}$ and $\gamma_{ji} = -\gamma_{ij}$. Note that this feature slightly generalizes the no-wait constraint present in the "classical" no-wait job shop (case where $\gamma_{ij} = p_i$ for a pair i, j of consecutive operations of a job) and allows to capture some scheduling problems arising typically in the process industry, e.g. in the pharmaceutical and chemical sector.

Another feature of the NWJS considered here is to allow for sequence-dependent set-up times: if i and j are two operations on a same machine and j follows i, then a *set-up* of duration s_{ij} occurs between completion of i and start of j. Also, for each operation i, an *initial set-up* of duration $s_{\sigma i}$ can be specified, signifying that the starting time of i is at least $s_{\sigma i}$ (earliest starting time), as well as a *final set-up* (or tail) of duration $s_{i\tau}$, meaning that a time of at least $s_{i\tau}$ lapses between completion time of i and the overall finish time (makespan).

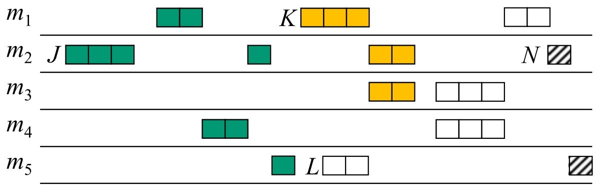

Fig. 1 Example with four jobs and five machines

The NWJS problem consists in finding starting times for all operations so that each machine is occupied by at most one operation at a time and the makespan is minimal. In the three-field notation, the problem could be denoted as $J|s_{ij}, tl|C_{\max}$, where s_{ij} refers to the sequence-dependent setup times, and tl to the fixed time lags.

Figure 1 displays in a Gantt chart an example with four jobs J, K, L, N and five machines $m_1, \dots, m_5$. For simplicity, no set-ups are present. The numerical data can be read directly in the chart, e.g. for J, the duration of its first operation is 3 and the time lag between its first and second operation is 4. The example will be used in the sequel.

2.2 A disjunctive graph formulation

The disjunctive graph $G = (I^+, A, E, \mathscr{E}, d)$ for the NWJS is constructed as follows. Each operation $i \in I$, as well as a fictive start operation σ and end operation τ is represented by a node. σ and τ are of duration 0 and must occur before, respectively after, all operations of I. We identify a node with the operation it represents and denote the node set by $I^+ = I \cup \{\sigma, \tau\}$.

The set A of conjunctive arcs consists of the following arcs: (i) for each $i \in I$, an *initial set-up arc* (σ, i) and a *final set-up arc* (i, τ) of respective length $d_{\sigma i} = s_{\sigma i}$ and $d_{i\tau} = p_i + s_{i\tau}$; (ii) for each job J and each ordered pair of consecutive operations $i, j \in J$, a pair of arcs (i, j) and (j, i) with respective length $d_{ij} := \gamma_{ij}$ and $d_{ji} := \gamma_{ji} = -d_{ij}$.

The set E of disjunctive arcs consists of all arcs (i, j) and (j, i) between operations i and j on a same machine and of different jobs. Formally, define for all $m \in M$, $I_m := \{i \in I : m_i = m\}$ and $E_m := \{(i, j) : i, j \in I_m$ such that $i \in J$, $j \in J' \Rightarrow J \neq J'\}$. Then $E := \bigcup_{m \in M} E_m$. The lengths are $d_{ij} = p_i + s_{ij}$ for all $(i, j) \in E$.

For any $m \in M$ and $i, j \in I_m$, arcs (i, j) and (j, i) form a (unordered) *pair* of disjunctive arcs. The family $\mathscr{E}$ is the collection of all such pairs. A general element of $\mathscr{E}$, i.e. a pair of disjunctive arcs, will be denoted by $\{e, \overline{e}\}$.

The following definitions in $G = (I^+, A, E, \mathscr{E}, d)$ will be useful. Any subset of disjunctive arcs $S \subseteq E$ is called a *selection*. A selection S is *positive acyclic* if the subgraph $G(S) = (I^+, A \cup S, d)$ contains no positive cycle, and is *positive cyclic* otherwise. A selection S is *complete* if $S \cap \{e, \overline{e}\} \neq \emptyset$ for all $\{e, \overline{e}\} \in \mathscr{E}$. A selection S is *feasible* if it is positive acyclic and complete.

The NWJS can be formulated as follows: "Among all feasible selections, find a selection S minimizing the length of a longest path from σ to τ in $G(S) = (I^+, A \cup S, d)$".

3 A compact formulation of the NWJS

In this section, we will define for any selection S a so-called job-graph $F(S)$ which can be seen as a compact representation of $G(S)$. Theorem 1 will formalize this notion and two structural properties will follow as corollaries. Then a disjunctive job-graph F will be introduced which leads to a compact formulation of the NWJS.

3.1 The job-graph $F(S)$

Given any selection $S \subseteq E$ in the disjunctive graph $G = (I^+, A, E, \mathscr{E}, d)$ and any distinct $J, K \in \mathscr{J}$, define the subsets

$$\begin{aligned} &S_{JK} = S \cap \delta(J, K), S_{KJ} = S \cap \delta(K, J) \\ &S_{[JK]} = S_{JK} \cup S_{KJ} \end{aligned} \tag{1}$$

and distances

$$c^S_{JK} = \max\{\gamma_{J_1,i} + d_{ij} + \gamma_{j,K_1} : (i, j) \in S_{JK}\}, \tag{2}$$

convening $c^S_{JK} = -\infty$ if $S_{JK} = \emptyset$ and $\gamma_{ii} = 0$ for all $i \in I$.

Observe that $\bigcup_{J,K\in\mathscr{J}} S_{[JK]}$ is a partition of S. Also, a distance c^S_{JK} $(> -\infty)$ is the length of a longest path in the subgraph $(I^+, A \cup S_{JK}, d)$ from the first operation J_1 of J to the first operation K_1 of K, and similarly $c^S_{KJ} > -\infty$ is the length of a longest path in $(I^+, A \cup S_{KJ}, d)$ from K_1 to J_1. For future use, we denote by e_{JK} the arc $(i, j) \in S_{JK}$ being the argument in (2).

The distances from σ to J and from J to τ for all $J \in \mathscr{J}$ are defined as:

$$c_{\sigma J} = c^S_{\sigma J} = \max\{d_{\sigma i} + \gamma_{i,J_1} : i \in J\}, \tag{3}$$

$$c_{J\tau} = c^S_{J\tau} = \max\{\gamma_{J_1,i} + d_{i\tau} : i \in J\}. \tag{4}$$

Note that the superscript S can be omitted here since $c^S_{\sigma J}$ and $c^S_{J\tau}$ do not depend on S; it has been kept as we also refer to the length vector c^S in the sequel. For future use, the arcs (σ, i) and (i, τ) for the indexes i being the arguments in (3) and (4) are denoted $e_{\sigma J}$ and $e_{J\tau}$.

Let $F(S) = (\mathscr{J}^+, B \cup U(S), c^S)$ be the digraph whose node set $\mathscr{J}^+ = \mathscr{J} \cup \{\sigma, \tau\}$ comprises a node for each job $J \in \mathscr{J}$, and two nodes representing σ and τ. We will identify these nodes with J, σ and τ. The arc set B consists of the arcs (σ, J) and (J, τ) for all $J \in \mathscr{J}$, with length $c^S_{\sigma J}$ and $c^S_{J\tau}$ given by (3) and (4). The arc set $U(S)$ comprises arcs (J, K) and (K, J) for all distinct J and $K \in \mathscr{J}$, with length c^S_{JK} and c^S_{KJ} given by (2). $F(S)$ will be called the *job-graph with respect to S*.

Figure 2 illustrates on the left the subgraph $G(S)$ for the selection S corresponding to the schedule displayed in Fig. 1. For clarity, only the arcs from σ and to τ that are incident to J have been drawn. The job-graph $F(S)$ is depicted on the right.

Theorem 1 (i) *$G(S) = (I^+, A \cup S, d)$ contains no positive cycle if and only if $F(S) = (\mathscr{J}^+, B \cup U(S), c^S)$ contains no positive cycle.* (ii) *Assuming $G(S)$ and*

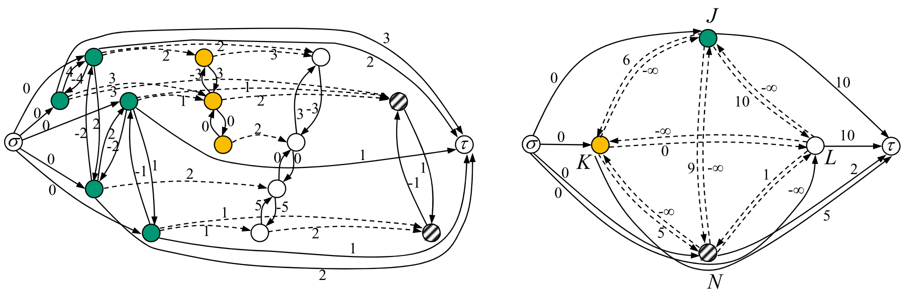

Fig. 2 Subgraph $G(S)$ and $F(S)$

$F(S)$ contain no positive cycle, the length of a longest σ-τ-path is the same in both graphs. If P with node sequence $\sigma, J^1, \ldots, J^p, \tau$ is a longest σ-τ-path in $F(S)$, the σ-τ-path P' in $G(S)$ visiting $J^1, \ldots, J^p$, entering J^1 through arc $e_{\sigma J^1}$, and, for $1 < r \le p$, entering J^r through $e_{J^{r-1}J^r}$, and exiting J^p through $e_{J^p\tau}$ is a longest σ-τ-path in $G(S)$.

Proof (i) By a classic result of combinatorial optimization (see for instance Cook et al. (1997), p. 25), $G(S)$ contains no positive cycle if and only if there exists a feasible potential, i.e. $y' \in R^{I^+}$ such that $y'_j - y'_i \ge d_{ij}$ for all $(i,j) \in A \cup S$, and similarly, $F(S)$ contains no positive cycle if and only if there exists $y \in R^{\mathscr{J}^+}$ such that $y_w - y_v \ge c^S_{vw}$ for all $(v,w) \in B \cup U(S)$. We show that $G(S)$ has a feasible potential if and only if $F(S)$ has a feasible potential.

Let $y' \in R^{I^+}$ be a feasible potential in $G(S)$. Then clearly, for any $J \in \mathscr{J}$,

$$y'_j - y'_i = \gamma_{ij} \quad \text{for all } i, j \in J. \tag{5}$$

Define $y \in R^{\mathscr{J}^+}$ by:

$$y_\sigma = y'_\sigma, \quad y_\tau = y'_\tau \quad \text{and} \quad y_J = y'_{J_1} \quad \text{for all } J \in \mathscr{J}, \tag{6}$$

where we recall that J_1 denotes the first operation of J. Then y is a feasible potential in $F(S)$. Indeed, let $i \in J$ be such that $c^S_{\sigma J} = d_{\sigma i} + \gamma_{i,J_1}$. By (5), $y_J = y'_{J_1} = y'_i + \gamma_{i,J_1}$. Then $y_J - y_\sigma = y'_i + \gamma_{i,J_1} - y_\sigma = y'_i - y'_\sigma + \gamma_{i,J_1} \ge d_{\sigma i} + \gamma_{i,J_1} = c^S_{\sigma J}$. Hence $y_J - y_\sigma \ge c^S_{\sigma J}$ for all $J \in \mathscr{J}$. Similarly, one shows $y_\tau - y_J \ge c^S_{J\tau}$ for all $J \in \mathscr{J}$. Finally, given any distinct J and K, let $i \in J$ and $j \in K$ be such $(i,j) \in S_{JK}$ and $c^S_{JK} = \gamma_{J_1,i} + d_{ij} + \gamma_{j,K_1}$. By (5), $y'_{K_1} = y'_j + \gamma_{j,K_1}$ and $-y'_{J_1} = \gamma_{J_1,i} - y'_i$. Then $y_K - y_J = y'_{K_1} - y'_{J_1} = y'_j - y'_i + \gamma_{j,K_1} + \gamma_{J_1,i} \ge d_{i,j} + \gamma_{J_1,i} + \gamma_{j,K_1} = c^S_{JK}$. Hence for any feasible potential y' in $G(S)$, there is a feasible potential y in $F(S)$ given by (6).

Conversely, let $y \in R^{\mathscr{J}^+}$ be a feasible potential in $F(S)$. Define $y' \in R^{I^+}$ by:

$$y'_\sigma = y_\sigma, y'_\tau = y_\tau \quad \text{and for all } J \in \mathscr{J}: \tag{7}$$

$$y'_{J_1} = y_J \quad \text{and} \quad y'_i = y'_{J_1} + \gamma_{J_1,i} \quad \text{for all } i \in J - \{J_1\}. \tag{8}$$

We show that y' is a feasible potential in $G(S)$. First, $y'_j - y'_i \geq d_{ij}$ for all $(i,j) \in A$. Indeed,

$$y'_j - y'_i = d_{ij} \quad \text{if both } i \text{ and } j \in J \text{ for some } J \in \mathscr{J}. \tag{9}$$

Also, $y'_i - y'_\sigma \geq d_{\sigma i}$ for all $i \in I$, since $y'_i - y'_\sigma = y_J + \gamma_{J_1,i} - y_\sigma \geq c^S_{\sigma J} + \gamma_{J_1,i} \geq (d_{\sigma i} + \gamma_{i,J_1}) + \gamma_{J_1,i} = d_{\sigma i}$, where the last inequality follows from the definition of $c^S_{\sigma J}$. Similarly, $y'_\tau - y'_i \geq d_{i\tau}$ for all $i \in I$, since, $y'_\tau - y'_i = y_\tau - (y_J + \gamma_{J_1,i}) \geq c^S_{J\tau} - \gamma_{J_1,i} \geq \gamma_{J_1,i} + d_{i\tau} - \gamma_{J_1,i} = d_{i\tau}$. Finally, consider any $(i,j) \in S$. $(i,j) \in S_{JK}$ for some J, K. Then $y'_j - y'_i = y_K + \gamma_{K_1,j} - (y_J + \gamma_{J_1,i}) \geq c^S_{JK} + \gamma_{K_1,j} - \gamma_{J_1,i} \geq (\gamma_{J_1,i} + d_{ij} + \gamma_{j,K_1}) + \gamma_{K_1,j} - \gamma_{J_1,i} = d_{ij}$. Hence for any feasible potential y in $F(S)$, there is a feasible potential y' in $G(S)$ given by (7)–(8).

Proof of (ii). Assume $G(S)$ and $F(S)$ contain no positive cycle and let ω_G and ω_F be the length of a longest σ-τ-path in $G(S)$ and $F(S)$ respectively. We show $\omega_G = \omega_F$. For all $v \in I^+$, let y'_v be the length of a longest σ-v-path in $G(S)$. y' is a feasible potential in $G(S)$ with $y'_\sigma = 0$ and $y'_\tau = \omega_G$. Hence y defined by (6) is a feasible potential in $F(S)$ with $y_\sigma = 0$, $y_\tau = y'_\tau$. This potential yields an upper bound y_τ on ω_F, therefore $\omega_G = y'_\tau = y_\tau \geq \omega_F$. Similarly, $\omega_F \geq \omega_G$ is shown using a "longest σ-w-path-potential" y in $F(S)$ and deriving from it with (7)–(8) a feasible potential y' in $G(S)$.

Finally, we prove that P' constructed from path P as indicated, is a longest σ-τ-path in $G(S)$. First note that P' is fully specified since its entry and exit arcs into and out of a job are specified, and, if P' enters a job through, say, arc (i,j) and leaves it through arc (k,l), the subpath of P' from j to k in the job is unique. Let y in $F(S)$ be the "longest σ-w-path potential" in $F(S)$ and y' in $G(S)$ be determined from y by (7)–(8). We prove that $d(P') = y'_\tau = \omega_F = \omega_G$ by showing that for any arc (v,w) of P', $y'_w - y'_v = d_{vw}$. This is easily shown, using that (a) in $F(S)$, $y_u - y_t = c^S_{tu}$ for each arc (t,u) of P, (b) equalities (9) and (c) the choice of the arcs $e_{\sigma J^1}$, $e_{J^{r-1}J^r}$, $1 < r \leq p$, and $e_{J^p\tau}$. Since the derivation is similar to the feasibility proof for y' above, the details are omitted. □

From the theorem follow immediately two key properties of G derived in Gröflin and Klinkert (2007) and stated here as a corollary.

Corollary 1 (i) *For any positive cycle Z in $(I^+, A \cup E, d)$, there exists a positive cycle Z' with $Z' \cap E \subseteq Z \cap E$ and Z' visits each job at most once.* (ii) *For any feasible selection S, there exists a longest σ-τ-path in $G(S)$ visiting each job at most once.*

Proof (i) Choose selection $S = Z$. Since S is positive cyclic, by Theorem 1, $F(S)$ contains a positive cycle which is expanded (similarly to the expansion of path P into P') into a positive cycle Z' with $Z' \cap E \subseteq Z \cap E$ and Z' visits each job at most once.

(ii) P' in Theorem 1 is such a path. □

3.2 A compact disjunctive graph formulation

Given distinct $J, K \in \mathscr{J}$, let $S^p_{[JK]} \subseteq \delta(J,K) \cup \delta(K,J)$, $p = 1,\dots,q_{JK}$, be all selections that are positive acyclic and complete on $\delta(J,K) \cup \delta(K,J)$, i.e. $S^p_{[JK]} \cap \{e,\bar{e}\} \neq \emptyset$ for all $\{e,\bar{e}\} \subseteq \delta(J,K) \cup \delta(K,J)$. In other words, selections $S^p_{[JK]}$, $p = 1,\dots,q_{JK}$, represent all feasible ways of positioning J and K with respect to each other.

The number q_{JK} of these selections is "small". If r_{J_m} denotes the number of operations of J on machine m, it is easy to see that $q_{JK} \leq 1 + \sum_{m \in M} r_{J_m} \cdot r_{K_m}$. In particular, in the case of a classical NWJS where $r_{J_m} \leq 1$ for all $J \in \mathcal{J}$ and $m \in M$, $q_{JK} \leq |M| + 1$.

We may assume that the $S^p_{[JK]}$'s are indexed with $p = 1,\dots,q_{JK}$ in such a way that

$$\begin{aligned} &S^1_{JK} = S^1_{[JK]},\ S^1_{KJ} = \emptyset, \\ &S^p_{JK} \supset S^q_{JK} \text{ and } S^p_{KJ} \subset S^q_{KJ} \text{ for } 1 \leq p < q \leq q_{JK}, \\ &S^{q_{JK}}_{JK} = \emptyset,\ S^{q_{JK}}_{KJ} = S^{q_{JK}}_{[JK]} \end{aligned} \tag{10}$$

i.e. $S^1_{[JK]}$ places job J "fully before" K, and, with increasing p, K moves ahead of an operation of J on some machine, until with selection $S^{q_{JK}}_{[JK]}$, K is fully before J.

Figure 3 illustrates these selections for the two jobs J and K in the example. The three positionings of J and K are depicted in the Gantt charts (left) and the corresponding selections $S^p_{[JK]}$, $p = 1, 2, 3$ are shown in the center (sets of dashed arcs).

Proposition 1 *Let c^p_{JK} and c^p_{KJ} be the lengths c^S_{JK} and c^S_{KJ} defined by (2) for $S = S^p_{[JK]}$, $p = 1,\dots,q_{JK}$. The following holds.*

(i) $c^1_{JK} > c^2_{JK} > \cdots > c^{q_{JK}}_{JK}$ *and* $c^1_{KJ} < c^2_{KJ} < \cdots < c^{q_{JK}}_{KJ}$,
(ii) $c^p_{JK} + c^p_{KJ} \leq 0$ *for* $p = 1,\dots,q_{JK}$,
(iii) $c^p_{JK} + c^q_{KJ} \leq 0$ *for* $1 \leq q < p \leq q_{JK}$,
(iv) $c^p_{JK} + c^q_{KJ} > 0$ *for* $1 \leq p < q \leq q_{JK}$.

Proof (i) Clearly, in view of (10), $c^p_{JK} \geq c^{p+1}_{JK}$ for $1 \leq p < q_{JK}$. Suppose $c^p_{JK} = c^{p+1}_{JK}$. Then there exist $(i,j) \in S^{p+1}_{JK}$ and $(k,l) \in S^p_{JK} - S^{p+1}_{JK}$ such that $c^p_{JK} = c^{p+1}_{JK} = \gamma_{J_1,i} + d_{ij} + \gamma_{j,K_1} \geq \gamma_{J_1,k} + d_{kl} + \gamma_{l,K_1}$, therefore $\gamma_{J_1,i} - \gamma_{J_1,k} + d_{ij} + \gamma_{j,K_1} - \gamma_{l,K_1} = \gamma_{ki} + d_{ij} + \gamma_{jl} \geq d_{kl}$. But then $\gamma_{ki} + d_{ij} + \gamma_{jl} + d_{lk} \geq d_{kl} + d_{lk} > 0$, and there is a positive cycle (through i, j, l and k) contained in $(I^+, A \cup S^{p+1}_{[JK]}, d)$, a contradiction to $S^{p+1}_{[JK]}$ being positive acyclic.

(ii) follows from the $S^p_{[JK]}$'s being positive acyclic and (iii) results from (i) and (ii). To show (iv), observe that for any $p < q$, $S^p_{[JK]} \cup S^q_{[JK]}$ is positive cyclic in G since $\{e,\bar{e}\} \subseteq S^p_{[JK]} \cup S^q_{[JK]}$ for $e \in S^p_{JK} - S^q_{JK}$, hence by Theorem 1, $\{(J,K)_p, (K,J)_p, (J,K)_q, (K,J)_q\}$ contains a positive cycle whose arcs, in view of ii) and iii), are $(J,K)_p$ and $(K,J)_q$, hence $c^p_{JK} + c^q_{KJ} > 0$. □

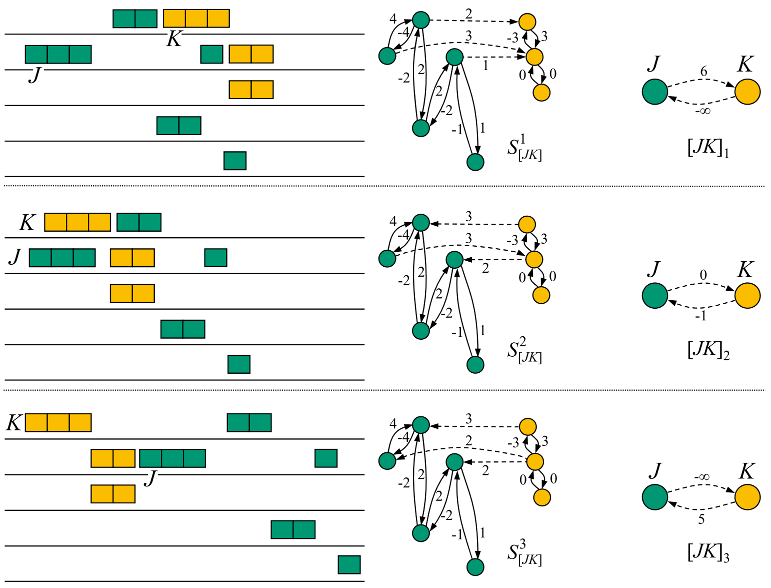

Fig. 3 Positionings of J and K (*left*) and corresponding selections (set of *dashed arcs*) in G (*center*) and F (*right*)

The *disjunctive job-graph* $F = (\mathscr{J}^+, B, U, \mathcal{P}, c)$ is now constructed. As in $F(S)$, the node set $\mathscr{J}^+ = \mathscr{J} \cup \{\sigma, \tau\}$ consists of all nodes representing a job or a fictive operation. The conjunctive arc set B comprises the arcs (σ, J) and (J, τ) with lengths $c_{\sigma J}$ and $c_{J\tau}$ defined by (3) and (4) for all $J \in \mathscr{J}$. The set U of disjunctive arcs comprises the following arcs. Between any distinct nodes J, K of $\mathscr{J}$, two arcs $(J, K)_p$ and $(K, J)_p$ with length c^p_{JK} and c^p_{KJ} are introduced for each $p = 1, \ldots, q_{JK}$. In the example, considering jobs J and K, two arcs $(J, K)_p$ and $(K, J)_p$ are introduced for $p = 1, 2, 3$. Each pair is displayed separately in Fig. 3, right. The disjunctive job-graph F is depicted in Fig. 4.

For any distinct J, K and $p \in \{1, \ldots, q_{JK}\}$, denote by $[J, K]_p$ the (unordered) *pair* of arcs $((J, K)_p, (K, J)_p) \in U \times U$. The pair $[J, K]_p$ represents selection $S^p_{[JK]}$. Let P be the set of all such pairs, i.e.

$$P = \{[J, K]_p : J, K \in \mathscr{J}, J \neq K \text{ and } 1 \leq p \leq q_{JK}\},$$

and for any distinct J and $K \in \mathscr{J}$, let

$$D_{JK} = \{[J, K]_p : 1 \leq p \leq q_{JK}\}.$$

The family $\mathscr{P}$ (of sets of arc pairs) is the family $\{D_{JK} : J, K \in \mathscr{J}, J \neq K\}$.

An F-*selection* is any set $T \subseteq P$ of arc pairs and $U_T \subseteq U$ denotes the set of all arcs used by T. An F-selection T is *complete* if $T \cap D_{JK} \neq \emptyset$ for all distinct $J, K \in \mathscr{J}$. T is *positive acyclic* if $F(T) = (\mathcal{J}^+, B \cup U_T, c)$ contains no positive cycle, and *feasible* if it is complete and positive acyclic.

Note that, by definition, a complete F-selection T contains at least one pair $[J, K]_p$ for any distinct jobs J and K. If additionally T is positive acyclic, i.e. if

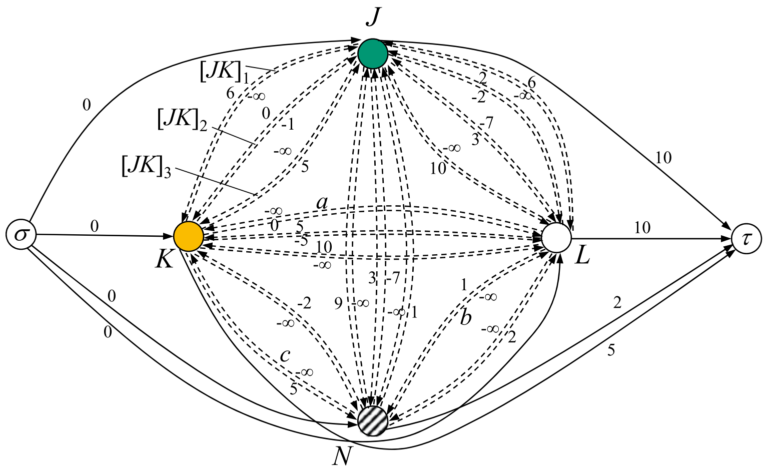

Fig. 4 The disjunctive job-graph F of the example

T is feasible, then by Proposition 1, T contains *exactly one* pair $[J, K]_p$ for distinct J and K.

Now, for any feasible selection S, distinct $J, K \in \mathscr{J}$ and $S_{[JK]}$ defined by (1), $S_{[JK]}$ is one of the selections $S^p_{[JK]}$, $p \in \{1, \dots, q_{JK}\}$, therefore to S corresponds an F-selection which is unique by construction, and is complete and positive acyclic. Conversely, to a complete and positive acyclic F-selection T corresponds a unique feasible selection S. (Note that Theorem 1 is used in both directions.)

The NWJS problem can therefore be formulated as the following problem in the disjunctive job-graph F: "Among all feasible F-selections, find an F-selection T minimizing the length of a longest path from σ to τ in the subgraph $F(T) = (\mathscr{J}^+, B \cup U_T, c)$".

4 Optimal job insertion in the NWJS

4.1 Formulation in the job insertion graph F^J

We studied in Gröflin and Klinkert (2007) general insertion problems whose so-called insertion graphs have certain properties (so-called through-connectedness and bi-connectedness). We showed that the optimal insertion problem in a bi-connected insertion graph can be solved in polynomial time, and mentioned as an application of this result the optimal job insertion problem in the NWJS (OJI-NWJS).

We formulate here the OJI-NWJS in the framework of our compact formulation of the NWJS and solve it with a highly efficient algorithm.

Inserting optimally a job can be thought of as the following problem. Given a feasible schedule for all other jobs, insert the job in such a way that the resulting schedule is feasible and its makespan is minimal.

In this section, we will consider the OJI-NWJS problem for a *specific* job $J \in \mathscr{J}$ and, for brevity, denote $\mathscr{J} - J$ by $\mathscr{J}^-$ and q_{JK} by q_K.

In the disjunctive job-graph F, a given feasible schedule for all other jobs $K \in \mathscr{J}^-$ is specified by an F-selection R that is positive acyclic and "complete", i.e. for any distinct $K, L \in \mathscr{J}^-$, $[K, L]_p \subseteq R$ for some $p \in \{1, \dots, q_{KL}\}$. Let $U^J = U \cap \delta(J)$ and $\mathscr{P}^J$ be the family of sets D_{JK} for all $K \in \mathscr{J}^-$. One can define the disjunctive graph $F^J = (\mathscr{J}^+, B \cup U_R, U^J, \mathscr{P}^J, c)$—where the restriction of c to $B \cup U_R \cup U^J$ is denoted again by c. F^J can be called the *insertion graph* for J and a F^J-selection T an *insertion* of job J. Note that T is a (positive acyclic, complete, feasible) insertion if and only if $T \cup R$ is a (positive acyclic, complete, feasible) F-selection in $F = (\mathscr{J}^+, B, U, \mathscr{P}, c)$.

Assume in the example that K, L and N are scheduled as in Fig. 1. The corresponding F-selection R consists of the arc pairs labeled a, b and c in the graph F of Fig. 4. The insertion graph F^J is obtained from F by retaining only a, b and c as arc pairs between K, L and N.

The OJI-NWJS can then be stated as follows: "Among all feasible insertions, find an insertion T minimizing the makespan, i.e. the length of a longest path from σ to τ in the subgraph $(\mathscr{J}^+, B \cup U_R \cup U_T, c)$".

4.2 The conflict graph H^ρ

In the graph $(\mathscr{J}^+, B \cup U_R, c)$ which contains no positive cycle, let l_{KL} be the length of a longest K-L-path for any nodes $K, L \in \mathscr{J}^+$, with the convention $l_{KL} = 0$ if $K = L$ and $l_{KL} = -\infty$ if $K \neq L$ and there is no K-L-path. Obviously,

$$l_{KL} + l_{LK} \leq 0 \quad \text{and} \quad l_{KL} + l_{LN} \leq l_{KN} \quad \text{for all } K, L, N \in \mathscr{J}^+. \tag{11}$$

Also, $l_{\sigma J} = c_{\sigma J}$ and $l_{J\tau} = c_{J\tau}$ where $c_{\sigma J}$ and $c_{J\tau}$ are defined by (3) and (4).

Definition 1 For any $\rho > l_{\sigma\tau}$, the conflict graph at ρ is the graph $H^\rho = (W^\rho, Y^\rho)$ with the node set W^ρ and the arc set Y^ρ defined by:

for all $p = 1, \dots, q_K$ and $K \in \mathscr{J}^-$:

$$w_K^p \in W^\rho \quad \Leftrightarrow \quad \begin{cases} c_{\sigma J} + c_{JK}^p + l_{K\tau} < \rho \text{ and} \\ l_{\sigma K} + c_{KJ}^p + c_{J\tau} < \rho \end{cases} \tag{12}$$

for all pairs of distinct nodes $w_K^p, w_L^q \in W^\rho$:

$$(w_K^p, w_L^q) \in Y^\rho \quad \Leftrightarrow \quad \begin{cases} c_{JK}^p + c_{LJ}^q + l_{KL} > 0 \text{ or} \\ c_{JK}^p + c_{LJ}^q + l_{K\tau} + l_{\sigma L} \geq \rho. \end{cases} \tag{13}$$

Three observations are in order. First, given any $K \in \mathscr{J}^-$, let α_K^ρ be the smallest $p \in \{1, \dots, q_K\}$ such that $c_{\sigma J} + c_{JK}^p + l_{K\tau} < \rho$, and β_K^ρ be the largest $p \in \{1, \dots, q_K\}$ such that $l_{\sigma K} + c_{KJ}^p + c_{J\tau} < \rho$. From Proposition 1 and (12) follows that the node subset $W_K^\rho := \{w_K^p \in W^\rho\}$ rewrites as

$$W_K^\rho = \begin{cases} \emptyset \text{ if } \alpha_K^\rho \text{ or } \beta_K^\rho \text{ does not exist, or if } \alpha_K^\rho > \beta_K^\rho; \\ \{w_K^p : \alpha_K^\rho \leq p \leq \beta_K^\rho\} \text{ otherwise.} \end{cases} \tag{14}$$

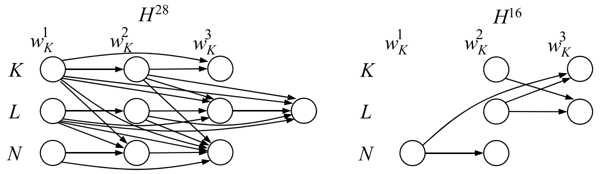

Fig. 5 Two conflict graphs

Second, K and L can be identical in the above definition of the arc set. In fact, if $K = L$, (13) is equivalent to:

$$\left(w_K^p, w_K^q\right) \in Y^\rho \quad \Leftrightarrow \quad p < q. \tag{15}$$

Indeed, by Proposition 1, $c_{JK}^p + c_{KJ}^q > 0$ if and only if $p < q$. Also, $l_{KK} = 0$ so that in (13) $c_{JK}^p + c_{LJ}^q + l_{KL} > 0$ for $K = L$ is equivalent to $p < q$. Moreover, $c_{JK}^p + c_{LJ}^q + l_{K\tau} + l_{\sigma L} \geq \rho$ for $K = L$ implies $c_{JK}^p + c_{KJ}^q \geq \rho - (l_{K\tau} + l_{\sigma K}) \geq \rho - l_{\sigma\tau} > 0$, hence $c_{JK}^p + c_{KJ}^q > 0$.

Finally, in view of (14) and (15), any non-empty W_K^ρ, $K \in \mathscr{J}^-$, is (the node set of) a *clique* in H^ρ.

Figure 5 displays two conflict graphs H^ρ for $\rho = 28$ and 16 in the example.

Conflict graphs allow to characterize feasible insertions as the following result holds.

Theorem 2 *For any $\rho > l_{\sigma\tau}$, let $\mathscr{F}^\rho$ be the family of all feasible insertions T of makespan $\omega(T) < \rho$. There is a* 1 *to* 1*-correspondence between the feasible insertions $T \in \mathcal{F}^\rho$ and the stable sets of size $|\mathscr{J}^-|$ in H^ρ.*

Proof Let $T \in \mathscr{F}^\rho$. Since T is complete and positive acyclic, given any $K \in \mathscr{J}^-$, $[J, K]_p \subseteq T$ for exactly one $p \in \{1, \ldots, q_K\}$, say p_K. We show first that $w_K^{p_K} \in W^\rho$. Indeed, if $w_K^{p_K} \notin W^\rho$, $c_{\sigma J} + c_{JK}^{p_K} + l_{K\tau} \geq \rho$ or $l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau} \geq \rho$. In both cases, there is a σ-τ-path in $(\mathcal{J}^+, B \cup U_R \cup U_T, c)$ of length $\geq \rho$, a contradiction to $\rho > \omega(T)$, $\omega(T)$ being the length of a longest σ-τ-path in $(\mathscr{J}^+, B \cup U_R \cup U_T, c)$. Hence to $T \in \mathscr{F}^\rho$ corresponds node set $T' = \{w_K^{p_K} : K \in \mathscr{J}^-\} \subseteq W^\rho$ of size $|\mathscr{J}^-|$ in H^ρ.

T' is a stable set in H^ρ. Indeed, suppose the contrary: for some $K \neq L$, $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$, i.e. $c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{KL} > 0$ or $c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{K\tau} + l_{\sigma L} \geq \rho$. In the first case, the positive cyclic insertion $\{[J, K]_{p_K}, [L, J]_{p_L}\}$ is contained in T, contradicting T being positive acyclic. In the second case, there is a σ-τ-path in $(\mathscr{J}^+, B \cup U_R \cup U_T, c)$ of length $\geq \rho$, contradicting $\rho > \omega(T)$.

Conversely, let T' be a stable set of size $|\mathcal{J}^-|$ in H^ρ. T' picks up at most one node, say $w_K^{p_K}$, from each clique W_K^ρ, $K \in \mathscr{J}^-$ and, since $\cup_{K \in \mathscr{J}^-} W_K^\rho$ is a partition of W^ρ and $|T'| = |\mathscr{J}^-|$, $T' = \{w_K^{p_K} : K \in \mathscr{J}^-\}$. The insertion $T = \bigcup_{K \in \mathscr{J}^-} [J, K]_{p_K}$ is obviously complete. T is also positive acyclic, otherwise there is a positive cycle in $(\mathscr{J}^+, B \cup U_R \cup U_T, c)$ which must go through J, entering J, through, say, arc $(L, J)_{p_L}$ and leaving J through $(J, K)_{p_K}$, implying $c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{KL} > 0$, hence $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$, a contradiction to the stability of T'. Finally, T has makespan $\omega(T) < \rho$, otherwise there is a longest σ-τ-path in $(\mathcal{J}^+, B \cup U_R \cup U_T, c)$ of length $\geq \rho$. This path must visit at least two jobs, otherwise it is in $(\mathscr{J}^+, B \cup U_R, c)$ and its length is at most $l_{\sigma\tau} < \rho$. If J is the first and K the second job on this path, its length is $c_{\sigma J} + c_{JK}^{p_K} + l_{K\tau}$. If J is the last and K the second to last job on the path, its length is $l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau}$. If J is between L and K on the path, its length is $c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{K\tau} + l_{\sigma L}$. The first two cases imply $w_K^{p_K} \notin W^\rho$, contradicting $T' \subseteq W^\rho$, the third case implies $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$, contradicting the stability of T'. □

An immediate consequence of Theorem 2 is the following characterization of all feasible insertions, regardless of makespan.

Corollary 2 *There is a* 1 *to* 1*-correspondence between the feasible insertions and the stable sets of size* $|\mathscr{J}^-|$ *in graph* $H = (W, Y)$ *with node set* $W = \{w_K^p : K \in \mathscr{J}^-,\ p = 1, \dots, q_K\}$ *and the arc set* Y *defined by*

$$\textit{for all } p = 1, \dots, q_K, q = 1, \dots, q_L, K \textit{ and } L \in \mathscr{J}^- :$$
$$\textit{if } K = L : \left(w_K^p, w_L^q\right) \in Y^\rho \quad \Leftrightarrow \quad p < q$$
$$\textit{if } K \neq L : \left(w_K^p, w_L^q\right) \in Y^\rho \quad \Leftrightarrow \quad c_{JK}^p + c_{LJ}^q + l_{KL} > 0.$$

Proof The result follows immediately from Theorem 2, choosing ρ_0 such that $\rho_0 > \omega(T)$ for any feasible insertion T. Then $H = H^{\rho_0}$. □

Theorem 2 can also be expressed as follows.

Corollary 3 *For any* $\rho > l_{\sigma\tau}$, *let* $\alpha(H^\rho)$ *denote the stability number of* H^ρ *and* $\mathscr{F}^\rho$ *the family of all feasible insertions* T *of makespan* $\omega(T) < \rho$. *Either* $\alpha(H^\rho) < |\mathscr{J}^-|$ *and* $\mathscr{F}^\rho = \emptyset$, *or* $\alpha(H^\rho) = |\mathscr{J}^-|$ *and the maximum size stable sets of* H^ρ *are in* 1*-to-*1 *correspondence with the members of* $\mathscr{F}^\rho$.

Proof Since $\bigcup_{K \in \mathscr{J}^-} W_K^\rho$ is a partition of W^ρ into $|\mathscr{J}^-|$ cliques, $\alpha(H^\rho) \leq |\mathscr{J}^-|$. Also, $|T| \leq \alpha(H^\rho)$ for any stable set T. Therefore if T is a stable set of size $|\mathscr{J}^-|$, $|T| = \alpha(H^\rho)$. □

Finding a feasible insertion T of makespan $\omega(T) < \rho$ or determining that no such insertion exists amounts therefore to finding a maximum size stable set in graph H^ρ, a difficult problem in a general graph. Fortunately, H^ρ is a so-called *comparability graph.*

Theorem 3 $H^\rho = (W^\rho, Y^\rho)$ *is acyclic and transitively oriented.*

Proof (a) We show that H^ρ is transitively oriented, i.e. for any triple of distinct nodes $w, w', w'' \in W^\rho$

$$\left(w, w'\right) \in Y^\rho \quad \text{and} \quad \left(w', w''\right) \in Y^\rho \quad \Rightarrow \quad \left(w, w''\right) \in Y^\rho.$$

Assume $(w, w') = (w_K^p, w_L^q) \in Y^\rho$ and $(w', w'') = (w_L^q, w_N^r) \in Y^\rho$. Then (16) or (17) and (18) or (19) must hold:

$$c^p_{JK} + c^q_{LJ} + l_{KL} > 0, \tag{16}$$

$$c^p_{JK} + c^q_{LJ} + l_{K\tau} + l_{\sigma L} \geq \rho, \tag{17}$$

$$c^q_{JL} + c^r_{NJ} + l_{LN} > 0, \tag{18}$$

$$c^q_{JL} + c^r_{NJ} + l_{L\tau} + l_{\sigma N} \geq \rho. \tag{19}$$

Assume (16) and (18) hold. Then adding both left and right hand sides and using $c^q_{JL} + c^q_{LJ} \leq 0$ and $l_{KL} + l_{LN} \leq l_{KN}$, yields $c^p_{JK} + c^r_{NJ} + l_{KN} > 0$. If (16) and (19) hold, then adding and using $c^q_{JL} + c^q_{LJ} \leq 0$ and $l_{KL} + l_{L\tau} \leq l_{K\tau}$, yields $c^p_{JK} + c^r_{NJ} + l_{K\tau} + l_{\sigma N} > \rho$. If (17) and (18) hold, $c^p_{JK} + c^r_{NJ} + l_{K\tau} + l_{\sigma N} > \rho$ and if (17) and (19) hold, $c^p_{JK} + c^r_{NJ} + l_{K\tau} + l_{\sigma N} \geq \rho + (\rho - (l_{\sigma L} + l_{L\tau})) \geq \rho + (\rho - l_{\sigma\tau}) > \rho$. In each case, $c^p_{JK} + c^r_{NJ} + l_{KN} > 0$ or $c^p_{JK} + c^r_{NJ} + l_{K\tau} + l_{\sigma N} \geq \rho$ holds, so that $(w^p_K, w^r_N) \in Y^\rho$.

(b) We prove that H^ρ is acyclic. Since H^ρ is transitively oriented, it suffices to show that H^ρ contains no $w, w' \in W^\rho$ with both (w, w') and $(w', w) \in Y^\rho$. Assume the contrary and let $w = w^p_K$ and $w' = w^q_L$. $(w^p_K, w^q_L) \in Y^\rho$ implies that (16) or (17) holds and $(w^q_L, w^p_K) \in Y^\rho$ implies

$$c^q_{JL} + c^p_{KJ} + l_{LK} > 0 \tag{20}$$

or

$$c^q_{JL} + c^p_{KJ} + l_{L\tau} + l_{\sigma K} \geq \rho. \tag{21}$$

Since $c^p_{JK} + c^p_{KJ} \leq 0$ and $c^q_{JL} + c^q_{LJ} \leq 0$, if (16) and (20) hold, then $l_{KL} + l_{LK} > 0$. If (16) and (21) hold, then $l_{\sigma\tau} \geq l_{\sigma K} + l_{KL} + l_{L\tau} > \rho$. If (17) and (20) hold, then $l_{\sigma\tau} \geq l_{\sigma L} + l_{LK} + l_{K\tau} > \rho$. If (17)and (21) hold, then $2l_{\sigma\tau} \geq l_{\sigma L} + l_{L\tau} + l_{\sigma K} + l_{K\tau} \geq 2\rho$. Therefore in each case (11) or $\rho > l_{\sigma\tau}$ is contradicted. □

4.3 The OJI-NWJS algorithm

Corollary 3 and Theorem 3 suggest the following *general approach* for solving the OJI-NWJS.

In an initialization step, let T be a stable set in $H = H^{\rho_0}$, for instance $T := \{w^{q_K}_K : K \in \mathscr{J}^-\}$, the insertion placing J after all other jobs. Set $\rho := \omega(T)$. Then execute the following loop. Determine a maximum size stable set T' in H^ρ. If $\alpha(H^\rho) := |T'| < |\mathscr{J}^-|$, then *stop*: T corresponds to an optimal insertion. Else reset $T := T'$ and $\rho := \omega(T)$.

Figure 6 depicts a possible run of this approach in the example. The initialization step is depicted in (i). In $H = H^{28}$, T is the stable set (grey nodes) which corresponds to the insertion placing J after all other jobs, with makespan 16 (see Gantt chart). (ii) illustrates the first pass of the loop. T is updated to a maximum size stable set

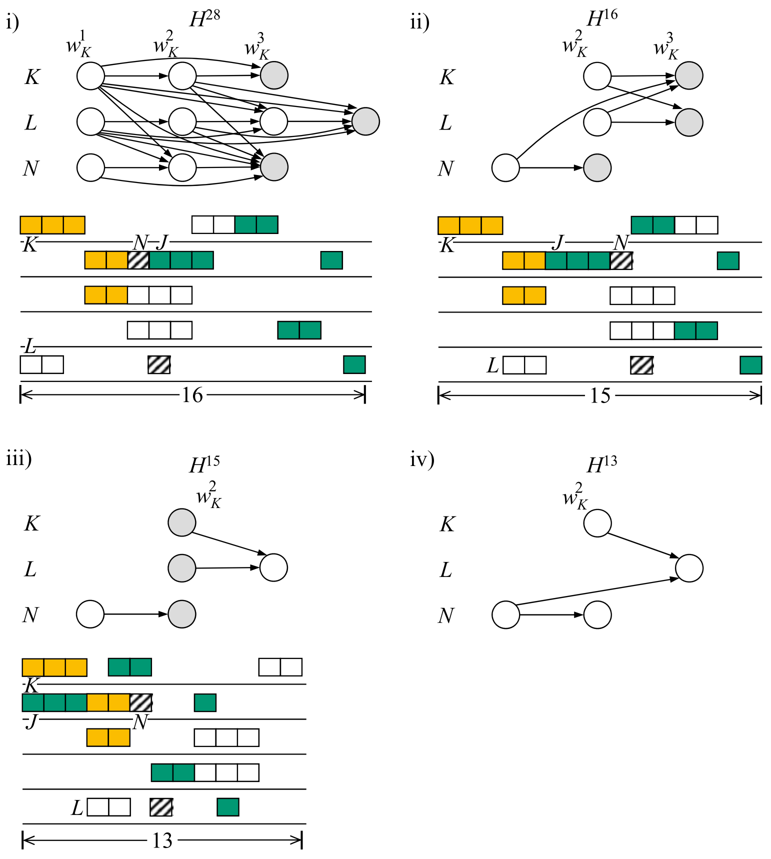

Fig. 6 A possible run of the general approach

in H^{16}. The corresponding insertion has makespan 15. The next pass is depicted in (iii) leading to an insertion with makespan 13. This is an optimal insertion since $\alpha(H^{13}) < |\mathscr{J}^-|$, see (iv).

At this point, we remark that this general approach yields a polynomial-time algorithm. Indeed, finding a maximum size stable set in a comparability graph can be found via network flow methods. Also, a (rough) upper bound on the number of iterations of the loop is $|W^{\rho_0}|^3 = (\sum_{K\in\mathscr{J}^-} q_K)^3$, observing that for graphs, say $H^{\rho_r} = (W^{\rho_r}, Y^{\rho_r})$ and $H^{\rho_{r+1}} = (W^{\rho_{r+1}}, Y^{\rho_{r+1}})$ of two consecutive iterations r and $r+1$, $W^{\rho_{r+1}} \subseteq W^{\rho_r}$, and if $W^{\rho_{r+1}} = W^{\rho_r}$, then $Y^{\rho_r} \subset Y^{\rho_{r+1}}$.

We propose however a substantially more efficient implementation which relies on the following two propositions. The first uses the fact that the maximum size stable sets of a comparability graph form a lattice (see for instance Schrijver (2003), p. 235). Assume $\alpha(H^\rho) = |\mathscr{J}^-|$.

Proposition 2 *The maximum size stable sets of H^ρ form a lattice $\mathscr{L}^\rho$ with order $\prec$, meet $\vee$ and join $\wedge$ defined as follows. For any two stable sets $T := \{w_K^{p_K} : K \in \mathcal{J}^-\}$*

and $T' := \{w_K^{p'_K} : K \in \mathscr{J}^-\}$,

$$T \preceq T' \quad \Leftrightarrow \quad p_K \leq p'_K \quad \textit{for all } K \in \mathscr{J}^-$$

$$T \vee T' = \{w_K^{\max\{p_K, p'_K\}} : K \in \mathscr{J}^-\}$$

$$T \wedge T' = \{w_K^{\min\{p_K, p'_K\}} : K \in \mathcal{J}^-\}$$

Proof Clearly, $\prec$ is a partial order. $T \vee T'$ is stable, otherwise there exist w and $w' \in T \vee T'$ with $(w, w') \in Y^\rho$ or $(w', w) \in Y^\rho$. We may assume $w \in T - T'$ and $w' \in T' - T$ so that $w = w_K^{p_K}$ for some K and $p'_K < p_K$ and $w' = w_L^{p'_L}$ for some $L \neq K$ and $p_L < p'_L$. If $(w, w') = (w_K^{p_K}, w_L^{p'_L}) \in Y^\rho$, then $(w_K^{p'_K}, w_L^{p'_L}) \in Y^\rho$ by $(w_K^{p'_K}, w_K^{p_K}) \in Y^\rho$ and transitivity, and if $(w', w) = (w_L^{p'_L}, w_K^{p_K}) \in Y^\rho$, then $(w_L^{p_L}, w_K^{p_K}) \in Y^\rho$, contradicting the stability of T' or T. Similarly, one can show that $T \wedge T'$ is stable. Finally, $|T \vee T'| = |T \wedge T'| = |T| = |T'|$. □

Assume now $\rho' > \rho > l_{\sigma\tau}$ and $\alpha(H^\rho) = \alpha(H^{\rho'}) = |\mathscr{J}^-|$, and let $\mathscr{L}^\rho$ and $\mathscr{L}^{\rho'}$ be the respective lattices of the maximum size stable sets in H^ρ and $H^{\rho'}$.

Proposition 3 *$\mathscr{L}^\rho$ is a sublattice of $\mathscr{L}^{\rho'}$.*

Proof By definition of $H^\rho = (W^\rho, Y^\rho)$, $H^{\rho'} = (W^{\rho'}, Y^{\rho'})$ and $\rho < \rho'$, $W^\rho \subseteq W^{\rho'}$, w, $w' \in W^\rho$ and $(w, w') \in Y^{\rho'}$ implies $(w, w') \in Y^\rho$. Therefore any member $T \in \mathscr{L}^\rho$ is a member of $\mathscr{L}^{\rho'}$. □

A version of the general approach described previously for solving the OJI-NWJS is now the following algorithm.

OJI-NWJS algorithm In an initialization step, set $T := \{w_K^{q_K} : K \in \mathscr{J}^-\}$ and $\rho := \omega(T)$. T is the maximal member of $\mathscr{L}^{\rho_0}$ and corresponds to the insertion placing J after all other jobs. Then repeat the following loop: While $\mathscr{L}^\rho \neq \emptyset$, determine the maximal member T' of $\mathscr{L}^\rho$ and set $T := T'$ and $\rho := \omega(T')$. At completion, if $T = \{w_K^{p_K} : K \in \mathscr{J}^-\}$, the insertion $\bigcup_{K \in \mathscr{J}^-} [J, K]_{p_K}$ is optimal.

The validity of the algorithm can be asserted as follows. Let $T^{(0)}$, $T^{(1)}, \ldots,$ be the sequence of sets T generated by the algorithm. By Propositions 2 and 3, $T^{(s)} \preceq T^{(s-1)}$, $s = 1, 2, \ldots,$ and since $T^{(s)} \in \mathscr{L}^{\omega(T^{(s-1)})}$, $T^{(s)} \prec T^{(s-1)}$, so that the number of generated sets is bounded by $\sum_{K \in \mathscr{J}^-} q_K$.

It remains to be shown how an iteration of the while loop is performed, i.e. how to solve the problem:

$$\begin{aligned} &\text{Given } T \text{ maximal in } \mathscr{L}^{\rho'}, \text{ let } \rho := \omega(T). \\ &\text{Determine that } \mathscr{L}^\rho = \emptyset \text{ or find } T' \text{ maximal in } \mathscr{L}^\rho. \end{aligned} \tag{22}$$

Let a node set $T \subseteq W^\rho$ be called *complete* if $|T \cap W_K^\rho| = 1$ for all $K \in \mathscr{J}^-$. Note that $T \in \mathscr{L}^\rho$ if and only if T is complete and stable. (22) is solved by a procedure which successively generates complete sets $T^1, \dots, T^R$ in W^ρ, such that for $r = 1, 2, \dots, R$,

$$T^r \prec T^{r-1} \text{ and} \tag{23}$$

$$T \preceq T^r \text{ for all } T \in \mathscr{L}^\rho \tag{24}$$

and R is the smallest r such that T^r is stable in H^ρ or $\mathscr{L}^\rho = \emptyset$ can be asserted.

At completion of the procedure, either $\mathscr{L}^\rho = \emptyset$ or T^R is the sought set T', since T^R is stable and complete, hence $T^R \in \mathscr{L}^\rho$, and T^R is maximal in $\mathscr{L}^\rho$ by (24).

Postponing temporarily the determination of the initial complete set T^0, we show how, given a non-stable set T^{r-1} such that $T \preceq T^{r-1}$ for all $T \in \mathscr{L}^\rho$, we either assert $\mathscr{L}^\rho = \emptyset$ or construct T^r fulfilling (23) and (24). Assume $T^{r-1} = \{w_K^{p_K} : K \in \mathscr{J}^-\} \subseteq W^\rho$. Since T^{r-1} is not stable, there exists $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$ for some K and L. Let $\Gamma(w_K^{p_K}) = \{w_L^p : (w_K^{p_K}, w_L^p) \in Y^\rho\}$ be the set of all successors of $w_K^{p_K}$.

Proposition 4 *$T \cap \Gamma(w_K^{p_K}) = \emptyset$ for all $T \in \mathscr{L}^\rho$*

Proof Suppose there is $T' \in \mathscr{L}^\rho$ with $T' \cap \Gamma(w_K^{p_K}) \neq \emptyset$, i.e. letting $T' = \{w_L^{p'_L} : L \in \mathscr{J}^-\}$, $(w_K^{p_K}, w_L^{p'_L}) \in Y^\rho$ for some $L \in \mathscr{J}^-$. Since $T \preceq T^{r-1}$ for all $T \in \mathscr{L}^\rho$, $T' \preceq T^{r-1}$ and therefore $p'_K \leq p_K$. Hence $p'_K = p_K$ or $(w_K^{p'_K}, w_K^{p_K}) \in Y^\rho$. In both cases, using transitivity in the second, $(w_K^{p'_K}, w_L^{p'_L}) \in Y^\rho$, contradicting the stability of T'. □

Hence given a non-stable $T^{r-1} = \{w_K^{p_K} : K \in \mathscr{J}^-\}$, find a node $w_K^{p_K}$ having a successor in T^{r-1}, and delete all successors of $w_K^{p_K}$. This deletion does not affect any $T \in \mathscr{L}^\rho$ by Proposition 4. If after deletion, some W_K^ρ is empty, then obviously $\mathscr{L}^\rho = \emptyset$; otherwise determine for all $K \in \mathscr{J}^-$ the node $w_K^p \in W_K^\rho$ with largest p, say $w_K^{p'_K}$. Then $T^r := \{w_K^{p'_K} : K \in \mathscr{J}^-\}$ satisfies (23) and (24) by Proposition 4.

The initial set $T^0 \subseteq W^\rho$ can be determined as follows. Assuming $T = \{w_K^{p_K} : K \in \mathscr{J}^-\}$ in (22) and $\rho := \omega(T)$, determine for all $K \in \mathscr{J}^-$ the largest $p \leq p_K$, say p'_K, such that $w_K^p \in W_K^\rho$. If for some K, p'_K does not exists, then $\mathscr{L}^\rho = \emptyset$, otherwise $T^0 := \{w_K^{p'_K} : K \in \mathcal{J}^-\}$. There is however a better choice for T^0 (see Appendix).

In order to determine its computational complexity, the OJI-NWJS algorithm needs to be specified in more detail. This is done in the Appendix where a pseudo-code implementation together with a proof of the following complexity result is provided.

Theorem 4 *The OJI-NWJS algorithm runs in time $\mathscr{O}(n \cdot \max\{n^2, \sum_K q_K\})$.*

Herein n denotes the number of jobs. We remark that in the "classical" case where a job has at most one operation on a given machine, this complexity can be related to the number m of machines. Then $\sum_{K \in \mathscr{J}^-} q_K \leq (n-1)(m+1)$, and the OJI-NWJS algorithm runs in time $\mathscr{O}(n^2 \cdot \max\{n, m\})$.

Table 1 Performance analysis of the OJI-NWJS algorithm

Problem size	Number of job insertions	Millisec. per job insertion		Performance ratio
		GK-Ins. Alg.	OJI-NWJS Alg.	
20×5	31130	3.212	0.359	9
15×10	14947	6.690	0.094	71
20×10	7392	13.528	0.175	77
30×10	2677	37.355	0.544	69
50×10	797	125.471	2.304	54
15×15	5839	17.126	0.081	211
20×15	2898	34.507	0.162	213
20×20	1565	63.898	0.171	374

5 An optimal job insertion-based heuristic for the NWJS problem

Prior to this work, we devised a job insertion algorithm for the NWJS that is a straightforward implementation of our general insertion algorithm described in Gröflin and Klinkert (2007). This implementation, which we refer to as the GK-insertion algorithm, yielded optimal insertions, however its computation time was relatively high and limited its use, for instance within a heuristic for the NWJS problem.

It is of interest to compare the computation times of the OJI-NWJS algorithm developed in the previous section and the GK-insertion algorithm. For this purpose, we measured the average computation time per job insertion in NWJS problems of size $n \times m$, where the numbers n of jobs and m of machines range from 20 to 50 and 5 to 20. The sample size (number of job insertions) ranges from 31130 for size 20×5 to 797 for size 50×10.

The average time per job insertion for the GK-insertion algorithm and the OJI-NWJS algorithm, as well as the ratio of the first to the second, are recorded in Table 1, columns 3, 4 and 5. The OJI-NWJS algorithm is many times faster than the GK-insertion algorithm as the ratios show.

The high performance of the OJI-NWJS algorithm motivated the development of a heuristic for the NWJS problem that is based on optimal job insertion. This method can be described as follows.

A feasible initial selection S is built up by successively inserting optimally a job. Then, a simple descent local search is used to improve the initial selection. A neighbor is generated by extracting and reinserting optimally a job. For each job on a longest path from σ to τ in graph $F(S)$, one such neighbor is generated. If a neighbor yields a lower makespan, the current selection is reset to this neighbor. The algorithm stops if a local optimum is reached. The neighborhood just described will be referred to as N^1. This neighborhood is rather small (of size $|\mathscr{J}|$ at most) and tests showed that local search using N^1 reaches a local optimal solution relatively fast. For this reason, we also examined a larger neighborhood N^2. A neighbor in N^2 is generated by extracting two jobs and reinserting them successively. One such neighbor is generated for each ordered pair of jobs. Our second local search algorithm uses neighborhood N^1 until a local optimal selection is found and continues then with neighborhood N^2. If a neighbor in N^2 yields a lower makespan, the current selection is reset to this neighbor and local search continues again with neighborhood N^1. The algorithm stops if no better neighbor is found in N^2. Local search is repeated from various initial selections generated by random job insertion orders until a given computation time limit is reached.

The two described algorithms, based on optimal job insertion (using job insertion algorithm OJI-NWJS) and local search with neighborhoods N^1 and $N^1 \cup N^2$ respectively, will be referred to as OJILS1 and OJILS2.

6 Computational results

OJILS1 and OJILS2 have been implemented in Java and run on a PC with 2.83 GHz processor and 2 GB memory. Extensive tests have been performed on the following well-known benchmark instances for job shop scheduling problems: abz7-9 proposed by Adams et al. (1988), la11-15/21-40 by Lawrence (1984), swv01-20 by Storer et al. (1992) and yn1-4 by Yamada and Nakano (1992). These instances are interpreted as "classical" no-wait job shop instances without setup times, i.e. $\gamma_{ij} = p_i$ for a pair i, j of consecutive operations of a job. For each instance, five independent runs were performed.

We first compared the algorithms OJILS1 and OJILS2 with each other. Table 2 shows numerical results for a time limit of 1800 seconds per run. The first block (column 2-3) displays the average results (over five runs) and the second block (column 4-5) the best results (over five runs) achieved by OJILS1 and OJILS2. The instances are grouped according to size, e.g. the group 20×5 contains instances with 20 jobs and 5 machines. The best values (between OJILS1 and OJILS2) are in boldface and values known to be optimal are annotated by an asterisk.

Compared with OJILS1, the average results (*avg*) of OJILS2 are better, equal and worse in 35, 16 and 1 instances, respectively. Over all instances, OJILS2 is 1.0% better. The best results (*best*) of OJILS2 are better, equal and worse in 20, 27 and 5 instances, respectively. Altogether, these numbers suggest that OJILS2 should be given preference over OJILS1.

We then compared the results of OJILS2 with the best current benchmarks for the NWJS to our knowledge, namely the results of Bozejko and Makuchowski (2009), van den Broek (2009) and Zhu et al. (2009). Before presenting this comparison, we briefly sketch the approaches taken by these authors.

Bożejko and Makuchowski (*BM*) apply a hybrid tabu search algorithm. A feasible schedule is built by successively inserting each job as early as possible without changing the starting times of already scheduled jobs. This procedure is repeated with various job insertion orders generated within a tabu search.

Zhu et al. (*Zhu*) propose a complete local search with limited memory based on a so-called shift timetabling procedure. This procedure builds a feasible schedule again by successively inserting each job without changing the starting times of already scheduled jobs. A job is either placed as early as possible or after all other jobs.

Van den Broek (*vdB*) proposes a heuristic which successively inserts optimally a job. He formulates each job insertion as a mixed integer linear programming problem

Table 2 Numerical results of OJILS1 and OJILS2 (bold: best, *: optimal)

	avg		*best*	
	OJILS1	OJILS2	OJILS1	OJILS2
(20 × 5)				
la11	1620	**1619***	**1619***	**1619***
la12	1429	**1422**	**1414***	**1414***
la13	1590	**1584**	1587	**1580***
la14	**1588**	1591	**1578***	**1578***
la15	1673	**1671***	**1671***	**1671***
(15 × 10)				
la21	**2030***	**2030***	**2030***	**2030***
la22	**1852***	**1852***	**1852***	**1852***
la23	**2021***	**2021***	**2021***	**2021***
la24	**1972***	**1972***	**1972***	**1972***
la25	**1906***	**1906***	**1906***	**1906***
(20 × 10)				
la26	2531	**2485**	2506	**2477**
la27	2673	**2656**	**2649**	**2649**
la28	2571	**2552**	2554	**2546***
la29	2362	**2335**	**2300***	**2300***
la30	2527	**2472**	**2452***	**2452***
swv01	**2318***	**2318***	**2318***	**2318***
swv02	**2417***	**2417***	**2417***	**2417***
swv03	**2381***	**2381***	**2381***	**2381***
swv04	**2462***	**2462***	**2462***	**2462***
swv05	**2333***	**2333***	**2333***	**2333***
(30 × 10)				
la31	3636	**3583**	3604	**3559**
la32	3982	**3901**	3964	**3863**
la33	3605	**3543**	3580	**3510**
la34	3695	**3602**	3670	**3583**
la35	3733	**3627**	3716	**3591**
(50 × 10)				
swv11	5694	**5514**	5679	**5456**
swv12	5658	**5537**	5601	**5481**
swv13	5769	**5628**	5701	**5571**
swv14	5519	**5403**	5404	**5382**
swv15	5493	**5382**	5420	**5351**
swv16	6108	**5936**	6026	**5857**
swv17	5881	**5729**	5839	**5689**
swv18	5950	**5758**	5922	**5668**
swv19	6178	**5965**	6064	**5885**
swv20	5859	**5752**	5818	**5624**
(15 × 15)				
la36	**2685***	**2685***	**2685***	**2685***
la37	**2831***	**2831***	**2831***	**2831***
la38	**2525***	**2525***	**2525***	**2525***
la39	**2660***	**2660***	**2660***	**2660***
la40	**2564***	**2564***	**2564***	**2564***
(20 × 15)				
abz7	1573	**1572**	**1528**	1555
abz8	1643	**1639**	**1569**	1627
abz9	1593	**1561**	1572	**1549**
swv06	3280	**3278***	**3278***	**3278***
swv07	3202	**3188**	**3188**	**3188**
swv08	**3423**	**3423**	**3423**	**3423**
swv09	3251	**3246**	**3246**	**3246**
swv10	3458	**3455**	**3451**	**3451**
(20 × 20)				
yn1	2401	**2392**	2378	**2366**
yn2	2387	**2320**	**2293**	2295
yn3	2338	**2319**	**2288**	2294
yn4	2485	**2463**	**2424**	2430

and solves it with CPLEX. He also develops an exact approach based on branch & bound, using his heuristic solution as an initial upper bound. The branch & bound method solves smaller instances within minutes and is also used for larger instances as a heuristic providing good solutions within a given run-time.

Table 3 displays for each instance average and best results over five runs (denoted below *avg* and *best*) achieved by OJILS2, together with benchmarks of *BM*, *Zhu* and *vdB*. We tried to use the same computation time as reported by these authors. For this reason, Table 3 has been divided into three blocks. The first block (column 2-4)

Table 3 Comparison of OJILS2 results *avg1, avg2, avg3* and *best* with benchmarks *BM, vdB, Zhu, BM2* and *Zhu2* (bold: best, *: optimal)

instance	*avg1*	*BM*	*T1*	*avg2*	*vdB*	*Zhu*	*T2*	*avg3*	*best*	*BM2*	*Zhu2*	*opt*
(20 × 5)												
la11	**1646**	1704	11	**1625**	1654	1716	447	**1619***	**1619***	1621	1671	1619
la12	**1480**	1500	10	**1441**	1451	1506	498	1421	**1414***	1434	1452	1414
la13	**1614**	1696	17	**1592**	1595	1661	640	1582	**1580***	**1580***	1624	1580
la14	**1670**	1722	10	1617	**1578***	1721	465	**1578***	**1578***	1610	1691	1578
la15	**1717**	1747	7	**1680**	1686	1749	484	**1671***	**1671***	1686	1694	1671
(15 × 10)												
la21	**2043**	2191	5	**2030***	**2030***	2104	306	**2030***	**2030***	**2030***	2048	2030
la22	**1891**	1922	3	**1852***	**1852***	1912	354	**1852***	**1852***	**1852***	1887	1852
la23	**2071**	2126	5	**2021***	**2021***	2098	307	**2021***	**2021***	**2021***	2032	2021
la24	**2006**	2132	5	**1972***	**1972***	2048	422	**1972***	**1972***	**1972***	2015	1972
la25	**1911**	2020	6	**1906***	**1906***	1971	297	**1906***	**1906***	**1906***	1917	1906
(20 × 10)												
la26	**2580**	2738	14	**2495**	2598	2707	812	2478	**2477**	2506	2553	2467
la27	**2728**	2794	27	**2666**	2755	2838	834	2638	**2611***	2675	2747	2611
la28	**2662**	2741	24	**2557**	2722	2752	810	2551	**2546***	2552	2624	2546
la29	**2425**	2596	12	**2340**	2427	2539	778	2320	**2300***	**2300***	2489	2300
la30	**2642**	2791	12	**2507**	2572	2743	822	**2452***	**2452***	**2452***	2665	2452
swv01	**2338**	2424	11	**2318***	2344	2389	641	**2318***	**2318***	**2318***	2328	2318
swv02	**2433**	2484	16	**2417***	2430	2493	777	**2417***	**2417***	**2417***	2418	2417
swv03	2411	**2404**	17	**2381***	2517	2483	757	**2381***	**2381***	**2381***	2415	2381
swv04	**2509**	2545	23	**2462***	2635	2562	675	**2462***	**2462***	**2462***	2542	2462
swv05	**2365**	2489	22	**2333***	2555	2495	712	**2333***	**2333***	**2333***	**2333***	2333
(30 × 10)												
la31	**3627**	3869	151	**3578**	3708	3884	2588	3578	3556	**3498**	3745	–
la32	**3990**	4045	176	**3873**	4337	4259	2698	3849	**3776**	3882	4028	–
la33	**3644**	3751	120	**3524**	3976	3842	2587	3521	3501	**3454**	3749	–
la34	**3695**	3936	102	**3601**	4161	3932	2754	3589	**3568**	3659	3824	–
la35	**3750**	3918	120	**3627**	3945	3984	2615	3627	3591	**3552**	3760	–
(50 × 10)												
swv11	**5514**	5634	1736	–	–	–	–	5512	**5456**	5564	–	–
swv12	5519	**5465**	2212	–	–	–	–	5518	5481	**5441**	–	–
swv13	**5621**	5807	2360	–	–	–	–	5606	**5571**	5628	–	–
swv14	**5417**	5458	1602	–	–	–	–	5388	**5369**	5401	–	–
swv15	**5372**	5619	2076	–	–	–	–	5366	**5334**	5435	–	–
swv16	**5943**	6233	1348	–	–	–	–	5886	**5779**	5843	–	–
swv17	**5729**	5900	1760	–	–	–	–	5689	**5646**	5780	–	–

Table 3 (*Continued*)

instance	*avg1*	*BM*	*T1*	*avg2*	*vdB*	*Zhu*	*T2*	*avg3*	*best*	*BM2*	*Zhu2*	*opt*
swv18	**5785**	5931	1430	–	–	–	–	5731	**5668**	5785	–	–
swv19	**5967**	6283	1481	–	–	–	–	5956	**5885**	5997	–	–
swv20	**5752**	5945	1843	–	–	–	–	5721	**5620**	5724	–	–
(15 × 15)												
la36	**2757**	2893	9	**2685***	2692	2778	535	**2685***	**2685***	**2685***	**2685***	2685
la37	**2931**	3107	7	**2831***	2977	3019	505	**2831***	**2831***	**2831***	2962	2831
la38	**2593**	2706	6	**2525***	2571	2676	497	**2525***	**2525***	**2525***	2617	2525
la39	**2687**	2725	9	**2660***	2706	2776	558	**2660***	**2660***	2687	2697	2660
la40	**2573**	2804	12	**2564***	2709	2709	283	**2564***	**2564***	2580	2594	2564
(20 × 15)												
abz7	**1643**	1775	20	–	–	–	–	1559	**1528**	1592	–	–
abz8	**1686**	1727	51	–	–	–	–	1628	**1606**	1642	–	–
abz9	**1619**	1705	52	–	–	–	–	1561	**1549**	1562	–	–
swv06	**3312**	3463	29	**3278***	3449	3457	1136	**3278***	**3278***	3290	3376	3278
swv07	**3225**	3299	32	**3188**	3357	3321	1176	**3188**	**3188**	**3188**	3271	–
swv08	**3428**	3567	29	**3423**	3949	3634	1149	**3423**	**3423**	**3423**	3530	–
swv09	**3293**	3439	39	**3246**	3355	3362	1053	**3246**	**3246**	3270	3307	–
swv10	**3511**	3561	23	**3455**	3790	3564	1142	**3451**	**3451**	3462	3488	–
(20 × 20)												
yn1	**2461**	2630	68	–	–	–	–	2374	2366	**2360**	–	–
yn2	**2457**	2647	41	–	–	–	–	2317	**2295**	2370	–	–
yn3	**2397**	2465	134	–	–	–	–	2299	**2288**	2320	–	–
yn4	**2542**	2630	53	–	–	–	–	2443	**2402**	2513	–	–

reports OJILS2 results (*avg1*) and results of *BM* with *their* run-time *T1*. The second block of four columns compares OJILS2 results (*avg2*), results of *vdB* and average results of *Zhu* over 20 runs with their run-time *T2*. The third block of four columns shows results with high run-times. Column *avg3* and *best* report OJILS2 average and best results after a computation time of 3600 seconds. Column *BM2* shows results of Bożejko and Makuchowski with so-called "unlimited" run-time and column *Zhu2* contains the best results over 20 runs of Zhu et al. The last column *opt* displays optimal values for some instances taken from (van den Broek 2009). The best values of each block are put in boldface and optimal values are annotated by an asterisk.

Results *avg1* are in 50 (out of 52) instances better than *BM* and, on average over all instances, they are 3.7% better. Results *avg2* systematically dominate *Zhu* and are better than *vdB* in 29 out of 35 instances. *vdB* reports a better value only for one instance (*la14)*. On average, *avg2* is 4.0% and 5.4% better than *vdB* and *Zhu*, respectively. Results *avg3* are slightly better than *BM2*. Indeed, out of 52 instances, *avg3* is better, equal and worse in respectively 28, 16 and 8 instances. On average, *avg3* is 0.3% better than *BM2*. Finally, observing the values *best*, in 25 out of 26 instances with known optimum, OJILS2 reached the optimum, and in the remaining

26 instances with unknown optimum, it improved the best benchmark 19 times and matched it two times.

Altogether, the algorithm OJILS2 appears competitive when compared to the best current approaches.

7 Concluding remarks

We provided a new compact formulation of the no-wait job shop problem (NWJS) and formulated in that framework the optimal job insertion problem (OJI-NWJS). We characterized all feasible insertions of makespan less than ρ (for any given bound ρ) as all stable sets of prescribed cardinality in an associated comparability graph and presented a strongly polynomial and efficient algorithm for the OJI-NWJS.

The new compact formulation of the NWJS is certainly in part accountable for the high efficiency of the proposed OJI-NWJS algorithm. We believe it also to be valuable in future solution approaches for the NWJS. Preliminary tests on (smaller) NWJS instances formulated as ILP's (based on the compact formulation) and computed with commercial solvers support this belief.

The high efficiency of the OJI-NWJS algorithm allowed it to be implemented in a simple local search scheme, as described in Sect. 5. The achieved improvements in many benchmarks provide support for this approach, and suggest that similar approaches in other job shop problems using optimal or "near-optimal" job insertion might be worth studying.

Acknowledgements We thank the anonymous referees for their constructive remarks which led to several improvements in the exposition of the paper.

Appendix

We provide an implementation of the OJI-NWJS algorithm, describe some algorithmic details necessary to achieve running time $\mathcal{O}(n \cdot \max\{n^2, \sum_K q_K\})$ and prove this complexity.

Before proceeding, we observe that, considering the sequence of *all* complete sets generated in the OJI-NWJS algorithm, for any set T in the sequence, the next set T' is such that $T' \preceq T$. Therefore finding T' in W^ρ given $T = \{w_K^{p_K} : K \in \mathscr{J}^-\}$ can be restricted to $W^\rho|T$, the node set W^ρ where for each $K \in \mathscr{J}^-$, all nodes $w_K^p \in W_K^\rho$ with $p > p_K$ are deleted from W_K^ρ, obtaining $W_K^\rho|T$. We now sketch our implementation. A pseudo-code with line numbers is provided in Listing 1, to which we will refer in the text.

In an initialization step (lines 1 to 3), the all-pairs longest paths in graph $(\mathscr{J}^+, B \cup U_R, c)$ are computed, e.g. with the algorithm of Floyd and Warshall, and T is set as $T := \{w_K^{q_K} : K \in \mathcal{J}^-\}$, the insertion placing J after all other jobs. Then, as long as optimality of T is not established, the while loop (lines 6 to 38) is executed.

A generic loop iteration starts with $T := \{w_K^{p_K} : K \in \mathscr{J}^-\}$ maximal in $\mathscr{L}^\rho$, calculates its makespan $\omega(T)$, resets ρ to $\omega(T)$, and determines an initial complete set $T' \subseteq W^\rho$ with $T' \prec T$. T' is then successively updated on the basis of Proposition 4

```
// Initialization
Compute $l_{KL}$ for all $K, L \in \mathcal{J}^+$; optimal := **false**;
**for_all** $K \in \mathcal{J}^-$ **do** $p_K := q_K$; **end_for**

**while** optimal = **false** **do**
  // Calculate makespan $\rho$ of $T := \{w_K^{p_K} : K \in \mathcal{J}^-\}$.
  $l^1 := \max\{l_{\sigma K} + c_{KJ}^{p_K} : K \in \mathcal{J}^-\}$;
  $\mathcal{J}^1 := \{K \in \mathcal{J}^- : l_{\sigma K} + c_{KJ}^{p_K} = l^1\}$;
  $l^2 = \max\{c_{JK}^{p_K} + l_{K\tau} : K \in \mathcal{J}^-\}$;
  $\rho := \max\{l_{\sigma\tau}; c_{\sigma J} + c_{J\tau}; l^1 + c_{J\tau}; c_{\sigma J} + l^2; l^1 + l^2\}$;

  // Initialize set $T'$ and set $\mathcal{S}$ of nodes to be scanned.
  **if** $\rho = \max\{l_{\sigma\tau}; c_{\sigma J} + c_{J\tau}; c_{\sigma J} + l^2\}$ **then**
    optimal := **true**; **return**;
  **end_if**
  **for_all** $K \in \mathcal{J}^-$ **do** $p'_K := p_K$; **end_for**
  $T' := \{w_K^{p'_K} : K \in \mathcal{J}^-\}$;
  $\mathcal{S} := \emptyset$;
  **for_all** $K \in \mathcal{J}^1$ **do**
    **if** $p'_K = 1$ **or** $c_{\sigma J} + c_{JK}^{p'_K - 1} + l_{K\tau} \geq \rho$ **then**
      optimal := **true**; **return**;
    **else** $p'_K := p'_K - 1$; $\mathcal{S} := \mathcal{S} \cup w_K^{p'_K}$;
    **end_if**
  **end_for**

  // Scanning Phase
  **while** $\mathcal{S} \cap T' \neq \emptyset$, get, say, $w_K^{p'_K} \in \mathcal{S} \cap T'$ and scan it:
    **for_all** $L \in \mathcal{J}^- - K$ **do**
      $p := p'_L$;
      **while** $(w_K^{p'_K}, w_L^p) \in Y^\rho$ **do**
        $p := p - 1$;
        **if** $w_L^p \notin W^\rho$ **then** optimal := **true**; **return**; **end_if**
      **end_while**
      **if** $p < p'_L$ **then** $p'_L := p$; $\mathcal{S} := \mathcal{S} \cup w_L^{p'_L}$; **end_if**
    **end_for**
    $\mathcal{S} := \mathcal{S} - w_K^{p'_K}$;
  **end_while**
  **for_all** $K \in \mathcal{J}^-$ **do** $p_K := p'_K$; **end_for**
**end_while**
// $T := \{w_K^{p_K} : K \in \mathcal{J}^-\}$ with makespan $\rho$
// corresponds to an optimal insertion.
```

Listing 1 OJI-NWJS algorithm

until T' is maximal in $\mathscr{L}^\rho$ or $\mathscr{L}^\rho = \emptyset$. This is achieved by repeatedly applying a *scanning* operation to an unscanned node w of T', which in effect deletes all successor nodes of w from $W^\rho|T'$. If in the process for some K, $W_K^\rho|T'$ becomes empty, $\mathscr{L}^\rho = \emptyset$, otherwise T' is updated. Eventually, either $\mathscr{L}^\rho = \emptyset$ or T' is such that its nodes have been scanned without changing T', hence T' is stable and therefore maximal in $\mathscr{L}^\rho$.

A straightforward implementation of the scanning operation does not yield however the claimed computational complexity, as a node might be scanned more than once. Fortunately, with the calculation of the makespan $\omega(T)$, either $\mathscr{L}^\rho = \emptyset$ is detected or an initial set $T' \prec T$ can be determined whose only nodes to be scanned are in $T' - T$ (lines 6 to 24).

Proposition 5 *Let* $T := \{w_K^{p_K} : K \in \mathscr{J}^-\}$, $l^1 := \max\{l_{\sigma K} + c_{KJ}^{p_K} : K \in \mathscr{J}^-\}$ *and* $l^2 := \max\{c_{JK}^{p_K} + l_{K\tau} : K \in \mathscr{J}^-\}$.

(i) $\omega(T) = \max\{l_{\sigma\tau};\ c_{\sigma J} + c_{J\tau};\ c_{\sigma J} + l^2;\ l^1 + c_{J\tau}; l^1 + l^2\}$
(ii) *If* $\rho := \omega(T) = \max\{l_{\sigma\tau};\ c_{\sigma J} + c_{J\tau};\ c_{\sigma J} + l^2\}$, *then* $\mathscr{L}^\rho = \emptyset$.

Proof (i) A longest $\sigma\tau$-path either avoids J, or goes through J, visiting only one job (J), or more than one job, with J being first or last, or between two jobs on the path.

(ii) If $\omega(T) = l_{\sigma\tau}$ or $c_{\sigma J} + c_{J\tau}$, T is optimal since both $l_{\sigma\tau}$ and $c_{\sigma J} + c_{J\tau}$ are lower bounds on $\omega(T)$. If $\omega(T) = c_{\sigma J} + l^2 = c_{\sigma J} + c_{JK}^{p_K} + l_{K\tau}$ for some $K \in \mathscr{J}^-$, by Proposition 1 and (12), $w_K^p \notin W^\rho$ for all $p \in \{1, \dots, p_K\}$, hence $\mathscr{L}^\rho = \emptyset$. □

Assume now $\rho := \omega(T) > \max\{l_{\sigma\tau}; c_{\sigma J} + c_{J\tau}; c_{\sigma J} + l^2\}$, therefore $\rho = \max\{l^1 + c_{J\tau}; l^1 + l^2\}$, and let

$$\mathscr{J}^1 := \left\{K \in \mathscr{J}^- : l_{\sigma K} + c_{KJ}^{p_K} = l^1\right\}, T_1 := \left\{w_K^{p_K} : K \in \mathscr{J}^1\right\} \subseteq T$$
$$\mathscr{J}^2 := \left\{K \in \mathscr{J}^- : c_{JK}^{p_K} + l_{K\tau} = l^2\right\}, T_2 := \left\{w_K^{p_K} : K \in \mathscr{J}^2\right\} \subseteq T.$$

Lemma 1 *Let* $T := \{w_K^{p_K} : K \in \mathscr{J}^-\}$. *Either* (i) *or* (ii) *holds*:

(i) $T \nsubseteq W^\rho$: *then* $T - W^\rho = T_1$ *and for all* $v \in T \cap W^\rho = T - T_1$, v *has no successor in* $H^\rho|T$.
(ii) $T \subseteq W^\rho$: *then for any* $v \in T$, $(v, w) \in Y^\rho|T \Leftrightarrow v \in T_2$ *and* $w \in T_1$.

Proof (a) We show first that

$$\text{for any } w_K^{p_K} \in T \cap W^\rho, \text{there is no } (w_K^{p_K}, w_L^p) \in Y^\rho \text{ with } p < p_L. \tag{25}$$

Assume the contrary. By (15), $K \neq L$ and by (13)

$$\left(w_K^{p_K}, w_L^p\right) \in Y^\rho \quad \Leftrightarrow \quad \begin{cases} c_{JK}^{p_K} + c_{LJ}^p + l_{KL} > 0 \text{ or} \\ c_{JK}^{p_K} + c_{LJ}^p + l_{K\tau} + l_{\sigma L} \geq \rho. \end{cases} \tag{26}$$

By Proposition 1, $c_{LJ}^p < c_{LJ}^{p_L}$. If the first inequality in (26) holds, $c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{KL} > 0$, contradicting the stability of T in H. Hence $c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{K\tau} + l_{\sigma L} > \rho$. But then $l^1 + l^2 \geq c_{JK}^{p_K} + c_{LJ}^{p_L} + l_{K\tau} + l_{\sigma L} > \rho$, a contradiction to $\rho = \max\left\{l^1 + c_{J\tau}; l^1 + l^2\right\}$.

(b) Assume now $T \nsubseteq W^\rho$. We show that $\rho = l^1 + c_{J\tau}$, $T - W^\rho = T_1$ and $T \cap W^\rho$ is stable in H^ρ, thus, in view of (25), proving i) of the lemma statement.

Let $w_K^{p_K} \in T - W^\rho$. Since $w_K^{p_K} \notin W^\rho$, by (12)

$$c_{\sigma J} + c_{JK}^{p_K} + l_{K\tau} \geq \rho \quad \text{or} \quad l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau} \geq \rho.$$

Since we assume $\rho > c_{\sigma J} + l^2$, the second inequality must hold, hence

$$l^1 + c_{J\tau} \geq l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau} \geq \rho = \max\{l^1 + c_{J\tau}, l^1 + l^2\}, \tag{27}$$

and therefore equality must hold in the inequalities of (27), so that $\rho = l^1 + c_{J\tau}$ and $l^1 + c_{J\tau} = l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau}$, i.e. $K \in \mathscr{J}^1$ and $w_K^{p_K} \in T_1$. Hence $T - W^\rho \subseteq T_1$. Also $T - W^\rho \supseteq T_1$ holds. Indeed, suppose $w_K^{p_K} \in T_1 \cap W^\rho$. Since $w_K^{p_K} \in W^\rho$, by (12)

$$c_{\sigma J} + c_{JK}^{p_K} + l_{K\tau} < \rho \quad \text{and} \quad l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau} < \rho.$$

Since $w_K^{p_K} \in T_1$, $K \in \mathscr{J}^1$, hence the second inequality yields $l^1 + c_{J\tau} = l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau} < \rho$, contradicting $\rho = l^1 + c_{J\tau}$.

Finally, we show that $T \cap W^\rho$ is stable in H^ρ. Let $w_K^{p_K}$ and $w_L^{p_L} \in T \cap W^\rho$, $K \neq L$, and suppose $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$. By (13), and since T is stable in H, $l_{\sigma L} + c_{LJ}^{p_L} + c_{JK}^{p_K} + l_{K\tau} \geq \rho$. Since $w_L^{p_L} \notin T_1$, $l_{\sigma L} + c_{LJ}^{p_L} < l^1$, and $c_{JK}^{p_K} + l_{K\tau} \leq l^2$ so that $l^1 + l^2 > \rho$, a contradiction to $\rho = \max\{\, l^1 + c_{J\tau}; l^1 + l^2\}$.

(c) Assume $T \subseteq W^\rho$. We show that $\rho = l^1 + l^2$ and for any $v, w \in T$, $(v, w) \in Y^\rho | T$ if and only if $v \in T_2$ and $w \in T_1$, thus, in view of (25), proving (ii) of the lemma statement.

First, since $T \subseteq W^\rho$, $w_K^{p_K} \in W^\rho$ for all $K \in \mathscr{J}^-$, hence by (12), $l_{\sigma K} + c_{KJ}^{p_K} + c_{J\tau} < \rho$ for all $K \in \mathscr{J}^-$, and therefore $l^1 + c_{J\tau} < \rho$ so that $\rho = \max\{l^1 + c_{J\tau}; l^1 + l^2\} = l^1 + l^2$.

Next, $T_1 \cap T_2 = \emptyset$. Indeed, suppose $w_K^{p_K} \in T_1 \cap T_2$. Then $l_{\sigma K} + c_{KJ}^{p_K} = l^1$ and $c_{JK}^{p_K} + l_{K\tau} = l^2$, hence $l_{\sigma K} + c_{KJ}^{p_K} + c_{JK}^{p_K} + l_{K\tau} = l^1 + l^2 = \rho$. But $c_{KJ}^{p_K} + c_{JK}^{p_K} \leq 0$, so that $l_{\sigma K} + l_{K\tau} \geq \rho$, a contradiction to $l_{\sigma\tau} < \rho$.

Moreover, let $v = w_K^{p_K}$, $w = w_L^{p_L}$. If $w_K^{p_K} \in T_2$ and $w_L^{p_L} \in T_1$, then $c_{JK}^{p_K} + l_{K\tau} = l^2$ and $l_{\sigma L} + c_{LJ}^{p_L} = l^1$, hence $c_{JK}^{p_K} + l_{K\tau} + l_{\sigma L} + c_{LJ}^{p_L} = l^2 + l^1 = \rho$, and by (13), $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$. Conversely, suppose $(w_K^{p_K}, w_L^{p_L}) \in Y^\rho$. Then by (13), and since T is stable in H, $c_{JK}^{p_K} + l_{K\tau} + l_{\sigma L} + c_{LJ}^{p_L} \geq \rho = l^2 + l^1$. Since also $c_{JK}^{p_K} + l_{K\tau} \leq l^2$ and $l_{\sigma L} + c_{LJ}^{p_L} \leq l^1$, equality must hold in these two inequalities, hence $w_K^{p_K} \in T_2$ and $w_L^{p_L} \in T_1$. □

The initial set T' (denoted T^0 in Sect. 4.3) is determined in lines 13 to 25 based on Lemma 1. All nodes of T_1 can be deleted, since either $T - W^\rho = T_1$ or $T \subseteq W^\rho$ and all nodes of T_1 are successors of all nodes of T_2. Also none of the remaining nodes of $T - T_1$ have successors in $W^\rho | T$. Then either for some $K \in \mathscr{J}^1$, $W_K^\rho | T$ after deletion of T_1 becomes empty and $\mathscr{L}^\rho = \emptyset$, or $w_K^{p_K - 1} \in W_K^\rho | T$ for all $K \in \mathscr{J}^1$: these nodes are appended to $T - T_1$ to form the initial T'. Also, only these nodes need to be scanned.

The following complexity result can now be proven.

Proposition 6 *The OJI-NWJS algorithm runs in time* $\mathcal{O}(n \cdot \max\{n^2, \sum_K q_K\})$.

Proof (i) Initial all-pairs longest paths-computation can be done in $\mathcal{O}(n^3)$.

(ii) For a given T, the computation effort for calculating the makespan and initializing T' and $\mathcal{S}$ (lines 6 to 24) is $\mathcal{O}(n)$. Also, at most $\sum_K q_K$ sets T are generated overall. Therefore the total effort spent in makespan and initialization of T' and $\mathcal{S}$ is $\mathcal{O}(n \sum_K q_K)$.

(iii) We estimate the overall effort spent in the scanning phase by estimating the number of iterations of the inner while loop of the scanning phase, i.e. the number of tests $(w_K^{p'_K}, w_L^p) \in Y^\rho$ (line 30). When scanning $w_K^{p'_K}$ and considering a given $L \in \mathcal{J}^- - K$, let test $(w_K^{p'_K}, w_L^p) \in Y^\rho$ be called a *first* test if $p = p'_L$ and an *additional* test if $p < p'_L$. Now, the number of first tests in the scan of a node is $|\mathcal{J}^- - K| = n - 2$. Also, since each node is scanned at most once, the number of scanned nodes is at most $\sum_K q_K$. Therefore the overall number of first tests is at most $(n-2)\sum_K q_K$. The number of additional tests is less than $\sum_K q_K$ since each additional test $(w_K^{p'_K}, w_L^p) \in Y^\rho$ is performed after p has been decremented by 1. Therefore the overall number of tests is less than $(n-1)\sum_K q_K$. □

References

Adams J, Balas E, Zawack D (1988) The shifting bottleneck procedure for job shop scheduling. Manag Sci 34(3):391–401

Bozejko W, Makuchowski M (2009) A fast tabu search algorithm for the no-wait job shop problem. Comput Ind Eng 56:1502–1509

Cook WJ, Cunningham WH, Pulleyblank WR, Schrijver A (1997) Combinatorial optimization. Wiley-Interscience, New York

Gröflin H, Klinkert A (2007) Feasible insertions in job shop scheduling, short cycles and stable sets. Cent Eur J Oper Res 177:763–785

Gröflin H, Klinkert A (2009) A new neighborhood and tabu search for the blocking job shop. Discrete Appl Math 157:3643–3655

Kis T (2001) Insertion techniques for job shop scheduling. Ph.D. thesis, Ecole Polytechnique Fédérale de Lausanne

Kis T, Hertz A (2003) A lower bound for the job insertion problem. Discrete Appl Math 128:395–419

Lawrence S (1984) Supplement to resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques. GSIA, Carnegie Mellon University, Pittsburgh

Schrijver A (2003) Combinatorial optimization, polyhedra and efficiency. Springer, Berlin

Schuster C (2006) No-wait job shop scheduling: tabu search and complexity of subproblems. Math Methods Oper Res 63(3):473–491

Storer RH, Wu SD, Vaccari R (1992) New search spaces for sequencing problems with application to job shop scheduling. Manag Sci 38(10):1495–1509

van den Broek J (2009) MIP-based approaches for complex planning problems. Ph.D. thesis, Technische Universiteit Eindhoven

Werner F, Winkler A (1995) Insertion techniques for the heuristic solution of the job shop problem. Discrete Appl Math 58(2):191–211

Yamada T, Nakano R (1992) A genetic algorithm applicable to large-scale job-shop problems. In: Parallel problem solving from nature, vol 2, pp 281–290

Zhu J, Li X, Wang Q (2009) Complete local search with limited memory algorithm for no-wait job shops to minimize makespan. Eur J Oper Res 198(2):378–386