Computers and Operations Research

A graph-based constraint programming approach for the integrated process planning and scheduling problem

--Manuscript Draft--

Manuscript Number:	COR-D-19-00455R2
Article Type:	Research Article
Keywords:	Integrated process planning and scheduling; constraint programming; CP Optimizer; graph-based modelling; AND/OR graph
Abstract:	Integration of process planning and scheduling (IPPS) is to carry out both functions simultaneously. This paper provides a graph-based constraint programming (GCP) approach to solve the type-2 IPPS problem that takes AND/OR graphs as input. The proposed GCP approach is implemented based on the IBM ILOG CP Optimizer. AND/OR graph is tailored to cope with IPPS instances. The directed arc defines both precedence and presence relationships. The or-link, a set of mutually-exclusive operations, is defined for alternative process routes. Interval variables and scheduling-oriented constraints are adopted to project the IPPS-specific AND/OR graph to a concise CP model with which minimizing makespan is incorporated as the objective. The GCP approach is tested on a set of benchmark problems. Experimental results show that the proposed approach outperforms compared algorithms on major IPPS instances.

- The integrated process planning and scheduling (IPPS) is solved.
- The AND/OR graph is defined to cope with IPPS representation.
- A graph-based constraint programming model is proposed for IPPS.
- Benchmark tests revealed outstanding performance of the proposed approach.

A graph-based constraint programming approach for the integrated process planning and scheduling problem

Luping Zhang[1], Chunxia Yu[2], T. N. Wong[3]
[1] Southwestern University of Finance and Economics, Chengdu, China.
nguzlp@gmail.com
[2] School of Economics and Management, China University of Petroleum, Beijing, China.
yuchunxiasd@163.com
[3] The University of Hong Kong, Pokfulam Road, Hong Kong.
tnwong@hku.hk

Abstract

Integration of process planning and scheduling (IPPS) is to carry out both functions simultaneously. This paper provides a graph-based constraint programming (GCP) approach to solve the type-2 IPPS problem that takes AND/OR graphs as input. The proposed GCP approach is implemented based on the IBM ILOG CP Optimizer. AND/OR graph is tailored to cope with IPPS instances. The directed arc defines both precedence and presence relationships. The or-link, a set of mutually-exclusive operations, is defined for alternative process routes. Interval variables and scheduling-oriented constraints are adopted to project the IPPS-specific AND/OR graph to a concise CP model with which minimizing makespan is incorporated as the objective. The GCP approach is tested on a set of benchmark problems. Experimental results show that the proposed approach outperforms compared algorithms on major IPPS instances.

Keywords

Integrated process planning and scheduling; constraint programming; CP Optimizer; graph-based modelling; AND/OR graph.

1. Introduction

Process planning and scheduling are two key functions in manufacture planning. Process planning is to design a series of operations for all features of a product with the consideration of manufacturing constraints and limited manufacturing resources. A process plan comprises a set of well-organized operations, sequence constraints, and resource requirements. Scheduling is to allocate manufacturing resources to all necessary operations. In conventional manufacturing, process planning and scheduling are carried out sequentially. At first, a set of alternative process plans is generated by the process planning module. The best process plan that optimises the planning goal is then selected and sent to the scheduling module for resource allocation. The process planning and scheduling modules usually maintain different sets of environmental factors and are performed to achieve different goals, which frequently raises conflicts. For instance, the process planning module cannot sense the real-time status in workshops, which may result in assigning unavailable machines to some operations. Besides, separately conducting the process planning and scheduling functions cannot quickly respond to changes in a dynamic manufacturing environment [1]. Integrated process planning and scheduling (IPPS) is to perform process planning and scheduling simultaneously in a unified framework. There are several IPPS frameworks, including the augmented scheduling system with an embedded process planning module [2], the collaborative

process planning and scheduling system [3], and the unified IPPS framework with AND/OR graphs as inputs [4]. Modelling the IPPS problem as a single decision-making problem has become mainstream in recent years. A set of predetermined alternative process plans is fed to the IPPS system which then selects a process plan for each job and allocates machining resources simultaneously. Since it is a single decision-making procedure, some common objectives can be incorporated. Due to the differences in organizing the predetermined process plans, IPPS falls into two categories: the type-1 IPPS problem that inputs enumerated alternative process plans, and the type-2 IPPS problem that inputs AND/OR graphs [5]. More details about the two types of IPPS problems were discussed in Barzanji et al.'s article [5].

Since the traditional jobshop scheduling problem (JSP) is NP-hard [6], the IPPS problem that combines process selection to the JSP is obviously NP-hard. The problem complexity of IPPS expands exponentially as more operations or machines are involved.

Existing articles reported several approaches to get near-optimal solutions for the IPPS problem. Several heuristic and exact approaches have been investigated, but there is still a strong need for research in the area of type-2 IPPS. Constraint programming (CP) is a powerful exact algorithm for combinatorial optimization. CP uses declarative expressions to formulate constraints and adopts separate technologies to generate feasible solutions, provide lower bounds, and do optimality proofs [7]. CP also. The IBM ILOG CP Optimizer is a model-and-run optimization engine that provides an intuitive modelling language for scheduling problems based on built-in interval variables and constraints, such as $nooverlap(\cdot)$, $alternative(\cdot)$, and $endbeforstart(\cdot)$ [8]. It is also able to perform an automatic search procedure with embedded algorithms. CP is a promising solution approach, according to several existing works. The work described in this paper introduces a graph-based CP (GCP) approach for solving type-2 IPPS problem. The proposed solution approach is implemented with the IBM ILOG CP Optimizer.

The remaining part of this paper is organized as follows. Section 2 reviews typical solution approaches, CP approaches, and AND/OR graph representations in the scheduling problem domain. The GCP model and an illustrative case are elaborated in Section 3. Section 4 exhibits the experimental results on the benchmark problems proposed by Kim, Park [4]. Concluding remarks and suggestions for future work are presented in Section 5.

2. **Related work**

The representation of alternative process plans with AND/OR graphs can be traced back to decades ago. Crowston [9] used AND/OR graphs to represent alternative jobs for a scheduling problem. Mello and Sanderson [10] proposed a compact AND/OR graph representation to support the development of assembly planning algorithms. Gillies and Liu [11] proposed two systems to schedule tasks with AND/OR precedence constraints on identical processors. Beck and Fox [12] provided another compact representation for alternative activities by integrating the XorNodes to indicate the start and end of alternative activity paths. Barták and Cepek [13] proposed a general temporal network named P/A graph, a precedence graph with parallel and alternative branching to model the alternative manufacturing processes. A fan-like structure for the P/A graph was defined. Moffitt, Peintner [14] augmented the Disjunctive Temporal Problems with a finite-domain component to each element of the problem. The finite-domain component reflects choice among a set of options. Tao and Dong [15] solved the multi-mode resource-constrained project scheduling problem with the consideration of alternative project

structure. The AND-OR network was employed to represent alternative bridge construction projects. A hybrid metaheuristic solution approach was provided as well. The experimental results revealed the advantages of AND/OR network representation in solving this specific problem.

In the last decade, IPPS studies were mainly focused on type-1 IPPS problem. Several mathematical models for type-1 IPPS problem can be found [5, 16]. Process plans for each job were fed to the models as a whole. Lihong and Shengping [17] and Jin, Zhang [18] proposed two models to deal with type-2 IPPS problem, but both work on the enumerated operation combinations for jobs generated in a preliminary stage. Both models do not directly use AND/OR graph input, which makes them more like models for type-1 IPPS problem. Solution approaches have to frequently replace an entire process plan for a job during the search procedure. AND/OR graph can establish a compact structure on a set of process plans for the same job, which provides a possibility to partly deal with process plans. Regarding solution approaches, several typical heuristic algorithms were successfully implemented, for instance, the symbiotic evolutionary algorithm [4]. Heuristics approaches for the IPPS problem usually were not constructed to solve mathematical models directly. Genetic algorithms (GAs) have drawn massive attention in the IPPS problem domain. Successful GA-based approaches include a simulation-based GA [19], a modified GA[20], an improved GA [17], an object-coding GA [21], etc. Li, Gao [22] combined the GA and variable neighbourhood search for the IPPS problem in a packaging machine workshop. An effective multi-objective GA can be found [23]. Other approaches such as a simulated annealing approach [24], a particle swarm optimization algorithm [25], a memetic algorithm [18], and an ant colony optimization algorithm [1] were also applied to solve the IPPS problem. Wong, et al. (2006) proposed a Multi-Agent Negotiation approach (MAN) and a Hybrid-based Agent Negotiation approach (HAN) for the integration of process planning and scheduling/rescheduling [26, 27]. Recently, an exact decomposition algorithm for type-1 IPPS problem was reported [5]. Besides, Lin, Li [28] addressed the IPPS problem in a distributed flexible jobshop. They also employed the and-only graph representation for the solution approach named GA_X. Until now, no available type-2 IPPS models that directly deal with AND/OR graphs can be found in literature.

Existing literature reported successful CP implementations for a wide range of scheduling problems. Baptiste, Le Pape [29] systematically reported applications of CP to scheduling problems. In their book, a CP framework for typical JSPs and corresponding search-based solution approaches were reported. At the early stage, CP was used solely as a solution approach for scheduling problems. Harjunkoski and Grossmann [30] coupled a MIP model and a CP model to solve scheduling problems in two separate stages. Sadykov and Wolsey [31] proposed a CP-based approach for a multi-machine assignment scheduling problem. Besides, some researchers combined CP and local search approaches. For instance, Beck, Feng [32] combined CP and tabu search to solve JSPs. Bartak, Salido [33] established a theoretical framework that implies constraint satisfaction technologies for planning and scheduling problems. The integration of planning and scheduling was discussed. However, it is just a theoretical framework that cannot be used directly in practices. Timpe [34] established a more specific model for general planning and scheduling problems with a combined IP and CP approach. Ham and Cakici [35] proposed a CP model for the flexible JSP with parallel batching machines. They also compared the CP model with a MIP model, which showed that CP outperforms MIP in solving the given scheduling problem. Zhang and Wang [36] formulated a CP model and a MIP model for the flexible assembly jobshop scheduling problem in a dynamic manufacturing environment. They also revealed that CP performs better than MIP in the specific problem category. Laborie [37] formulated a standalone tiny CP model for resource allocation and

scheduling problem and solved it with an improved commercial CP solver. Experimental results showed that the CP model outperforms all compared approaches on the 335 benchmark instances. Kreter, Schutt [38] proposed six different CP models to solve the resource-constrained project scheduling problem in terms of minimizing project duration, and the extended benchmark tests showed that the CP model is highly competitive compared to existing MIP models.

To conclude, the IPPS approach that integrates process selection and JSPs is an important scheduling problem. type-2 IPPS problem packs alternative process plans in AND/OR graphs, which provides a higher level of flexibility to modelling and solution approaches. Several works reported the outstanding performance of CP in solving scheduling problems. This research takes advantages of CP and AND/OR graph. An approach that combines CP and AND/OR graphs to solve type-2 IPPS problem is proposed. A concise GCP model based on the CP optimizer is provided.

3. The GCP model

A typical IPPS problem is to process n jobs on m machines in a jobshop environment. A job consists of a set of operations. It is possible to arrange a set of operations in different orders, which is referred to as process routes, to complete the same job. Precedence constraints between operations have to be satisfied. Besides, it is possible to process an operation with different machines [26].

3.1 The type-2 IPPS problem and AND/OR graph representation

An illustrative case is given in Fig. 1, supposing there are two parts, a holder and a bolt, to be manufactured. The holder has 5 features, including 6 surfaces (F1.1, F1.2), a groove (F1.3), a hole (F1.4), and a chamfer (F1.5). The top surface of the holder has to be specially handled due to a high level of required accuracy. The bolt has 7 features, including two surfaces (F2.1, F2.3), an outer diameter (F2.2), a slot (F2.7), a relief groove (F2.5), a thread (F2.6), and a chamfer (F2.4).

3.1.1 The AND/OR graph for type-2 IPPS problem

A set of operations, as shown in Fig. 2, is designed to process the features of two parts. A node represents an operation, and a directed arc represents a sequence constraint.

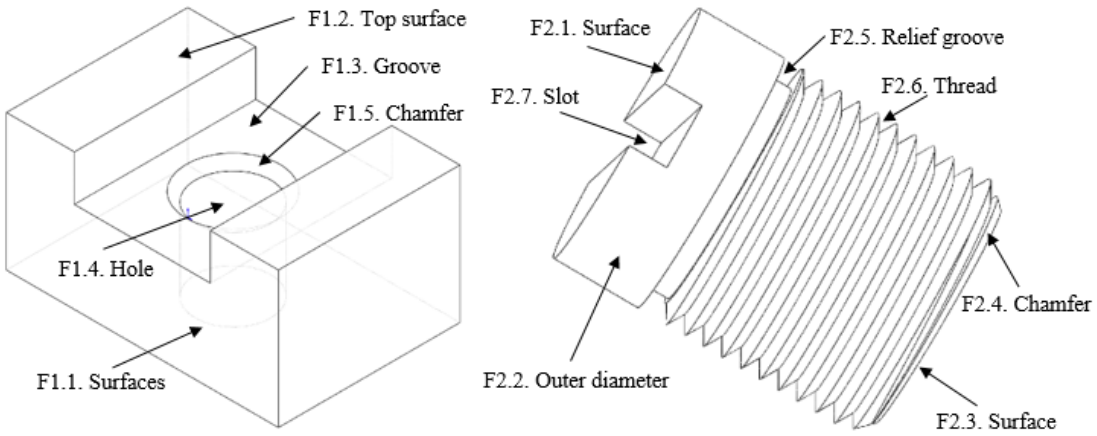

Fig. 1. An IPPS example with two parts.

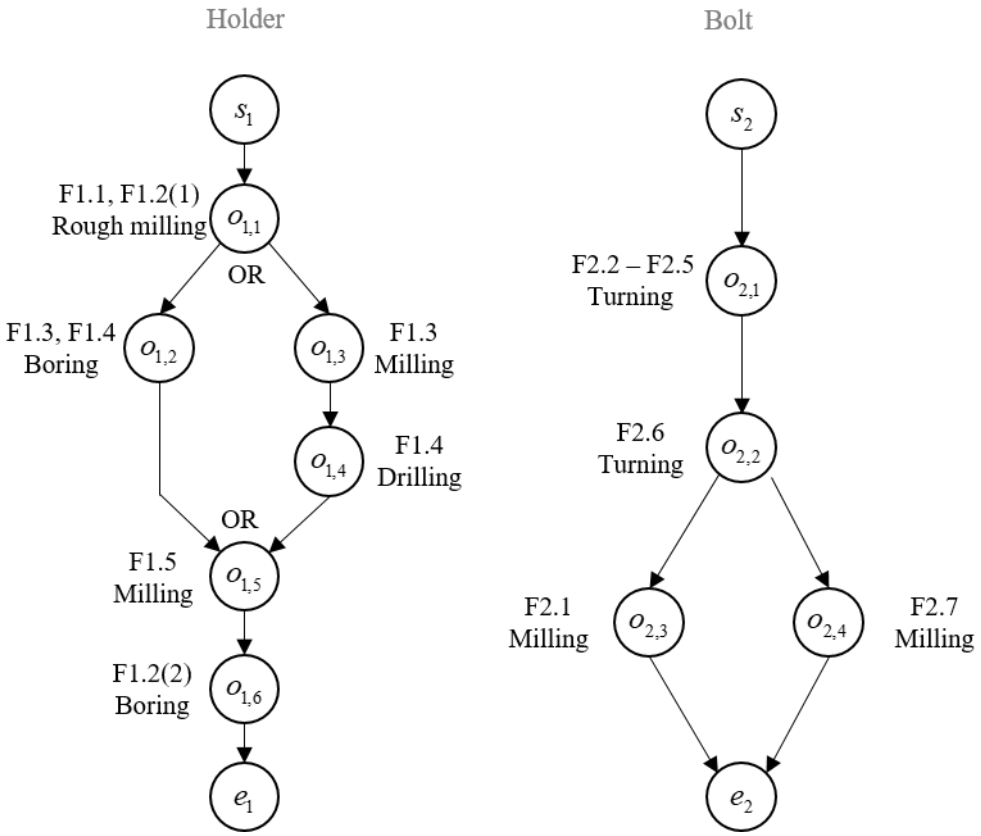

Fig. 2. The AND/OR graph for the IPPS example.

A pair of dummy nodes s_j and e_j are introduced to indicate the start and end points of job j. Operation $o_{1,1}$ is designed to process Feature F1.1 and roughly machine the top surface F1.2 with a single setup. Since the top surface requires a higher level of accuracy, another operation $o_{1,6}$ is designed to achieve it. Both the groove F1.3 and hole F1.4 can be done by either to milling $o_{1,3}$ on a milling machine and drilling $o_{1,5}$ on a driller sequentially, or performing the single operation $o_{1,2}$ on a boring machine with a single setup. Therefore, either the operation sequence $(o_{1,2})$ or $(o_{1,3}, o_{1,4})$ can finish both features. As a result, they are two mutually exclusive operation sequences indicated by an or-relationship. As reflected in Fig. 2, two alternative paths are initiated by operation $o_{1,1}$ and terminated by operation $o_{1,5}$. Directed arcs explicitly indicate sequence constraints.

Besides, the surfaces F2.3, outer diameter F2.2, chamfer F2.4, and relief groove F2.5 of the bolt have to be finished before threading F2.6. A lathe can finish features F2.2–F2.5 with the single operation $o_{2,1}$. Operation $o_{2,2}$ for turning the thread follows. After that, either operation $o_{2,3}$ for

cutting surface F2.1 or $o_{2,4}$ for cutting the slot can be started. There is no sequence constraint between them. As a result, both $o_{2,3}$ and $o_{2,4}$ are connected to $o_{2,2}$, indicating that both of them have to be performed and either can be started immediately after finishing $o_{2,2}$. The structure among $o_{2,2}$, $o_{2,3}$ and $o_{2,4}$ implicitly sketches an and-relationship. Since there are both or-relationships and and-relationships, the graph is defined as an AND/OR graph [4, 10]. For the IPPS problem, three types of flexibility are inherited from Kim et al.'s work [39].

(1) Processing flexibility: the possibility to fulfil a job with different sets of operations. As shown in Fig. 2, two operation sets $\{o_{1,1}, o_{1,2}, o_{1,5}, o_{1,6}\}$ and $\{o_{1,1}, o_{1,3}, o_{1,4}, o_{1,5}, o_{1,6}\}$ are both capable of finishing the handler. The operation sets for the same job are mutually exclusive. Only one process sequence should be selected for a job and present in the final schedule.

(2) Operation flexibility: the possibility to process an operation with different machines. Taking operation $o_{1,4}$ as an example, drilling the hole can be done by a lathe, CNC lathe, milling machine, or boring machine.

(3) Sequencing flexibility: the possibility to arrange a set of operations in different orders for a job. Regarding the illustrative IPPS instance in Fig. 2, the set of operations for the bolt can be arranged in two different sequences by putting one of $o_{2,3}$ and $o_{2,4}$ before the other, since both of them need to be processed, but there is no sequence constraint between them.

3.1.2 General definition and augmentation of AND/OR graph

AND/OR graph $G=(O,A,L)$ for the IPPS problem comprises a set of nodes O for operations, a set of directed arcs A for sequence and presence relationships, and a set of or-links L for alternative process routes. A node is denoted by $o_{j,u}$ with a subscript j to index the job and u to index the operation. An arc is written as $o_{j,u} \rightarrow o_{j,v}$, indicating that $o_{j,v}$ cannot be started until $o_{j,u}$ is finished. $o_{j,u}$ is called a direct predecessor of $o_{j,v}$, and $o_{j,v}$ is called a direct successor of $o_{j,u}$. In almost all cases, sequential relationships are only defined between operations of the same job since jobs are usually independent of each other. An alternative process route is called an or-branch. The or-branches for the same set of features are called a group of peer or-branches. Peer or-branches in the same group are mutually exclusive. A pair of or-links are used to indicate the start or end points of peer or-branches in the same group. If any or-branch has only one start node and one end node, an or-link can be represented by the set of start nodes or end nodes of the peer or-branches. Besides, a pair of dummy nodes, an or-initiator and an or-terminator is introduced to enclose a group of peer or-branches. The augmented AND/OR graph for the illustrative IPPS case

is shown in Fig. 3. There are two or-links $OR\{o_{1,2},o_{1,3}\}$ and $OR\{o_{1,2},o_{1,4}\}$ for the two peer or-branches which are enclosed by the or-initiator $o_{1,11}$ and or-terminator $o_{1,12}$ as shown in Fig. 3. The prefix OR distinguishes an or-link from an ordinary set. A hyperarc $o_{1,11} \to OR\{o_{1,2},o_{1,3}\}$ is defined, which can be regarded as a set of mutually-exclusive arcs $OR\{o_{1,11} \to o_{1,2}, o_{1,11} \to o_{1,3}\}$. Likewise, a hyperarc $OR\{o_{1,2},o_{1,4}\} \to o_{1,12}$ is defined to terminate the or-branches. Other than precedence relationship, arcs in this framework carry presence relationships as well. For example, the simple arc $o_{1,3} \to o_{1,4}$ implies the co-presence of operations $o_{1,3}$ and $o_{1,4}$. The presence of an or-link synchronizes with the presence of any operation in it. As a result, a hyperarc implies the co-presence of an or-initiator and its succeeding or-link (resp. an or-terminator and its preceding or-link). For instance, the hyperarc $o_{1,11} \to OR\{o_{1,2},o_{1,3}\}$ implies the co-presence of exactly one operation in or-link $OR\{o_{1,2},o_{1,3}\}$ and operation $o_{1,11}$.

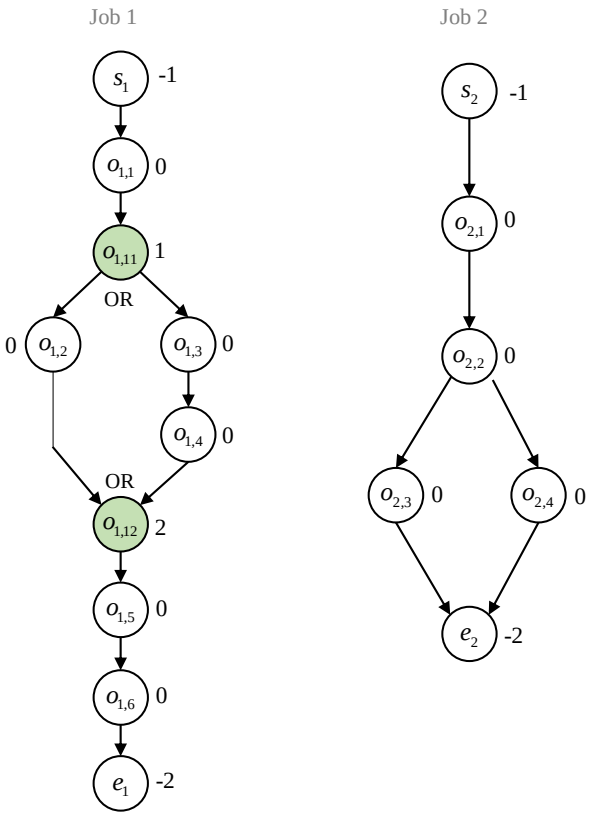

Fig. 3. The augmented AND/OR graph

3.1.3 Assumptions

Problem-oriented assumptions P1–P5 as given in the following are inherited from literature [4].

Besides, to make AND/OR graph effective for the IPPS representation, three more assumptions G1–G3 are exerted.

Assumptions on the IPPS problem

P1. Operations are not-preemptive. An operation once started cannot be interrupted until it is finished.

P2. Operations of the same job cannot overlap.

P3. Operations allocated to the same machine cannot overlap.

P4. All jobs are released at time *zero*. An operation can be started immediately after all its necessary predecessors are finished.

P5. Internal logistics and setup consumption are neglected or absorbed by the processing times of operations.

Assumptions on AND/OR graph for the IPPS problem

G1. A node can be the source (or target) of either a set of simple arcs or a single hyperarc, but not both.

G2. An or-branch has exactly one start node and one end node.

G3. An or-branch can be disconnected from its peer or-branches by removing the or-initiator and or-terminator that enclose it.

The purpose of proposing assumptions G1–G3 is threefold. First, it prevents illegal IPPS representations. Regarding the two or-branches $(o_{1,2})$ and $(o_{1,3}, o_{1,4})$, a hypothetical arc $o_{1,2} \rightarrow o_{1,4}$ will raise a logic conflict: $o_{1,2}$ and $o_{1,4}$ cannot be present at the same time since they are on two different peer or-branches. However, arc $o_{1,2} \rightarrow o_{1,4}$ forces the co-presence of the two operations. Assumption G3 prevents the existence of such illegal connection, since the branch $(o_{1,2})$ is still connected to its peer or-branch $(o_{1,3}, o_{1,4})$ even if the or-initiator $o_{1,11}$ and the or-terminator $o_{1,12}$ are removed. Secondly, Assumptions G1 and G2 simplify the modelling in a way that the presence relationship between an or-initiator and its succeeding or-link (resp. or-terminator and its preceding or-link) can be easily formulated. Satisfying assumptions G1 and G2 is not difficult since it is always possible to add a dummy node at the beginning or to the end of an or-branch without numerically change the instance. Thirdly, assumptions G1-G3 establish a one-to-one dependency between an or-initiator and its succeeding or-link (resp. an or-terminator and its preceding or-link). There will be an or-link if and only if there is a preceding or-initiator or a succeeding or-terminator. In that case, it just needs to define a set of or-initiators and a set of or-terminals for the model.

3.2 Variables and notations

Let $(o_{j,u}, k)$ be the assignment of allocating $o_{j,u}$ to machine k. A dummy machine k^d is

assigned to process all dummy operations with zero processing time. The time interval variable $I(o_{j,u},k)$ is designated for any possible assignment. Besides, a time interval $I(o_{j,u})$ is needed for each operation without specified processing machine. The variable $I(o_{j,u})$ will be synchronized with a time interval in the set $\{(I_{j,u},k)\,|\,k\in M(o_{j,u})\}$ by the constraint $alternative(I(o_{j,u}),\{(I_{j,u},k)\,|\,k\in M(o_{j,u})\})$, implying the allocation of a processing machine to operation $o_{j,u}$. Due to the operation flexibility, $I(o_{j,u},k)$ is optional if there are multiple machines capable of processing $o_{j,u}$. Due to the processing flexibility, $I(o_{j,u})$ is optional if $o_{j,u}$ is on an or-branch. To simplify the modelling, all interval variables are set to be optional. The presences of $I(o_{j,u})$ are regulated by arcs. Table 1 summarizes the notations.

Table 1. Notations

Problem-oriented factors	
j,k	Indices for jobs and machines respectively.
u,v	Indices for operations.
J	The set of jobs of an IPPS instance $J=\{j\,|\,j=1,2,3,...,n\}$.
M	The set of machines $M=\{k\,|\,k=1,2,3,...,m\}$.
$o_{j,u}$	An operation of job j indexed by u.
$M(o_{j,u})$	The set of machines capable of processing $o_{j,u}$.
O	The set of all operations of the IPPS instance.
O_j	The set of operations of job j, $O_j=\{o_{i,u}\in O\,|\,i=j\}$.
$(o_{j,u},k)$	The assignment of allocating $o_{j,u}$ to machine k.
P	The set of all possible assignments $P=\{(o_{j,u},k)\,|\,o_{j,u}\in O,k\in M(o_{j,u})\}$.
$\tau(o_{j,u},k)$	The processing time of assignment $(o_{j,u},k)$.
$o_{j,u}\rightarrow o_{j,v}$	The directed arc that connects from $o_{j,u}$ to $o_{j,v}$.
A	The set of directed arcs.
s_j and e_j	The dummy start and end nodes of job j.
R^s,R^e	The set of or-initiators and the set of or-terminators respectively.

Variable-oriented factors

$I(o_{j,u})$	The time interval for operation $o_{j,u}$.
$I(o_{j,u},k)$	The time interval for assignment $(o_{j,u},k)$.
I	A general time interval.
$presenceOf(I)$	The function associated to the presence status of interval I.
Q_j^J	A set of time intervals for all operations of job j, $Q_j^J=\{I(o_{j,u})\mid o_{j,u}\in O_j\}$.
Q_k^M	A set of time intervals for all assignments allocated to machine k, $Q_k^M=\{I(o_{j,u},k_0)\mid (o_{j,u},k)\in P,k_0=k\}$.
$c_{\max}$	The makespan.

3.3 The GCP model

A formulation in terms of minimizing the makespan is formulated as follows. More details about the CP constraints can be found in the CP Optimizer reference manual and Laborie [40].

The GCP Model

$$Minimize\ c_{\max}=\max\{end(I(e_j))\mid j\in J\} \tag{1}$$

subject to:

$$alternative(I(o_{j,u}),\{I(o_{j,u},k)\mid k\in M(o_{j,u})\}),\ \forall o_{j,u}\in O \tag{2}$$

$$nooverlap(Q_j^J),\ \forall j\in J \tag{3}$$

$$nooverlap(Q_k^M),\ \forall k\in M \tag{4}$$

$$endbeforestart(I(o_{j,u}),I(o_{j,v})),\ \forall o_{j,u}\to o_{j,v}\in A \tag{5}$$

$$presenceOf(I(s_j))=1,\ \forall j\in J \tag{6}$$

$$presenceOf(I(o_{j,u}))=presenceOf(I(o_{j,v})),\ \forall o_{j,u}\to o_{j,v}\in A, o_{j,u}\notin R^s, o_{j,v}\notin R^e \tag{7}$$

$$presenceOf(I(o_{j,u}))=\sum_{o_{j,v}\in O,s.t.\,o_{j,u}\to o_{j,v}\in A} presenceOf(I(o_{j,v})),\ \forall o_{j,u}\in R^s \tag{8}$$

$$\sum_{o_{j,u}\in O,s.t.\,o_{j,u}\to o_{j,v}\in A} presenceOf(I(o_{j,u}))=presenceOf(I(o_{j,v})),\ \forall o_{j,v}\in R^e \tag{9}$$

$$Interval\ I(o_{j,u}),optional,\ for\ all\ o_{j,u}\in O \tag{10}$$

$$Interval\ I(o_{j,u},k),optional,size=\tau(o_{j,u},k),\ for\ all\ (o_{j,u},k)\in P \tag{11}$$

The two types of time interval variables are defined by (10) and (11) respectively. The objective (1) is to minimize the makespan. Constraint (2) is to allocate an operation to exactly one capable machine. Constraint (3) states that the operation intervals of the same job do not overlap. Likewise, Constraint (4) regulates that assignment intervals allocated to the same machine do not overlap. Constraint (5) regulates precedence relationships defined by directed arcs. Constraints (6)–(9) spread out presence relationships along arcs: Constraint (6) states the release of each job; Constraint (7) forces the co-presence of two directly-connected ordinary operations that are neither or-initiators nor or-terminators; Constraint (8) regulates the co-presence of an or-initiator and its succeeding or-link; Constraint (9) forces the copresence of an or-terminator and its preceding or-link.

3.4 An illustrative case

The projection of the illustrative IPPS graph to the main constraints of the GCP model is provided in Fig. 4. Details are listed in the following.

- $nooverlap\left(Q_2^J=\left\{I\left(o_{2,1}\right),I\left(o_{2,2}\right),I\left(o_{2,3}\right),I\left(o_{2,4}\right)\right\}\right)$. The intervals $I\left(o_{2,1}\right),I\left(o_{2,2}\right),I\left(o_{2,3}\right)$ and $I\left(o_{2,4}\right)$ for job 2 cannot overlap with each other. The intervals for the dummy start and end operations s_2 and e_2 are excluded from Q_2^J since their processing times are equal to zero.
- $presenceOf\left(I\left(s_2\right)\right)=1$. Job 2 must be started.
- $presenceOf\left(I\left(o_{1,11}\right)\right)=\sum_{o_{1,v}\in\left\{o_{1,2},o_{1,3}\right\}} presenceOf\left(I\left(o_{1,v}\right)\right)$ defines the co-presence of the or-initiator $o_{1,11}$ and its succeeding or-link $OR\left\{o_{1,2},o_{1,3}\right\}$.
- $presenceOf\left(o_{2,4}\right)=presenceOf\left(I\left(o_{2,2}\right)\right)$ takes care of the co-presence of two ordinary operations $o_{2,4}$ and $o_{2,2}$ connected by the simple arc $o_{2,2}\rightarrow o_{2,4}$.
- $\sum_{o_{1,u}\in\left\{o_{1,2},o_{1,4}\right\}} presenceOf\left(I\left(o_{1,u}\right)\right)=presenceOf\left(I\left(o_{1,12}\right)\right)$ defines the presence dependency between the or-link $OR\left\{o_{1,2},o_{1,4}\right\}$ and its succeeding or-terminator $o_{1,12}$.
- $endbeforestart\left(I\left(o_{1,5}\right),I\left(o_{1,6}\right)\right)$ exerts the sequence constraint defined by the arc $o_{1,5}\rightarrow o_{1,6}$.

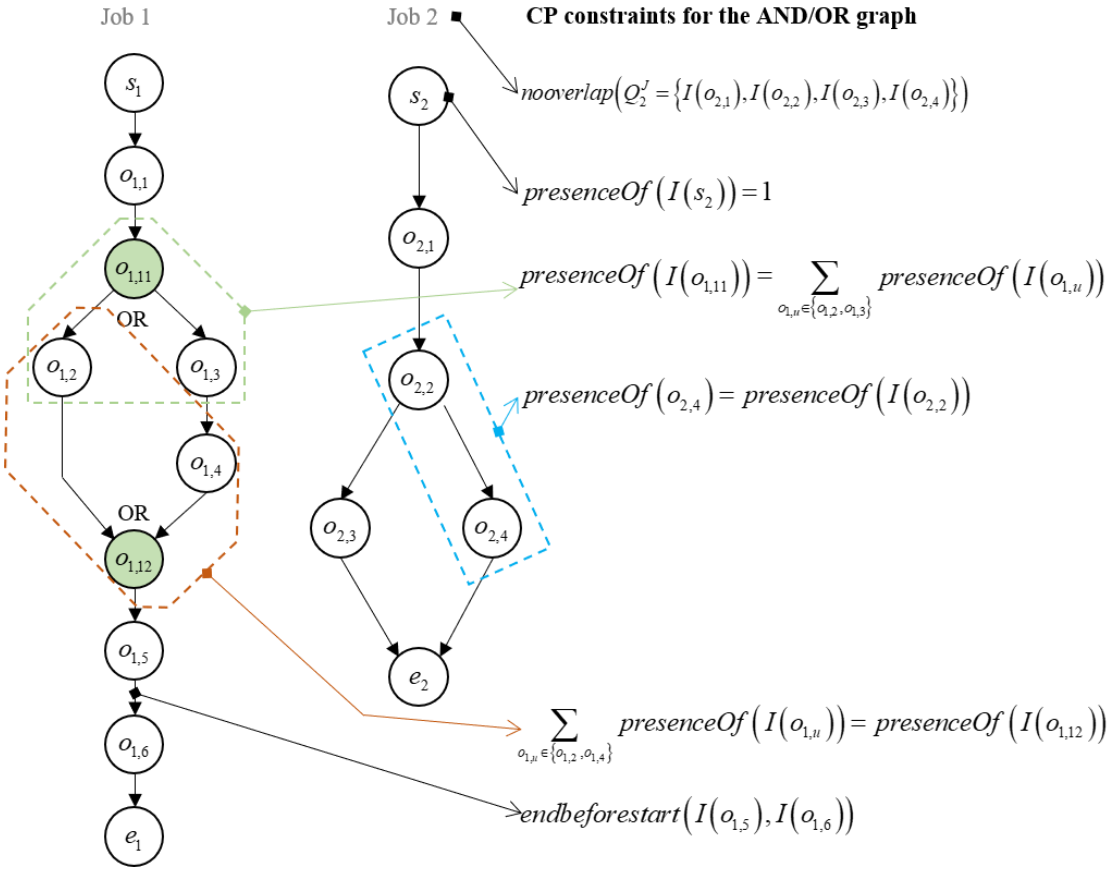

Fig. 4. Projecting the AND/OR graph to CP constraints.

Suppose there are 5 available machines, including a lathe (M1), a CNC lathe (M2), a milling machine (M3), a boring machine (M4), and a drilling machine (M5) to process the illustrative IPPS case. A dummy machine M0 is designated to process all dummy nodes $\{s_1,s_2\}\cup\{e_1,e_2\}\cup\{o_{1,11}\}\cup\{o_{1,12}\}$. Given the process data in Table 2, the graph-oriented data can be generated, as shown in Table 3.

Table 2. Processing data for the illustrative case

Operation $o_{j,u}$	Capable machines $M\left(o_{j,u}\right)$	Processing times $\tau\left(o_{j,u},k\right)$	Operation $o_{j,u}$	Capable machines $M\left(o_{j,u}\right)$	Processing times $\tau\left(o_{j,u},k\right)$
$o_{1,1}$	M3, M4	30, 24	$o_{1,6}$	M4	16
$o_{1,2}$	M4	16	$o_{2,1}$	M1, M2	34, 14
$o_{1,3}$	M3	8	$o_{2,2}$	M1, M2	18, 14
$o_{1,4}$	M1, M2, M5	18, 10, 14	$o_{2,3}$	M3	16

$o_{1,5}$	M3, M4	14, 10	$o_{2,4}$	M3	12

Table 3. Graph-oriented data for the illustrative IPPS case

Set	Data
J	$\{1,2\}$
M	$\{0,1,2,3,4,5\}$
O	$\{o_{1,1}, o_{1,2}, o_{1,3}, o_{1,4}, o_{1,5}, o_{1,6}, o_{2,1}, o_{2,2}, o_{2,3}, o_{2,4}, s_1, s_2, e_1, e_2, o_{1,11}, o_{1,12}\}$
$M(o_{j,k})$	$M(o_{1,1})=\{3,4\}$, $M(o_{1,2})=\{4\}$, $M(o_{1,3})=\{3\}$, $M(o_{1,4})=\{1,2,5\}$, etc.
O_j	$O_1=\{o_{1,1}, o_{1,2}, o_{1,3}, o_{1,4}, o_{1,5}, o_{1,6}, o_{1,11}, o_{1,12}, s_1, e_1\}$, $O_2=\{o_{2,1}, o_{2,2}, o_{2,3}, o_{2,4}, s_2, e_2\}$
P	$\{(o_{1,1},3),(o_{1,1},4),(o_{1,2},4),(o_{1,3},3),(o_{1,4},1),...\}$
A	$\{s_1 \to o_{1,1}, o_{1,1} \to o_{1,11}, o_{1,11} \to o_{1,2}, o_{1,11} \to o_{1,3}, o_{1,2} \to o_{1,12}, o_{1,3} \to o_{1,4}, o_{1,4} \to o_{1,12}, o_{1,12} \to o_{1,5}, o_{1,5} \to o_{1,6}, o_{1,6} \to e_1, s_2 \to o_{2,1}, o_{2,1} \to o_{2,2}, o_{2,2} \to o_{2,3}, o_{2,2} \to o_{2,4}, o_{2,3} \to e_2, o_{2,4} \to e_2\}$
R^s & R^e	$R^s=\{o_{1,11}\}$, $R^e=\{o_{1,12}\}$.

It can be seen that the operations fall into five categories. It should be easier to maintain a single set of nodes and identify the type of each node with a parameter. For instance, let o_i be an operation in set O, and $T(o_i)$ be the type parameter of o_i. Assign $T(o_i)=-1$ to start nodes, $T(o_i)=-2$ to end nodes, $T(o_i)=1$ to or-initiators, $T(o_i)=2$ to or-terminators, and $T(o_i)=0$ to ordinary nodes as shown in Fig. 3. Presence constraint of two ordinary nodes can be written as

$$presenceOf(I(o_1)) = presenceOf(I(o_2)), \forall o_1 \to o_2 \in A, T(o_1) \neq 1, T(o_2) \neq 2 \quad (12)$$

Other constraints can be rewritten in the same way. The type definition can provide convenience to the implementation. The IBM ILOG CPLEX Optimization Studio V12.10 is used for this case. An optimal schedule generated in a run is sketched in Fig. 5, where all dummy operations are neglected. It is easy to verify that all constraints are satisfied. The system proposes the use of CNC lathe M2, milling machine M3, and boring machine M4 for the IPPS instance. It is meaningful since the three machines are more efficient than lathe M1 and drilling machine M5 according to the processing data.

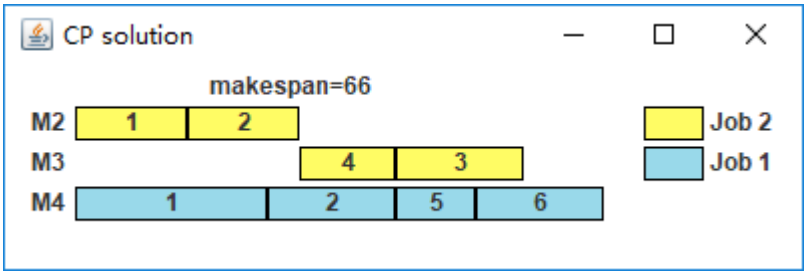

Fig. 5. An optimal schedule for the illustrative case.

4. Experiments

The problem set proposed by Kim, et al. [4] is adopted for benchmark tests. Purposely-designed 18 jobs consisting of around 300 operations are combined in different ways, which generates 24 problems of different levels of flexibility. The problems' specification of the testbed as shown in Table 4 is archived in the dataset document. The lower bound is the maximum of the smallest possible total processing time of each job. Let O_j^* be a set of operations capable of fulfilling job j. A simple lower bound is calculated by

$$\max_{j\in J}\left\{\min_{all\ possible\ O_j^*}\left\{\sum_{o_{j,u}\in O_j^*}\min_{k\in M(o_{j,u})}\tau(o_{j,u},k)\right\}\right\} \tag{13}$$

The experiments are conducted on Lenovo ThinkStation P720 equipped with 12-core Xeon® Silver 4116 CPU @2.1GHz. IBM ILOG CPLEX Optimization Studio V12.10 is used to formulate and run the model. All settings are default. The CP Optimizer automatically implements the newest iterative diving search method for all problems and performs aggressive dives in search tree without backtrack [8].

Table 4. Specification of the benchmark problems

Problem	Job set	Number of jobs	Lower bound
1	1, 2, 3, 10, 11, 12	6	427
2	4, 5, 6, 13, 14, 15	6	343
3	7, 8, 9, 16, 17, 18	6	344
4	1, 4, 7, 10, 13, 16	6	306
5	2, 5, 8, 11, 14, 17	6	304
6	3, 6, 9, 12, 15, 18	6	427
7	1, 4, 8, 12, 15, 17	6	372
8	2, 6, 7, 10, 14, 18	6	342
9	3, 5, 9, 11, 13, 16	6	427
10	1, 2, 3, 5, 6, 10, 11, 12, 15	9	427
11	4, 7, 8, 9, 13, 14, 16, 17, 18	9	344
12	1, 4, 5, 7, 8, 10, 13, 14, 16	9	306
13	2, 3, 6, 9, 11, 12, 15, 17, 18	9	427
14	1, 2, 4, 7, 8, 12, 15, 17, 18	9	372
15	3, 5, 6, 9, 10, 11, 13, 14, 16	9	427

16	1, 2, 3, 4, 5, 6, 10, 11, 12, 13, 14, 15	12	427
17	4, 5, 6, 7, 8, 9, 13, 14, 15, 16, 17, 18	12	344
18	1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17	12	306
19	2, 3, 5, 6, 8, 9, 11, 12, 14, 15, 17, 18	12	427
20	1, 2, 4, 6, 7, 8, 10, 12, 14, 15, 17, 18	12	372
21	2, 3, 5, 6, 7, 9, 10, 11, 13, 14, 16, 18	12	427
22	2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18	15	427
23	1, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 17, 18	15	372
24	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18	18	427

4.1 Global performance test

The *FailLimit* is a parameter embedded in the IBM ILOG CP Optimizer that limits the number of failures allowed before terminating a search. Setting the *FailLimit* number usually depends on problem scales. In this experiment, the *FailLimit* is set to be 500,000 for all problems to test the performance of the GCP model with reasonable computing resources. The GCP model is compared to the Symbiotic Evolutionary Algorithm (SEA) [4], Improved Genetic Algorithm (IGA) [17], Object-Coding Genetic Algorithm (OCGA) [21], and Enhanced Ant Colony Optimization (E-ACO) [41].

Table 5. Average makespans found by SEA, IGA, OCGA, and GCP

Problem	Lower bound	SEA	IGA	OCGA	E-ACO	GCP	Best	Improvement rate (%)	Runtime of GCP (s) †
1	427	437.6	**427***	**427***	427.1	**427***	–	–	0.62
2	343	349.7	344.5	343.5	343.1	**343***	GCP	0.03	3.2
3	344	355.2	351	346.4	345	**344***	GCP	0.29	2.5
4	306	306.2	307.4	310.1	307.6	**306***	GCP	0.07	3.43
5	304	323.7	**309.8**	323	319.6	318	IGA	-2.65	2.78
6	427	443.8	**427***	**427***	427.1	**427***	–	–	0.89
7	372	372.4	372.7	373.3	**372***	**372***	–	–	2.07
8	342	348.3	357	343.5	343.3	**343**	GCP	0.09	3.43
9	427	434.9	**427***	**427***	427.1	**427***	–	–	0.86
10	427	456.5	431.6	427.1	427.6	**427***	GCP	0.02	0.64
11	344	378.9	379.7	350.6	350.2	**344***	GCP	1.77	4.47
12	306	332.8	323.7	324.7	323.4	**318**	GCP	1.67	4.61
13	427	469	442.8	427.2	427.6	**427***	GCP	0.05	0.96
14	372	402.4	415.3	377.4	374.3	**372***	GCP	0.61	3.01
15	427	445.2	427.4	**427***	427.3	**427***	–	–	1.07
16	427	478.8	449.4	428.1	430.9	**427***	GCP	0.26	1.36
17	344	448.9	426	383.1	381.2	**344***	GCP	9.76	7.39
18	306	389.6	373.6	354.5	361.5	**344**	GCP	2.96	6.05
19	427	508.1	471.3	433.7	434.9	**427***	GCP	1.54	2.03
20	372	453.8	446.6	390.5	392.4	**372***	GCP	4.74	4.64
21	427	483.2	447.8	**427***	429.4	**427***	–	–	1.5
22	427	548.3	508.1	452.9	447.2	**427***	GCP	4.52	3.98

23	372	507.5	477.8	410	420.3	**378**	GCP	7.8	6.51
24	427	602.2	548.5	471.2	479.3	**436**	GCP	7.47	10.33

–: Some compared algorithm generated the best makespan together with GCP.

*: The makespan is optimal.

†: The 's' stands for seconds.

Table 5 reports the makespans found by different approaches for the 24 problems. The first column lists problem indices. The second column gives the lower bounds. The third to seventh columns list the mean makespans found by each algorithm in several runs respectively. Best makespans are highlighted in bold. The improvement rate is the improved percentage by GCP to the best makespan found by the compared algorithms. The runtimes of GCP in the given experimental environment are listed in the last column. For problem No. 5, IGA outperforms GCP by 2.65%. GCP outperforms the compared algorithms on the other 17 problems. For further six problems, both GCP and some compared algorithm find the optimal solutions. GCP finds optimal solutions for 18 problems in total. For simpler problems No. 1 to 16, the improvement is not significant since lower bounds are already or nearly reached by the OCGA. With regard to more complex problems No. 17 to 24, except for problem No. 21, the solutions are significantly improved. The runtimes show that the GCP approach works efficiently on this problem set. Near-optimal solutions can be found quickly in the given experimental environment.

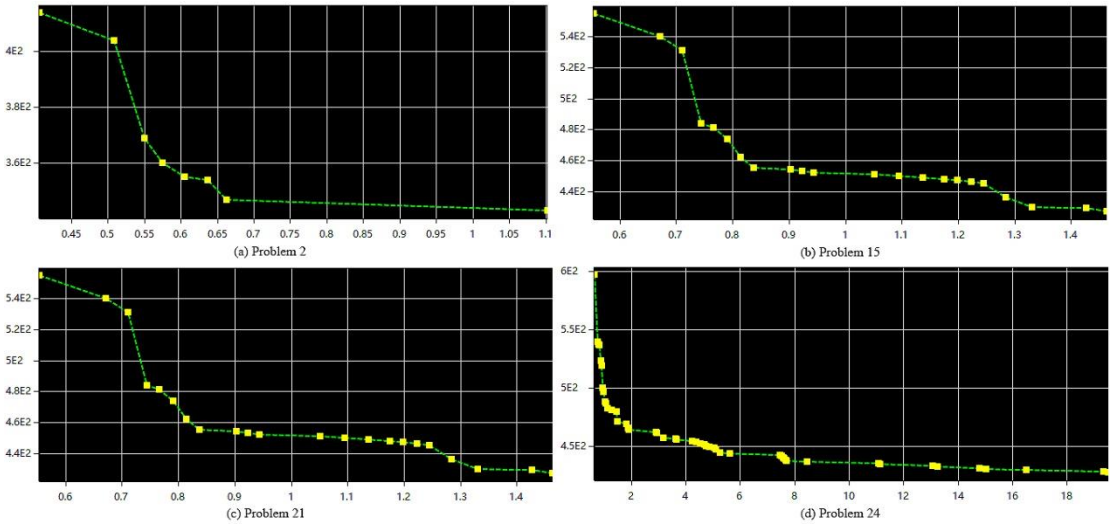

Fig. 6. The search procedures for solving problems No. 2, 15, 21 and 24

The search procedures for solving representative problems No. 2, 15, 21, and 24 are depicted in Fig. 6. The horizontal axis represents the timeline, and the vertical axis represents the makespan. The problem-scale extends gradually as more operations are involved. It shows that the search dives quickly to near-optimal solutions at the early stage. For more complex problems No. 15, 21, and 24, GCP keeps exploring better solutions.

4.2 Extreme test

In this experiment, the *FailLimit* parameter is removed. The program is terminated manually if the runtime exceeds 1000 seconds. The best makespans found by GCP are summarized in Table 6, compared with the best makespans found by listed algorithms. IGA outperforms GCP in solving

problems No. 5, 8, and 12. However, GCP performs better in solving more complex problems No. 17, 18, 20, 22, 23, and 24. Besides, it finds optimal solutions for further two problems No. 23 and 24. For problem No. 11, GCP finds the optimal solution in seconds, but it keeps running till it is manually terminated. It happens in solving the other 12 problems. Setting proper *FailLimit* parameter is helpful to deal with this issue. An optimal schedule for problem No. 24 is depicted in Fig. 7.

Table 6. Comparison of best makespans

Problem	Lower bound	SEA	IGA	OCGA	E-ACO	GCP	Best	Improved rate (%)	Runtime of GCP (s)
1	427	428	**427***	**427***	**427***	**427***	–	–	0.25
2	343	**343***	**343***	**343***	**343***	**343***	–	–	>1000†
3	344	347	**344***	**344***	**344***	**344***	–	–	>1000
4	306	**306***	**306***	**306***	**306***	**306***	–	–	>1000
5	304	319	**304***	318	318	318	IGA	-4.61	>1000
6	427	438	**427***	**427***	**427***	**427***	–	–	0.96
7	372	**372***	**372***	**372***	**372***	**372***	–	–	>1000
8	342	343	**342***	343	343	343	IGA	-0.29	>1000
9	427	428	**427***	**427***	**427***	**427***	–	–	0.8
10	427	443	**427***	**427***	**427***	**427***	–	–	0.56
11	344	369	368	348	348	**344***	GCP	1.15	>1000
12	306	328	**312**	318	322	318	IGA	-1.92	>1000
13	427	452	429	**427***	**427***	**427***	–	–	0.95
14	372	381	386	**372***	373	**372***	–	–	>1000
15	427	434	**427***	**427***	**427***	**427***	–	–	1.45
16	427	454	433	**427***	429	**427***	–	–	1.69
17	344	431	415	370	377	**344***	GCP	7.03	>1000
18	306	379	364	351	357	**344**	GCP	1.99	>1000
19	427	490	450	**427***	431	**427***	–	–	1.82
20	372	447	429	384	386	**372***	GCP	3.13	>1000
21	427	477	433	**427***	428	**427***	–	–	1.53
22	427	534	491	446	444	**427***	GCP	3.83	3.81
23	372	498	465	394	413	**372***	GCP	5.58	>1000
24	427	587	532	458	460	**427***	GCP	6.77	22.93

*: Optimal solution.

†: The runtime is longer than 1000 seconds.

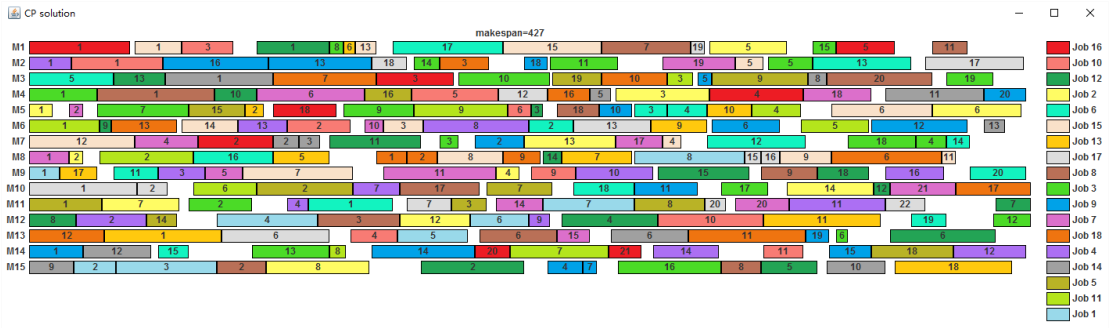

Fig. 7. An optimal schedule for problem No. 24 with makespan=427

5. Conclusions and discussions

This paper proposes a graph-based constraint programming approach for type-2 IPPS problem that takes AND/OR graphs as input in a jobshop manufacturing environment. Three types of flexibility, namely, processing flexibility, operation flexibility, and sequencing flexibility, are considered.

Based on the interval variable and scheduling-oriented constraints built in the IBM ILOG CP Optimizer, a concise model is formulated. Using type-parameters for operations is suggested to simplify implementation. The experiments on a set of benchmark problems show that the CP Optimizer can solve the graph-based CP model efficiently with embedded search algorithms.

It is promising to implement this model in industries based on the model-and-run CP optimizer. It is suggested to further develop the model to cater to process planning and scheduling problems in other circumstances. For instance, the assumption of internal logistic and setup consumption can be relaxed. Moreover, the non-preemption assumption can be removed to cope with a more general circumstance, for which the *cumul functions* and relevant constraints can be used to restrict resource usage. Providing a tighter lower bound is suggested to improve current model further. The development of corresponding solution approaches is suggested to cope with problems of larger scales. Furthermore, it is promising to incorporate the agent-based framework to cater to environmental dynamics.

Acknowledgment

The authors would like to acknowledge the financial support of the National Natural Science Foundation of China (No. 71501187).

Reference

[1] Leung CW, Wong T, Mak K-L, Fung RY. Integrated process planning and scheduling by an agent-based ant colony optimization. Computers & Industrial Engineering. 2010;59(1):166-80.

[2] CHEN Q, KHOSHNEVIS B. Scheduling with flexible process plans. Production Planning & Control. 1993;4(4):333-43.

[3] Kempenaers J, Pinte J, Detand J, Kruth J-P. A collaborative process planning and scheduling system. Advances in Engineering Software. 1996;25(1):3-8.

[4] Kim YK, Park K, Ko J. A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling. Computers & Operations Research. 2003;30(8):1151-71.

[5] Barzanji R, Naderi B, Begen MA. Decomposition algorithms for the integrated process planning and scheduling problem. Omega. 2020;93:102025.

[6] Sotskov YN, Shakhlevich NV. NP-hardness of shop-scheduling problems with three jobs. Discrete Applied Mathematics. 1995;59(3):237-66.
[7] Rossi F, Van Beek P, Walsh T. Handbook of constraint programming: Elsevier; 2006.
[8] Laborie P, Rogerie J, Shaw P, Vilím P. IBM ILOG CP optimizer for scheduling. Constraints. 2018;23(2):210-50.
[9] Crowston WB. Decision CPM: Network Reduction and Solution. Journal of the Operational Research Society. 1970;21(4):435-52.
[10] Mello LSHd, Sanderson AC. AND/OR graph representation of assembly plans. IEEE Transactions on Robotics and Automation. 1990;6(2):188-99.
[11] Gillies DW, Liu JW-S. Scheduling tasks with AND/OR precedence constraints. SIAM Journal on Computing. 1995;24(4):797-810.
[12] Beck JC, Fox MS. Scheduling alternative activities. AAAI/IAAI: Citeseer; 1999. p. 680-7.
[13] Barták R, Cepek O. Temporal Networks with Alternatives: Complexity and Model. FLAIRS Conference2007. p. 641-6.
[14] Moffitt MD, Peintner B, Pollack ME. Augmenting disjunctive temporal problems with finite-domain constraints. Proceedings of the National Conference on Artificial Intelligence: Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999; 2005. p. 1187.
[15] Tao S, Dong ZS. Multi-mode resource-constrained project scheduling problem with alternative project structures. Computers & Industrial Engineering. 2018;125:333-47.
[16] Li X, Gao L, Shao X, Zhang C, Wang C. Mathematical modeling and evolutionary algorithm-based approach for integrated process planning and scheduling. Computers & Operations Research. 2010;37(4):656-67.
[17] Lihong Q, Shengping L. An improved genetic algorithm for integrated process planning and scheduling. The International Journal of Advanced Manufacturing Technology. 2012;58(5-8):727-40.
[18] Jin L, Zhang C, Shao X, Tian G. Mathematical modeling and a memetic algorithm for the integration of process planning and scheduling considering uncertain processing times. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture. 2016;230(7):1272-83.
[19] Lee H, Kim S-S. Integration of process planning and scheduling using simulation based genetic algorithms. The International Journal of Advanced Manufacturing Technology. 2001;18(8):586-90.
[20] Shao X, Li X, Gao L, Zhang C. Integration of process planning and scheduling—a modified genetic algorithm-based approach. Computers & Operations Research. 2009;36(6):2082-96.
[21] Zhang L, Wong TN. An object-coding genetic algorithm for integrated process planning and scheduling. European Journal of Operational Research. 2015;244(2):434-44.
[22] Li X, Gao L, Pan Q, Wan L, Chao K-M. An effective hybrid genetic algorithm and variable neighborhood search for integrated process planning and scheduling in a packaging machine workshop. IEEE Transactions on Systems, Man, and Cybernetics: Systems. 2018;49(10):1933-45.
[23] Luo G, Wen X, Li H, Ming W, Xie G. An effective multi-objective genetic algorithm based on immune principle and external archive for multi-objective integrated process planning and scheduling. The International Journal of Advanced Manufacturing Technology. 2017;91(9-12):3145-58.
[24] Li W, McMahon CA. A simulated annealing-based optimization approach for integrated process planning and scheduling. International Journal of Computer Integrated Manufacturing. 2007;20(1):80-95.
[25] Guo Y, Li WD, Mileham AR, Owen GW. Applications of particle swarm optimisation in integrated process planning and scheduling. Robotics and Computer-Integrated Manufacturing. 2009;25(2):280-8.

[26] Wong TN, Leung CW, Mak KL, Fung RYK. Dynamic shopfloor scheduling in multi-agent manufacturing systems. Expert Systems with Applications. 2006;31(3):486-94.
[27] Wong T, Leung C, Mak K, Fung R. Integrated process planning and scheduling/rescheduling—an agent-based approach. International Journal of Production Research. 2006;44(18-19):3627-55.
[28] Lin C-S, Li P-Y, Wei J-M, Wu M-C. Integration of process planning and scheduling for distributed flexible job shops. Computers & Operations Research. 2020;124:105053.
[29] Baptiste P, Le Pape C, Nuijten W. Constraint-based scheduling: applying constraint programming to scheduling problems: Springer Science & Business Media; 2012.
[30] Harjunkoski I, Grossmann IE. Decomposition techniques for multistage scheduling problems using mixed-integer and constraint programming methods. Computers & Chemical Engineering. 2002;26(11):1533-52.
[31] Sadykov R, Wolsey LA. Integer programming and constraint programming in solving a multimachine assignment scheduling problem with deadlines and release dates. INFORMS Journal on Computing. 2006;18(2):209-17.
[32] Beck JC, Feng TK, Watson JP. Combining Constraint Programming and Local Search for Job-Shop Scheduling. Informs Journal on Computing. 2011;23(1):1-14.
[33] Bartak R, Salido MA, Rossi F. Constraint satisfaction techniques in planning and scheduling. Journal of Intelligent Manufacturing. 2010;21(1):5-15.
[34] Timpe C. Solving planning and scheduling problems with combined integer and constraint programming. Or Spectrum. 2002;24(4):431-48.
[35] Ham AM, Cakici E. Flexible job shop scheduling problem with parallel batch processing machines: MIP and CP approaches. Computers & Industrial Engineering. 2016;102:160-5.
[36] Zhang S, Wang S. Flexible assembly job-shop scheduling with sequence-dependent setup times and part sharing in a dynamic environment: Constraint programming model, mixed-integer programming model, and dispatching rules. IEEE Transactions on Engineering Management. 2018;65(3):487-504.
[37] Laborie P. An update on the comparison of MIP, CP and hybrid approaches for mixed resource allocation and scheduling. International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research: Springer; 2018. p. 403-11.
[38] Kreter S, Schutt A, Stuckey PJ. Using constraint programming for solving RCPSP/max-cal. Constraints. 2017;22(3):432-62.
[39] Kim YK, Park K, Ko J. A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling. Comput Oper Res. 2003;30(8):1151-71.
[40] Laborie P. IBM ILOG CP Optimizer for detailed scheduling illustrated on three problems. International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems: Springer; 2009. p. 148-62.
[41] Zhang S, Wong TN. Integrated process planning and scheduling: an enhanced ant colony optimization heuristic with parameter tuning. Journal of Intelligent Manufacturing. 2018;29(3):585-601.

Response to Reviewers' Comments

For resubmission of the manuscript entitled

A graph-based constraint programming approach for the integrated process planning and scheduling problem

Ref. No.: COR-D-19-00455R1

We would like to sincerely thank Professor Francisco Saldanha da Gama, the Editor-in-Chief, for giving us a further opportunity to revise and resubmit our manuscript entitled *A graph-based constraint programming approach for the integrated process planning and scheduling problem* (COR-D-19-00455R1), as part of a major revision. We also appreciate the review team who have offered invaluable guidance for developing not only the paper but also the authors' academic professionalism. At the same time, we are very sorry for the grammar errors as our focus was mainly on the logic and contents. We agree with the well-founded criticisms. The main changes include but not restricted to:

1) Section 3 has been restructured, following the guidance of the review team.
2) The applied CP model (referred to as the CP manufacturing model by the review team) has been removed. The suggestion of using type-parameters to identify operation types is stated with several sentences.
3) Many redundant descriptions have been removed. Regarding last version, we attempted to enhance the readability. But we noticed such redundancy is unnecessary and may cause troubles to readers. Thanks for the reviewers' comments.
4) We have improved the language, and tried our best to fix grammar errors.

In the following, we describe in detail how we have addressed each comment. For ease of review, we address each comment below in the order that it was provided. The reviewers' comments use the font-style Calibri Light, and responses use the Times New Roman. We believe we have accommodated all review comments. Should there however be any additional comments, we will be happy to accommodate them.

Reviewers' comments:

Editor: The presentation is far from acceptable. A major revision is needed.

Reviewer #1:

Overall, it can be stated that the authors really improved their paper. However, there are still several problems in the work, which have to be solved before being in a mature enough state to be published:

1. General:

The biggest (most important) issue is the segmentation and arrangement of the contents of Section 3, i.e. the segmentation and arrangement of the CP model and its related notations and explanations and there are several parts where the same/similar issues appear).

Answer:

Thanks for the comments. We have rewritten Section 3. In current version, it is in the following

form:

3. The GCP model

The description of a general IPPS problem.

3.1 The Type-2 IPPS problem and AND/OR graph representation

The two parts and features to be manufactured.

3.1.1 The AND/OR graph for Type-2 IPPS problem

The AND/OR graph that corresponds to the illustrative IPPS case

The three types of flexibility

3.1.2 General definition and augmentation of the AND/OR graph

A general definition of the AND/OR graph for the IPPS problem

3.1.3 Assumptions

Five problem-oriented assumptions inherited from existing literature, and three graph-oriented assumptions for valid representation and modelling of the IPPS problem

3.2 Variables and notions

Two types of time interval variables used by the model

Table 1. Notations

3.3 The GCP model

3.4 An illustrative case

The projection from the AND/OR graph to constraints of the GCP model, using the illustrative IPPS instance

The machine configuration

Table 2. Processing data for the illustrative case

Table 3. Graph-oriented data for the illustrative IPPS case

The suggestion of type-parameters for implementation

An optimal solution and discussions

We hope the new structure is proper to illustrate the proposed model.

There are several other smaller issues, which can be found below.

2. Moreover: Although many spelling mistakes have been corrected, I am afraid, there are still many mistakes appearing in all parts of the work: they have to be corrected 100% in order to satisfy the requirements for publication in a high standard academic journal. E.g., missing article "the" or inserted the article "the" in places where it is not necessary, missing "a" or "an" in many cases, 3rd person s mistakes, sometimes still appearing capital letter within sentences etc. etc. Here are only some examples out of the beginning of the paper, several others are not listed:
"It selects one process plan for each job and allocate machining…" -> allocateS instead of allocate

Answer:

We are really sorry for the spelling mistakes and misuse of articles. There is no excuse. The authors have carefully revised the manuscript and used some proof reading. We have referred to several works published in reputable journals and carefully revised the hundreds of articles. With regard to errors like “allocate”, we have carefully checked such errors.

The whole sentence “"It selects one process plan for each job and allocate machining…" has been rewritten as “A set of predetermined alternative process plans is fed to the IPPS system

which then selects a process plan for each job and allocates machining resources simultaneously."

3. „…predetermined process plans, the IPPS problems fall into…" -> problem instead of problems and falls instead of fall

Answer:
Since we noticed that there are several frameworks for the integration of processing planning and scheduling, we used "problems" instead of "problem" to imply several problems of slight differences in the IPPS domain. Thanks to the reviewer's comments, we noticed it may cause confusion to the readers. In current manuscript we rewrote it as "*Due to the differences in organizing the predetermined process plans, IPPS falls into two categories.*"

4. "It is known that traditional jobshop" -> THE traditional…
Answer:
We noticed "It is known that" conveys no information. We improved it to be "*Since the traditional jobshop scheduling problem …*" with the definite article properly placed. Thanks.

5. Abstract:
„Integration of process planning and scheduling is to carry out the two functions simultaneously": please change „the two" -> „both"

„… integrated process planning and scheduling (IPPS) problem in the…" -> please delete „the" and add „a" instead
Answer:
We have improved the entire abstract. First, we have replaced "*the two*" with "*both*" as suggested by the reviewer. Secondly, we have removed "in the jobshop environment" since we attempt to put the most important things in the abstract. However, we have updated the use of article for the "jobshop" in the manuscript.

6. Introduction:
(6.1) „At first, a set of alternative process plans are generated" -> IS generated and not are (a set is generated)
Answer:
We are sorry for the misuse of plural verbs. We have corrected this error. Besides, we rewrote the explanation about processing planning and scheduling.

(6.2) "The sequential process planning and scheduling may ruin the global optimality of the planning work," Perhaps you could find a smoother word for „ruin"
Answer:
We replaced the whole sentence with a more straightforward explanation, "*The process planning and scheduling modules usually maintain different sets of environmental factors and are performed to achieve different goals, which frequently raises conflicts. For instance, the process planning module cannot sense the real-time status in workshops, which may result in*

assigning unavailable machines to some operations. Besides, separately conducting the process planning and scheduling functions cannot quickly respond to changes in a dynamic manufacturing environment [1]."

(6.3) „Recently, two exact approaches, a mathematical modelling and a memetic algorithm [11], and a decomposition algorithm [5]…": a memetic algorithm is not an exact approach.
I think you have written here two paragraphs about already existing solution approaches in order to motivate the usage of CP, i.e. the CP Optimizer. However, as you also describe solution approaches in Section 2, this is a redundancy which should be avoided: Please take these two parts out of here and insert them into the literature review in Section 2 - instead, include here only two or three sentences by briefly motivating that (1) the problem is NP-hard, (2) several heuristic and exact approaches have already been investigated but there is still a strong need for research in the area of IPPS Type-2 and that (3) CP is a promising solution approach according to several existing works -> therefore, this work introduces a CP solution approach for the IPPS Type-2.

Answer:

Thanks very much for the reviewer's careful and thoughtful comments.

First, we are sorry that we made the incorrect description about the memetic algorithm. We have removed the redundancy and moved the review of solution approaches to Section 2. In the introduction, we followed the reviewer's guidance and just briefly introduced the motivation of this research. Thanks a lot for the comment which provides valuable guidance to develop our academic professionalism.

(6.4) "…provides conclusions and discussions": provides a discussion and a conclusion.

Answer:

We are sorry that we get confused. We have checked several academic articles and noticed the use of singular or plural nouns for "conclusion" and "discussion" does not coincide. However, we totally respect the expertise of the reviewer's comment. We used "*Concluding remarks and suggestions for future work are presented in Section 5*" instead.

7. Related work

(7.1) "Barzanji, Naderi [5] solved"…: the description of their work does not match what they really did in their work. As far as I see, the authors mixed the contents of Barzanji & Naderi on the one hand and Tao & Dong on the other hand.

Answer:

We are sorry we mixed the two works. We have rewritten this paragraph. This improved paragraph serves the review of mathematical models for the IPPS problem. The article Barzanji, Naderi [5] provides the discussion about two types of IPPS problem and a mathematical model for type-1 IPPS problem. Tao and Dong [15] considered an alternative project structure for the multi-mode resource-constrained project scheduling problem. This work is placed together with other works for reviewing related problem domain.

(7.2) "An effective multi-objective GA was also proposed for the IPPS problem with multi-objectives [26]": please delete „multi-objectives"

Answer:
We are sorry for the repetition. We have rewritten this sentence as "*An effective multi-objective GA can be found [23]*".

(7.3) "Some latest literature reported methods that combine CP and local search to increase the efficiency in solving complex…": Please reformulate the whole sentence, this is no proper English.

Answer:
We have reformulated this sentence. The current form is "*Besides, some researchers combined CP and local search approaches. For instance, Beck, Feng [32] combined CP and tabu search to solve JSPs.*".

(7.4) There is no "taboo" search. It is "tabu" search (consisting of a tabu list)

Answer:
We are sorry for the improper word. We found 8,120 records for "taboo search" and 178,000 records for "tabu search" on Google Scholar. We thought the "taboo search" was for British English and "tabu" was for American English. Obviously, we made a mistake. We have changed the term to "tabu". Thanks for the comment.

(7.5) "The integration of planning and scheduling was even considered" - the „even" here does not make much sense.

Answer:
Thanks for the comment. We have rewritten this sentence as "*The integration of planning and scheduling was discussed*".

(7.6) "Repeatedly researchers reported the outstanding performance of CP in solving scheduling problems.": If you want to include this sentence, you have to insert references. Instead, you could reformulate it, e.g. „As a result, it can be stated that several works reported the outstanding performance…"

Answer:
We respect the reviewer's preciseness. The sentence "several works reported the outstanding performance" is borrowed to conclude the literature review. Many thanks.

(7.7) Please reformulate the following paragraph: "As a powerful problem-solving and optimization tool, the CP has great potential to solve the IPPS problem. This paper will report a concise graph-based CP model with global constraints built in IBM ILOG CP Optimizer for the Type-2 IPPS problem." e.g. by: As a powerful problem-solving and optimization tool, CP has great potential to solve the IPPS problem. Therefore, in the following, a concise graph-based CP model with global constraints built in IBM ILOG CP Optimizer for the Type-2 IPPS problem is presented.

Answer:
We have rewritten this concluding paragraph, which is currently in the following form:

"To conclude, the IPPS problem that integrates process selection and JSPs is an important scheduling problem. Type-2 IPPS problem packs alternative process plans in AND/OR graphs, which provides a higher level of flexibility to modelling and solution approaches. Several works reported the outstanding performance of CP in solving scheduling problems. An approach that combines CP and AND/OR graph to solve type-2 IPPS problem is proposed. A concise GCP model based on the CP optimizer is provided."

8. The graph-based CP model for IPPS

This section has already improved in a very good way! Nevertheless, there are further content-wise issues which have to be improved:

(8.1) Fig. 2 F1.5 Milling: Is this "milling" intentional? In the description for this figure, it is called "machining" (and not milling)

Answer:

We have checked the term on Wikipedia (https://en.wikipedia.org/wiki/Milling_(machining)), which defines the milling (machining) as "a cutting process that uses a milling cutter to remove material from the surface of a work piece". It says that "It is one of the most commonly used processes for machining custom parts to precise tolerances."

Besides, we have checked several academic papers, such as the J. Tlusty (1986), Insperger et al. (2003), and Altintaş et al. (1995), which used the term "milling". We also checked the term on the websites of some machine providers. As a result, we kept this term.

Reference

TLUSTY, J. (1986). Dynamics of High-Speed Milling. Trans. ASME, J. Eng. Ind., 108, 59.

Insperger, T., Mann, B. P., Stépán, G., & Bayly, P. V. (2003). Stability of up-milling and down-milling, part 1: alternative analytical methods. International journal of Machine tools and manufacture, 43(1), 25-34.

Altintaş, Y., & Budak, E. (1995). Analytical prediction of stability lobes in milling. CIRP annals, 44(1), 357-362.

(8.2) "Besides, for processing the bolt, either operation….": After introducing the left part of the figure very well, now you start by talking about operation 2,3 - the description for the preceding operations is missing and has to be added.

Answer:

We are sorry we didn't complete the description of job 2. The following description for the missing part has been added.

"Besides, the surfaces F2.3, outer diameter F2.2, chamfer F2.4, and relief groove F2.5 of the bolt have to be finished before threading F2.6. A lathe can finish features F2.2–F2.5 with the single operation $o_{2,1}$. *Operation* $o_{2,2}$ *for turning the thread follows. After that, either operation* $o_{2,3}$ *for cutting surface F2.1 or* $o_{2,4}$ *for cutting the slot can be started.".*

(8.3) "The IPPS discussed in this paper follows the general assumptions made for this category of problem." Please add „…, which are explained in the following:"

Answer:
We have put assumptions in a separate section 3.1.3. There are two categories of assumptions, the problem-oriented assumptions and graph-oriented assumptions. Thanks very much for the reviewer's comment. We have added the "in the following" in the sentence and made a bit more modification. This sentence in current form is
"Problem-oriented assumptions P1–P5 as given in the following are inherited from literature [4]. Besides, to make the AND/OR graph effective for the IPPS representation, three more assumptions G1–G3 are exerted."

(8.4) „…the graph-based CP model is much simpler and easier to be implemented in industries.": please additionally explain why it is simpler and easier by giving an example/explanation.
Answer:
We are sorry for the imprecise assertation. We noticed it is difficult to compare the simplicity. As a result, we removed this assertation. In the literature review, we argued that "*The AND/OR graph can establish a compact structure on a set of process plans for the same job, which provides a possibility to partly deal with process plans*", which we hope is more precise.

(8.5) Although the whole model formulation is now clearer, it still has several problems:
„Fig. 3 as an example, there are two or-links": this paragraph starts by introducing the operation numbers of the figure. However, in the directly following sentences, they are not used any longer. Please use them in the whole paragraph or do not use them in the whole paragraph.
Answer:
We are sorry we mixed general discussion and case-based explanation. We have rewritten this part. Currently only the case-based explanation is provided, since we think it should be more straightforward.

(8.6) „…on the right branch by any paths without passing the or-initiator o_1,11 or the or-terminator
O_1,12": where can I find o_1,11 and o_1,12?
„Or say, there will be an or-link defined…": This whole sentence can be omitted.
Table 1: "Indices" instead of "indexes"
Answer:
We are sorry for the unclear expression. This whole paragraph together with the assumptions of AND/OR graph has been rewritten. The or-initiator $o_{1,11}$ and or-terminator $o_{1,12}$ are in green in Fig. 3. We added "as shown in Fig. 3" to the place where $o_{1,11}$ and $o_{1,12}$ appear for the first time. The whole sentence "Or say, there will be an or-link defined…" has been removed. The incorrect word "indexes" in the manuscript has been corrected.

(8.7) Tables and Models: It is not standard to introduce several notation tables and several models for the very same problem, i.e. the very same CP problem model. In order to comply with the academic standard, this issue has to be fixed: 1 notation table and 1 CP model. Therefore, instead of having several parts in Section 3, partly repeating the same content and

switching between CP and the model/problem itself, clearly separate and re-write: you need one subsection for the problem description (IPPS), one subsection for the CP relevant parts (decision variable, functions etc.: please do not too much into detail since in the works of Laborie 2009 and Laborie et al 2018 the very same contents are described - it is enough to briefly explain the used expressions), one subsection including the CP model.

Answer:

We are sorry for the unprofessional writing style.

a) We have merged the notation tables. In current version, all notations are listed in Table 1.

b) We have reorganized the section for the GCP model (Section 3 in current form). Three subsections are provided: Section 3.1 severs the description of the IPPS problem and the AND/OR graph representation with a simple case. Assumptions are listed at the end of this section as well; Section 3.2 introduces decision variables and notations; Section 3.3 exhibits the GCP model; Section 3.4 gives an illustrative case.

c) The general description of CP Optimiser constraints is removed, since we think the illustrative case is enough for the explanation.

Thanks very much for the reviewer's suggestions.

(8.8) Decision variables: The function presenceOf is not a decision variable. It is a function that gives the decision variable a direction - there is a huge difference between these two notifications, which has to be clearly described.

Answer:

We are sorry that we misunderstood and mis-explained the function *presenceOf*. We have carefully reviewed several articles to make sure we get better understanding about this function. The error has been fixed. In current form, the description of *presenceOf* is given in Table 1, together with time intervals. The current description is "The function associated to the presence status of interval I".

(8.9) Since you use CP relevant functions: be aware of how to correctly write them (have a look at publications using these functions and how they format/write them).

Answer:

Thanks very much for the comment. We have carefully checked several pieces of articles reporting CP approaches for some scheduling problems, including Lunardi et al. (2020), Laborie (2009), and Delgado et al. (2012). We noticed there are slight differences when authors wrote CP constraints. For instance, Lunardi et al. (2020) fed the alternative constraint with an array as the second argument, while Laborie (2009) fed a set. We have learnt a lot from these articles and improved the way we wrote CP constraints. Many thanks.

Reference

Lunardi, W. T., Birgin, E. G., Laborie, P., Ronconi, D. P., & Voos, H. (2020). Mixed integer linear programming and constraint programming models for the online printing shop scheduling problem. Computers & Operations Research, 123, 105020.

Laborie, P. (2009, May). IBM ILOG CP Optimizer for detailed scheduling illustrated on three problems. In International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems (pp. 148-162). Springer, Berlin,

Heidelberg.

Delgado, A., Jensen, R. M., Janstrup, K., Rose, T. H., & Andersen, K. H. (2012). A constraint programming model for fast optimal stowage of container vessel bays. European Journal of Operational Research, 220(1), 251-261.

(8.10) Moreover, the enumeration after "Built-in constraints in CP Optimizer…." and the enumeration on the following page after "Fig. 3 illustrates" contains - in parts- redundant aspects. Only have one enumeration, including the content of both parts.

Answer:

We are sorry for redundant enumerations. We have removed several redundant enumerations "Built-in constraints in CP Optimizer….". Besides, we have taken care of other redundancy in this article.

(8.11) The explanation of the constraints: If you introduce one constraint, e.g. Constraint (11), then include the whole content of this constraint and do not again refer to the same constraint on the next page and explain an additional fact for it.

Answer:

Thanks very much for the comment. We have rewritten the explanation of constraints which is currently at the end of Section 3.3, following the guidance of the reviewer. Thanks a lot.

(8.11) Why is there no objective function introduced for the generic CP model but only for the manufacturing case? The authors refer to the objective function in the results section.

Answer:

Because we were criticised by some other reviewers for putting an objective function in a CP model in another work, we didn't put the objective function in the GCP model. However, we have added the objective function back to the GCP model since we now believe it is an academic standard, regarding the articles published in reputable journals we checked and our respect for the professional reviewer.

(8.12) It is also very questionable if it is necessary to insert the CP manufacturing model - please better justify the reason why this is necessary as for the current status, I do not see why it is.

Answer:

Our intension was to provide suggestions to the readers who would like to replicate the GCP approach. The data structure is complicated once or-relationships are involved. We suffered a lot in maintaining different categories of nodes. The definition of type-parameters can substantially simplify the programming. As a result, we inserted the CP manufacturing model. Thanks to the comment, we noticed that it may raise confusion. As a result, we have removed the CP manufacturing model. Instead, we inserted several sentences to suggest type-parameters for implementation. Many thanks.

(8.13) The text after the manufacturing CP model is not comprehensible: "The constraints listed in the Applied Graph-Based CP Model correspond to the constraints (6) -(13)." - Yes, absolutely

true, as listed on one of the previous pages, constraints (6)-(13) belong to the generic CP model - you have to give a reason and explain why you write that here.

Answer:

We referred to several works which include the expression like "constraints (6) -(13)" in a model if there is a previous model that has the same set of constraints. We guessed it would be for simplification and completeness purpose. However, we respect the reviewer's comments, and have removed the manufacturing model.

(8. 14) The section number 3.2.2. is missing!

Answer:

We are sorry for the carelessness. The error has been corrected, and we have carefully checked section numbers.

9. Experiments

(9.1) "...is used to formulate the mode": formulate and run the model

Answer:

We have improved this sentence according to the reviewer's suggestion.

(9.2) Table 6: format is not fully correct

Answer:

We tried to follow the reviewer's previous suggestions. Best makespans are in bold. Optimal makespans are asterisked and in bold since they are also the best for corresponding problems. If there is some compared algorithm generated the best makespan together with GCP, the "Best" indicator and "Improvement rate" are dashes. We attempted to highlight the key data to the readers as clearly as possible. The font size of Table 5 and 6 is slightly smaller since we want to avoid multiple lines for each problem. We would like to further improve the format of these two tables if necessary.

(9.2) "For the other 6 problems, both the GCP and one compared algorithm reach the optimal solutions": For further 6 problems, both …. . Then, it additionally has to be described that for 1 instance, the CP Optimizer is not as good as the other solution methods (this point is missing now).

Answer:

We are sorry for the missing description. We didn't purposely avoid the bad solution. We have added the description of this result: "*For problem No. 5, IGA outperforms GCP by 2.65%*". Besides, we added the description "*IGA outperforms GCP in solving problems No. 5, 8, and 12*" to the illustration of results for the extreme test.

(9.3) The text description for Table 7 does not comply with the results presented in Table 7. Please refine this part! (e.g. "GCP slightly improves the makespans for problem 5, 12, 17, 18, 23 and 24": this is not true, an improvement is made by other algorithms or it has already been reached in Table 6 - The same holds for the following sentences.)

Answer:

We are really sorry for the vital mistake. Thanks very much. We have rewritten the description of results for the extreme test. Besides, we have carefully checked all data and descriptions.

Reviewer #2: The authors improved the manuscript a lot since their original version and I think the current version should be accepted for publication.
I have mainly some minor comments that would help improving the current version of the submission.

1. * First, I think that the authors should better highlight the fact that the CP approach proposed in the paper is an *exact algorithm* that, beside producing good solutions, also provides lower bounds and optimality proofs. This is something that the meta-heuristic approaches are not able to do.

Answer:

Thanks very much for the comment which is quite important for a better understanding of the graph-based CP approach in solving type-2 IPPS problem. We have highlighted that "*Constraint programming (CP) is a powerful exact algorithm for combinatorial optimization. CP uses declarative expressions to formulate constraints and adopts separate technologies to generate feasible solutions, provide lower bounds, and do optimality proofs*" in the introduction. We are sorry we are not able to properly place the citation for the anonymous reviewer's sentence. We found evidences from the Rossi et al. (2006), so we placed that citation for this description. We would like to thank the reviewer here with the greatest respect.

Reference

Rossi F, Van Beek P, Walsh T. Handbook of constraint programming: Elsevier; 2006.

2. * In section 4, you say that "The CP Optimizer uses the newest iterative diving search method to perform aggressive dives in the search tree without backtrack [4]. Other settings are default." Did you really change some settings to only use this new Iterative Diving search (typically SearchType="IterativeDiving") ? Or did you use the default settings that use this new type of search as "one" of the ingredients of the search ?
Also, I think that here, citation [4] is wrong.

Answer:

We are sorry we didn't properly describe this part. We tried both approaches: use default settings and manually set SearchType="IterativeDiving". We noticed that it performed better in case of using default setting. We carefully checked engine logs which reported the use of IterativeDiving for all problems. To remove the ambiguity, we have rewritten this part. In current form, it is "*All settings are default. The CP Optimizer automatically implements the newest iterative diving search method for all problems and performs aggressive dives in search tree without backtrack*". We have removed the improper citation.

3. * In the model formulation, I think that it would be very interesting to try a variant of the current formulation that introduces 'alternative' constraints for the 'OR' nodes (both initiators and terminators). It would lead to stronger constraint propagation in the engine (because it would consider the conjunction of the startBeforeEnd (constraints 6) and constraints on the

presence (constraints 10) instead of considerating them separately) and would probably improve the performance and the capability to provide optimality proofs.
Concretely, suppose an OR node a -> OR {b,c}, it would mean creating two additional optional interval variables a_b and a_c, post an alternative(a, {a_b,a_c}) and post precedences and presence constraints on these intervals: endBeforeStart(a_b,b); presenceOf(a_b)==presenceOf(b); endBeforeStart(a_c,c); presenceOf(a_c)==presenceOf(c). A similar formulation can be adopted for OR-terminator nodes. With this model for instance, the naive lower bound consisting of the shortest operation durations and the shortest length of "OR" sub-processes would be naturally found by constraint propagation at the root node. Which is probably not the case with the current formulation.

Answer:

Thank the reviewer very much for the great idea. We tried the GCP model that incorporated the suggested constraints for or-initiators and or-terminators (thereafter referred to as GCP-2). Several other constraints have to be updated. The GCP-2 model is in the following.

The GCP-2 Model

$$Minimize\ c_{\max} = \max\left\{end\left(I\left(e_j\right)\right) \mid j \in J\right\} \tag{2-1}$$

subject to:

$$alternative\left(I\left(o_{j,u}\right), \left\{I\left(o_{j,u},k\right) \mid k \in M\left(o_{j,u}\right)\right\}\right), \forall o_{j,u} \in O \setminus \left(R^s \cup R^e\right) \tag{2-2}$$

$$nooverlap\left(Q_j^J\right), \forall j \in J \tag{2-3}$$

$$nooverlap\left(Q_k^M\right), \forall k \in M \tag{2-4}$$

$$endbeforestart\left(I\left(o_{j,u}\right), I\left(o_{j,v}\right)\right), \forall o_{j,u} \to o_{j,v} \in A, o_{j,u} \notin R^s, o_{j,v} \notin R^e \tag{2-5}$$

$$presenceOf\left(I\left(s_j\right)\right) = 1, \forall j \in J \tag{2-6}$$

$$presenceOf\left(I\left(o_{j,u}\right)\right) = presenceOf\left(I\left(o_{j,v}\right)\right), \forall o_{j,u} \to o_{j,v} \in A, o_{j,u} \notin R^s, o_{j,v} \notin R^e \tag{2-7}$$

$$alternative\left(I\left(o_{j,u}\right), \left\{I\left(o_{j,v}\right) \mid o_{j,u} \to o_{j,v} \in A\right\}\right), \forall o_{j,u} \in R^s \tag{2-8}$$

$$alternative\left(I\left(o_{j,v}\right), \left\{I\left(o_{j,u}\right) \mid o_{j,u} \to o_{j,v} \in A\right\}\right), \forall o_{j,v} \in R^e \tag{2-9}$$

$$Q_j^J = \left\{I\left(o_{j,u}\right) \mid o_{j,u} \in O_j \setminus \left(R^s \cup R^e\right)\right\} \tag{2-10}$$

$$Q_k^M = \left\{I\left(o_{j,u},k_0\right) \mid \left(o_{j,u},k_0\right) \in P, o_{j,u} \notin R^s \cup R^e, k_0 = k\right\} \tag{2-11}$$

$$Interval\ I\left(o_{j,u}\right), optional,\ for\ all\ o_{j,u} \in O \tag{2-12}$$

$$Interval\ I\left(o_{j,u},k\right), optional, size = \tau\left(o_{j,u},k\right),\ for\ all\ \left(o_{j,u},k\right) \in P \tag{2-13}$$

The main changes include:

- Constraints (2-8) and (2-9) that synchronize an or-initiator or or-terminator with one operation in its connected or-link.
- Constraint (2-2) excludes intervals for or-initiators and or-terminators since it would cause conflicts with constraint (2-8) and (2-9) if the dummy machine is assigned to the two types of dummy nodes.
- Constraint (2-5) for precedence relationships excludes or-initiators and or-terminators, since the size of the two types of dummy nodes is no longer zero after synchronized with a regular node.
- The interval sequence Q_j^J for each job excludes or-initiators and or-terminators due to their non-zero sizes. Likewise, the interval sequence Q_k^M also excludes or-initiators and or-terminators.

We used the GCP-2 to solve the benchmark problems. Results are given in the following.

Table 1. Comparison between GCP and GCP-2

Problem	GCP			GCP-2		
	Makespan	Runtime (s)	Number of branches	Makespan	Runtime (s)	Number of branches
1	427	0.62	440,100	427	0.24	270,040
2	343	3.2	3,051,206	343	3.45	3,422,637
3	344	2.5	2,210,281	344	4.52	4,338,363
4	306	3.43	2,686,082	306	4.31	4,471,692
5	318	2.78	2,629,019	318	4.11	4,516,151
6	427	0.89	366,560	427	0.6	335,148
7	372	2.07	1,990,169	372	3.64	3,905,389
8	343	3.43	3,327,514	343	3.31	3,449,609
9	427	0.86	275,686	427	0.25	182,437
10	427	0.64	292,836	427	0.77	361,248
11	344	4.47	2,682,425	344	6.52	4,957,870
12	318	4.61	2,720,715	318	6.17	5,223,073
13	427	0.96	403,193	427	1.17	605,044
14	372	3.01	2,389,631	372	4.81	4,327,345
15	427	1.07	324,585	427	0.75	289,578
16	427	1.36	510,069	427	1.56	735,612
17	344	7.39	1,842,762	346	9.55	6,333,814
18	344	6.05	1,778,135	344	8.13	5,949,566
19	427	2.03	652,352	427	2.24	932,966
20	372	4.64	2,314,140	372	6.91	4,986,359
21	427	1.5	423,623	427	2.08	732,919
22	427	3.98	1,151,273	427	4.15	1,532,842
23	378	6.51	1,714,486	375	9.79	6,206,810
24	436	10.33	2,075,928	439	12.12	7,475,719

We found that in solving 16 problems (shaded in Table 1) the number of branches explored by GCP-2 is significantly greater than that by GCP. In solving other 4 problems, the number of branches explored by GCP-2 is slightly larger than that by GCP. Our explanation is that the four constraints (6)-(9) related to presence of each operation form a presence network. the presence relationships spread out along directed arcs. It is easy to reason the presences of all nodes using constraints (6)-(9) if the presence of some node is changed. However, this part should be investigated in future work.

The makespans and runtimes of the two models are similar. As a result, we would like to keep the GCP model in the manuscript. No doubt that the reviewer provided a promising idea and deep thinking, which will inspire our future work in this area.