Control Engineering Practice 21 (2013) 1204–1225

Contents lists available at SciVerse ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Benchmarking flexible job-shop scheduling and control systems

Damien Trentesaux [a,b], Cyrille Pach [a,b,*], Abdelghani Bekrar [a,b], Yves Sallez [a,b], Thierry Berger [a,b], Thérèse Bonte [a,b], Paulo Leitão [c,d], José Barbosa [a,b,c]

[a] University Lille Nord de France, F-59000 Lille, France
[b] UVHC, TEMPO Research Center, F-59313 Valenciennes, France
[c] Polytechnic Institute of Bragança, Campus Sta Apolonia, Apartado 1134, 5301-857 Bragança, Portugal
[d] LIACC—Artificial Intelligence and Computer Science Laboratory, R. Campo Alegre 102, 4169-007 Porto, Portugal

ARTICLE INFO

Article history:
Received 13 July 2012
Accepted 10 May 2013

Keywords:
Benchmarking
Flexible job-shop
Scheduling
Optimization
Simulation
Control

ABSTRACT

Benchmarking is comparing the output of different systems for a given set of input data in order to improve the system's performance. Faced with the lack of realistic and operational benchmarks that can be used for testing optimization methods and control systems in flexible systems, this paper proposes a benchmark system based on a real production cell. A three-step method is presented: data preparation, experimentation, and reporting. This benchmark allows the evaluation of static optimization performances using traditional operation research tools and the evaluation of control system's robustness faced with unexpected events.

1. Introduction

Research activities in manufacturing and production control are constantly growing, leading to an increasing variety of scheduling and control solutions, each of them with specific assumptions and possible advantages. Despite this, a very small number attain the stage of industrial implementation or even tests in real conditions for several reasons. One of these reasons is the difficulty to provide robust, reliable performance evaluation of the control systems proposed that would convince industrials to take the risk to implement it. A first step towards a robust, reliable performance evaluation was made several years ago by the operational research (OR) community, which has proposed several benchmarks allowing the algorithms that try to solve static NP-hard optimization problems for production (e.g., routing, scheduling) to be compared.

Benchmarking is comparing the output of different systems for a given set of input data in order to improve the system's performance. In the OR literature, several benchmarks are often cited and widely used: Taillard (1993), Beasley (1990), Reinelt (1991), Kolisch and Sprecher (1996), Demirkol, Mehta, and Uzsoy (1998) and Bixby, Ceria, McZeal, and Savelsbergh (1998). The advantages of all these benchmarks are well known: very large databases using instance generators, and/or updating mechanisms for the community for improved bounds or optimal solutions. The problems are related to traditional OR problems (e.g., traveling salesman) and formalized problems (e.g., MILP); a large number of these benchmarks deal with scheduling problems (e.g., Hybrid Flow Shop Scheduling, Job Shop Scheduling, Hoist Scheduling Problem, Resource Constrained Project Scheduling). Thus, these benchmarks are useful to evaluate the quality of a scheduling method with a structured set of data – with all the data being complete, exact and available at the initial date – with no use of feedback control. As a result, the data handled in these benchmarks are quantitative and static, which allows a clear comparison of performances, in terms of makespan or the number of late jobs, for example.

From a control point of view, these benchmarks respond to part of the problem: the design of a scheduling plan in a static environment sometimes with *a priori* robustness analysis of results. However, it does not allow the dynamic behavior to be evaluated from a control perspective (i.e., a control feedback approach), updating real-time decisions based on observations of real-time events and unstructured data. Furthermore, these OR benchmarks were designed mainly from a theoretical point of view, with little attention paid neither to several constraints imposed by the reality of production systems such as limited production/storage/transport capacity, maintenance/inventory/tool/spacing constraints nor to dynamic events or data such as breakdowns or urgent/canceled orders. Moreover, these benchmarks cannot be adapted to emerging control architectures (e.g.,

* Corresponding author at: UVHC, TEMPO Research Center, F-59313 Valenciennes, France. Tel.: +33 327511322.
E-mail address: cyrille.pach@univ-valenciennes.fr (C. Pach).

0967-0661/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.conengprac.2013.05.004

distributed or coordinated control architectures), other than centralized in which all information is gathered and used by a unique central controller, which leads to incoherent comparisons or results if applied in these emerging architectures.

Despite this, an increasing production control activity is presently being led, focusing on alternative control architectures and their ability to behave in a dynamic environment, such as the proposal by Fattahi and Fallahi (2010). This is mainly due to the evolving industrial need, which can be summed up as follows: from traditional static optimized scheduling towards more reactive, sustainable or agile control. This evolving need leads to the need for more complex performance evaluation, not only expressed traditionally in terms of production delays for a given set of tasks, but also in terms of sustainability or the ability to evolve in a constantly changing world (e.g., energy consumption, carbon footprints).

The OR community has changed to consider this evolution. For example, one interesting action, directed by the French Operational Research and Decision Support Society (ROADEF), has led to the organization of several challenges since 1999. A challenge is a set of complex problems to be solved by the community, and the research team that proposed the best results is rewarded.[1] In our opinion, these challenges can be considered as benchmarks that were proposed to the whole community. In the beginning of these challenges, problems were purely static. However, more recently, the problem definition may contain some dynamic data, leading to re-assignment decisions to be made within fixed time window (e.g., the 2009 challenge). Meanwhile, even though production and scheduling were sometimes studied in these challenges, flexible production system's manufacturing and scheduling, and their specific constraints, have never been addressed.

The production control community has also proposed benchmarks intending to allow the coherent comparison of production control architectures and systems, taking the dynamics of the environment into account. For example, Valckenaers et al. (2006) proposed a benchmark that is methodologically oriented, dealing with the way to construct a benchmark for control evaluation. Brennan and O.W. (2002) proposed a benchmark designed to integrate dynamic data. Cavalieri, Macchi, and Valckenaers (2003) proposed a web simulation testbed for the manufacturing control community. Pannequin, Morel, and Thomas (2009) proposed an emulation-based benchmark case study devoted to a product-driven system, and Mönch (2007) proposed a simulation benchmarking system.

These benchmarks are interesting since they try to deal with the dynamic behavior of the system to be controlled, which is harder to formalize in a simple and exclusively quantitative way, like benchmarks from the OR community. If dynamic data, real-time considerations and unpredictable events must be managed and their impact evaluated, this drastically increases the complexity to develop a usable, clearly designed benchmark. In our opinion, this increasing complexity forces the production control benchmarks to focus on specific aspects of benchmarking (e.g., methodological or simulation aspects), restrict the control architecture too much (e.g., product-driven, distributed), or compel the researchers to use specific tools (e.g., simulators). In addition, none of these benchmarks offers operational, fully informed data sets for coherent tests and comparisons. Therefore, despite some very interesting trials and the huge effort, these benchmarks are not often used as the OR community benchmarks.

In the constantly evolving research environment, with a unceasingly increasing importance paid to quality of results, researchers from the production control community, and a growing number of researchers from the OR community, are still seeking for a benchmark that can help them to characterize the static and dynamic behaviors of their control system, taking realistic production constraints into account.

Drawn from the experience of the authors, the conclusion of this literature review is that it is interesting to define a benchmark, allying the advantages of the benchmarks proposed by both communities, usable by both communities, and based upon a physical, real-world system to stimulate benchmarking activities to be grounded in reality. To propose such a benchmark to researchers is the aim of this paper. This conclusion is also consistent with the current determination of the IFAC TC 5.1, which tries to design, use and disseminate of manufacturing control benchmarks.

The rest of the paper is organized as follows. First, Section 2 introduces the proposed benchmark process. Sections 3, 4, and 5 detail the three steps of this benchmarking process: data preparation, experimentation, and reporting. Section 6 presents three applications of the benchmark for illustration purpose. Finally, Section 7 draws the conclusions and presents the prospective for future research.

2. The benchmarking process

Three consecutive steps compose the proposed benchmarking process, which are presented in Fig. 1.

The first step, called *data preparation*, concerns the sizing and the parameterization of the case study. Given a generic model of the target system to schedule and control, the *first benchmarking decision* is to choose the “data set”. A data set includes usually an instance of a model of the target system on which the benchmark is applied, accompanied with the input data needed to make this model work. Once this data set is chosen, *the second decision* concerns the definition of the objective function. The couple (data set, objective function) defines the **reference scenario**, called scenario #0. The *third decision* to make in this step is to decide whether or not dynamic behavior should be tested. If yes, then the *fourth and last decision* is that the researchers must decide which dynamic scenarios they are willing to test in a list of dynamic scenarios.

Once defined, scenario #0 contains only static data (i.e., all the data is known at the initial date), which allows researchers to test deterministic optimization mechanisms for a given set of inputs (e.g., OR approaches, simulation or emergent approaches, multi-agents approaches), especially if only few constraints are relaxed. In this stage, performance measures are purely quantitative. Scenario #0 can be used to test different optimization approaches, to evaluate the improvement of certain criteria (e.g., $C_{\max}$ values), or to check the basic behavior of a control system in real time where all data are known initially.

The second step, called *experimentation*, is composed of two kinds of experiments:

1. The *static stage*, which concerns the treatment of the reference scenario (i.e., scenario #0), and
2. The *dynamic stage*, which concerns the treatment of the dynamic scenarios.

If the researchers had selected the second option in the previous step, they will execute two types of dynamic scenarios, introducing perturbations (1) on the target system and (2) on the control system itself. In this stage, researchers can test control approaches and algorithms, using the scenario #0 into which some dynamic events are inserted, which defines several other scenarios with increasing complexity.

[1] http://challenge.roadef.org/2012/en/index.php.

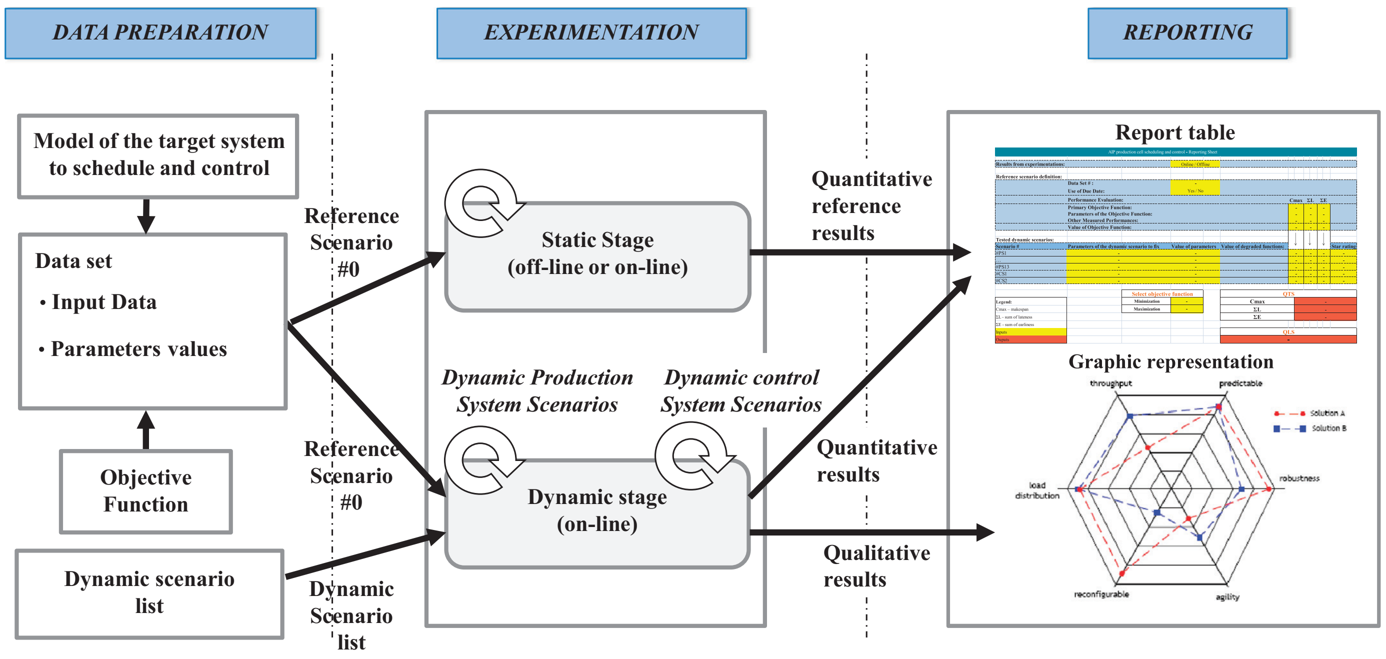


Fig. 1. The benchmarking process.

The output of the static stage concerns only quantitative data, while performance measures can be both quantitative (e.g., relative degradation of performance indicators) or qualitative (e.g., robustness level) in the dynamic stage. It is important to note that only the static stage is compulsory in the research; the dynamic stage is optional. However, if researchers need to quantify certain dynamic features of their control system, it is necessary for the static stage to be performed before the dynamic stage. Researchers could then design solutions that dissociate or integrate both static and dynamic stages, leading to designing coupling static optimization with dynamic behavior (e.g, proactive, reactive, predictive–reactive scheduling methods, the interested reader can consult Davenport and Beck (2000), who did a widespread literature survey for scheduling under uncertainty.)

The third step, called *reporting*, concerns the reporting of the experimental results in a standardized way:

- The parameters and the quantitative results from the static and dynamic stages are summarized in a report table, which facilitates future comparisons and characterization of the approaches, and
- The qualitative results obtained using the dynamic scenarios are presented via graphic representations.

This benchmark has been generically designed: it can be applied to different target systems, typically production systems, but not only (e.g., healthcare systems). In this paper, this benchmark is applied on the flexible job-shop scheduling problem, whose model is inspired from an existing flexible cell. The remaining of the paper deals with such an application and is organized according the three introduced steps: data preparation, experimentations and reporting.

3. First step: data preparation

The output of the data preparation step is the reference scenario #0, and if used, the dynamic scenario list to be tested. During this step, the target system must be defined. According to the method, this step is decomposed into three stages: the elaboration of the formal model of the target system, the definition of the data set and the objective function, all this defining the static scenario #0, and if used, the elaboration of the dynamic scenario list.

3.1. *The formal model of the target system: a flexible job shop system*

Since our desire is to ground the benchmark into reality, we propose to get inspiration from a real assembly cell: the AIP-PRIMECA cell at the University of Valenciennes. From an OR perspective, this system can be viewed as Flexible Job Shop, leading to the formulation of a Flexible Job-Shop Scheduling Problem (FJSP).

This section presents then the corresponding generic model and a static instantiation of this model to the AIP-PRIMECA cell. The idea is to first formalize a generic FJSP. This formal, highly parameterized model guarantees a certain level of genericity for future studies or for the development of a parameterized linear program. In a second subsection, an instantiation of this model is proposed according to the *static* parameters of the AIP-PRIMECA production cell, in other words, all the parameters of the FJSP that will be assumed constant all along this benchmark (e.g., transportation system and its topology, location of machines, standard production times).

3.1.1. *Generic model of the flexible job shop scheduling problem*

In OR literature, FJSP is considered as a generalization of the traditional job shop problem (JSP). The underlined terms are important to understand the formalization for this benchmark:

Let consider n jobs to be processed on different m machines. Each job j has its own production sequence composed of some elementary manufacturing operations. Those operations or tasks can be executed on one or more machines.

The main difference between FJSP and the JSP is that a machine can perform different types of operations in FJSP. The assignment of operations to the machines is not *a priori* fixed like in the traditional job shop problem. For this reason, many papers used two-phase methods to deal with the FJSP. The assignment problem is solved in the first step, while the second step aims to solve the sequencing of the assigned operations on machines.

Most of the flexible job shop problems are proved to be NP-hard (Conway, Maxwell, & Miller 1967). The flexibility increase the complexity of the problem greatly because it requires an additional level of decisions (i.e., the selection of machines on which job should be proceed) (Brandimarte, 1993). In addition to the basic constraints (e.g., precedence constraints, disjunctive constraints), we take into account in our study realistic constraints that are generally omitted. This concerns the transportation between two machines in the production system, the queue capacity of machines, and jobs limitation in the shop floor.

In the following, we present a mixed integer linear program (MILP) of the FJSP problem. However, we assume that the behavior is ideal for the best execution of this method (e.g., no machine breakdown is considered; no maintenance tasks are planned).

In order to formulate the FJSP, we introduce first some parameters, variables and the constraints.

Notations for parameters

P	set of jobs, $P=\{1,2,\dots,n\}$
R	set of machines, $R=\{1,2,\dots,r\}$
I_j	set of operations of the job j, $I_j=\{1,2,\dots,\lvert I_j\rvert\}$, $j\in P$
O_{ij}	operation i of the job j
R_{ij}	set of machines that can perform operation O_{ij}, $R_{ij}\in R$
p_{ij}	processing time of operation i ($i\in I_j$)
tt_{r1r2}	transportation time from machine r_1 to r_2
MJ	maximum simultaneous jobs in the shop floor
ci_r	input queue capacity of machine r
d_j	due date of job j, $j=1,\dots,n$
α_j	tardiness penalties associated to the job j, $j=1,\dots,n$
β_j	earliness penalties associated to the job j, $j=1,\dots,n$

Notations for variables

t_{ij}	completion time of operation O_{ij} ($i\in I_j$), $t_{ij}\in N$
μ_{ijr}	a binary variable set to 1 if operation O_{ij} is performed on machine r; 0, otherwise.
b_{ijkl}	a binary variable set to 1 if operation O_{ij} is performed before operation O_{kl}; 0, otherwise.
tr_{ijr1r2}	a binary variable set to 1 if job j is transported to machine r_2 after performing operation O_{ij}; 0, otherwise.
w_{ijr}	waiting time of operation O_{ij} in the queue of machine r
wv_{ijklr}	a binary variable set to 1 if operation O_{ij} is waiting for operation O_{kl} in the queue of machine r; 0, otherwise.
z_{lj}	a binary variable set to 1 if job l and job j are in the shop floor in the same time; 0, otherwise.

3.1.2. Detail of the constraints

Disjunctive constraints: A machine can process one operation at time, and an operation is performed by only one machine.

$$t_{ij}+p_{kl}\mu_{klr}+BMb_{ijkl}\le t_{kl}+BM,\quad \forall i,k\in I,\ \forall j,l\in P,\ \forall r\in R_{ij} \tag{1}$$

where BM is a large number.

$$b_{ijkl}+b_{klij}\le 1\quad \forall i\in I_j,k\in I_l,\forall j,l\in P \tag{2}$$

$$\sum_{r\in R_{ij}}\mu_{ijr}=1\quad \forall i\in I_j,\forall j\in P \tag{3}$$

Precedence constraints: These constraints insure job's production sequence. The completion time of the next operation considers the completion time of the previous one, the waiting time and the transportation time if the two operations are not performed in the same machine.

$$t_{(i+1)j}\ge t_{ij}+p_{(i+1)j}+w_{(i+1)jr2}+\sum_{r1,r2\in R}tt_{r1r2}tr_{ijr1r2}\quad \forall i\in I_j,\forall j\in P,\forall r1,r2\in R_{ij} \tag{4}$$

$$\sum_{\substack{r1,r2\in R\\ r1\ne r2}}tr_{ijr1r2}\le 1,\quad \forall i\in I_j,\forall j\in P \tag{5}$$

Allocation and transportation relationship: If successive operations of a job are performed on different machines, there is a transportation operation between those two machines. Transportation delays are set to zero, and the transportation system has unlimited capacity.

$$\mu_{ijr1}+\mu_{(i+1)jr2}-1\le tr_{ijr1r2}\quad \forall i\in I_j,\forall j\in P,\forall r1,r2\in R_{ij},r1\ne r2 \tag{6}$$

$$\mu_{ijr1}+\mu_{(i+1)jr2}\ge(1+\varepsilon)tr_{ijr1r2}\quad \forall i\in I_j,\forall j\in P,\forall r1,r2\in R_{ij},r1\ne r2 \tag{7}$$

where ε is a small number.

Queue capacity of the machine input and FIFO rule: Each machine has a limited queue capacity. No more operations than this queue capacity can wait in the queue. The first job arriving in the queue is the first treated.

$$b_{ijkl}+wv_{ijklr}\le 1\quad \forall i,k\in I,\forall j,l\in P,\ \forall r\in R_{ij}\cap R_{kl} \tag{8}$$

$$b_{ijkl}-wv_{klijr}\ge 0\quad \forall i\in I_j,\forall k\in I_l,\forall j,l\in P,\forall r\in R_{ij}\cap R_{kl} \tag{9}$$

$$wv_{ijklr}+wv_{klijr}\le 1\quad \forall i\in I_j,\ \forall k\in I_l,\forall j,l\in P,\forall r\in R_{ij}\cap R_{kl} \tag{10}$$

$$t_{ij}-p_{ij}+BMb_{ijkl}+BMwv_{klijr}\le t_{kl}-p_{kl}-w_{klr}+2BM\quad \forall i\in I_j,\ \forall k\in I_l,\forall j,l\in P,j\ne l,\forall r\in R_{ij}\cap R_{kl} \tag{11}$$

$$w_{klr}\le \sum_{\substack{i\in I_j\\ j\in P,j\ne l}}p_{ij}wv_{klijr}\quad \forall k\in I_l,\forall l\in P,\forall r\in R_{kl} \tag{12}$$

$$\mu_{ijr}+\mu_{klr}\ge 2(wv_{ijklr}+wv_{klijr})\quad \forall i\in I_j,\forall k\in I_k\forall j,l\in P,\forall r\in R_{ij}\cap R_{kl} \tag{13}$$

$$t_{ij}+BMb_{ijkl}\le t_{kl}+BM\quad \forall i\in I_j,\forall k\in I_k,\forall j,l\in P \tag{14}$$

$$t_{ij}-p_{ij}\mu_{ijr}+BMb_{ijkl}\le t_{kl}-p_{kl}\mu_{klr}+BM\quad \forall i\in I_j,\forall k\in I_k,\forall j,l\in P,\forall r\in R \tag{15}$$

$$t_{ij}-p_{ij}\mu_{ijr}-w_{ijr}+BMb_{ijkl}\le t_{kl}-p_{kl}\mu_{klr}-w_{klr}+BM\quad \forall i\in I_j,\forall k\in I_k,\forall j,l\in P,\forall r\in R \tag{16}$$

$$\sum_{\substack{l\in P,k\in I_l\\ l\ne j}}wv_{ijklr}\le ci_r-1,\quad \forall i\in I_j,\forall j\in P,\forall r\in R_{ij}\cap R_{kl} \tag{17}$$

Limitation of the number of jobs in the system: The number of simultaneous jobs in the shop floor can be limited by *MJ*. O_{0j} defines the first operation the job j (i.e., loading), and O_{uj} defines the last one (i.e., unloading).

$$\sum_{\substack{l\in P\\ l\ne j}}z_{lj}\le MJ-1\quad \forall j\in P \tag{18}$$

$$z_{jl}\ge b_{0luj}+b_{0j0l}-1\quad \forall j,l\in P \tag{19}$$

$$z_{jl}\le 1-b_{0l0j}+b_{ujul}\quad \forall j,l\in P \tag{20}$$

$$z_{jl}\ge b_{0l0j}+b_{ujul}-1\quad \forall j,l\in P \tag{21}$$

Variables and constraints in the case of due-date based production

Some constraints and variables must be added only in the case of due-date-based production.

Variables:

E_j: Earliness of job j

T_j: Tardiness of job j

A job j is early (resp. tardy) if $E_j>0$ (resp. $T_j>0$).

Constraints:

$$E_j = \max\{dj - t_{ij}, 0\} \quad \forall i \in I_j, j \in P \tag{22}$$

$$T_j = \max\{t_{ij} - dj, 0\} \quad \forall i \in I_j, j \in P \tag{23}$$

Constraints for the type of each variable

$$t_{ij} \geq p_{ij} \quad \forall i \in I_j, j \in P \tag{24}$$

$$b_{ijkl} \in \{0, 1\} \quad \forall i \in I_j, j \in P, \forall k \in I_l, l \in P \tag{25}$$

$$tr_{ijrr'} \in \{0, 1\} \quad \forall i \in I_j, j \in P, \forall r, r' \in R_{ij}, \tag{26}$$

$$m_{ijr} \in \{0, 1\} \quad \forall i \in I_j, j \in P, \forall r \in R_{ij}, \tag{27}$$

3.1.3. Quantitative performance

Different criteria are used in the measurement of the quantitative performance. The well-known criteria are cited below, according to the variables and parameters presented above.

Makespan is the time at which the last job is completed. In general, the makespan is denoted as $C_{\max}$. Its value is calculated by the formula

$$C_{\max} = \max_{\forall i \in I_j \forall j \in P} t_{ij} \tag{28}$$

Flow time is the time spent by the job in the shop which is equal to the sum of the processing time on each machine, including the process plan for the considered part and the waiting time in queues. Let C_j be the completion time of the last operation of the job j. The flow time of the job j is then C_j. In this case, the objective is to minimize the total completion time: $\Sigma_{\forall j \in P} C_j$

Earliness and tardiness of jobs is the comparison of the actual completion time of jobs with the desired completion time. The earliness of a job i is $E_i = \max(0, d_i - C_i)$. In the same way, the tardiness T_i of a job i is the positive difference between the completion time and the due date: $T_i = \max(0, C_i - d_i)$. Different criteria can be optimized. The goal is to minimize the amount of earliness ($\Sigma_{\forall i \in P} E_i$), the amount of tardiness ($\Sigma_{\forall i \in P} T_i$) or both criteria ($\Sigma_{\forall i \in P}(\alpha_i T_i + \beta_i E_i)$ or its quadratic form), where α_i and β_i are respectively delay cost and handling cost.

Machine utilization depends on the shop rather than the jobs. It is a fraction of the machine capacity used in the processing. The average utilization of m machines and n jobs is proportional to the maximum flow time, as expressed this formula:

$$U = \frac{\Sigma_{\forall i \in I_j \forall j \in P} p_{ij}}{m C_{\max}} \tag{29}$$

This criterion is rather a behavioral performance indicator than a production performance indicator, but it is used when bottleneck analysis are performed.

Work in process is the time spend by jobs in the queue before a machine. The objective is to minimize the total waiting time in the machine inputs W.

$$W = \sum_{\forall r \in R} \sum_{\forall j \in P} \sum_{\forall i \in I_j} w_{ijr} \tag{30}$$

Multi-objective optimization: The different criteria cited above can be mixed to optimize more than one objective. In the literature, many papers used multi-objective optimization for FJSP (Azardoost & Imanipour, 2011; Ho and Tay, 2008; Taboun and Ulger, 1992), but to the best of our knowledge, no one has taken into account the additional constraints presented in this paper.

3.1.4. Instantiation of the model: application to the AIP-PRIMECA cell

The previous formalization of a FJSP is generic enough to consider its application to the AIP-PRIMECA cell. A short instantiation of this model applied to the AIP cell and relevant static parameters are given below. These data are not assumed to change during the static or the dynamic stages.

The following data are relative to products.

Components

Six components are available ("Plate", "Axis_comp", "I_comp", "L_comp", "r_comp" and "screw_comp"). Purchase orders are assumed to insure sufficient quantities of these components as desired. In future research, a limited supply will be considered as a new constraint.

Jobs (Sub-assemblies)

In this paper, seven types of jobs (sub-assemblies) can be manufactured. They are denoted "B", "E", "L", "T", "A", "I" and "P". Components are used to manufacture these types of job.

Production sequence (Ordered manufacturing operations list)

A manufacturing operation is an elementary action carried out on sub-assemblies, which is not a transportation task. There are eight manufacturing operation types ("Plate loading", "Axis mounting", "r_comp mounting", "I_comp mounting", "L_comp mounting", Screw_comp mounting", "Inspection", and "Plate unloading"). For example, "I_comp mounting" means that the I component must be mounted on the plate. The inspection is completed by an automatic inspection unit (i.e., vision system).

A production sequence (ordered manufacturing operation list) is associated to each type of job. The operation lists have the same structure: a single load, a series of component mountings, a single inspection and a single unloading. Between two successive manufacturing operations, it may be required a transportation operation if the two operations are not done at the same place. Constraints (6) and (7) ensure this relationship in the MILP. Table 1 shows the different production sequences.

In static scenario, there is no quality issue. In dynamic scenarios, after an inspection, when the job has a quality problem, an extra manufacturing operation, called "recovery", must be inserted just before the plate unloading operation. This is a manual operation that tries to fix the quality problem. This operation is assumed to fix 100% of the quality problems.

Table 1
Production sequence for each type of job.

	"B"	"E"	"L"	"T"	"A"	"I"	"P"
#1	Plate loading	Plate loading	Plate loading	Plate loading	Plate loading	Plate loading	Plate loading
#2	Axis mounting	Axis mounting	Axis mounting	Axis mounting	Axis mounting	Axis mounting	Axis mounting
#3	Axis mounting	Axis mounting	Axis mounting	Axis mounting	Axis mounting	Axis mounting	Axis mounting
#4	Axis mounting	Axis mounting	Axis mounting	r_comp mounting	Axis mounting	I_comp mounting	r_comp mounting
#5	r_comp mounting	r_comp mounting	I_comp mounting	L_comp mounting	r_comp mounting	Screw_comp monting	L_comp mounting
#6	r_comp mounting	r_comp mounting	I_comp mounting	Inspection	L_comp mounting	Inspection	Inspection
#7	I_comp mounting	L_comp mounting	Screw_comp mounting	Plate unloading	I_comp mounting	Plate unloading	Plate unloading
#8	Screw_comp mounting	Inspection	Screw_comp mounting		Screw_comp mounting		
#9	Inspection	Plate unloading	Inspection		Inspection		
#10	Plate unloading		Plate unloading		Plate unloading		

3.1.5. *Products, client order*

Three kinds of products are proposed to clients. They are called "BELT" "AIP" and "LATE". A product is thus a subset of jobs, or sub-assemblies, among the seven possible job types, corresponding to different arrangements of letters. The jobs that compose these products can be manufactured in any order.

Fig. 2 shows pictures of components, jobs and products. The relationships between these elements are highlighted.

A set of different products to complete for a client, possibly with an associated global due date, is called a client order. A product is considered finished when its latest job is finished and leaves the cell. In the MILP, this is handled by considering that a client order is as a set of products, and each of these products considered as a set of jobs. Thus, the due date of the client order can be assigned to this whole set of jobs. In other words, each job in the set is assigned this due date in the MILP (constraints (22) and (23)).

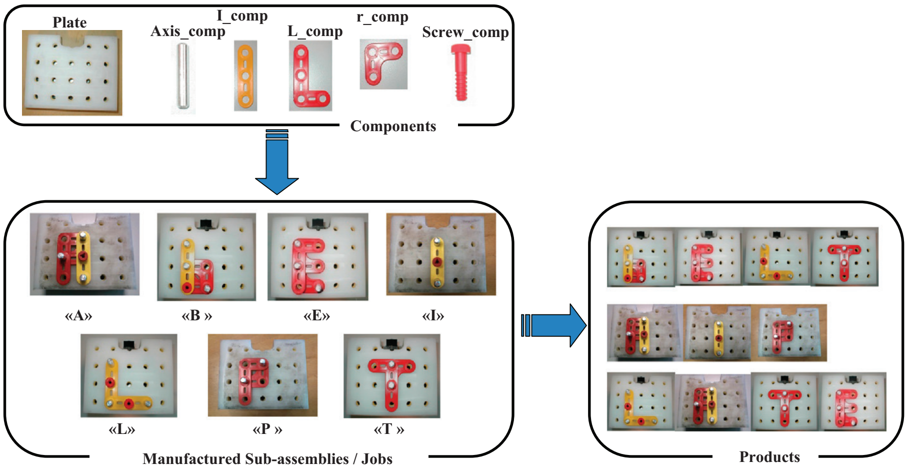


Fig. 2. Components, jobs and products.

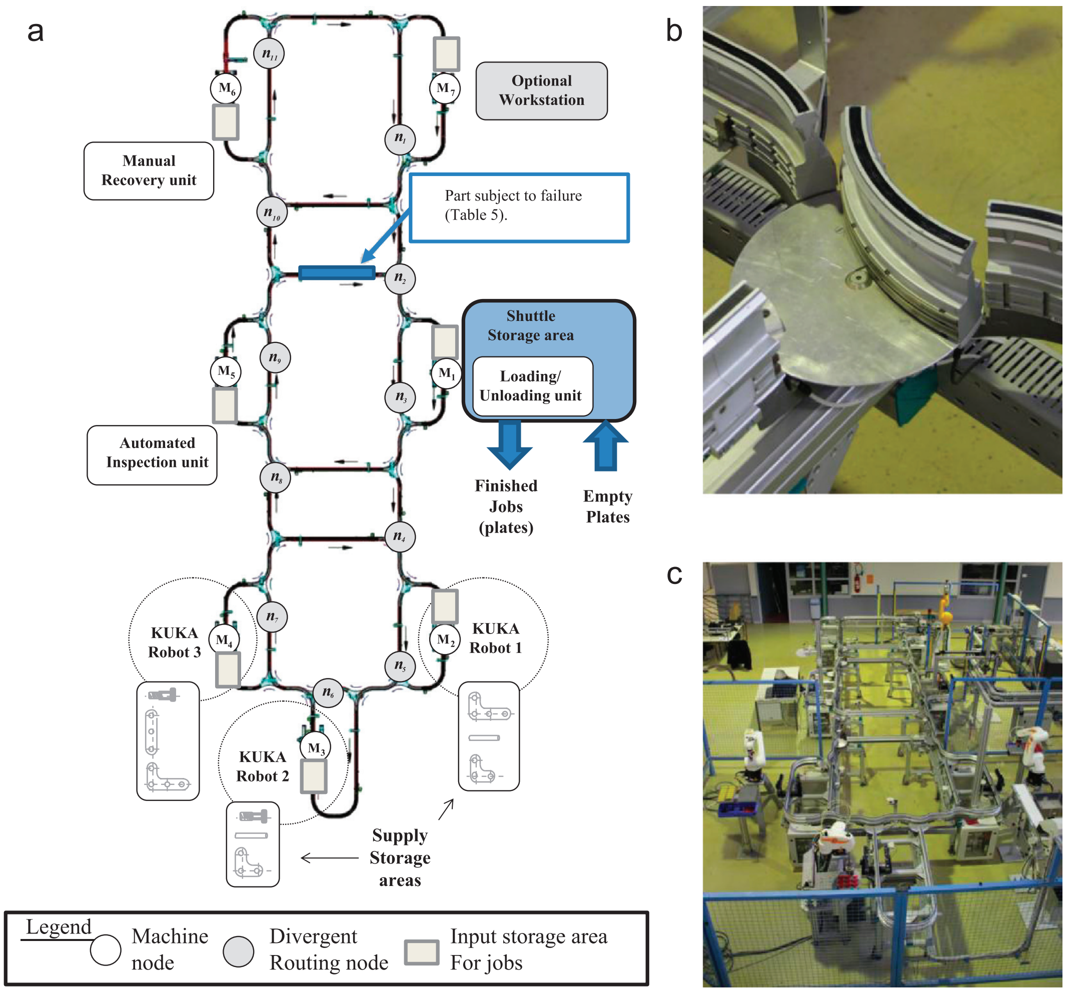


Fig. 3. Cell plan (a), picture of divergent node n_5 (b), and the corresponding cell picture (c).

The following data are relative to machines.

Machines:
The cell is composed of seven machines (Fig. 3a):

- a loading/unloading unit (M_1),
- three assembly workstations (M_2, M_3 and M_4),
- an automatic inspection unit (M_5),
- a recovery unit (M_6), which is the only manual workstation in the cell, and
- an optional workstation (M_7), only used for a given dynamic scenario.

Machines are responsible for the completion of manufacturing operations to do the jobs. Some machines are able to complete the same manufacturing operation (flexibility, variable μ_{ijr}, decides which machine performs which operation in the MILP), while some manufacturing operations can be completed on a single machine.

Supply storage area:
Each assembly workstation has also its own supply storage area, which is used for supply purpose in components (Fig. 3). Since, in this benchmark, this supply is assumed infinite, this supply storage area is not used in the scenarios described in the previous mathematical model.

Manufacturing operation processing times:
Table 2 provides the manufacturing operations feasible on the different machines by providing the corresponding manufacturing operation processing times. Otherwise, these machines cannot carry out the manufacturing operations.

The following data are relative to the transportation system.

Topology of the transportation system:
The machines are connected using a transportation system. The transportation system is a one-direction monorail system with rotating transfer gates at routing nodes (Montech Technology 2009). Thus, this transportation system can be considered as a directed, strongly-connected graph, composed of the following nodes (Fig. 3a):

- M_1, M_2, M_3, M_4, M_5, M_6, M_7 (white nodes in Fig. 3a) are the machine nodes, and
- n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9, n_{10}, n_{11} (gray nodes in Fig. 3a) are divergent routing nodes in which routing decisions must be made (e.g., from n_{11}, it is possible to reach the adjacent nodes n_1 or M_7).

Fig. 3 gives the whole topology of the cell, including nodes, highlighting the transportation system, the exact machine location, and the types of components available in its supply storage area.

Table 2
Manufacturing operations processing times (in seconds).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7 For PS#3
Plate loading	10						
Product unloading	10						
Axis		20	20				(20) For PS#3
r_comp		20	20				(20) For PS#3
I_comp			(20) For PS#2	20			(20) For PS#3
L_comp		20		20			(20) For PS#3
Screw_comp			20	20			
Inspection					5		
Recovery						60	

Note: Some values in this table have been modified recently in dynamic scenarios that focus on perturbations applied to the production system (i.e., dynamic scenarios PS#2 and PS#3). The corresponding cells are light gray with data between parentheses.

3.1.6. Shuttles and shuttle storage area

Shuttles are self-propelled transportation resources that transport a job from node to node (Fig. 4a). A maximum of 10 shuttles are available. At each moment, a maximum of 10 jobs can be manufactured at the same time (constraints (18)–(21) in the MILP). Each shuttle embeds a basic behavior to avoid colliding with other shuttles, detect a transfer gate (i.e., stop-and-go system), manage speed in curves and in strait lines, and dock in front of the machines. Because of the technological solution adopted for conveyance, shuttles are governed according to a FIFO rule.

For simplification purposes, it is assumed that empty shuttles are stored in a specific area near the machine M_1, with unlimited storage capacity. They enter and leave the cell there. Empty shuttles are loaded with the plates (operation #1) when desired on M_1 before entering the cell, go to production and then return to M_1 where they are unloaded for delivery. Next, they return in this empty shuttle storage area near M_1 until their next use. Thus, there is no possible empty trip in the cell; in the cell, a shuttle has always a job embedded on it.

3.1.7. Job input storage areas

Before each machine stands a job input storage area with limited capacity, excluding the operating area in front of the machines (Fig. 4b). This capacity is set in the cell for one shuttle for all machines (constraint (17) in the MILP). More than one shuttle in the input storage area is risky since it may lock the transportation system for other products and block transfer gates by parking on them. Specific rules must be designed to handle this situation (e.g., shuttles must keep moving in the central loop until a place is free). Despite this, this value can be increased for simulation or theoretical studies, but not for real experimental studies. The transfer time between the job input storage area and the machine corresponds to the time the shuttle has to move from the storage area and to dock in front of the machine. This time is neglected for all the machines.

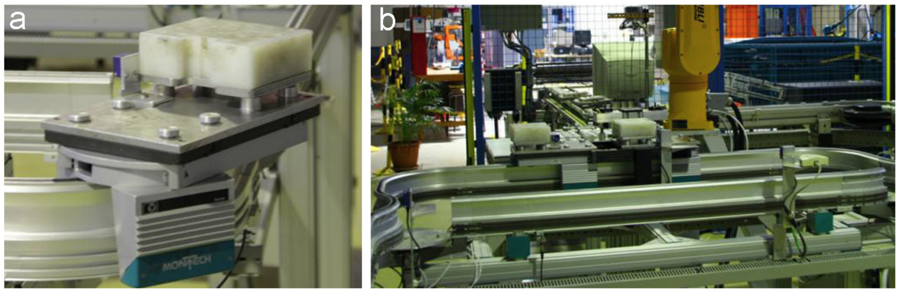


Fig. 4. (a) Shuttle with an empty plate and (b) two queued shuttles in the job input storage area of a machine.

Table 3
Theoretical transportation times between adjacent nodes (in seconds).

		Destination nodes																	
Source nodes		**N_1**	**N_2**	**N_3**	**N_4**	**N_5**	**N_6**	**N_7**	**N_8**	**N_9**	**N_{10}**	**N_{11}**	**M_1**	**M_2**	**M_3**	**M_4**	**M_5**	**M_6**	**M_7**
	N_1		**4**	–	–	–	–	–	–	–	**5**	–	–	–	–	–	–	–	–
	N_2	–		**4**	–	–	–	–	–	–	–	–	**5**	–	–	–	–	—	–
	N_3	–	–		**4**	–	–	–	**5**	–	–	–	–	–	–	–	–	–	–
	N_4	–	–	–		**4**	–	–	–	–	–	–	–	**5**	–	–	–	–	–
	N_5	–	–	–	–		**3**	–	–	–	–	–	–	–	**11**	–	–	–	–
	N_6	–	–	–	–	–		**4**	–	–	–	–	–	–	–	**5**	–	–	–
	N_7	–	–	–	**5**	–	–		**4**	–	–	–	–	–	–	–	–	–	–
	N_8	–	–	–	–	–	–	–		**4**	–	–	–	–	–	–	**5**	–	–
	N_9	–	**5**	–	–	–	–	–	–		**4**	–	–	–	–	–	–	–	–
	N_{10}	–	–	–	–	–	–	–	–	–		**4**	–	–	–	–	–	**7**	–
	N_{11}	**9**	–	–	–	–	–	–	–	–	–		–	–	–	–	—	–	**10**
	M_1	–	–	–	**6**	–	–	–	**7**	–	–	–		–	–	–	–	–	–
	M_2	–	–	–	–	–	**5**	–	–	–	–	–	–		**13**	–	–	–	–
	M_3	–	–	–	–	–	–	**6**	–	–	–	–	–	–		**7**	–	–	–
	M_4	–	–	–	**7**	–	–	–	**6**	–	–	–	–	–	–		–	–	–
	M_5	–	**7**	–	–	–	–	–	–	–	**6**	–	–	–	–	–		–	–
	M_6	**12**	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–		**13**
	M_7	–	**6**	–	–	–	–	–	–	–	**7**	–	–	–	–	–	–	–	

Table 4
Description of the possible data sets to design scenario #0.

				Client order				
	Number of shuttles	Input storage capacity	Transportation times	Order #	Products			Due date (s) (if pull/JIT mode, else not used)
					BELT	AIP	LATE	
A0	Infinite	Infinite	Zero	#1	1	–	–	325
				#2	–	1	–	209
B0	10	1	Table 3	#1	–	2	–	327
C0	4	1	Table 3	#1	1	–	–	382
				#2	–	1	–	238
D0	Infinite	1	Zero	#1	1	–	–	321
				#2	2	1	–	863
E0	Infinite	Infinite	Table 3	#1	2	1	–	947
				#2	–	2	1	786
				#3	–	–	2	637
F0	Infinite	1	Table 3	#1	–	1	3	961
				#2	2	1	–	880
				#3	1	1	1	919
G0	8	Infinite	Zero	#1	1	2	1	1032
				#2	2	3	1	1409
				#3	3	2	3	2480
H0	10	Infinite	Table 3	#1	1	2	1	992
				#2	2	3	1	1676
				#3	3	2	3	1801
I0	10	1	Zero	#1	2	2	3	1709
				#2	2	4	–	1356
				#3	3	2	3	2276
J0	10	1	Table 3	#1	4	4	4	2793
				#2	6	5	1	3175
				#3	–	8	3	2266
K0	10	1	Table 3	#1	8	10	12	9067
				#2	10	8	10	7342
				#3	12	10	8	6841
L0	10	1	Table 3	#1	15	20	25	17122
				#2	20	25	15	13,104
				#3	25	15	20	18,054

3.1.8. *Theoretical transportation times*

Table 3 shows the theoretical transportation time that is associated to each couple of adjacent nodes. This time is called theoretical since it depends on the actual speed of shuttles. In fact, in reality, a transportation process may last longer that this time due to unexpected events or jamming phenomena.

3.2. *The data set and the objective function*

Given a formal generic model of the target flexible job-shop system, this section tries to define the reference scenario #0 to be chosen by the researchers for the whole benchmark. In this scenario, all the data is known initially. In other words, there are no perturbations. In scenario #0, there is no quality problem, and 100% of the jobs pass inspection. Thus, machine M_6 (i.e., recovery) must never be used in the static scenarios.

To design the scenario #0, the researchers must choose a data set, fixing two parameters: (1) the set of client orders, eventually with due dates, and (2) the constraints to be handled. In addition to this data set, the objective function, or performance function, must be determined. Table 4 presents the proposed different data sets. Data sets, in which the constraints are relaxed (e.g., infinite number of shuttles and/or infinite storage capacity and/or neglected transportation times), can be used to obtain bounds or to test basic optimization mechanisms. If the number of shuttles is assumed infinite, then constraints (18)–(21) in MILP should be relaxed. If the input storage capacity is assumed infinite, then it is constraint (17) that needs to be relaxed. Finally, if transportation times are neglected, then constraints (6) and (7) need to be relaxed. If a due-date-based strategy is chosen, the objective function will be different from the makespan; thus, constraints (22) and (23) should be considered in this case.

For researchers who pay attention to due-date based production, desired due dates for client orders are provided in the last column of Table 4. These dates have been fine-tuned according to the possible constraint relaxation that may lead to shorten production delays. The problem to be solved has to provide an accurate value for these due dates. As Gordon, Proth, and Chu (2002) noted, there are quite few results on flow and job shop with due-date assignment, and most of the papers are on single and parallel machine shops. Most studies on due-date based scheduling assume uniform distribution of due dates. However, in real production system, this assumption does not always hold true (Thiagarajan and Rajendran, 2005). Sourd (2005) presented a different method for generating due dates that defines the interval bounds in which due dates belong. Those bounds, or range factors, try to control the tightness and variance of due dates. This technique was used by many other authors, for example, Cho and Lazaro (2010).

For instance, let consider two parameters: the average tardiness factor τ and the range factor ρ. Due dates are thus generated from the uniform distribution:

$$\left[\max\left(0,(1-\tau-\rho/2)(dd_o+\sum_{j\in J, i\in I_j} p_{ij})\right),(1-\tau+\rho/2)(dd_o+\sum_{j\in J, i\in I_j} p_{ij})\right] \tag{31}$$

where dd_o is the minimum transportation time to perform the client-order o. When the transportation time is neglected, dd_o is set to 0. τ and ρ can take different values in the sets: $\tau\in\{0{,}2;\ 0{,}4;\ 0{,}6;\ 0{,}8\}$, $\rho\in\{0;\ 0{,}2;\ 0{,}4;\ 0{,}6;0{,}8\}$. The due dates in Table 4 are generated for $\tau=0.2$ and $\rho=0.5$, and values are rounded to the nearest integer.

Obviously, the model's objective function that takes into account due dates must logically take into consideration tardiness or earliness of products. For researchers who do not pay attention to due dates, these dates can be ignored, and other traditional criteria (e.g., $C_{\max}$) can be used. At the designer's discretion, new scenario can be designed, and a combination of these quantitative indicators or a multicriteria analysis can be performed.

3.3. The dynamic scenario list

If testing the dynamic scenarios, several dynamic scenarios must be chosen to evaluate the control system in case of unexpected events. Tables 5 and 6 give a list of dynamic scenarios, numbered according to the increasing complexity level. In Table 5, dynamic scenarios concern the perturbation of the production system; Table 6 provides the dynamic scenarios related to the perturbation of the control system itself. Obviously, researchers can introduce new scenarios or use these scenarios with different parameters. Otherwise, this list can be considered as a reference list for future comparisons.

For all these scenarios, when a resource (i.e., machine or conveyor) becomes unavailable, the jobs that are in the resource's waiting queue are able to leave, if desired. Thus, they do not stay blocked at this resource and can be reallocated elsewhere.

4. Second step: experimentation

In the experimentation step, the production control system or model is tested under the conditions defined in the reference scenario, possibly with a list of dynamic scenarios.

4.1. Static stage

In the static stage, the control system is running under the reference scenario #0, and the experimental results, reflecting the system performance, are collected. These results describe the observed performance level (e.g., throughput rate or machine utilization rate) that make it possible to analyze the system performance and to compare the performance of different systems; however, they do not tell why the performance is as it is. These results cannot reveal which factors account for differences in different measured performance levels.

In this stage, the experiment considers static data (i.e., perturbations are not considered) that lead to quantitative performance indicators, which are based on statistical theory. These quantitative performance indicators defined in the benchmarking framework have been previously introduced (e.g., makespan, throughput).

The static stage can be performed *off-line*, allowing, for example, optimization mechanisms before the real or simulated on-line applications. Usually, real-time constraints are not considered in such off-line approaches, and very efficient optimization tools, typically metaheuristics, can be designed. This stage can also be performed *on-line* – in other words, “on the fly” or in real time – applied to the real cell or a real-time simulator/emulator of the cell. The difference between the off-line and on-line experimentation is that the time to construct the scheduling plan may impact the results if the experimentation are on-line. However, this remains a static stage in that all the input data used to construct the scheduling plan are deterministically known at the initial date.

4.2. Dynamic stage

In the dynamic stage, certain data are not known at the initial time, usually perturbations. If the benchmark's dynamic feature is dealt with, for each dynamic scenario to be tested, the researchers always compare scenario #0: the reference scenario. Thus, the same conditions (i.e., data set and objective function) chosen for scenario #0 must be applied to allow a coherent relative comparison. In addition, the same models and algorithms as the ones proposed by the researchers for the static stage must be used for a coherent analysis.

These scenarios must be logically considered on-line: the real or simulated production system continues to evolve, and faced with unpredicted events, the control system must react. If the control system takes too much time (simulated or real time) to react, it must affect the production and degrade the performance. Dynamic algorithms and simulation must be extensively used by researchers. Obviously, the control system must not know about the perturbations before they occur.

In dynamic environments, performance is harder to evaluate than in static environments. We can identify two ways to perform this performance evaluation:

- *Quantitative*: Since scenario #1 and the following scenarios (Tables 5 and 6) are all based upon scenario #0, it is possible to evaluate the performance deterioration by comparing the evolution of quantitative indicators (e.g., percentage increase of the $C_{\max}$ values faced with a breakdown).
- *Qualitative*: Quantitative indicators may not be sufficient from a control perspective, whereas other more qualitative criteria should be analyzed: robustness and/or reactivity. These qualitative indicators are more subjective. They cannot be directly obtained from the experimental data, but they can be estimated. Since several different scenarios in dynamic situations have been proposed, it would be possible to define a lexicographical notations based upon the class of dynamic scenario effectively supported by the candidate control system.

In terms of qualitative performance indicators, this benchmark focuses on the robustness parameter, which is the ability of a control system to remain working correctly and relatively stable, even in presence of perturbations. Ideally, measuring robustness requires analyzing the system with all possible natural errors that can occur, which is not possible in reality. Verifying the system's operation and waiting for the occurrence of errors that occur infrequently is too time consuming. Until now, there has been no effective approach to quantitatively measure the robustness of a manufacturing control system. This benchmark considers the set of dynamic scenario tests, defined in Tables 5 and 6, to verify if the control system remains working correctly after the perturbation occurs. The evaluation of the qualitative aspect is based on notation using stars (Fig. 5). A number of stars (0, 1 or 2) is awarded for each scenario tested by the control system. This will be used to evaluate the global robustness.

Table 5
Dynamic production system scenarios.

Scenario	Description	Highlighted control behavior	Parameters
#PS1	All orders are canceled before production.	This scenario simulates a capacity to cancel orders and wait for a full restart	Machines: all Start time: date 0
#PS2	At a given time, one of the machines was improved and is now able to perform a new kind of manufacturing operation, which increases the flexibility level of the cell.	This scenario simulates a technological evolution forwarded in a production system. The evaluated control capacity is the capacity of the control system to adapt to evolution of machines.	Machine: M_3 Operation: I_comp Start time: just after the departure of the second shuttle from M_3 Updated processing time: 20 s. (see Table 2, Manufacturing Processing Times)
#PS3	At a given time, a new machine is added to the manufacturing cell and is immediately available for production.	This scenario simulates that machines are sometimes under maintenance for a long time and cannot be available. The evaluated control capacity is the capacity of the control system to adapt to evolution of the cell topology.	Machine: M_7 Operations available: Axis_comp, R_comp, I_comp and L_comp Start Time: just after the departure of the first shuttle from M_2. New processing times: see Table 2, last column, Manufacturing Processing Times.
#PS4	At a given time, the machine processing time increases for all its operations in a given time window.	This scenario simulates the wear of a tool that is replaced (i.e., maintenance). The evaluated control capacity is the capacity of the control system to adapt to evolution of the machine processing time.	Machine: M_2 New processing time: 40 s Start Time: just after the departure of the second shuttle from M_2. Duration (seconds): 25 × *Total number of jobs* If a job is processed by M_2, when the processing time returns to 20 s, the 40-s processing time is used for the current operation.
#PS5	At a given time, a rush order appear.	This scenario simulates the fact that client desires may evolve over time. The evaluated control capacity is the capacity of the control system to manage the introduction of a rush order.	Type of order: AIP Arrival Time: just after the end of the production of the fourth job in the cell. Due date: ASAP
#PS6	After a quality control, a product is canceled.	This scenario simulates the fact that product may not satisfy the client. The evaluated control capacity is the capacity of the control system to manage product cancellation.	Product canceled: BELT Date: just after the inspection of the first job of a BELT product.
#PS7	At a given time, a part of the conveyor system is due for maintenance in a given time window.	This scenario simulates that the conveying system has a limited reliability. The evaluated control capacity is the capacity of the control system to manage a routing change.	Start time: just after the fourth job is unloaded. The conveyor must no longer accept shuttles, and as soon as it is empty, the maintenance starts. Location of the part: part between nodes n_9 and n_2 located on Fig. 3a Duration (seconds): 25 × *Total number of jobs*
#PS8	After a number of components mounted with a certain machine, the component is lacking in a given time window.	This scenario simulates that supplies are always limited and must be adjusted.The evaluated control capacity is the capacity of the control system to manage stock re-provisioning.	Type of Component: Axis_comp Machine: M_3 Number of mounted components before being lacking: 10 Duration (seconds): 25 × *Total number of jobs.*
#PS9	At a given time, one of the redundant machines will go down in a given time window.	This scenario simulates that the machines have a limited reliability. The evaluated control capacity is the capacity of the control system to manage a redundant machine's breakdown.	Machine: M_2 Start time: just after the departure of the first shuttle from M_2. Duration (seconds): 25 × *Total number of jobs*
#PS10	At a given time, one of the critical machines (i.e., the only one that can perform a given task) will go down in a given time window.	This scenario simulates that a critical machine may result in the breakdown of the whole production system. The evaluated control capacity is the capacity of the control system to manage a critical machine's breakdown.	Machine: M_4 Start time: just after the departure of the second shuttle from M_4. Duration (seconds): 25 × *Total number of jobs.*
#PS11	A quality problem is added to the production system, and the recovery M_6 workstation must be used each time the inspection detects a quality problem.	This scenario simulates that quality issues are sometimes discovered during the production and must be solved on-line. The evaluated control capacity is the capacity of the control system to manage products that have defects.	Frequency of the quality problem: in a random way, 50% of inspected jobs (M_5) are to be recovered (M_6).

Table 5 (*continued*)

Scenario	Description	Highlighted control behavior	Parameters
#PS12	At a constant rate, a given machine becomes unavailable due to some malfunction.	This scenario simulates a repetitive machine breakdown problem. The evaluated control capacity is to check if the control system is able to detect the malfunction pattern and use it as an expected event.	Machines: M_2 Start Time: after the end of processing; at every 4 jobs. Duration: 60 s.
#PS13	Each time a product is finished and leaves the system, there is n% of chance that a new product of the same kind has to be done again as soon as possible if $C_{\max}$ is used, or if due dates are used, with an expected due date that equals the current time, plus the initial expected due date of the first product.	This scenario highlights the ability of the control system to mix predictive and reactive approaches. If n equals or is near to 100%, it can also be used to evaluate the ability of the control system to discover production patterns, then adapt itself in real time and improve performances with time.	Frequency of the problem: n={50, 75, 90, 95, 100}.
#PS14	The whole production system takes a pause; the system stays powered but must wait until a given time to continue production.	This scenario simulates the capacity to resume production after a major quality issue, discovered on the first job that leaves the system. No control decision can be applied during the pause. For simplification purposes, at the start time, all automatic cycles (manufacturing/routing) that are in progress must finish. The evaluated control capacity is the capacity of the control system to handle production pauses for any reason (e.g., quality problem, end of the day).	Machines: all Start time: just after the first job is unloaded. Duration (seconds): 25 × *Total number of jobs*
#PS15	At a given time, the entire production system is stopped (i.e., unpowered), and then restarted.	This scenario simulates the fact that major crisis are possible in production. The evaluated control capacity is the capacity of the control system to manage a full restart.	Start Time: just after the end of the fourth job. Duration: 10 s.

Table 6
Dynamic control system scenarios.

Scenario	Description	Highlighted control behavior	Parameters
#CS1	At a given time, a decisional entity breaks down for a pre-determined amount of time.	This scenario simulates that hardware supporting decisional functions have a limited reliability in a centralized or distributed architecture. The evaluated control capacity is the capacity of the control system to manage the loss of one of its decisional entities.	Targeted decisional entity: in centralized architecture, the computer supports the whole control process. In distributed architecture, the entity is selected randomly among the set of decisional entities (e.g., jobs, machines). Start Time: : just after the end of the third job Duration (seconds): 25 × *Total number of jobs*
#CS2	At a given time, the network supporting the communication among decisional entities breaks down for a pre-determined amount of time.	This scenario simulates that network communication may be a constraint in a distributed architecture. The evaluated control capacity is the capacity of the control system to manage the loss of the decisional network.	Start Time: just after the end of the third job Duration (seconds): 25 × *Total number of jobs*

Three steps are defined: (1) star awarding system, (2) qualitative scoring of robustness, and (3) quantitative scoring of robustness.

4.2.1. *Star awarding system*

Fig. 5 proposes the star awarding system, which is given for each dynamic scenario tested.

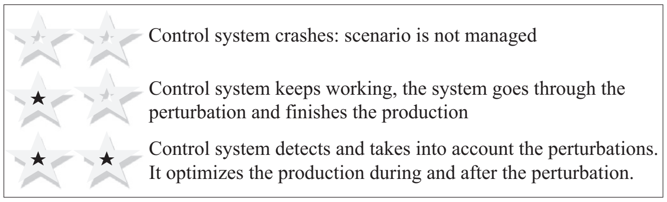
Table 7 summarizes the overall criteria for the 0 and 2 star rates for each of the dynamic scenarios. The one between these two limits would be rated with 1 star, since they could have displayed better behavior.

4.2.2. *Qualitative scoring of robustness*

The final qualitative classification depends of the number of stars that each approach obtains. To generalize this classification method, balancing the number of 2-star and 1-star ratings, researchers are invited to use the QuaLitative Score (QLS) formula:

$$QLS = max\left(2^*, \tfrac{1}{2}1^*\right) \tag{32}$$

This means that the researchers must select the row that maximizes the number of 2-star ratings and half of the 1-star ratings. This choice is justified by the maximum number of higher classification must be ranked better. For example, an approach that obtains 5 times a 2-star rating and 6 times a 1-star rating, the researchers must select the row 5, since $QLS = \max(5, 0.5 \times 6) = 5$. Thus, to obtain the maximum rating, the approach gives all the


Control system crashes: scenario is not managed

Control system keeps working, the system goes through the perturbation and finishes the production

Control system detects and takes into account the perturbations. It optimizes the production during and after the perturbation.

Fig. 5. Star awarding system.

dynamic scenarios with the maximum 2-star score, as given by: $QLS = \max(17,\ 0.5 \times 0) = 17$. Fig. 6 presents the different ratings obtained and the name of each classification.

Of course, this rating should be only used if at least one dynamic scenario is performed.

4.2.3. Quantitative scoring of robustness

The qualitative scoring does not allow researchers to compare the quality of the control system. It only states how well the control system handles the dynamic scenario (i.e., well: 2 stars, average: 1 star, or poorly: 0 star). This study can be completed to quantify to what degree the handling is effective. For example, if two control systems faced with the same dynamic scenario are rated with 2 stars, the resulting deteriorated makespan can be higher for one than the other. The degree of robustness is quantified in this step. This quantification is based on the results obtained with the dynamic scenarios weighted by the performance deterioration and evaluated with the reference scenario #0. This quantification also takes into account whether or not the objective function minimized or maximized.

If a minimized objective function is the goal, the QuanTitative Score (QTS) formula must be used:

$$QTS = \sum_{0<i\le 15} g_i \left(\begin{cases} e^{(Op_{ref} - Op_i/Op_{ref})}, \text{ if } Op_i > Op_{ref} \\ 1, \text{ otherwise} \end{cases} \right) \left(\begin{cases} e^{(Op_{best} - Op_i/Op_{best})}, \text{ if } Op_i > Op_{best} \\ 1, \text{ otherwise} \end{cases} \right) \tag{33}$$

If a maximized objective function is the goal, the following formula must be used:

$$QTS = \sum_{0<i\le 15} g_i \left(\begin{cases} (Op_i/Op_{ref}), \text{ if } Op_i < Op_{ref} \\ 1, \text{ otherwise} \end{cases} \right) * \left(\begin{cases} (Op_i/Op_{best}), \text{ if } Op_i < Op_{best} \\ 1, \text{ otherwise} \end{cases} \right) \tag{34}$$

where g_i is the grade of scenario $\mathbf{i}$ (0, 1 or 2 stars); Op_{ref} is the value obtained in the optimization criteria in the reference scenario; Op_i is the value obtained in the optimization criteria in the dynamic scenario $\mathbf{i}$; and Op_{best} is the overall best result obtained for the reference scenario #0. With this formula, the best overall score is 30 but is unreachable. It corresponds to the situation in which the control system is awarded with 2 stars for each dynamic scenario, no performance deterioration occurs in all these scenarios, and the result for the reference scenario #0 is the best one.

Evaluating the robustness of a control system in the dynamic stage is not simple and is a hard research problem in itself. Our evaluation method can be discussed and improved, especially if the quantitative score requires the best solution (i.e., the optimal one). It has the advantage to be a first evaluation method for robustness, and researchers are encouraged to use it and improve it. This can open up new interesting research areas. Of course, this evaluation can be skipped by researchers using this benchmark or can be stopped at the second step.

5. Third step: reporting

When the experimental results are available, it is important to present them explicitly. First of all, for each scenario (scenario #0 and dynamic scenarios), a Gantt diagram (e.g., Gantt machine) can be provided for graphic representation and intuitive behavioral analysis of our approach. However, this is not enough. A clear reporting of the results would help the researchers to understand the inputs and outputs of the benchmark when designing their models, forcing them to use a common method to summarize and standardize their results. This would facilitate the objectives for future comparisons between different approaches when other researchers publish their results. In that direction, we propose a reporting sheet to summarize the information related to the selected static and dynamic scenarios, design choices and performance indicators. Table 8 gives an example of such a sheet.

Finally, a conclusion must be proposed to point out the main features of the proposed approach, the lessons learned and the proposal's limits.

6. Three illustrative applications

To facilitate the appropriation of this benchmark, three different applications are presented consecutively. The first uses the benchmark through a mixed-integer linear program. In the two other applications, the same potential fields control approach is the focus, in simulation (second application) and in real experiments (third application). For each application, the benchmarking process is strictly applied, and the three steps (i.e., data preparation, experimentation and reporting) are given in detail.

These applications do not consider complex data sets since they are provided just for illustration purpose. Designing complete, exhaustive studies using the benchmark is beyond the scope of this paper. Meanwhile, the variety from the three proposed applications enable to show how this benchmark can be used with an OR approach (first application) or a control approach (second and third applications), and for the control approach, both in simulation (second application) and in a real system (third application). Of course, full, in-depth studies will be led in the near future, featuring innovative scientific approaches.

6.1. Using the benchmark through a linear program

The benchmark's formal model of the Mixed-Integer Linear Program (MILP) was implemented using Cplex.[2]. The Cplex solution is based on branch-and-cut algorithm and other OR techniques implemented by IBM ILOG. The main idea of these techniques is to subdivide the whole problem into sub-problems by fixing variables at each iteration. A search tree is built and explored to meet the optimal solution. At each node of the tree, a lower bound and an upper bound is computed. If Cplex is stopped before finding an optimal solution, the best feasible solution is considered as an upper bound.

Solving the MILP leads to an off-line solution that takes into account the whole list of constraints. The complexity of the model depends on the number of constraints and variables, especially the integer ones. In our case, the number of constraints and variables are very important, and some of them are binary, which make solving the problem very hard.

6.1.1. Data preparation

The chosen data set is C0 (see Table 4). It is composed of two client orders: (1) one product, BELT and (2) one product, AIP. The objective function chosen is the makespan (i.e., $C_{\max}$), thus due dates are not considered. Using a linear program or another OR

[2] IBM ILOG CPLEX Optimizer, High-performance mathematical optimization engines: http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

Table 7
Star rating of the dynamic scenarios.

	☆☆	★★
#PS1	Scenario is not managed or orders cannot be canceled or restarted.	Orders are canceled. The control system checks the orders to detect possible changes and restarts them.
#PS2	Scenario is not managed or the control system crashes.	The control system detects the new operation and re-optimizes its production with it.
#PS3	Scenario is not managed or the control system crashes.	The control system detects the new machine and re-optimizes its production with it.
#PS4	Scenario is not managed or the control system crashes.	The control system detects the new time and re-optimizes its production with it.
#PS5	Scenario is not managed or the control system crashes.	The control system detects the rush order and minimizes its completion time.
#PS6	Scenario is not managed or the control system crashes.	The control system detects the faulty products and makes them rapidly quit the cell.
#PS7	Scenario is not managed or the control system crashes.	The control system detects maintenance and products are rerouted.
#PS8	Scenario is not managed or the control system crashes.	The control system detects the component out of stock and reallocates the products.
#PS9	Scenario is not managed or the control system crashes.	The control system detects the breakdown and reallocates products to other machines.
#PS10	Scenario is not managed or the control system crashes.	The control system detects the breakdown and products are changed or are in a waiting zone.
#PS11	Scenario is not managed or the control system crashes.	The control system detects the defect and re-optimizes the products at the manual station.
#PS12	Scenario is not managed or the control system crashes.	The control system detects the pattern and uses it at expected event.
#PS13	Scenario is not managed or the control system crashes.	The control system detects the pattern and uses it at expected event.
#PS14	Scenario is not managed or the control system crashes.	The control system checks the state of all the entities and restarts them in their previous state.
#PS15	Scenario is not managed or the control system crashes.	The control system checks the state of all the entities and restarts them in their previous state.
#CS1	The control system is totally crashed and is not able to recover.	The control system is able to see the missing entity, adapts itself by re-optimizing the plan, recovers and can fulfill all of its plans
#CS2	The control system is not able to start over when the communication network becomes available.	The system continues the normal functioning and adjusts its plans, when the communication network becomes available.

Number of:	QLS	Classification
Maximum	17	AAA+
	16	AAA
	15	AAA-
	14	AA+
	13	AA
	12	AA-
	11	A+
	10	A
	9	A-
	8	BBB
	7	BB+
	6	BB
	5	BB-
	4	B+
	3	B
	2	B-
	1	CCC
Minimum	0	CC

Fig. 6. Approach ratings.

approach requires *a priori* knowledge of the whole system behavior. However, if a dynamic scenario is chosen, these approaches are hardly suitable if unexpected events occur.

Thus, no dynamic stage is tested in our study, Liu, Ong, and Ng (2005), Fattahi and Fallahi (2010) and Adibi, Zandieh, and Amiri (2010) can be cited to justify this choice. These authors' approaches are based on pure adaption of the OR methods to the dynamic behavior or on hybridization with artificial intelligence, such as agent systems. The picture of the solution can be taken when the unexpected event occurs and then the non-finished tasks can be rescheduled, taking into account the new behavior (Huang, Lau, Mak, & Liang 2005).

6.1.2. *Experimentation*

The linear programming technique used in Cplex gives an off-line solution for the static stage. Fig. 7 presents a GANTT diagram for scenario #0.

The allocation of operations is well-balanced on different machines, with a little overloading of the machine M_3. This overload is due to the decision of allocating the jobs B, E and P that require many operations of type 1 (Axis) and type 2 (r_comp). Successive Axis and r_comp operations are performed in the same machine when possible. Choosing machine M_3 instead of M_2 in some cases can be explained by the fact that the machine M_2 is not able to perform the L_comp operations, so it is discharged. The makespan obtained with Cplex is the optimal one. It will be used for the robustness evaluation of both simulations and experiments proposed in the rest of this paper.

As previously written in the preparation step, the dynamic stage is not dealt with the MILP program presented above. The resolution time of the MILP for this simple case is long (i.e., more than 1 h). If a perturbation occurs the model has to be re-parameterized and solved again by the MILP. Thus, in a dynamic context, this kind of approach cannot be considered without releasing some constraints, which must be carefully studied otherwise. In a dynamic context, this clearly militates to design for example effective meta-heuristics approaches or more reactive control systems, which is currently under development in our team.

6.1.3. *Reporting*

Table 9 presents the reporting sheet that summarizes this experiment; it only considers the static scenario.

6.2. *Using the benchmark through simulation*

In this section, a simulation tool based on potential fields is proposed for the scheduling and the control of the applied AIP-PRIMECA flexible job-shop. According to the potential field approach, the machines emit attractive fields to attract the shuttles (i.e., jobs) depending on the services they provide, their

Table 8
Example of a reporting sheet.

AIP production cell scheduling and control - Reporting Sheet								
Results from experimentations:		Online / Offline						
Reference scenario definition:								
	Data Set # :	-						
	Use of Due Date:	Yes / No						
	Performance Evaluation:				**Cmax**	**ΣL**	**ΣE**	
	Primary Objective Function:				-	-	-	
	Parameters of the Objective Function:				-	-	-	
	Other Measured Performances:				-	-	-	
	Value of Objective Function:				-	-	-	
Tested dynamic scenarios:					↓	↓	↓	
Scenario #	**Parameters of the dynamic scenario to fix**	**Value of parameters**	**Value of deteriorated functions:**					**Star rating**
#PS1	-	-			-	-	-	-
…	-	-			-	-	-	-
#PS15	-	-			-	-	-	-
#CS1	-	-			-	-	-	-
#CS2	-	-			-	-	-	-

Select objective function	
Minimization	-
Maximization	-

QTS	
Cmax	-
ΣL	-
ΣE	-

QLS
-

Legend:
Cmax – makespan
ΣL - sum of lateness
ΣE - sum of earliness
Inputs
Ouputs

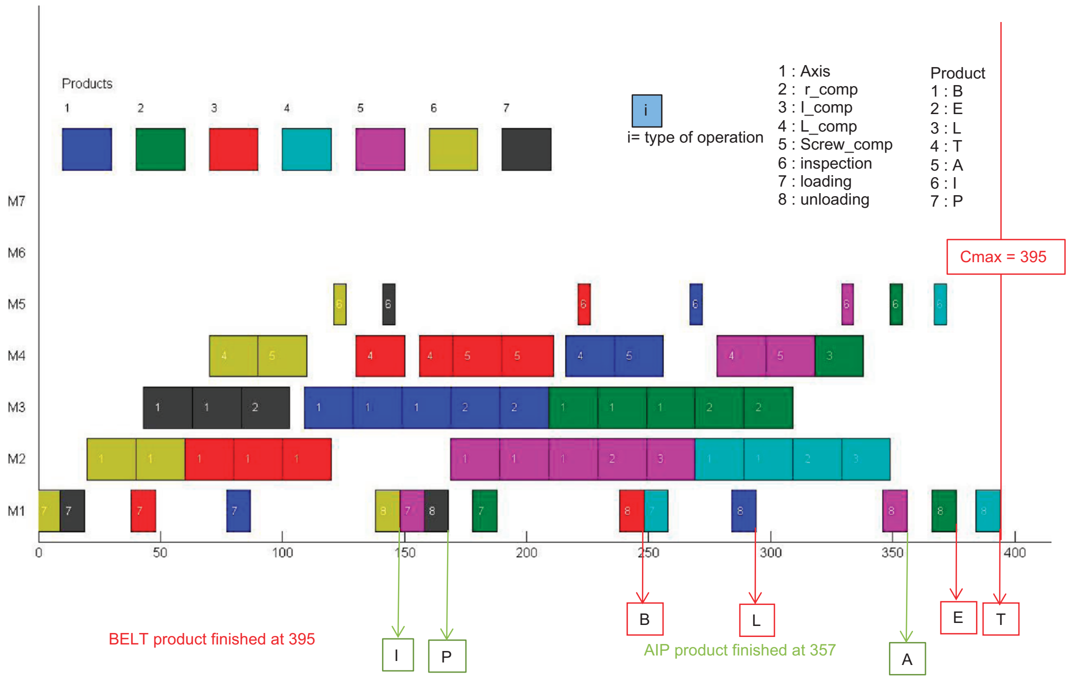


Fig. 7. Gantt diagram for scenario # 0, C0.

queue and their real-time availability. To take transportation times into account, these fields are modified for the distance in which they are sensitive. Shuttles sense the fields through the cell at each node and move to the most attractive one until they reach a service node, and the decisions are made on routing divergent node. Thus, if several machines emit concurrent fields, the shuttle's choice is reduced to the comparison between the different fields, which is a very simple 'max' operation. (For more details on the conceptual and application components, interested readers can consult Zbib, Pach, Sallez, and Trentesaux (2012).

Since the potential field approach is naturally distributed, with no centralized information storage or information processing, the NetLogo multi-agent environment (Wilensky, 1999) was used for the simulation. NetLogo provides appropriate tools for modeling entities and their interactions. The NetLogo environment is also user-friendly and provides a well-documented programming language with a smart graphic interface. These characteristics make NetLogo an effective tool for the rapid prototyping of reactive multi-agent systems.

A NetLogo simulator was designed for the production system, which handles all the previously introduced constraints of the AIP cell, including transportation times and limited capacity of storage areas. A control system based on potential fields was implemented as the production system's rule behavior, which dealt with all the control decisions made by the products. Fig. 8 shows a screenshot of the NetLogo simulator for this case study.

6.2.1. *Data preparation*

To illustrate the ability of the benchmark to facilitate the comparison of various scheduling & control approaches, the same data set has been chosen (C0) and the same objective function is used ($C_{\max}$).

Specifically to this study, the queue in front of machines is limited to one unit. When no machine is available for a requested operation and corresponding queues are full, shuttles turn into a waiting loop. We studied 3 scenarios. The first was a static scenario #0; then we studied two dynamic scenarios considering unexpected events: scenario #PS9, featuring a redundant machine breakdown, and scenario #PS11, featuring a quality problem.

6.2.2. *Experimentation*

a. Static stage
   Due to the highly reactive behavior of the potential field approach, all decisions made at the control level by shuttles are “in the loop” with the production system's simulation. Thus, the whole system can be considered as an emulated on-line control. Fig. 9 presents a GANTT diagram, showing the arrangement of operations for each job making up the BELT and AIP products.
   Operations 1 (Axis) and 2 (r_comp) are well-balanced between machines M_2 and M_3. Operation 4 (L_comp) is well-balanced between machines M_4 and M_2. Operation 5 (Screw_comp), which can be processed on M_3 or M_4, is always carried out on M_4 because it always follows operation 3 (I_comp), which is only provided by M_4. Loading and unloading are only performed by M_1, forcing shuttles to queue. Operation 6 (inspection) is only provided by M_5 and is compulsory for all jobs. The obtained $C_{\max}$ is 448 s. Compared to the $C_{\max}$ given by the MILP program, the results are different due to another launching sequence for jobs. The use of the same sequence gives the same results as in the MILP program which seems to be a very interesting fact to consider for future research.
b. Dynamic stage
   Since the control is on-line with the simulation of the behavior of the controlled production system, dynamic scenarios can be handled by the NetLogo simulator. For illustrating the benchmark, two scenarios are presented. The way the control layer behaves faced with unexpected events is highlighted.
   *Scenario #PS*9
   In this dynamic scenario, a perturbation occurs on M_2, which is a redundant machine. From a control perspective, if that situation occurs, since queue capacity is limited to 1, the shuttles are directed to a waiting loop to wait for free place in the parallel machines. The breakdown must start when the first shuttle leaves M_2, after processing 5 operations on job B, at 122. It must end at 297 (122+25 $\times$ number of jobs).
   When a perturbation occurred on M_2, one shuttle carrying job L was queued in front of M_2, waiting for operation 1 (Axis). The occurrence of the breakdown on M_2 forced this shuttle to leave the M_2 queue to turn into the waiting loop, waiting for a place in the M_3 queue, currently filled with a shuttle loaded with the T job. When the breakdown was finished, the shuttle loaded with job 7 (P) was in the waiting loop, waiting for a place in the M_3 queue to process operation 1 (Axis). Using potential field approach, the availability of M_2 was immediately considered, and the shuttle carrying job P turned in M_2.
   Compared to the Gantt diagram of the static reference scenario #0, operation 1 (Axis) and 2 (r_comp) were dynamically transferred to machine M_3, while operation 4 (L_comp) was transferred to M_4. In addition, the $C_{\max}$ was increased from 448 to 491. The Gantt diagram of this scenario is given in Fig. 10.
   *Scenario #PS*11
   This scenario considers quality problems during the production: 50% of inspected jobs on M_5 are re-routed to M_6 for a recovery operation. The Gantt diagram in Fig. 11 shows that jobs 3, 4, 5, and 6 (L,T, A and I) were recovered and were directed to M_6 machine after being inspected instead of being directed to M_1. Compared to the static reference scenario #0, $C_{\max}$ has increased from 448 to 549.

6.2.3. *Reporting*

In this paper, two dynamic scenarios (#PS9 and #PS11) are studied with the NetLogo simulator. Perturbations are taken into account in both cases, and the system reorganizes itself to optimize the global performance. Thus two stars are given to each scenario.

$$QLS = \max(2^*, \tfrac{1}{2}1^*) = \max(2, \tfrac{1}{2}0) = 2$$

In the light of this result, the corresponding qualitative classification is B- (Fig. 6).

The final rating is calculated with the following formula, with the objective being the minimization of $C_{\max}$:

$$\begin{aligned} QTS = & \sum_{0<i\le 15} g_i \left(\begin{cases} e^{(Op_{ref}-Op_i/Op_{ref})}, \text{ if } Op_i > Op_{ref} \\ 1, \text{ otherwise} \end{cases} \right) \\ & * \left(\begin{cases} e^{(Op_{best}-Op_i/Op_{best})}, \text{ if } Op_i > Op_{best} \\ 1, \text{ otherwise} \end{cases} \right) \\ & = Scoresc9 + Scoresc11 \\ & = 2e^{(448-491/491)}e^{(395-491/574910)} + 2\,e^{(448-549/549)}\,e^{(395-549/549)} = 2.51 \end{aligned}$$

The global QTS is 2.51/30 for this approach. This rating is poor because only two scenarios are tested in this example.

Table 10 presents the reporting sheet that summarizes all these experiments.

This first study realized on the proposed benchmark enable us to compare MILP and potential field approaches.

The difference between the makespan obtained in this experiment and the optimal one obtained using the MILP is only 13% (448 vs. 395). In addition, the ability to react to perturbations allows the potential field approach to manage both scenarios PS9 and PS11, which are some of the most complicated scenarios. Thus, a preliminary conclusion of this benchmark is that potential fields seem to be a very promising approach. Of course, this study must be pursued with theoretical studies for eliciting the properties and testing this benchmark more profoundly, with other scenarios than C0 and other dynamic scenarios to improve QLS and QTS values.

Using approaches such as MILP can lead to data model lock-in and may introduce severe limitations, forbidding its adaptation to other type of controlled production systems and rendering the developments highly dependent of the target systems. In approaches such as potential fields, only data models for the elementary exchanges are fixed; the interactions are completed

Table 9
Reporting sheet for a linear program

AIP production cell scheduling and control - Reporting Sheet					
Results from experimentations:		**Offline**			
Reference scenario definition:					
	Data Set # :	**C0**			
	Use of Due Date:	**No**			
	Performance Evaluation:			**Cmax**	
	Primary Objective Function:			**X**	
	Parameters of the Objective Function:				
	Other Measured Performances:				
	Value of Objective Function:			**395**	
Tested dynamic scenarios:				↓	
Scenario #	**Parameters of the dynamic scenario to fix**	**Value of parameters**	**Value of deteriorated functions:**		**Star rating**
#PS1					
…					
#PS15					
#CS1					
#CS2					

Select objective function	
Minimization	X
Maximization	-

QTS	
Cmax	-
QLS	
-	

Legend:
Cmax – makespan
ΣL - sum of lateness
ΣE - sum of earliness
🟨 Inputs
🟧 Ouputs

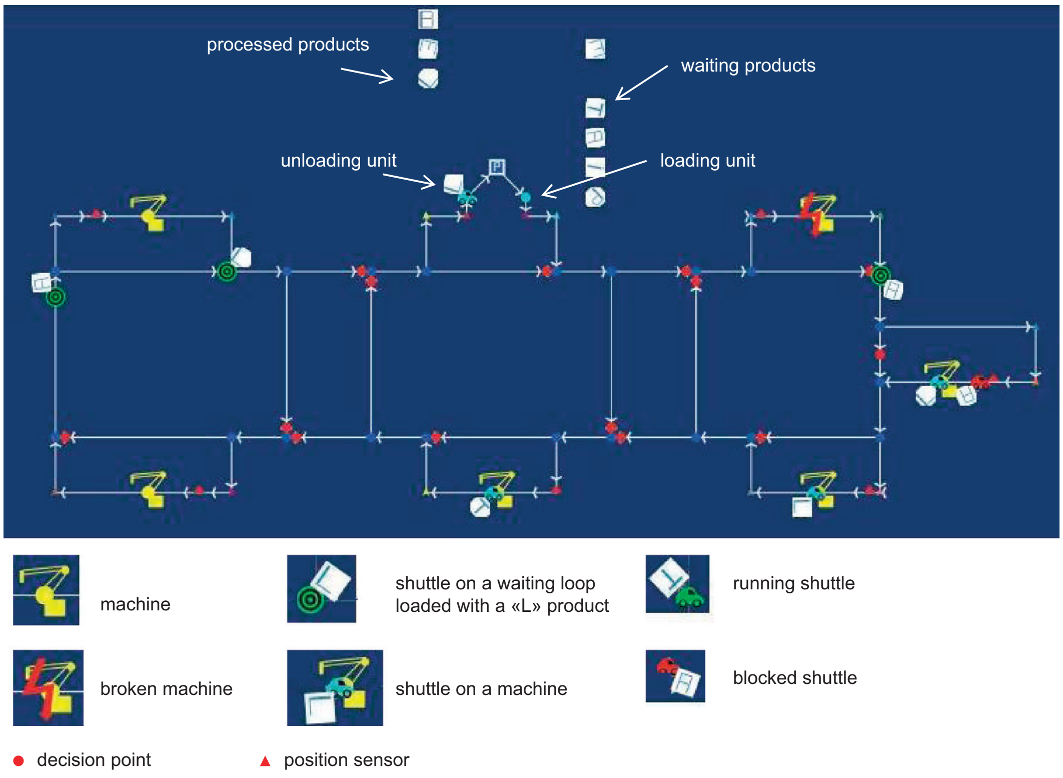


Fig. 8. Model of the AIP cell in NetLogo simulator.

at a low level in a highly dynamic way. Thus, the data model lock-in is limited. A high level of reactivity and adaptability to other controlled system is possible since the assumptions simplifying the behavioral model of this system are limited.

6.3. *Using the benchmark through a real experiment*

In this illustration, the experiments concern both the static and dynamic stages and were made on-line using the real AIP-

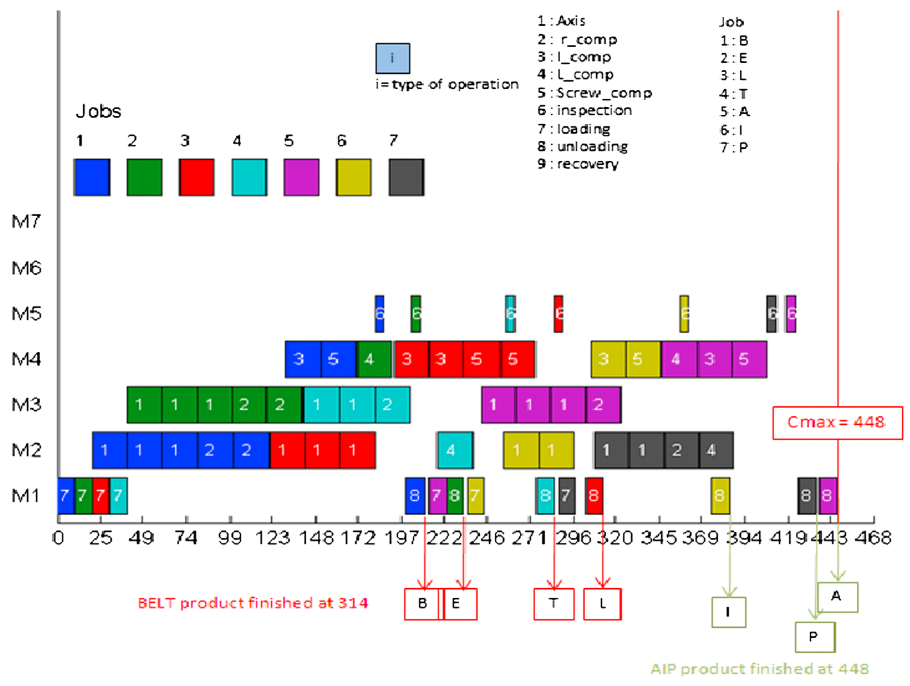


Fig. 9. Gantt diagram for static scenario #0.

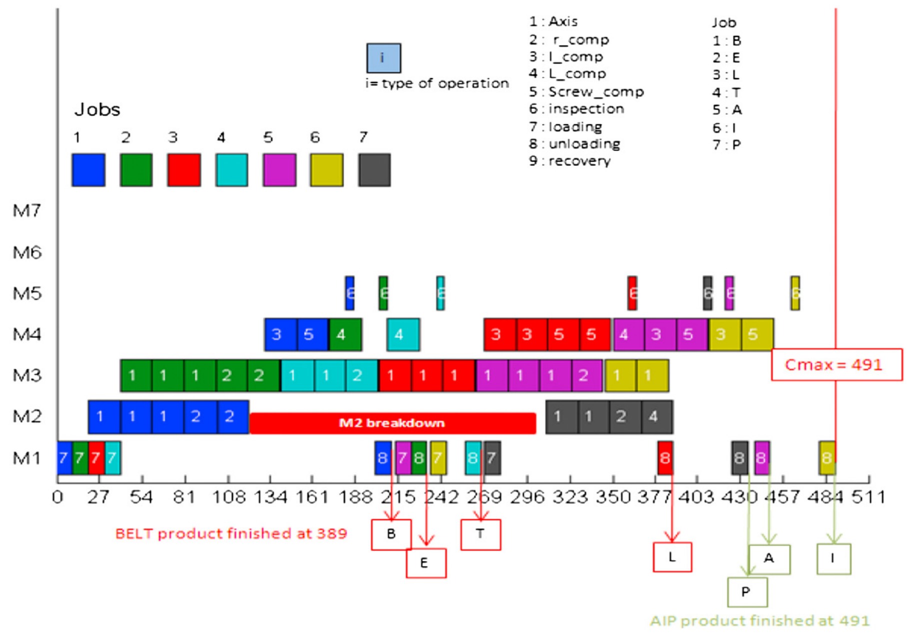


Fig. 10. Gantt diagram for the dynamic scenario #PS9.

PRIMECA production cell. No simulator was used. Since the potential field approach seems to be very promising, the control is implemented using this approach.

6.3.1. *Data preparation*

The data set chosen is still C0 with the objective function being $C_{\max}$. In this section, the dynamic scenarios tested are scenarios #PS7 and #PS9.

6.3.2. *Experimentation*

a. Static stage
In this static stage, all scheduling decisions are made on-line, according to a real-time control approach since the real cell is used. No preliminary off-line stage is used, which can be considered as a purely reactive scheduling approach (Davenport and Beck, 2000). Fig. 12 presents a Gantt diagram, in which the tasks correspond to the different manufacturing

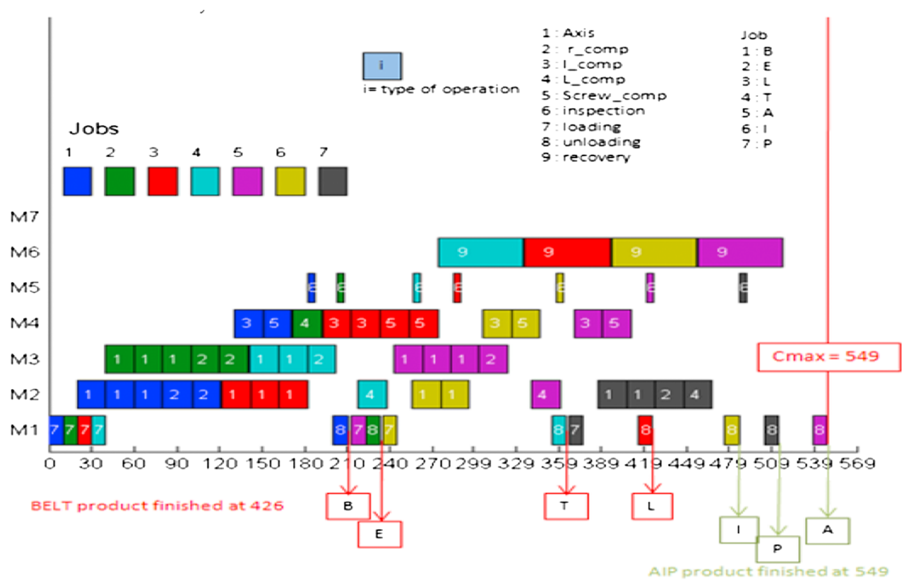


Fig. 11. Gantt diagram for dynamic scenario #PS11.

orders, highlighting the due dates of all the jobs (A, I, P, B, E, L and T) and of the products (AIP and BELT).

Transportation times, limited capacity of storage area and limited number of shuttles were handled in a very simple natural way by the potential field approach. The load on M_2–M_3 is well-balanced from the start, since these machines propose the "Axis" operation that occurs several times in every job. This illustrates the capability of the control approach to balance the workload.

The machine M_4 is the only one to provide the "l_comp", and M_2 and M_3 provide the "L_comp" and "Screw_comp", respectively. Then, most of the jobs must pass by M_4. All of the shuttles have to go through M_5 for the inspection and M_1 to load and unload their products. The C_{max} obtained is 528. The differences with the simulation are due to real environment experimentations. For example, one Programmable Logic Controller (PLC) manages one machine. Each PLC is responsible for the assembly operation start and stop, but also for the launching of the shuttle to the next node. When a shuttle leaves the machine, the PLC waits that it reaches the next node before beginning the operation on the next shuttle for safety purpose. This mechanism can be seen in the GANTT diagram: there always exist a short production delay between two successive jobs on the same machine.

b. Dynamic stage

The dynamic scenarios are still #PS7 and #PS9 for comparison purpose. Obviously, the dynamic events were not previously known by the potential-field control system. They were artificially integrated in real time in the production cell.

Scenario #PS7

According to scenario #PS7, the section of the conveyor in maintenance is the section between the nodes n_9 and n_2. This was done by an artificial increase of the corresponding transportation time (Table 4), fixing the value to a greater number than 5 s. The maintenance team sets this new value, which will lead to a drastic reduction in potential fields emitted in this path, does the maintenance operation, and then resets the initial value. Thus, no direct intervention on the control system is done.

Without any human intervention on the control system, the A and I jobs took a secondary path (n_9–n_{10}–n_{11}–n_1–n_2). The Gantt diagram is given in Fig. 13.

According to Table 5, the maintenance began after the end of the fourth job, which occurs in the experiments at t=369 s. Without the perturbation, it took almost 18 s to go to the machine M_1 from machine M_5. Please compare Fig. 12 (the Gantt diagram of scenario #0) with Fig. 13 (Gantt diagram for the scenario #PS7). To highlight the difference with the reference scenario #0, the arrival times at M_1 without perturbation are represented by the dashed lines in Fig. 13. During the maintenance operation, the jobs took the alternative path and reach M_1 in almost 35 s from M_5. Despite this delay, the system is not blocked, and the jobs are still produced during this maintenance operation. The C_{max} is now 548 s. If the control system had to wait until the end of the maintenance operation, it could have restarted at time t=544 and could have finished at t=723.*Scenario #PS9*

According to the scenario #PS9, the machine M_2 breaks down when the job 1 leaves it, which occurs at 133 s. According to this scenario, M_2 stays unavailable until the time 308 s (133 s. $+7\times25$ s.). This was done by shutting down the emitted potential field of this machine to zero. Fig. 14 presents the results.

Like in the simulation, this perturbation was successfully handled by the potential-field control system in the real situation. Job 3, which was in M_2 queue in scenario #0, had identified the breakdown and decided to go M_3 to continue its production. Since there were already the jobs 2 and 4 at M_3, it had to wait. When M_2 reappeared as available, using the potential field approach, the jobs instantly re-detected and reassigned to it. Thus, job 6 came to this newly available machine (M_2) to get its "Axis_comp". Job 5 chose M_3 so the balance between M_2 and M_3 is reestablished. The C_{max} with this breakdown of 175 s is 589 so the consequences of the breakdown are slight in terms of delay, compared to the C_{max} obtained in scenario #0.

6.3.3. Reporting

The reporting sheet that summarizes all these experiments is given in Table 11, pointing out the deterioration of the chosen objective functions (scenarios #PS7 and #PS9 vs. Scenario #0).

In the static stage, the difference between the obtained C_{max} in simulation (4 4 8) and in the corresponding real experiment (5 2 8) is 17%, which is acceptable, since in real situations, there

Table 10
Reporting sheet for simulation scenarios

AIP production cell scheduling and control - Reporting Sheet

Results from experimentations:		**Online**			
Reference scenario definition:					
	Data Set # :	**C0**			
	Use of Due Date:	**No**			
	Performance Evaluation:			**Cmax**	
	Primary Objective Function:			**X**	
	Parameters of the Objective Function:				
	Other Measured Performances:				
	Value of Objective Function:			**448**	
Tested dynamic scenarios:				↓	
Scenario #	**Parameters of the dynamic scenario to fix**	**Value of parameters**	**Value of deteriorated functions:**		**Star rating**
#PS9	Machine: Start Time: Duration:	**M2** **122 seconds** **7*25 = 175 seconds**		**491**	**2**
#PS11	Frequency of the quality problem:	**50% of jobs**		**549**	**2**

Select objective function	
Minimization	**X**
Maximization	-

QTS	
Cmax	**2,51**

QLS
B -

Legend:
Cmax – makespan
ΣL - sum of lateness
ΣE - sum of earliness
🟨 Inputs
🟥 Ouputs

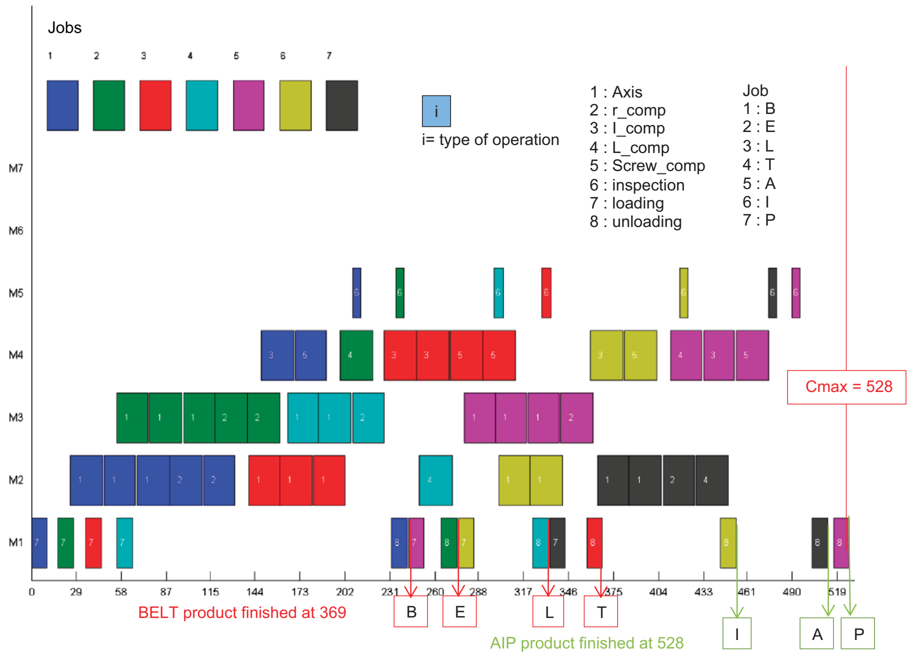


Fig. 12. Gantt diagram for the static stage—scenario #0.

are always some delays in communication, transportation (eg., the shuttle speed is not constant) and production, which are ignored.

In the dynamic stage (scenario #PS9), the experimental system behaves in the same way as in the simulation: the deterioration of the C_{max} (from 491 to 589) is at the same order of amplitude (19%) in the static stage. The QTS values are also close each other, which seem to confirm the similarities of the control simulation and the experimental one, validating the simulator at the same time.

Since this potential field approach is purely reactive, it handles the static or dynamic data easily for the real-time events. All these experiments have shown that it can gain from being interfaced with an off-line mechanism for optimization purposes, which corresponds to one of our current research objective.

7. Conclusion and future works

This paper has presented a fully operational benchmark intended to be used not only from an OR perspective (i.e., use of formal linear models or metaheuristics), but also from a control

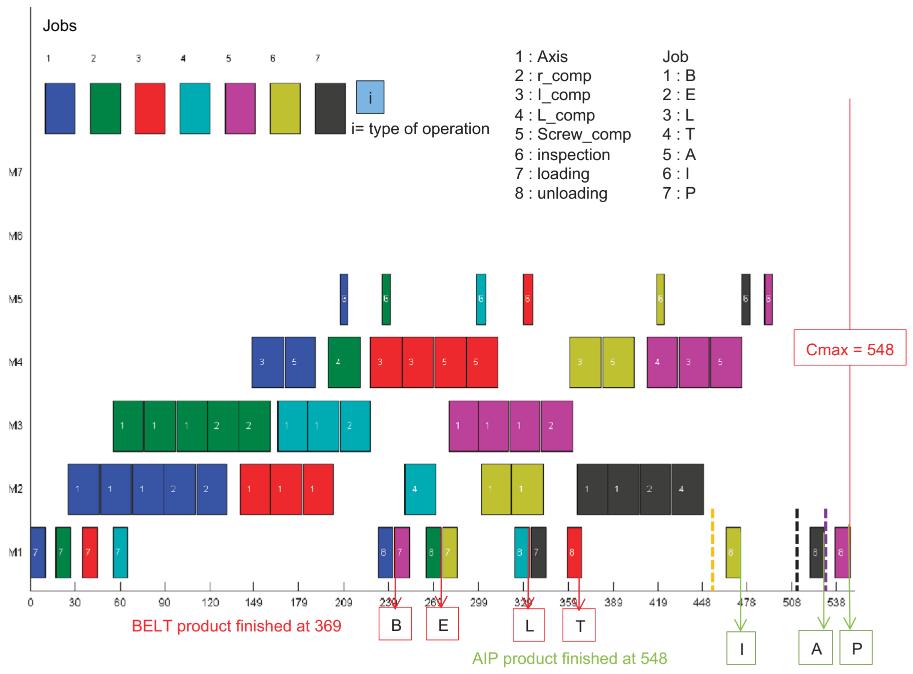


Fig. 13. Gantt diagram for the scenario #PS7.

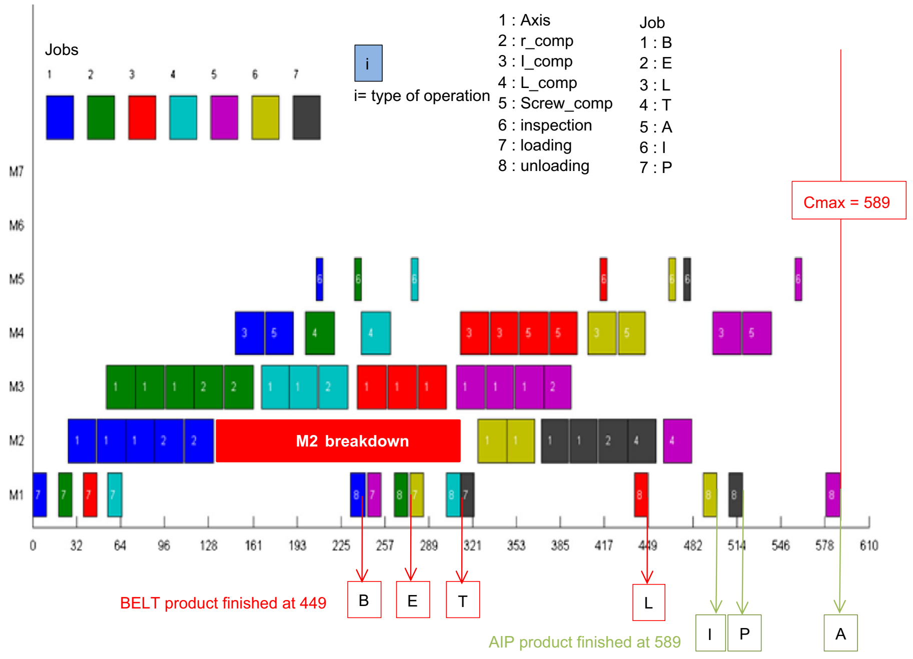


Fig. 14. Gantt diagram for scenario #PS9.

perspective (i.e., on-line dynamic control of a simulated or real target system). One of the authors' desire is to bridge the gap between the OR and control communities to allow cross-pollination. This could lead to the design of new innovative hybrid approaches, typically by coupling in a more efficient way predictive mathematical models with advanced simulation tools and reactive control-based models. For example, with the suggested benchmark, it would be possible, in the dynamic stage, to evaluate iteratively several possible response behaviors on a simulated version of the controlled system using the proposed scenarios. The results can be stored in a database. Generic decision rules (e.g., “when this kind of perturbation occurs, then react in this way”) may then be designed. When moving on the real system, still in the dynamic stage, the control system can access this database. It can detect similar situations in real time when a real perturbation occurs. Then, it chooses the “best” corresponding anticipated behavior to be applied. The database can also be updated with new generated knowledge from real experimentations and updated real behaviors from experiments. Metaheuristics, data mining, reinforcement learning and clustering technics, as well as

Table 11
Reporting sheet for experimental scenarios

AIP production cell scheduling and control - Reporting Sheet					
Results from experimentations:		**Online**			
Reference scenario definition:					
	Data Set # :	**C0**			
	Use of Due Date:	**No**			
	Performance Evaluation:			**Cmax**	
	Primary Objective Function:			**X**	
	Parameters of the Objective Function:				
	Other Measured Performances:				
	Value of Objective Function:			**528**	
Tested dynamic scenarios:				↓	
Scenario #	**Parameters of the dynamic scenario to fix**	**Value of parameters**	**Value of deteriorated functions:**		**Star rating**
#PS7	Start Time: Location: Duration:	**369 seconds** **N2-N9 conveyor part** **7*25 = 175 seconds**		**548**	**2**
#PS9	Machine: Start Time: Duration:	**M2** **133 seconds** **7*25 = 175 seconds**		**589**	**2**

Select objective function	
Minimization	X
Maximization	-

QTS	
Cmax	**2,40**

QLS
B -

Legend:
Cmax – makespan
ΣL - sum of lateness
ΣE - sum of earliness
Inputs
Ouputs

simulation, multicriteria analysis, design of experiments and multi-agent control architectures could be each useful in such as context.

More, our benchmark, based on real full-size production cell, is clearly grounded in reality. Supporting benchmarking on a physical, real-world system is an essential activity for researchers. Without the "roots into reality", the researchers may fail to develop proper answers to the needs of the industry.

Aiming for a wide adoption by the OR and control communities, the following issues will be taken into account in the near future:

- A website[3] has been designed to capitalize results, tools and contributions. It will be updated progressively with results and improved with ongoing extensions of the benchmark presented in this paper.
- A remote connection to the real flexible manufacturing cell will be enabled for the researchers who would like to validate their control algorithm and evaluate the robustness of their results.
- Global energy consumption or global cost for a given production scenario will be proposed to integrate the other efficiency measures (not addressed in this paper).

From a control perspective, one interesting prospective for future research is related to the design of dedicated simulation tools of the AIP-PRIMECA production cell's behavior, to be provided to the Control community in complement to this benchmark. The use of the platform Emulica (Pannequin et al., 2009), presented in the survey part of this paper, is a very interesting open simulation environment. This should facilitate future comparisons of control strategies since it would not force researchers to develop a new simulator each time a new contribution is under consideration, leading them to deploy their effort rather on control strategy instead of the simulation of the operating system.

Last, other application areas of our benchmark will be considered: healthcare systems, logistics systems or transportation systems.

Acknowledgment

The authors wish to sincerely acknowledge their debt to the anonymous reviewers for their highly accurate comments and their constructive help in this article's improvement.

References

Adibi, M. A., Zandieh, M., & Amiri, M. (2010). Multi-objective scheduling of dynamic job shop using variable neighborhood search. *Expert Systems with Applications*, *37*(1), 282–287.

Azardoost, B. E., & Imanipour, N. (2011). A hybrid algorithm for multi objective flexible job shop scheduling problem. In *Proceedings of the 2011 international conference on industrial engineering and operations management*. Kuala Lumpur, Malaysia, January 22–24.

Beasley, J. E. (1990). OR-library: distributing test problems by electronic mail. *Journal of the Operational Research Society*, *41*(11), 1069–1072.

Bixby, R., Ceria, S., McZeal, C. M., & Savelsbergh, M. W. P. (1998). An updated mixed integer programming library: MIPLIB 3.0. *Optima*, *58*, 12–15.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*, *41*(3), 157–183.

Brennan, R. W., & O, W. (2002). Evaluating alternative manufacturing control strategies using a benchmark system. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *216*, 927–932.

Cavalieri, S., Macchi, M., & Valckenaers, P. (2003). Benchmarking the performance of manufacturing control systems: design principles for a web-based simulated testbed. *Journal of Intelligent Manufacturing*, *14*(1), 43–58.

Cho, S., & Lazaro, A. (2010). Control theoretic model using PID controller for just-in-time production scheduling. *International Journal of Advanced Manufacturing Technology*, *51*, 699–709, http://dx.doi.org/10.1007/s00170-010-2639-x.

[3] http://www.univ-valenciennes.fr/bench4star/.

Conway, R. W., Maxwell, W. L., & Miller, L. W. (1967). *Theory of scheduling*. Massachussetts: Addison-Wesley Publishing Company Reading.

Davenport, A. J., & Beck, J. C. (2000). *A survey of techniques for scheduling with uncertainty*. *Technical report*. .

Demirkol, E., Mehta, S., & Uzsoy, R. (1998). Benchmarks for shop scheduling problems. *European Journal of Operational Research*, *109*(1), 137–141.

Fattahi, P., & Fallahi, A. (2010). Dynamic scheduling in flexible job shop systems by considering simultaneously efficiency and stability. *CIRP Journal of Manufacturing Science and Technology*, *2*(2), 114–123.

Gordon, V., Proth, J.-M., & Chu, C. (2002). *A survey of the state-of-the-art of common due date assignment and scheduling research European Journal of Operational Research*, *139*, 1–25.

Huang, G. Q., Lau, S. K., Mak, K. L., & Liang, L. (2005). Distributed supply-chain project rescheduling: Part I: Impacts of information-sharing strategies. *International Journal of Production Research*, 1–23.

Ho, N. B., & Tay, J. C. (2008). Solving multiple-objective flexible job shop problems by evolution and local search. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, *38*(5), 674–685.

Kolisch, R., & Sprecher, A. (1996). PSPLIB—a project scheduling problem library. *European Journal of Operational Research*, *96*, 205–216.

Liu, S. Q., Ong, H. L., & Ng, K. N. (2005). Metaheuristics for minimizing the makespan of the dynamic shop scheduling problem. *Advances in Engineering Software*, *36* (3), 199–205.

Mönch, L. (2007). Simulation-based benchmarking of production control schemes for complex manufacturing systems. *Control Engineering Practice*, *15*, 1381–1393.

Montech Technology (2009). 〈http://www.montech.com〉.

Pannequin, R, Morel, G., & Thomas, A. (2009). The performance of product-driven manufacturing control: An emulation-based benchmarking study. *Computers in Industry*, *60*(3).

Reinelt, G. (1991). TSPLIB—a traveling salesman problem library. *ORSA Journal on Computing*, *3*, 376–384.

Sourd, F. (2005). Earliness–tardiness scheduling with setup considerations. *Computers and Operations Research*, *32*, 1849–1865.

Taboun, S. M., & Ulger, T. (1992). Multi-objective modelling of operation-allocation problem in flexible manufacturing systems. *Computers and Industrial Engineering*, *23*(1–4), 295–299.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, *64*(2), 278–285.

Thiagarajan, S., & Rajendran, C. (2005). Scheduling in dynamic assembly job-shops to minimize the sum of weighted earliness weighted tardiness and weighted flowtime of jobs. *Computers and Industrial Engineering*, *49*, 463–503.

Valckenaers, P., Cavalieri, S., Saint Germain, B., Verstraete, P., Hadeli, R., Bandinelli, S., Terzi, H., & Brussel, Van (2006). A benchmarking service for the manufacturing control research community. *Journal of Intelligent Manufacturing*, *17*(6), 667–679.

Wilensky, U. (1999). *Center for connected learning and computer-based modeling*. Evanston, IL: Northwestern University. 〈http://ccl.northwestern.edu/netlogo/〉.

Zbib, N., Pach, C., Sallez, Y., & Trentesaux, D. (2012). Heterarchical production control in manufacturing systems using the potential fields concept. *Journal of Intelligent Manufacturing*, *23*(5), 1649–1670.