Expert Systems with Applications 41 (2014) 2286–2299

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Integrated scheduling of resource-constrained flexible manufacturing systems using constraint programming

Juan M. Novas [1], Gabriela P. Henning *

INTEC (UNL-CONICET), Güemes 3450, S3000GLN Santa Fe, Argentina

ARTICLE INFO

Keywords:
Flexible manufacturing systems
Resource-constrained scheduling
Constraint-programming
Automated job-shop

ABSTRACT

This contribution presents a novel approach to address the scheduling of resource-constrained flexible manufacturing systems (FMSs). It deals with several critical features that are present in many FMS environments in an integrated way. The proposal consists in a constraint programming (CP) formulation that simultaneously takes into account the following sub-problems: (i) machine loading, (ii) manufacturing activities scheduling, (iii) part routing, (iv) machine buffer scheduling, (v) tool planning and allocation, and (vi) AGV scheduling, considering both the loaded and the empty movements of the device. Before introducing the model, this work points out the problems that might appear when all these issues are not concurrently taken into account. Then, the FMS scheduling model is presented and later assessed through several case-studies. The proposed CP approach has been tested by resorting to problems that consider dissimilar number of parts, operations per part, and tool copies, as well as different AGV speeds. The various examples demonstrate the importance of having an integrated formulation and show the important errors that can occur when critical issues such as AGV empty movements are neglected.

1. Introduction

Flexible Manufacturing Systems (FMSs) are highly automated production systems, consisting of a computer-controlled integrated configuration of multipurpose workstations, storage buffers and one or more automated guided vehicles (AGVs). These manufacturing environments combine an important productivity with high levels of flexibility and an efficient use of limited resources, characteristics that are required to remain competitive in current markets. To increase the efficiency of the overall FMS, manufacturing activities, as well as transport and storage tasks, need to be properly scheduled. The FMS scheduling activity is affected by many features, such as the specific characteristics of the FMS, the plant in which it is located and its operational policies, the level of automation, as well as the resources belonging to the FMS (Grieco, Semeraro, & Tolio, 2001). The development of good quality schedules that consider all the FMS constrained resources, such as machines, AGVs, tools, buffers, is one of the main operational problems to be tackled in this kind of environment (Blazewicz, Eiselt, Finke, Laporte, & Weglarz, 1991).

FMS scheduling comprises the following problem elements: machine loading, part routing, manufacturing tasks scheduling, tool planning and allocation, as well as the generation of the buffers usage agenda and the AGVs schedule. The FMS loading problem is concerned with the assignment of manufacturing operations to machines, considering resource and technological constraints. Part routing determines manufacturing routes for parts, specifying the sequence of machines that each part visits throughout the system in order to be processed. Manufacturing tasks scheduling defines the start, duration and end times of each machining activity. Tool planning specifies the number of tool instances of each available type that are needed to achieve the production requirements, and the tool allocation problem tackles the tool assignment to the magazines of the various machines. Finally, buffers and AGVs scheduling specify the agenda of the buffers and the transport devices, respectively.

FMS scheduling problems have been extensively addressed during last decades. In order to reduce their complexity, researchers have usually resorted to decomposition approaches, not taking into account all the limiting resources at the same time and/or neglecting some others. Within the vast literature concerning FMS scheduling, there is a set of contributions that considers the tool-related limitations as the most important constraints, leaving aside the transportation issues, and another group that takes into consideration the AGVs as the main limiting resource, neglecting tool aspects. Thus, the literature review presented in this work organizes previous contributions in two main groups; first, the ones that address the job scheduling and tool allocation problems, and then, those that deal with the scheduling of vehicles and jobs in the FMS environment.

* Corresponding author. Tel.: +54 342 455 9175x2102; fax: +54 342 455 0944.
E-mail addresses: mnovas@intec.unl.edu.ar, jmnovas@famaf.unc.edu.ar (J.M. Novas), ghenning@intec.unl.edu.ar, gphenning@gmail.com (G.P. Henning).
[1] Permanent address: CIEM-CONICET/UTN-FRC, Ciudad Universitaria, X5000HUA Córdoba, Argentina.

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.09.026

Most of the papers belonging to the first group, which tackle the loading machine and tool planning/allocation problems, were published in the last two decades. Atmani and Lashkari (1998) developed a mixed integer linear programming (MILP) model that addresses the FMS loading and tool allocation problem. The formulation takes into account constraints on tool magazines capacity and tools life-time, but does not consider the number of available tool instances. Another drawback concerns the large MILP formulations that are obtained.

Gamila and Motavalli (2003) proposed an approach to address the FMS machine loading, tool allocation and part scheduling problems, which consists of two steps. First, an MILP formulation solves the machine loading and tool allocation problems. Afterwards, a simple heuristic tackles the detailed part scheduling problem. Constraints regarding tool life-time and tool magazines of limited capacity are taken into account. Besides, it is assumed that only one copy per tool type is available, which is not always true in real settings.

Chan and Swarnkar (2006) presented a fuzzy goal programming approach to tackle the machine loading and tool selection problems, as well as the operations scheduling. They included constraints to consider tool magazine capacity, tool life-time, and machine resources as limiting features. The approach assumes that each tool magazine cannot hold more than a single copy of each tool type, which hampers its use in real settings.

Zeballos, Quiroga, and Henning (2010) proposed a constraint programming (CP) formulation that simultaneously considers machine loading, part routing, tool allocation and operation scheduling in FMS environments. It employs two different two-index variables in order to model machining activities, instead of a four index one. This feature considerably reduces the dimensionality of the approach and facilitates the modeling of machine and tool specific constraints. Furthermore, the proposal represents tool management features in terms of tool types. Indeed, the tool instances demand is calculated indirectly, based on tool type, tool life-time, and tool magazine constraints. Despite the fact that the work by Zeballos et al. (2010) is, to the best of our knowledge, one of the most complete contributions regarding tool loading and allocation issues, it has several shortcomings: it assumes that every part requires the same number of operations and ignores all part intermediate storage and transportation features. Simultaneously, Zeballos (2010) emphasized other aspects of the previous CP-based approach, presenting the search strategy that was used to reduce the computational time.

Regarding the second group of contributions, it is worth noticing that although there are many works on the AGV scheduling problem, there are few contributions on the simultaneous scheduling of AGVs and manufacturing activities (Ganesharajah, Hall, & Sriskandarajah, 1998; Vis, 2006). One of the first attempts was made by Blazewicz et al. (1991), who were motivated by an actual FMS environment. To address the machine scheduling and vehicle routing problems, two situations were considered. In the first one, the assignment of jobs to machines is assumed to be known and the goal is to find a feasible vehicle schedule. The second one aims at finding a solution by simultaneously taking into account the assignment of operations to machines and the vehicle routing problem. For the former sub-problem, a simple polynomial-time algorithm was developed, whereas for the later a pseudopolynomial-time algorithm based on dynamic programming was proposed. This algorithm obtains a minimum length schedule with its corresponding feasible AGV schedule. The main shortcoming consists in considering single-operation parts and identical parallel machines. Besides, the proposal was only tested with small size problems.

Bilge and Ulusoy (1995) developed an iterative procedure to address the scheduling of jobs and vehicles. The problem was decomposed into two sub-problems, which were iteratively solved by means of two algorithms. First, machine schedules are generated by means of a set of dispatching rules. Afterwards, for each machine agenda, a feasible AGV schedule is obtained by means of a heuristic based on a sliding time window. In a later work, Ulusoy, Sivrikaya-Serfioglu, and Bilge (1997) presented a genetic algorithm (GA) to simultaneously address the scheduling of machine jobs and automated transport vehicles. In 59% of the test examples, the GA proposal outperformed the solutions reported by Bilge and Ulusoy (1995). The reverse was true for only 6% of the problems.

Liu and MacCarthy (1997) developed an MILP-based model with the aim of addressing storage and AGV related aspects. Its main shortcomings are its size and complexity, even for small problems. Despite these features, the analysis of the formulation provided the basis for the development of two heuristic algorithms, named "loading then sequencing" and "global heuristic procedure", which are also presented in the paper. It was shown that the iterative global heuristic procedure is much more effective than the "loading then sequencing" one.

In recent years, El Khayat, Langevin, and Riopel (2006) developed a mathematical programming model and a constraint programming formulation to tackle the integrated scheduling of production and material handling activities. The two approaches do not cope with the machine loading problem; i.e. it is assumed that jobs have predefined routes. In addition, they consider that machine buffers have unlimited storage capacity. The mathematical and the CP approaches were tested and compared by means of examples proposed by Bilge and Ulusoy (1995), considering two and three vehicles as limiting resources. The CP approach rendered better results in terms of makespan minimization and computational times.

Jerald, Asokan, Saravanan, and Rani (2006) addressed the concurrent scheduling of parts and AGVs using an adaptive GA-based approach. In the environment being considered machines are grouped into cells, which are connected by means of two AGVs. The approach does not cope with the machine loading problem, since part routes are already fixed. Thus, transport times are also prescribed beforehand and considered as part of the processing times. However, it is not clear how deadlocks are prevented. These assumptions make the proposal of little practical use. More recently, Caumond, Lacomme, Moukrim, and Tchernev (2009) presented an MILP formulation that considers one automated vehicle and adopts a "first in first out" (FIFO) buffer management rule. The proposal considers limited input/output machine buffers, but it does not address the part routing problem since each operation can be executed only by a single machine, which is already defined. The proposal does not ensure that a part waiting in the input buffer of a machine will start its operation as soon as the unit becomes idle. Small problems are successfully solved by the MILP formulation, while a heuristic is proposed for larger ones.

As it was previously pointed out, due to simplicity reasons, decomposition-based approaches do not simultaneously consider all the FMS scheduling elements. Some proposals tackle loading and sequencing problems assuming there are no other critical features to address (Prakash, Chan, & Deshmuck, 2011). However, it will be shown in Section 2 that neglecting some of the FMS main features might lead to schedules that cannot be implemented in practice. Therefore, to obtain good quality solutions of industrial relevance, an integrated approach simultaneously coping with all the FMS scheduling sub-problems is required. This work addresses this challenge by presenting a novel CP contribution that holistically tackles the problem. In fact, the proposed formulation simultaneously takes into account the following subproblems: (i) machine loading, (ii) manufacturing activities scheduling, (iii) part routing, (iv) machine buffer scheduling, (v) tool planning and allocation, and (vi) AGV scheduling, considering both the loaded and the empty movements of the device.

This paper is structured as follows. Section 2 describes the main characteristics of the FMS environment and the problem features to be addressed, stressing tool and AGV related features and in Section 3, the constraint programming model is presented. Afterwards, in Section 4 the computational results obtained for several test examples are discussed. Finally, in Section 5 the conclusions of this work are presented.

2. FMS problem description

The main components of the FMS considered in this work, are: (i) one loading/unloading station (L/U station), devoted to these two activities, (ii) automatic machines or workstations, each one having a tool magazine of limited capacity, (iii) tools, (iv) a local buffer associated with each machine, for holding parts until the station becomes idle, (v) a single automated guided vehicle, (vi) a guidance network. Fig. 1 depicts the structure of the FMS lay-out adapted from Bilge and Ulusoy (1995) that has been adopted in this work.

This contribution addresses a scheduling problem in which a known set of parts, available at the beginning of the scheduling horizon, is to be manufactured in the FMS. All parts enter and leave the FMS throughout the L/U station. Each part requires an ordered set of operations and each operation is executed in one workstation from a set of alternative multipurpose machines. Each machine can perform just one operation on a single part at a time and has a local buffer of limited capacity, at which parts coming from other machines can wait for the workstation to become idle. Machine buffers are also used by parts having two consecutive operations assigned to the same machine when the second task does not start immediately after the first one. On the contrary, these buffers are not employed when the two consecutive machining operations are executed one immediately after the other on a given machine.

Tools are classified into types and a limited number of instances or copies of each type is available. Tool instances are assigned to machine magazines, having limited capacity. Each tool can be allocated at most to a workstation, where it can perform different operations. Two or more instances of the same tool type can be used on different machines at the same time. Each machining operation demanded by a given part requires one tool instance of a set of available types and its processing time depends on the chosen machine and tool type.

In order to be manufactured, parts are generally moved through the FMS machines, since the machine magazines may not have all the tool instances required to manufacture a given part. Part transfers are done by an automated guided vehicle, throughout a directed network. The AGV should not be used for work in process (WIP) storage. Robotic handlers are in charge of picking up parts from the AGV and introducing them inside the machines or, if they are busy, leaving them in the machine buffers. In this last case, when a station becomes idle, the robot picks up a part from the local buffer and puts it inside. Once the consecutive machining operations executed on a part at a given machine are finished, the robotic handler removes it and puts the part on the AGV in order to be transferred to another machine or to the L/U station.

2.1. Tool management

Tools are important components of FMS environments since they are expensive resources. In fact, a reduced number of instances of each type is available at the beginning of the scheduling horizon. Since tools are subject to wear they need reconditioning or replacement. Therefore, tool life-time needs to be considered in the scheduling problem. In addition, each tool copy requires a particular number of slots depending on its type, and each magazine has a fixed number of slots, which constrains the number of tools that it can accommodate and, thus, the type and number of activities that can be performed on its associated machine.

Tool allocation is a very important problem that is strongly connected with the loading and part routing ones. In fact, when a part requires a given machining operation, it has to be transported to a machine that can perform it and that has at least a suitable tool, with enough life-time to execute such operation. Therefore, tool management issues must be taken into account without decoupling them from the other subproblems. In this contribution, the adopted tool management strategy is the batching one. Before manufacturing is started, and based on the predictive scheduling problem solution, tools are selected from a pool of available ones and are allocated to the corresponding magazines. Once fixed, the configuration of each magazine should not be changed; thus, each machine can only employ the tool copies already allotted in its magazine.

2.2. AGV features

When an FMS scheduling solution technique ignores relevant AGV features or makes important assumptions, the resulting schedules may not be feasible and the approach might not be applicable in industrial settings. Among the major simplifications is the one of instantaneous part transfers. Another one considers that all the first operations on parts start simultaneously at the beginning of the scheduling horizon, which implies that all the machines are available at such time and that all parts had been already transferred.

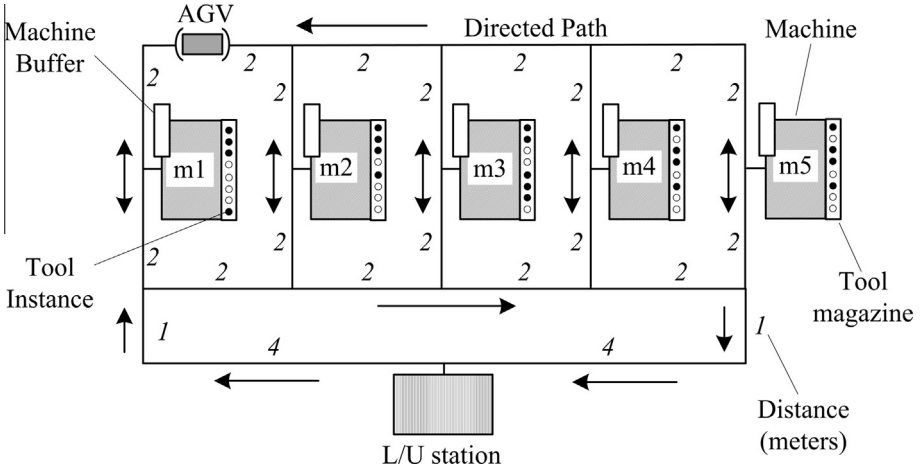

Fig. 1. FMS layout with a single AGV. Adapted from Bilge and Ulusoy (1995).

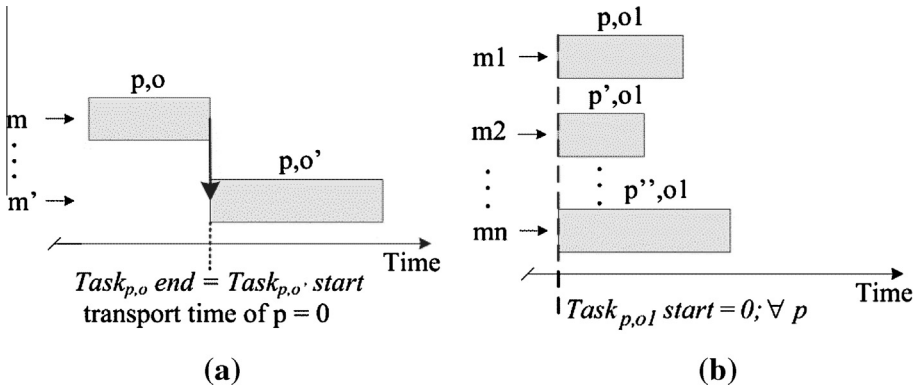

Fig. 2. Schematic representations of some unrealistic assumptions associated with the modeling of the AGV: (a) Instantaneous part transfers. (b) Simultaneous start of the first operations demanded by all the parts.

These two situations are illustrated in Fig. 2 and both are related to simplified models of the AGV that can be found in the specialized literature, and which need to be improved. For instance, Caumond et al. (2009) assumed that the number of AGV trips has a one-to-one relation with the number of operations, which is only true when each manufacturing activity demanded by a part is executed in a different machine. The same contribution also considers that an empty trip of the AGV begins just after the part that is being picked up has finished its processing in a given machine, statement that is also considered as a modeling simplification.

A proper representation of an FMS environment needs to capture that in order to move a part between machines or to transfer it from/to the L/U station, an AGV is required. The time needed by the AGV depends on both the distance between the machines (or the machine and the L/U station and vice versa) and the speed of the transport device, which in turn may vary if it performs the trip loaded or unloaded. This contribution takes into account both types of AGV movements, as well as the idle status of the AGV, which are schematically illustrated in Fig. 3. Loaded AGV trips take place when: (i) a pre-processed part is picked up from the L/U station and delivered to the machine that will execute the first operation, (ii) a part needs to be transferred between two machines in order to follow its manufacturing route, and (iii) an already finished part is taken from the machine that executed the last operation and moved to the L/U station. On the other hand, empty AGV trips are movements performed each time the AGV travels unloaded from its last delivery destination to another resource (machine or L/U station), where it is required to pick up a part that needs to be transferred to another FMS resource. After delivering a part to a given machine or to the L/U station, if the next AGV service is not immediately required, the vehicle remains idle. This waiting interval finishes when the AGV starts a new trip. It is worth mentioning that, since the AGV can only execute one activity at a time, there is no conflict between any pair of movements it performs.

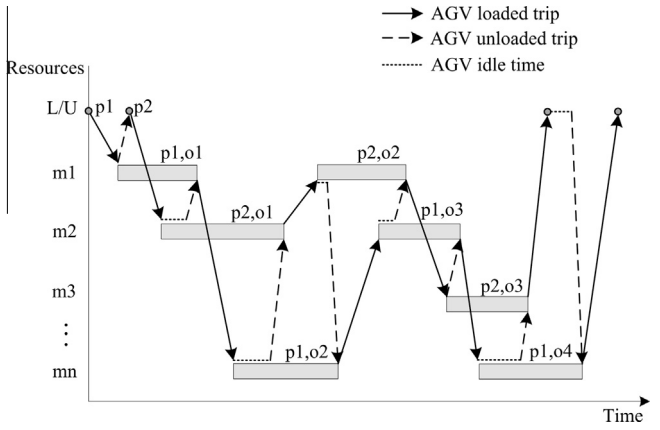

Fig. 3. Gantt chart depicting loaded AGV movements, unloaded ones, and idle intervals.

As a coordinator device, the AGV orchestrates the parts flow through the system, which is not a trivial matter. The FMS lay-out and the vehicle features are very important issues to take into account since they define transport times, which have a critical impact on the resulting schedule.

2.3. FMS addressed in this work

The physical and operational characteristics of the FMS considered in this contribution are the following: (i) the FMS works under a part movement policy; while parts follow a manufacturing route, tools are allocated to machine magazines at the beginning of the scheduling horizon and cannot be reassigned during this period, (ii) machines buffers only operate as input buffers, (iii) there is no cell central buffer for Work-In-Process (WIP), (iv) no pre-emption is allowed, (v) the FMS has a single AGV, independently of the addressed lay-out, (vi) since there is a single transport device, no congestion occurs at crossings, (vii) part transport activities are critical because the machine buffers have limited capacity and the transfer times are comparable to the processing ones, (viii) allocated tool instances are considered as new at the beginning of the scheduling horizon, (ix) one or more tool copies of the same type can be used to execute a particular operation; i.e. when a worn tool instance cannot complete a given machining operation, another one allocated to the same tool magazine, can do it. This is the strongest assumption of the model.

3. Constraint programming formulation

Constraint programming (CP) comprises computational implementations of efficient algorithms devised to tackle constraint satisfaction problems (Brailsford, Potts, & Smith, 1999). In the last two decades CP has been extensively used to address various combinatorial problems. This type of approach provides several advantages, as the flexibility to incorporate new constraints, the capability to immediately detect infeasibilities, as well as to obtain initial feasible solutions quite fast. Furthermore, optimal and good quality suboptimal solutions can be instantiated in low computational times. In addition, CP languages are highly declarative in nature, making the model development process easier. A comprehensive review on constraint-based scheduling has been presented by

Baptiste, Le Pape, and Nuijten (2001). Constraint programming approaches have been extensively applied to a variety of scheduling problems, such as staff scheduling (Bourdais, Galinier, & Pesant, 2003), scheduling of trains (Rodriguez, 2007), and assembly lines (Topaloglu, Salum, & Supciller, 2012; Öztürk, Tunali, Hnich, & Örnek, 2012). There are also CP contributions for batch plant scheduling (Harjunkoski, Jain, & Grossman, 2000; Castro, Grossmann, & Novais, 2006; Maravelias & Grossmann, 2004; Novas & Henning, 2010; Zeballos, Novas, & Henning, 2011). However, there are still few ones employing this technology to tackle FMS scheduling problems (El Khayat et al., 2006; Zeballos et al., 2010).

In this section, the resource-constrained scheduling problem of an FMS environment is addressed by means of a CP model that has been implemented in the OPL language, which underlies the ILOG OPL Studio environment (ILOG, 2002). It employs some specific scheduling primitives available in the ILOG Scheduler package (ILOG, 2000a): (i) *precedes*, which ensures the proper sequencing of non-overlapping activities, (ii) *requires*, which handles the assignment of resources demanded by the tasks, and (iii) the predicate *activityHasSelectedResource*, which evaluates to one when a given task has been assigned to a particular machine belonging to a set of alternative resources.

3.1. Nomenclature

Sets/indices	
P/p,p'	Parts to be manufactured
O/o,o'	Machining operations
T/t,t'	Tool types
M/m,m'	Machines
$MBuffer_m$	Input buffer associated with machine *m*
O_p	Subset of machining operations demanded by part *p*
$PMach_{p,o}$	Subset of machines that can carry out the operation *o* required by part *p*
$PTool_{p,o}$	Subset of tool types able to execute the operation *o* required by part *p*
$PMTool_{p,o,m}$	Subset of tool types that can perform the operation *o* demanded by part *p*, at machine *m*
TaskSet	Manufacturing tasks to be scheduled
ToolTaskSet	Manufacturing tasks to be executed by the tools that need to be scheduled
agv	Automated guided vehicle
lus	Loading/unloading (L/U) station
r,r'	Resource items, which refer either to machines or the L/U station
Parameters	
$pt_{p,o,t,m}$	Processing time of operation *o* demanded by part *p* when employing a tool of type *t* on machine *m*
$tt_{r,r'}$	Time required by the AGV to transport a part from resource *r* to resource *r'*
$et_{r,r'}$	Time demanded by the AGV to perform an empty movement between resources *r* and *r'*
lt_t	Life-time of a tool instance of type *t*
st_t	Number of slots required to allocate an instance of tool type *t*
sm_m	Number of slots available at the magazine of machine *m*
nti_t	Overall number of instances of tool type *t*
mbc_m	Capacity of the input buffer associated with machine *m*
Variables	
$Task_{p,o}$	Manufacturing task corresponding to operation *o* demanded by part *p*
$ToolTask_{p,o}$	Machining operation executed by a given tool instance that corresponds to the operation *o* required by part *p*. This activity has a tight 1:1 relation with $Task_{p,o}$
$InTranspTask_p$	Task belonging to the AGV agenda, which represents the transport of part *p* from the L/U station to the machine where its first manufacturing task has to be executed
$OutTranspTask_p$	Task belonging to the AGV agenda, which represents the transport of part *p* from the machine where its last manufacturing task has to be executed, to the L/U station
$InterTranspTask_{p,o}$	Task belonging to the AGV agenda, which represents the intermediate transport of part *p* from a machine where the operation *o* has already finished, to another machine where the next operation is going to be executed on *p*
$EmptyMove_{r,r'}$	Activity representing the empty movement of the AGV from resource *r* to resource *r'*
$LocalStgTask_{p,o}$	Activity representing the storage of part *p* before executing operation *o*. It takes place at the input buffer of the machine where such operation is going to be performed
$FirstAssignedM_p$	Machine that carries out the first operation on part *p*
$LastAssignedM_p$	Machine that carries out the last operation on part *p*
$AssignedOrigM_{p,o}$	Machine from where part *p* is picked up after finishing operation *o*, in order to be taken to another machine where its next operation is to be performed
$AssignedDestM_{p,o}$	Machine where the part *p* is delivered to, after the completion of operation *o*, in order to carry out the next machining task
$AssignedTool_{p,o}$	Tool assigned to operation *o* demanded by part *p*
$Ti_{m,t}$	Number of tool instances of type *t* allocated to machine *m*
$VM_{p,o}$	Binary variable denoting that a vehicle movement is performed to transport part *p* after carrying out operation *o* in a given machine in order to execute the next operation in another one
Mk	Makespan

The ILOG OPL modeling language distinguishes among various resource types, such as: (i) unary, representing renewable resources with capacity equal to one that cannot be associated with more than one task at a time, (ii) discrete, which are resources of finite capacity (greater than one), that can be shared by various activities at a time, depending on their capacity. How resources are declared in the formulation has an important impact on the solution quality and, therefore, it is worth describing the way they are handled in this proposal:

(a) Machines are declared as unary resources. This ensures that each machine m belonging to M can perform at most one task at a time within the scheduling horizon.
(b) The AGV is also a unary resource since it can do just one transportation task at a time.
(c) Tool types are declared as discrete resources. Defining tool types belonging to set T as discrete resources allows different instances of the same category to execute machining operations on dissimilar stations, at the same time. The maximum number of instances of a given class that can perform concurrent operations (at the same moment) is defined by the nti_t parameter, which represents the overall number of tool instances of type t that are available in the FMS.
(d) Machine buffers are declared as discrete resources having a limited capacity to temporarily store in-process parts. The buffer capacity depends on the machine and it is captured by the mbc_m parameter.

On the other hand, ILOG characterizes the activities $Task_{p,o}$, $ToolTask_{p,o}$, $InTranspTask_p$, $OutTranspTask_p$, $InterTranspTask_{p,o}$, $EmptyMove_{r,r'}$ and $LocalStgTask_{p,o}$, in terms of, start, end and duration variables, of which only two are independent.

3.2. Constraint programming model

Constraints (1)–(12) model manufacturing activity features associated with the operation of the FMS environment, as well as those tool management issues that have been described in Section 2.1. Machining activities are represented in the same way they were modeled in Zeballos et al. (2010). Manufacturing tasks are decoupled into two activity variables: $Task_{p,o}$ and $ToolTask_{p,o}$, which represent the utilization of machines and tools to perform machining operations on parts, respectively. This decoupling approach enhances the representation by: (i) giving more flexibility to the model, (ii) allowing the formulation of specific machine and tool related constraints, and (iii) reducing the model dimensionality in one order of magnitude.

3.2.1. Machine loading constraints

Constraint (1) prescribes that each manufacturing operation o required by each part p must be assigned to a workstation belonging to the set of FMS machines. As each station has been declared as a unary resource, constraint (1) also enforces the non-overlapping of tasks assigned to each machine. In consequence, the sequencing of tasks on each machine is automatically established. Constraint (2), by negating the *ActivityHasSelectedResource* predicate, forbids the assignment of each manufacturing activity $Task_{p,o}$ to those machines that do not belong to the set $PMach_{p,o}$ of permitted ones

$$Task_{p,o}\ requires\quad M;\ \forall\, Task_{p,o} \in TaskSet,\ \forall p \in P,\ \forall o \in O_p \tag{1}$$

$$not\quad activityHasSelectedResource(Task_{p,o}, M, m);\\ \forall\, Task_{p,o} \in TaskSet, \forall p \in P, \forall o \in O_p,\quad \forall m \notin PMach_{p,o} \tag{2}$$

3.2.2. Tool allocation constraints

Constraint (3) enforces the assignment of a tool belonging to the set of tool types T to each $ToolTask_{p,o}$ activity. In turn, constraint (4) forbids the assignment of each $ToolTask_{p,o}$ to a tool instance t that does not belong to the set $PTool_{p,o}$ that can execute such machining operation. Constraint (5) complements expressions (3) and (4) in order to prescribe that when a manufacturing activity $Task_{p,o}$ is assigned to a particular machine m, then it cannot be associated with a tool of type t if such type is not part of the $PMTool_{p,o,m}$ set.

$$ToolTask_{p,o}\ requires(1)\quad T;\quad \forall\, ToolTask_{p,o} \in ToolTaskSet,\\ \forall p \in P, \forall o \in O_p \tag{3}$$

$$not\quad activityHasSelectedResource(ToolTask_{p,o}, T, t);\\ \forall ToolTask_{p,o} \in ToolTaskSet, \forall p \in P, \forall\, o \in O_p,\\ \forall t \notin PTool_{p,o} \tag{4}$$

$$activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \Rightarrow\\ not\quad activityHasSelectedResource(ToolTask_{p,o}, PTool_{p,o}, t);\\ \forall Task_{p,o} \in TaskSet, \forall\, ToolTask_{p,o} \in ToolTaskSet,\\ \forall p \in P, \forall o \in O_p,\quad \forall\, t \notin PMTool_{p,o,m} \tag{5}$$

Constraint (6) specifies the number of tool copies $Ti_{m,t}$ of a given type t that are allocated to an m machine. The product of the lifetime of each instance of type t and the number of tool copies assigned to m imposes an upper limit on the total processing time of those machining operations associated with tool type t (left-hand side of the expression). The $Ti_{m,t}$ variable also participates in constraints (7) and (8). Constraint (7) specifies that the total number of copies of each type t allocated to all the machine magazines is bounded by the number of available tool instances of such type, nti_t. In turn, constraint (8) takes into account the tool magazine capacity of the machines. For each machine, it ensures that the number of slots occupied by all the tools allocated in the magazine, does not surpass the number of available slots, sm_m.

$$\sum_{p\in P, o\in O_p} (activityHasSelectedResource(ToolTask_{p,o}, PTool_{p,o}, t) *\\ activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) *\\ ToolTask_{p,o}.duration) \leqslant lt_t * Ti_{m,t};\quad \forall m \in M, \forall t \in T,\\ \forall Task_{p,o} \in TaskSet, \forall ToolTask_{p,o} \in ToolTaskSet \tag{6}$$

$$\sum_{m\in M} Ti_{m,t} \leqslant nti_t;\quad \forall t \in T \tag{7}$$

$$\sum_{t\in T} Ti_{m,t} * st_t \leqslant sm_m;\quad \forall m \in M \tag{8}$$

3.2.3. Tasks scheduling constraints

Constraint (9) enforces the proper sequencing of the $Task_{p,o}$ activities that pertain to the manufacturing route of part p. Constraint (10) has the same purpose; however, it refers to their related $ToolTask_{p,o}$ activities. Constraint (11) assigns the proper processing time to $Task_{p,o}$, which depends on the part p, operation o, and on the assigned machine m and tool type t. In addition, constraint (12) ensures that the related activities $Task_{p,o}$ and $ToolTask_{p,o}$ have the same start time and duration

$$Task_{p,o}\quad precedes\quad Task_{p,o'};\\ \forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P,\\ \forall o, o' \in O_p, o < last(O_p),\quad ord(o') = ord(o) + 1; \tag{9}$$

$$ToolTask_{p,o}\quad precedes\quad ToolTask_{p,o'};\\ \forall ToolTask_{p,o}, ToolTask_{p,o'} \in ToolTaskSet, \forall p \in P,\\ \forall o, o' \in O_p, o < last(O_p),\quad ord(o') = ord(o) + 1; \tag{10}$$

$$activityHasSelectedResource(ToolTask_{p,o}, PTool_{p,o}, t) \wedge\\ activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \Rightarrow\\ Task_{p,o}.duration = pt_{p,o,t,m};\\ \forall Task_{p,o} \in TaskSet, \forall\, ToolTask_{p,o} \in ToolTaskSet,\\ \forall p \in P, \forall o \in O_p,\quad \forall m \in PMach_{p,o}, \forall t \in PMTool_{p,o,m} \tag{11}$$

$$Task_{p,o}.start = ToolTask_{p,o}.start \wedge$$
$$Task_{p,o}.duration = ToolTask_{p,o}.duration;$$
$$\forall Task_{p,o} \in TaskSet, \forall ToolTask_{p,o} \in ToolTaskSet, \forall p \in P, \forall o \in O_p \quad (12)$$

Constraints (13)–(29) model the AGV trips, as well as the local storage of parts in machine buffers. These constraints overcome the limiting simplifications that have been described in Section 2.2. The following three types of loaded AGV movements are modeled:

(i) The input transport of a non-processed part from the L/U station to the machine where the first machining operation is to be carried out. This activity is represented by the $InTranspTask_p$ variable, whereas the $FirstAssignedM_p$ one models the station where the part p is assigned.
(ii) The output transfer of an already processed part from its last assigned machine to the L/U station, which is modeled by the $OutTranspTask_p$ variable. The $LastAssignedM_p$ variable represents the machine where the last operation demanded by p takes place.
(iii) The transport of an in-process part between two machines, which is represented by the $InterTranspTask_{p,o}$ variable. This task in turn is associated with the $AssignedOrigM_{p,o}$ and $AssignedDestM_{p,o}$ variables. These last ones model the machine where part p is picked-up and delivered to, respectively.

In contrast, empty AGV movements are modeled by means of the $EmptyMove_{r,r'}$ variable. They occur when the vehicle travels unloaded from its last destination (resource r) to another resource r' (machine or the L/U station), where it is required to pick-up a part.

Local storage tasks that take place at machine buffers are represented by means of the $LocalStgTask_{p,o}$ variable. A local storage activity occurs every time the execution of operation o demanded by part p cannot start immediately after the arrival of the part to the assigned machine due to unavailability. Fig. 4 illustrates the relation among the different loaded transport activities, their associated machine allocation variables, empty vehicle moves, and the local storage and processing tasks.

3.2.4. Loaded AGV scheduling constraints

Constraint (13) is an assignment relation prescribing that for each part p, both input and output transport activities must be assigned to the AGV. Expression (14) specifies that when the first operation o demanded by a part p has been assigned to a machine m, the start time of operation o imposes an upper bound on the completion of the input transport task of p. In addition, the duration of this input transport activity that connects the L/U station and machine m is defined by the parameter $tt_{lus,m}$. This constraint also instantiates the $FirstAssignedM_p$ variable that models the first machine that part p visits. Similarly, constraint (15) enforces the output transport of part p to start at the end of its last required operation o, and sets its duration as the transport time $tt_{m,lus}$ demanded to move the part between the machine m, where the last operation o has been executed, and the L/U station. In addition, it instantiates the $LastAssignedM_p$ variable that captures the last machine that part p visits.

$$InTranspTask_p \ requires \quad agv \wedge$$
$$OutTranspTask_p \quad requires \quad agv; \quad \forall p \in P \quad (13)$$

$$activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \Rightarrow$$
$$InTranspTask_p.end \leqslant Task_{p,o}.start \wedge$$
$$InTranspTask_p.duration = tt_{lus,m} \wedge FirstAssignedM_p = m;$$
$$\forall Task_{p,o} \in TaskSet, \forall p \in P, \forall o \in O_p, \forall m \in PMach_{p,o}, o = first(O_p) \quad (14)$$

$$activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \Rightarrow$$
$$OutTranspTask_p.start = Task_{p,o}.end \wedge$$
$$OutTranspTask_p.duration = tt_{m,lus} \wedge LastAssignedM_p = m$$
$$\forall Task_{p,o} \in TaskSet, \forall p \in P, \forall o \in O_p, \forall m \in PMach_{p,o}, o = last(O_p) \quad (15)$$

When intermediate transport tasks occur (duration greater than zero) they must be assigned to the AGV. This condition is modeled by constraint (16). Constraint (17) represents the relation between two consecutive operations o and o' on a part p, taking place on different machines m and u, and the intermediate transport task carried out between them. The expression prescribes that the start of operation o' must be equal or greater than the end of its predecessor o plus the transport time $tt_{m,u}$. In addition, it enforces the transport task to start at the end of the operation o and instantiates the $AssignedOrigM_{p,o}$ and $AssignedDestM_{p,o}$ variables. Constraint (18) ensures that when two consecutive operations are assigned to the same machine, no intermediate transport is required and the start of the operation o' is equal or greater than the end of its predecessor o.

$$InterTranspTask_{p,o}.duration > 0 \iff$$
$$InterTranspTask_{p,o} \ requires \quad agv \quad (16)$$
$$\forall Task_{p,o} \in TaskSet, \forall p \in P, \forall o \in O_p$$

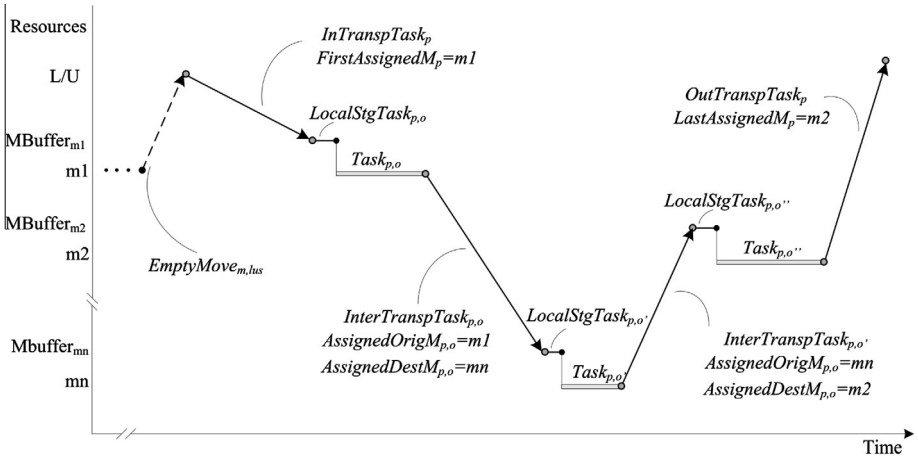

Fig. 4. Conceptual graph depicting the different AGV moves, their associated variables, as well as the storage and processing tasks.

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \wedge \\ &activityHasSelectedResource(Task_{p,o'}, PMach_{p,o'}, u) \Rightarrow \\ &Task_{p,o'}.start \geqslant Task_{p,o}.end + tt_{m,u} \wedge \\ &InterTranspTask_{p,o}.duration = tt_{m,u} \wedge \\ &InterTranspTask_{p,o}.start = Task_{p,o}.end \wedge \\ &AssignedOrigM_{p,o} = m \wedge AssignedDestM_{p,o} = u; \\ &\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p, \\ &ord(o') = ord(o) + 1, o < last(O_p), \forall m, u \in M, u \neq m\end{aligned} \tag{17}$$

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \wedge \\ &activityHasSelectedResource(Task_{p,o'}, PMach_{p,o'}, m) \Rightarrow \\ &Task_{p,o'}.start \geqslant Task_{p,o}.end \wedge \\ &InterTranspTask_{p,o}.duration = 0 \wedge \\ &AssignedOrigM_{p,o} = m \wedge AssignedDestM_{p,o} = m; \\ &\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p, \\ &ord(o') = ord(o) + 1, o < last(O_p), \forall m \in M\end{aligned} \tag{18}$$

3.2.5. Empty AGV trips associated constraints

The unloaded trips between resources r and r' (machines or L/U station) are represented by the $EmptyMove_{r,r'}$ activity. These empty movements take place at the beginning of the scheduling horizon or, more frequently, between two loaded trips. Constraint (19) represents the precedence relation between two input transport activities associated with two parts p and k. It considers the unloaded trip carried out from the destination machine of the first input transport trip and the L/U station where the second input movement will start. Similarly, constraint (20) models the relation between two output transport tasks, which at least are separated by an empty movement. In this case, the unloaded trip starts at the L/U station and ends at the machine where the second output movement will begin.

$$\begin{aligned}&(InTranspTask_p \quad precedes \quad InTranspTask_k \wedge \\ &InTranspTask_k.start - InTranspTask_p.end \geqslant \\ &EmptyMove_{FirstAssignedM_p, lus}.duration) \vee \\ &(InTranspTask_k \quad precedes \quad InTranspTask_p \wedge \\ &InTranspTask_p.start - InTranspTask_k.end \geqslant \\ &EmptyMove_{FirstAssignedM_k, lus}.duration); \forall p, k \in P, p \neq k\end{aligned} \tag{19}$$

$$\begin{aligned}&(OutTranspTask_p \quad precedes \quad OutTranspTask_k \wedge \\ &OutTranspTask_k.start - OutTranspTask_p.end \geqslant \\ &EmptyMove_{lus, LastAssignedM_k}.duration) \vee \\ &(OutTranspTask_k \quad precedes \quad OutTranspTask_p \wedge \\ &OutTranspTask_p.start - OutTranspTask_k.end \geqslant \\ &EmptyMove_{lus, LastAssignedM_p}.duration); \forall p, k \in P, p \neq k\end{aligned} \tag{20}$$

Expression (21) prescribes that every time an output transport task precedes an input one, no empty move is required since the AGV is already located at the L/U station to start the second movement activity. On the other hand, when an input transfer task precedes an output one, an empty move must take place between the machine ($FirstAssignedM_p$), where the input movement delivers the part, and the machine ($LastAssignedM_k$) where the output transport task will begin.

$$\begin{aligned}&OutTranspTask_p \quad precedes \quad InTranspTask_k \vee \\ &(InTranspTask_k \quad precedes \quad OutTranspTask_p \wedge \\ &OutTranspTask_p.start - InTranspTask_k.end \geqslant \\ &EmptyMove_{FirstAssignedM_k, LastAssignedM_p}.duration); \ \forall p, k \in P, p \neq k\end{aligned} \tag{21}$$

Constraint (22) ensures the proper temporal arrangement between input and intermediate transport tasks. In this situation, when the input trip precedes the intermediate transport one or the other way around, an unloaded AGV movement has to be taken into account. A similar expression (constraint 23) has been developed to represent the precedence relations between output and intermediate transport activities. Finally, constraint (24) models the relation between two intermediate transport tasks in the case the AGV has to move itself unloaded from the destination machine of the first movement, $AssignedDestM_{p,o}$, to another machine, $AssignedOrigM_{k,n}$, where the k part is required to be picked up in order to be transferred.

$$\begin{aligned}&(InTranspTask_p \quad precedes \quad InterTranspTask_{k,n} \wedge \\ &InterTranspTask_{k,n}.start - InTranspTask_p.end \geqslant \\ &EmptyMove_{FirstAssignedM_p, AssignedOrigM_{k,n}}.duration) \vee \\ &(InterTranspTask_{k,n} \quad precedes \quad InTranspTask_p \wedge \\ &InTranspTask_p.start - InterTranspTask_{k,n}.end \geqslant \\ &EmptyMove_{AssignedDestM_{k,n}, lus}.duration); \\ &\forall p, k \in P, p \neq k, n \in O_k, n < last(O_k)\end{aligned} \tag{22}$$

$$\begin{aligned}&(OutTranspTask_p \quad precedes \quad InterTranspTask_{k,n} \wedge \\ &InterTranspTask_{k,n}.start - OutTranspTask_p.end \geqslant \\ &EmptyMove_{lus, AssignedOrigM_{k,n}}.duration) \vee \\ &(InterTranspTask_{k,n} \quad precedes \quad OutTranspTask_p \wedge \\ &OutTranspTask_p.start - InterTranspTask_{k,n}.end \geqslant \\ &EmptyMove_{AssignedDestM_{k,n}, LastAssignedM_p}.duration); \\ &\forall p, k \in P, p \neq k, n \in O_k, n < last(O_k)\end{aligned} \tag{23}$$

$$\begin{aligned}&AssignedDestM_{p,o} \neq AssignedOrigM_{k,n} \Rightarrow \\ &(InterTranspTask_{p,o} \quad precedes \quad InterTranspTask_{k,n} \wedge \\ &InterTranspTask_{k,n}.start - InterTranspTask_{p,o}.end \geqslant \\ &EmptyMove_{AssignedDestM_{p,o}, AssignedOrigM_{k,n}}.duration) \vee \\ &(InterTranspTask_{k,n} \quad precedes \quad InterTranspTask_{p,o} \wedge \\ &InterTranspTask_{p,o}.start - InterTranspTask_{k,n}.end \geqslant \\ &EmptyMove_{AssignedDestM_{k,n}, AssignedOrigM_{p,o}}.duration); \\ &\forall p, k \in P, p \neq k, o \in O_p, n \in O_k, o < last(O_p), n < last(O_k)\end{aligned} \tag{24}$$

3.2.6. Storage-related constraints

When a part p that has been assigned to machine m in order to execute operation o' cannot start such operation because m is not idle, the part must be placed in the local buffer $MBuffer_m$ in order to wait to be processed. In fact, a storage activity, $LocalStgTask_{p,o'}$, occurs if one of the following situations arises: (i) the intermediate transport task that has moved p to machine m ends before the actual start of the $Task_{p,o'}$ processing activity; i.e. p has to wait for m to become idle; (ii) two consecutive operations demanded by part p, o and o', are assigned to the same machine m, but operation o' does not start immediately after the end of its predecessor o. In this case, p is assigned to the machine buffer after operation o has finished (no transport is needed), and it is temporally stored there until the next operation o' starts. While part p waits at the buffer, other parts can be processed in the machine. These conditions are captured by the expression (25).

$$\begin{aligned}&LocalStgTask_{p,o'} \quad requires(1) \quad MBuffer_m \Longleftrightarrow\\&((InterTranspTask_{p,o}.end < Task_{p,o'}.start \wedge\\&AssignedDestM_{p,o} = m) \vee\\&(activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \wedge\\&activityHasSelectedResource(Task_{p,o'}, PMach_{p,o'}, m) \wedge\\&Task_{p,o}.end < Task_{p,o'}.start));\\&\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p,\\&ord(o') = ord(o) + 1, o < last(O_p), \forall m \in M\end{aligned} \tag{25}$$

Expressions (26) and (27) are timing constraints that consider local storage tasks demanded by a part p. Constraint (26) models the two alternative situations that appear when $Task_{p,o}$ is assigned to a machine m and its subsequent activity $Task_{p,o'}$ to another workstation u: (i) The local storage activity begins when the transfer between m and u finishes and ends when $Task_{p,o'}$ starts. (ii) The start of $Task_{p,o'}$ occurs immediately after the intermediate transfer between m and u ends; therefore, no local storage occurs and the associated task has a duration equal to zero.

Constraint (27) models the two alternative situations that take place when two consecutive manufacturing tasks required by part p are assigned to the same workstation m: (i) a local storage task of p at m is required because a manufacturing activity on another part takes place between the two ones on part p. In this case, the start time of the successor activity $Task_{p,o'}$ is greater than the end time of its predecessor $Task_{p,o}$, The storage activity begins when $Task_{p,o}$ finishes and lasts until $Task_{p,o'}$ starts. (ii) When $Task_{p,o'}$ starts immediately after the end of $Task_{p,o}$, no local storage is required.

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \wedge\\&activityHasSelectedResource(Task_{p,o'}, PMach_{p,o'}, u) \Rightarrow\\&(Task_{p,o'}.start \geqslant Task_{p,o}.end + tt_{m,u} \wedge\\&LocalStgTask_{p,o'}.start = InterTranspTask_{p,o}.end \wedge\\&LocalStgTask_{p,o'}.end = Task_{p,o'}.start) \vee\\&(Task_{p,o'}.start = Task_{p,o}.end + tt_{m,u} \wedge\\&LocalStgTask_{p,o'}.duration = 0);\\&\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p,\\&ord(o') = ord(o) + 1, o < last(O_p), \forall m \in PMach_{p,o},\\&\forall u \in PMach_{p,o'}, m \neq u\end{aligned} \tag{26}$$

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \wedge\\&activityHasSelectedResource(Task_{p,o'}, PMach_{p,o'}, m) \Rightarrow\\&(Task_{p,o'}.start > Task_{p,o}.end \wedge\\&LocalStgTask_{p,o'}.start = Task_{p,o}.end \wedge\\&LocalStgTask_{p,o'}.end = Task_{p,o'}.start) \vee\\&(Task_{p,o'}.start = Task_{p,o}.end \wedge\\&LocalStgTask_{p,o'}.start = Task_{p,o'}.start \wedge\\&LocalStgTask_{p,o'}.duration = 0);\\&\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p,\\&ord(o') = ord(o) + 1, o < last(O_p), \forall m \in M\end{aligned} \tag{27}$$

Constraints (28) and (29) represent the relationships that exist between an input transport task and a local storage activity. Expression (28) establishes that a local storage task is demanded when the start of the first manufacturing activity on part p does not take place immediately after the end of the input transport task. Constraint (29) coordinates the start and the end of the storage task with the end of the input transport activity and the start of the first manufacturing operation, respectively. In case the machining of part p takes place immediately the input transport activity, constraint (29) sets the duration of the local storage to be equal to zero.

$$\begin{aligned}&LocalStgTask_{p,o} \quad requires(1) \quad MBuffer_m \Longleftrightarrow\\&InTranspTask_p.end \neq Task_{p,o}.start \wedge FirstAssignedM_p = m;\\&\forall Task_{p,o} \in TaskSet, \forall p \in P, \forall o \in O_p, \forall m \in M, o = first(O_p)\end{aligned} \tag{28}$$

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, PMach_{p,o}, m) \Rightarrow\\&(InTranspTask_p.end \neq Task_{p,o}.start \wedge\\&LocalStgTask_{p,o}.start = InTranspTask_p.end \wedge\\&LocalStgTask_{p,o}.end = Task_{p,o}.start) \vee\\&(InTranspTask_p.end = Task_{p,o}.start \wedge\\&LocalStgTask_{p,o}.start = InTranspTask_p.end \wedge\\&LocalStgTask_{p,o}.duration = 0);\\&\forall Task_{p,o} \in TaskSet, \forall p \in P, \forall o \in O_p, \forall m \in M, o = first(O_p)\end{aligned} \tag{29}$$

3.3. Objective functions

Makespan (Mk) has been chosen as one of the performance measures to be minimized (Expression (30)). As the makespan represents the maximum completion time among all parts, the end time of the last operation of each part cannot exceed the Mk value; this condition is captured by constraint (31).

$$Min \quad Mk \tag{30}$$

$$Task_{p,o}.end \leqslant Mk; \quad \forall Task_{p,o} \in TaskSet, \forall p \in P, o = last(O_p) \tag{31}$$

In addition, a multiobjective function (U) first introduced by Gamila and Motavalli (2003) and then used by Zeballos et al. (2010), has been adopted. This function (see expression 32) integrates three terms, which are minimized. The first one represents the makespan, the second the number of required AGV movements and the last one stands for the total processing time. Regarding vehicle movements, they are calculated by means of expressions (33) and (34), i.e. no explicit variables are considered for the AGV. When the machines assigned to two consecutive operations executed on a given part p are different, the binary variable $VM_{p,o}$ evaluates to one (33). Otherwise, it takes zero value (34). The multiobjective function presented in expression (32) has not been normalized and each term contributes differently to the overall assessment. Therefore, as it was pointed out by Zeballos et al. (2010), the U value has a relative significance. Nevertheless, it has been chosen just for comparison purposes.

$$Mk + \sum_{p \in P, o \in O_p} VM_{p,o} + \sum_{TaskSet} Task_{p,o}.duration \leqslant U; \tag{32}$$

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, M, m) \wedge\\&activityHasSelectedResource(Task_{p,o}, M, u) \Rightarrow VM_{p,o} = 1;\\&\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p,\\&ord(o') = ord(o) + 1, o < last(O_p), \forall m, u \in M, m \neq u\end{aligned} \tag{33}$$

$$\begin{aligned}&activityHasSelectedResource(Task_{p,o}, M, m) \wedge\\&activityHasSelectedResource(Task_{p,o}, M, m) \Rightarrow VM_{p,o} = 0;\\&\forall Task_{p,o}, Task_{p,o'} \in TaskSet, \forall p \in P, \forall o, o' \in O_p,\\&ord(o') = ord(o) + 1, o < last(O_p), \forall m \in M\end{aligned} \tag{34}$$

4. Computational results and discussion

The CP formulation presented in this work has been validated with several case studies. To the best of our knowledge, test instances for FMS scheduling problems that consider machines, machine buffers, tools, and an AGV, altogether as constraining resources, have not been reported in the literature. Due to this fact, the case studies presented in this section have been generated combining data from different sources. Thus, data concerning machines, tools, tool magazines, parts and manufacturing routes, have been taken from the problem instances described in Zeballos et al. (2010) and Gamila and Motavalli (2003). In addition, AGV related issues, as well as the adopted FMS layout and vehicle network, are based on an extensively used example which was first introduced by Bilge and Ulusoy (1995).

Table 1 summarizes the main characteristics of the problem instances addressed in this work. All the case studies consider an FMS environment involving 5 multipurpose workstations, in which there are 22 types of tools to execute the various machining operations. Each machine has a tool magazine with a limited capacity of 60 slots.

Problems P2 and P3 correspond to the first and fourth test cases presented by Zeballos et al. (2010), while P4 matches the second example addressed by Gamila and Motavalli (2003). In turn, problems P1 and P5 introduce minor modifications into the first and sixth cases reported in Zeballos et al. (2010), in order to consider four and six parts, respectively. Data concerning machines and tools (i.e., operations per part, available machines and tools per part and operation, as well as machining processing times), for problems P1, P2, P3, and P5, were taken from Table A.1 of Zeballos et al. (2010). Data for problem P4 can be found in Table 3 of Gamila and Motavalli (2003). With respect to the information associated with transportation features, such as the distances between machines and the AGV network characteristics, these data have been taken from one of the examples proposed by Bilge and Ulusoy (1995). In this last work, the FMS environment is composed of four machines, two AGVs and has no limits on the machine buffer size. This original example has been adapted by increasing the number of machines to five stations, restricting the number of vehicles to one, and defining two parts as the maximum machine buffer capacity. Fig. 1 depicts the layout adopted in this contribution, showing the vehicle track, as well as the distances between machines (in meters).

Table 1
Characteristics of the different problem instances.

Problem ID	Number of parts	Number of operations per part	Number of tool copies per tool type	Tool life- time
P1	4	4	4	70
P2	5	4	3	70
P3	5	7	4	70
P4	5	5	ET*	150
P5	6	4	4	70

* ET: Enough number of tool copies to accomplish all tasks.

Table 2
Time(s) that the AGV needs to carry a part between resources. AGV speed = 40 m/min, $tp = 0.13$.

Origin resource r	Destination resource r'					
	L/U	*m1*	*m2*	*m3*	*m4*	*m5*
L/U	0	11	14	17	20	23
m1	23	0	9	12	15	18
m2	20	9	0	9	12	15
m3	17	12	9	0	9	12
m4	14	15	12	9	0	9
m5	11	18	15	12	9	0

Table 3
Results for the various instances of problems P1–P5 when considering loaded AGV trips. Makespan minimization is pursued.

Problem	Variables	Constraints	*tp* ratio	Optimal/best solution in 1000 s		
				Makespan	Total number of tool instances	CPU time[a]
P1	383	3653	0.04	363	26	1.3
			0.05	365	26	1.4
			0.07	372	27	2.8
			0.09	381	26	2.1
			0.13	402	25	1.9
P2	451	4538	0.04	375	30	2.8
			0.05	378	30	3.3
			0.07	383	30	3.8
			0.09	395	29	18.1
			0.13	434	31	721.2
P3	676	9963	0.04	652	54	212.5
			0.05	655	51	231.1
			0.07	719[b]	56	31.2
			0.09	751[b]	52	74.7
			0.13	762[b]	52	753.9
P4	526	6164	0.04	439	21	41.1
			0.05	441	21	79.9
			0.07	449	20	97.7
			0.09	458	20	219.6
			0.13	620[b]	23	89.5
P5	519	5421	0.04	420	38	586.3
			0.05	439[b]	39	92.7
			0.07	447[b]	38	650.1
			0.09	493[b]	39	17.5
			0.13	638[b]	41	1.5

[a] Time required to reach optimal solutions or to instantiate suboptimal ones.
[b] Best suboptimal solution instantiated within 1000 s.

In order to evaluate the CP approach presented in Section 3, several test examples have been addressed. Problems were solved to optimality or to a maximum time limit of 1000 s of CPU by the commercial software ILOG OPL Studio 3.7, based on the ILOG Solver (ILOG, 2000b) and Scheduler packages (ILOG, 2000a). The rationale behind this time limit has grounds on the time that a scheduler can reasonably wait in an FMS industrial setting to get a solution. A computer consisting of a Pentium Dual Core 3.0 GHz processor with 3 GB of RAM was used.

As it was mentioned in Section 2.2, an important issue to take into account when dealing with transportation features is the relative magnitude of the transfer times in relation to the processing ones. This relation is represented by the $\overline{tt}/\overline{pt}$ ratio, named *tp* in the rest of the work, in which $\overline{tt}$ is the average transport time and $\overline{pt}$ is the average processing time. With the aim of analyzing the impact of variations on this ratio, different transport times have been generated for each problem instance, while maintaining the processing times. The changes on transfer times were attained by considering the following vehicle speeds: 40, 60, 80, 100, and 120 m/min. The corresponding ratios belong to the [0.04, 0.13] interval, which reflects AGV transport times in real settings.

Table 2 presents the time required by the loaded AGV to travel between the various resources at a speed of 40 m/min, when taking into account distances shown in Fig. 1. Note that due to the vehicle network characteristics, the time spent to go from resource *r* to *r'* can differ from the one needed for the inverse trip, i.e. from *r'* to *r*. For instance, from the L/U station to m1, the transport time is 11 s, but the time required for the inverse route is 23 s. The vehicle transport times that result from considering the other vehicle speeds can be easily calculated. They are not reported due to lack of space.

Even though the loading and unloading times needed by a robot handler to move a part from the AGV to a machine, and vice versa, are not considered in this work, this situation can be easily addressed by adding these times to the transport ones, as it was done by Caumond et al. (2009).

In order to measure the impact of considering the two types of AGV trips (loaded and unloaded), two scenarios are proposed. In the first one, examples are solved considering only loaded AGV movements, while in the second scenario, unloaded vehicle trips are also taken into account.

4.1. Scenario 1. Only loaded AGV trips are considered

Table 3 reports the results for problems P1–P5, when minimizing makespan. Each problem has been solved using the following set of *tp* ratios: 0.04, 0.05, 0.07, 0.09, and 0.13 (obtained from AGV speeds of 40, 60, 80, 100, and 120 m/min, respectively). As it was expected, even though the number of variables and constraints for each problem remain the same, the makespan values that are obtained increase as the *tp* ratio rises. Note that a high *tp* ratio indicates larger transport times, as well as more constrained relations between processing and transfer activities, i.e. it reveals a strong coupling between the machines and AGV scheduling problems.

It should be mentioned that not only the makespan value grows with higher ratio values, but also the computational effort to reach optimal solutions. In the case of the various P1 instances, which are the smallest size problems, optimal solutions were found for all ratios in about 1 or 2 s of CPU time. In the case of P2, optimal solutions were reached too, but with a larger effort for the problem instance having the highest *tp* ratio. A similar situation is observed for the P4 test cases. Optimal solutions are instantiated in less than 1000 s for all cases, except for the 0.13 ratio one. In this last case, the optimal solution was instantiated in 2112 s, with a makespan value of 604. Finally, the results of those problem instances that correspond to examples P3 and P5 reveal that optimal solutions can only be obtained for the very low *tp* ratios (0.04–0.05), within the imposed time limit. Nevertheless, the suboptimal solutions reached in the other situations are in general of good quality. An overall analysis of the results shows that in 17 out of the 25 examples optimal solution were obtained.

Another aspect to note is that in all the case studies, the required number of tool copies does not vary significantly for the various *tp* ratios being adopted. Since the processing time has remained constant, as it is expected, the number of tools did not change too much.

Fig. 5 presents the Gantt chart corresponding to the optimal solution of problem P2 – *tp* = 0.09. It depicts the manufacturing tasks, the loaded transport activities and storage tasks. As seen, the unloaded vehicle movements were considered instantaneous. Manufacturing activities are labeled with a triplet that indicates the part, the operation being executed, and the tool type employed by such task (e.g., the triplet 2-3-12 denotes the manufacturing of part 2, operation 3, using an instance of tool type 12). Storage activities are identified by a pair that first contains the part number and then a *bx* term that corresponds to the local buffer ID (e.g., 5-*b*1 identifies the holding of part 5 in the local buffer associated with machine 1).

Fig. 5 shows that taking into account the loaded vehicle trips in the CP model allows overcoming two of the main drawbacks identified in Section 2.2, which were instantaneous part transfers and the simultaneous start of all the first operations (See Fig. 2). However, the assumption of instantaneous empty AGV trips still leads to unfeasible and unrealistic solutions. For instance, the AGV moves part #4 from machine *m4* to *m5*, where operation 3 is going to take place and immediately after this movement it goes from machine *m1* to *m4* in order to transport part #5, i.e. the AGV makes an instantaneous trip between *m5* and *m1*.

4.2. Scenario 2. Loaded/Unloaded AGV trips considered

This scenario corresponds to a full scale CP model that includes constraints associated with both the loaded and unloaded AGV trips. It is assumed that the time required by the empty device to travel in a given direction between any two resources is half the time needed by a loaded trip between the same two resources.

The same set of problems tackled in the previous section was solved with a computational time bound of 1000 s. Table 4 reports the obtained results. As it was expected, in all the examples solutions having larger makespan values were obtained. In addition, it can be noted that the performance of the CP model deteriorates as the *tp* ratio increases. Contrary to scenario 1, very few optimal solutions were instantiated within the enforced time limit. Optimal solutions correspond to P1 for the 0.04, 0.05, and 0.07 ratio examples, while for problem P2 optimal solutions were reached only for the 0.04 and 0.05 ratio problem instances.

Note that the computational times are larger than those obtained when the empty trips were neglected (see Table 3). This behavior can be associated with a more complex model. Although the example corresponding to P2 with *tp* = 0.04 has a slightly larger makespan than the one reached when the unloaded trips were considered (377 against 375), the CPU time needed to instantiate the solution is considerably higher (138.5 s against 2.8 s). The same behavior can be observed for P2 with *tp* = 0.05. When loaded and unloaded AGV trips are considered, a solution with an associated makespan value of 385 was attained in 848.2 s, while a solution having a makespan of 378, which was instantiated in only 3.3 s, was reached when no empty trips were taken into account. A similar pattern can be found in those solutions that correspond to P1 with the 0.04, 0.05, and 0.07 ratios.

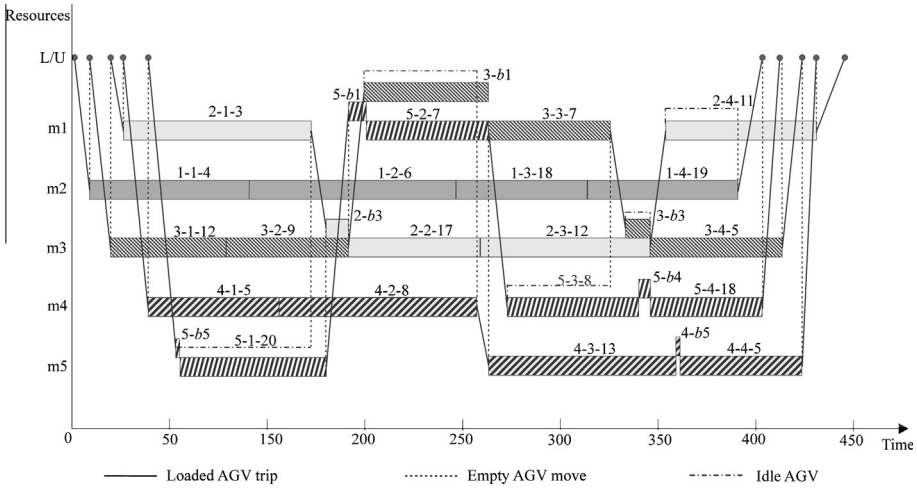

Fig. 5. Gantt diagram showing the optimal solution for P2- *tp* = 0.09, when considering empty AGV moves as instantaneous.

Table 4
Results for the various instances of problems P1–P5 when considering loaded and unloaded AGV trips. Makespan minimization is pursued.

Problem	Variables	Constraints	*tp* ratio	Optimal/best solution in 1000 s			
				Makespan	Total number of tool instances	Total number of loaded AGV trips	CPU time[a]
P1	803	3869	0.04	370	26	7	213.6
			0.05	374	25	5	221.3
			0.07	381	26	5	37.5
			0.09	401[b]	25	5	29.3
			0.13	433[b]	25	5	126.5
P2	1151	4898	0.04	377	31	8	138.5
			0.05	385	32	8	848.2
			0.07	398[b]	31	9	586.7
			0.09	438[b]	32	8	877.8
			0.13	478[b]	30	9	609.5
P3	2696	10983	0.04	706[b]	53	14	8.8
			0.05	707[b]	54	17	94.7
			0.07	750[b]	54	10	54.2
			0.09	817[b]	58	16	14.2
			0.13	973[b]	56	12	3.5
P4	1586	6704	0.04	590[b]	23	11	552.9
			0.05	626[b]	23	13	2.7
			0.07	611[b]	22	11	456.5
			0.09	756[b]	21	12	21.6
			0.13	768[b]	22	12	40.4
P5	1569	5961	0.04	455[b]	36	14	2.1
			0.05	487[b]	40	13	2.2
			0.07	536[b]	38	11	343.2
			0.09	686[b]	42	12	36.4
			0.13	702[b]	42	8	2.5

[a] Time required to reach optimal solutions or to instantiate suboptimal ones.
[b] Best suboptimal solution instantiated within 1000 s.

An important characteristic of the proposed CP approach concerns its capability to instantiate, in most cases, a first feasible solution in very low CPU time. This behavior can be illustrated by those cases corresponding to P2, with *tp* equal to 0.07, 0.09, and 0.13, in which the first feasible solutions were found in 25.3 s, 26.7 s and 38.2 s, respectively.

Table 4 also shows the total number of tool instances for each example. For most problems, the average total number of tool copies that are required is slightly larger than the one needed when no empty trips are taken into consideration. For instance, the five P2 instances employed in average 31.2 tool copies against 30.0 when no unloaded trips were modeled; similarly, the P3 test cases needed 55.0 tool instances against 53.0 used when the empty trips were neglected.

By capturing the number of loaded AGV trips, also shown in Table 4, it is possible to observe that for most case studies, the required loaded vehicle movements are at least equal to the number of trips that could be estimated by a CP model that completely ignores AGV trips. The results show that, independently of the adopted *tp* ratio, when vehicle travels are explicitly modeled the number of loaded trips is generally higher. Also, the transport of the to-be-manufactured parts from the L/U station to their first assigned machines, as well as the trips of the already manufactured parts to the L/U station, should also be considered.

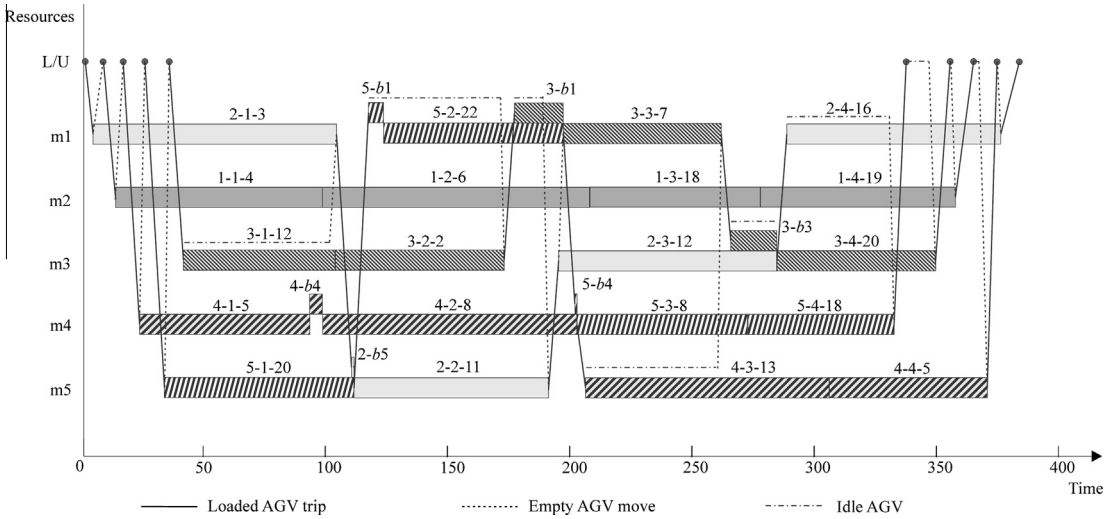

Fig. 6. Gantt diagram showing the optimal solution for P2- *tp* = 0.05, when considering loaded and empty AGV trips.

Table 5
Solutions for Problem P4 with multi-objective function *U*. Comparison of solutions when considering or not, the AGV trips and the machine buffers capacities.

CP approach	Neglecting AGV movements and machine buffers capacity	Taking into account AGV loaded and empty movements and machine buffers capacity
U Value	2201	2230[a]
Makespan	437	462
Total number of part movements	11	11
Total processing time	1753	1757
Total processing cost	5100	5220
CPU time	2.8	26.5

[a] Best suboptimal solution instantiated within 1000 s.

Fig. 6 depicts the optimal solution for the problem P2 with a 0.05 *tp* ratio. It shows the machine loading, the manufacturing tasks schedule, the tool used by each operation, the agenda of the machine buffers, as well as the schedule of the loaded and empty vehicle trips.

Problem P4 with a 0.13 *tp* ratio was also solved using the *U* multiobjective function. Table 5 presents the best suboptimal solution obtained when considering AGV trips and buffers capacity. As seen, the makespan value is 26% higher than the one reached without taking into account the AGV movements and the limitations on buffers. It can also be noticed that the computational effort is higher.

The results previously presented show the impact of considering the AGV trips in the model. Even if the vehicle moves quite fast, and the transport times are very small when compared with the processing ones, they cannot be neglected. To demonstrate this assertion, example P2 was modified to obtain a very small *tp* ratio of 0.005 (e.g., 0.5 s of average transport time and 85 s of average processing time). In this case an optimal makespan value of 357 was obtained in only 1.2 s. This value is slightly larger than the completion time reached when neither the vehicle nor the machine buffers were modeled, which was of 355 time units. This result shows that even for a fictitious example, in which transport activities take place very fast, a simplified model can lead to an erroneous solution.

The results presented in this section show the impact of a fully integrated scheduling approach. Since the model is naturally more complex in this case, the computational effort is greater, and it is more difficult to find optimal solutions in reduced CPU times. Nevertheless, the schedules that are obtained in up to 1000 s become realistic and are of good quality. Thus, they could be implemented in the shop floor.

5. Conclusions

This contribution presents a novel CP formulation that addresses the resource-constrained short-term FMS scheduling problem, dealing with several critical features that are present in many FMS environments in an integrated way. To the best of our knowledge, this is the first approach that tackles several limiting resources in a comprehensive formulation that addresses altogether the following problems: (i) machine loading, (ii) manufacturing activities scheduling, (iii) part routing, (iv) machine buffers scheduling, (v) tool planning and allocation, and (vi) AGV scheduling. Regarding this last issue it should be remarked that the model takes into account both loaded and empty trips.

This work stresses the importance of not only dealing with the machine allocation and tasks scheduling problems, but also considering the tool management and AGV issues. After describing the main features associated with the tool management policies, the shortcomings that characterize the solutions in which the AGV trips were neglected have been thoughtfully pointed out. The significance of overcoming these drawbacks has been highlighted.

The proposed CP approach has been tested by resorting to a set of problems that considers dissimilar number of parts, operations per part, and tool copies. Also, each problem has been solved using different AGV speeds, which lead to diverse transport times and, therefore, distinct *tp* ratios. Initially, problem instances were solved considering loaded AGV trips and machine buffers of

limited capacity. Optimal solutions were instantiated for most cases within the imposed time limit. On the contrary, suboptimal schedules were found for a few problems, especially the ones with higher *tp* ratios. Then, the empty AGV trips were also taken into account. When solving this fully-integrated CP model, good quality suboptimal solutions were found for most case studies. Also, some optimal schedules were instantiated for small-size problems and small values of the *tp* ratio. It is worth noticing that the lack of benchmark examples that would consider in an integrated way all the features that this work has addressed, has prevented us to make more comparisons.

It is important to remark that for all the examples, at least one feasible good quality solution was reached within the prescribed time limit. As it was expected, the performance of the approach decreases for large-size problems. Thus, in order to cope with this difficulty, specific domain search strategies will be developed and tested in future work.

The obtained results have shown the impact that the *tp* ratio has on the solution of the integrated FMS scheduling problem. This feature can become more critical with more complex FMS lay-outs. Therefore, it is also matter of future work the performance analysis of the proposed approach when varying the configuration of the FMS. This study will also assess the effect of the relative magnitude of the transport times with respect to the processing ones.

Acknowledgments

The authors acknowledge the financial support received from CONICET (PIP 2754), ANPCYT (PAE-PICT00051), and Universidad Nacional del Litoral (CAI+D 2009, III, R4 – N 12).

References

Atmani, A., & Lashkari, R. S. A. (1998). Model of machine tool selection and operation allocation in FMS. *International Journal of Production Research, 36*, 1339–1349.

Baptiste, P., Le Pape, C., & Nuijten, W. (2001). *Constraint-based scheduling: applying constraint programming to scheduling problems*. New York: Springer+Business Media.

Bilge, Ü., & Ulusoy, G. (1995). A time window approach to simultaneous scheduling of machines and material handling system in an FMS. *Operations Research, 43*, 1058–1070.

Blazewicz, J., Eiselt, H. A., Finke, G., Laporte, G., & Weglarz, J. (1991). Scheduling tasks and vehicles in a flexible manufacturing system. *International Journal of Flexible Manufacturing Systems, 4*, 5–16.

Bourdais, S., Galinier, P., & Pesant, G. (2003). HIBISCUS: A constraint programming application to staff scheduling in health care. *Lectures Notes in Computer Science, 2833*, 153–167.

Brailsford, S. C., Potts, C. N., & Smith, B. M. (1999). Constraint satisfaction problems: algorithm and applications. *European Journal of Operational Research, 119*, 557–581.

Castro, P. M., Grossmann, I. E., & Novais, A. Q. (2006). Two new continuous-time models for the scheduling of multistage batch plants with sequence dependent changeovers. *Industrial & Engineering Chemistry Research, 45*, 6210–6226.

Caumond, A., Lacomme, P., Moukrim, A., & Tchernev, N. (2009). An MILP for scheduling problems in an FMS with one vehicle. *European Journal of Operational Research, 199*, 706–722.

Chan, F. T. S., & Swarnkar, R. (2006). Ant colony optimization approach to a fuzzy goal programming model for a machine tool selection and operation allocation problem in an FMS. *Robotics and Computer-Integrated Manufacturing, 22*, 353–362.

El Khayat, G. E., Langevin, A., & Riopel, D. (2006). Integrated production and material handling scheduling using mathematical programming and constraint programming. *European Journal of Operational Research, 175*, 1818–1832.

Gamila, M. A., & Motavalli, S. A. (2003). Modeling technique for loading and scheduling problems in FMS. *Robotics and Computer-Integrated Manufacturing, 19*, 45–54.

Ganesharajah, T., Hall, N. G., & Sriskandarajah, C. (1998). Design and operational issues in AGV-served manufacturing systems. *Annals of Operations Research, 76*, 109–154.

Grieco, A., Semeraro, Q., & Tolio, T. (2001). A review of different approaches to the FMS Loading Problem. *International Journal of Flexible Manufacturing Systems, 13*, 361–384.

Harjunkoski, I., Jain, V., & Grossman, I. E. (2000). Hybrid mixed-integer/constraint logic programming strategies for solving scheduling and combinatorial optimization problems. *Computers & Chemical Engineering, 24*, 337–343.

ILOG (2000a). ILOG Scheduler 6.0. User's manual. France.

ILOG (2000b). ILOG Solver 6.0. User's Manual. France.

ILOG (2002). ILOG OPL Studio 3.7.1. User's manual. France.

Jerald, J., Asokan, P., Saravanan, R., & Rani, A. D. C. (2006). Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithm. *The International Journal of Advanced Manufacturing Technology, 29*, 584–589.

Liu, J., & MacCarthy, B. L. (1997). A global MILP model for FMS scheduling. *European Journal of Operational Research, 100*, 441–453.

Maravelias, C. T., & Grossmann, I. E. (2004). A hybrid MILP/CP decomposition approach for the continuous time scheduling of multipurpose batch plants. *Computers & Chemical Engineering, 28*, 1921–1949.

Novas, J. M., & Henning, G. P. (2010). Reactive scheduling framework based on domain knowledge and constraint programming. *Computers & Chemical Engineering, 34*, 2129–2148.

Öztürk, C., Tunali, S., Hnich, B., & Örnek, A. M. (2012). A Constraint Programming Model for Balancing and Scheduling of Flexible Mixed Model Assembly Lines with Parallel Stations. In T. Borangiu, A. Dolgui, I. Dumitrache, F. G. Filip (Eds.), *Proc. 14th IFAC symposium of information control problems in manufacturing* (pp. 420–425).

Prakash, A., Chan, F. T. S., & Deshmuck, S. G. (2011). FMS scheduling with knowledge based genetic algorithm approach. *Expert Systems with Applications, 38*, 3161–3171.

Rodriguez, J. (2007). A constraint programming model for real-time train scheduling at junctions. *Transportation Research Part B: Methodological, 41*, 231–245.

Topaloglu, S., Salum, L., & Supciller, A. A. (2012). Rule-based modeling and constraint programming based solution of the assembly line balancing problem. *Expert Systems with Applications, 39*, 3484–3493.

Ulusoy, G., Sivrikaya-Serfioglu, F., & Bilge, Ü. (1997). A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. *Computers & Operations Research, 24*, 335–351.

Vis, I. F. A. (2006). Survey of research in the design and control of automated guided vehicle systems. *European Journal of Operational Research, 170*, 677–709.

Zeballos, L. J. (2010). A constraint programming approach to tool allocation and production scheduling in flexible manufacturing systems. *Robotics and Computer-Integrated Manufacturing, 26*, 725–743.

Zeballos, L. J., Novas, J. M., & Henning, G. P. (2011). A CP formulation for scheduling multiproduct multistage batch plants. *Computers & Chemical Engineering, 35*, 2973–2989.

Zeballos, L. J., Quiroga, O. D., & Henning, G. P. (2010). A constraint programming model for the scheduling of flexible manufacturing systems with machine and tool limitations. *Engineering Applications of Artificial Intelligence, 23*, 229–248.