A Control Method for Distributed Continuous Mono-T-Semiflow Petri nets

-draft-

Hanife Apaydin-Özkan, Cristian Mahulea, Jorge Júlvez, Manuel Silva [*]

December 23, 2014

Abstract

A distributed continuous Petri net system (DCPN) is composed of several subsystems which are interconnected through buffers that are origin and destination private (i.e., modeled by places such that its input transitions belong to only one subsystem and its output transitions also belong to only one subsystem). In this framework, the control problem of driving a DCPN from an initial state to a target one is considered. Such a control problem can be divided into two tasks: (i) computation of a firing count vector that ensures the reachability of the target marking in the untimed subsystem; (ii) implementation of a local control law executing the computed firing count vector in every timed subsystem. This work mainly focuses on the first task which is achieved by the network of interconnected local coordinators located at subsystems. In particular, the firing count vectors are obtained as a result of a negotiation among those local coordinators. A distributed algorithm that implements the negotiation performed by each coordinator is presented and it is formally proved that the subsystems reach the target marking.

Published as:
H. Apaydin-Ozkan, C. Mahulea, J. Julvez, and M. Silva, "A Control Method for Distributed Continuous Mono-T-Semiflow Petri nets," International Journal of Control, vol. 87, no. 2, pp. 223-234, 2014. DOI: 10.1080/00207179.2013.826822

[*]Corresponding author is: Cristian Mahulea, María de Luna 1, 50018 Zaragoza, Spain, tel. + (34) 976 76 2517, Fax: + (34) 976761914, e-mail: {cmahulea@unizar.es}. This work has been partially supported by CICYT - FEDER grant DPI2010-20413. This paper is based on our results in [4] and [6]. The authors thank L. Wang for his help with the simulations.
The authors are with Aragón Institute of Engineering Research (I3A), University of Zaragoza, Spain

1 Introduction

A recurrent difficulty in the analysis and synthesis of populated discrete event systems is the so called *state explosion problem.* This well known problem makes the use of many analysis and verification techniques computationally prohibitive when applied to many systems of interest in practice. One way of overcoming such a problem is to relax the original discrete model and deal with a continuous approximation of it. In the Petri net framework this leads to continuous Petri nets [10, 28]. Continuous Petri nets have been used in many domains, e.g., manufacturing systems [19, 2, 1], traffic systems [13] or supply chains by using hybrid extension [11]. They may be studied by means of net based structural analysis techniques [7, 14, 26]. Many works exist in the literature dealing with the control of continuous timed or hybrid Petri nets [19, 20, 7, 5]. In many cases, the system is distributed and the controller cannot have access to all variables. Thus a distributed controller must be considered. Moreover, it is frequently the case when limited communication among controllers is available and they can just communicate locally with their *neighbors.* Therefore, it is not possible to implement a single coordinator for the whole system and local coordinators for the different subsystems must be designed.

The main goal of this paper is to propose a control method for distributed systems modeled by continuous Petri nets. This paper primarily focuses in developing an algorithm that computes firing amounts for each subsystem that ensure the global reachability of the desired target marking. Depending on the initial and final states and the structures, the major problems that can arise are related to the consensus that must be reached by subsystems in order to provide enough tokens in the buffers and the global reachability of the final state.

Several approaches have been presented to model modular discrete event dynamical systems, e.g., [31, 23, 9, 21]. In [23], the class of Deterministically Synchronized Sequential Processes (DSSP) is proposed. In such a class, each process is modeled by a state machine and the communicating buffers are output private (a necessary condition to be distributable). This is different with respect to our approach since here we deal with different structures for subsystems, and not only the output but also the input transitions of each buffer must belong to only one subsystem, i.e., buffers are output (input) private. The work in [31] deals with a class of discrete nets, called Extended-Controllable-Output (ECO) nets, whose composition retains liveness and reversibility. In contrast to DCPN, ECO nets must by acyclic. In [9], analysis techniques that can be carried out modularly are described. In such a work, modules can share places and transitions, this is not allowed in our framework since the overall system is distributed and therefore each subsystem could be in a different location. In [21], a particular methodology for modular modeling of manufacturing systems using PNs is presented. The modular strategy takes into account the shared resources and the alternative process planning, but it is not related to distributability.

There exist some works in the literature about the control of distributed (timed) discrete event systems. For example, distributed timed automata is defined in [16] while an optimal planning using weighted automata calculus is presented in [12]. A supervisory control for a distributed manufacturing process is studied by using discrete Petri nets in [8], while an architecture for distributed implementation of Petri nets in control applications is proposed in [22]. In [15], timed Petri nets are used for supervisory control and stability purposes. The work in [3] is a survey on hybrid control systems and on approaches to model hybrid systems, most of which are timed but not distributed. A synthesis method of a coordinated control strategy for large CPN systems that can be seen as a set of T-disjoint modules interconnected by places has been presented in [29]. In such paper, it is required to compute the set of vertices of the reachability set what might increase the computational complexity. This paper extends the results presented in [4] and [6] by considering a more general framework.

The kind of systems considered in this work are composed by several continuous net *subsystems* that could model different parts of a plant. They are interconnected by *buffers* modeled by places. These buffers contain the items produced by a given subsystem and needed by another one. As a basic introductory example let us consider a simplified car manufacturing factory composed by two subsystems in two different cities. The net model is given in Fig. 1. The subsystem 1 produces car bodies (place p_1) and then sends them to the subsystem 2 (transition t_3). The subsystem 1 can produce a limited number of car bodies in parallel (the initial marking of p_3). In subsystem 2, the engines are assembled (p_4) and then it is put in an intermediate deposit p_5. The same plant paints the body received from subsystem 1 in p_6 and puts it in p_7 to be assembled together with the engine. The firing of t_8 means the production of a new car. We assume that subsystem 2 can produce a limited number of engines in parallel (initial marking of p_8) and can paint a limited set of car bodies in parallel (initial marking of p_9). Place b_b is the buffer containing the car bodies produced by subsystem 1 while b_a is the maximum number of cars that can be in fabrication simultaneously.

The buffers connecting subsystems are assumed to be *input* and *output private*, i.e., given a buffer b, only one subsystem i can put tokens in b, and only one subsystem j can remove tokens from b. We consider that each subsystem

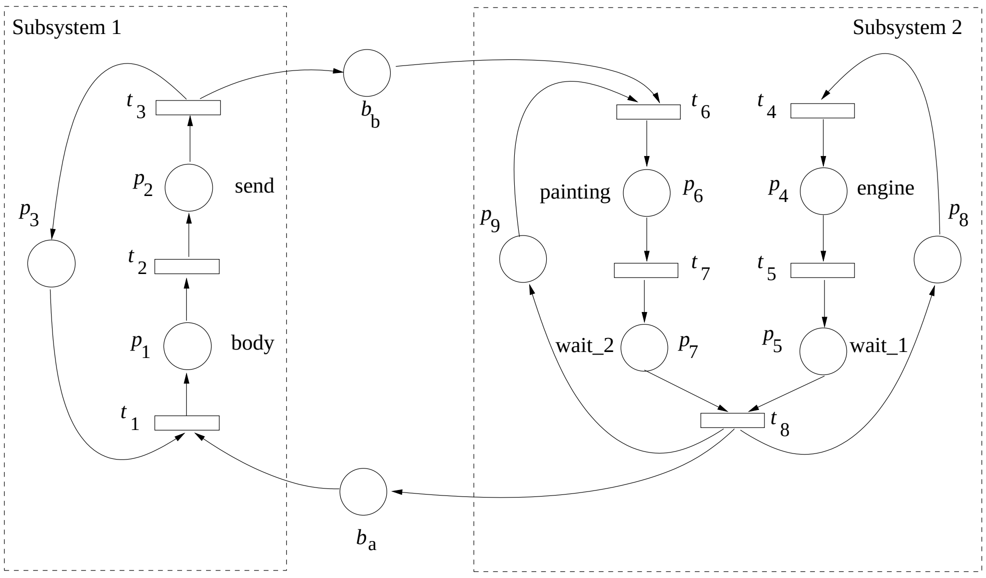


Figure 1: A DCPN marked graph modeling a car manufacturing plant where b_a and b_b are buffer places.

k knows its structure and initial marking, and that the structure and marking of the rest of subsytems $i \neq k$ are unknown by subsystem k. Each subsystem has its own goal which consists of reaching a given local target marking. Notice that in order to reach a given target local marking by subsystem k, it might be necessary that other subsystems produce enough tokens in its input buffers. The method is required to drive the distributed system to a final global marking in which all target local markings are reached.

Being the system distributed, subsystems do not have access to the state variables of the other subsystems, and therefore it is impossible that they compute global control laws to reach desired markings and a decentralized solution should be developed. In this paper we consider a particular class of continuous Petri nets, called mono-T-semiflow (bounded net systems with only one repetitive behavior) and we propose a decentralized algorithm that will be executed by a coordinator at each subsystem. It is proved that the decentralized firing amounts coincide with the firing amount obtained if the problem would have been solved as centralized.

The remainder of the paper is organized as follows: Section 2 introduces DCPN. The control problem under consideration is presented and a control algorithm is proposed in Section 3. A case study consisting of a five Automated Guided Vehicle (AGV) system is shown in Section 4. Finally, Section 5 summarizes the main conclusions of the work.

2 Distributed Continuous Petri nets

Definition 1. *A continuous Petri net system is a pair $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle$, where $\mathcal{N} = \langle P, T, \boldsymbol{Pre}, \boldsymbol{Post} \rangle$ is the net structure with: P and T the sets of places and transitions; $\boldsymbol{Pre}, \boldsymbol{Post} \in \mathbb{R}_{\geq 0}^{|P|\times|T|}$ the pre and post matrices; $\boldsymbol{m}_0 \in \mathbb{R}_{\geq 0}$ the initial marking.*

For $v \in P \cup T$, the sets of its input and output nodes are denoted as ${}^{\bullet}v$ and $v^{\bullet}$, respectively. Let p_i, $i = 1, \ldots, |P|$ and $t_j, j = 1, \ldots, |T|$ denote the places and transitions. Each place can contain a non-negative real number of tokens, this number represents the marking of the place. The distribution of tokens in places is denoted by $\boldsymbol{m}$. A transition $t_j \in T$ is enabled at $\boldsymbol{m}$ if $\forall p_i \in^{\bullet} t_j$, $\boldsymbol{m}(p_i) > 0$ and its enabling degree is given by

$$enab(t_j, \boldsymbol{m}) = \min_{p_i \in^{\bullet} t_j} \left\{ \frac{\boldsymbol{m}(p_i)}{\boldsymbol{Pre}(p_i, t_j)} \right\} \tag{1}$$

which represents the maximum amount in which t_j can fire at $\boldsymbol{m}$. An enabled transition t_j can fire in any real amount α, with $0 < \alpha \leq enab(t_j, \boldsymbol{m})$ leading to a new state $\boldsymbol{m'} = \boldsymbol{m} + \alpha \cdot \boldsymbol{C}(\cdot, t_j)$ where $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$ is the *token flow matrix* and $\boldsymbol{C}(\cdot, j)$ is its j^{th} column.

If $\boldsymbol{m}$ is reachable from $\boldsymbol{m}_0$ through a finite sequence $\sigma = t_{j_1}(\alpha_1) t_{j_2}(\alpha_2) \ldots t_{j_p}(\alpha_p)$, the state (or fundamental) equation is satisfied: $\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{s}$, where $\boldsymbol{s} \in \mathbb{R}_{\geq 0}^{|T|}$, $s_j = \sum\limits_{i=1,\ldots,p; j_i = j} \alpha_i$, is the *firing count vector*, i.e., s_j is the cumulative amount of firings of t_j in the sequence σ.

Left and right natural annullers of the token flow matrix $\boldsymbol{C}$ are called *P-semiflows* (denoted by $\boldsymbol{y}$) and *T-semiflows* (denoted by $\boldsymbol{x}$), respectively. If $\exists\, \boldsymbol{y} > 0,\ \boldsymbol{y} \cdot \boldsymbol{C} = 0$, then the net is said to be *conservative.* If $\exists\, \boldsymbol{x} > 0,\ \boldsymbol{C} \cdot \boldsymbol{x} = 0$ it is said to be *consistent.* Intuitively, a T-semiflow corresponds to a potential repetitive behavior of the system, i.e., if it is executed from a given marking $\boldsymbol{m}$ the same marking $\boldsymbol{m}$ is reached. The support of a vector $\boldsymbol{v}$ is the set of nonzero components. A semiflow $\boldsymbol{v}$ is said to be *minimal* when its support is not a proper superset of any other and the greatest common divisor of its elements is one.

Definition 2 (MTS). *A (continuous) Petri net is* mono T-semiflow *(MTS) if it is conservative, consistent and has only one minimal T-semiflow.*

Definition 3 (DCPN). *A* distributed (continuous) Petri net system *is a (continuous) Petri net composed of: (i) a set of (continuous) Petri net systems called* subsystems*; (ii) a set of* buffers *modeled as places that connect subsystems; (iii) each buffer is* input *and* output private*, i.e., given a buffer b, only one subsystem i can put tokens in b, and only one subsystem j can remove tokens from b.*

Let K denote the set of subsystems of a given DCPN. The set of places, transitions and token flow matrix of subsystem $k \in K$ is denoted by P^k, T^k and $\boldsymbol{C}^k = \boldsymbol{Post}^k - \boldsymbol{Pre}^k \in \mathbb{R}^{|P^k|\times|T^k|}$, respectively. The partition into disjoint subsystems leads to $P^k \cap P^l = \emptyset$ and $T^k \cap T^l = \emptyset$, $\forall k,l \in K, k \neq l$. The directional connection between subsystems is provided by a set of places, B, called *buffers*. In particular, the directional connection from subsystem k to l is provided by a set of places denoted by $B^{(k,l)}$, whose input transitions are contained only in subsystem k and output transitions are contained only in subsystem l, i.e., $B^{(k,l)} = \{b \in P|{}^\bullet b \in T^k,\ b^\bullet \in T^l, b \notin P^q, \forall q \in K\}$ for every $k,l \in K$, $k \neq l$, and $B^{(l,l)} = \emptyset$ for every $l \in K$.

Note that $b \in B^{(k,l)}$ is *input buffer* of subsystem l and *output buffer* of subsystem k. The set of all output buffers of subsystem k is denoted by $B^{(k,*)}$, i.e., $B^{(k,*)} = \bigcup_{l\in K} B^{(k,l)}$, and the set of all input buffers of subsystem k is denoted by $B^{(*,k)}$, i.e., $B^{(*,k)} = \bigcup_{l\in K} B^{(l,k)}$. The marking vector of subsystem k is denoted by $\boldsymbol{m}(P^k) \in \mathbb{R}^{|P^k|}_{\geq 0}$. This vector is called *local marking* of subsystem k. In contrast, we call the *global marking* the vector $\boldsymbol{m}$ containing all local markings and the markings of the buffer places.

A firing count vector $\boldsymbol{\sigma}^k \in \mathbb{R}^{|T^k|}_{\geq 0}$ is *locally fireable* if there exists a fireable sequence σ^k in the PN $\langle P^k, T^k, \boldsymbol{Post}^k, \boldsymbol{Pre}^k\rangle$ with initial marking $\boldsymbol{m}_0(P^k)$ whose firing count vector is $\boldsymbol{\sigma}^k$. A firing count vector $\boldsymbol{\sigma} \in \mathbb{R}^{|T|}_{\geq 0}$ is *globally fireable* if there exists a fireable sequence σ in the PN $\langle P, T, \boldsymbol{Post}, \boldsymbol{Pre}\rangle$ with initial marking $\boldsymbol{m}_0$ whose firing count vector is $\boldsymbol{\sigma}$. We say that marking $\boldsymbol{m}(P^k) \in \mathbb{R}^{|P^k|}_{\geq 0}$ is *locally reachable* if there exists a locally fireable firing count vector $\boldsymbol{\sigma}^k \in \mathbb{R}^{|T^k|}_{\geq 0}$ such that $\boldsymbol{m}(P^k) = \boldsymbol{m}_0(P^k) + \boldsymbol{C}^k \cdot \boldsymbol{\sigma}^k$. A marking $\boldsymbol{m} \in \mathbb{R}^{|P|}_{\geq 0}$ is *globally reachable* if there exists a globally fireable firing count vector $\boldsymbol{\sigma} \in \mathbb{R}^{|T|}_{\geq 0}$ such that $\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}$.

3 Design of a distributed coordinator for DCPN

3.1 Control Architecture

Our control problem aims at reaching a particular target marking $\boldsymbol{m}_f$ at each subsystem. Depending on the interpretation given to the buffers, their desired final markings can also be specified or not. Here, it is considered that the final marking of buffers is not relevant, being any non-negative value appropriate as final marking (if final markings of the buffers are required the method can be easily extended by considering that the output buffers are part of subsystems). In some practical applications, it is useful to specify a minimum number of times a given activity (transition) has to be executed. In order to consider this, the control problem is enriched by considering that some transitions must fire a number of times greater than a given value. For this purpose we define a vector $\bar{\boldsymbol{z}} \geq \mathbf{0}$, where $\bar{z}(t_i)$ is the minimum quantity of required firings of t_i.

In contrast to a centralized control, each subsystem is equipped with its own coordinator that computes the firing count vector that drives the subsystem to the target marking. We naturally assume that each subsystem knows only its marking and the marking of its input buffers. The marking of the other subsystems and their output buffers are unknown.

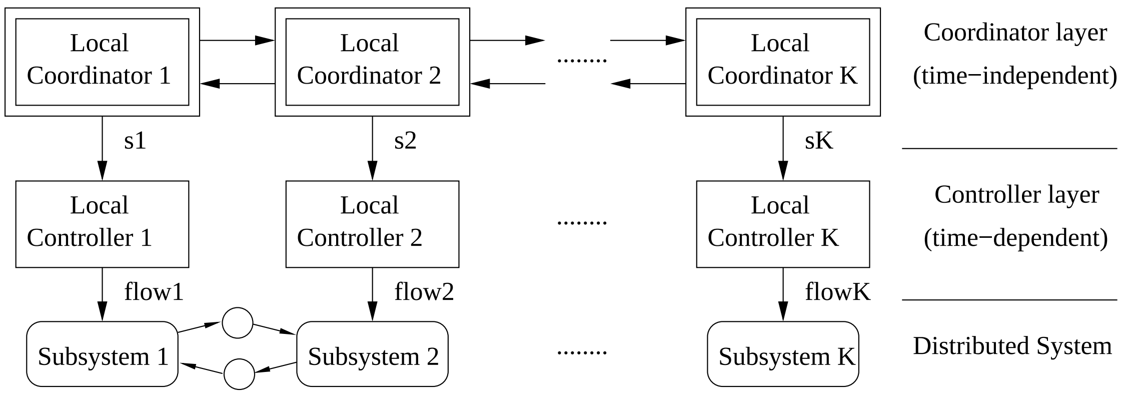


Figure 2: Sketch of the control approach for DCPN. Coordination layer on the top, controller layer in the middle and subsystems modeled as Petri nets and connected by buffers at the bottom.

Fig. 2 sketches the proposed architecture of the control for the DCPN. The control architecture mainly consists of two layers: the coordinator and the controller layers. According to this architecture, a local coordinator and a local controller is associated to each subsystem. The coordinator layer is responsible for computing feasible firing count vectors whose implementation should drive the subsystems to the specified target markings. The controller layer implements these firing count vectors for a given time interpretation, i.e., firing semantics. For such purpose there exist several strategies, e.g., an ON-OFF controller [30], model predictive control [20], etc. This work focuses on the coordinator layer and proposes an iterative technique to compute the mentioned firing count vectors in the untimed systems. In section 4 we illustrate how the implementation of the control, once the firing count vector is computed, can be carried out with an ON-OFF strategy.

Since each local coordinator will compute its own firing count vector, in order to obtain a firing vector that permits the global reachability of the desired marking, some interchange of information, i.e., negotiation, between subsystems must be considered. Obviously, it is desirable that coordinators exchange as few information as possible. The underlying idea of the strategy is to design a local coordinator for each subsystem. Each coordinator will compute a firing count vector independently of the time interpretation to reach the target state of its subsystem, and will ask the other coordinators to produce enough resources in the buffers to execute its control actions.

There are two basic problems that must be addressed when designing local coordinators for the described framework: (i) given that the subsystems are interconnected, they may require resources, i.e., tokens, to be available in the communication buffers to reach the target marking; (ii) another problem that can appear is the impossibility of reaching the target marking due to the system structure or the initial marking of the buffers.

Example 3.1. *Consider the DCPN in Fig. 1 with* $\boldsymbol{m}_0(P^1) = [0\ 0\ 3]^T$ *for subsystem 1,* $\boldsymbol{m}_0(P^2) = [0\ 0\ 0\ 0\ 2\ 2]^T$ *for subsystem 2,* $m_0(b_a) = 1$, $m_0(b_b) = 0$ *and let* $\boldsymbol{m}_f(P^1) = [0\ 0\ 3]^T$, $\boldsymbol{m}_f(P^2) = [0\ 0\ 1\ 0\ 2\ 1]^T$ *be the target markings of each subsystem. Let the firing count vectors of subsystem 1 and 2 be denoted as* $\boldsymbol{s}^1$ *and* $\boldsymbol{s}^2$ *respectively.*

A coordinator for the second subsystem could compute $s^2(t_6) = 1$, $s^2(t_4) = s^2(t_5) = s^2(t_7) = s^2(t_8) = 0$ *so that the subsystem reaches the target marking. Given that the initial and target markings of subsystem 1 are the same, a coordinator for that subsystem could yield:* $s^1(t_1) = s^1(t_2) = s^1(t_3) = 0$. *Since* $m_0(b_b) = 0$, *transition* t_6 *cannot fire unless* t_3 *fires. Unfortunately, according to the computed controls,* t_3 *will not fire* ($s^1(t_3) = 0$). *Hence, these controls are not valid to reach the desired target marking of subsystem 2. In order to solve this situation, subsystem 2 may ask subsystem 1 to put enough tokens in* b_b. *This can be achieved easily by firing* t_3. *However, this will imply that subsystem 1 moves away from its desired target marking.*

Consider now $m_0(b_a) = m_0(b_b) = 0$ *and the same initial and target markings for subsystems. In this case, the target markings are locally reachable but not globally reachable.* ■

Our control objective is not only to reach the desired target marking at each subsystem, but also to minimize the overall number of times the transitions are fired. This implies minimum movement of tokens from and to buffer places.

3.2 Distributed coordinator

This subsection is devoted to the design of a distributed coordinator for a DCPN. We first present the algorithm associated to the local coordinator of each subsystem and then prove its correctness. The control algorithm applies to those DCPN systems satisfying the following assumptions:

(A1) The DCPN is composed of $|K|$ MTS disjoint subsystems. The minimal T-semiflows are denoted by $\boldsymbol{x}^k$, $k \in \{1, 2, ..., |K|\}$.

(A2) The overall system is MTS.

(A3) Every transition is fireable from $\boldsymbol{m}_0$.

The first assumption reduces the class of DCPN to those systems having MTS subsystems, i.e., consistent and conservative nets with one T-semiflow. Notice that consistency and conservativeness are necessary and sometimes sufficient conditions for the often desired properties of structural liveness and boundedness [24, 25]. Thus, in addition to these properties, systems are just assumed to have only one minimal T-semiflow. Notice that the class of MTS net systems subsumes marked graph systems (PN in which any place has only one input and one output transition), weighted-T systems (marked graph with weighted arcs), structural persistent PN (all places has at most only one output transitions). Moreover, because the marked graph class is included in MTS systems, all systems linearly modeled in (max,+) algebra can be modeled as MTS net systems.

Assumption (A2) implies that the overall system is MTS, since this is a structural property, it just needs to be checked once at the beginning of the process. The last assumption ensures that each transition can eventually fire. One way of easily checking this assumption is by solving a linear programming problem (LPP) that looks for initially empty siphons [27]. Together with (A2) it guarantees the absence of spurious solutions in the continuous subsystem [24].

The coordinator algorithm that will be executed in each subsystem separately is given in Alg. 3.3. Each coordinator executes asynchronously and communicates with neighbor subsystems by means of *communication channels*. The information sent through these channels is used to meet an agreement between coordinators and ensure that the final marking is globally reached. Each coordinator has a queue where the request from neighbor subsystems are kept. In each iteration one element of the queue is processed, if the queue is empty the coordinator waits until a request is queued. This represent an implicit synchronization among subsystems avoiding the need of a global clock.

Example 3.2. *Let us consider again the net system in Fig. 1 used also in Ex. 3.1. Assume for the first subsystem the same initial and desired markings:* $\boldsymbol{m}_0(P^1) = \boldsymbol{m}_f(P^1) = [0\ 0\ 3]^T$ *while for the second one:* $\boldsymbol{m}_0(P^2) = [0\ 0\ 0\ 0\ 2\ 2]^T$ *and* $\boldsymbol{m}_f(P^2) = [0\ 0\ 1\ 0\ 2\ 1]^T$*. For the buffers, let us assume* $m_0(b_a) = 1$*,* $m_0(b_b) = 0$ *and let* $\bar{\boldsymbol{z}} = \boldsymbol{0}$*, i.e., no minimum firing amount is required.*

Let us compute a firing vector in the global system to reach the final marking assuming $m_f(b_a) = m_f(b_b) = 0$*. Being the system mono-T-semiflow, the minimal firing count vector is unique and it is obtained by solving: min* $\boldsymbol{1}^T \cdot \boldsymbol{s}_G$ *subject to* $\boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{s}_G, \boldsymbol{s}_G \geq 0$*. Being the net MTS, the unique solution of this LPP is*

$$\boldsymbol{s}_G = [1\ 1\ 1 \mid 0\ 0\ 1\ 0\ 0]^T,$$

where the first three components correspond to the transitions belonging to the subsystem 1 and the last five ones to transitions belonging to the subsystem 2. Observe also that the firing vectors correspond to firing sequences that can be locally fired.

Now, let us compute local control laws in each subsystem. For the first one, since $\boldsymbol{m}_0(P^1) = \boldsymbol{m}_f(P^1)$*, the minimum firing vector is unique and equal to* $\boldsymbol{s}^1 = [0\ 0\ 0]^T$*, i.e., not firing any transition. For the second subsystem, it is easy to observe that the minimum firing vector is* $\boldsymbol{s}^2 = [0\ 0\ 1\ 0\ 0]^T$*, i.e., firing* t_6 *in an amount equal to* 1*.*

Observe that $\boldsymbol{s}^2$ *is exactly equal to the projection of* $\boldsymbol{s}_G$ *to the set of transitions of subsystem 2 while* $\boldsymbol{s}^1$ *is not. However, it is always possible to fire once the T-semiflow of subsystem 1 (equal to the vector of ones). Because a T-semiflow corresponds to a repetitive behavior, adding to any firing vector the T-semiflow, the same final marking is obtained. The idea of Alg. 3.3 is to ensure the firing of the T-semiflows of the subsystems (if it is necessary) such that if the local laws computed are put together, the minimum firing vector of global system is obtained. Hence, the global marking will be reached. This consensus between subsystems is obtained by using the marking of buffers since the firing of the T-semiflow produces some tokens in the output buffers. Being the net mono-T-semiflow, there exists only one way to produce tokens in the output buffers without modifying the target subsystem marking, i.e., firing the T-semiflow.*

The input parameters of the algorithm are: the token flow matrix, initial and target markings, input and output buffers, their initial markings and the minimum quantity of firing required for each transition (denoted by $\bar{\boldsymbol{z}} \geq 0$, where

Algorithm 3.3. *[Distributed coordinator of subsystem k].*
Input: $\boldsymbol{C}^k, \boldsymbol{m}_0(P^k), \boldsymbol{m}_f(P^k)$, $B^{(k,*)}$, $B^{(*,k)}$, $\boldsymbol{m}_0(B^{(k,*)})$, $\bar{\boldsymbol{z}}(T^k) \geq \boldsymbol{0}$
Output: *firing count vector* $\boldsymbol{s}$

```
1. Solve
$$\begin{aligned} \min \quad & \boldsymbol{1}^T \cdot \boldsymbol{z} \\ s.t. \quad & \boldsymbol{m}_f(P^k) - \boldsymbol{m}_0(P^k) = \boldsymbol{C}^k \cdot \boldsymbol{z} \\ & \boldsymbol{z} \geq \bar{\boldsymbol{z}}(T^k) \end{aligned} \qquad (2)$$

2. For Iteration=1 to $|K| - 1$ do

    (a) For every $p \in B^{(*,k)}$ calculate $q_p^{req} = (\sum_{t \in p^\bullet} \boldsymbol{Pre}(p,t) \cdot \boldsymbol{z}(t)) - \boldsymbol{m}_0(p)$

    (b) For all $p \in B^{(*,k)}$ send $q_p^{req}$ to the connected subsystem

    (c) For all $p \in B^{(k,*)}$ receive $r_p^{req}$ from the connected subsystem

    (d) For all $p \in B^{(k,*)}$ calculate $h_p = (\sum_{t \in {}^\bullet p} \boldsymbol{Post}(p,t) \cdot \boldsymbol{z}(t)) - r_p^{req}$

    (e) If $\min_{p \in B^{(k,*)}} \{h_p\} < 0$ then solve

$$\begin{aligned} \min \quad & \boldsymbol{1}^T \cdot \boldsymbol{s} \\ s.t. \quad & \boldsymbol{m}_f(P^k) - \boldsymbol{m}_0(P^k) = \boldsymbol{C}^k \cdot \boldsymbol{s} \\ & \left(\sum_{t \in {}^\bullet p} \boldsymbol{Post}(p,t) \cdot s(t)\right) \geq r_p^{req}, \forall p \in B^{(k,*)} \\ & \boldsymbol{s} \geq \bar{\boldsymbol{z}}(T^k) \end{aligned} \qquad (3)$$

        Else $\boldsymbol{s} = \boldsymbol{z}$
        End_If

    (f) $\boldsymbol{z} = \boldsymbol{s}$

   End_For

3. return $\boldsymbol{s}$
```

$\bar{z}(t_i)$ is the minimum quantity of required firings of t_i). In step 1, each subsystem computes the firing count vector $\boldsymbol{z}$ required to reach its target marking without taking into account the marking of the buffers, this step is carried out by means of the LPP (2).

Step 2 implements a loop executed $|K| - 1$ times, where $|K|$ is the number of subsystems. Step 2.a) computes the amounts of tokens q_p^{req} to be produced in each input buffer p in order to be able to fire $\boldsymbol{z}$. The connected subsystems send information about the amounts of required tokens q_p^{req} in step 2.b). This information is sent through communication channels and remains queued until it is got by the receiver. In step 2.c), each subsystem gets information from the communication channels about the amount of tokens it has to produce in its output buffers. The *receive* command in step 2.c) is understood as a blocking instruction, i.e., the algorithm is not executed further until the required amount of tokens is available in all the incoming communication channels. This ensures that all requests are properly processed. Step 2.d computes the number of tokens that would remain in each output buffer if $\boldsymbol{z}$ was applied. If this value is negative, more tokens must be produced in the output buffers, and therefore the firing count vector must be recomputed. This re-computation is achieved in step 2.e) using LPP (3). Observe that comparing with LPP (2) of step 1 as many constraints as there are output buffers are added in order to ensure that enough tokens are produced in the output buffers.

Observe that in general, the longest path between a couple of subsystems is less than or equal to $|K| - 1$. Hence, the number of iterations that the iterative algorithm must perform is $|K| - 1$. In each iteration, each local coordinator solves a LPP (3) for which there exist polynomial time algorithms. Because initially, LPP (2) is solved by each subsystem, the maximum number of LPP solved is $|K|^2$. Therefore, the complexity of the algorithm is polynomial in the number of subsystems. After the execution of the algorithm, if one h_p is still negative then the desired marking is not globally reachable.

In the following we will prove the correctness of the algorithm.

Lemma 3.4. *Let $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle$ be a MTS system and $\boldsymbol{m}_f$ a reachable marking. Then, the LPP:*

$$\begin{aligned} \min \quad & \mathbf{1}^T \cdot \boldsymbol{\sigma} \\ s.t. \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma} \\ & \boldsymbol{\sigma} \geq \bar{\boldsymbol{z}} \end{aligned} \tag{4}$$

has a unique solution, and there exists t such that $\boldsymbol{\sigma}(t) = \bar{\boldsymbol{z}}(t)$.

Proof. Let us show that the constraints of the LPP are feasible. The first constraint is the state equation which is a necessary condition for reachability.

The second constraint imposes a minimum number of firings of transitions, this number can always be satisfied by firing the unique minimal T-semiflow as many times as neeeded. On the other hand, assume that the LPP has two solutions $\boldsymbol{\sigma}_a$ and $\boldsymbol{\sigma}_b$, then, $\boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma}_a$ and $\boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma}_b$, therefore $\boldsymbol{C}\cdot(\boldsymbol{\sigma}_a - \boldsymbol{\sigma}_b) = \boldsymbol{0}$ and hence $(\boldsymbol{\sigma}_a - \boldsymbol{\sigma}_b)$ is a T-semiflow. Given that the system is MTS, it holds that there exists a real $\gamma \neq 0$ such that $\boldsymbol{\sigma}_a = \boldsymbol{\sigma}_b + \gamma\cdot\boldsymbol{x}$ where $\boldsymbol{x}$ is the T-semiflow. Thus, either $\boldsymbol{\sigma}_a$ or $\boldsymbol{\sigma}_b$ is not minimizing the objective function. □

Lemma 3.5. *Let $\langle\mathcal{N}, \boldsymbol{m}_0\rangle$ be a DCPN satisfying assumptions* (*A*1), (*A*2) *and* (*A*3) *and $\boldsymbol{m}_f$ be the target marking of subsystems. For the following LPP:*

$$\begin{array}{ll} \min & \boldsymbol{1}^T\cdot\boldsymbol{\sigma} \\ s.t. & \boldsymbol{v} = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma} \\ & \boldsymbol{v}(p) = \boldsymbol{m}_f(p), \quad \forall\, p \in P^i, i \in \{1,\dots,|K|\} \\ & \boldsymbol{v}(p) \geq 0, \quad \forall\, p \in B \\ & \boldsymbol{\sigma} \geq \bar{\boldsymbol{z}} \end{array} \tag{5}$$

it holds that: a) if $\boldsymbol{m}_f$ is not reachable then (5) *is not feasible; b) if $\boldsymbol{m}_f$ is reachable then* (5) *is feasible and has a unique solution.*

Proof. a) Given that the system is consistent and every transition is assumed to be fireable, the state equation of the continuous net system does not contain spurious solutions, i.e., it represents a necessary and sufficient condition for reachability [24]. Hence, if $\boldsymbol{m}_f$ is not reachable then the first constraint of the LPP is not feasible.
b) Since every subsystem is MTS, LPP (5) is equivalent to:

$$\begin{array}{ll} \min & \sum_{i=1}^{|K|}\alpha_i \\ \text{s.t.} & \boldsymbol{v} = \boldsymbol{m}_0 + \boldsymbol{C}\cdot\boldsymbol{\sigma} \\ & \boldsymbol{\sigma} = [\boldsymbol{\sigma}^1 + \alpha_1\cdot\boldsymbol{x}_1 \;\; \boldsymbol{\sigma}^2 + \alpha_2\cdot\boldsymbol{x}_2 \;\; \dots] \\ & \boldsymbol{v}(p) = \boldsymbol{m}_f(p), \quad \forall\, p \in P^k \\ & \boldsymbol{v}(p) \geq 0, \quad \forall\, p \in B \\ & \boldsymbol{\sigma} \geq \bar{\boldsymbol{z}} \end{array} \tag{6}$$

where vectors $\boldsymbol{\sigma}^i$ are obtained from the solution of LPP (4) applied to each subsystem i. LPP is feasible given that $\boldsymbol{m}_f$ is reachable and the system is consistent and the transitions fireable. Let us see that there exists $i \in \{1,\dots,|K|\}$ such that $\alpha_i = 0$. Assume that $\alpha_i > 0$ for every $i \in \{1,\dots,|K|\}$, then there would exist $\boldsymbol{\sigma}_b = \boldsymbol{\sigma} - \rho\cdot\boldsymbol{x}$ with $\rho > 0$ and $\boldsymbol{x}$ the minimal T-semiflow of the whole system (which is unique and necessarily exists due to assumption (A2)) that is also solution of (6). This would imply that $\boldsymbol{\sigma}$ is not giving the minimum of the objective function. The interpretation $\alpha_i = 0$ is that subsystem i does not have to produce tokens in its output buffers. Notice that the set of such subsystems just depends on $\boldsymbol{m}_0$, $\boldsymbol{m}_f$ and $\bar{\boldsymbol{z}}$. The same reasoning can be applied to subsystems having a positive α_i, those subsystems will fire in the minimum amount in order to satisfy the constraints being that amount depending exclusively on $\boldsymbol{m}_0$, $\boldsymbol{m}_f$ and $\bar{\boldsymbol{z}}$. This implies that the solution is necessarily unique. $\square$

Notice that, according to the previous Lemma, reachability of $\boldsymbol{m}_f$ is a necessary and sufficient condition for the feasibility of LPP (5). Moreover, if LPP (5) is feasible, the solution is unique.

Theorem 3.6. *Let $\langle \mathcal{N}, \boldsymbol{m}_0 \rangle$ be a DCPN satisfying assumptions $(A1)$, $(A2)$ and $(A3)$ and $\boldsymbol{m}_f$ a reachable global marking of subsystems. The firing count vector computed by Alg. 3.3 is equal to the solution of LPP* (5)*.*

Proof. At each iteration one firing count vector per subsystem is obtained. Notice that since subsystems are MTS, in each iteration the initial firing count vector $\boldsymbol{z}^i$ is modified by adding a given amount of the T-semiflow. We can express the global vector at the end of iteration j as $\boldsymbol{\sigma}_j = [\boldsymbol{z}^1 + \beta_1^j \cdot \boldsymbol{x}_1 \;\; \boldsymbol{z}^2 + \beta_2^j \cdot \boldsymbol{x}_2 \ldots]$, where $\boldsymbol{x}_i$ is the unique minimal T-semiflow of subsystem i. Observe that if $\boldsymbol{\sigma}_0 = [\boldsymbol{z}^1 \;\; \boldsymbol{z}^2 \ldots]$ is used in the state equation, the marking of the subsystems will be equal to the final marking but there could be negative markings at buffers. We will show that after each iteration the output buffers of at least one subsystems will be non-negative until the end of the procedure.

Let us consider the solution of LPP (6) and the resulting firing count vector: $[\boldsymbol{\sigma}^1 + \alpha_1 \cdot \boldsymbol{x}_1 \;\; \boldsymbol{\sigma}^2 + \alpha_2 \cdot \boldsymbol{x}_2 \;\; \ldots]$. Let $\mathcal{S}_0$ be the set of subsystems k such that $\alpha_k = 0$. According to Lemma 3.5, $\mathcal{S}_0 \neq \emptyset$. Let us assume that there exists an iteration h such that one subsystem in $\mathcal{S}_0$ is requested to recompute its local firing count vector. This would imply a circular dependence of the markings of the involved buffer places. Notice that the global system is conservative (due to assumption (A2)) and therefore there exists a P-semiflow containing such places. Hence, this would result in the non-reachability of the final marking. This is a contradiction of our hypothesis. Therefore, the subsystems in $\mathcal{S}_0$ compute their local control law in step 1) of the algorithm.

Let us define $\mathcal{O}_1 = K \setminus \mathcal{S}_0$. If $\mathcal{O}_1 = \emptyset$ the theorem trivially holds. Otherwise, by applying a similar reasoning, it can be proved that there exists a subsystem in $\mathcal{O}_1$ such that their output buffers are greater than or equal to zero for every $\boldsymbol{\sigma}_i, i > 1$. Denote by $\mathcal{S}_1$ the set of subsystems that do not have to recompute after the first iteration.

The same reasoning can be applied to $\mathcal{O}_2 = K \setminus (\mathcal{S}_0 \cup \mathcal{S}_1)$, $\mathcal{O}_3 = K \setminus (\mathcal{S}_0 \cup \mathcal{S}_1 \cup \mathcal{S}_2)$, ..., implying that at most $|K| - 1$ iterations are sufficient to ensure that no subsystem has to recompute. $\square$

3.3 On the distributed and centralized approaches

In this subsection we discuss some properties of the distributed implementation proposed in the previous subsections with respect to a centralized method.

Let us observe that a centralized method requires first the computation of a globally fireable firing count vector. This can be done by using an LPP having constraints which depend on the global PN structure and on the global marking, and a number of variables equal to $|P| + |T|$. The computation complexity of the distributed approach, based on the iterative algorithm, requires in the worst case $|K| - 1$ iterations, in each iteration each subsystem k might need to solve an LPP with a number of variables equal to $|P^k| + |T^k|$. Although, it is not easy to compare the computation complexity of both methods, in many cases is preferable to compute many small LPP instead of a bigger one. Moreover, in the case of a distributed approach, these small LPP could be solved in parallel by each subsystem.

One of the most important advantages of the distributed approach appears when a time interpretation is considered. In order to apply the local control laws it is not necessary to coordinate the full set of states (global marking of the system). This opens the possibility to design independently controllers for each subsystem. And thus, each controller could optimize different objective functions. Moreover, in many practical situations, it is not possible to have a global view (structure and state) of the system making a centralized controller infeasible.

4 Case study

We consider the AGV system taken from [17] as a case study. This system describes a factory floor which consists of three workstations which operate on parts, two input stations, one output station and five AGVs which move parts from one station to another. A Petri net model is presented in [18] but there, the communication between modules is done via common transitions. In order to apply our method we need communication through buffers. We can simply transform the common transitions to buffers by following the scheme in Fig 1.4 (d) of [25], i.e., a *master/slave* structure, leading to the DCPN model in Fig. 3. Tokens in workstations represent available workers and we assume

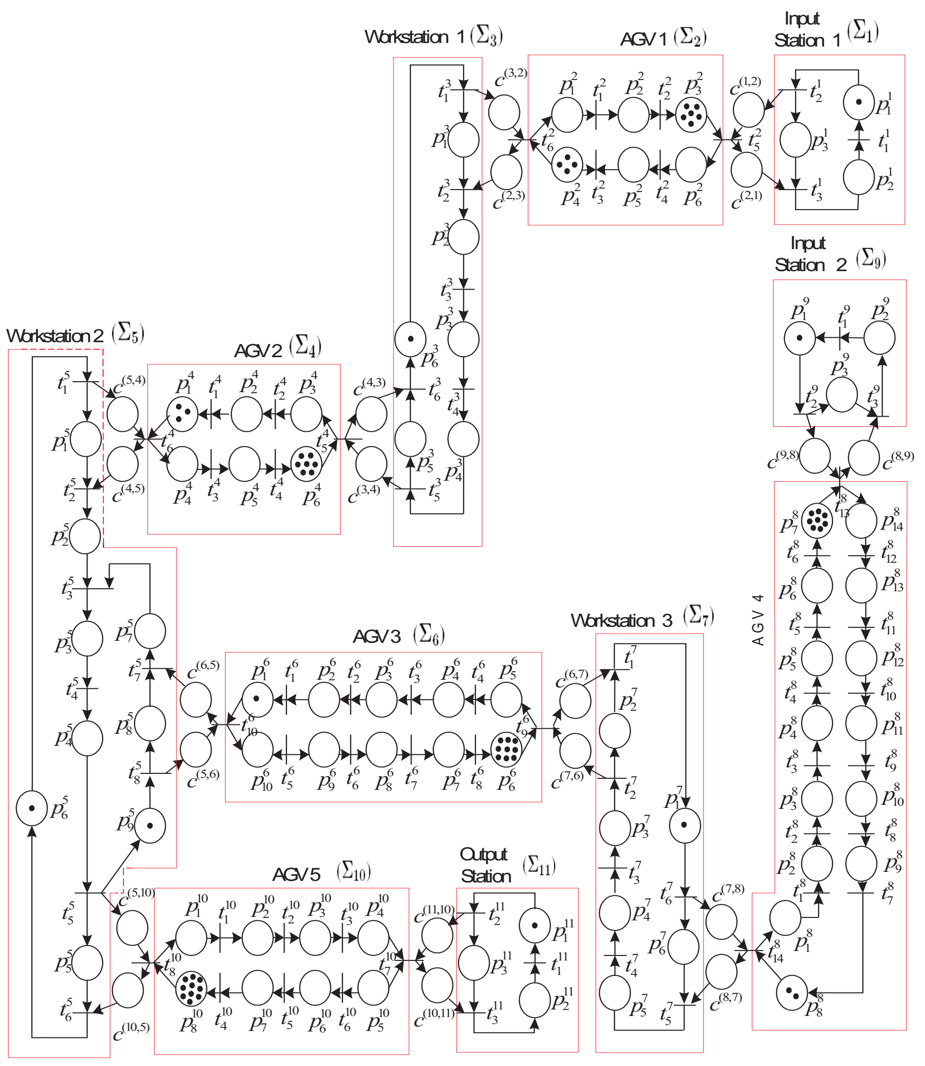


Figure 3: A marked graph DCPN model for 5 AGVs taken from [17]. Buffers are places $c^{(i,j)}$.

for workstations 1 and 3 only one worker and two workers for workstation 2 (initially, one worker is in p_6^5 and one worker in p_9^5). Additionally, the number of tokens in the part corresponding to an AGV represents its capacity. We assume capacity 10 for each AGV.

4.1 Local coordinators

We want to compute the firing vectors for each subsystem to produce at least 20 final products at the output. This implies the firing of transition t_1^{11} at least in an amount equal to 20, i.e., $\bar{\boldsymbol{z}}(T^{11}) = [20\ 0\ 0]^T$.

Let us consider the same initial and desired state for input and output stations and workstations. For the input and output stations, we assume that they are IDLE being ready to provide a raw material or accept a final product, i.e., $\boldsymbol{m}_0(p_1^1) = \boldsymbol{m}_f(p_1^1) = 1$, $\boldsymbol{m}_0(p_1^9) = \boldsymbol{m}_f(p_1^9) = 1$, $\boldsymbol{m}_0(p_1^{11}) = \boldsymbol{m}_f(p_1^{11}) = 1$. For workers we assume the following initial/final distribution: $\boldsymbol{m}_0(p_6^3) = \boldsymbol{m}_f(p_6^3) = \boldsymbol{m}_0(p_6^5) = \boldsymbol{m}_f(p_6^5) = \boldsymbol{m}_0(p_9^5) = \boldsymbol{m}_f(p_9^5) = 1$ and $\boldsymbol{m}_0(p_1^7) = \boldsymbol{m}_f(p_1^7) = 1$.

For the AGV, we consider the following initial states: AGV 1: $\boldsymbol{m}_0(p_3^2) = 6$, $\boldsymbol{m}_0(p_4^2) = 4$; AGV 2: $\boldsymbol{m}_0(p_1^4) = 3$, $\boldsymbol{m}_0(p_6^4) = 7$; AGV 3: $\boldsymbol{m}_0(p_1^6) = 1$, $\boldsymbol{m}_0(p_6^6) = 9$; AGV 4: $\boldsymbol{m}_0(p_7^8) = 8$, $\boldsymbol{m}_0(p_8^8) = 2$; AGV 5: $\boldsymbol{m}_0(p_8^{10}) = 10$. At the final state we require: AGV 1: $\boldsymbol{m}_f(p_3^2) = 10$; AGV 2: $\boldsymbol{m}_f(p_6^4) = 10$; AGV 3: $\boldsymbol{m}_f(p_6^6) = 10$; AGV 4: $\boldsymbol{m}_f(p_7^8) = 8$, $\boldsymbol{m}_f(p_1^8) = 2$; AGV 5: $\boldsymbol{m}_f(p_8^{10}) = 10$.

Each subsystem, as well as the global PN system, is MTS (in fact, it is a *marked graph*), being the unique minimal T-semiflow equal to $\mathbf{1}$. This means that for each subsystem, if all transitions are fired in the same quantity g, the local initial marking is recovered. This can be interpreted as the production and consumption of g items. Let us apply Alg. 3.3 to compute (for each subsystem) the firing count vector driving the subsystem from its initial state to the final one ensuring the global reachability in all subsystems. Initially, each subsystem solves LPP (2) and the following firing count vectors are obtained:

- $\boldsymbol{z}^1 = [0\ 0\ 0]^T$;
- $\boldsymbol{z}^2 = [4\ 4\ 0\ 0\ 0\ 4]^T$;
- $\boldsymbol{z}^3 = [0\ 0\ 0\ 0\ 0\ 0]^T$;
- $\boldsymbol{z}^4 = [0\ 0\ 3\ 3\ 0\ 3]^T$;
- $\boldsymbol{z}^5 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$;
- $\boldsymbol{z}^6 = [0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 1]^T$;
- $\boldsymbol{z}^7 = [0\ 0\ 0\ 0\ 0\ 0]^T$;

- $\boldsymbol{z}^8 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 2]^T$;
- $\boldsymbol{z}^9 = [0\ 0\ 0]^T$;
- $\boldsymbol{z}^{10} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$;
- $\boldsymbol{z}^{11} = [20\ 20\ 20]^T$.

Since the DCPN has 11 subsystems, Alg. 3.3 performs 10 iterations.

According to the firing vector ensuring the reachability of the global marking, the subsystems can be split into sets according to the quantity they have to fire their local T-semiflow (see the proof of Theorem 3.6). In this case, $\mathcal{S}_0 = \{\Sigma_{11}\}$, $\mathcal{S}_1 = \{\Sigma_{10}\}$, $\mathcal{S}_2 = \{\Sigma_5\}$, $\mathcal{S}_3 = \{\Sigma_4, \Sigma_6\}$, $\mathcal{S}_4 = \{\Sigma_3, \Sigma_7\}$, $\mathcal{S}_5 = \{\Sigma_2, \Sigma_8\}$, $\mathcal{S}_6 = \{\Sigma_9, \Sigma_1\}$. Therefore, after the first computation, Σ_{11} computes its final control law, i.e., $\boldsymbol{z}^{11}$ will not change during the iterations. Notice that, since the required number of final product is 20 and no intermediate products are initially available in Σ_{11}, the T-semiflow of the subsystem must fire 20 times.

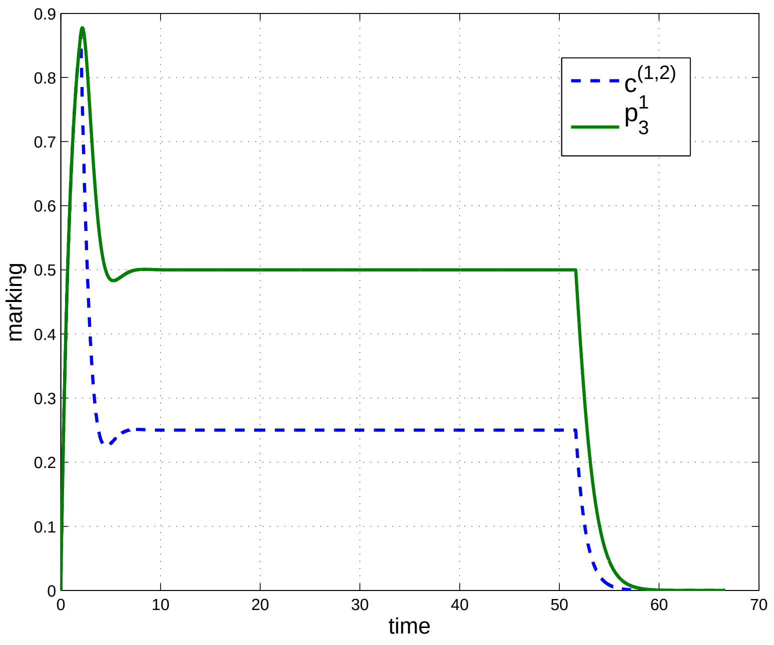
Since, $\mathcal{S}_1 = \{\Sigma_{10}\}$, after the first iteration, the subsystem Σ_{10} computes its final firing vector. In particular, the following vector is obtained: $\boldsymbol{z}^{10} + 20 \cdot \mathbf{1}$ by solving LPP (3). Again, there are no intermediate products at the initial state of Σ_{10}.

In the second iteration, the information that 20 final products must be manufactured arrives to Σ_5 that computes its final firing vector in order to satisfied the buffer requirements. Since, Σ_5 is also connected with Σ_4 and Σ_6, in the third iteration, these subsystems receive the final buffer requirements. According to the initial and final states of the AGV of subsystem Σ_4, the T-semiflow of this subsystem must fire 17 times. Therefore, the final vector is $\boldsymbol{z}^4 + 17 \cdot \mathbf{1}$. For Σ_6, the values of the initial and final states yield a vector $\boldsymbol{z}^6 + 19 \cdot \mathbf{1}$. This processes repeats and the particular final vector of each subsystem can be obtained from the number of times T-semiflows are fired (these numbers are given in Table 1).

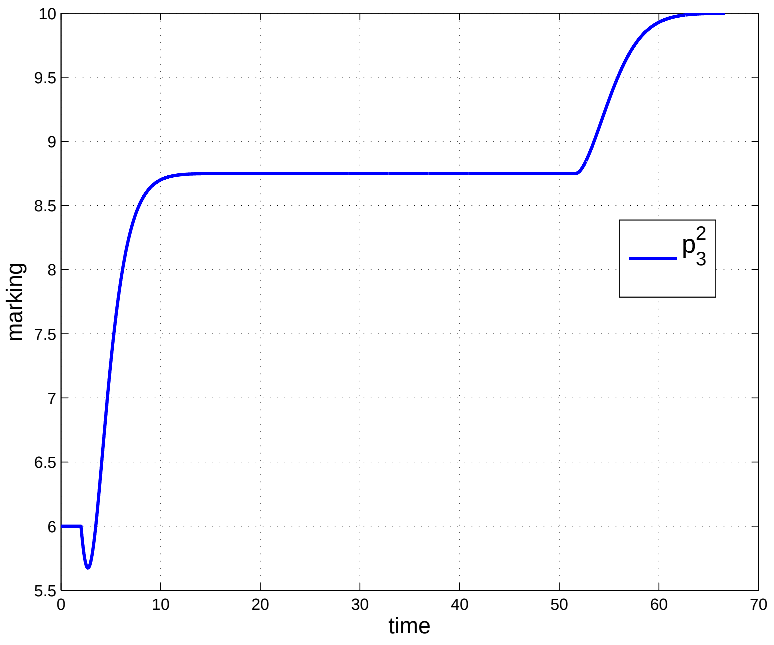
For this final marking, the set of subsystems has been dived into seven subsets, implying a number of six iterations of the algorithm in order to reach an agreement. However, this partition is depending on the initial and final markings and in general, the number of necessary iteration is not not six but upper bounded by 10 as it is proved in Theorem 3.6.

4.2 Local controllers

Let us consider now a time interpretation, in particular *infinite server semantics* [26]. In this case, we assign to each transition t_j a firing rate λ_j and the



(a) Evolution of markings $c^{(1,2)}$ and p_3^1



(b) Evolution of marking p_3^2

Iteration	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Σ_7	Σ_8	Σ_9	Σ_{10}	Σ_{11}
1	0	0	4	0	3	0	2	0	0	20	20
2	0	0	4	4	20	2	2	0	0	20	20
3	0	0	4	17	20	19	2	0	0	20	20
4	0	0	17	17	20	19	19	0	0	20	20
5	0	13	17	17	20	19	19	17	0	20	20
$6-10$	13	13	17	17	20	19	19	17	17	20	20

Table 1: Execution of Alg. 3.3 on subsystems of DCPN in Fig. 3. Observe that from the sixth iteration there is no update.

instantaneous firing flow of t_j at a marking $\boldsymbol{m}$ is given by

$$f_j = \lambda_j \cdot enab(t_j, \boldsymbol{m}), \tag{7}$$

where $enab(t_j, \boldsymbol{m})$ is the enabling degree of transition t_j given in eq. (1).

If the net system is subject to external control actions, the only admissible control law consists in slowing down the (maximal) firing flow of transitions (defined for the uncontrolled systems as in eq. (7)). This implies that the controlled flow of the system is: $\boldsymbol{w}(\tau) = \boldsymbol{f}(\tau) - \boldsymbol{u}(\tau)$, with $\boldsymbol{0} \leq \boldsymbol{u}(\tau) \leq \boldsymbol{f}(\tau)$. The overall behavior of the system is ruled by: $\dot{\boldsymbol{m}} = \boldsymbol{C} \cdot (\boldsymbol{f}(\tau) - \boldsymbol{u}(\tau))$.

Once a server semantics is chosen, we can fix a control goal on the dynamic subsystems. For example, a minimum time control goal for this system leads to an ONOFF strategy for which every transition is fired as fast as possible at any moment until the required minimal firing count is reached (for more details see [30]).

Figures 4(a) and 4(a) show the time trajectories of places $c^{(1,2)}$, p_1^3 and p_3^2 and when an ON-OFF control strategy is applied to the subsystems using the computed firing count vectors and all firing rates of transitions are 1. We assume that the subsystem Σ_2 is applying its corresponding control law 2 time units later than subsystem Σ_1. Notice that during the time interval 10 to 52 approximatively, the markings remain constant so that subsystems fire their T-semiflow 13 times. Finally, around 65 time units, both subsystems reach their final markings.

5 Conclusions

This paper studies the control of distributed continuous Petri net systems. Following a Divide & Conquer approach, a two level solution is proposed. The first level corresponds to the untimed distributed system while the second level corresponds to the timed system. This paper has focused on the first level.

Each disjoint subsystem is modeled as a subnet, and the communication among subsystems is achieved by means of *buffers*. The approach developed here is based on the design of a set of local coordinators (one per subsystem) that negotiates with its neighbor coordinators to meet an agreement in order to reach the target markings. It is proved that, under certain assumptions on the system, the proposed algorithm always yields an appropriate firing count vector. This firing amount is equal to the global firing vector that would have been obtained by solving the reachability problem on the overall system. The number of iterations of the algorithm is equal to the number of subsystems minus one being the complexity of the operations in the loop polynomial. The procedure assumes no time interpretation and therefore it can be potentially applied to timed systems with any firing semantics and control goals, e.g., minimum time, minimum quadratic cost.

References

[1] M. Allam and H. Alla. Modeling and simulation of an electronic component manufacturing systems using hybrid Petri nets. *IEEE Trans. on Semiconductor Manufacturing*, 11(3):374–383, 1998.

[2] A. Amrah, N. Zerhouni, and A. El Moudni. On the control of manufacturing lines modelled by controlled continuous Petri nets. *International Journal of Systems Science*, 29(2):127–137, 1998.

[3] P. Antsaklis, X. Koutsoukos, and J. Zaytoon. On hybrid control of complex systems: a survey. *European Journal of Automation, APII-JESA*, 32(9–10):1023–1045, 1998.

[4] H. Apaydin-Özkan, J. Júlvez, C. Mahulea, and M. Silva. A Control Method for Timed Distributed Continuous Petri nets. In *2010 American Control Conference*, pages 2593 – 2600, Baltimore, USA, 2010.

[5] H. Apaydin-Özkan, J. Júlvez, C. Mahulea, and M. Silva. Approaching Minimum Time Control of Timed Continuous Petri nets. *Nonlinear Analysis: Hybrid Systems*, 5(2):136–148, 2011.

[6] H. Apaydin-Özkan, C. Mahulea, J. Júlvez, and M. Silva. An iterative control method for distributed continuous Petri nets. In *49th IEEE Conference on Decision and Control*, pages 6753–6758, 2010.

[7] F. Balduzzi, G. Menga, and A. Giua. First-order hybrid Petri nets: a model for optimization and control. *IEEE Trans. on Robotics and Automation*, 16(4):382–399, 2000.

[8] B. Bordbar, L. Giacomani, and D.J. Holding. Design of Distributed Manufacturing Systems Using UML and Petri nets. In *6th IFAC Workshop on Alg. and Arch. for Real-Time Control*, pages 91–96, 2000.

[9] S. Christensen and L. Petrucci. Modular Analysis of Petri Nets. *The Computer Journal*, 43(3):224–242, 2000.

[10] R. David and H.Alla. *Autonomous and timed continuous Petri nets.* Springer, Berlin, 2010. 2^{nd} edition.

[11] M. Dotoli, M.P. Fanti, A. Giua, and C. Seatzu. Modelling systems by hybrid Petri nets. An application to supply chains. In V. Kordic, editor, *Petri Net, Theory and Application.* I-Tech Education and Publishing, Austria, 2008.

[12] E. Fabre and L. Jezequel. Distributed Optimal Planning: An Approach by Weighted Automata Calculus. In 48^{th} *Conference on Decision and Control (CDC)*, pages 211 – 216, Shanghai, P.R. China, December 2009.

[13] J. Júlvez and R.K. Boel. A Continuous Petri Net Approach for Model Predictive Control of Traffic Systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(4):686 –697, 2010.

[14] J. Júlvez, L. Recalde, and M. Silva. Steady–state performance evaluation of continuous mono-T-semiflow Petri nets. *Automatica*, 41(4):605–616, 2005.

[15] X. Koutsoukos, K. He, M. Lemmon, and P. Antsaklis. Timed Petri nets in Hybrid Systems: Stability and Supervisory Control. *Journal of Discrete Event Dynamic Systems: Theory and Applications*, 8:137–174, 1998.

[16] P. Krishnan. Distributed timed automata. In *WDS'99, Workshop on Distributed Systems (A satellite workshop to FCT'99)*, volume 28 of *Electronic Notes in Theoretical Computer Science*, pages 5 – 21, 2000.

[17] B.H. Krogh and L.E. Holloway. Synthesis of feedback control logic for discrete manufacturing systems. *Automatica*, 27:641–651, July 1991.

[18] C. Lakos and L. Petrucci. Modular Analysis of Systems Composed of Semiautonomous Subsystems. In *4th International Conference on Application of Concurrency to System Design*, Hamilton Canada, 2004.

[19] D. Lefebvre. Feedback control design for manufacturing systems modeled by continuous Petri nets. *International Journal of Systems Science*, 30(6):591–600, 1999.

[20] C. Mahulea, A. Giua, L. Recalde, C. Seatzu, and M. Silva. Optimal model predictive control of Timed Continuous Petri nets. *IEEE Transactions on Automatic Control*, 53(7):1731 – 1735, August 2008.

[21] O. Morandin Jr, ERR Kato, PR Politano, HA Camargo, AJV Porto, and RY Inamasu. A modular modeling approach for automated manufacturing systems based on shared resources and process planning using Petri nets. In *2000 IEEE Int. Conf. on Systems, Man, and Cybernetics*, volume 4, pages 3057 –3062, 2000.

[22] R. P. Moreno, D. Tardioli, and J.l.V. Salcedo. Distributed Implementation of Discrete Event Control Systems based on Petri nets. In *ISIE08: IEEE Int. Symposium on Industrial Electronics*, pages 1738–1745, June 2008.

[23] L. Recalde, E. Teruel, and M. Silva. Modeling and analysis of sequential processes that cooperate through buffers. *IEEE Trans on Robotics and Automation*, 14:267–277, 1998.

[24] L. Recalde, E. Teruel, and M. Silva. Autonomous continuous P/T systems. In *Proc. ICATPN*, pages 107–126. Springer, LNCS 1689, 1999.

[25] M. Silva. Introducing Petri nets. In F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva, and F. B. Vernadat, editors, *Practice of Petri Nets in Manufacturing*, pages 1–62. Chap. & Hall, 1993.

[26] M. Silva, J. Júlvez, C. Mahulea, and C.R. Vázquez. On fluidization of discrete event models: observation and control of continuous Petri nets. *Discrete Event Dynamic Systems: Theory and Applications*, 21(4):427–497, 2011.

[27] M. Silva, J. Júlvez, C. Mahulea, and C.R. Vázquez. On fluidization of discrete event models: observation and control of continuous Petri nets. *Discrete Event Dynamic Systems: Theory and Applications*, 2011. in press.

[28] M. Silva and L. Recalde. On fluidification of Petri net models: from discrete to hybrid and continuous models. *Annual Reviews in Control*, 28(2):253–266, 2004.

[29] C.R. Vazquez, J.H. van Schuppen, and M. Silva. A Modular-Coordinated Control for Continuous Petri Nets. In *18th World Congress of IFAC*, Milan, Italy, 2010.

[30] L. Wang, C. Mahulea, J. Julvez, and M. Silva. Minimum-time Decentralized Control of Choice-Free Continuous Petri Nets. *Nonlinear Analysis: Hybrid Systems*, 7(1):39–53, 2013.

[31] L. Wang and X. Xie. Modular modeling using Petri nets. *Robotics and Automation, IEEE Transactions on*, 12(5):800 –809, October 1996.