Neural Comput & Applic
DOI 10.1007/s00521-013-1391-1

ORIGINAL ARTICLE

Petri net-based modeling and control of the multi-elevator systems

Farooq Ahmad · Ilyas Fakhir · Sher Afzal Khan · Yaser Daanial Khan

Received: 3 September 2012 / Accepted: 20 March 2013

Abstract This paper extends the Petri net (PN)-based modeling of multi-elevator control system for M floors and N elevators which provides the generic PN model of the system. A new class of Petri nets is introduced known as elevator control Petri net (ECPN) for such purpose. The model of the multi-elevator control system is developed through components, whereas the model of each elevator is defined as a component. The interaction between these elevators is implemented through *control places* (CPs) of its PN model. A bottom-up modeling approach is adopted by adding the CPs and using the arc-addition operator to the single-elevator modules. Mixture of collective and selective approaches, that is, collective-selective/up–down approach, is used for the control. The proposed Petri net class in the paper resolves the bunching problem among multiple elevators. The bunching problem is tackled by introducing the request places with the capacity of one in the ECPN. A case study of ECPN is also presented by taking the two elevators and four-floor model, and it is analyzed by the incidence matrix–based invariant method.

F. Ahmad (✉) · Y. D. Khan
Faculty of Information Technology, University of Central Punjab, Lahore, Pakistan
e-mail: dr.farooq@ucp.edu.pk

Y. D. Khan
e-mail: ydk@ucp.edu.pk

I. Fakhir
Department of Computer Science, GC University, Lahore, Pakistan
e-mail: fakhir@gcu.edu.pk

S. A. Khan · Y. D. Khan
Department of Computer Science, Abdul Wali Khan University, Mardan, Pakistan
e-mail: sher.afzal@awkum.edu.pk

Keywords Petri net · Multi-elevator control system · Control place · Component-based modeling

1 Introduction

Elevators are an imperative part of buildings from the time of their invention. Motility among floors especially in conjunction with heavy goods is an onerous task if accomplished by means of stairs. An elevator system is designed to control multiple elevators in a multi-floor building to transport passengers or goods rapidly and efficiently. Nowadays, in high-rise buildings, modern elevator systems consist of multiple elevators tempered usually by a centralized control mechanism. Single-elevator systems are less problematic, but a number of the convoluted issues arise in systems controlling more than one elevator in the building such as online scheduling, resource allocation and stochastic control. These issues are often handled by job shop systems or automated manufacturing systems which behave as discrete event systems (DESs) [1, 2] with time properties. In stochastic systems, each passenger introduces three types of random variables: arrival time, arrival floor and destination floor. Usually, two types of controls are used to request elevators: one is by hall-call and other is by car-call. An elevator control system is used to allocate an appropriate elevator to a passenger requested by a hall-call or a car-call.

By using destination control system (DCS) in a modern elevator system, each passenger selects his destination before entering in an elevator [3–5]. DCS can group passengers going to the same floor in the same car, reducing the waiting time and the number of stops as the destination calls have been reduced [6]. Different stochastic and

Published online: 04 April 2013

probabilistic paradigms have been adopted for an elevator group control system which includes fuzzy logic, neural networks, genetic algorithms, Markov chains, among others [7–10]. Another programmable logic controller (PLC) approach is adopted by Yang et al. [11] without a group controller and used two elevators for nine floors.

All these techniques are of computational nature as they help to deduce the most optimal outcomes using some logical, probabilistic or intelligent model. These techniques do not address issues related to the design, modeling, analysis and overall performance of the system. On the other hand, Petri net [12, 13] provides design, modeling, specification and then analysis and verification of the system. Moreover, Petri nets have the ability of formal verification to identify the significant behavioral properties of the models for discrete event control. A Petri net provides the logical sequencing of a system, and this inherent property indwells within a Petri net by the virtue of its graphic nature. Petri net modeling incorporates concurrency, non-determinism, schedule and resource-sharing; besides this, it captures structural interactions to model deadlocks, conflicts, buffer sizes and precedence relations [14, 15]. In addition, Petri nets can perform both the qualitative and quantitative analysis of the modeled system due to their underlying mathematical foundation [16].

The Petri net approach has been used for modeling elevator systems [17] and simulation of control strategies in intelligent building [18]. A hybrid model of a multiple-elevator system is proposed, consisting of a color-timed transition Petri net (CTTPN) model and a set of control rules implemented via the so-called control places in the CTTPN model [19]. In [20], a hybrid model of a multiple-elevator system is proposed, consisting of a timed place Petri net (TPPN) model and a set of control places that are used to enforce control rules. The Petri net model is a highly modulated structure, whose constituent modules are classified into four types: basic movement module, loading/unloading module, direction reversing module and call management module. Timed Petri nets (TPNs) approach is also used for modeling group elevator control system [21]. This approach is used to describe the dynamic behaviors of elevator systems. To represent the behavior of DESs, Petri nets are suitable models because of their flexibility and visualization.

In this paper, a Petri net-based approach is used for the generic model of multi-elevator control system (MECS) for m floors and n elevators. This multi-elevator system works as a multi-agent system (MAS), and each elevator behaves like an agent. The main issue discussed here is the interaction between elevators. The interaction between elevators (agents) is achieved by introducing the control places in the model; thus, the consistency of functions is guaranteed. Actually, this model has two levels: a higher level called host net and a lower level called agents net. Each agent becomes a token of host net. According to Fig. 1 in Sect. 3, MECS is a host net and all elevators are agents net. We use P/T net [12] which is a low-level PN as it has many advantages over high-level PN. The inherent nature of the underlying system makes its modeling easier using a P/T net. Moreover, a low-level PN is used to illustrate the comprehensive features of the system. Also, use of the described Petri net models makes the verification and analysis of the model easier.

One of the consequences of using the principle of collective control in relay-based automatic controllers is that the car stops at the nearest call in their running directions [22]. The major drawback that arises as a result of this scheme is that the system is defenseless against a phenomenon called bunching. Bunching is a traffic pattern formed when a number of elevators move around a building together, instead of being separated about the building. It is often caused due to sudden heavy traffic demand or an inadequate traffic supervisory system [23]. Moreover, the bunching problem leads toward the situation where several cars arrive at a floor at about the same time, making the interval, and thus the average waiting time, much longer, that is, several elevators move close to each other and compete for the same hall-call.

The proposed Petri net class, that is, ECPN, in the paper resolves the bunching problem among multiple elevators. The bunching problem is tackled by introducing the request places with the capacity of one and a set of control places in the ECPN.

2 Concepts of Petri net

This section introduces the basic concept and notations of the Place/Transition net which is a class of Petri nets. These concepts are mostly taken from [12] and [13].

Definition 1 A marked Petri net (Place/Transition net) is a 5-tuple PN $= (P, T, F, W, M_0)$ where $P = \{p_1, p_2, \ldots, p_n\}$

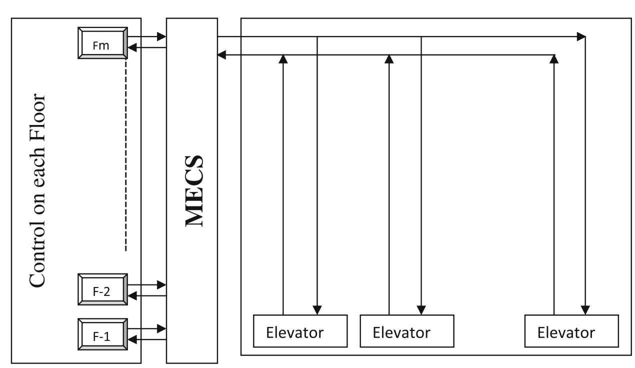

Fig. 1 A general design for multi-elevator control system (MECS)

finite set of places for $n > 0$, $T = \{t_1, t_2, \ldots, t_m\}$ finite set of transitions for $m > 0$, $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation), W: $F \to \{1, 2, 3, \ldots\}$ is the weight function, M_0: $P \to \{0, 1, 2, 3, \ldots\}$ is the initial marking, $P \cap F = \Phi$ and $P \cup F \neq \Phi$.

A PN structure with arc weight 1, known as ordinary P/T-net (P, T, F), is denoted by N, so that a PN with specific initial marking is denoted by (N, M_0) also known as marked net.

The preset and post-set of a transition t are, respectively,

$$^{\bullet}t = \{p \in P | (p, t) \in F\},$$
$$t^{\bullet} = \{p \in P | (t, p) \in F\},$$

where (p, t) $\in F$ represents the arc from place p to transition t and (t, p) $\in F$ represents the arc from transition t to place p

The preset and post-set of a place p are, respectively,

$$^{\bullet}p = \{t \in T | (t, p) \in F\},$$
$$p^{\bullet} = \{t \in T | (p, t) \in F\},$$

Definition 2 (*Incidence matrix*) a net is pure if it has no self-loops, if a net is pure, the incidence functions can be represented by a single matrix. C^- is the input matrix with $|P|$ rows and $|T|$ columns, where the entry of row i column j is $C^-[i, j] = 1$ if there is an arc (p_i, t_j) and 0 otherwise; C^+ is the output matrix with $|P|$ rows and $|T|$ columns, where the entry of row i column j is $C^+[i, j] = 1$ if there is an arc (t_j, p_i) and 0 otherwise. $C = C^+ - C^-$ is called an incidence matrix.

A marking is a vector M: $P \to N$ that assigns to each place of a P/T net a non-negative integer number of tokens, represented by black dots. $M(p)$ denotes the number of tokens assigned by marking M to place p. A transition $t \in T$ is enabled at a marking M if for each $p \in {}^{\bullet}t$, $M(p) \geq 1$. If t is enabled at M, then t may fire yielding a new marking M' such that for each $p \in P$, $M'(p) = M(p) - 1$, if $p \in {}^{\bullet}t$, $M'(p) = M(p) + 1$, if $p \in t^{\bullet}$ and $M'(p) = M(p)$ *otherwise*. We write $M[t\rangle M\prime$ to denote that t may fire at M yielding M'.

Definition 3 [12] (*P-invariant, T-invariant*) for a PN (N, M_0), a P-invariant is a $|P|$-vector $y \geq 0$ such that $y^T \cdot C = 0$, where C is the $|P| \times |T|$ incidence matrix. Similarly, a T-invariant is a $|T|$-vector $x \geq 0$ such that $C \cdot x = 0$.

Definition 4 [14] (*Boundedness and safeness*) for all $p_i \in P$ of PN (N, M_0), PN is b-bounded if $\forall M_k \in R(M_0)$: $M_k(p_i) \leq b$ and said to be safe if $M_k(p_i) \leq 1$.

Definition 5 [12] (*Liveness*) a transition $t_j \in T$ is said to be live if and only if there is a marking M' reachable from each $M \in RM(M_0)$ such that M' enables t_j and Petri net (N, M_0) is live if $\forall t_j \in T$: t_j is live.

Definition 6 [14] (*Deadlock and deadlock-free*) a transition $t_j \in T$ is said to be a dead transition at marking $M \in RM(M_0)$ if there is no reachable marking to make transition t_j enabled. A marking $M \in RM(M_0)$ is said to be a deadlock if $\forall t_j \in T$, t_j is dead. A PN (N, M_0) is said to be deadlock-free if and only if there is no deadlock.

Definition 7 [14] (*Reversible*) Petri net (N, M_0) is reversible if the initial marking M_0 is reachable from every reachable marking $M \in R(N, M_0)$.

3 Multi-elevator control system specifications

This section addresses the specifications for the MECS. It is indispensable to develop such a system that can help passengers for traveling between different floors of a building in an efficient way. A model of n elevators for m floors of a building is proposed as shown in Fig. 1. The main issue is the interaction between n elevators.

Typically, elevator system can be controlled by two types of calls. A hall-call is used for requesting an elevator, while a car-call is used by passengers after entering in elevator to decide the destination. Further, an elevator system has two types of button on each floor: one is used for entertaining hall-calls in upward direction, and second is for entertaining hall-calls in a downward direction. By pressing a hall-call button, the elevator control system registers the hall-call and then assigns an elevator to serve that hall-call. After entering in the elevator, passengers press the car-call button to select the destination floor and the elevator must move upward/downward to reach the destination floor. Moreover, inside the elevator, there are two options: one button for opening and closing door and second for selecting a destination floor. After entertaining all calls, the elevators come in idle state and we can park elevators to a specific floor mostly on a ground floor. Mostly, single-elevator systems use collective up and collective down principles, and according to the principle, an elevator must stop on the requested floor in sequence given by hall-calls in both directions. In this type of principle, order of hall-calls is ignored and sequence of floor is observed. A selective principle is also used in which highest or lowest floors are selected upward or downward, respectively. The combination of both principles is also used. A collective up and selective down principle is used when up-traffic is in the majority, and a selective up and collective down principle is used when down-traffic is in the majority. Similarly, a more organized principle collective-selective in both directions is used for getting more suitable results. To move upward direction, an elevator first chooses lowest up-hall-call by selective method and then selects all other up-hall-calls by collective method. For downward direction, first of all highest down-hall-call is chosen by selective method and then all other down-hall-call is handled by collective methods.

In a proposed MECS model, we suggest an approach, by which both the hall-call's order and the floor sequence are taken under consideration. In this way, our proposed model is also based on collective-selective/up–down algorithm [24]. For example, hall-calls' order in the upward direction is $\{h_1, h_5, h_2, h_7\}$; the selected elevator by MECS first moves to 1st floor and if before moving upward, the 2nd floor's call is generated then it moves to this floor and then forward to 5th and 7th floors. In case that the elevator has crossed 2nd floor and then h_2 hall-call is generated, the elevator moves forward and completes its tasks of 5th and 7th floors. The 2nd floor's call is completed by another free elevator and if no one will be free then waits for being free any one elevator.

3.1 The flow of MECS

The flowchart in Fig. 2 describes the flow of the proposed model and its work begins when a hall-call is generated. First of all, the availability of elevators is checked, if more than one elevator is free, then an elevator having least waiting time will be selected and send to the requested floor by selective approach. For single free elevator, it is sent to the requested floor by selective approach. When no elevator is free, then the movement direction and capacity of the elevator is calculated, and by matching these parameters, a more appropriate elevator is forwarded to the requested floor by collective approach. Two types of variables, that is, state and action command, are used. The current state of elevators is designated as state variables and the decisions making is done by the method when a hall-call or car-call is generated to indicate as action command variables.

A MAS is formed by the cooperation between many agents to perform any specific task. According to our proposed model, multi-elevator system does the same jobs as MAS. The main issue we solved here is the cooperative behavior among multi-elevators for concurrency. The Petri net-based approach is used for the interaction between multiple agents [25] to analyze their structural and behavioral characteristics for the correctness of the framework. Petri nets guarantee the correctness of the cooperation between agents as the perception of reachability in a given Petri net. A group of agents is said to be homogeneous if the capabilities of the individual agents are identical and heterogeneous otherwise. Heterogeneity introduces complexity because task allocation becomes more involved and agents need to model other individuals in the group.

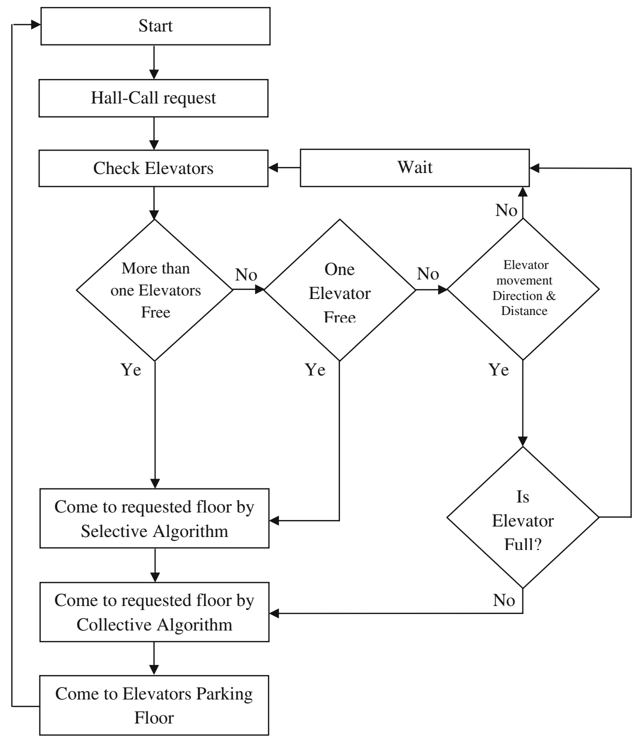

Fig. 2 Flowchart of multi-elevator control system

4 Petri net-based system modeling

4.1 Single-elevator model

First of all, we discuss the working of the single elevator between different floors. This single elevator works independently according to the need of its users. Petri net provides us a better approach to verify the working of an elevator. The elevator is called to requested floor with a hall-call, and the destination is selected by a car-call request. The places represent floors, and transitions represent control of the elevator. The collective-selective/up–down algorithm is used to facilitate passengers moving upward and downward as well.

Figure 3 shows a general Petri net model for a single elevator. Place p_i represents the ith floor where $i = 1, 2, \ldots, m$. It is important to be noted that the elevator resides on the first floor (at ground) when it is idle. Therefore, p_{i-1}, p_{i+1}, represent the $(i-1)$th and $(i+1)$th floors, respectively, and floors in a building can be in the hundreds. The place r_i represents the request of an elevator from ith floor, while the button transition b_i is used to generate the request. The transitions of the model illustrate as t_i and t_j, where t_i (represented by white bar) is used to move upward direction and t_j (represented by black bar) is used to move downward direction. Suppose there are k transitions for moving between m floors. For m floors, there are $2(m-1)$ transitions for each place representing a floor and the total number of transitions used to move the elevator in upward and downward directions are $2m(m-1)$. Therefore, for p_i representing the ith floor, there would be $(m-i)$ transitions for moving upward direction and $(i-1)$ transitions for coming to ith floor from the lower floors. Similarly, there would be $(m-i)$ transitions for moving downward direction from higher floors and $(i-1)$ transitions for moving from ith floor to lower floors. For example, if a building has 7 floors, then there are 12 transitions on each floor and the total number of transitions would be 84. Further, on 4th floor, 3 transitions are used for moving upward directions and 3 transitions are used for coming on the 4th floor from lower floors which are represented by white bars. Moreover, 3 transitions are used for moving downward directions from higher floors to 4th floor and 3 transitions for moving from 4th floor to lower which are represented by black bars in the Petri net model.

4.2 Petri net model for the multi-elevator control system (MECS)

The MECS consists of n elevators for m floors in an intelligent building for transferring passengers between different floors. The MECS model works as MAS because agents can perform complex tasks by interacting with each other.

The idea of *Controlled PN* (CtrlPN) is given by Krogh [26] and similarly by Ichikawa and Hiraishi [27]. For the synthesis of the MECS in this research, an elevator control Petri net (ECPN) is introduced by considering the PN model for a single elevator as a module.

To construct the final model, the composition of the modules of the elevators is performed by introducing the

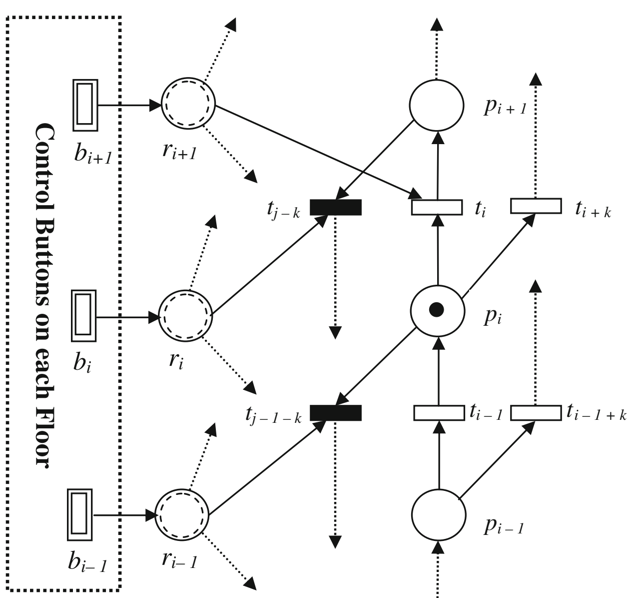

Fig. 3 A general Petri net model for single elevator

control places, request places (with the capacity of 1) and transitions for generating the request for different floors. In this way, bottom-up modeling approach [28] is adopted for the synthesis of ECPN. The composition of the modules of a single elevator is performed by using the arc-addition operator [29]. The arcs are added from control places to the transitions of the modules used to move the elevator downward direction as well as arcs from transitions used to move upward direction to control places and vice versa. Thereafter, request places r_i with capacity one and request-generating transitions b_i are added to the final model for m floors and n elevators. These request places $r_i \in P_R$ in ECPN further solve the bunching problem [23].

Figure 4 represents a model of n elevators for m floors, and all elevators interact with each other by control places. Because our model consists of n elevators, and consequently control places are marked with n tokens. By measuring performance criteria and checking traffic conditions, some necessary tokens can be added to CPs for achieving better results. The number of elevators represents by the tokens in CPs, and the interaction between these token will be through any agent communication protocol as proposed in [30]. The movement of tokens between CPs is based on the hall/car-calls for elevators. When a hall-call is generated, a token is added to r_i places by source transition b_i. The car-call control buttons are attached with all other transitions for selecting destination floor either upward or downward direction.

Elevator control Petri net has a set of places representing the floors P_F, a set of control places P_C and a set of request places P_R. Further, the transitions of ECPN consist of a set of transitions for upward movement T_U, a set of transitions T_D for downward movement and a set of transitions T_B (button transitions) for generating the request of an elevator at different floors.

Definition 8 (*Elevator control Petri net*) elevator control Petri net (ECPN) is defined as ECPN $= (P, T, K, F, M_0)$, where

i. $P = P_F \cup P_C \cup P_R$ such that $P_F \cap P_C \cap P_R = \varnothing$
ii. $T = T_U \cup T_D \cup T_B$ such that $T_U \cap T_D \cap T_B = \varnothing$
iii. $F \subseteq (P \times T) \cup (T \times P)$
iv. $K: P_R \to \{1\}$
v. ${}^{\bullet}P_R = T_B$ and $P_R^{\bullet} \subseteq T_U \cup T_D$
vi. $\forall t \in T_U,\ \exists p \in P_C$ such that either $(t, P) \in F$ or $(p, t) \in F$
vii. $\forall t \in T_D,\ \exists p \in P_C$ such that either $(t, P) \in F$ or $(p, t) \in F$
viii. ${}^{\bullet}T_B = \varnothing$ and $T_B^{\bullet} = P_R$

The cardinality of transitions and places is checked for m floors and n elevators:

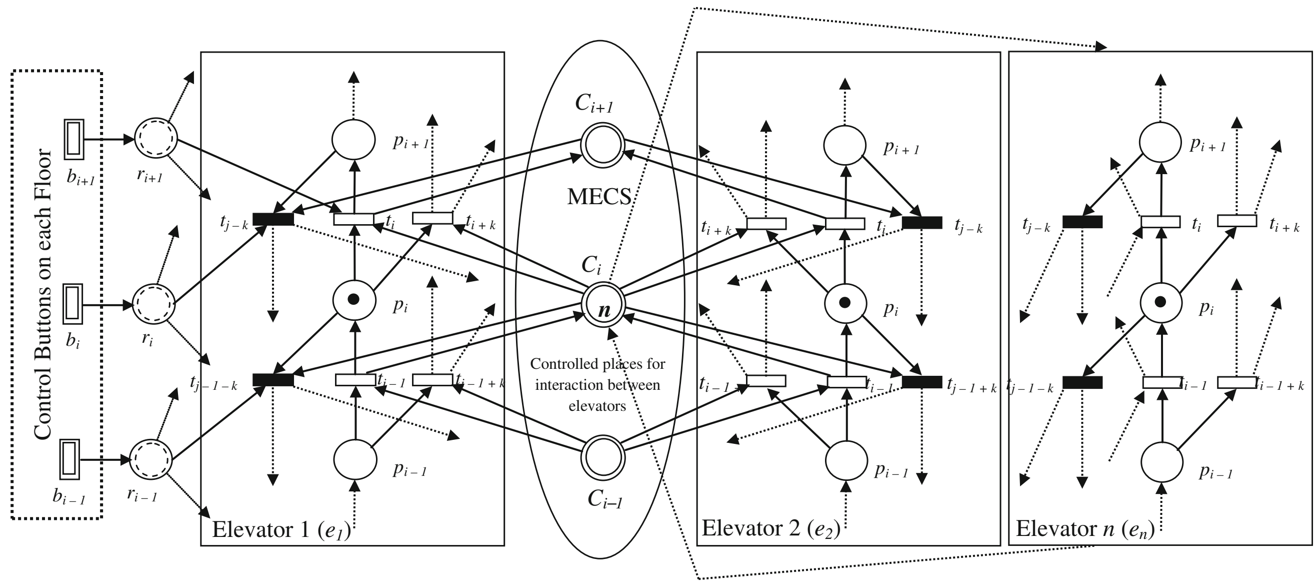

Fig. 4 Petri net model for N elevators and M floors

ix.

$$|T_B| = m, |T_U| + |T_D| = 2mn(m-1) \Rightarrow |T| = |T_B| + (|T_U| + |T_D|) = m[2n(m-1)+1]$$

x.

$$|P_R| = |P_C| = m \text{ and } |P_F| = nm \Rightarrow |P| = |P_R| + |P_C| + |P_F| = m(n+2)$$

Property 1 *Each place $p_i \in P_F$ for ith floor in ECPN has $p_i^\bullet \subseteq T_U \cup T_D$ and ${}^\bullet p_i \subseteq T_U \cup T_D$ where $|p_i^\bullet| = (m-1)$ and $|{}^\bullet p_i| = (m-1)$.*

Proof The set of transitions T_U has the transitions used to move the elevator upward direction, and the set of transitions T_D has the transitions used to move the elevator in a downward direction. Further $\forall t \in p_i^\bullet$, t is used to move upward from ith floor or to move downward from ith floor, that is, either $t \in T_U$ or $t \in T_D$ therefore $p_i^\bullet \subseteq T_U \cup T_D$. Now from condition (ix) of ECPN, each place $p_i \in P_F$ for ith floor has $2(m-1)$ transitions to move away from ith floor and to move toward ith floor. Out of $2(m-1)$ transitions, exactly half of the transitions are used to come at ith floor and half of the transitions are used to move from ith floor. Therefore, $|p_i^\bullet|$ represents the number of transitions used to move away from ith floor thus we have $|p_i^\bullet| = (m-1)$.

Similarly, $\forall t \in {}^\bullet p_i$, t is used to move upward from lower floors toward ith floor or to move downward from higher floors toward ith floor, that is, either $t \in T_U$ or $t \in T_D$ therefore ${}^\bullet p_i \subseteq T_U \cup T_D$. Further $|{}^\bullet p_i|$ represents the number of transitions used to move toward ith floor thus we have $|{}^\bullet p_i| = (m-1)$.

Property 2 *(t_{b_m}, p_i), …, $(t_{b_{(i+1)}}, p_i) \in F$ are $(m-i)$ incoming arcs from the transitions of a set T_D to each place $p_i \in P_F$ for ith floor where $t_{b_m}, \ldots, t_{b_{(i+1)}} \in T_D$ and (t_{w_1}, p_i), …, $(t_{w_{(i-1)}}, p_i) \in F$ are $(i-1)$ incoming arcs to $p_i \in P_F$ from the transitions of a set T_U where $t_{w_1}, \ldots, t_{w_{(i-1)}} \in T_U$. In addition, $p_i \in P_F$ has $(i-1)$ outgoing arcs to the transitions of a set T_D, i.e., (p_i, t_{b_1}), …, $(p_i, t_{b_{(i-1)}}) \in F$ where $t_{b_1}, t_{b_2}, \ldots, t_{b_{(i-1)}} \in T_D$ and $(m-i)$ outgoing arcs to the transitions of a set T_U as $(p_i, t_{w_{(i+1)}})$, …, $(p_i, t_{w_M}) \in F$ where $t_{w_{(i+1)}}, \ldots, t_{w_M} \in T_U$.*

Proof From Property 1, each place $p_i \in P_F$ has preset of transitions with the cardinality equal to $(m-1)$. There are $(m-i)$ floors above and $(i-1)$ floors below the ith floor, therefore $(m-i)$ incoming arcs from the transitions of the set T_D to p_i for the downward movement of the elevator to ith floor. Similarly, there are $(i-1)$ incoming arcs from the transitions of the set T_U to p_i for the upward movement of the elevator to ith floor.

Further, $p_i \in P_F$ has a post-set of transitions with the cardinality equal to $(m-1)$. There are $(m-i)$ floors above and $(i-1)$ floors below the ith floor, therefore $(m-i)$ outgoing arcs to the transitions of the set T_U from p_i for upward movement of the elevator from ith floor. Similarly, there are $(i-1)$ outgoing arcs to the transitions of the set T_D from p_i for downward movement of the elevator from ith floor. Hence, the above arguments prove the statement of the Property.

Property 3 *the set of places for ith floor in n-elevators ECPN, i.e., $P_i = \{p_{i_1}, p_{i_2}, \ldots, p_{i_n}\} \subseteq P_F$ has*

$P_i^\bullet \subseteq T_U \cup T_D$ and $^\bullet P_i \subseteq T_U \cup T_D$ where $|P_i^\bullet| = n(m-1)$ and $|^\bullet P_i| = n(m-1)$.

Proof From Property 1, each place $p_{ij} \in P_i$ has $(m-1)$ incoming arcs and $(m-1)$ outgoing arcs. Since $|P_i| = n$, therefore $|P_i^\bullet| = n(m-1)$ and $|^\bullet P_i| = n(m-1)$.

Property 4 *by removing the set of control places P_C and the corresponding arcs in the n-elevator ECPN, we have n single-elevator models.*

Proof Since ECPN is constructed by composing the modules for a single-elevator model, these modules are merged by adding the control places and arcs by using the arc-addition operator. The control places have incoming arcs and outgoing arcs to the transitions of each module for single-elevator models. Hence, by removing the control places and the corresponding arcs, we have N single-elevator models.

Property 5 *each control-place $p_{c_i} \in P_C$ for ith floor in 2-elevator ECPN model has $p_{c_i}{}^\bullet \subseteq T_U \cup T_D$ and $^\bullet p_{c_i} \subseteq T_U \cup T_D$ where $|\ p_{c_i}{}^\bullet| = 2(m-1)$ and $|^\bullet p_{c_i}| = 2(m-1)$.*

Proof for each transition t in the post-set of p_{c_i}, t is used to move upward from ith floor or to move downward from ith floor, that is, either $t \in T_U$ or $t \in T_D$, therefore, $p_{c_i}{}^\bullet \subseteq T_U \cup T_D$. From property 1, ith floor represented by a place p_i has $(m-i)$ transitions of the set T_U to move upward floors and $(i-1)$ transitions of the set T_D to move downward away from ith floor. Therefore, there are total of $(m-1)$ transitions in post-set of p_i to move from ith floor for single elevator. From Property 4, each control place $p_{c_i} \in P_C$ is connected to both the single-elevator models which follow that $\left|p_{c_i}^\bullet\right| = 2(m-1)$. Similar arguments for $\left|^\bullet p_{c_i}\right| = 2(m-1)$.

Property 6 *each control-place $p_{c_i} \in P_C$ for ith floor in n-elevator ECPN model has $p_{c_i}^\bullet \subseteq T_U \cup T_D$ and $^\bullet p_{c_i} \subseteq T_U \cup T_D$ where $\left|p_{c_i}^\bullet\right| = n(m-1)$ and $\left|^\bullet p_{c_i}\right| = n(m-1)$.*

Proof The arguments to establish the Property 5 can be extended to N-elevator ECPN model which follow $p_{c_i}^\bullet \subseteq T_U \cup T_D$ and $^\bullet p_{c_i} \subseteq T_U \cup T_D$ where $\left|p_{c_i}^\bullet\right| = n(m-1)$ and $\left|^\bullet p_{c_i}\right| = n(m-1)$.

Property 7 *each white-transition $t_{w_i} \in T_U$ for jth elevator model in n-elevator ECPN has $^\bullet t_{w_i} = \{r_{i+1}, p_{i-1}, p_{c_i}\}$ and $t_{w_i}^\bullet = \{p_i, p_{c_{(i+1)}}\}$ where $r_{i+1} \in P_R$, p_i, $p_{i+1} \in P_F$ and p_{c_i}, $p_{c_{(i+1)}} \in P_C$.*

Proof The proof of property directly follows from condition (v) and (vi) of the Definition 8 of ECPN.

Property 8 *each black-transition $t_{b_i} \in T_D$ for jth elevator model in N-elevator ECPN has $^\bullet t_{b_i} = \{r_{i-1}, p_{i+1}, p_{c_{(i+1)}}\}$ and $t_{b_i}^\bullet = \{p_{i-1}, p_{c_{(i-1)}}\}$ where $r_{i-1} \in P_R$, p_{i-1}, $p_{i+1} \in P_F$ and $p_{c_{(i-1)}}, p_{c_{(i+1)}} \in P_C$.*

Proof The proof of property directly follows from condition (v) and (vii) of the Definition 8 of ECPN.

5 An application example

For simulation and verification purpose of the proposed system, we construct a model of two elevators with four floors. By using the condition (ix) and (x) of the Definition 8 of ECPN, 16 places and 28 transitions are needed; Fig. 5 describes the complete PN-based model of two elevators. We have 4 transitions represented by $b_1, \ldots, b_4$ in our model which are used as a button to request an elevator and 4 places with the capacity one represented by $r_1, \ldots, r_4$ to enable the elevator to reach at the requested floor. Floor 1, floor 2, floor 3 and floor 4 for single-elevator model is represented by places p_0, p_1, p_2, p_3, p_4, respectively, while places p_5, p_6, p_7, p_8 are used to represent the floor for the second elevator model. Place p_0 representing the first floor (at ground) for elevator-1 has $4 - 1 = 3$ outgoing arcs toward white transitions to move upward on 2nd, 3rd and 4th floors, and p_0 has 3 incoming arcs from black transitions to move downward from 2nd, 3rd and 4th floors. Similarly, all of the other 7 places have 3 incoming and 3 outgoing arcs from the set $T_U \cup T_D$. The set PC has four places, that is, c_1, c_2, c_3, c_4, which are used for the interaction purpose between the two elevators, and each place has six incoming arcs and six outgoing arcs.

The description of its places and transitions is given in Tables 1 and 2. Invariant-based analysis is done by using Petri net Toolbox for MATLAB [31] and PIPE 4.0 [32].

5.1 Analysis of the elevator model

It is important to know whether our Petri net model and original specification of the system has one-to-one functional correspondence. For instance, Petri net model may differ from the original specification of the system. The existence of one-to-one correspondence between Petri net model and its original specification gives a perception about the original system. Another issue during the analysis is the completeness of the requirements specification. This correspondence is basically an input–output relationship of the system. Inputs generated by the environment of the system, while outputs are the response from the system against those inputs. If some inputs given by the

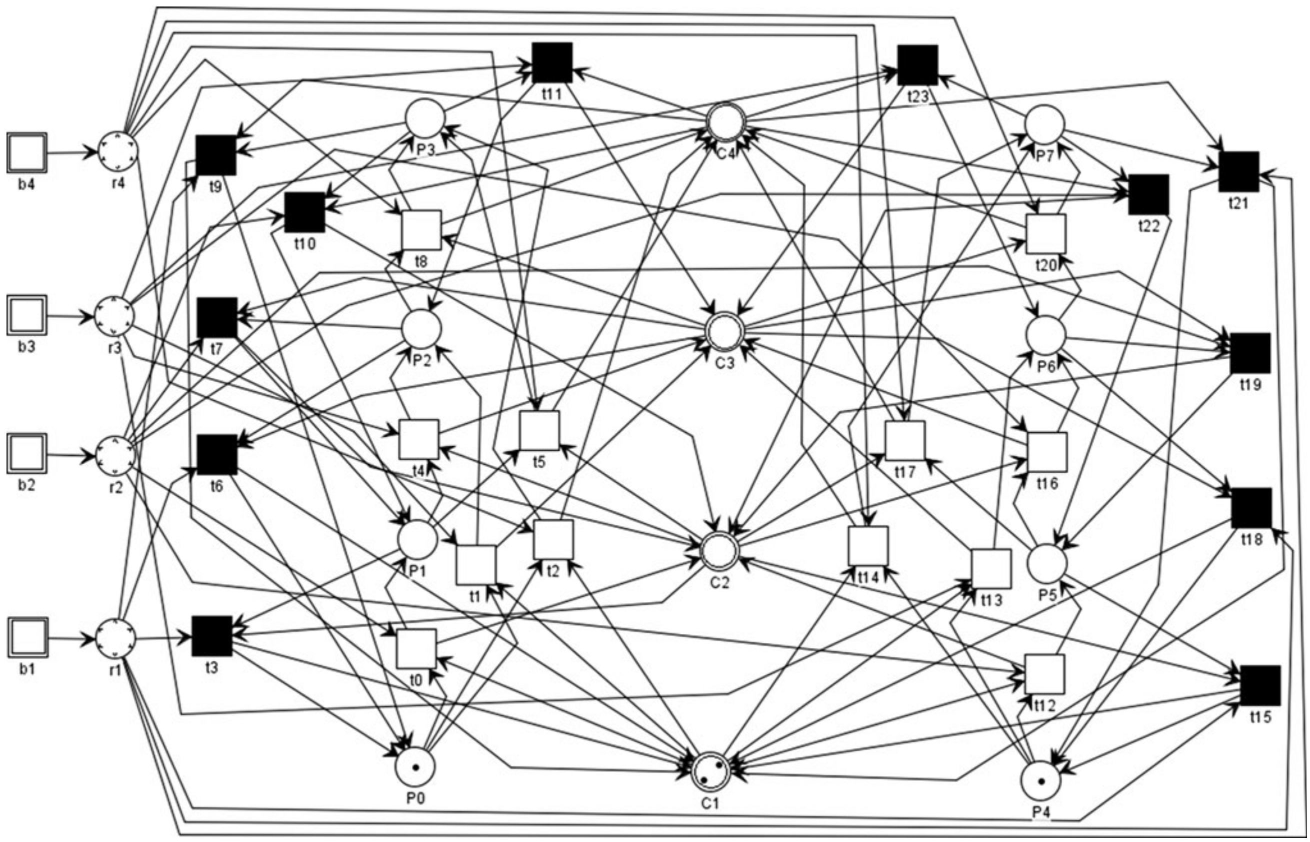

Fig. 5 Petri net model (N) for two elevators

Table 1 Places of Fig. 5

Places	Description
p_0	Ground floor (elevator 1)
p_1	1st floor (elevator 1)
p_2	2nd floor (elevator 1)
p_3	3rd floor (elevator 1)
c_1, c_2, c_3, c_4	Control places for choice of elevators (tokens)
p_4	Ground floor (elevator 2)
p_5	1st floor (elevator 2)
p_6	2nd floor (elevator 2)
p_7	3rd floor (elevator 2)
r_1	Hall-call request from the ground floor
r_2	Hall-call request from 1st floor
r_3	Hall-call request from 2nd floor
r_4	Hall-call request from 3rd floor

Table 2 Transitions of Fig. 5

Transition	Description
t_0 and t_{12}	A car-call request from the ground floor to 1st floor
t_1 and t_{13}	A car-call request from the ground floor to 2nd floor
t_2 and t_{14}	A car-call request from the ground floor to 3rd floor
t_3 and t_{15}	A car-call request from 1st floor to ground floor
t_4 and t_{16}	A car-call request from 1st floor to 2nd floor
t_5 and t_{17}	A car-call request from 1st floor to 3rd floor
t_6 and t_{18}	A car-call request from 2nd floor to ground floor
t_7 and t_{19}	A car-call request from 2nd floor to 1st floor
t_8 and t_{20}	A car-call request from 2nd floor to 3rd floor
t_9 and t_{21}	A car-call request from 3rd floor to ground floor
t_{10} and t_{22}	A car-call request from 3rd floor to 1st floor
t_{11} and t_{23}	A car-call request from 3rd floor to 2nd floor
b_1	A hall-call generated on the ground floor
b_2	A hall-call generated on 1st floor
b_3	A hall-call generated on 2nd floor
b_4	A hall-call generated on 3rd floor

environments of the system are not in requirements specification, the system does not respond properly to these inputs. Similarly, consistency of the requirement specification is also an issue. For a given permissible, temporal combination of inputs, a requirement specification allows two or more permissible, temporal combination of outputs inconsistency. This is due to unclear, imperfect and frequently incorrect perception of system functions. In this paper, we are going to discuss the analysis of elevator system by using *incidence matrix* [12, 13]-based method. The matrix-based approach of the analysis also represents the dynamic behavior of Petri nets.

Now, we use the two main concepts of incidence matrix: *P-invariant* and *T-invariant* for analyzing liveness and boundedness of the Petri net model. Definition 3 in Sect. 2 follows that for any matrix C, there is an integer solution x of $C \cdot x = 0$, which is known as T-invariant. The nonzero entries of a T-invariant represent the firing counts of transitions belonging to that firing sequence which transform a marking M_0 back to M_0. As a T-invariant states the transitions containing a firing sequence transforming a marking M_0 into M_0 and the number of times these transitions appear in this sequence, it does not specify the order of firing these transitions.

5.1.1 Incidence matrix (C)

$$
\begin{bmatrix}
-1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
-1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
-1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 \\
0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

5.1.2 T-invariants

Transitions (t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}, t_{20}, t_{21}, t_{22}, t_{23}, b_1, b_2, b_3, b_4)

According to the definition, we obtained all T-invariants x_i, where $i = 1, 2, 3, \ldots, 40$, as $C \cdot x_i = 0$.

5.1.3 P-invariants

Places (p_0, p_1, p_2, p_3, c_1, c_2, c_3, c_4, p_4, p_5, p_6, p_7, r_1, r_2, r_3, r_4)

Similarly all P-invariants y_j, where $j = 1, 2, 3, \ldots, 17$, are obtained as $yj^T C = 0$.

Liveness Liveness of the application example of ECPN can be analyzed by using the concept of T-invariant. A Petri net model is covered by T-invariants if and only if, for each transition t in the net, there exists a positive

$$x_1 = [1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0]^{\mathrm{T}}, \quad x_2 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0]^{\mathrm{T}}$$

$$x_3 = [0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2\ 0]^{\mathrm{T}}, \quad x_4 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 2\ 0]^{\mathrm{T}}$$

$$x_5 = [0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0]^{\mathrm{T}}, \quad x_6 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0]^{\mathrm{T}}$$

$$x_7 = [1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 2\ 0]^{\mathrm{T}}, \quad x_8 = [0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0]^{\mathrm{T}}$$

$$x_9 = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 2\ 0]^{\mathrm{T}}, \quad x_{10} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0]^{\mathrm{T}}$$

$$x_{11} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1]^{\mathrm{T}}, \quad x_{12} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1]^{\mathrm{T}}$$

$$x_{13} = [0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1]^{\mathrm{T}}, \quad x_{14} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1]^{\mathrm{T}}$$

$$x_{15} = [0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1]^{\mathrm{T}}, \quad x_{16} = [0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1]^{\mathrm{T}}$$

$$x_{17} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1]^{\mathrm{T}}, \quad x_{18} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1]^{\mathrm{T}}$$

$$x_{19} = [1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1]^{\mathrm{T}}, \quad x_{20} = [0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1]^{\mathrm{T}}$$

$$x_{21} = [0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1]^{\mathrm{T}}, \quad x_{22} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1]^{\mathrm{T}}$$

$$x_{23} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]^{\mathrm{T}}, \quad x_{24} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 1]^{\mathrm{T}}$$

$$x_{25} = [0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1]^{\mathrm{T}}, \quad x_{26} = [0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1]^{\mathrm{T}}$$

$$x_{27} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0\ 1\ 1]^{\mathrm{T}}, \quad x_{28} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1]^{\mathrm{T}}$$

$$x_{29} = [0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1]^{\mathrm{T}}, \quad x_{30} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1]^{\mathrm{T}}$$

$$x_{31} = [1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}, \quad x_{32} = [0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}$$

$$x_{33} = [1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}, \quad x_{34} = [0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}$$

$$x_{35} = [0\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 2\ 1]^{\mathrm{T}}, \quad x_{36} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}$$

$$x_{37} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}, \quad x_{38} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1]^{\mathrm{T}}$$

$$x_{39} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 1]^{\mathrm{T}}, \quad x_{40} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1\ 2\ 1]^{\mathrm{T}}$$

$$y_1 = [1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_2 = [0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_3 = [1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_4 = [0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_5 = [0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_6 = [1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_7 = [0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_8 = [1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_9 = [0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_{10} = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_{11} = [0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_{12} = [0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_{13} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_{14} = [1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_{15} = [0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}, \quad y_{16} = [1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

$$y_{17} = [1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0]^{\mathrm{T}}$$

T-invariant x such that $x(t) > 0$ [12]. Furthermore, a Petri net is live and bounded if it is covered by T-invariants [12]. This is a necessary but not sufficient condition. From the T-invariants given above, we have $\forall t \in T$, $\exists x_i$ such that $x_i(t) > 0$, where $i = 1, 2, 3, \ldots, 40$. Thus, the given 2-elevator, 4-floor model of ECPN is live.

Boundedness It is known that a Petri net model is covered by P-invariants if and only if, for each place p in the net, there exists a positive P-invariant y such that $y(p) > 0$ [12]. Furthermore, a Petri net is structurally bounded if it is covered by P-invariants and the initial marking M_0 are finite. From the P-invariants given above, each place is a part of y_j, $j = 1, 2, 3, \ldots, 17$. Thus, given model is structurally bounded.

Reversibility Reversibility is the important property which enforces the multi-elevator model to reach its initial state. By using the definition of T-invariant, the given model is reversible.

By the help of this property, we are sure that our system is structurally bounded. As our net starts from a bounded initial marking and each place of our net is also belongs to some invariant support, so the net is bounded. Similarly, our net N is covered by T-invariants, so the net is live and reversible as well.

6 Conclusions

The elevator control system is basically a complex system to implement even for a single elevator. It becomes more complex and critical when more than one elevator is being used by a single control system. We proposed a component-based Petri nets approach for modeling the multi-elevator system and performed its analysis.

In this paper, we introduced a conceptual model know as MECS based on multi-agent control system. In this technique, collective-selective/up–down approach was used. By using this method in MECS, the waiting time of passengers was reduced. The main objective of this research was to find the cooperation between elevators. After generating a hall-call at any floor, an elevator used selective method, and during the movements of elevators, other calls were handled by collective method. An elevator was selected by MECS after checking different parameters by using an interactive protocol proposed in [30], for example, movement direction, capacity, distance between elevators and requested floor and waiting time of elevators if all were free. All elevators worked as agents of MAS, and interaction between elevators was centrally controlled by MECS on the basis of tokens' movement in CPs.

We used P/T-net which is a low-level PN because their verification and implementation is simple even for large models as compared to high-level PNs. Our proposed MECS is a general model which is for m floors and n elevators. Incidence matrix-based analysis using P-invariant and T-invariant was used to verify the functionality of the model. For testing purposes, a two-elevator model for four floors was presented. The results generated by P/T-invariants, and it has been shown that our proposed model was live, bounded, reversible and deadlock-free.

References

1. Cassandras CG, Lafortune S (2008) Introduction to discrete event systems, 2nd edn. Springer, New York
2. Ramadge PJ, Wonham WM (1989) The control of discrete event systems. Proc IEEE 77:81–98
3. Sandor M, ichi Ken A, Nakagawa M, Sudo T (2008) Recent trends in elevator group control systems. In: The 23rd International Technical Conference on Circuits/Syst, Computers and Communications (ITC-CSCC 2008), pp 697–700
4. Yang S, Tai J, Shao C (2009) Dynamic partition of elevator group control system with destination floor guidance in up-peak traffic. J Comput 4(1):45–52
5. Alvaro A, Daniel M, de Carvalho GC, Llanos CH (2010) Modeling of an elevator group control system using programmable logic control and destination control system. ABCM Symp Ser Mechatron 4:433–444
6. Sorsa J, Hakonen H, Siikonen ML (2005) Elevators selection with destination control system. Elevators Technol
7. Kim CB, Seong KA, Lee-Kwang H, Kim JO (1998) Design and implementation of a fuzzy elevator group control system. IEEE Trans Syst Man Cybern Part A 28(3):277–287
8. Liu J, Liu Y (2007) Ant colony algorithm and fuzzy neural network based intelligent dispatching algorithm of an elevator group control system. IEEE Int Conf Control Autom
9. Ikeda K, Suzuki H, Kita H, Markon S (2008) Examplar-based control of multi-car elevators and its multiple-objective optimization using genetic algorithm. Conference system computers and communication
10. Nikovski D, Brand M (2004) Exact calculation of expected waiting times for group elevator control. Mitsubishi electric research laboratories
11. Yang X, Zhu Q, Xu H (2008) Design and practice of an elevator control system based on PLC. Workshop on Power Electronics and Intelligent Transportation System
12. Reisig W (1991) Petri nets, an introduction, 2nd edn. Springer, Berlin
13. Peterson JL (1981) Petri net theory and the modeling of systems. Prentice-Hall, Englewood Cliffs
14. Ahmad F, Huang HJ, Wang XL (2011) Analysis of the Petri net model of parallel manufacturing processes with shared resources. Inf Sci 181:5249–5266
15. Ahmad F, Khan SA (2013) Specification and verification of safety properties along a crossing region in a railway network control. Appl Math Model 37(7):5162–5170
16. Tasi C-J, Fu L-C, Hsu Y-J (1992) Modeling and simulation for flexible manufacturing systems using Petri net. 2nd Int Conf Automat 4:31–38
17. Lin C-H, Fu L-C (1996) Petri net based dynamic scheduling of an elevator system. IEEE Int Conf Robotics Automat 1:192–199
18. Miyagi PE, Villani E, Gustin GDB, Maruyama N, Santos Filho DJ (1999) Petri net approach for modelling system integration in

intelligent buildings. Presented at COBEM 99—15th Brazilian Congress of Mechanical Engineering, pp 22–26
19. Ho Y-W, Fu L-C (2000) Dynamic scheduling approach to group control of elevator systems with learning ability. Proceedings of the 2000 IEEE International Conference on Robotics & Automation, San Francisco
20. Huang Y-H, Fu L-C (1998) Dynamic scheduling of elevator systems over hybrid Petri net/rule modeling. IEEE International Conference on Robotics & Automation Leuven
21. Cho YC, Gagov Z, Kwon WH (2000) Timed Petri net based approach for elevator group controls. Trans Control Autom Syst Eng 2(2):121–130
22. Strakosch GR (1998) Vertical transportation: elevators and escalators. Wiley, New York
23. Barney G, dos Santos SM (1985) Elevator traffic analysis, design and control, 2nd edn. Institution of Engineering and Technology, England
24. Munoz DM, Llanos CH, Ayala-Rincon M, van Els RH (2008) Distributed approach to group control of elevator systems using fuzzy logic and FPGA implementation of dispatching algorithms. Eng Appl Artif Intell 21:1309–1320
25. Kotb YT, Beauchemin SS, Barron JL (2007) Petri net-based cooperation in multi-agent systems. Fourth Canadian Conference on Computer and Robot Vision CRV 07, IEEE 2007, pp 123–130
26. Krogh BH (1987) Controlled Petri nets and maximally permissive feedback logic. Proceedings of the 25th Annual Allerton Conference. University of Illinois, Urbana, pp 317–326
27. Ichikawa A, Hiraishi K (1988) Analysis and control of discrete event systems represented by Petri nets. Discrete event systems: models and applications, (Varaiya P, Kurzhanski AB (eds)) Lecture notes in control and information sciences. Springer, New York, 103:115–134
28. Ahmad F, Khan SA (2012) Module-based architecture for periodic job-shop scheduling problem. Comput Math Appl 64(1): 1–10
29. Girault C, Valk R (2010) Petri net of systems engineering: a guide to modeling, verification and applications, 2nd edn. Springer, Berlin
30. Capkovic F (2007) Modelling analysing and control of interactions among agents in MAS. Comput Inf 26:507–541
31. Petri Net Toolbox for MATLAB. http://www.ac.tuiasi.ro/pntool/
32. PIPE. http://pipe2.sourceforge.net/