

# Whether to Invest in Energy Saving Technologies or Elsewhere - A Decision Support Tool

Mohammad Moiez Gohar

A Thesis Submitted to  
Indian Institute of Technology Hyderabad  
In Partial Fulfillment of the Requirements for  
The Degree of Master of Technology

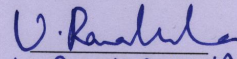


Department of Computer Science & Engineering

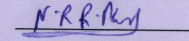
June 2015

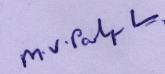
## Approval Sheet

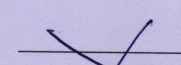
This Thesis entitled Whether to Invest in Energy Saving Technologies or Elsewhere - A Decision Support Tool by Mohammad Moiez Gohar is approved for the degree of Master of Technology from IIT Hyderabad

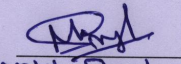
  
U. RAMAKRISHNA  
( ) Examiner

Dept. of CSE  
IITH

  
( ) Examiner  
Dept. CSE  
IITH

  
(Dr. M V Panduranga Rao) Adviser  
Dept. of CSE  
IITH

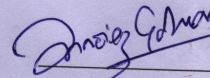
  
( ) Co-Adviser  
Dept. of CSE  
IITH

  
Manish Singh  
( ) Chairman  
Dept. of CSE  
IITH



## Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.



(Signature)

\_\_\_\_\_  
(Mohammad Moiez Gohar)

CS13M1004

(Roll No.)

## **Acknowledgements**

At the very outset, I would like to express my sincere gratitude to my thesis adviser Dr. M.V. Panduranga Rao for his constant encouragement, patience and immense knowledge. Without his guidance and persistent help, this dissertation would not have been possible. I would like to thank my colleague Kanishka Chauhan for his continued support and help in understanding the domain related concepts, specifically the discussions on Timed Automata and Model Checking. Finally, I thank the Almighty, my family and friends for their support and constant encouragement.

## Dedication

This work is dedicated to my father Mohammad Moghees Gohar who has instilled in me the principles of truth, honesty, hardwork, and dedication.

## Abstract

Most small scale industries and household complexes in developing countries face a financial dilemma regarding installation of energy saving technologies like smart lighting. Given an initial capital, it is often not clear whether to invest it in such technologies or elsewhere that gives better returns on investment. For example, if the users of the building are sufficiently energy conscious to turn off their personal appliances and lights before leaving a room, it can be argued that the initial capital is better invested elsewhere.

We introduce a tool that helps concerned decision makers compare the savings due to installation of smart technologies against alternative investment that provide returns at (say)  $r$

Our approach is summarized below:

1. For each user  $U_i$  of the building, we create the time sequence of occupancy in various rooms. This is done using existing agent based building occupancy simulation techniques that model the movement of each person as a Markov process.

2. User  $U_i$  turns off personal appliances with a probability of  $p_i$  when leaving a room. Thus,  $p_i$  quantifies the energy consciousness of user  $U_i$ . We suggest methods for estimating  $p_i$ . Had smart technologies been installed, the appliances would have been necessarily turned off for the interval the user remains out of the room. Therefore, if smart technologies are installed, energy savings occur precisely during the intervals when the user leaves the room forgetting to turning off personal devices and lights.

3. This saving is compared against the returns of the alternative investment.

# Contents

Declaration . . . . .	ii
Approval Sheet . . . . .	iii
Acknowledgements . . . . .	iv
Abstract . . . . .	vi
<b>1 Introduction</b>	<b>1</b>
1.1 Background and Motivation . . . . .	1
1.2 Problem Statement . . . . .	1
1.3 Related Work . . . . .	2
1.4 Contribution of this thesis . . . . .	2
1.5 Organization of thesis report . . . . .	2
<b>2 Preliminaries</b>	<b>3</b>
2.1 Probabilistic Timed Automata - PTA . . . . .	4
2.2 Probabilistic Timed Computation Tree Logic - PTCTL . . . . .	5
2.3 Statistical Model Checking using UPPAAL SMC . . . . .	6
2.3.1 Model Formalism . . . . .	6
2.3.2 Query Formalism . . . . .	7
<b>3 Methodology</b>	<b>9</b>
3.1 Assumptions and Inputs . . . . .	9
3.2 The Setting . . . . .	10
3.2.1 Model Checking Based Approach . . . . .	11
3.2.2 Simulation Based Approach . . . . .	20
3.3 Simulation v/s Model Checking - A comparison . . . . .	24
<b>4 Conclusion</b>	<b>25</b>
<b>References</b>	<b>26</b>

# Chapter 1

## Introduction

### 1.1 Background and Motivation

Non-renewable energy is depleting at an alarming rate today, and hence, it is necessary that we take effective measures to save as much energy as possible. One major area where a lot of energy is wasted is household and office electrical power. Modern day homes have multitudes of power consuming devices viz. fans, bulbs, etc. that are left in the state of power-consumption even while there is no one to consume it.

There are many ways to save non-renewable energy, including avoiding the consumption of energy when not in use, using a high-efficiency device to consume power efficiently, and using alternative technology. Of these methods, the easiest and the most economical is to reduce power consumption is avoiding consumption of power when not in use. And this is where the usage of smart-technology comes into picture. Smart-homes today are equipped with smart-lighting systems which automate the consumption of power, which includes switching off the power when not in use. But the usage of such smart technologies also often requires heavy investment. The energy awareness of the consumers plays a major role in the when dealing with the problem of installation of smart technologies to automate the consumption of power. Hence, the investment dilemma.

### 1.2 Problem Statement

Given an initial investment of  $P$  units of money and a rate of return on investment  $r$ , the energy awareness of each user, i.e., the average probability of each user switching off the power, and the occupancy profile based on the average probability of movement of the occupants from one location to another, is it a better option to invest the money in deployment of smart technology (such as smart lighting and smat heating systems) or to invest it elsewhere.



## 1.3 Related Work

In this section we briefly review some important recent work in the area of building occupancy profile generation. Liao [1] presents an agent-based modeling approach to simulate the behavior of all the occupants of a building. Richardson [2] presents a model that generates synthetic active occupancy data, based upon survey data of people's time-use in the UK [6] using the Markov-Chain technique. Page et al [3] present a model that can generate a time series of the state of presence of occupants within a specific zone of a building, and the transition probabilities of the model, corresponding to arriving, leaving and staying in the respective states. However, the model does not simulate the movement of occupants from one zone to another. Wang [4] handles occupant movement as a Markov chain process, using which the model can generate the location for each occupant at discrete time instants.

Although these approaches do give us the occupancy profile, they cannot be used to solve our problem which requires us to answer queries like what is the expected saving if one deploys smart-technology such as smart-lighting and heating systems. For this, we need a model checking approach which will model the system, and check if the desired properties viz. occupancy profile and the expected savings are within a certain range over a certain period of time.

## 1.4 Contribution of this thesis

Broadly speaking, this work is an attempt to solve the investment dilemma using a novel approach - Statistical Model Checking. We aim to provide answers to some very interesting questions arising out of the dilemma such as the expected energy and hence, monetary savings, given the energy awareness of the occupants and the probabilities associated with their movement about the various locations in a building or office environment. We first use the simulation approach to get the energy savings based on the energy awareness of the occupants, and then provide a Stastical Model Checking approach to obtain energy savings due to deployment of smart technology. We then proceed to answer some more non-trivial questions associated with this problem, and show why plain simulation is not suitable to solve the problem at hand.

## 1.5 Organization of thesis report

This thesis report is organized into multiple chapters. Chapter 2 gives an overview of the various concepts, viz. Probabilistic Timed Automata, Probabilistic Timed Computation Tree Logic, Statistical Model Checking, etc involved in this work, along with suitable examples. Chapter 3 of the report explains the two approaches namely, Simulation and Model Checking, in detail, as well as the results obtained from each of the approaches. The results obtained from the two approaches are compared in the last chapter and appropriate conclusions drawn.

## Chapter 2

# Preliminaries

There are numerous ways of testing a system for its correctness. Simulation is the most widely used method for testing the correctness of the system [5]. However, simulation alone cannot cover all the possible scenarios in the system, and hence some errors may remain undetected. This is where formal methods come into play. Formal methods employ mathematical techniques to check the system behaviour covering all the possible scenarios. Model checking is a formal method used to verify the correctness of a program or system. For complex stochastic systems, probabilistic model checking is a feasible approach.

Numerical model checking algorithms that employ location space exploration, although give the correct answer, suffer from location-space explosion. That is, they can only be used to verify small systems with preferably less no. of locations. The statistical model checking algorithms, on the other hand, are scalable, and give approximate answers. Statistical model checkers combine testing techniques with statistics based algorithms. Statistical Model Checking techniques (SMC) [6, 7], are thus, some kind of a trade-off between testing and formal verification. SMC differs from Monte Carlo simulations used in industries in that it employs a formal model of the system. SMC based verification technique monitors several simulations of a system that behaves in a stochastic manner, i.e., the locations and the events that the system goes into is determined based on some probability associated with each event. The results obtained from such simulations are then used together with statistical methods to get an overall estimate of the probability of the system's behaviour. SMC allows us to check some very complex behavioural properties of the system, such as "Does the system reach some location within certain time limit?", which can not be (easily) inferred from Monte Carlo simulations. As opposed to the exhaustive approach (eg. numerical model checking), the statistical model checking does not guarantee a result with 100% confidence. But, the error-probability can be bound by using sufficient number of runs. Simulation-based methods are known to be far less memory and time intensive than exhaustive ones, and are sometimes the only option [8].

Our approach employs the statistical model checking approach which requires that the model be

constructed as a network of Stochastic (or Probabilistic) Timed Automata. The properties to be checked are represented in the form of Probabilistic Timed Computation Tree Logic (PTCTL), which is an extension of TCTL for querying systems that exhibit both timed as well as stochastic behaviour. The following section discusses PTA and PTCTL, and finally Statistical Model Checking (SMC). But before that, let us discuss the basics of building occupancy simulation.

## Building Occupancy Simulation

The building consists of several rooms. Occupancy of each occupant successive time intervals is modeled as a Markov process. The occupancy profile represents which occupant is in which room at what time. This is obtained using a p-matrix, which denotes the probability of movement of each occupant from one location to another. The p-matrix is defined for what is called as an *event*. There are three events, viz. *Coming to office*, *Walking Around*, and *Going back from office*.

### 2.1 Probabilistic Timed Automata - PTA

The Probabilistic Timed Automata (due to Beauquier [9]) are basically an extension of Timed Automata (due to Alur and Dill) [10]) used to model systems that exhibit both timed and stochastic behaviour.

#### Definition

Before we define PTA, let us see what Timed Automata is. A timed automaton is a finite location machine extended with a finite set of real-valued clocks. All the clocks are synchronized and thus increase at the same speed. The transitions of the automaton are enabled upon satisfaction of constraint(s) on clock values and/or integers. Clocks can be reset. PTAs are a probabilistic version of a timed automaton obtained by associating a set of actions with each location. Each of these actions defines the probabilistic distribution of the next locations if this action is chosen.

Formally, a probabilistic timed automaton is a tuple  $A = (S, S_i, C, E, F, A, \rho)$  where

- $S$  is a finite set of locations,
- $S_i \in S$  is the initial location,
- $C$  is a finite set of real-valued clocks,
- $E \subseteq S \times S(C) \times \text{guard}(C)$  is the set of edges between locations, each edge being labeled by a set of clocks and by a clock constraint.

For the edge  $(s, s', X, \delta) \in E$  from  $s$  to  $s'$ , the set  $X \subseteq C$  gives the set of clocks to be reset to zero and is a clock constraint in  $\text{guard}(C)$  to be satisfied when using this edge.  $P(C)$  is the power set of  $C$ .

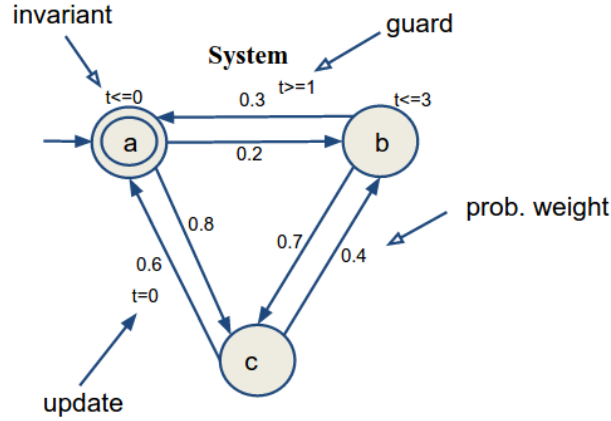


Figure 2.1: Example PTA

- $F$  is the set of accepting locations which is used to define acceptance conditions;
- Associated with each location  $s$  is a non-empty set of actions  $A_s$  which gives the set of actions that can be scheduled when the automaton is in location  $s$ . The sets  $A_s$  are disjoint;
- $\rho : A \times E \rightarrow [0,1]$  is the probability transition function, such that for all  $s \in S$  and  $a \in A_s$ ,

$$\sum_{e \in E_s} \rho(a, e) = 1$$

where  $E_s$  is the set of outgoing edges from  $s$ .

Informally, the maximum time an automaton can remain in a location is determined by the invariant condition(s) associated with the clock variables in that location. A guard on the other hand, determines the earliest the automaton can make transition to another location. Such automata can be constructed to model real time reactive systems. Queries can then be framed in temporal logic to investigate the temporal behaviour of such an automata. Model checking algorithms are designed to answer such queries.

An example PTA explaining the various aspects, viz. invariant, guard and updates is shown in figure 2.1. The system starts in the location  $a$ , and makes transition from one location to another based on the *probabilistic weight* assigned to the transition. For a detailed explanation of PTA, see [9].

## 2.2 Probabilistic Timed Computation Tree Logic - PTCTL

Computation tree logic (CTL) is a branching-time logic used in formal verification of software or hardware systems, which determines if a given system possesses safety or liveness properties. These properties are expressed in CTL. PTCTL is an extension of TCTL or Timed Computation

Tree Logic, that enables us to answer such queries for systems that exhibit both timed and probabilistic behaviour. PTCTL is obtained by extending TCTL with probabilistic operator  $P \bowtie_p$  [.] used in PCTL (Probabilistic Computation Tree Logic). Armed with this operator, PTCTL can be used to ask queries like ‘With probability at least 0.96, does the system reach location s’ within time period 5?’. In PTCTL this query/property to be checked can be expressed as :  $P_{\geq 0.96}[x \leq 3 \text{ and System.s}]$ , where System is the name of the PTA representing the system. Some example PTCTL queries are listed below for the PTA shown in figure 2.1.

- $P[\leq 5] \text{ System.a}$ , enquires about the probability of the system being in location a within time 5.
- $P[\leq 4] \text{ System.c} \geq 0.98$ , asks if the probability of the system being in location c within time 4 is at least 0.98.
- $P[\leq 6] \text{ System.c} \geq P[\leq 4] \text{ System.c or System.b}$ , asks if the probability of the system being in location c within time 6 is at least as much as that of it being in either location b or location c within time 4.

Let us now discuss Statistical Model Checking using UPPPAL.

## 2.3 Statistical Model Checking using UPPAAL SMC

UPPAAL is a toolbox for verification of real-time systems. Uppaal models the systems as network of timed automata extended with integer variables, structures, and synchronization signals. Classical SMC model checkers are incapable of handling timed systems. UPPAAL SMC not only handles timed systems but also implements those tests that can compare two probabilities without computing them. Moreover, the UPPAAL SMC extension allows the user to visualize the results in the form of probability distributions, computation of expected values, etc.

Uppaal SMC is used to represent systems that may behave both stochastically and non-deterministically using networks of timed automata whose clocks may have different rates of progress.

### 2.3.1 Model Formalism

Let us look at a concrete example depicting how stochastic systems are formulated using Uppaal SMC. We consider the Train-Gate example, that illustrates the crossing of a finite number of trains over a bridge. Figures 2.2 and 2.3 are the templates representing the PTAs for train and gate systems, respectively .

The PTA models the system of the crossing of a number of trains over a bridge that has only one track. The passage of trains over the bridge is controlled by a controller which utilizes timing constraints like guards and invariants to stop the trains when necessary and to allow them to pass through when feasible. Figure 2.2 shows the Uppaal SMC model for the train component of the train-gate system. The location Safe doesn’t have any invariant and has a rate



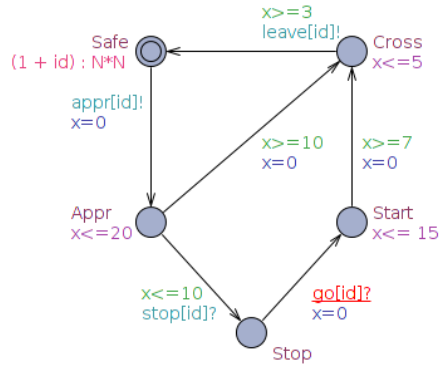


Figure 2.2: Train PTA [13]

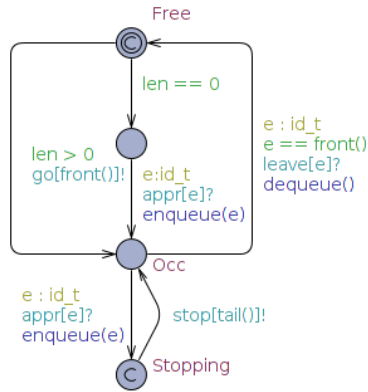


Figure 2.3: Gate-Controller PTA [13]

of exponential distribution associated with it as  $(1 + id)/N^2$ , where  $id$  is the identifier of the train and  $N$  is the number of trains. Trains delay according to this rate and then approach the gate by synchronizing with the gate-controller on signal  $appr[i]$ . The time at which the transition occurs from locations with invariants but without any associated rate(s), e.g. *Cross*, *Appr* and *Start*, is determined by a uniform distribution over the time interval defined by the invariant. For example, the time a transition takes place from the location *Cross* is picked uniformly and randomly between 3 and 5 time units. The gate controller shown in Figure 2.3 is used to track the trains by queuing them in a queue data structure (when the bridge is occupied by some other train), and dequeuing them when the bridge is free. the bridge using an internal queue data structure.

### 2.3.2 Query Formalism

Uppaal SMC provides a number of new queries (in addition to reachability, liveness, safety, and implies) to analyze the stochastic behaviour of timed automata. The Simulator Module of

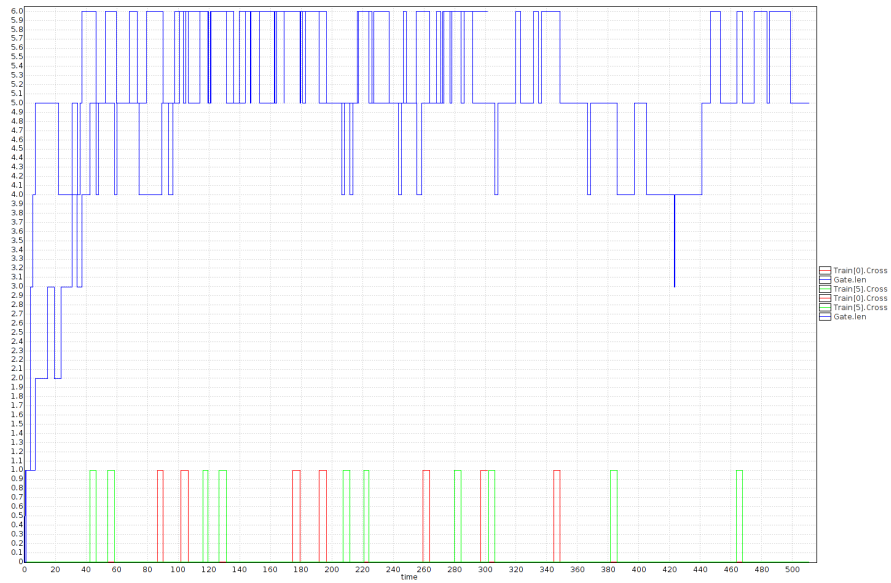


Figure 2.4: Gate Controller PTA

Uppaal SMC allows the user to visualize the values of expressions obtained from the simulation runs. This enables the user to get useful insight on the behavior of the system and ask more relevant queries through the Verifier Module. The syntax to get this done in Uppaal SMC is : *simulate*  $N[\leq bound]$   $Q_1, \dots, Q_k$ , where  $N$  is the number of simulations,  $bound$  is the time bound on the simulations, and  $Q_1, \dots, Q_k$  are the  $k$  (location-based) expressions to be monitored and visualized. For example, *simulate*  $1[\leq 500]$ *Train(0).Cross, Train(5).Cross, Gate.len* runs one simulation over a time bound of 500, and checks at what times *Train0* and *Train5* cross the bridge as well the length of the train queue. The result is shown in Figure 2.4.

# Chapter 3

## Methodology

In this section we discuss our approaches towards solving the problem at hand. But before we discuss the methodology, it is important to go through the assumptions made.

### 3.1 Assumptions and Inputs

- The no. of rooms is fixed and known.
- The no. of occupants is fixed and known.
- Each occupant is assigned an initial room.
- Maximum capacity of each room is equal to the total no. of occupants.
- Energy consumed per device per unit time is fixed and is a constant across all the devices.
- Each occupant uses only one device, and hence consumes only that amount of energy as is consumed by a single device in one unit time.

Some of the assumptions listed above, viz. capacity being equal to the total no. of occupants, or the energy consumption rate of each device being equal seem to be too ideal or unrealistic, but can be relaxed without any significant change in our methodology. This will become clear once we discuss our methodology.

Let us now look at the methodology being proposed.

## 3.2 The Setting

As discussed in earlier section, following are the two approaches our methodology is comprised of:

- Simulation Based Approach
- Model Checking Based Approach

In either of the approaches, following are the inputs:

- Number of rooms
- Number of occupants
- Probability matrix of movement of occupants between locations
- Average energy awareness of each occupant

With the inputs listed as above, each approach generates the **occupancy profile** and the **energy consumption profile** for each occupant at discrete steps of time, known as a **timestep** over a period of time from morning 7:00 am to evening 9 pm. Thus, there are a total of 84 timesteps.

The observation period is divided into 3 parts, known as **events**. The first event is *coming to office*, which takes place from morning 7:00 am to 8:30 am. The event succeeding this event is *walk-around* which lasts from 8:30 am to 5:00 pm, during which the occupants move from one location to another based on the movement probability matrix. The last event is *going to home*, which starts after 5 pm and lasts till 9 pm by which time none of the occupants is allowed to remain in the office.

The occupancy profile of each occupant is maintained in the form of a 3-dimensional array, representing which room each occupant is in at each time. Similarly, the energy consumption profile tells us whether energy was being consumed by a particular person at a particular timestep. The occupancy profile of each occupant is updated at each time step according to the movement of each occupant across various locations. Alongwith the occupancy profile, we maintain a sincerity profile for each occupant, which tells us if the occupant switched off the power being consumed by them upon leaving the room or not. Using these two arrays, we can easily obtain the energy consumption profile for each occupant in each room at each timestep. Once we have the energy profile, we can easily compute the total energy consumed by the occupants of the building in a single day over the aforementioned period of time, which in turn will tell us how many energy units, if any, will be saved if smart technology is deployed.

The following sections describe the two approaches in detail.

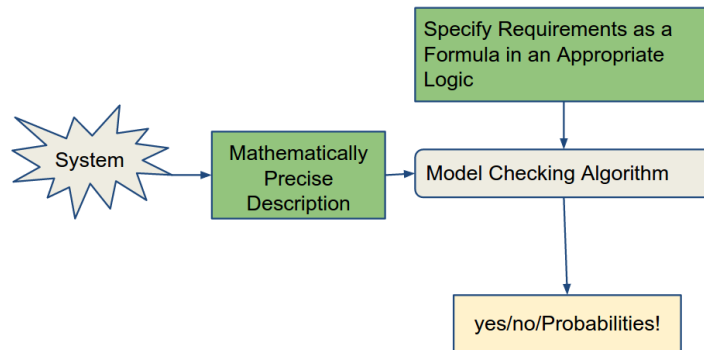


Figure 3.1: Model Checking Approach

### 3.2.1 Model Checking Based Approach

In this approach, we model the system as PTAs or Probabilistic Timed Automata described in earlier sections. Figure 3.1 presents an overview of the model checking approach.

The Model Checking Algorithm takes as input, the mathematical description of the system (Timed Automata in our case), and the requirements or property that need to be satisfied by the system, and outputs a yes or a no with certain probability.

In the section below we discuss how the system is modeled as a network of Timed Automata. And in the section that follows next describes some of the important properties that were checked against the system. The properties are expressed in the form of PTCTL queries as discussed before.

#### Modeling the System

This is the broad strategy of the model checking approach. The system is modeled as three timed automata.

- Timer Automaton
- Person Automaton
- Control Automaton

Let us discuss each automaton in detail.

#### Timer Automaton

The Timer Automaton maintains the timestep for each event. At the end of all transitions, it computes the energy saved if smart technology is used. The automaton works as follows:

The automaton moves from initial state start to the wait state at the end of each 10-minute



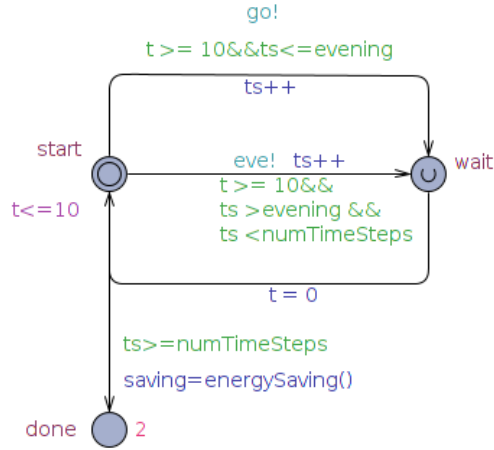


Figure 3.2: Timer Automaton

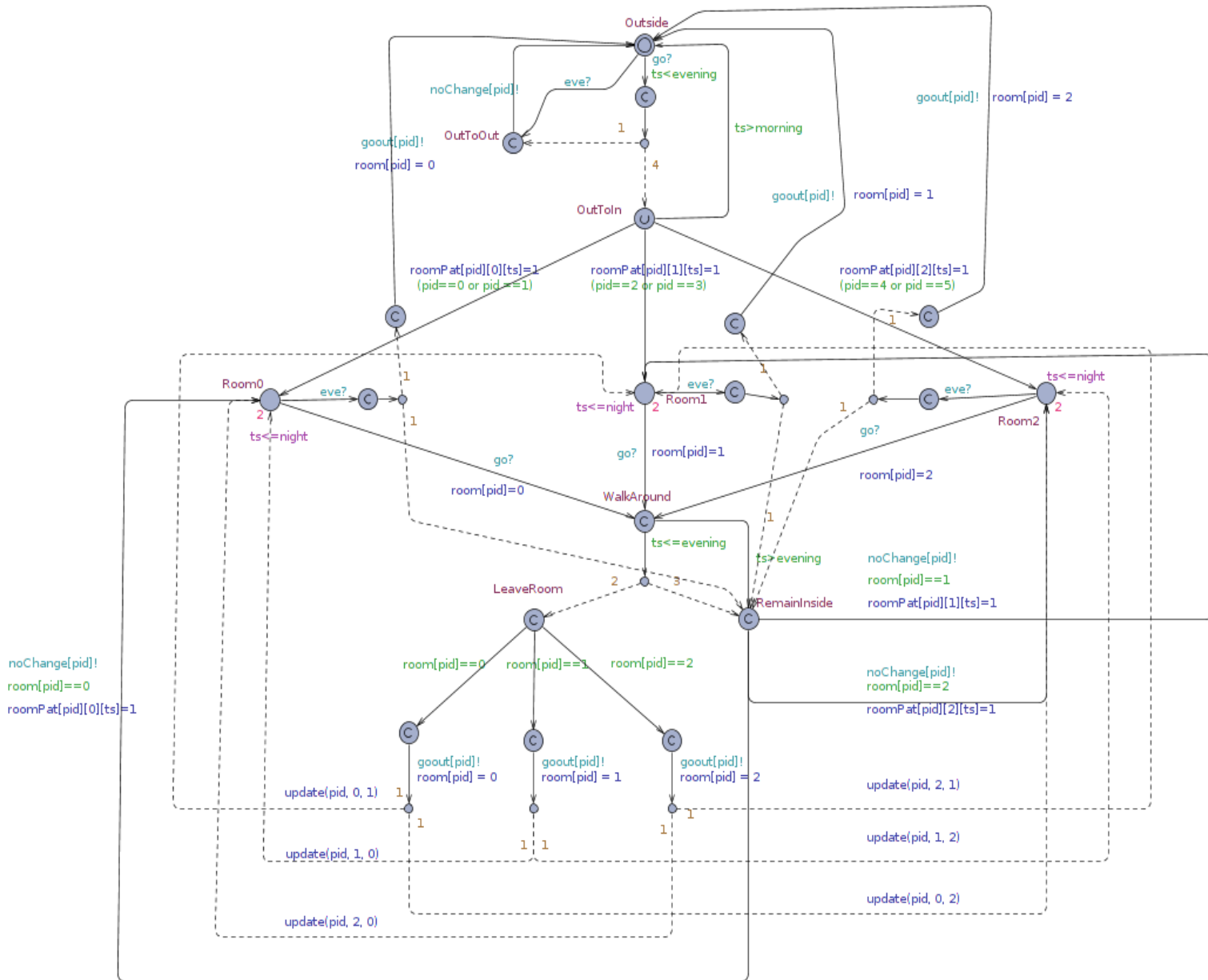
interval. This is ensured by the guard  $t \geq 10$ . A synchronization signal “go” is sent across to the Person Automaton to indicate that the occupant can now move to some other location or stay at the same place, depending on the probability of movement of that occupant to other locations including his own location. The invariant  $t \leq 10$  is added to the start state to ensure that the transition (from start to wait) happens no later than 10 time units. There is one more guard on the edge going from start to wait -  $ts < numTimeSteps$  which indicates that this automaton will run only till the number of timesteps equivalent to one day observations. When  $ts$  becomes equal to or exceeds  $numTimeSteps$ , the automaton moves from start state to done state, and during this transition, the function to compute the energy saved in terms of timesteps is called. This function returns the energy saved based on the energy consumption pattern and the occupancy profile of the occupants as described in earlier sections.

### Person Automaton

This automaton models the movement of the occupants from one location to another. There are 3 rooms, and an outside location. Each occupant starts from outside (represented by the “Outside” state) and moves in to the room assigned to them after receiving the go signal from the timer automaton. The guard  $ts < evening$  ensures that this transition happens only before the last event, ie. evening. This is ensured by the guard  $(pid == 0 \text{ or } pid == 1)$  going from state *OutToIn* to state *Room0*, and so on. This ensures that only person with the  $id=0$  or  $1$ , can move to room 0. Similarly the guards  $(pid == 2 \text{ or } pid == 3)$  and  $(pid == 4 \text{ or } pid == 5)$  ensure that each occupant only goes to their own room when they come from outside. The update  $roomPat[pid][0][ts] = 1$  updates the room occupancy info of room0 in the matrix *roomPat* which basically captures the occupancy profile of each occupant at each timestep. This captures the first event - *Coming to Office*.

The next event is *walking around* in which the occupants move from one room to another based on the probability of movements. The go signal from the timer automaton sets this event in motion, and the automaton moves to the state *WalkAround* from either of the three states viz. *Room0*, *Room1*, or *Room2*. From this state, the automaton either transits to state *LeaveRoom* or *RemainInside* based on the probability weights on the dashed-edges (2/5 and 3/5, here). These weights are directly taken from the movement matrix discussed earlier, and represented as ratio of weights. For example, if the probability of moving from locationA to locationB and to locationC are 0.4 and 0.6 respectively, then it is represented as the probability weights 2 and 3, because the ratio of the probabilities is 0.4:0.6, which is 2:3. Now based upon which Room the occupant was in before leaving his room, the automaton transits to either location A, or B or C. From each of these locations, the automaton transits to either of the states *Room0* or *Room1* or *Room2* depending again, on the probability weights. This transition sets out the signal `goout[pid]` to the control automaton indicating that the person is leaving the room. The update function `update(pid, x, y)` is called to update the occupancy matrix indicating that the person has moved from room x to room y. The guard `ts ≤ evening` on the edge going out from *WalkAround* ensures that walk-around happens no later than evening, after which the occupant has to go home or remain inside the room. If the automaton moves to the state *RemainInside* from the state *WalkAround*, then it will transit to the location it was earlier in. This is ensured by the guard `room[pid] == x`, checking if the person was in room x earlier or not. This is indicated by sending out the signal `noChange` to the control automaton. Along the transition from *RemainInside* to either of the states representing the rooms, `roomPat` matrix is updated to denote that person is still in that room by setting `roomPat[pid][x][ts]` to 1, where *x* is the room number, and *ts* is the timestep.

The last event is *going to home*. This event is captured by the signal *eve* received from the timer automaton. The automaton upon receiving the *eve* signal, moves to either *Outside* or *RemainInside* state denoting that the person has gone home or has remained inside the room he was in. If the person goes out, then the signal *goout* is sent to the control automaton, and the *nochange* signal is sent if he stays inside his room. The loop between the locations *Outside* and *OutToOut* is to indicate that all the events are over and the person stays outside. This is achieved through the synchronization channels *eve* and *noChange*. Thus we have seen how the Person Automata captures each of the three events. Let us now discuss the Control Automaton



### Control Automaton

The control automaton (Figure 3.3) basically deals with the energy part of our problem, that is, it updates the *energypattern* array which captures the information about the energy consumption by the occupants at each timestep in each room. The automaton works as follows.

Initially the automaton is in the start state and then it moves to either the *switchOff* or *forgetBeforeLeaving* based on the energy awareness of the occupant indicated by  $of[pid]$  and

$on[pid]$ . In either case the sincerity array  $sincerity[pid][ts]$  is updated to indicate if the power being utilized was switched off or not. Again, the probabilities of switching off or leaving it on are represented by the probability weights,  $off[pid]$  and  $on[pid]$ .

These three automata thus represent the system and work in synchronization with each other to emulate building occupancy and energy consumption. Let us now discuss the second part of model checking approach - checking the model against required properties.

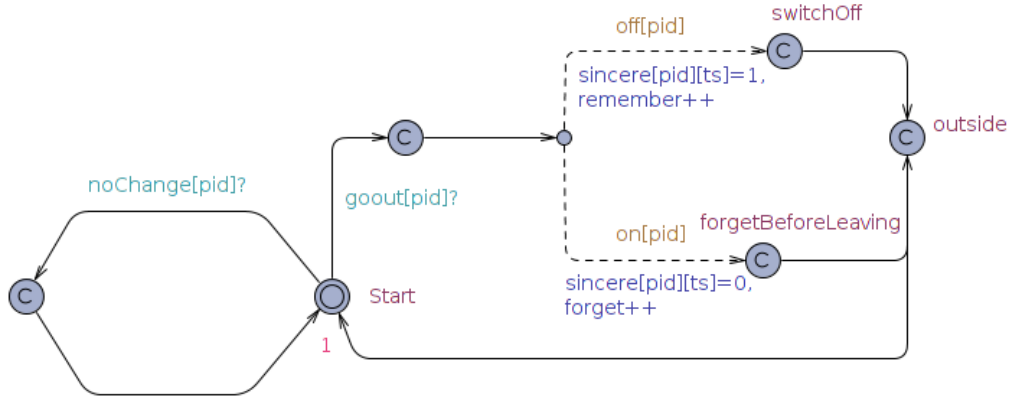


Figure 3.3: Control Automaton

### Checking the Model

As described earlier, the model checking is done by expressing the properties to be checked in the form of PTCTL queries. Some of the most relevant and important queries that were checked against the system are represented in the following table. The first two queries simply enquire about the probability of savings (in terms of timesteps) being in a certain range of values. The next two queries As described earlier, the model checking is done by expressing the properties to be checked in the form of PTCTL queries. Some of the most relevant and important queries that were checked against the system are represented in the following table. The first three queries simply enquire about the probability of savings (in terms of timesteps) being in a certain range of values, and that of count of forget being more than or equal to that of remember. The last three queries are to get the expected maximum values of the variables forget, remember, and savings.

PTCTL Query	Meaning	Result
$\text{Pr} [\leq 1000] (\text{saving} \geq 50 \text{ and saving} \leq 100)$	Prob. that saving is between 50 and 100 within 1000 time period	[0.3898, 0.4898]
$\text{Pr} [\leq 1000] (\text{saving} \geq 100 \text{ and saving} \leq 150)$	Prob. that saving is between 100 and 150 within 1000 time period	[0.0, 0.09738]
$\text{Pr}([100,600]$ (forget $\geq$ remember))	Prob. that the total no. of times occupants forgot to switch off was at least as much as the no. of times they remembered to switch off within time period 100 and 600	[0.9500,1.0000]
$\text{E}[\leq 1000,100] \{\text{max:saving}\}$	With avg. sincerity 0.7, what is the expected max. saving within time period 1000 in a total of 100 simulations	(100 runs) $\text{E}(\text{max:saving})$ =316.31
$\text{E}[\leq 600,100] \{\text{max:remember}\}$	With avg. sincerity 0.3, what is the expected max. no. of times the occupants remember to switch off within time period 600 in a total of 100 simulations	(100 runs) $\text{E}(\text{max:remember})$ =40.96
$\text{E}[\leq 600,100] \{\text{max:forget}\}$	With avg. sincerity 0.3, what is the expected max. no. of times the occupants forget to switch off within time period 600 in a total of 100 simulations	(100 runs) $\text{E}(\text{max:forget})$ =99.51

The plot shown in Figure 3.4 depicts the probability distribution of expected maximum savings (in timesteps) when the average energy awareness of the occupants is 0.3. The x-axis denotes expected values of savings, and the y-axis denotes the probability.

The next plot shown in Figure 3.5 represents the values of difference between the no. of times the occupants forget to switch off with the no. of times when they don't when the average sincerity is 0.3. The horizontal axis denotes time period. The plot was generated with the query: simulate 365 [ $\leq 850$ ] forget-remember]. It is clear from the plot that the count of forget exceeds that of remember when the sincerity is as low as 0.3.

Lastly, the plot in Figure 3.6 shows the plot of the difference in the counts of remember and forget when the average sincerity value is 0.9. As is expected, the difference comes out to be positive after a certain time period owing to high value of average sincerity of the occupants.



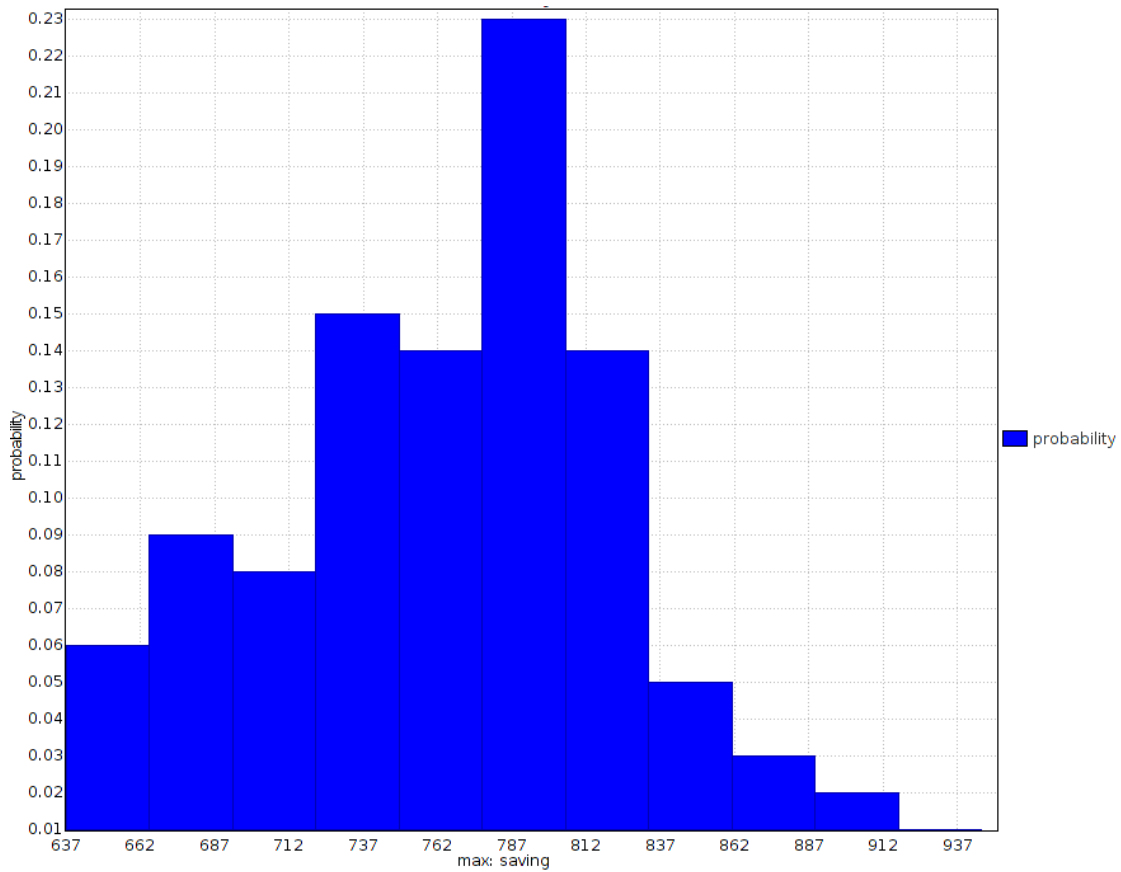


Figure 3.4: Probability Distribution Plot for savings with  $p = 0.3$

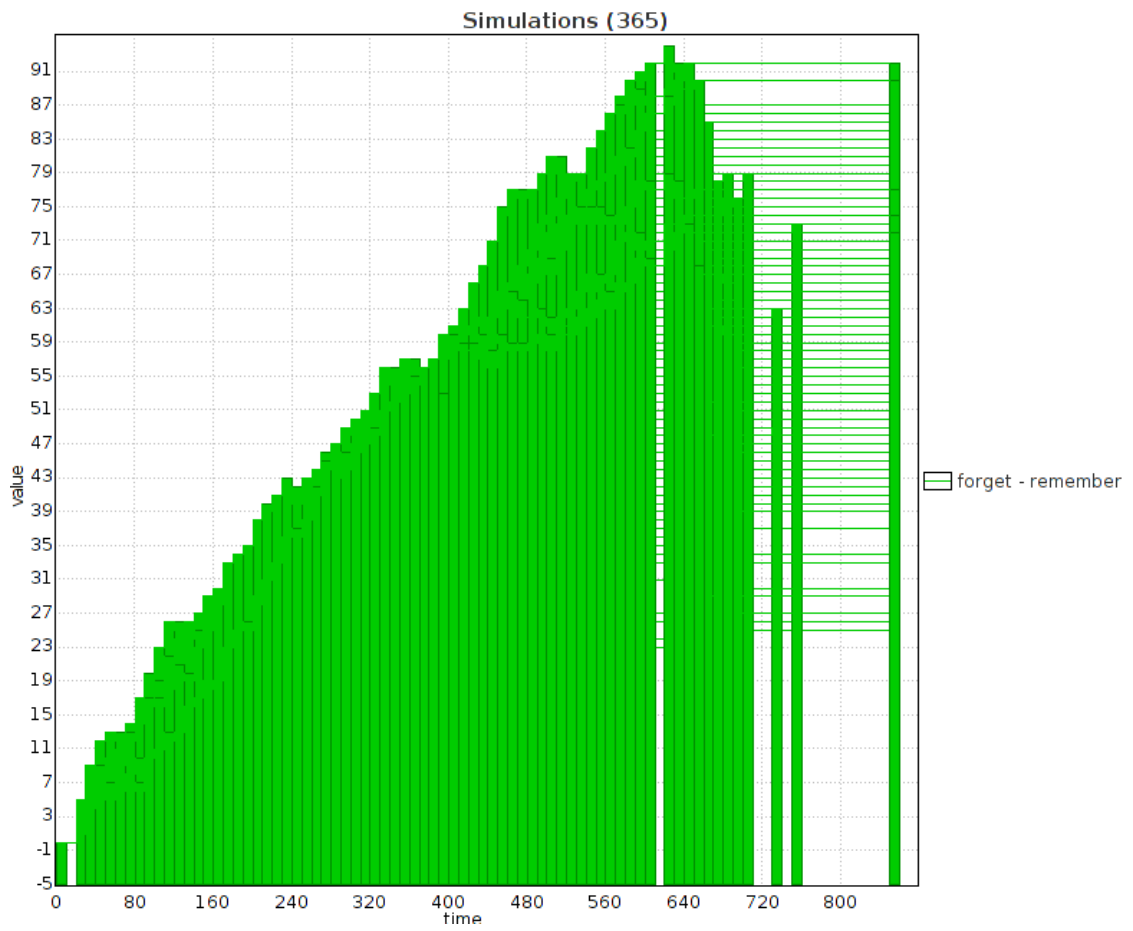


Figure 3.5: Plot showing forget-remember with  $p = 0.3$

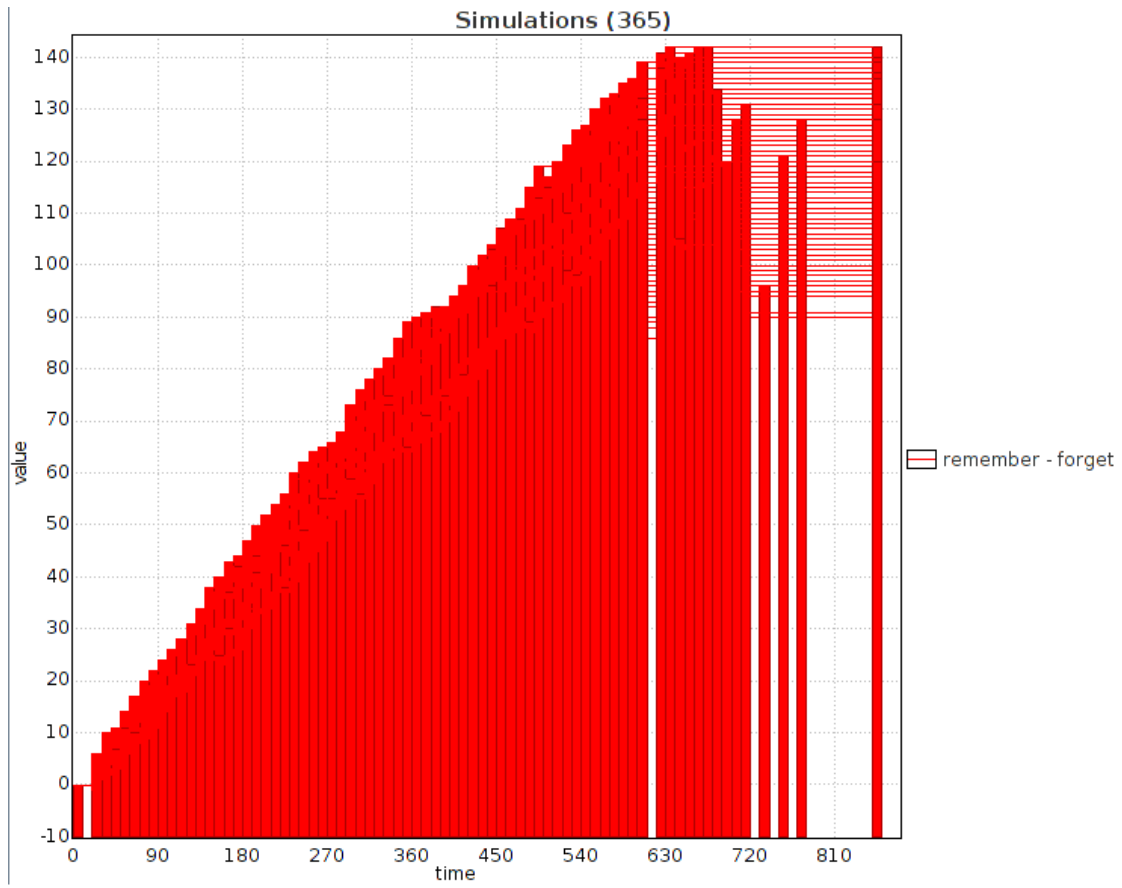


Figure 3.6: Plot showing remember-forget with  $p = 0.9$

Again the query used to generate this plot is: `simulate 365 [≤850] remember-forget]`

	1	2	3
1	0.2	0.3	0.5
2	0.4	0.4	0.2
3	0.2	0.7	0.1

Table 3.1: P-matrix row,col=roomId

	1	2	3
1	0	1	0
2	0	0	1
3	1	0	0

Table 3.2: Location Matrix  
row=roomId, col=personId

	OFF	ON
1	0.2	0.8
2	0.4	0.6
3	0.7	0.3

Table 3.3: Sincerity Matrix row=personId

	1	2	3	...	84
1	0	1	0		1
2	0	0	1		0
3	1	0	0		1

Table 3.4: Person 1 Energy Matrix  
row=roomId, col=timestep

### 3.2.2 Simulation Based Approach

#### Broad Strategy

- Generate occupancy profile using existing building occupancy simulation algorithms. Through this, we get to know the times when a user leaves the room.
- Whenever an occupant leaves the room, he/she remembers to switch off his/her devices with probability  $p$ .
- We are interested in those time intervals when she leaves her room without switching off her devices until she returns because power would have been saved in this interval if smart technologies had been deployed.

#### Data Structures

- P-matrix (Table 3.1)
- Location Matrix (Table 3.2)
- Sincerity Matrix (Table 3.3)
- Energy Consumption Matrix (Table 3.4)

	1	2	3
2	0.4	0.4	0.2

Table 3.5:  $C = LP$

	1	2	3
2	0.4	0.8	1.0

Table 3.6:  $C'$

The first step towards obtaining the occupancy profile for each occupant is to get the next location or the current location at a particular timestep. This is explained as follows.

### Predicting Next Location

Given the location matrix  $L$ , and the  $p$ -matrix  $P$ , we compute matrix  $C = LP$  [Table 3.5] which is the movement probabilities of occupant 2 from their initial location to other rooms.

Next, we compute the cumulative matrix  $C'$  [Table 3.6].

The next step is to generate a random number  $r$  between 0 and 1, and compare it with each of the values in  $C'$ . If the random number generated is 0.6, for example, then the next location of occupant 2 is returned as 2. Similarly if it is 0.3, then the next location will be 1. This approach gives us a complete occupancy profile of each occupant at each timestep.

Based on a single run of the program, the occupancy profile for room 1 shown in Figure 3.7 was obtained:

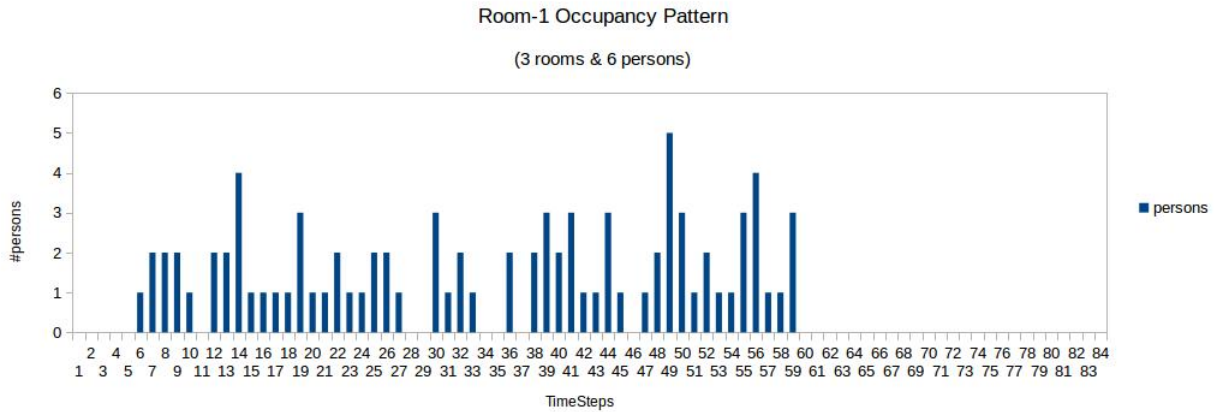


Figure 3.7: Room 1 Occupancy

### Computing Energy Saved

Recall that, the amount of energy saved is computed as depicted in Figure 3.8 Therefore, Power saved per room per timestep if smart technology is deployed = (no. of persons leaving a location - no. of persons switching off their devices)\* $W$ , where  $W$  is the average power consumed by one device in one timestep, in kWh. For example, a Laptop/desktop of 20 watts if run for 10 min, consumes  $W=20*(10/60)/1000$  kWh.

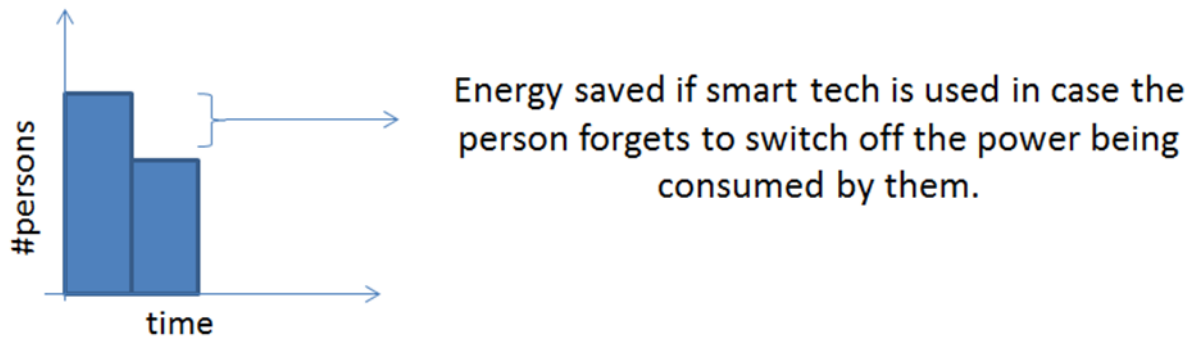


Figure 3.8: Computing Energy Saved

Based on the occupancy profile and the energy profile thus generated, we have the energy saved as depicted in Fig. 3.9.

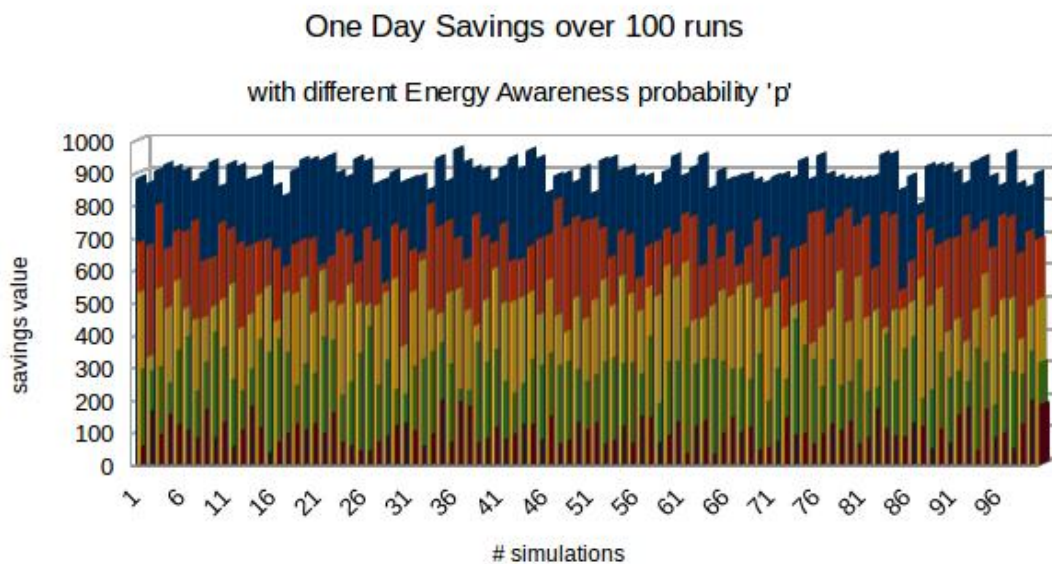


Figure 3.9: Energy Saved

This figure depicts one day savings in terms of the total no. of timesteps in which the occupant forgot to switch off the power they were using just before leaving the room. The x-axis is the simulation number, and the y-axis is the savings in timesteps. The graph is plotted with different values of avg. sincerity level or energy awareness of the occupants. As is clear from the plot, energy savings is quite high when the avg. sincerity level is less. But as the sincerity level increases, the energy saved decreases, which is as expected.

### Conversion of timesteps to money

Initial Investment: INR 5108.80 (\$80<sup>1</sup>) x 3 rooms = INR 15326.4

Cost per unit: 1kWh costs INR 5. (7 for commercial)

No. of Units Consumed = (Watt x Hour) / 1000

Average wattage per device: 20 W (Assumption)

Since we are calculating saving per TimeStep of 10 min each, no of hours = 10/60.

Average One Day Savings = Power Consumption corresponding to number of TimeSteps (in which the power wasn't switched off when not in use).

Having obtained the energy savings from the simulations, and given the initial capital and the rate of return on investment, we can now answer whether the money should be invested in deployment of smart technology or not.

Table 3.7 given below draws a comparison between the investment in smart technology and elsewhere.

Avg. Sincerity,p	1 day TimeSteps	5 year savings	r=6%, ST?	r=8%, ST?	r=10%, ST?
0.1	901.14	50159.70	y	y	y
0.3	694.24	38643.13	y	y	y
0.5	502.61	27976.53	y	y	y
0.7	299.05	16645.87	y	y	y
0.9	103.56	5764.41	y	n	n
1.0	0	0	n	n	n

Table 3.7

### Pros & Cons of this approach

This approach is easy to implement and is scalable, in that the size of the system (viz. number of rooms, number of persons, etc.) can be increased easily without any significant changes. But since a simulation covers only one random trace of the system, the accuracy relies heavily on the number of simulations performed. Also plain simulation can be used only to study straightforward properties. For example, queries like “What is the probability of saving P amount of money in the first 1000 Time units due to the deployment of energy aware technology?” cannot be (easily) answered from the results obtained from simulation.

Before we conclude, it is important for us to compare the two approaches discussed, viz. simulation and model checking. The following section presents a comparison of the two approaches.

<sup>1</sup><http://www.pocket-lint.com/news/130002-smart-lighting-solutions-here-are-seven-options-to-choose-from>

### 3.3 Simulation v/s Model Checking - A comparison

The table below shows the comparison between the two approaches. The comparison is based upon the values of energy saved in terms of timeteps for varying values of avg. sincerity level of the occupants.

<b>Sincerity Level</b>	<b>Simulation</b> (Expected 1 day savings)	<b>Model Checking</b> (Expcted max savings in 1 day)
0.1	901.14	965.68
0.3	694.24	766.32
0.5	502.61	536.71
0.7	299.05	316.31
0.9	103.56	105.36
1.0	0	0

As is apparent from the comparison, the values of savings in both the approaches are comparable and close to each other as expected.



## Chapter 4

# Conclusion

Small scale businesses and households face a dilemma regarding investment in smart technologies. A deciding factor in making this decision is the average energy awareness of the users of the building. Based on this observation, and using the occupancy profile of the users over different time intervals in a day, we report two decision making approaches - Statistical Model Checking, and Plain Simulation. The results obtained from both the approaches seem to complement each other. However, Statistical Model Checking has an edge over simulation in that it answers a wide variety of queries pertaining to the stochastic behaviour of the system, which cannot be (easily) answered using plain simulation.

We have thus seen that the model checking approach can be applied to solve the dilemma of investment given the parameters discussed above and earlier.

# References

- [1] C. Liao and P. Barooah. An integrated approach to occupancy modeling and estimation in commercial buildings. In American Control Conference (ACC), 2010. IEEE, 2010 3130–3135.
- [2] I. Richardson, M. Thomson, and D. Infield. A high-resolution domestic building occupancy model for energy demand simulations. *Energy and Buildings* 40, (2008) 1560–1566.
- [3] J. Page, D. Robinson, N. Morel, and J.-L. Scartezzini. A generalised stochastic model for the simulation of occupant presence. *Energy and buildings* 40, (2008) 83–98.
- [4] C. Wang, D. Yan, and Y. Jiang. A novel approach for building occupancy simulation. In Building simulation, volume 4. Springer, 2011 149–167.
- [5] M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, and A. Pretschner. Model-based testing of reactive systems: advanced lectures, volume 3472. Springer, 2005.
- [6] T. Héruault, R. Lassaigne, F. Magniette, and S. Peyronnet. Approximate probabilistic model checking. In Verification, Model Checking, and Abstract Interpretation. Springer, 2004 73–84.
- [7] H. L. Younes. Verification and planning for stochastic processes with asynchronous events. Technical Report, DTIC Document 2005.
- [8] D. N. Jansen, J.-P. Katoen, M. Oldenkamp, M. Stoelinga, and I. Zapreev. How fast and fat is your probabilistic model checker? an experimental performance comparison. In Hardware and Software: Verification and Testing, 69–85. Springer, 2008.
- [9] D. Beauquier. On probabilistic timed automata. *Theoretical Computer Science* 292, (2003) 65–84.
- [10] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical computer science* 126, (1994) 183–235.
- [11] K. Sen, M. Viswanathan, and G. A. Agha. VESTA: A Statistical Model-checker and Analyzer for Probabilistic Systems. In QEST, volume 5. 2005 251–252.

- [12] J.-P. Katoen, I. S. Zapreev, E. M. Hahn, H. Hermanns, and D. N. Jansen. The ins and outs of the probabilistic model checker MRMC. *Performance evaluation* 68, (2011) 90–104.
- [13] A. David, K. G. Larsen, A. Legay, M. Mikučionis, and D. B. Poulsen. Uppaal smc tutorial. *International Journal on Software Tools for Technology Transfer* 1–19.