Automatic Timeliness Verification of a Public Mobile Network

Ester Ciancamerla[1], Michele Minichino[1], Stefano Serro[2], and Enrico Tronci[3]

[1] ENEA CR Casaccia, Roma, Italy
{ciancamerlae, minichino}@casaccia.enea.it
[2] TECSIT Telecontrollo e Sistemi, Roma, Italy
stefanoserro@inwind.it
[3] Dip. di Informatica, Università di Roma "La Sapienza", Roma, Italy -
tronci@dsi.uniroma1.it

Abstract. This paper deals with the automatic verification of the timeliness of Public Mobile Network (PMN), consisting of *Mobile Node*s (MNs) and *Base Stations* (BSs). We use the Murphi Model Checker to verify that the waiting access time of each MN, under different PMN configurations and loads, and different inter arrival times of MNs in a BS cell, is always below a preassigned threshold. Our experimental results show that Model Checking can be successfully used to generate worst case scenarios and nicely complements probabilistic methods and simulation which are typically used for performance evaluation.

1 Introduction

This work is in the frame of the evaluation activity of the EU Project SAFETUNNEL [1]. The Project aims at reducing the number of incident inside mono tube alpine road tunnels by preventive safety actions which essentially consist of vehicle prognostics, before tunnel entrance, and vehicle telecontrol, inside the tunnel, to keep safety speed and safety distances among vehicles. Such safety actions are foreseen to be performed by a digital control system, based on a PMN. The system basically consists of a Tunnel Control Centre (TCC) interconnected to on board vehicle subsystems by a PMN. For implementing preventive safety actions it is essential that each vehicle, approaching the tunnel, can get communication with the TCC in real time. Time violations in detecting dangerous tunnel conditions and/or in taking the corrective actions could lead both the control system and the tunnel under control to unsafe situations. The time response of the system, which relies on a PMN, also depends upon the time response of the PMN.

Alternative design solutions for PMN, which include the use of GSM mobile network (circuit-switched connections/reserved bandwidth) and GPRS data connection (packed switched connections /shared, unreserved bandwidth), are currently under consideration in SAFETUNNEL Project. In this paper we consider a PMN architecture at a *Cell Level*, which implements a circuit switched connection. The architecture consists of *Mobile Node*s (MNs) and *Base Stations* (BSs), where a

S. Anderson et al. (Eds.): SAFECOMP 2003, LNCS 2788, pp. 35–48, 2003.

BS is viewed as a router connecting the wireless cellular network to the wired part of the network. We investigate the use of the Murphi Model Checker to verify that the maximum waiting access time of any MN, evaluated on many network configurations, network loads and inter arrival times of MNs, is always below a given threshold. The *waiting access time* is intended as the interval of time between the instant in which a MN asks for a communication channel to a BS, and the instant in which the MN gets such a communication channel. In SAFETUNNEL context, MNs represent vehicles (e.g. trucks).

Here we are dealing with a protocol like system. From [2] it is known that for such systems *explicit* model checking can outperform *implicit* (i.e. OBDD based model checking [3, 4, 5]). This is why we decided to use an explicit Model Checker, Murphi [4,6] for our analysis.

As usual when using model checking, *state explosion* is the main obstruction to be overcome. To delay state explosion, rather then using standard Murphi [6], we used Cached Murphi [7,8] which, w.r.t. standard Murphi, saves about 40% of RAM possibly with a time overhead. Note also that, given the size of our system, tools using dense time timed automata (e.g. as HyTech, [9]) cannot be used because of state explosion.

Performances are typically evaluated by solving stochastic models in terms of average measures and distributions. Stochastic modeling can be performed by using Markov Chains, Petri Nets or other modeling concepts [10,11,12] and can hardly generate worst case scenarios. This is instead exactly what model checking can do: generate a worst case scenario and compute performances for such worst case scenario. Note, on the other hand, that model checking is not suited for an average case analysis.

The paper is organized as follows. Section 2 describes the PMN at cell level. Section 3 deals with the modeling assumptions of the PMN. Section 4 describes the Murphi model for our PMN. Section 5 shows our experimental results and Section 6 presents some discussions and conclusions.

2 Public Mobile Network

The architecture of the PMN under consideration is at cell level and consists of *Mobile Node*s (MNs) and *Base Stations* (BS) interconnected by a circuit-switched connections/reserved bandwidth, such as the GSM connection. A BS is viewed as a router connecting the wireless cellular network to the wired part of the network.

A MN enters the BS range of action and requests a connection to the PMN. If the BS has the necessary resource, that is a communication channel, it starts the procedure to assign such a channel to the MN. Once connected, the MN communicates through the BS until the MN doesn't leave the BS cell or the BS signal power falls below a certain threshold. A BS may, or may not, be able to provide the connection to a MN, within real time constraints, depending upon the PMN configuration and parameters. Such constraints have also to be satisfied when a MN is accessing a new BS, coming from an old BS (handoff procedure management). If the new BS doesn't promptly provide the requested connection, an unsafe timing condition could occur.

The communication between MNs and a BS is based on the RSVP protocol [13], which allows resources reservation in advance, namely before a MN effectively uses them (passive reservation).

A MN can be connected only to one BS at once. To establish a connection, the MN requests the resource to the BS, through an active reservation. When the use of the resource is authorized by the BS, the active reservation becomes an active communication. Even if a MN is in an active communication with a BS, the RSVP protocol allows the MN to request resources in advance to a new BS for a future communication. Before interrupting the connection with the current BS, the MN, during its movement, can request to an adjacent BS to turn the passive reservation into an active one (figure 1).

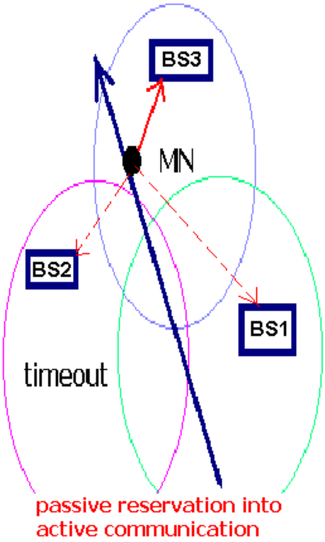


Fig. 1. Turning passive reservations into an active communication

The new BS, having already reserved the necessary resources for the MN during the passive reservation, will finalize the switching. Once the MN receives the acknowledge message, it initiates the connection with the new BS, interrupting the communication with the old BS. This mechanism allows the MN movement in the PMN range, without causing communication quality deterioration.

A BS broadcasts a *beacon signal* to contact a MN crossing the BS cell. The signal carries BS information, such as its IP address. The rate of beacon signal is correlated to the current load and the available resources of the BS. A MN can receive different beacon signals from different BSs. The MN recognizes which is the BS to request the active reservation, on the basis of the BS current load, expressed by the rate of its beacon signal.

Figure 2 shows the timing sequence of the Local Handoff Protocol messages, which carry out the handoff procedure. The initial message is the *Beacon* signal, which is broadcasted by a BS. Once such a signal is received by a MN, the MN Control Layer starts the handoff procedure with the BS. The MN sends an *Announce* message to the BS to request the PMN connection. The *Announce* message contains information on the type of service required to the BS (i.e. real-time or best-effort), the IP address of the old BS.

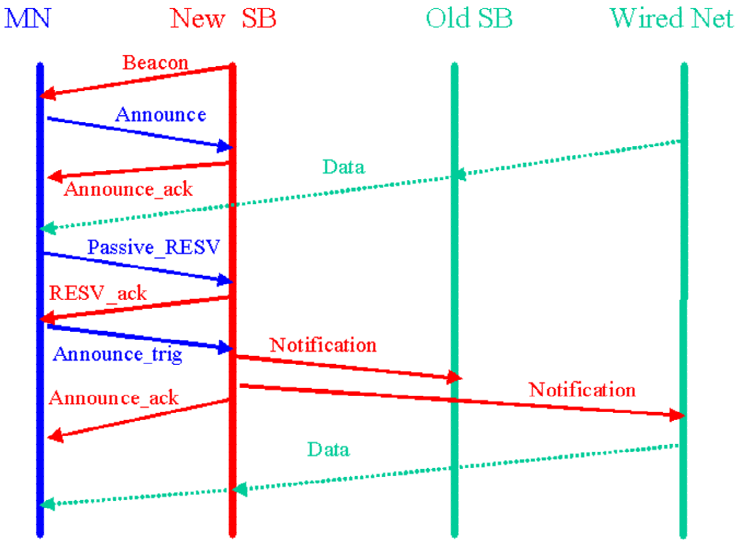


Fig. 2. Timing sequence of Local Handoff Protocol

The BS confirms the reception of the *Announce* message by an *Announce_ack* message. The MN sends a passive reservation request by a *Passive_RESV* message, in which the required resources are listed. The BS acknowledges these messages by sending a *RESV_ack* message, including the timeout reservation value.

At this point the MN can decide whether to confirm the passive reservation by sending back the *Passive_RESV* message within the timeout, or to request the use of the resource, by sending an announce message marked as triggered, *Announce_trig,* to ask for the switching of the passive reservation into an active one. After receiving the triggered message, the BS turns the passive reservation into an active one, creates a routing table entry for the new MN, transmits a *Notify* message to the old BS and sends a signal to the higher hierarchical network level to notify the current correspondence between MN and the new BS. After receiving the *Notify* message, the old BS removes the MN from the routing table entry.

If the request of real-time communication arrives from a MN that has to start up a new connection to the network, once received the *Announce_ack* message from the BS, the MN can decide whether to require an active reservation (*Announce_trig* message) or to require a passive reservation (*Passive_RESV* message). No passive reservation is allowed in case of a best-effort request of service.

The BS bandwidth determines the maximum number of MNs that a BS is able to serve, at the same time. Real-time and best-effort services are provided by the BS. Real-time service has higher priority than the best-effort service. In real-time service, higher priority is assigned to the requests originated by a MN, already connected to the PMN and moving from the current BS to a new BS (continuity of service), respect to the requests of an initial PMN connection.

To accommodate both real time and best effort services, The BS bandwidth is divided in three classes:

- *Continuous reservation band*, assigned to active and passive reservations of MNs, already connected with another BS;
- *Initial reservation band*, assigned to active reservations of MNs, which are requesting an initial PMN access;
- *No reservation band*, dedicated to best-effort service, without passive reservation.

A reservation timeout management is performed at administrative domain level. The more busy the continuous band the shorter the reservation timeout. If the continuous band is not so busy, the BS will be in the position of affording a longer reservation timeout.

3 Assumptions for the PMN Model

In this section we describe the main assumptions of our PMN model, in terms of initial conditions of the PMN, values of PMN parameters and undesired events to consider for the automatic verification of the *waiting access time*. The initial conditions of PMN, in terms of configuration and parameters are varied, by non-deterministic choices, in order to generate the worst case scenarios for the waiting access time of each MN. The *waiting access time* is the interval of time between the instant of time in which MN asks to BS for a communication channel and the instant of time in which MN gets such communication channel.

Preassigning a threshold value for the waiting access time, the automatic verification of the PMN timeliness consists in verifying that for all MN the waiting time is below such a threshold for all network configurations and parameters.

The PMN parameters and their initial values are:

- Number of channels reserved to the initial communication: *10 channels*;
- Number of channels reserved to the continuous communication: *40 channels*;
- MN arrival time in BS cell: *1 MN every 5 sec*;
- Range of beacon signal transmission rate: *1- 10 signals x sec*, by steps of *1 sec*;
- Values of timeout reservation time: *8/15 sec*, depending on the current number of available channels of continuous band;
- Duration of active communication: 15 secs.

Just real-time service is considered, distinguishing between initial communication and continuous communication. A MN can transmit the access request to the BS, after the reception of the beacon signal from BS. Both the beacon signal rate and the reservation timeout are defined according to the available channels. The higher the number of available channels the higher the beacon signal rate and the duration of the timeout. Continuous communication implies a passive reservation of the communication channel which has a maximum duration. Once the communication between

MN and BS starts, it lasts a fixed time. MNs arrive into the BS cell with a fixed timing.

To perform the automatic verification of the PMN timeliness, the following *non deterministic choices* have been implemented into the model:

- *A deviation of the fixed timing of arrival of MNs* (+1, +0 or-1 secs)
- *The request of service that a MN makes to the BS* (active communication or passive reservation)
- *Turning a passive reservation into an active communication*.

The *undesired events* taken into account to generate the worst case scenarios for the waiting access time of any MN are:

- *Failed initial communication request*: MN does not obtain the communication channel in the initial band, and the maximum waiting time expires;
- *Unsuccessful passive reservation request*: MN doesn't succeed in getting a channel in the continuous band within the maximum waiting time;
- *Out of range*: the waiting time has expired before the node has made its request of service (active communication or passive reservation); this last case can occur when a MN waits too much time before receiving the beacon signal from BS.

4 Murphi Model of PMN

Murphi Model Checker [6,7] is a tool to perform formal verifications of systems modeled as Finite-State concurrent Machines (FSM). The system is defined by using a Pascal like programming language. Essentially a Murphi program consists of: *Constant declarations, Type declarations, Variable declarations, Function definitions, Transition Rules, Start State definition, Invariants.*

Figure 3 and 4 show some Constant and Type Declarations.

```
              Const
     num_channels_init : 10;
     num channels cont : 40;
```

Fig. 3. Constant declarations

```
                   type
          arrival_deviation: -1..1;
      type_service :enum{com_init,com_cont};
  trasf_pass_res : enum{release,trasf_active};
```

Fig. 4. Type declarations

In figure 3, the model is initialized with 50 communication channels of which 40 are reserved to continuous communication and 10 are reserved to initial communication. In figure 4, a deviation of *+1, 0 or -1 secs* from the fixed timing of arrival of MN is used; the request of service (*type_service*) that a MN asks to the BS is established as initial or continuous communication; a passive reservation can turn into an active communication or release the channel (*trasf_pass_res*).

The Base Station and the Mobile Nodes are the main elements of the PMN model and their implementation will be described in detail in the following sections 4.1 and 4.2. The communication between MN and BS is implemented by the *Communication* function, as sketched in figure 5. The *Communication* function receives messages from a MN, according to its state, as current PMN parameters and gives back the appropriate answering messages coming from BS.

```
type_messagge_node: enum{nm_null, nm_announce,
nm_announce_trig_init, nm_announce_trig_cont,
nm_rsvp_resv};
type_messagge_BS: enum {sb_null, ack, ack_trig, conf_resv,
no_ack};
function communication
    (messagge_node: type_messagge_node;
     init_band: type_init_band;
     cont_band: type_cont_band): type_messagge_BS;
  begin
     switch messagge_node
      case nm_announce : return(ack);
      case nm_announce_trig_init :
                          if init_band >0
                              then return(ack_trig);
                              else return(no_ack);
                          endif;
      case nm_rsvp_resv : if cont_band >0
                             then return(conf_resv);
                             else return(no_ack);
                          endif;
      case nm_announce_trig_cont : return(ack_trig);
      else return(sb_null);
     endswitch;
   end;
```

Fig. 5. Communication Function

Another main function is the *Timeout* function which implements the management of timeout reservation time, figure 6. When a MN obtains a passive reservation, such a function verifies the available number of channels reserved to the continuous band and sends the timeout value to MN.

```
Function timeout
        (cont_band: type_cont_band) :service_time;
  begin
       -- max tempo di timeout 15 secondi
       -- min 8
     if (cont_band <=num_channels_cont)&
        (cont_band>= num_channels_cont/2)
        then return(15);
        else return(8);
     endif;
   end;
```

Fig. 6. Timeout function

The PMN model evolution is granted by transition rules (Figure 7). The rule *startstate* defines the initial state of the PMN from which the evolution of the model starts. We set BS with the whole available resource and all MN in the *absent* state.

The rule *next_state* triggers transitions among model states;it calls the procedure *next_state* that allows to:

1. Insert a new MN at MN inter arrival time;
2. Update the *beacon* variable;
3. Analyze each MN which is in a state different from *absent* and call the procedure *state_evolution*, which updates the state of each MN.
4. Update both *clock_ds* and *arrival_time* variables.

The rule *next_state* is preceded by 3 rulesets: *arrival_deviation*, *request_trasf, service*. A ruleset allows a rule implementation for each value of the variable which is defined in the ruleset. In our model one ruleset has been used to implement each non-deterministic choice.

The use of the rulesets allows, during the verification phase, to take into consideration the model's evolution in all its possible combinations of nondeteministic choices.

```
Ruleset arrival_deviation:time_arrival_deviation do
ruleset request_trasf: trasf_pass_res do
ruleset service: type_service do
  rule "next_state"
       true
       ==>
       next_state (S,N,clock_ds,
                  arrival_time,arrival_deviation,
                  request_trasf,service);
       end;
end;
end;
end;
```

Fig. 7. Transition rules

The properties to be preserved in the model are described in terms of invariants (figure 8), i.e. properties that must hold on any reachable state. In our PMN model we have the following invariants:

- No MN has to reach the state *Com_init_failed*;
- No MN has to reach the state of *Pass_res_failed*;
- No MN has to reach the *Out_of_range* state;

```
invariant
   controllo_out_of_range (N)=false;
invariant
access_init_failed(N)=false;
invariant
  passive_reservation_failed(N)=false;
```

Fig. 8. Invariants

Such controls are accomplished by three functions, which receive the values of the parameters of each MN, and verify if there is an MN which is in an undesired state.

Once the model is written in the Murphi language, it is compiled by the Murphi compiler that produces the program for verification (verifier). The execution of the verifier allows an exhaustive analysis of the state space of the model under analysis in order to verify system properties, such as error assertions, invariants and deadlocks.

The analysis of the state space can be performed by using an algorithm of breadth-first search procedure, or by a depth-first search. There is the possibility to reduce the number of bit used to represent the state space with reduction techniques (symmetry, multiset, hash compaction).

Each transition of the PMN model occurs in a unit of time (step). We assume that a step time lasts 0.1 seconds and that transmission and the processing of messages between a BS and MNs takes a negligible amount of time. The beacon signal transmission rate is given in deciseconds. The permanence of a MN into a BS range is given in seconds and tens of second. In a step time, MN can send a message to BS and receive an answer.

4.1 Model of the Base Station

The current status of a BS have been described by the number of channels reserved to the initial communication (*type_init_band*), the number of channels reserved to the continuos communication (*type_cont_band*) and the current value of the beacon signal transmission rate (*type_beacon*).

A record has been used to model a BS. Figure 9 shows the type declaration as well as the declaration of the variable *BS* (*BS: type_station;*). The fields of the record *init_band* and *cont_band* represent the number of channels of a BS reserved to initial and continuous communication. Note that in our model we only need one variable of type BS since our modelling is BS centric.

```
type_station: record
                   init_band: type_init_band;
                   cont_band: type_cont_band;
                   beacon: type_beacon;
                 end;
BS: type_station;
```

Fig. 9. Base Station declarations

When an MN obtains a channel, the record field *init_band* is lowered by one, while, if an MN finishes a communication, it is increased by one. The same mechanism is applied for the record field *cont_band.*

The record field *beacon* represents the beacon signal transmission rate of BS. It assumes values included between *0* and *10.* A beacon value of *0* represents the action of the transmission of the beacon signal from BS to MN; only in this case a MN, that is looking for the station, is able to pass in the state announce. In the following transition the beacon is set to some value (greater than 0). The beacon value is then decreased of one unit at each transition, until it reaches to the value 0. The choice of the value to assign to beacon when it reaches the value 0, is performed by a function that considers the current available resource (initial band + continuous band) of BS. Varying such a value, the rate of dispatching of the beacon signal ranges from Fmin (1 per second) to Fmax (10 per second).

4.2 Model of the Mobile Nodes

Each MN entering the range of BS is characterized by its own state. Figure 10 describes the evolution of MN states according to the local handoff protocol reported in section 2.

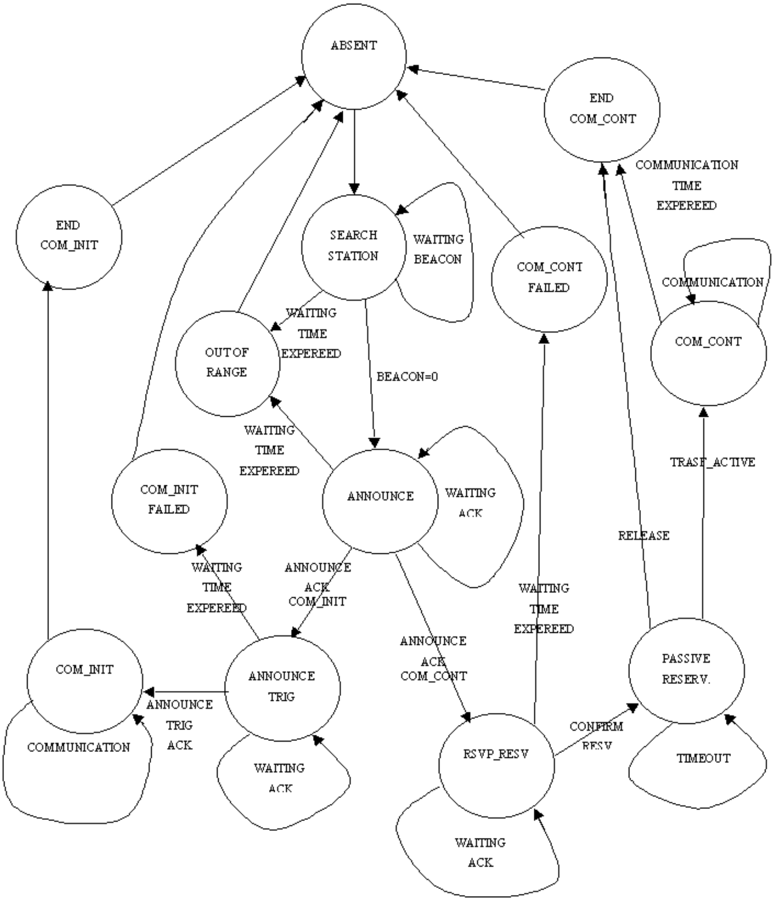


Fig. 10. Evolution of MN states

Each MN state is defined as following:

- *absent***:** MN is not in the range of BS,
- *station_research*: MN is waiting for the beacon;
- *announce:* MN has sent an announce message to BS and it is waiting for the announce_ack;
- *announce_trig*: MN has asked for an active communication and it is waiting for BS acknowledge;
- *rsvp_resv*: MN has asked for a passive reservation and it is waiting for BS acknowledge;
- *init_communication*: MN is in active communication, in the initial band;

- *cont_communication*: MN is in active communication, in the continuous band;
- *passive_reservation*: MN is in passive reservation;
- *out_of_range*: the threshold waiting time has expired while MN was in announce or station_research state;
- *end_communication*: MN has concluded an active communication;
- *com_init_failed*: the threshold waiting time has expired while MN was in announce_trig state;
- *pass_res_failed*: the threshold waiting time has expired while MN was in the *rsvp_resv* state.

A MN is modeled by a record as shown in figure 11. The record field *state* gives indication of the current MN state among the ones reported in figure 10. The record field *max_time* is a counter needed to count how many seconds have elapsed since the occurrence of some specific event.

```
type_node: record
              state: node_state;
              max_time: permanence_time;
            end;
```

Fig. 11. MN declaration

5 Experimental Results

The timeliness verification of our PMN consisted in formally verifying that the waiting access time of each MN is always below a preassigned threshold. Such verification has been performed for different PMN configurations and parameters, starting from an *initial PMN configuration* and parameter values. Murphi found no errors in the *initial PMN configuration*. This means that no reachable state violates the properties to be preserved along the model evolution (invariants).

Table 1 gives some information on running Cached Murphi (with hash compaction enabled) on our Murphi model for the *initial PMN configuration* on a SUN machine.

MN initial connection and MN continuous connection to PMN have been investigated.

Table 2 reports some numerical results, for MN initial connection, obtained by varying the inter arrival time of MN in the BS cell from *5 ± 1 [sec]* to 3 ± 1 *[sec],* through non deterministic choices, and progressively decreasing of one unit the number of channels of the initial band from *10* to *5*. Particularly, fixed the threshold for *waiting access time to 1 sec,* the property under investigation (*waiting access time < 1 sec*) resulted *true* for each MN*,* for a number of channels of the initial band from *10* to *8*, both for arrival time of *5 ± 1 [sec]* and 3 ± 1 *[sec].*

As far as concerns MN continuous communication to PMN, the main parameters which affect the *waiting access time* are the *arrival time* of the incoming MNs, the number of *channels of the continuous band* and the value of *timeout*.

Table 1. Murphi performances on a SUN machine

Bytes	Reach	Rules	Diam	Mem (MB)	Time (sec)
16	11,143,131	33,429,393	258	157	20153.16

Bytes*: the number of bytes needed to represent each state value;*
Reach*: the number of reachable states*;
Rules*: the number of rules fired during verification*;
Diam*: the diameter of our model transition graph***;**
Mem, Time*: the RAM memory and the amount of time to carry out verification.*

Table 2. MN waiting access time in the initial band

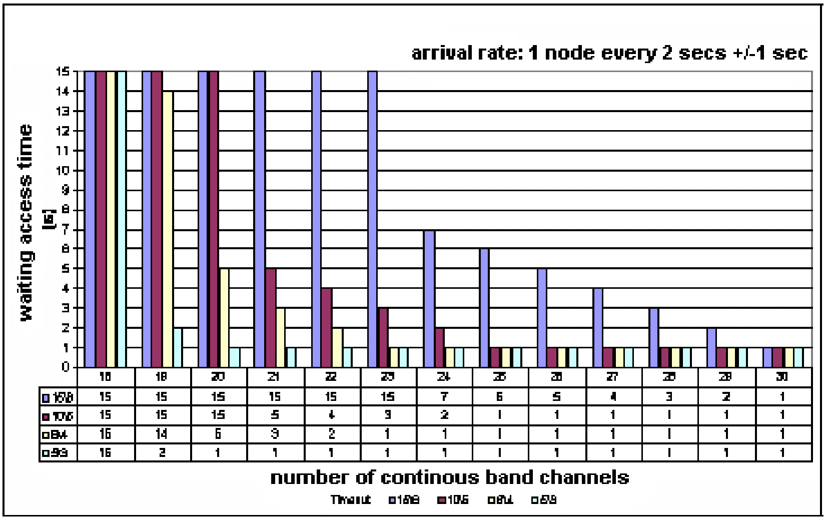
Arrival time (secs)	N. Channels Initial band	MN waiting access Time (secs)
5	10	< 1
3	10	< 1
3	9	< 1
3	8	< 1
3	7	> 10
3	6	> 20
3	5	> 28

Figure 12 shows some numerical results of verification for continuous connection, when the arrival time of the incoming MNs in the BS cell is *1 node every 2 ± 1 [sec].* The number of channels of the continuous band ranges from *18* to *30*. The *timeout* is changed according to the Base Station workload, as implemented by the *Timeout* function. At any request of passive reservation the available channels of the continuous band are computed and one of two different values t_1 and $t_{2,}$ (with $t_1 > t_2$) are dinamically assigned to the t*imeout*. If the number of available channels is greater than an half of the total number of the channels the value t_1 is assigned to the t*imeout.* Otherwise the value t_2 is assigned to the *timeout*. In particular the following sets of values of t_1 and t_2 in seconds, have been used: $t_1 = 15$ *and* $t_2 = 8; t_1 = 10$ *and* $t_2 = 5;$ $t_1 = 8$ *and* $t_2 = 4; t_1 = 5$ *and* $t_2 = 3.$

The results, shown in figure 2, mean that, for a fixed network configuration, at least a case occurs in which a Mobile Node waits the *waiting access time* before obtaining the channel from BS.

6 Discussion and Conclusions

In this paper we investigate how model checking can be used for the automatic verification of the waiting access time of Mobile Nodes (MN) to a Public Mobile Network (PMN), based on a circuit switching connection. The PMN belongs to a real time control system for a critical infrastructure, the Frejus road tunnel. For such a system, time violations in detecting dangerous process conditions and /or in taking corrective control actions could lead both the process under control, the tunnel infrastructure, and the control system to unsafe situations. The presence of a even public PMN, as a part of the system, poses problems of dependability analysis on the frontier of the modeling efforts. That is at least due to a) the novelty and complexity of PMN b) the topology of the network, that dinamically changes for the presence of MNs c) the presence of security aspects that could weaken safety and timeliness properties of the system. Moreover it has to be considered that the current modeling methods are inadequate to deal with the complexity of PMN based real time systems, both in terms of the modeling power of the current tools and the analytical tractability of the resulting models, against the modeling power and the analytical tractability necessary to deal with such systems.



	16	18	20	21	22	23	24	25	26	27	28	29	30
□ 168	15	15	15	15	15	15	7	6	5	4	3	2	1
■ 106	15	15	15	5	4	3	2	1	1	1	1	1	1
□ 84	16	14	6	3	2	1	1	1	1	1	1	1	1
□ 63	16	2	1	1	1	1	1	1	1	1	1	1	1

Fig. 12. Mobile Node waiting access time for continuous connections

Currently modeling and analysis for safety and dependability is actually dominated by two main lines: the functional analysis, whose goal is to ascertain for reachability properties and a stochastic analysis, whose aim is to provide performance and probability measures.

The paper follows the line of reachability analysis. Using the Murphi verifier we generated a finite representation of the PMN (the model of the PMN) and validated its timeliness, considered as a safety critical property, by exhaustively exploring all possible behaviors of the model.

Of course the *formalization* activity (i.e. going from the informal protocol specifications to the formal Murphi model) cannot be formally proved correct. What can be done is, as usual, to use simulation and verification of known properties to validate the formal model. We did this to make sure our Murphi model is indeed a model of our PMN. Our goal here was to check timeliness properties. This guided us in the choice of the invariants to be verified.

Model checkers allow an exhaustive state space exploration in a highly automated way. Exhaustive state space exploration, in turn, allows detection of seldom occurring property violations. In this sense model checking nicely complements stochatic modeling and simulation, which aim at performance evaluation, in term average measures and distributions and can hardly generate worst case scenarious. This is instead exactly what model checking can do: generate a worst case scenario and compute performances for such worst case scenario. Note, on the other hand, that model checking is not suited for an average case analysis.

Acknowledgements. This research was partially supported by EU Project SAFE-TUNNEL and MIUR Project MEFISTO.

References

[1] SAFETUNNEL Project (IST – 2000 – 28099), http://www.crfproject-eu.org/)

[2] D. L. Dill, A. J. Drexler, A. J. Hu, C. H. Yang, *Protocol Verification as a Hardware Design Tool*, In IEEE International Conference on Computer Aided Design, 1992

[3] R. Bryant, *Graph-Based algorithms for Boolean function manipulation*, IEEE Trans. On Computers, C-35 (8), Aug. 1986

[4] A. J. Hu, G. York, D. L. Dill, *New techniques for efficient verification with implicitily conjoined BDDs*. In 31st IEEE Design Automation Conference, 1994.

[5] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, L. J. Hwang, *Symbolic Model Checking: 1020 states and beyond*. Information and Computation, (98), 1992

[6] url: http://sprout.stanford.edu/~dill/murphi.html

[7] url : http://www.dsi.uniroma1.it/~tronci/cached.murphi.html

[8] E. Tronci, G. D. Penna, B. Intrigila, M. Venturini-Zilli, *Exploiting Transition Locality in Automatic Verification*, CHARME, LNCS Springer, Sept. 2001.

[9] url: http://www.eecs.berkeley.edu/~tah/HyTech

[10] A. Bobbio, E. Ciancamerla, G.Franceschinis, R. Gaeta , M. Minichino, L. Portinale - *Methods of increasing modelling power for safety analysis, applied to a turbine digital control system* - SAFECOMP2002, Catania, Italy, September 2002

[11] A. Bobbio, L. Portinale, M. Minichino, E. Ciancamerla - *Improving the Analysis of Dependable Systems by Mapping Fault Trees into Bayesian Networks* - Reliability Engineering and System Safety Journal – 71/3- pp 249–260 - 2001

[12] M. Ajmone Marsan, M.Gribaudo, M.Meo, M. Sereno - *On Petri Net based modelling paradigms for the performance analysis of wireless internet accesses* -Petri Net and Performance Model - PNPM'01 - Aachen - Sept 2001

[13] J.Sokol and J.Widmer -*USAIA Ubiquitous Services Access Internet Architecture* -TR -01-003 International Computer Science Institute, Berkeley 2001