Reliability Analysis of Consensus in Cooperative Transport Systems

Emilia Villani, Negin Fathollahnejad, Risat Pathan, Raul Barbosa, Johan Karlsson

Reliability Analysis of Consensus in Cooperative Transport Systems
(Invited Paper)

Emilia Villani**, Negin Fathollahnejad*, Risat Pathan*, Raul Barbosa***, Johan Karlsson*

* Department of Computer Science and Engineering,
Chalmers University of Technology, Gothenburg, Sweden.
** Instituto Tecnologico de Aeronautica,
ITA, Praca Marechal Eduardo Gomes, Sao Jose dos Campos SP
*** Department of Informatics Engineering,
University of Coimbra, Coimbra, Portugal

Abstract. Our work is concerned with the reliability analysis of a *1-of-n* selection algorithm with the aim of solving the problem of reaching to an *agreement* among n processes in presence of unrestricted communication failures. We show the use of probabilistic model checking as a technique to calculate the probability of reaching to an outcome (e.g., disagreement). We describe how we used PRISM model checker to build and verify a model of a *1-of-n* selection algorithm. We also discuss how the performance of PRISM scales for different number of processes and rounds.

1 Introduction

The use of cooperative transport systems that rely on wireless networks to communicate and take safety-critical decisions is growing in automotive and aeronautics domains. Examples are systems for improving traffic safety and fuel-efficiency of road vehicles, such as vehicle platooning, virtual traffic lights[1] and coordinated lane change [2].

One of the new challenges raised by these systems is the ability of reaching agreement among the cooperating vehicles (processes), in the presence of unrestricted communication failures. Previous works [6, 7] shows that it is impossible to construct an algorithm that guarantees deterministic consensus if we consider no restrictions on the number, timing or pattern of the lost messages.

Based on these important impossibility results, in our previous works, we investigated different practical decision criteria for round-based synchronous consensus protocols that aim at solving the *1-of-n* selection problem, where each process (in a system of n processes) proposes a value and then all processes must either select the same value, or decide to abort. Disagreement occurs if

[1] In a virtual traffic light, road vehicles approaching an intersection interact via wireless communication to form a virtual, or imaginary, traffic light.

some processes decide to select a value, while the remaining nodes decide to abort.

In this paper we focus on the use of the probabilistic model checking tool PRISM [5] to perform a reliability analysis of the consensus algorithm and determine the probability of *disagreement* among cooperating units under different system configurations. We also discuss how the performance of PRISM scales for different number of processes and rounds.

The remainder of the paper is organized as follows. In Section 2 we describe briefly the *1-of-n* selection algorithm. Using a simple system of three processes as an example in Section 3 we explain a PRISM model in detail. In Section 4 we show and discuss some of our probabilistic model checking results. Finally we conclude and give some directions of our future work in Section 5.

2 Protocol Description

Previously we investigated a *1-of-n* selection algorithm with three decision criteria called the *optimistic*, *pessimistic* and the *moderately pessimistic* decision criterion. In this paper our focus is on the *optimistic* decision criterion. For more details on other decision criteria we refer the reader to our previous work in [4] and [3].

We model our system by adopting the classical round-based computational model as in [1] that considers a synchronous system of n processes indexed respectively with their identifiers as: $\Pi = \{p_1, \ldots, p_n\}$. We assume that processes are fully connected to one another via wireless broadcast links and execute the *1-of-n* selection algorithm in R rounds of message exchange. The aim of the protocol is that processes reach to a consensus on a value among n proposed values by n processes. Without the loss of generality we assume that processes are to select the highest value. Each process $p_i \in \Pi$ initially proposes a value and

Algorithm 1 The *1-of-n* selection algorithm run by (p_i)

```
for r = 1 to R do
    begin_round
    send_to_all(msg_i);
    receive_from_all();
    compute(msg_i);
    end_round
end for
if v_i is complete then
    p_i selects the highest proposed value;
else
    abort;
end if
```

constructs a message (msg_i) that contains its proposed value ($proposed_i$) and a

bit-vector (v_i) of length n which represents the view of p_i of the system. Initially, $v_i[j] = 0$ for all $j \neq i$ (p_i has not received any message from other processes), and $v_i[i] = 1$. $v_i[j] = 1$ means that process p_i has received the information from p_j either directly from p_j or by relaying information from other processes. Each process iterates three phases for R rounds including first to send a message to all, second to receive messages from all and third to compute (See Algorithm 2). In the compute phase each process updates its proposed value and view vector according to the received messages. Process p_i updates $proposed_i$ to the highest value among its current value and the received values. It also updates v_i as follows: for all $v_i[k] = 0$, if it receives a message from p_k or from a process as p_j with $v_j[k] = 1$, then p_i updates $v_i[k]$ to 1. After R rounds, each process executes the decision algorithm. For this purpose, we define v_i as *complete* if all elements of this vector are set to 1. If the view of a process p_i is complete by the end of round R, it decides to select the highest proposed value. Indeed a process with a complete view optimistically assumes that all the other processes have also complete views and select a value. A process with an incomplete view by the end of round R decides to abort.

3 Modelling the Consensus Protocol in PRISM

In this section we present a PRISM model of a *1-of-3* selection algorithm as an example of a simple system of three processes subject to unrestricted asymmetric communication failures. We define Q as the probability of losing a message. In order to model the protocol we use discrete time Markov chains (dtmc).

In Fig. 1 we present the conceptual model of a process as p_i. From the initial state (*State* 1), the process sends its message (transition $T1$). In the sequence, $T2$ and $T3$ are probabilistic transitions that model the choice between receiving (with probability $1 - Q$) or losing (with probability Q) the messages from the other two processes. At any round except the last one, the compute phase is performed by transition $T4$. In the last round, $T5$ fires instead of $T4$ and performs both the round computation and the decision making. Then the process moves to the final state (*State* 0). In PRISM each process is defined

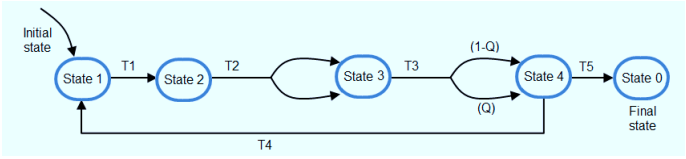

Fig. 1. Conceptual model of a process.

as a module. Following we explain each section of our PRISM model in detail (See Listing. 1.1). The first line of the model declares its class (dtmc). Then, we declare constants and global variable. Two constants define the configurations

of the system under modelling: number of rounds (RN) and the probability of losing a message (Q). vi_ini store the initial value of process p_i. Ni associates an identity number to each process p_i. We use global variables to model the message exchange among processes. Variable vi_ext stores the last value sent by process p_i, while *Boolean* variables wi_vj_ext store the view that process p_i has of the value of process p_j $(i, j \in [1, 3], i \neq j)$. The global variable $round$ stores the current round of the protocol, while $token$ is used to synchronize the modules. It receives its value within the range of [1..3] and indicates the process that must perform the next transition. Following, we define three formulas to be used by process p_1 in the compute phase: $v1\_new$ computes the new proposed value of p_1 by comparing the current value with the values received from other processes, while $w1\_v2\_new$ and $w1\_v3\_new$ compute the view that process p_1 has of p_2 and p_3, respectively.

The description of the module that corresponds to process p_1 is composed of two parts: *declaration of internal variables* and *transitions*. In order to represent the structure of the dtmc given in Fig. 1, we define the internal variable $s1$ (state of process p_1). The current value of p_1 is stored in $v1$, while the *Boolean* variables $w1\_v2$ and $w1\_v3$ stores the current view that p_1 has of p_2 and p_3 values. *Boolean* variables $n1\_nf2$ and $n1\_nf3$ register which messages are received and lost by p_1 at each round. The final decision of process p_1 is stored in the *Boolean* variable $d1$. Internal variables are followed by the description of the transitions $T1$ to $T6$. In PRISM, a transition is composed of a guard and an action ($[]guard- > action$). In the case of probabilistic transitions, the action is composed of a set of possible actions with the corresponding probabilities. Transition $T1$ copies the value and view of p_1 to the global variables. $T2$ and $T3$ defines the messages that are received and lost by p_1 by attributing true or false to $n1\_nf2$ and $n1\_nf3$. Transitions $T4$ and $T5$ implement the compute phase. Additionally, transition $T5$ makes a decision and attributes a value to $d1$: $d1 = true$ means that process p_1 decided on the proposed value, while $d1 = false$ indicates a decision to abort. The synchronization of the modules is achieved using the global variable $token$. A process p_i receives the token when $token = Ni$. It passes it to the next process after one of the following transitions: $T1$, $T4$, $T5$. Finally, processes p_2 and p_3 are defined as a copy of process p_1.

In order to compute the probability that the protocol results in an outcome (agreement, abort or disagreement), we specify properties in Probabilistic Computation Tree Logic ($PCTL$), the specification language used by PRISM. As an example, we present the query for the probability of disagreement, which means what is the probability that eventually (operator F) the system reaches a state where all the processes are in the final state which is and have reached different decisions:

$$P =?[F(s1 = 0)\&(s2 = 0)\&(s3 = 0)\&((d1! = d2)|(d2! = d3)|(d3! = d4))]$$

We introduced some simplifications in the PRISM model to reduce the number of states of the model. The choice of the initial value of each process is not

```
dtmc
// Constants
const RN=2;  const double Q=0.5;
const v1_0=3;  const v2_0=2;  const v3_0=1;
const N1=1;  const N2=2;  const N3=3;

// Global variables
global v1_ext: [0..3] init 0;   global v2_ext: [0..3] init 0;  global v3_ext: [0..3] init 0;
global w1_v2_ext: bool init false;  global w1_v3_ext: bool init false;
global w2_v1_ext: bool init false;  global w2_v3_ext: bool init false;
global w3_v1_ext: bool init false;  global w3_v2_ext: bool init false;
global token: [1..3] init 1;  global round: [1..RN] init 1;

// Formulas
formula v1_new = max(v1,(n1_nf2?v2_ext:0),(n1_nf3?v3_ext:0));
formula w1_v2_new = w1_v2 | n1_nf2 | (n1_nf3 & w3_v2_ext);
formula w1_v3_new = w1_v3 | n1_nf3 | (n1_nf2 & w2_v3_ext);

module Process_1
// Internal variables
s1: [0..4] init 1;  v1: [0..3] init v1_0;
w1_v2 : bool init false;  w1_v3 : bool init false;
n1_nf2 : bool init true;  n1_nf3 : bool init true;
d1 : bool init false;

// Transitions
[] s1=1 & token=N1-> 1:(s1'=2) & (v1_ext'=v1) & (w1_v2_ext'=w1_v2) & (w1_v3_ext'=w1_v3) & (token'=N2);// T1
[] s1=2 & token=N1-> (1-Q): (s1'=3) & (n1_nf2'=true) + Q: (s1'=3) & (n1_nf2'=false); // T2
[] s1=3 & token=N1-> (1-Q): (s1'=4) & (n1_nf3'=true) + Q: (s1'=4) & (n1_nf3'=false); // T3
[] s1=4 & token=N1 & round<RN-> 1: (s1'=1) & (v1'=v1_new) & (w1_v2'=w1_v2_new) & (w1_v3'=w1_v3_new) &
          (n1_nf2'=false) & (n1_nf3'=false) & (round'=round+((N1=3)?1:0)) & (token'=N2); // T4
[] s1=4 & token=N1 & round=RN-> 1: (s1'=0) & (v1'=v1_new) & (w1_v2'=w1_v2_new) & (w1_v3'=w1_v3_new)
          & (d1'= w1_v2_new & w1_v3_new) & (token'=N2); // T5
endmodule

module Process_2=Process_1 [s1=s2, v1=v2, w1_v2=w2_v3, w1_v3=w2_v1, n1_nf2=n2_nf3, n1_nf3=n2_nf1, d1=d2,
                v1_ini=v2_ini, N1=N2, N2=N3, N3=N1, v1_ext=v2_ext, v2_ext=v3_ext, v3_ext=v1_ext,
                w1_v2_ext=w2_v3_ext, w1_v3_ext=w2_v1_ext, w2_v1_ext=w3_v2_ext, w2_v3_ext=w3_v1_ext,
                w3_v1_ext=w1_v2_ext, w3_v2_ext=w1_v3_ext] endmodule
module Process_3=Process_1 [s1=s3, v1=v3, w1_v2=w3_v1, w1_v3=w3_v2, n1_nf2=n3_nf1, n1_nf3=n3_nf2, d1=d3,
                v1_ini=v3_ini, N1=N3, N2=N1, N3=N2, v1_ext=v3_ext, v2_ext=v1_ext, v3_ext=v2_ext,
                w1_v2_ext=w3_v1_ext, w1_v3_ext=w3_v2_ext, w2_v1_ext=w1_v3_ext, w2_v3_ext=w1_v2_ext,
                w3_v1_ext=w2_v3_ext, w3_v2_ext=w2_v1_ext] endmodule
```

Listing 1.1. PRISM model, *1-of-3* selection algorithm with optimistic decision criterion

entirely non-deterministic. The initial values should be specified by the user. This simplification is adopted because a process only makes a decision with a complete view, which means that, no matter the number of lost messages, a process never decides on a wrong value and the initial values have no impact on the protocol's outcome (agreement, abort and disagreement). Furthermore, the processes execute the protocol in a predefined sequence (p_1, p_2 and p_3), not in a random order. Again, this simplification is adopted because the order does not affect the results, a process only updates its value and view at the end of the sending and receiving phases of the consensus protocol.

Another important observation about the PRISM model is that although we can easily increase the number of rounds or change the probability of message loss, we cannot use the same model for systems with different number of processes because, in a PRISM model, all variables must individually be named. In order to increase the number of process, we have to: add the corresponding constants and variables; add new transitions to the module of process p_1; update formulas, transitions, and definitions of existing modules; and add the definition of the modules for the new processes.

4 Results and Discussion

In this section we present some results showing the probability of disagreement calculated by PRISM. We show how the number of states produced by the PRISM model varies for different configurations of a system.

Fig. 2 presents the probability of disagreement as a function of the probability of message loss (Q), assuming $R = 2, 3, 4$ and $n = 3, 4, 5$. As we can see, with a fixed number of processes, when we increase the number of rounds, the peak of the probability of disagreement moves to the right side of the x-axis. However, it does not show a significant reduction in the maximum value of the probability of disagreement. On the other hand, when we increase the number of processes with fixed R, the maximum value of the probability of disagreement significantly increases, reaching values as high as 0.8 for $n = 5$. For a more detailed discussion and analyses of the behaviour of the *1-of-n* selection algorithm we refer the reader to our previous works [4] and [3].

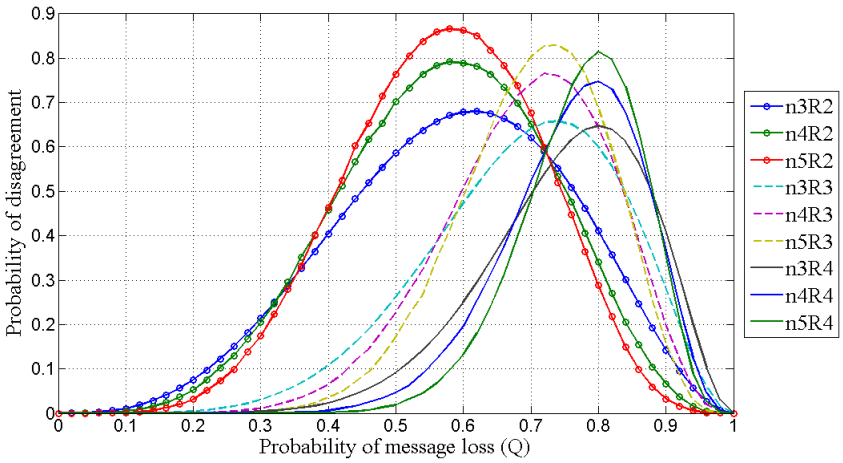

Fig. 2. Probability of disagreement for varying n and R.

The results in Fig. 2 are obtained using probabilistic simulation for $n = 4, 5$. This is because with increasing the number of processes ($n > 3$), the number of reachable states of the model increases considerably which makes the verification process infeasible. This concerns with one of the most important challenging problems in model checking, the state space explosion. This problem arises when it is computationally too expensive for a model checker as PRISM to examine the entire reachable state-space. In Table 1 we show the number of states of the PRISM model with $n = 3$ and $n = 4$. We see that with increasing R the number of states grows linearly with R. However with increasing R, the

Table 1. Number of states of PRISM models.

R	$n = 3$	$n = 4$
2	13846	3268746
3	18053	10717177
4	22260	18666673
5	26467	26616169
6	30674	34565665

number of different combinations of lost and sent messages for a system of n processes, which is $2^{n\cdot(n-1)\cdot R}$, increases exponentially. The reason is that state space is limited by the possible values that the variables can assume and different scenarios converge to common states.

5 Conclusion and Future Work

In this paper we are concerned with the problem of reaching agreement among cooperating units of a safety-critical system where the underlying communication links among processes are unreliable. We showed using a simple example how we used a probabilistic model checking tool to analyse the reliability of a consensus protocol with unrestricted communication failures. We present how to build and verify a model of a *1-of-n* selection algorithm with optimistic decision criterion using PRISM. We also show some analyses of the problem of state space explosion when we vary the system parameters. Due to the limitations imposed by this problem, as future work we will try to find the close-form expressions that can be used to configure the protocol at run time, according to the quality of communication channels.

Acknowledgements

This work was partially supported by the Swedish National Automotive Research and Innovation Programme (FFI) and VINNOVA, through the DFEA2020 project, and the European Commission, through project FP7-STREP-288195, KARYON (Kernel-based ARchitecture for safetY-critical cONtrol). Emilia Villani was supported by the Brazilian National Council for Scientific and Technological Development (CNPq), Swedish-Brazilian Research and Innovation Centre (CISB) and SAAB AB, under grant proc. 236722/2012-3.

References

1. B. Charron-Bost and A. Schiper. The heard-of model: computing in distributed systems with benign faults. *Distributed Computing*, 22:49–71, 2009.
2. E. Coelingh and S. Solyom. All aboard the robotic road train. *Spectrum, IEEE*, 49(11):34 –39, november 2012.

3. N. Fathollahnejad, R. Pathan, E. Villani, R. Barbosa, and J. Karlsson. Probabilistic analysis of disagreement in synchronous consensus protocols. *Technical Report, Chalmers University of Technology, Sweden*, April 2013. `http://www.cse.chalmers.se/~johan/publications/TR2013.pdf`.
4. N. Fathollahnejad, E. Villani, R. Pathan, R. Barbosa, and J. Karlsson. On reliability analysis of leader election protocols for virtual traffic lights. In *in conjunction with the 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks(DSN)*, June 2013.
5. M. Kwiatkowska, G. Norman, and D. Parker. Prism: probabilistic model checking for performance and reliability analysis. *SIGMETRICS Perform. Eval. Rev.*, 36(4):40–45, Mar. 2009.
6. N. Santoro and P. Widmayer. Time is not a healer. In B. Monien and R. Cori, editors, *STACS 89*, volume 349 of *Lecture Notes in Computer Science*, pages 304–313. Springer Berlin Heidelberg, 1989.
7. U. Schmid. How to model link failures: A perception-based fault model. In *IEEE International Conference on Dependable Systems and Networks (DSN)*, pages 57 – 66, July 2001.