Use of probabilities and formal methods to control system criticality levels

Jasdeep Singh\textsuperscript{1}, Zhishan Guo\textsuperscript{2}, Luca Santinelli\textsuperscript{1}, Guillaume Infantes\textsuperscript{1}, David Doose\textsuperscript{1} and Julien Brunel\textsuperscript{1}
\textsuperscript{1}ONERA - DTIS Toulouse, name.surname@onera.fr
\textsuperscript{2} Missouri University of Science and Technology Rolla MO, guozh@mst.edu

\textbf{Introduction and motivations.} The gap between the actual execution and the worst-case bounds may be significantly large with current real-time systems. Instead of completely wasting the processor capacities within the gap, latest trends start to implement functionalities of different degrees of importance, or criticality, upon a common platform. This allows less important tasks to execute in these gaps under normal circumstances, and may be dropped in an occasional situation where jobs of higher importance execute beyond their estimated running time \cite{1, 2}.

A Mixed Criticality (MC) real-time system is one that has two or more distinct criticality levels e.g., safety critical jobs (HI-criticality), mission critical or low critical jobs (LO-criticality). Such systems are defined to execute in a number of criticality modes, each mode specifying execution conditions and system criticality. All the possible modes have to be characterized and analyzed in order to guarantee the predictability of the system. See \cite{3} for an overview of the MC problems.

As drawn by \cite{3}, MC problems can be approached with probabilities to quantify and manage the unlikely events and reduce the pessimism. We believe that a tighter coupling between MC problems and probabilistic frameworks can end up into smarter scheduling decisions and more efficient utilization of the computational resource.

With this abstraction, we propose to model and investigate critical mode behaviors of MC systems by using probabilities and formal methods. The goals are: 1) quantifying the probability of jobs and system entering HI-criticality mode, and 2) controlling this probability by selectively removing LO-criticality jobs from the schedule. Formal methods and probabilities are used to model jobs and system HI-criticality modes. The probabilities are actively applied into scheduling decisions.

\textbf{Background - probabilistic computational models:} A real-time application is a set $\Gamma = \{J_{ij}\}$, $J_{ij}$ is the $j$-th job of $i$-th task. A probabilistic real-time application is called so whenever at least one of its elements is described with a probabilistic parameter; in this case it is the probabilistic Worst-Case Execution Time (pWCET) for task execution, but it can be extended to any model parameter. The pWCET is the worst case distribution which is able to upper bound any job execution behavior \cite{4}. Hereby, we assume the pWCETs to be continuous distributions, but the same conclusions can be drawn with discrete distributions.

The pWCET as a continuous random variable $X$ is represented with the Probability Density Function (PDF) $f_X(x)$, the Cumulative Distribution Function (CDF) $F_X(x) = \int_0^x f_X(y)dy$, and the Inverse Cumulative Distribution Function (ICDF) $F_X^{-1}(x) = 1 - \int_0^x f_X(y)dy$ which gives the exceeding threshold probability.

The job model $J_{ij} = \{C_{ij}, a_{ij}, d_{ij}, p_{ij}\}$ is such that $J_{ij}$ arrives at time $a_{ij}$, $d_{ij}$ is the job absolute deadline, $p_{ij}$ is the job priority, and $C_{ij}$ is the job pWCET. Jobs can be seen as task instances in a periodic or sporadic task application. A task $\tau_i$ is the tuple $\tau_i = \{C_{i}, T_i, D_i\}$ where $C_{i}$ is the pWCET ($C_{ij} = C_{i}, \forall j$), $T_i$ is the period, and $D_i$ is the deadline of the task ($D_i \leq T_i$).

The scheduling policy defines the job ordering by imposing job-wise priorities $p_{ij}$. Both job static priority or dynamic priority schemes can be used here. The hyperperiod is defined as the least common factor lcm() of all the task periods, $lcm(T_i)$, $i = 1, 2, \ldots, m$. It is the scope of the schedulability analysis and the criticality analysis we propose, since we assume the job execution suspended at the job deadline. In the hyperperiod there are $n$ jobs from $m$ tasks.

In a probabilistic framework, for each job there exists a probabilistic Worst-Case Response Time which is a distribution to represent the worst-case response time of the job. pWCRT is the result of probabilistic schedulability analysis. Same as for the pWCET, the pWCRT is assumed to be a continuous random variable and can be represented with PDF, CDF and ICDF. Discrete pWCRTs are applicable without modifying the proposed reasoning. To compute pWCRT, it is possible to apply any of the existing probabilistic schedulability analysis approaches e.g., \cite{6, 7} for discrete distributions, and \cite{8} for continuous distributions.

\textbf{Background - Markov decision process:} Markov decision processes (MDPs) are mathematical frameworks for modeling decision making in situations where the outcomes are partly random and partly under the control of a decision maker \cite{6}. More precisely, a MDP decision process is a discrete time stochastic control process where at each time step, the process is in some state $s$, and the decision maker may choose any action $a$ that is available in $s$. The probability that the process moves into its new state $s'$ is influenced by the chosen action. Specifically, it is given by the probability state transition function $P_{a}(s, s')$. A Markov decision process $M$ is defined as a set of states $S_{MDP}$ and state transitions given by a Q-matrix $Q_{MDP}$. Forman model checking can be done on MDP in order to know the probability of reaching certain states (property to formally verify) by taking certain paths. Figure 1 shows an example of MDP with states and state transitions weighted over the probability $p$ for the transition of happening. We propose to use MDP to model jobs and system criticality and make use of its mathematical foundations.

\textbf{Mixed criticality.} At first instance, we consider the two-criticality level case for mixed criticality probabilistic real-time applications. In it, each job is designated as being of either higher criticality HI-criticality or lower criticality LO-criticality. Two different behaviors (modes) are specified for each HI-criticality job: a high mode, where the job executes in highly critical and more demanding conditions; a low mode which is the nominal working condition for the job where it executes in normal conditions. A LO-criticality job has only low mode.

A HI-criticality job $J_{ij}^{HI}$ is $d_{ij}^{HI} = \{C_{ij}, a_{ij}, d_{ij}, I_{ij}, \chi_{ij}\}$, where $C_{ij}$ is the job pWCET, $I_{ij}$ is the arrival instant, $d_{ij}$ is the deadline of the job, and $\chi_{ij}$ is the job criticality level \cite{1}. $\chi_{ij}$ can take two values at runtime: HI and LO; $\chi_{ij} \in \{\text{HI, LO}\}$. $I_{ij}$ describes the threshold with which we define the job criticality mode. A LO-criticality job $J_{ij}^{LO}$ is $J_{ij}^{LO} = \{C_{k}, a_{kr}, d_{kr}, \chi_{kr}\}$. $\chi_{kr}$ for a LO-criticality job can take only one value, LO, $\chi_{kr} \in \{\text{LO}\}$.

The criticality mode of the jobs can change at runtime depending on their scheduling. The threshold $l_{ij}$ applies to the job WCET and defines the HI-criticality job criticality mode – response time threshold. It is such that if the job finishing time is beyond the threshold, the job is considered to execute in HI-criticality mode. Otherwise, the job executes in LO-criticality mode. Figure 2 illustrates that with HI and LO criticality regions which are defined such that job $J_{ij}$ is in high criticality mode if it finishes executing in the interval $[l_{ij}, \infty)$. It is in LO-criticality mode if it finishes executing in the interval $[0, l_{ij})$. The HI-criticality mode there is an associated probability $p_{ij}$ that is the probability that the execution of a job $J_{ij}$ exceeds $l_{ij}$ (the exceeding probability for $I_{ij}$ as the probability for the job of entering
HI-criticality mode); $P_{k}^\text{LO} = 1 - P_{k}^\text{HI}$ is the probability that $J_{ij}$ remains in LO-criticality mode. It is also possible to have the threshold $l_{ij}$ applying to the job pWCET – execution time threshold, for a more classical MC modeling of the jobs criticality levels. We choose to apply pWCRT to characterize the job criticality level because this way all the interference effects are included and accounted for when deciding which conditions trigger a HI-criticality mode for the task.

For MDP modeling the MC behavior of the system, the HI-criticality jobs are represented with two states: HI-criticality and LO-criticality, and the state transitions is the probability of mode transitions $P^\text{HI}$ and $P^\text{LO}$; Figure 1 as an example of MDP for two-critical probabilistic real-time application. In there, HI-criticality states (HC) and LO-criticality states (LC) are represented for each job together with the probability for $J_{ij}$ of being in one state or the other $P_{ij}$ and $1 - P_{ij}$. From the job criticality level, the system criticality level can be defined as: the system criticality level $\chi$ is equal to HI if at least $k$ out of $n$ HI-criticality jobs enter high criticality mode. This is a generic definition of system criticality level for $1 \leq k \leq n$, which extend to more conservative definitions where as soon as one HI-criticality job moves to HI-criticality mode, the system moves to HI-criticality mode.

**Open problem.** The problem we intend to tackle concerns to the investigation of the system criticality at runtime with the help of probabilities and formal methods. We can state the problem as: the probability that a mixed criticality real-time application enters high criticality mode should be less than or equal to certain probability $P_{\text{sys}}^\max$. $P_{\text{sys}}^\max$ is a requirement given and the analysis we propose is such that the resulting MC scheduling will allow to meet the requirement.

This is an open problem in the sense that there are no solutions to that. We also believe it is an interesting one because it represents a way to apply probabilities for characterizing the system criticality behavior as well as use them into scheduling decisions. To approach it, we would like to define what we call criticality analysis.

From $\Gamma$ and the pWCRTs computed with probabilistic timing analysis, we identify the criticality modes of each jobs (with the threshold $l$ applied to pWCRTs) and the probability $P^\text{HI}$ associated. $P^\text{HI}$ is the probability used to label the MDP in order to model criticality mode transitions. We can assume that the MDP is made from the HI-criticality jobs only, since the LO-criticality jobs would have only one state associated and will not contribute to the analysis.

Every combination of job modes are accounted for and encoded into the MDP as a path between states. Path analysis and model checking can be carried out for the MDP in order to compute the probability of the system criticality mode from the generic definition $(k, n)$, and any interpretation e.g., $k = 1, n = n$.

1) At first, we propose to **quantify** the probability of HI-criticality mode $P$ for the system. This is done from the MDP by considering the probability of each path corresponding to the $(k, n)$ system criticality definition: an exact probabilistic representation of job modes and system mode which could happen at runtime (probability of occurrence) is possible. 2) Then, with $P \leq P_{\text{sys}}^\max$ we act on the system scheduling in order to meet such constraint and reduce the system HI-criticality. This control can apply to system criticality or to specific set of jobs. Figure 1 illustrates an example of MDP for the MC problem with paths between states and the state transition probabilities to be composed into path probabilities.

The scheduling strategies we intend to develop consist of finding the LO-criticality jobs to drop from schedule in order to reduce $P$. They are meant to select LO-criticality jobs with the most impact on HI-criticality jobs. Those strategies will be conceived as an off-line analysis that minimizes the number of LO-criticality jobs to be removed in order to guarantee the occurrence probability of HI-criticality modes.

As today, we are beginning to formalize the **criticality analysis** defining how to model criticality with MDP and how to formally explore MDPs. There is a MDP implementation already available, that we apply to study the impact that HI-criticality jobs have on the probability HI-criticality mode $P$. The implementation uses a Continuous Time Markov Chain based schedulability analysis for pWCRT computation. This is a follow up for the RTSOPS 2017 abstract ‘Markov Chain Modeling of Probabilistic Real-Time Systems’. We note that the MDP modeling can interface with any probabilistic schedulability analysis, with or without formal methods, and for discrete or continuous distributions. The use of MDP can benefit form more-than-two criticality levels. Furthermore, by utilizing the notion of rewards in the MDP, it is possible to quantify the cost of a job entering HI-criticality mode. In future work, we want proposing to combine MDP with the satisfiability modulo theories for developing optimal or pseudo-optimal scheduling strategies to control system criticality probability.

![Fig. 1. System MDP model assuming $J_{11} , J_{12} , \ldots , J_{mn}$ Hi and Hi the HI-criticality jobs in the hyperperiod for representation.](image)

**Fig. 1.** System MDP model assuming $J_{11} , J_{12} , \ldots , J_{mn}$ Hi and Hi the HI-criticality jobs in the hyperperiod for representation.

![Fig. 2. Response time distribution in its ICDF form of a job $J_{ij}$.](image)

**Fig. 2.** Response time distribution in its ICDF form of a job $J_{ij}$. High and low criticality regions are separated by $l_{ij}$.

**REFERENCES**


