Resource Allocation for a Limited Real-Time Agent
Haksun Li, Ella M. Atkins*, Edmund H. Durfee, Kang G. Shin
EECS Dept., University of Michigan, Ann Arbor MI 48109
*Aerospace Engineering Dept., Univ. of Maryland, College Park MD 20742
{haksunli, durfee, kgshin}@umich.edu, atkins@eng.umd.edu

Categories and Subject Descriptors
I.2.8 [Artificial Intelligence]: Problem Solving, Control Methods, and Search – heuristic methods, plan execution, formation, and generation, scheduling.

General Terms
Algorithms, Performance, Design, Experimentation

Topics/Keywords
Action selection and planning for agents, Agent architectures, Probability, Real-time systems

Introduction
A real-time agent with limited perceptual, computational and actuator resources must carefully allocate these resources at execution time to do the best job it can in monitoring for aspects of the world state and responding quickly to emerging hazards in its complex and dynamic environment. If the resource-limited agent allocates more of its resources (or some of its resources more frequently) to monitoring for some states (or state features), then it will be less capable of tracking others successfully. Similarly, it cannot be assumed to execute all the resource-demanding actions equally well. Designing an effective schedule of monitoring for and responding to the most important situations at the right times with the available resources is thus a complex optimization problem.

Consider this exceedingly simplified driving example: Suppose a driver driving at a high speed has to do three recognition-reaction tasks periodically. (1) Look ahead for up to 3 seconds every 5 seconds to gather information about the road conditions and reduce speed if appropriate; (2) check the side mirrors for up to 2 seconds every 10 seconds and switch lanes if necessary; (3) check the rear-view mirror every 10 seconds for up to 3 seconds, and accelerate if being tailgated. Although the driver could perform each of these tasks separately, he is not able to guarantee hard real-time performance to all these tasks combined. He may miss a deadline because it would require a total of 11 seconds to complete all operations every 10 seconds if each were to take its worst-case time. The utilization of his resources (here attention), is greater than 1! The driver has no choice but to find a resource-satisfying subset of tasks such that his safety is maximized, or equivalently, the failure probability is minimized. For instance, if being hit by a car approaching from behind is sufficiently unlikely, then he may drop the desire to reliably check the rear-view mirror. He can hold this task on an “if-time” queue (Musliner and et. al., 1993), and do it only when the guaranteed tasks of looking ahead and checking the sides take less than their worst-case times. Essentially, he has purposely chosen to make hard real-time guarantees for the reactions that are more probably going to be needed, and to do only “best effort” on other tasks as slack resources permit.

However, determining the likelihood of a real-time agent encountering a particular situation or a state can be challenging because the likelihood is dependent not only on the actions the agent should perform (what to do if the driver is being tailgated), but also on its choices of how frequently it checks whether to perform them (how often the driver looks in the mirrors). By definition, a dynamic environment is one in which the state can change via events outside the agent’s control. In general, the sooner a real-time agent detects and responds to a situation, the less opportunity there is for the environmental dynamics to intervene, and the higher the chance that the agent is going to meet its deadlines. The probability of encountering a state thus depends on a complex temporal convolution between the agent’s plan and the exogenous events.

One contribution of our work is that we have developed a probabilistic action and event model to efficiently approximate the transition probabilities for a real-time agent. A second contribution is our strategy for using these probabilities to prioritize resource expenditures when the agent cannot be prepared for all eventualities.

Real-Time Execution as a Continuous Time Stochastic Process
We schedule run-time execution resource usage based on a model of how the world might transition from state to state. To model stochastic transitions, we give each a probability function describing its likelihood of occurrence over time. Specifically, let $T$ be the random variable denoting the time that a transition fires since it was enabled $f(t)$. $(F(t))$ is the (cumulative) probability density function of $T$. $t$ is the transition time, ranging from 0 to infinity. We also denote $(1 - F)$ by $F$. To model the execution of a real-time agent as a continuous-time stochastic process, we assume that all transitions are mutually independent. After numbering the transitions in a state, we denote the $i$-th transition by $T_i$. We assume in this paper that all the clocks associated with the transitions start counting from the time the agent enters a new state. The transition probability of $T_i$, relative to all other applicable temporal transitions in a state, is the probability that its firing time, $T_i$, is the minimum, $T_i$ among all transition times. Thus, the transition probability is given in Eq. 1.

$$P(T_i = T) = P(T_i = \min[T_{i-1}, ..., T_{i-1}]) = \int F_{T_i}(t) \cdots F_{T_{i-1}}(t) \cdots F_{T_{i-2}}(t) f_{T_i}(t) dt$$ (Eq. 1)

Unfortunately, it is in general very difficult to compute transition probabilities using Eq. 1, because it is an integral of a product of a number of continuous functions from time 0 to
The Unlikely State Strategy

Every semi-Markov process (SMP) has an embedded discrete time Markov chain (DTMC) with the same state space and the same transition probabilities. Moreover, the state probabilities of the states in the DTMC are exactly the same as those in the SMP (Heyman 1982). With transition probabilities computed using the heuristic in the last section, we can compute the state probabilities of the states in a stochastic process of a real-time agent by computing the state probabilities of the states in the corresponding embedded DTMC using standard Markov techniques (Kemeny and Snell 1960).

With the transition and state probability computations, we are ready to describe the resource allocation process. After an agent generates a tentative plan, it is passed to the scheduler, which tries to schedule the actions according to the resource constraints of the execution platform. If all planned actions are schedulable, then it is done. Otherwise, the state probabilities of the states are computed. The actions planned for the least likely states are removed in increasing order of their state probabilities until the remaining set of actions is schedulable.

The intuition is that if an agent has a very low probability of reaching a state, then ignoring the hazards or temporal transitions to failure (ttfs) in this state does the least harm (assuming all failures are equally bad). We call this the unlikely state (cutoff) strategy. The agent’s failure probability increases by no more than the probability of the state the action was planned for. Note that, if all failure states are not equally bad, our strategy can be modified to accommodate domains with varying degrees of failure. For instance, instead of ignoring the least likely states, we could ignore the least deleterious (disutility) states weighted by their state probabilities.

Acknowledgement

The authors gratefully acknowledge the detailed comments of their Honeywell colleagues, Robert Goldman, David Musliner and Hakan Younes. This research was supported in part by DARPA/AFRL Contract F30602-00-C-0017.

References


infinity. Moreover, it requires a user to specify the knowledge of a transition at each time point. This is an intimidating task, if at all possible. An obvious alternative is to use probability mass functions instead of probability density functions. However, such a naive discrete approximation does not work. Instead, we use what we call probability rate functions (Nelson 1990). For a (temporal) transition \( h \), a user specifies a probability rate function \( \overline{\lambda}(h) \), \( h \geq 1 \), over the \( h \)-th time interval \( [t_h, t_{h+1}) \). \( \overline{\lambda}(h) \) is the probability that \( h \) fires in the \( h \)-th time interval, given that the transition has not fired before \( t_h \) in any of the previous \( h-1 \) time intervals. For example, if a fair coin is flipped each second, the probability rate for the “heads-to-tails” transition is \( 0.5 \) over each second, regardless of how much time has elapsed, given that the state is still “heads” after the flips so far.

To compute transition probabilities, we rewrite Eq. 1 using probability rate functions. Then, we approximate the “true” (temporal) transition probability functions by piecewise constant probability rate functions (Powers and Xie 2000). To model the dependency among a set of concurrent events and actions that match a state \( s \) – to calculate their transition probabilities – we compute the dependent probability rate function \( \overline{\lambda}(h, s) \) for each transition \( \text{trans}_s \) in the state \( \text{trans}_s \) in each time interval \( [t_h, t_{h+1}) \), where \( h \) ranges from 1 to infinity. A dependent probability rate function \( \overline{\lambda}(h, s) \) of a transition in state \( s \) describes the probabilistic temporal dynamics of a transition in that state when there are other concurrent applicable transitions. If there are no other concurrent transitions in the state, then \( \overline{\lambda}(h, s) = \overline{\lambda}(h) \). Otherwise, we approximate \( \overline{\lambda}(h, s) \) by Eq. 2.

\[
\overline{\lambda}(h, s) = \frac{\ln(1 - \overline{\lambda}(h)) (1 - P_{\text{NONE}}(h, s))}{\sum_{\text{trans}_s \in \text{trans}(s)} \ln(1 - \overline{\lambda}(h))} \tag{Eq. 2}
\]

\( \overline{\lambda}(h, s) \) is the set of applicable transitions \( \text{trans}_s \) in state \( s \). \( P_{\text{NONE}}(h, s) \) is the probability that no transition fires in the \( h \)-th time interval in state \( s \). It is given by:

\[
P_{\text{NONE}}(h, s) = \prod_{\eta \in \text{trans}, \eta \in \text{trans}(s)} (1 - \overline{\lambda}(h)) \tag{Eq. 3}
\]

The transition probability for a transition in state \( s \), \( p(\text{trans}_s, s) \), is simply the weighted sum of the transition probabilities over all time intervals, or until residual probability for further time intervals is negligible.

\[
p(\text{trans}_s, s) = \sum_{n=0}^{\infty} \overline{\lambda}(\eta, s) F(\eta, s) \tag{Eq. 4}
\]

\[
F(h, s) = F(h-1, s) P_{\text{NONE}}(h-1, s) = \prod_{\eta=0}^{h-1} P_{\text{NONE}}(\eta, s) \tag{Eq. 4}
\]

\[
P_{\text{NONE}}(0, s) = 1
\]

Probability rate functions, as opposed to, e.g., probability density functions, allow us to approximate the relative likelihood of each transition firing in the same time interval, because they satisfy the properties of exponential distributions. To evaluate our discrete approximation, we have proven analytically that the transition probabilities computed using Eqs. 2 to 4 converge to the values computed using Eq 1 as the lengths of the discretized time intervals get smaller.