Supervision of multiple-robot systems

Kiriakos Kiriakidis
United States Naval Academy
Annapolis, MD 21402
Email: kiriakid@novell.nadn.navy.mil

Diana Gordon
Naval Research Laboratory, Code 5515
Washington, D.C. 20375-5337
Email: gordon@aic.nrl.navy.mil

Abstract

Teams of miniature robots (land-based, airborne, or submerged) constitute a new breed of robotic systems where the issue of controlled behavior arises naturally. At the highest level, the dynamics of such systems are event-driven. Herein, we consider multiple-robot systems whose robot units go offline upon the occurrence of certain events. To supervise such event-varying structure, we propose an adaptive supervisory control method.

Introduction

A platoon of robots is an example of a new breed of robotic systems that comprise multiple units of a certain kind. Another example is a swarm of micro-air-vehicles or a fleet of autonomous submarines. At the system level, the behavior of multiple-unit systems is event-based, hence it admits a finite-state automaton (FSA) model. Furthermore, such discrete event system model enables the design of well-behaved multiple-unit systems via supervisory control theory [1]. A practical and, at the same time, challenging problem arises when one or more units go offline and hence are no longer part of the system. Then, the structure of the multiple-unit system changes; thus, the FSA model of the system is, in general, uncertain.

One way to cope with such uncertainty is to cast the problem in the adaptive supervision framework [2]. This approach, however, is very general and results in a restrictive supervisor when the uncertainty in the model is large. In this paper, we consider a class of systems that is event-varying. Although the evolution of the system structure is unknown a priori, the system model is known posterior any string of events. We propose an adaptive supervisory control method that takes into account all pieces of information on the system structure as they become available.

Modeling and Supervision of Multi-Unit Systems

Let us consider a multiple-unit system whose alphabet, \( \Sigma \), comprises controllable events only. Each unit in such system admits an FSA model, \( G_i \), \( i \in I_M \), where \( I_M \) is an index set and \( M \) the number of units in the system initially. To take into account the possibility that a particular subsystem goes offline and hence is no longer part of the system, we define another FSA, namely, \( \Lambda \) whose alphabet

\[
\Sigma_\Lambda = \{ \lambda_1, \ldots, \lambda_N \}
\]

consists of uncontrollable events. Occurrence of the event \( \lambda_i \) coincides with the subsystem \( G_i \) going offline. At any point in time, the model of the multiple-unit system comprises \( M - k \) FSA, where \( k \) is the number of subsystems that have gone offline until then. Hereafter, we assume that \( k \leq N \) and, in the worst case that if all events in \( \Sigma_\Lambda \) have occurred, the language of the
remaining system of $M - N$ units is still a superset of the desired language.

We define the system where $M$ units are online as follows:

$$G_0 := G_1 \times G_2 \times \ldots \times G_M$$

Let us use an FSA-valued function, $G(s)$, to denote the dependence of the system on the string of events, $s$. For the empty string, $\epsilon$, the model of the multi-unit system $G(\epsilon) = G_0$. Clearly, the structure of the multi-unit system after the first $\lambda$-event occurs depends on the product $G(\epsilon) \times \Lambda$. Indeed, if $\lambda_1$ is the first $\lambda$-event that occurs, the multiple-unit system becomes

$$G_{\lambda_1} := G_1 \times \ldots \times G_{\lambda_1-1} \times G_{\lambda_1+1} \times \ldots \times G_M$$

In general, after a string of events, $s$, which includes the events $\lambda_{i_1}, \ldots, \lambda_{i_k}$, we have

$$G(s\sigma) = \begin{cases} 
G_{\lambda_{i_1} \ldots \lambda_{i_k}}, & \sigma \notin \Sigma_\lambda \\
G_{\lambda_{i_1} \ldots \lambda_{i_k} \lambda_{i_{k+1}}}, & \sigma \in \Sigma_\lambda
\end{cases}$$

In regard to specifications, suppose that, at any point in time, it is necessary that the multiple-unit system satisfies a certain collection of desired properties, $P$. Let $D \subset L(G(\epsilon))$ be a desired language, in the sense that it guarantees $P$. After the event $\lambda_{i_1}$, the first such event, has occurred, we are able to obtain from $D$ the language $D_{\lambda_{i_1}}$ via the algorithm of [3]. This algorithm ensures that $D_{\lambda}$ also guarantees $P$ and, therefore, is a desired language. In general, after the multiple-unit system has executed the events $\lambda_{i_1}, \ldots, \lambda_{i_k}$, the desired language becomes $D_{\lambda_{i_1} \ldots \lambda_{i_k}}$ and it is controllable.

Consider the supervisor $S(s)$. Initially, we have $S(\epsilon) = S_0$, where $L(S_0/G_0) = D$. To implement the aforementioned desired (and controllable) language, we use the following adaptive law, which stems from [2],

$$S(s\sigma) = \begin{cases} 
S(s), & \sigma \notin \Sigma_\lambda \\
D_{\lambda_{i_1} \ldots \lambda_{i_k} \lambda_{i_{k+1}}}, & \sigma \in \Sigma_\lambda
\end{cases}$$

Such supervisor guarantees that the desired properties, $P$, hold regardless of the occurrence of $\lambda$-events.

Conclusion

Compared with existing approaches to adaptive supervisory control, the proposed approach reconfigures the supervisor based on a language that guarantees desired properties and is always controllable. The time complexity of the algorithm that produces the language $D_{\lambda_{i_1} \ldots \lambda_{i_k+1}}$ depends on the number of states in the FSA of its predecessor, i.e., $D_{\lambda_{i_1} \ldots \lambda_{i_k+1}}$, but does not directly depend on the number of units, $M$. Because the new language is controllable this is also the complexity of the supervisor's synthesis step. Outside the proposed approach, however, the computational complexity of adaptive supervision depends on the number of states in $G(s\sigma)$, which increases with the number of units, $M$. Another advantage of the method of this paper is that the desired language is revised after the occurrence of each $\lambda$-event. Hence it is based on the structure of the system at that time rather than all possible structures from that point and on; thus, yielding a less restrictive supervisor.

References

