Event-driven Modelling and Control of a Mobile Robot Population

F.A. Melo, P. Lima, M. I. Ribeiro *
Institute for Systems and Robotics
Instituto Superior Técnico
1049-001 Lisboa
e-mail: {fmelo,pal,mir}@isr.ist.utl.pt

Abstract. This paper proposes the use of a discrete event system to model the navigation of a homogeneous population of mobile robots moving in an environment composed of a number of discrete sites. We derive results that relate the blocking and controllability properties of the automaton modeling the complete system with the blocking and controllability properties of the smaller dimension automata modeling each robot navigating in the environment.

1 Introduction

Autonomous navigation of mobile robots has been considered a key subject of investigation since the pioneer work in mobile robotics. In fact, the ability of a robot to accomplish a certain task in a given environment depends, quite often, on the robot's ability to navigate in its environment and, according to the task purpose, the navigation strategy may change. As an example, Simmons and Koenig [1] used a Partially Observable Markov model to guide and keep track of a robot, such that the robot is able to perform several tasks. In other works, behaviour based navigation is developed using Dynamic Systems theory to generate the behaviours [2, 3]. In the framework of Discrete-event Systems, Hale, Rokonuzzaman and Gosine [4] use Petri Nets as a modeling tool for the navigation of a robot in unstructured environments.

With the bloom of multi-robot systems, the existence of multiple robots navigating simultaneously in a common environment raised new and challenging problems. Some of the results already known regarding the navigation of a single robot were extended to multiple robots. For example, in the work by Balch and Hybinette [5], potential fields are used to achieve multi-robot navigation. The navigation of multiple robots also raised new and interesting problems, such as cooperation and formation control or flocking, as addressed by Balch and Arkin [6], where a reactive behaviour-based approach to formation control is described.

This paper models the navigation and control of a robot population in a discrete event system framework. The navigation of the population of mobile robots is addressed in an environment modeled as a topological map and considers the problem of driving the robots

*Work partially supported by Programa Operacional Sociedade de Informação (POSI) in the frame of QCA III and by the FCT Project Rescue - Cooperative Navigation for Rescue Robots (SRI/32546/99-00). The first author acknowledges the PhD grant SFRH/BD/3074/2000.

from some initial configuration to a final configuration. The multi-robot system is modeled as a finite-state automaton (FSA) whose size is related to the number of robots and to the number of nodes in the topological map representing the environment.

The main contribution of this paper is the analysis of the blocking and controllability properties of this automaton. Since the automaton models the movement of the complete robot population in the environment, from a start configuration to a given goal configuration, properties such as blocking and controllability have direct correspondence with the successful completion of this objective. In fact, a blocking state in the automaton corresponds to a distribution of the robot population in the environment from which the desired goal configuration is not attainable. On the other hand, controllability of the automaton implies that blocking states are avoidable, which means that it is possible to disable some actions to prevent the robots from reaching such blocking configurations.

The presented results relate the blocking and controllability properties of the automaton modeling the multi-robot system (which can be a large-dimension automaton, for complex systems) with the blocking and controllability properties of smaller automata, named *navigation automata*, that model the navigation of each individual robot.

In a decentralized architecture, where each robot has only knowledge of its own movement in the environment, we prove that the verification of some conditions regarding the individual automata are necessary and sufficient to assure the reachability of the objective for the entire population.

The paper is organized as follows. In Section 2 the system under study is described and the FSA modeling the system is presented. Section 3 analyzes blocking properties of the FSA and derives results relating the blocking of the automaton modeling the complete system with the automata modeling each of the robots. In Section 4, the controllability/observability properties of the FSA are analyzed, and the results of Section 3 are extended to controllability/observability. Finally, Section 5 concludes the paper and presents directions for future work.

2 The Finite State Automaton Model

Consider a system of N robots, navigating in a discrete environment (represented by a topological map) consisting of M distinct sites. This will be referred to as a N-R-M-S situation (N robots and M sites). The set of sites is denoted by $\mathcal{S}$. The function $\Omega : \mathcal{S} \to 2^{\mathcal{S}}$ establishes a correspondence between a site i and a set $\mathcal{S}_i \subset \mathcal{S}$ of sites reachable from i in one movement of a robot.

The State-Space

In a general situation, with N non-homogeneous robots and M sites, with no constraints on the number of robots present in each site, each robot can be in M different positions and there are M^N different possible configurations. In such situation, the state-space, X, is the set of all possible robot configurations, yielding $|X| = M^N$. Each state will then be a N-tuple $(x_1, x_2, ..., x_N)$, where x_i is the site where robot i is.

For the simpler case of a homogeneous set of robots and from the point of view of the final objective, some of these states are equivalent, since the robots are indistinguishable. This leads to a simplification in the automaton, since the equivalent configurations can be merged into one single state. This simplified model has a state space with ${}^{M+N-1}\mathcal{C}_{M-1}$ states where each state consists of a M-tuple $(n_1, n_2, ..., n_M)$, with n_i being the number of robots in site i.

The Event Set

Since the environment is composed by a finite number of sites, each robot can take, at each moment, a finite set $\mathcal{A}$ of possible actions of the type $Go(i,j)$, corresponding to going from site i to site j. The events in the system correspond to the movements of the robots. For simplicity, consider that only one robot moves at a time.

The FSA

The notation used throughout this work is similar to the one used in the book by Cassandras and Lafortune [7]. The Finite-State Automaton G describing the system is a six-tuple $(X, E, f, \Gamma, x_0, x_m)$ where:

- X is the state space, with each state being a M-tuple $(n_1, n_2, ..., n_M)$ and n_i being the number of robots in site i;
- E is the set of possible events. The events are denoted by $Go(i,j)$, corresponding to the movement of a robot from site i to site j;
- $f : X \times E \longrightarrow X$ is the transition function;
- $\Gamma : X \longrightarrow 2^E$ is the active event function that can be determined from the function Ω;
- x_0 is the initial state, corresponding to the initial configuration;
- x_m is the only marked state, corresponding to the final configuration.

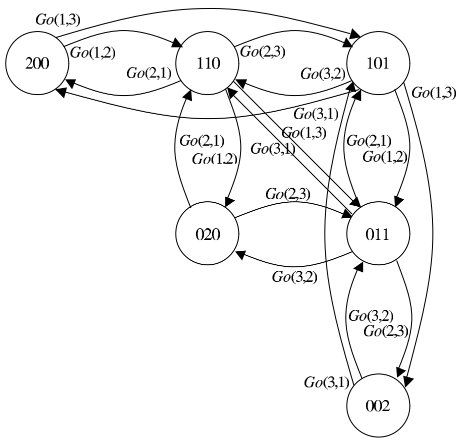

Figure 1: Example of a 2-R-3-S system.

Figure 1 depicts the FSA for a 2-R-3-S system where no initial or final configurations were established.

3 Blocking

Let G be the automaton modeling a N-R-M-S system, as described in Section 2.

A generic automaton $M = (Y, E_M, f_M, \Gamma_M, y_0, Y_m)$ is said to be blocking if $\overline{\mathcal{L}_m(M)} \subsetneq \mathcal{L}(M)$ [7], i.e., there are strings generated by M (strings in $\mathcal{L}(M)$) that are not a prefix to a marked string (strings in $\overline{\mathcal{L}_m(M)}$). This means that there is a set $Y_C \subset Y$, denoted as *blocking* *set*, such that there are no transitions coming out from any state $y \in Y_C$ to any other state $z \notin Y_C$.

In terms of the system under study, if G is blocking, with a blocking set X_C, whenever the robot population reaches a configuration corresponding to a state $x \in X_C$, it is not possible to drive it to the desired goal configuration anymore.

Usually, blocking is checked by verifying exhaustively if $\overline{\mathcal{L}_m(G)} \subsetneq \mathcal{L}(G)$. In the present case, as the system can lead to relatively large automata for not so large M and N, a more effective way to check the blocking properties of G is desirable. Whether the system corresponds or not to a blocking automaton is directly related to the function Ω. By definition, whenever site j is reachable from site i, $j \in \Omega(i)$. If

$$j \in \Omega(i) \Rightarrow i \in \Omega(j), \forall i, j \in S, \tag{1}$$

then G is non-blocking.

Defining the function $\Omega^{-1}(i) : 2^{\mathcal{S}} \rightarrow 2^{\mathcal{S}}$, as $\Omega^{-1}(i) = \bigcap_{j \in \Omega(i)} \Omega(j)$, condition (1) can be re-stated as:

Result 1. *The automaton G describing a N-R-M-S system is non-blocking if*

$$i \in \Omega^{-1}(i), \forall i \in \mathcal{S}. \tag{2}$$

Proof. If condition (1) is met, there is no closed subset of $\mathcal{S}$([1]). This implies that any robot can reach any site from any other site, and, thus, any configuration is reachable from any other configuration, which means that G is non-blocking. It remains only to be proved that conditions (2) and (1) are equivalent.

(1) ⇒ (2): Suppose that there is $i \notin \bigcap_{j \in \Omega(i)} \Omega(j)$. This means that there is $j \in \Omega(i)$ such that $i \notin \Omega(j)$. But then (1) is not met and (1)⇒(2).

(2) ⇒ (1): Suppose that there are $i, j \in S$ such that $i \in \Omega(j)$ but $j \notin \Omega(i)$. Since $j \notin \Omega(i)$, this means that $j \notin \bigcap_{k \in \Omega(j)} \Omega(k)$. But then (2) is not met, this completing the proof. □

Condition (2), though being sufficient, is not necessary for G to be non-blocking. In the sequel, a more general result is derived.

Consider a 1-R-M-S system. Define $\mathcal{R}(x_i)$ as the set of all states x_j such that there is a path from site i to site j. Similarly, if X is a set of states, $\mathcal{R}(X) = \bigcup_{x \in X} \mathcal{R}(x)$. For a N-R-M-S system, $\mathcal{R}$ is defined similarly in terms of the corresponding automaton. The automaton modeling the 1-R-M-S system is a replica of the topological map of the environment, with each state corresponding to a node. The marked state x_m will be the one corresponding to the target site, m.

For a N-R-M-S situation described by an automaton G, define a navigation automaton $G(y_m)$ as a six-tuple $(Y, E_m, f_m, \Gamma_m, y_0, \{y_m\})$, where:

- Y is the state space, where each state corresponds to a site;
- E_m is the set of possible events, corresponding to the allowed transitions between sites;
- Γ_m is the active event function, which is equivalent to function Ω;
- $\{y_m\}$ is a singleton since y_m is the only marked state—it corresponds to a site in $\mathcal{S}_F$,

[1]We define a set of sites $\mathcal{S}_C$ to be closed if there is no path out of $\mathcal{S}_C$ to some site $i \notin \mathcal{S}_C$.

with $\mathcal{S}_F$ being the set of all sites with at least one robot in the desired configuration.

Notice that each of these automata models the navigation of a single robot in the environment. Moreover, each automaton $G(y_m)$ corresponds to a different target site in $\mathcal{S}_F$. It is now possible to prove the following result.

Result 2. *The automaton G describing a N-R-M-S system is non-blocking iff* all *the* navigation automata $G(y_m)$ *defined above are non-blocking, with any initial condition y_0 corresponding to a state from the initial configuration.*

Proof. **(If)** Consider one of the navigation automata $G(y_m)$. If this automaton is non-blocking, the marked state $y_m \in \mathcal{R}(Y_0)$ (Y_0 is the set of states corresponding to sites with some robot in the initial configuration). If this is true for all automata $G(y_m)$, this means that each of the robots can reach any of the sites in $\mathcal{S}_F$, yielding

$$x_m \in \mathcal{R}(x), \forall x \in X_0, \tag{3}$$

and, thus, G is non-blocking.

(Only if) If G is non-blocking, the target configuration is within reach from any other configuration reachable from x_0, which implies that all $G(y_m)$ are non-blocking. ☐

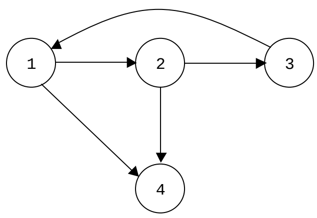

Figure 2: Topological map for a N-R-4-S system.

As an example on the application of Result 2, consider a N-R-4-S system with the topological map of Figure 2. Notice that, with proper labeling, initial state and marked state, this map coincides with one of the navigation automata. Since there are no paths from site 4 to any other site, Result 2 leads to the conclusion that the system is non-blocking if and only if the only marked state is $x_m = (0, 0, 0, N)$.

If, on the previous example, $N = 2$, the complete automaton is represented in Figure 3 and the previous statement can be checked by inspection. In fact, notice that the set

$$X_C = \{(1, 0, 0, 1); (0, 1, 0, 1); (0, 0, 1, 1); (0, 0, 0, 2)\},$$

is a closed set and $x_m \notin X_C$.

With Result 2, blocking properties of an automaton with ${}^{M+N-1}\mathcal{C}_{M-1}$ states can be inferred from the blocking properties of, at most, $\min\{M, N\}$ automata each one with M states. For large systems, this greatly reduces the analysis effort.

4 Supervisory Control

In this section, the controllability of the automaton describing a N-R-M-S system is analyzed.

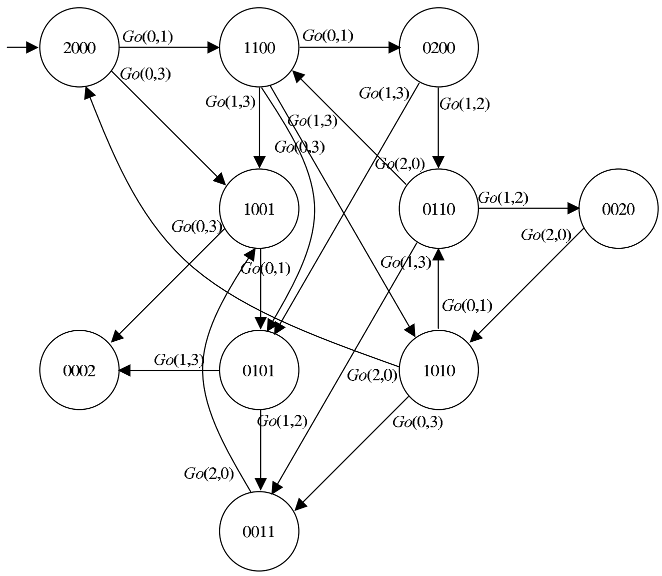

Figure 3: Automaton G for a 2-R-4-S system.

As referred in Section 3, the only marked state in the automaton is the one corresponding to the goal configuration. Therefore, the automaton G was built in order to mark the desired language. Nevertheless, in a situation where the automaton is blocking, it is not desirable that the system reaches a blocking state, since this will prevent the final configuration to be reached. The presence of a supervisor S necessarily relates to this situation, where blocking must be prevented.

For a general automaton M, if K is the desired marked language, there is a non-blocking supervisor S if and only if $\overline{K}E_{uc} \cap \mathcal{L}(M) \subset \overline{K}$ and $K = \overline{K} \cap \mathcal{L}_m(S/M)$ [7], where $E_{uc} \subset E$ is the set of uncontrollable events. In the present situation the system already marks the desired language, and, thus, if the system is controllable, the controlled system is necessarily non-blocking.

As seen in Section 3, blocking states prevent the system from accomplishing its objective. Therefore, it is important to determine the existence of a supervisor that can prevent blocking, i.e., it is important to determine if the system is controllable.

This analysis will be conducted in two different steps: in a first approach, some transitions are considered uncontrollable, but all are considered observable. In a second approach, the uncontrollable transitions are considered unobservable.

One remark should be made regarding this modeling of the uncontrollable and unobservable events. The unobservable events seek to model the uncertainty in the transitions of the system. By introducing unobservable transitions of the robots between states, it is possible to model uncertainty about the actual position of the robots. Since each event represents motion actions for the robots, the uncertainty is related to the state reached after some action, and therefore it is strictly coupled to controllability. The uncontrollability of these unobservable events is, therefore, meant to avoid artificial disabling of the uncertainty in these transitions by a supervisor.

4.1 Limited Control with Full Observation

Consider that there is a non-empty set of uncontrollable events $E_{uc} \subset E$. As stated, supervisory control only makes sense when blocking is involved. This means that at least one of the navigation automata $G(y_m)$ is blocking. This, in turn, means that there is a closed set of states, Y_C, such that $y_m \notin \mathcal{R}(Y_C)$.

This navigation automaton will not meet the controllability condition if and only if the set of uncontrollable events E_{uc} is such that a string $s = e_1...e_k$ exists which verifies $\{e_i, i = 1, ..., k\} \subset E_{uc}$ and $f_m(y, s) \in Y_C$ for some state y between y_0 and y_m. Generalizing for a N-R-M-S system, Result 3 is obtained, which is dual of the Result 2 derived in Section 3.

Let I be the set such that, for $i \in I, G(y_i)$ are blocking navigation automata (notice that $I \neq \emptyset$, since G is assumed to be blocking). Y_{Ci} will denote the blocking set of automaton $G(y_i)$. Define $Y_B = \bigcup_{i \in I} Y_{Ci}$ and $Y_{NB} = Y \backslash Y_B$.

Result 3. *The blocking automaton G describing a N-R-M-S is controllable iff the automaton $G_{uc}(Y_{NB}), i \in I$ is controllable, with respect to the language $K = \mathcal{L}_m(G_{uc}(Y_{NB}))$, with any initial condition y_0 corresponding to a state from the initial configuration..*

The automaton $G_{uc}(Y_{NB})$ is defined by the six-tuple $(Y, E_m, f_m, \Gamma_m, y_0, Y_m)$, where

- *Y, E_m, f_m and Γ_m are defined as in Result 2;*
- *y_0 is the initial condition;*
- *$Y_m = Y_{NB}$.*

Proof. **(If)** Suppose the automaton $G_{uc}(Y_{NB})$ is controllable for any initial condition. This means that any string that takes a robot to the blocking set Y_{Ci} is not composed uniquely by uncontrollable events. This implies that blocking can be prevented for all the automata $G(y_i)$ as defined in Result 2, by disabling only controllable events (which correspond to controllable events in the general automaton G) which, in turn, implies that system G is controllable.

(Only if) The automaton G is uncontrollable if there is a string of uncontrollable events that drives it to its blocking set. Similarly, it is controllable if no such string exists. This means that the controlled system is non-blocking and, thus, the events disabled by the supervisor are not uncontrollable. These events correspond to controllable events that would lead some of the blocking navigation automata $G(y_i)$ to its blocking set. But then $G_{uc}(Y_{NB})$ is controllable for any initial condition. □

4.2 Limited Control and Observation

In this subsection, it is considered that the uncontrollable events referred in Section 4.1 are also unobservable.

Let E_{uo} be the set of unobservable events. Given Result 3, the hypothesis of $E_{uo} = E_{uc} \neq \emptyset$ does not add any considerable complexity to the problem. In fact, in this situation, if the system is controllable, it is also observable (see [7]), and Result 3 allows to simultaneously conclude about controllability and observability and it can be restated as follows.

Result 4. *The blocking automaton G describing a N-R-M-S is controllable (and, thus, observable) iff the automaton $G_{uc}(Y_{NB})$ defined in Result 3 is controllable (and, thus, observable), with respect to the language $K = \mathcal{L}_m(G_{uc}(Y_{NB}))$. The set of uncontrollable and unobservable events of G, E_{uc} and E_{uo}, verify $E_{uc} = E_{uo}$.*

As an example of the application of Result 4, consider again the system described by the automaton G of Figure 3. The only events that drive the automaton to the set X_C are events of the type $Go(i, 3)$, with $i \in \{0, 1\}$. This leads to the immediate conclusion that G is controllable except if $E_{uc} = U = \{Go(i, 3), i \in \{0, 1\}\}$.

Notice that, for this example, $Y_{NB} = \{1, 2, 3\}$. From Result 4, this means that, if the automaton $G_{uc}(Y_{NB})$ is controllable, so is G. But $G_{uc}(Y_{NB})$ is controllable as long as $E_{uc} \neq U$, as expected.

5 Conclusions and Future Work

In this paper the problem of controlling the navigation of a set of mobile robots operating in a discrete environment was discussed. Relevant results have been derived, that allow the use of small dimension automata (navigation automata) to infer about the blocking properties of the general automaton that describes the complete system. These results were extended to cope with controllability and observability issues.

In a situation where a specific configuration is aimed for a set of robots, the presented results allow to determine in a decentralized manner (i.e., locally at each individual robot), if the global objective is achievable, and if blocking configurations are avoidable. As an example, in the situation of Figure 2, where a set of robots is to navigate with decentralized control, supposing that the marked state was other than $[\ 0\quad 0\quad 0\quad N\ ]$, each robot would realize on its own that state 4 should be avoided, or that some coordination strategy was to be implemented in order to assure that the number of robots going to site 4 was strictly the essential in order for the goal configuration to be achieved.

The extension of the present work to the situation where a heterogeneous set of robots is considered is envisaged for future research. It is also of interest to relate the blocking properties of the navigation automata with the ergodicity of the Markov Chain which can be used to model the complete system, when a probabilistic uncertainty is associated to the events representing the movements of the robots.

References

[1] R. Simmons and S. Koenig, "Probabilistic Robot Navigation in Partially Observable Environments," *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1080–1087, 1995.

[2] A. Steinhage, "Dynamical Systems for the Generation of Navigation Behaviour," Ph.D. dissertation, Institut für Neuroinformatik, Ruhr-Universität, Bochum Germany, November 1997.

[3] P. Althaus, H. I. Christensen, and F. Hoffmann, "Using the Dynamical System Approach to Navigate in Realistic Real-world Environments," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1023–1029, 2001.

[4] R. D. Hale, M. Rokonuzzaman, and R. G. Gosine, "Control of Mobile Robots in Unstructured Environments Using Discrete Event Modeling," *SPIE International Symposium on Intelligent Systems and Advanced Manufacturing*, September 1999.

[5] T. Balch and M. Hybinette, "Social potentials for scalable multirobot formations," *IEEE International Conference on Robotics and Automation (ICRA-2000), San Francisco, USA*, pp. 73–80, 2000.

[6] T. Balch and R. C. Arkin, "Behavior-based formation control for multi-robot teams," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, December 1998.

[7] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, ser. The Kluwer International Series On Discrete Event Dynamic Systems. Kluwer Academic Publishers, 1999.