Discretizing the State Space of Multiple Moving Robots to Verify Visibility Properties

Ali Narenji Sheshkalani(✉), Ramtin Khosravi, and Mohammad K. Fallah

School of Electrical and Computer Engineering, University of Tehran, Tehran, Iran
{narenji,r.khosravi,mk.fallah}@ut.ac.ir

Abstract. In a multi-robot system, a number of autonomous robots sense, communicate, and decide to move within a given domain to achieve a common goal. To prove such a system satisfies certain properties, one must either provide a manual proof, or use an automated verification method. To enable the second approach, we propose a method to automatically generate a discrete state space of a given robot system. This allows using existing tools and algorithms for model checking a system against temporal logic properties. We construct the state space such that properties regarding the visibility of the robots moving along the boundaries of a simple polygon can be model checked. Using our method, there is no need to manually prove that the properties are preserved with every change in the motion algorithms of the robots.

Keywords: Moving robots · Model checking · Visibility

1 Introduction

Mobile robots are able to sense, communicate, and interact with the physical world, and are able to collaboratively solve problems in a wide range of applications (see, for example, [2,3,5,14]).

There has been a close relationship between robot motion planning and computational geometry in the applications where the robots are constrained to move within a geometric domain, like extensions of art-gallery problems [13] to the cases where a number of moving robots must guard a polygonal domain. Efrat et al. [7] considered the problem of sweeping simple polygons with a chain of guards. They developed an algorithm to compute the minimum number of guards needed to sweep an n-vertex polygon that runs in $O(n^3)$ time and $O(n^2)$ working space.

Traditionally, the correctness of robot motion planning algorithms within the context of computational geometry is investigated by manual proofs. It may be hard for certain types of planning algorithms to prove they correctly satisfy the problem's constraints. On the other hand, when it comes to practical applications of motion planning algorithms, the designer may heuristically adjust the algorithm's parameters or even the whole strategy in order to find the best solution that fits both the problem constraints and practical restrictions. In these cases, manually proving the algorithm with every change may be impractical.

© Springer International Publishing Switzerland 2015
C. Dixon and K. Tuyls (Eds.): TAROS 2015, LNAI 9287, pp. 186–191, 2015.
DOI: 10.1007/978-3-319-22416-9_22

An alternative and more reliable approach to examine the correctness of the planning algorithms is formal verification, and more specifically, model-checking [4]. In a few existing works such as [8,9], model checking has been employed in motion planning algorithms. Another related area to which model-checking techniques have been applied are robot swarms. In [12] a swarm of foraging robots is analyzed using the probabilistic model-checker PRISM [11]. Dixon et al. [6] used model-checking techniques to check whether temporal properties are satisfied in order to analyze emergent behaviors of robotic swarms.

In this paper, we focus on discretizing the state space to verify certain properties (*Connectivity* and *Covering*) on a multi-robot system where each robot is programmed with an arbitrary navigation algorithm.

2 Problem Definition

A simple polygon P is defined as a closed region in the plane bounded by a finite set of line segments such that there exists a path between any two points inside P which intersects no edge of P [10]. For simplicity, the boundary of P is denoted by $\beta(P)$. Two points p and q in P are said to be *visible* if the line segment joining p and q contains no point on the exterior of P.

For a simple polygon P, we use the notation V_p for the visibility polygon of a point $p \in P$. Removing V_p from P may result in a number of disconnected regions we call *invisible regions*. Any invisible region has exactly one edge in common with V_p, called a *window* of p, which is characterized by a reflex vertex of P visible from p, like p'. The window is defined as the extension of the (directed) segment pp' from p' to $\beta(P)$. We denote such a window by $w(p, p')$.

Consider a simple polygon P whose boundary is specified by the sequence of n vertices $< p_1, p_2, \ldots, p_n >$ including the set of reflex vertices P_{ref} and convex vertices P_{conv}, and a set $R = \{r_1, r_2, \ldots, r_k\}$ of k robots. The set of robot navigation algorithms $Alg = \{a_1, a_2, \ldots, a_k\}$ (a_i is the navigation algorithm of robot r_i) is given with the following properties:

1. The robots only move on $\beta(P)$,
2. Each step in the movement of each robot is specified by two parameters: direction (clockwise or counterclockwise) and distance (real positive number).

To discretize the state space of the problem, we assume that the robots have turn-based movements. It means that during the movement of a robot, the position of other robots is fixed. By decreasing the time units, we can get closer to a more realistic movement of robots. Having the state space of the robot system in terms of a transition system, we can apply existing model-checking algorithms to verify the correctness of the desired properties.

The correctness properties may be built using temporal logics which are formalisms to describe temporal properties of reactive systems [1]. We define the following two atomic propositions to be used in temporal logic formulas:

Definition 1 (Connectivity). *The set of robots are connected if the graph induced by the visibility relation between pairs of robots is connected.*

Definition 2 (Covering). *The robots cover P if the union of the visibility polygons of all robots covers the whole P.*

Since we do not deal with the details of model-checking algorithms directly in this paper, we refer the reader for a detailed description of temporal logics to [1]. However, to bring an example, the LTL formula $\Diamond(\mathit{Covering} \wedge \neg \mathit{Connectivity})$ describes the property that eventually (represented by $\Diamond$) the system reaches a state in which P is covered by the robots, but the robots are not connected.

We define a robot system RS as the triple $(P, \mathit{init}, \mathit{Alg})$ in which P indicates the environment of the robots to navigate, *init* specifies the initial positions of robots along $\beta(P)$, and *Alg* is the set of navigation algorithms of robots. Our goal is to define a transition system equivalent of RS, over which temporal logic formulas may be model-checked.

3 Constructing the Discrete State Space

With the ultimate goal of verifying a temporal logic formula over a robot system $RS = (P, \mathit{init}, \mathit{Alg})$, we must first construct the equivalent transition system of RS. As mentioned before, the states are labeled with the atomic propositions. Hence, the transition system is called a labeled transition system (LTS) [1].

We define the LTS of RS as the tuple $(S, \mathit{Act}, \hookrightarrow, s_0, AP, L)$ where

- S is the set of states (defined bellow),
- $\mathit{Act} = \{\overrightarrow{\mathit{move}}_r, \overleftarrow{\mathit{move}}_r | r \in R\}$ is the set of actions denoting the movement of robot r clockwise or counterclockwise respectively,
- $\hookrightarrow \subseteq S \times \mathit{Act} \times S$ is the transition relation, (we use the notation $s \overset{\alpha}{\hookrightarrow} s'$ whenever $(s, \alpha, s') \in \hookrightarrow$),
- $s_0 \in S$ is the initial state (determined based on *init*),
- $AP = \{\mathit{Connectivity}, \mathit{Covering}\}$ is the set of atomic propositions,
- $L : S \longrightarrow 2^{AP}$ is the labeling function.

3.1 System States

The satisfiability of AP depends on the distribution of robots on $\beta(P)$. We model each state of the system based on the topology of robots and vertices of P.

Definition 3. *Let the window $w(p, p')$ has two endpoints p' and p''. We call the endpoint p'' the projection of p on $\beta(P)$ with respect to p', and denote it by $\pi_p(p')$. Also, we lift the notation to $\pi_p = \bigcup_{p' \in P_{ref}} \pi_p(p')$, and further to $\Pi = \bigcup_{p \in P \cup R} \pi_p$.*

We define a state as the sequence $s =< q_1, q_2, \ldots, q_m >$ of all points in $\Pi \cup P \cup R$ on $\beta(P)$ in the clockwise order. Without loss of generality, we consider $q_1 = p_1 \in P$ as the starting point in s. Obviously, there exist some sequences which they do not present any feasible state.

Since model checking algorithms assume each atomic proposition is either true or false in a state, the following lemma states that moving of the robots does not change the validity of the propositions *Covering* and *Connectivity*, as long as the sequence defined above remains the same.

Lemma 1. *Each state s can be uniquely labeled with the atomic propositions $AP = \{Connectivity, Covering\}$.*

Proof. Assume that the labeling $L(s) \in 2^{AP}$ is satisfied by the current state s. It is sufficient to prove that by moving the robots, $L(s)$ is valid while s does not change. We discuss two atomic propositions separately.
Connectivity. The change in connectivity of the robots may happen only when the visible set (of robots) of at least one robot is changed. The visible set of a robot r changes if r crosses a window of another robot r'. Note that in this case the visible set of r' changes too. Since the endpoints of all windows are included in the set $\Pi \cup P_{ref}$, the sequence of the points in s is changed in this case.
Covering. Assume that robot r is going to move. The covering of P may change only in the following two cases, in both there is a change in the current state.

(a) The number of vertices of P visible by r changes. In this case, the point $r \in R$ crosses some points in the set $\bigcup_{p \in P} \pi_p$.
(b) There exists a robot r' such that one of the endpoints of the visible part of $\beta(P)$ to r crosses an endpoint of the visible part of $\beta(P)$ to r'. In this case, some point in the set π_r crosses some point in the set $\pi_{r'} \subseteq \Pi$. □

Corollary 1. *The satisfiability of each element of AP is decidable for all states, and each state can be uniquely labeled with respect to Lemma 1.*

3.2 Transitions Events

We define $\overrightarrow{move}_r$ to be the tuple (CW, δ), where δ is the smallest distance robot r can move in clockwise direction which causes a change in state. we define $\overleftarrow{move}_r$ for the counterclockwise (CCW) direction similarly. We define $\hookrightarrow$ as the smallest relation containing the triples (s, α, s'), where $s \in S$, $\alpha \in \bigcup_{r \in R} \{\overrightarrow{move}_r, \overleftarrow{move}_r\}$, and s' is the state obtained from s by taking the action α.

A transition $s \overset{\alpha}{\hookrightarrow} s'$ can occur for two reasons:

(a) Robot r crosses a point in $\Pi \cup P \cup R$,
(b) One of the points in π_r crosses a point in Π.

Assume that robot r moves in some direction (actions $\overrightarrow{move}_r$ or $\overleftarrow{move}_r$), and the current state s changes s'. Based on the definition of transitions, one of the following cases happens:

1. r crosses a point $p \in P \cup R$: in this case, the order of r and p are swapped with each other in the sequence of s, and for all $p' \in P_{ref}$ such that $rp', pp' \in \Pi$ the order of rp' is swapped with pp'.
2. r crosses a point in Π: in this case, there exists a point $p' \in P_{ref}$ such that $pp' \in \Pi$. So, r is swapped with pp', and rp' is swapped with p in s.
3. r crosses a point in Π': in this case, there exist points $p \in P \cup R$ and $p', p'' \in P_{ref}$ such that $pp' \in \Pi$, and r crosses the point $pp'p'' \in \Pi'$. So, the point pp' is swapped with rp'' in the sequence of s.

4 Analysis

Lemma 2 demonstrates an upper bound on the maximum number of states for the given robot system $RS = (P, init, Alg)$. The upper bound obtained in the lemma is not tight. In other words, the geometrical properties of polygon P and the geometry of the projections between the sets R and $\bigcup_{p\in R} \pi_p$ highly affects on the size of the state space.

Lemma 2. *The maximum number of states in order to verify the given robot system $RS = (P, init, Alg)$ has the complexity of $O(n^{2^{k+2}})$ in which n indicates the number of vertices of P, and k specifies the number of robots on $\beta(P)$.*

Proof. Let $\Pi_P = \bigcup_{p\in P} \pi_p$. We define the set of event points $I = \bigcup_{p\in \Pi_P} \pi_p \cup \bigcup_{p\in P} \pi_p \cup P$ as the set of events whose placements are fixed on $\beta(P)$. The following recursive relation $T(k) \in O(n^{2^{k+2}})$ formulates an upper bound on the maximum number of states with presence of k robots on $\beta(P)$:

$$T(k) = \begin{cases} |I| & \text{if } k = 1 \\ T(k-1) \times \big(T(k-1) + O(|P_{ref}|^2)\big) & \text{if } k \geq 2 \end{cases} \qquad \square$$

We have implemented a program to enumerate the states for a given robot system. Table 1 indicates the maximum number of possible states with respect to the polygons shown in Fig. 1 and the number of robots on the boundary of the polygon.

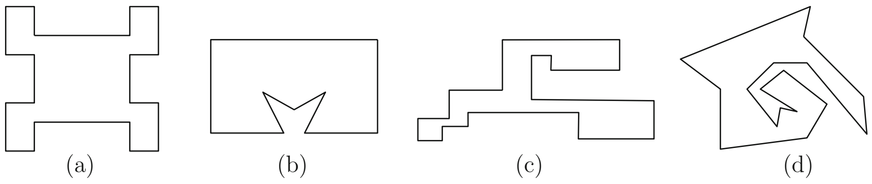

Fig. 1. The polygons used for experimental results

Table 1. The maximum number of states computed with respect to type of the polygons in Fig. 1 and the number of robots

Polygon	Complexity			
	$k = 1$	$k = 2$	$k = 3$	$k = 4$
Polygon (a)	133	3,389	233,683	363,719,760
Polygon (b)	23	406	9,674	482,640
Polygon (c)	135	4,254	308,706	418,697,132
Polygon (d)	84	2,481	137,806	61,352,369

5 Conclusion

We presented a method to construct a discrete state space for a multi-robot system with the aim of verifying correctness properties expressed in temporal logic formulas. The notion of state has been defined in such a way that each state can be uniquely labeled with the atomic propositions *Connectivity* and *Covering*. This way, the modeler can provide the navigation algorithms and verify temporal formulas constructed over the mentioned propositions using existing model checking algorithms. This eliminates the need for manually proving the algorithm(s) each time a change in the algorithm is made.

References

1. Baier, C., Katoen, J.P.: Principles of model checking. MIT Press (2008)
2. Beard, R.W., McLain, T.W., Nelson, D.B., Kingston, D., Johanson, D.: Decentralized cooperative aerial surveillance using fixed-wing miniature uavs. Proceedings of the IEEE **94**(7), 1306–1324 (2006)
3. Cannell, C.J., Stilwell, D.J.: A comparison of two approaches for adaptive sampling of environmental processes using autonomous underwater vehicles. In: Proceedings of the MTS/IEEE OCEANS, Washington, DC, pp. 1514–1521 (2005)
4. Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press (1999)
5. Davison, A., Kita, N.: Active visual localisation for cooperating inspection robots. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Takamatsu, Japan, pp. 1709–1715 (2000)
6. Dixon, S., Winfield, A.F.T., Fisher, M., Zeng, C.: Towards temporal verification of swarm robotic systems. Robotics and Autonomous Systems **60**(2), 1429–1441 (2012)
7. Efrat, A., Leonidas, J.G., Har-Peled, S., Lin, D.C., Mitchell, J.S.B., Murali, T.M.: Sweeping simple polygons with a chain of guards. In: SODA 2000, pp. 927–936 (2000)
8. Fainekos, G.E., Girard, A., Kress-Gazit, H., Pappas, G.J.: Temporal logic motion planning for dynamic robots. Automatica **45**(2), 343–352 (2009)
9. Fainekos, G.E., Kress-Gazit, H., Pappas, G.: Temporal logic motion planning for mobile robots. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 2020–2025. IEEE Computer Society Press (2005)
10. Ghosh, S.K.: Visibility algorithms in the plane. Cambridge University Press (2007)
11. Hinton, A., Kwiatkowska, M., Norman, G., Parker, D.: PRISM: a tool for automatic verification of probabilistic systems. In: Hermanns, H., Palsberg, J. (eds.) TACAS 2006. LNCS, vol. 3920, pp. 441–444. Springer, Heidelberg (2006)
12. Konur, S., Dixon, C., Fisher, M.: Analysis robot swarm behaviour via probabilistic model checking. Robotics and Autonomous Systems **60**(2), 199–213 (2012)
13. O'rourke, J.: Art gallery theorems and algorithms. Oxford University Press (1987)
14. Sugiyama, H., Tsujioka, T., Murata, M.: Collaborative movement of rescue robots for reliable and effective networking in disaster area. In: Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Worksharing, San Jose, CA (2005)