Proceedings of the 2006 American Control Conference
Minneapolis, Minnesota, USA, June 14-16, 2006

Network Configuration Control Via Connectivity Graph Processes

Abubakr Muhammad
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104
abubakr@seas.upenn.edu

Magnus Egerstedt
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332
magnus@ece.gatech.edu

Abstract— **In this paper, we discuss how to generate trajectories, modeled as connectivity graph processes, on the space of graphs induced by the network topology. We discuss the role of feasibility, reachability, and optimality in this context. In particular, we study in detail the role of reachability and the computation of reachable sets by using the cylindrical algebraic decomposition (CAD) algorithm.**

I. INTRODUCTION

The problem of coordinating multiple autonomous agents has attracted significant attention in recent years. Due to advances in many enabling technologies such as advanced communication systems, novel sensing platforms, and cheap computation devices, the realization of large scale networks of cooperating mobile agents has become possible. An important theme in the study of such systems is the use of graph-theoretic models for describing the local interactions in the network. Notable results have been presented in [1], [2], [3], [4], [5]. The conclusion to be drawn from these research efforts is that a number of questions can be answered in a natural way by abstracting away the continuous dynamics of the individual agents.

In [6], [7], the authors have presented a detailed study of graphs that arise due to the limited sensory perception or communication of individual agents in a formation. We have shown that the graphs that can represent formations do in fact correspond to a proper subset of all graphs, denoted by the set of connectivity graphs. We have presented several examples of graphs that fail to exist as connectivity graphs. The idea of infeasibility in the configuration space has been explored further in [8], where we have used techniques from semi-definite programming to obtain the required infeasibility certificates. Based on these results, we have presented in [9] a computational framework in the context of formation switching for achieving a global objective. At the heart of this framework lies the concept of a connectivity graph process, that is made possible by understanding issues concerning feasibility, reachability and optimality. Several applications of this framework have been outlined in [9], that include the production of low-complexity formations and collaborative beamforming in sensor networks. In this paper, we focus on one particular aspect of this framework, namely the question of reachability in connectivity graph processes. In particular, we give details on the use of the cylindrical algebraic decomposition (CAD) algorithm from real algebraic geometry for obtaining connectivity graph processes.

This paper is organized as follows. We first summarize our previous work on connectivity graphs (Section II). We introduce the concept of connectivity graph processes and use semi-definite programming techniques for determining feasible switchings (Section III). We then discuss the computation of reachable sets from a given initial graph and provide details on the use of the CAD algorithm (Section IV). We set up the framework for obtaining optimal trajectories as graph processes (Section V). We then provide some simulation results to show that our method is computable, followed by our conclusions (Section VI).

This work was sponsored by the US Army Research Office through the grant #99838.

II. FORMATIONS AND CONNECTIVITY GRAPHS

Graphs can model local interactions between agents, when individual agents are constrained by limited knowledge of other agents. In this section we summarize some previous results [6] of a graph theoretic formalism for describing formations in which the primary limitation of perception for each agent is the limited range of its sensor. Suppose we have N such agents with identical dynamics evolving on $\mathbb{R}^2$. Each agent is equipped with a range limited sensor by which it can sense the position of other agents. All agents have identical sensor ranges δ. Let the position of each agent be $\mathbf{x}_n \in \mathbb{R}^2$, and its dynamics be given by

$$\dot{\mathbf{x}}_n = f(\mathbf{x}_n, u_n), \tag{1}$$

where $u_n \in \mathbb{R}^m$ is the control for agent n and $f : \mathbb{R}^2 \times \mathbb{R}^m \to \mathbb{R}^2$ is a smooth vector field. The configuration space $\mathcal{C}^N(\mathbb{R}^2)$ of the agent formation is made up of all ordered N-tuples in $\mathbb{R}^2$, with the property that no two points coincide, i.e.

$$\mathcal{C}^N(\mathbb{R}^2) = (\mathbb{R}^2 \times \mathbb{R}^2 \times \dots \mathbb{R}^2) - \Delta, \tag{2}$$

where $\Delta = \{(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N) : \mathbf{x}_i = \mathbf{x}_j \text{ for some } i \neq j\}$. The evolution of the formation can be represented as a trajectory $\mathcal{F} : \mathbb{R}_+ \to \mathcal{C}^N(\mathbb{R}^2)$, usually written as $\mathcal{F}(t)(\mathbf{x}_1(t), \mathbf{x}_2(t), \dots \mathbf{x}_N(t))$ to signify time evolution. The spatial relationship between agents can be represented as a graph in which the vertices of the graph represent the agents, and the pair of vertices on each edge tells us that the corresponding agents are within sensor range δ of each other.

1-4244-0210-7/06/$20.00 ©2006 IEEE

Let $\mathcal{G}_N$ denote the space of all possible graphs that can be formed on N vertices $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$. Then we can define a function $\Phi_N : C^N(\mathbb{R}^2) \to \mathcal{G}_N$, with $\Phi_N(\mathcal{F}(t)) = \mathcal{G}(t)$, where $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}(t)) \in \mathcal{G}_N$ is the *connectivity graph* of the formation $\mathcal{F}(t)$. $v_i \in \mathcal{V}$ represents agent i at position $\mathbf{x}_i$, and $\mathcal{E}(t)$ denotes the edges of the graph. $e_{ij}(t) = e_{ji}(t) \in \mathcal{E}(t)$ if and only if $\|\mathbf{x}_i(t) - \mathbf{x}_j(t)\| \leq \delta$, $i \neq j$. The graphs are always undirected because the sensor ranges are identical. The motion of agents in a formation may result in the removal or addition of edges in the graph. Therefore $\mathcal{G}(t)$ is a dynamic structure. Lastly and most importantly, every graph in $\mathcal{G}_N$ is not a connectivity graph. The last observation is not as obvious as the others, and it has been analyzed in detail in [6]. A *realization* of a graph $\mathcal{G} \in \mathcal{G}_N$ is a formation $\mathcal{F} \in \mathcal{C}^N(\mathbb{R}^2)$, such that $\Phi_N(\mathcal{F}) = \mathcal{G}$. An arbitrary graph $\mathcal{G} \in \mathcal{G}_N$ can therefore be *realized* as a connectivity graph in $\mathcal{C}^N(\mathbb{R}^2)$ if $\Phi_N^{-1}(\mathcal{G})$ is nonempty. We denote by $\mathcal{G}_{N,\delta} \subseteq \mathcal{G}_N$, the space of all possible graphs on N agents with sensor range δ, that can be realized in $\mathcal{C}^N(\mathbb{R}^2)$.

Formations can produce a wide variety of graphs for N vertices. This includes graphs that have disconnected subgraphs or totally disconnected graphs with no edges. However the problem of switching between different formations or of finding interesting structures within a formation of sensor range limited agents can only be tackled if no subformation of agents is totally isolated from the rest of the formation. This means that the connectivity graph $\mathcal{G}(t)$ of the formation $\mathcal{F}(t)$ should always remain *connected* (in the sense of connected graphs) for all time t.

III. Feasible connectivity graph transitions

Connectivity graph processes are generated through the movement of individual nodes. For a connectivity graph $G_j = (\mathcal{V}_j, \mathcal{E}_j) = \Phi_N(\mathbf{x}(t_j))$ let the nodes be partitioned as $\mathcal{V}_j = \mathcal{V}_j^0 \cup \mathcal{V}_j^m$, where the movement of the nodes in V_j^m facilitates the transition from G_j to the next graph G_{j+1} and V_j^0 is the set of nodes that are stationary. With the positions $\mathbf{x}_j^0 = \{\mathbf{x}_m(t_i)\}_{m \in \mathcal{V}_j^0}$ being fixed, let $\mathrm{Feas}(G_j, \mathcal{V}_j^m, \mathbf{x}_j^0) \subseteq \mathcal{G}_{N,\delta}$ be the set of all connected connectivity graphs that are feasible by an unconstrained placement of positions corresponding to $\mathcal{V}_j^m$ in $\mathbb{R}^2$. (We will often denote this set as $\mathrm{Feas}(G_j, \mathcal{V}_j^m)$, when the the positions $\mathbf{x}_j^0$ are understood from context.)

It will be appropriate to explain the reason for keeping track of mobile and stationary nodes at each transition. In principle, it is possible to compute this entire set of feasible transitions by an enumeration procedure. However, in order to manage the combinatorial growth in the number of possible graphs, it is desirable to let the transitions be generated by the movements of a small subset of nodes only. In fact, we will investigate the situation where only one node is allowed to move at a time. Hence, we let $\mathcal{V}_j^0 = \{1, \ldots, k-1, k+1, \ldots, N\}$ and $\mathcal{V}_j^m = \{k\}$. It should be noted that the movement of node k can only result in the addition or deletion of edges that have node k as one of its vertices. Therefore the enumeration of the possible resulting graphs should count all possible combinations of such deletions and additions. This number can be easily seen to be 2^{N-1} for N nodes. Since we are also required to keep the graph connected at all times, this number is actually $2^{N-1} - 1$, obtained after removing the graph in which node k has no edge with any other node.

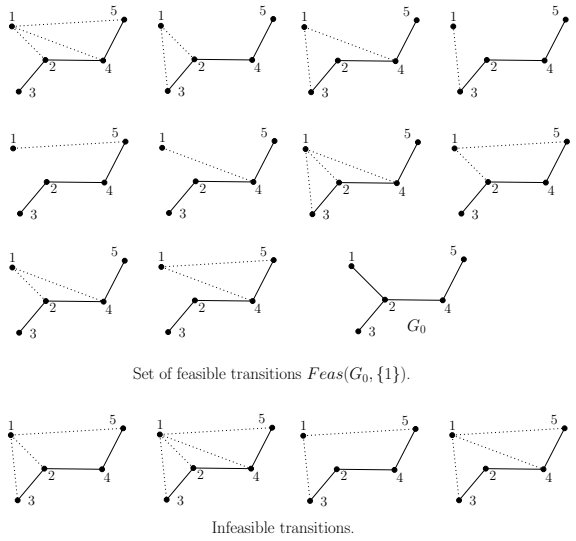

Fig. 1. Feasible and infeasible graphs by movement of node 1.

Now, we can use the S-procedure to evaluate whether each of the new graphs resulting from this enumeration is feasible. Since all nodes are fixed except for $\mathbf{x}_k = (x, y)$, the semi-algebraic set we need to check for non-feasibility is defined by $N-1$ polynomial inequalities over $\mathbb{R}[x, y]$. Each of these inequalities has either of the following two forms,

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & x_i \\ 0 & -1 & y_i \\ x_i & y_i & \delta^2 - x_i^2 - y_i^2 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \geq 0, \text{ if } e_{ik} \in \mathcal{E},$$

or

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_i \\ 0 & 1 & -y_i \\ -x_i & -y_i & x_i^2 + y_i^2 - \delta^2 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} > 0, \text{ if } e_{ik} \notin \mathcal{E}.$$

where $2 \leq i \leq N$, $\mathcal{E}$ is the edge set of the new graph and we denote $\mathbf{x}_i(t_j)$ by (x_i, y_i) for $i \neq k$. This computation can be repeated for all N nodes so that we have a choice of $N(2^{N-1} - 1)$ graphs. If we let $\rho_i = \delta^2 - x_i^2 - y_i^2$ then by denoting

$$A_i = \begin{bmatrix} -1 & 0 & x_i \\ 0 & -1 & y_i \\ x_i & y_i & \rho_i \end{bmatrix}, B_j = \begin{bmatrix} 1 & 0 & -x_j \\ 0 & 1 & -y_j \\ -x_j & -y_j & -\rho_j \end{bmatrix},$$

and ignoring the lossy aspect of the S-procedure [10], we need to solve the LMI,

$$-A_{\alpha_1} - \sum_{i \neq 1, e_{\alpha_i k} \in \mathcal{E}} \lambda_{\alpha_i} A_{\alpha_i} - \sum_{j, e_{\alpha_j k} \notin \mathcal{E}} \lambda_{\alpha_j} B_{\alpha_j} \geq 0.$$

An example of such a calculation is given in Figure 1, where $\mathcal{V}^0 = \{2, 3, 4, 5\}$ and $\mathcal{V}^m = \{1\}$. The LMI control toolbox [11] for MATLAB has been used to solve the LMI for each of these graphs in order to get the appropriate certificates.

We now give a detailed study of reachability in the context of connectivity graph processes as the main contribution of this paper.

IV. Reachability and Connectivity Graph Processes

Note that the set $\mathrm{Feas}(G_0, \mathcal{V}_0^m)$ does not depend on the actual movement of the individual nodes. In fact, even if $G \in \mathrm{Feas}(G_0, \mathcal{V}_0^m)$, it does not necessarily mean that there exists a trajectory by which $G_0 \to G$ or even that $G_0 \to G_1 \to G_2 \to \ldots \to G$. The geometrical configuration of nodes may create an obstruction in obtaining a graph process that takes G_0 to G. There are two ways by which this obstruction is created: The requirement to maintain connectivity and conformity to a fixed set of mobile nodes. We therefore need some notion of *reachability* on the space $\mathcal{G}_{N,\delta}$. We say that a connectivity graph G_f is *reachable* from an initial graph G_0 if there exists a connectivity graph process of finite length $G_0 \to G_1 \to \ldots \to G_f$ and a sequence of vertex-sets $\{\mathcal{V}_k^m\}$ such that each $G_{k+1} \in \mathrm{Feas}(\mathcal{G}_k, \mathcal{V}_k^m)$. If $\mathcal{V}_k^m = \mathcal{V}^m$ at each transition, then every $G_k \in \mathrm{Feas}(G_0, \mathcal{V}^m)$. (In particular, $G_f \in \mathrm{Feas}(G_0, \mathcal{V}^m)$.) Consider all such G's that are reachable from G_0 with a fixed $\mathcal{V}_m$. We will denote this set by $\mathrm{Reach}(G_0, \mathcal{V}^m)$. It is easy to see that $\mathrm{Reach}(G_0, \mathcal{V}^m) \subseteq \mathrm{Feas}(G_0, \mathcal{V}^m)$.

In the previous sub-section, it was shown how to determine the membership for the set $\mathrm{Feas}(G_0, \mathcal{V}^m)$. For all such graphs, determining whether they also belong $\mathrm{Reach}(G_0, \mathcal{V}^m)$ is not very straightforward, specially under the restriction that the intermediate graphs have to be connected. For the special case of a single mobile node, the situation is manageable as discussed below. We first describe what is called a *cylindrical algebraic decomposition* or CAD of a semi-algebraic set [12], [13].

Cylindrical Algebraic Decomposition: We first give some definitions. A *Nash manifold* $M \in \mathbb{R}^n$ is an analytic submanifold which is a semialgebraic set. Let $U \subset \mathbb{R}^n$. A *Nash function* $f : U \to \mathbb{R}$ is a smooth function for which there exists a polynomial $p(x,t) = p(x_1, \ldots x_n, t)$ such that $p(x, f(x)) = 0$ for all $x \in U$. A *Nash cell* in $\mathbb{R}^n$ is a Nash manifold which is diffeomorphic to an open box $(-1,1)^d$ of dimension d. Every semialgebraic set can be decomposed into a disjoint union of Nash cells. More precisely [13];

Theorem 4.1: Let $A_1, \cdots, A_q$ be semialgebraic subsets of $\mathbb{R}^n$. Then there exists a finite semialgebraic partition of $\mathbb{R}^n$ into Nash cells such that each A_j is a union of some of these cell.

The existence of such a decomposition is given by a technique in real algebraic geometry known as the cylindrical algebraic decomposition (CAD) [12]. A CAD of $\mathbb{R}^n$ is a partition into finitely many semialgebraic subsets. The CAD of $\mathbb{R}^n$ is given by induction on n as follows.

1) A CAD of $\mathbb{R}$ is a subdivision by finitely many points $a_1 < \ldots < a_l$. The cells are the singletons $\{a_i\}$ and the open intervals delimited by these points.
2) For $n > 1$, a CAD of $\mathbb{R}^n$ is given by a CAD of $\mathbb{R}^{n-1}$and for each cell C of $\mathbb{R}^{n-1}$, the Nash functions $\zeta_{C,1} < \cdots < \zeta_{C,l_C} : C \to \mathbb{R}$.

The cells of the CAD of $\mathbb{R}^n$ are the graphs of $\zeta_{C,j}$ and the bands in the cylinders $C \times \mathbb{R}$ determined by the graphs. Observe that the cells generated by CAD are Nash, thus providing a partition satisfying Theorem 4.1. For our need, we only need a CAD of $\mathbb{R}^2$. For notational convenience, let us denote the collection of Nash cells corresponding to a semialgebraic set S by $\mathrm{CAD}(S)$.

With $\mathcal{V}^m = \{k\}$ and $\mathcal{V}^0 = \{1, \cdots, k-1, k+1, \cdots, N\}$, consider the semialgebraic set

$$X(\mathcal{V}^0) = \bigcup_{j \in \mathcal{V}^0} \{(x,y) : (x - x_j)^2 + (y - y_j)^2 \leq \delta^2\} \subset \mathbb{R}^2. \tag{3}$$

The first thing to note about this set is that it is compact. This relieves us of any complicated compactification procedures that are needed to get the CAD of non-compact sets. Let C be a cell in $\mathrm{CAD}(X(\mathcal{V}^0))$, then we call $r \in C$ a *configuration* point in C if $r \neq \mathbf{x}_j$ for all $j \in \mathcal{V}^0$. We then have the following result.

Proposition 4.1: Let $C \in \mathrm{CAD}(X(\mathcal{V}^0))$. If p, q are any two configuration points in C then $\Phi_N((\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, p, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N)) = \Phi_N((\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, q, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N))$.

Proof: If C is a 0-cell we have the result trivially. We therefore assume that $\dim(C) > 0$. For notational convenience denote $(\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, p, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N)$ by $\mathbf{x}_p$ and the graph $\Phi_N((\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, p, \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N))$ by G_p for a point $p \in C$.

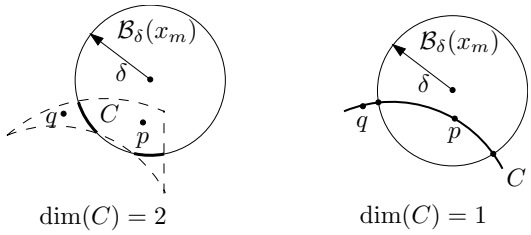

Fig. 2. Topological obstruction used in the proof of Proposition 4.1.

First note that any pair of positions $\mathbf{x}_i, \mathbf{x}_j$, where $i, j \in \mathcal{V}^0$ are always different. This is because they are the product of a valid configuration on the configuration space $\mathcal{C}^N(\mathbb{R}^2)$ described in Equation 2. If p is a configuration point then the N-tuple $\mathbf{x}_p = (\mathbf{x}_1, \cdots, \mathbf{x}_{k-1}, p, \mathbf{x}_{k+1}, \cdots, \mathbf{x}_N)$ is a valid configuration on $\mathcal{C}^N(\mathbb{R}^2)$, hence the name configuration point. The configuration point p would correspond to the k-th node in G_p. This shows that the mapping $\Phi_N(\mathbf{x}_p)$ is well defined for all configuration points in C. We now prove the proposition by contradiction.

Assume that there exist points $p, q \in C$ such that $G_p \neq G_q$. The graph G_p would differ from G_q in at least one edge incident on node k. Without loss of generality, assume that e_{km} is an edge between a node $m \in \mathcal{V}^0$ and the k-th node corresponding to p in G_p, and that there is no edge between nodes k and m in G_q. The existence of this edge in G_p means that p is inside the closed ball $\mathcal{B}_\delta(\mathbf{x}_m)$, while $q \notin \mathcal{B}_\delta(\mathbf{x}_m)$ as shown in Figure 2. From the definition of

a Nash cell, C is diffeomorphic to $(-1,1)^{\dim(C)}$ which is simply connected.[1] This means that C, in addition to being an open set, is also simply connected. Therefore the ball $\mathcal{B}_\delta(\mathbf{x}_m)$ induces a partition

$$C = (C \setminus \mathcal{B}_\delta(\mathbf{x}_m)) \cup (C \cap \mathcal{B}_\delta(\mathbf{x}_m)),$$

such that $\partial\mathcal{B}_\delta(\mathbf{x}_m)) \cap C \neq \emptyset$. From Figure 2, it is clear that each connected component of $\partial\mathcal{B}_\delta(\mathbf{x}_m)) \cap C$ induces $\dim(C) - 1$ Nash cells that are not already present in the CAD of $\mathrm{CAD}(X(\mathcal{V}^0))$. Moreover, this means that there exist lower dimensional Nash cells that are properly contained in C. Both implications are contrary to the CAD construction described above. Therefore, the connectedness of C creates a topological obstruction in getting $G_p \neq G_q$. This proves the Proposition. ∎

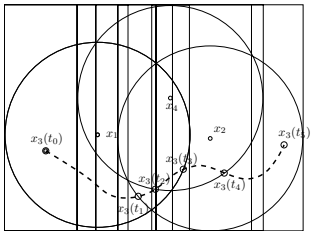

Fig. 3. Trajectory of a mobile node in the CAD generated by the fixed nodes.

From this we get the following useful result.

Corollary 4.1: Let $\mathbf{x}(t) = (\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \gamma(t), \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N)$ be a trajectory on $\mathcal{C}^N(\mathbb{R}^2)$. If $\gamma(t) \in C$ for all $t \in [t_0, t_f)$ then the connectivity graph process has no transitions while $t \in (t_0, t_f)$.

For a trajectory $\mathbf{x}(t) = (\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, \gamma(t), \mathbf{x}_{k+1}, \ldots, \mathbf{x}_N)$ on $\mathcal{C}^N(\mathbb{R}^2)$ that is not confined to a single Nash cell, the graph may change as the trajectories goes from one cell to another. Moreover, since the connectivity graph remains unchanged inside one cell, the transition must take place at the boundary of Nash cells. We can now begin to appreciate the connection between the CAD decomposition and the geometric origin of transitions in a connectivity graph process. We further observe that the trajectory is partitioned into a *finite* number of pieces, where each piece is confined to a single Nash cell. In more precise terms, for each graph process $G_0 \to G_1 \ldots \to G_N$ with $\mathcal{V}_i^m = \{k\}$ for $0 \leq i \leq N$ and transitions at $t_1 < t_2 < \ldots < t_N$, there exist a finite number of Nash cells $\mathcal{C}_\mathbf{x} = \{C_0, C_1, \ldots, C_M\} \subseteq \mathcal{C}$, where $M \geq N$, such that the trajectory $\mathbf{x}(t)$ intersects with a finite sub-collection $\mathcal{C}_{\mathbf{x},i} \subseteq \mathcal{C}_\mathbf{x}$ for $t \in (t_i, t_{i+1})$. One such trajectory is depicted in Figure 3.

We will construct an explicit planning algorithm to go from one point in the CAD to another. Let us define the k-th skeleton of the CAD of a set S as

$$\mathrm{CAD}^{(k)}(S) = \{C \in \mathrm{CAD}(S) \mid \dim(C) = k\}.$$

[1]Topologically, a simply connected set means path-wise connectedness and the absence of any *holes* in the set.

In particular, the 1-skeleton $\mathrm{CAD}^{(1)}(S)$ consists of all 1-dimensional Nash cells in $CAD(S)$. Recall that the set $X(\mathcal{V}^0)$ described by Equation 3 is a union of closed disks in $\mathbb{R}^2$. Moreover the boundary of of this set

$$\partial X(\mathcal{V}^0) \subseteq \mathrm{CAD}^{(0)}(X(\mathcal{V}^0)) \cup \mathrm{CAD}^{(1)}(X(\mathcal{V}^0)).$$

If $C^2 \in \mathrm{CAD}^{(2)}(S)$ and $\bar{C^2}$ denotes its closure in $\mathbb{R}^2$, then by construction of the CAD, $\partial\bar{C^2} \subseteq \mathrm{CAD}^{(0)}(X(\mathcal{V}^0)) \cup \mathrm{CAD}^{(1)}(X(\mathcal{V}^0))$. Therefore, any connected component of $X(\mathcal{V}^0)$ cannot be made up of cells in $\mathrm{CAD}^{(2)}(S)$ alone. In fact, we get the following useful lemma.

Lemma 4.1: The set $\mathrm{CAD}(X(\mathcal{V}^0))$ is connected if and only if $\mathrm{CAD}^{(0)}(X(\mathcal{V}^0)) \cup \mathrm{CAD}^{(1)}(X(\mathcal{V}^0))$ is connected.

The above lemma suggests the existence of a path planning algorithm for moving the mobile node between any two points of a connected component of $X(\mathcal{V}^0)$. Given p, q in the same connected component of $X(\mathcal{V}^0)$, we construct a path $\gamma : [0, T] \to X(\mathcal{V}^0)$ such that $\gamma(0) = p$ and $\gamma(T) = q$. The placement of both points p, q in the same connected component ensures that $T < \infty$. Also, each of the points p, q lie in their unique cells of $\mathrm{CAD}(X(\mathcal{V}^0))$, which we denote by C_p, C_q respectively. We build our algorithm from the following observations.

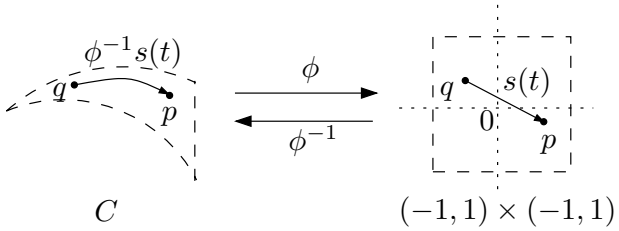

Fig. 4. Path between two points p, q in a 2-cell of the CAD.

Case 1: When $C_p = C_q = C$, i.e. when both points lie in the same cell, consider the diffeomorphism $\phi : C \to (-1,1)^{\dim(C)}$. Then the points $\phi(p)$ and $\phi(q)$ in the open cube can be joined by a straight line $s(t) = t\phi(p) + (1-t)\phi(q)$. Now map this line back to C by

$$\gamma(t) = \phi^{-1}(s(t)),\ t \in [0, 1].$$

This gives us the desired path. Such an explicit construction in terms of the diffeomorphism may only be needed when $\dim(C) = 2$, as depicted in Figure 4. For lower dimensions the construction of $\gamma(t)$ is rather obvious.

Case 2: When $C_p \neq C_q$. We study various situations in this case.

(a). When $\dim(C_p), \dim(C_q) < 2$. In this case one can construct the path γ explicitly by building a graph $\mathcal{G}'_P = (\mathcal{V}'_P, \mathcal{E}'_P)$ in the following way. Each $v_i \in \mathcal{V}'_p$ corresponds to a 0-cell C_i^0 in the $\mathrm{CAD}^{(0)}(X(\mathcal{V}^0))$ and an edge $e_{ij} \in \mathcal{E}'_p$ if and only if there exists a 1-cell $C^1 \in \mathrm{CAD}^{(1)}(X(\mathcal{V}^0))$ such that $C_i^0, C_j^0 \in \partial C^1$. Note that by construction of the CAD, if such a 1-cell exists, it will be unique. Moreover, each 1-cell will be mapped to a unique edge in $\mathcal{G}'_P$. If $\dim(C_p) = \dim(C_q) = 0$ we are done. If not, we modify $\mathcal{G}'_P$ as follows. We add one vertex for each of the points

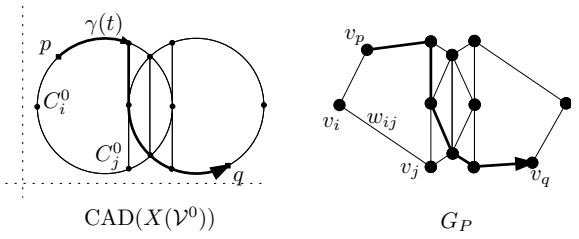

Fig. 5. Path planning via planning graph.

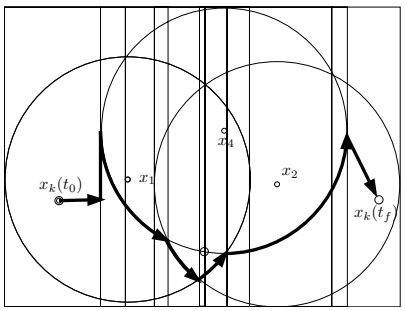

Fig. 6. Path between two points that belong to their respective 2-cells in the CAD.

p, q that belong to a 1-cell. Suppose $\dim(C_p) = 1$. Then C_p corresponds to an edge $e_{jk} = (v_j, v_k)$ induced by 0-cells C_j^0 and C_k^0. We add a vertex v_p corresponding to p. Now modify the graph $\mathcal{G}'_P$ by removing the edge e_{jk} and inserting two edges $e_{jp} = (v_j, v_p)$ and $e_{pk} = (v_p, v_k)$. A similar procedure modifies $\mathcal{G}'_P$ if $\dim(C_q) = 1$. We call this modified graph our planning graph G_P. An example of this construction is shown in Figure 5. We can now convert this graph into a *weighted* graph, by assigning each edge e_{ij} a real number w_{ij}. One choice of weights can be a constant weight on all edges. We will show later how to assign a more natural set of weights. Once we have that, we can use a standard discrete planning algorithm such as the *Dijkstra's algorithm* [14] to get a path on $\mathcal{G}_P$ that connects v_p to v_q by a sequence of edges and vertices. This graphical path can now be mapped back to $\mathbb{R}^2$ to get an explicit path $\gamma(t)$ that connects p to q. We now give this mapping.

Note first that a 1-cell C_j^1 has two 0-cells say $C_{j_0}^0$ and $C_{j_1}^0$ on its boundary. If this 1-cell is also a subset of $\partial\bar{\mathcal{B}}_\delta(\mathbf{x}_k)$ for some $k \in \mathcal{V}^0$ then by the properties of CAD construction, C_j^1 has a natural parametrization as a curve in $\mathbb{R}^2$ given by

$$c_j : s \mapsto (s, \mathrm{sgn}(C_j^1)\sqrt{\delta^2 - (s - y_k)^2} + x_k), \quad s \in (x_{j_0}, x_{j_1}),$$

where $\mathrm{sgn}(C_j^1)$ can be determined uniquely for each such 1-cell. If $w_{j_0 j_1}$ is the weight on the corresponding edge $e_{j_0 j_1} \in \mathcal{E}_P$, then we can re-parameterize by a linear mapping $T_j : \mathbb{R} \to \mathbb{R}$ that sends $[0, w_{j_0 j_1}) \mapsto [x_{j_0}, x_{j_1})$.We can therefore define a parameterized curve $\gamma_{j_0 j_1}(t) = c_j(T_j(t))$, where $t \in [0, w_{j_0 j_1})$.

In case the 1-cell C_k^1 is a vertical line segment, the parametrization is more simple. Simply, let $\gamma_{k_0 k_1}(t) = T_k(t)$ where $t \in [0, w_{k_0 k_1})$ and T_k maps $[0, w_{k_0 k_1}) \mapsto [y_{k_0}, y_{k_1})$. In this way we have a method to map back an edge in the planning graph to $\mathbb{R}^2$. We have omitted here details for edges that have either v_p or v_q as one of the vertices. It is easy to see that the mapping for these edges is very similar.

Now, given a path in $\mathcal{G}_P$ as a sequence of vertices and edges, one can build explicitly a path in $\mathbb{R}^2$ as a concatenation of the curves described above. More precisely let the path be given by a sequence $v_p, e_{pk_1}, v_{k_1}, e_{k_1 k_2}, v_{k_2}, \ldots, v_{k_M}, e_{k_M q}, v_q$, then we define a path that connects p to q by

$$\gamma(t) = \gamma_{pk_1} \circ \gamma_{k_1 k_2} \circ \ldots \circ \gamma_{k_M q}(t), \qquad t \in [0, W)$$

where $W = w_{pk_1} + w_{k_1 k_2} + \ldots w_{k_M q}$ and $\circ$ is the path concatenation operation defined by

$$\gamma_i \circ \gamma_j(t) = \begin{cases} \gamma_i(t), \ t \in [0, w_i); \\ \gamma_j(t), \ t \in [w_i, w_i + w_j). \end{cases}$$

(b). General Case: We now discuss the most general case. We present a strategy for the case when either or both of p and q belong to a 2-cell. Suppose p, q are inside $C_p^2, C_q^2 \in \mathrm{CAD}^{(2)}(X(\mathcal{V}^0))$ respectively. Now consider the line segment $\gamma_0(s) = sq + (1-s)p$, where $s \in [0, 1]$. By the construction of CAD, this line segment cuts at least 2 cells of dimension less than 2. Let s_1 be the minimum value of s at which the line cuts a lower dimensional cell. Also let s_2 be the corresponding maximum value. Now let $p' = \gamma_0(s_1)$ and $q' = \gamma_0(s_2)$. Using the procedure described in the previous case, a planning graph can be constructed to find a path that connects p' to q'. Concatenating the resulting curve with $\gamma_0(t)$ for $t \in [0, s_1)$ in the beginning and $\gamma_0(t)$ for $t \in [s_2, 1]$ at the end we get the resulting path. One such construction is shown in Figure 6.

The above discussion makes it clear that it possible to find a path that transports the k-th node at time t_0 from any initial position $x(t_0)$ to a desired position inside the same connected component of $\mathrm{CAD}(X(\mathcal{V}^0))$ in a finite time. In order to give a meaning to this transportation from the point of view of control, it enough to assume that the dynamics of the nodes described by Equation 1 are globally controllable. Combining these observations, we get our main result.

Theorem 4.2: Assume that the individual nodes are globally controllable. Let $\mathcal{V}_i^m = \{k\}$ be fixed. Given two connectivity graphs $\Phi_N(\mathbf{x}(0))$ and G_f, there exists a finite connectivity graph process $\Phi_N(\mathbf{x}(0)) \to G_1 \to \ldots \to G_f$, and a corresponding trajectory $\mathbf{x}(t) \subseteq C^N(\mathbb{R}^2), t \in [t_0, t_f]$ such that $\mathbf{x}(t_f) \in \Phi_N^{-1}(G_f)$ if and only if there exist a finite collection of Nash cells $\mathcal{C}_\mathbf{x} = \{C_0, C_1, \ldots, C_M\} \subseteq \mathrm{CAD}(X(\mathcal{V}^0))$ such that $\mathbf{x}(0) \in C_0$, $\mathbf{x}(t_f) \in C_M$ and $\mathbf{x}(t) \in C_j$ for some $C_j \in \mathcal{C}_\mathbf{x}$ for all $t \in [t_0, t_f]$

This gives us a way to realize trajectories on the space of connectivity graphs and a method to characterize the reachable set with one mobile node. In fact, by construction this node only needs two different typed of motions, namely along constant vector fields and rotations about fixed points.

V. Global Objectives, Desirable Transitions and Optimality

The purpose of a coordinated control strategy in a multi-agent system is to evolve towards the fulfilment of a global objective. This typically requires the minimization (or maximization) of a cost associated with each global configuration. Viewed in this way, a planning strategy should basically be a search process over the configuration space, evolving towards this optimum. If the global objective is fundamentally a function of the graphical abstraction of the formation, then it is better to perform this search over the space of graphs instead of the full configuration space of the system. By introducing various graphical abstractions in the context of connectivity graphs, we have the right machinery to perform this kind of planning. In other words, we will associate a cost or score with each connectivity graph and then work towards minimizing it.

Given $\mathrm{Reach}(G_0, \mathcal{V}^m)$, a decision need to be taken regarding what $G_f \in \mathrm{Reach}(G_0, \mathcal{V}^m)$ the system should switch to. For this we define a cost function $\Psi : G_{N,\delta} \to \mathbb{R}$ and we choose the transition through

$$G_f = \underset{G \in \mathrm{Reach}(G_0, \mathcal{V}^m)}{\arg\min} \Psi(G)$$

Here Ψ is analogous to a terminal cost in optimal control. If, in addition, we also take into account the cost associated with every transition in the graph process $G_0 \to G_1 \to \ldots \to G_M = G_f$ that takes us to G_f, then we would instead consider the minimization of the cost

$$J = \Psi(G_f) + \sum_{i=0}^{M-1} \beta(i) L(G_i, G_{i+1}),$$

where $L : \mathcal{G}_{N,\delta} \times \mathcal{G}_{N,\delta} \to \mathbb{R}$ is the analogue of a discrete Lagrangian, $\beta(i)$ are weighting constants, and $G_{i+1} \in \mathrm{Reach}(G_i, \mathcal{V}^m)$ at each step i. The choice of a Lagrangian lets us control the transient behavior of the system during the evolution of the graph process.

In principle, the framework described in this paper can be used for any application that required optimization over connectivity graphs. We have presented such several such applications in a recent work [9]. Here we reproduce one such simulation result in Figure 7, where a low-complexity formation called as a δ-chain [7] is achieved by a series of maneuvers in which different nodes take the role of the mobile node at appropriate steps.

VI. Conclusions

We have presented a generic framework for connectivity graph processes. The concepts of feasibility and reachability are useful for obtaining optimal trajectories on the space of connectivity graphs. These graphical abstractions are computable using the techniques of semi-definite programming and CAD, as verified by simulation results.

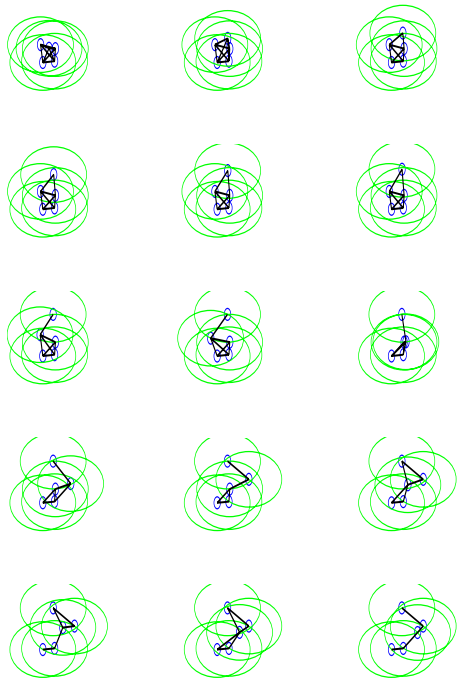

Fig. 7. Snapshots of a connectivity graph process that generates a δ-chain by choosing different mobile nodes at appropriate intermediate transitions.

References

[1] R.Saber and R. Murray, "Agreement Problems in Networks with Directed Graphs and Switching Topology," in *Proc. IEEE Conference on Decision and Control*, 2003.

[2] A. Jadbabaie, J. Lin, and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, Vol. 48, No. 6, pp. 988-1001, 2003.

[3] M. Mesbahi, "On State-Dependent Dynamic Graphs and their Controllability Properties," *IEEE Conference on Decision and Control*, 2004

[4] E. Klavins, R. Ghrist, and D. Lipsky, "A Grammatical Approach to Self-Organizing Robotic Systems," *IEEE Transactions on Automatic Control*, 2005. (To appear)

[5] Z. Lin, B. Francis, and M. Maggiore, "Necessary and sufficient graphical conditions for formation control of unicycles," *IEEE Transactions on Automatic Control*, 2005.

[6] A. Muhammad and M. Egerstedt, "Connectivity Graphs as Models of Local Interactions." *Journal of Applied Mathematics and Computation*, Vol. 168, No. 1, pp. 243-269, Sept. 2005.

[7] A. Muhammad and M. Egerstedt, "On the Structural Complexity of Multi-Agent Agent Formations," in *Proc. American Control Conference*, Boston, Massachusetts, USA, 2004.

[8] A. Muhammad and M. Egerstedt, Positivstellensatz Certificates for Non-Feasibility of Connectivity Graphs in Multi-agent Coordination, 16th IFAC World Congress, Prague, July 4-8, 2005.

[9] A. Muhammad and M. Egerstedt, "Applications of Connectivity Graph Processes in Networked Sensing and Control," Workshop on Networked Embedded Sensing and Control, University of Notre Dame, 2005.

[10] S. Boyd, L. El Ghaoui, E. Feron and V. Balakrishnan, *Linear Martix Inequalities in Systems and Control Theory*, SIAM Studies in Applied Mathematics, 1994.

[11] The Math- Works Inc., "LMI Control Toolbox," Version 1.0.7, May 2001.

[12] S. Basu, R. Pollack and M. Roy, *Algorithms in Real Algebraic Geometry*, Algorithms and Computation in Mathematics Series, Vol. 10, Springer, 2003.

[13] J. Bochnak, M. Coste, M. Roy, *Real Algebraic Geometry,* Springer-Verlag, Berlin, 1998.

[14] E. Dijkstra, "A Note on Two Problems in Connexion With Graphs," in *Numerische Mathematik*, vol. 1, pp. 269271, 1959.