ELEMENTARY OPERATIONS FOR THE REORGANIZATION OF MINIMALLY PERSISTENT FORMATIONS

JULIEN M. HENDRICKX , BARIŞ FIDAN, CHANGBIN YU,
BRIAN D.O. ANDERSON AND VINCENT D. BLONDEL

Abstract. In this paper we study the construction and transformation of two-dimensional minimally persistent graphs. Persistence is a generalization to directed graphs of the undirected notion of rigidity. In the context of moving autonomous agent formations, persistence characterizes the efficacy of a directed structure of unilateral distances constraints seeking to preserve a formation shape. Analogously to the powerful results about Henneberg sequences in minimal rigidity theory, we propose different types of directed graph operations allowing one to sequentially build any minimally persistent graph (i.e. persistent graph with a minimal number of edges for a given number of vertices), each intermediate graph being also minimally persistent. We also consider the more generic problem of obtaining one minimally persistent graph from another, which corresponds to the on-line reorganization of an autonomous agent formation. We show that we can obtain any minimally persistent formation from any other one by a sequence of elementary local operations such that minimal persistence is preserved throughout the reorganization process.

1. Introduction

The recent progress in the field of autonomous agent systems has led to new problems in control theory [2,3] and graph theory [4,6,11]. By autonomous agent, we mean here any human controlled or unmanned vehicle that can move by itself and has a local intelligence or computing capacity, such as ground robots, air vehicles or underwater vehicles. The results derived in this paper concern mostly autonomous agents evolving in a two dimensional space.

Many applications require some inter-agent distances to be kept constant during a continuous move in order to preserve the shape of a multi-agent formation. In other words, some inter-agent distances are explicitly maintained constant so that all the inter-agent distances remain constant. The information structure arising from such a system can be efficiently modelled by a graph, where agents are abstracted by vertices and actively constrained inter-agent distances by edges. We assume here that those constraints are unilateral: the responsibility for maintaining a distance is not shared by the two concerned agents but relies on only one of them while the

J. M. Hendrickx and V. Blondel are with Department of Mathematical Engineering, Université catholique de Louvain, Avenue Georges Lemaitre 4, B-1348 Louvain-la-Neuve, Belgium; hendrickx,blondel@inma.ucl.ac.be. Their work is supported by the Belgian Programme on Interuniversity Attraction Poles initiated by the Belgian Federal Science Policy Office, and The Concerted Research Action (ARC) "Large Graphs and Networks" of the French Community of Belgium. The scientific responsibility rests with its authors. Julien Hendrickx holds a FNRS fellowship (Belgian Fund for Scientific Research). This work was also supported in part by DoD AFOSR URI for "Architectures for Secure and Robust Distributed Infrastructures", F49620-01-1-0365 (led by Stanford University).

B. Fidan, C. Yu and B. Anderson are with Australian National University and National ICT Australia , 216 Northbourne Ave, Canberra ACT 2601 Australia ; baris.fidan,brad.yu,brian.anderson@nicta.com.au. Their work is supported by an Australian Research Council Discovery Project Grant and by National ICT Australia, which is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through the Backing Australia's Ability Initiative. Changbin Yu is an Australia-Asia Scholar supported by the Australian Governments Department of Education, Science and Training through Endeavours Programs.

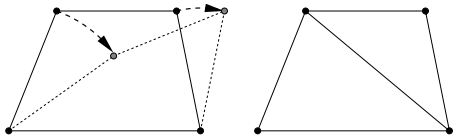

FIGURE 1. Representation of (a) a non-rigid and (b) a rigid graph/formation. The solid structure in (a) can indeed be deformed to the dotted structure without breaking any distance constraint, while this cannot be done in (b)

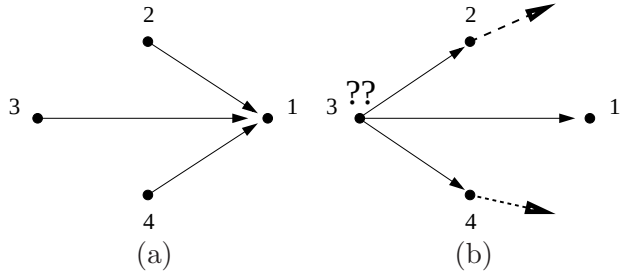

FIGURE 2. Representation of (a) a constraint consistent and (b) a non-constraint consistent (in 2 dimension) graph/formation. In (b), for almost any uncoordinated continuous displacement of the agents 2 and 4 (which are unconstrained), the agent 3 is indeed unable to move in such a way that it maintains its distances to all of 1, 2 and 4 constant. However, such a situation could not happen in graph (a).

other one is unaware of it. This asymmetry is modelled by the use of directed edges in the graph. The characterization of the directed information structures which can efficiently maintain the formation shape has begun to be studied under the name of "directed rigidity" [2, 4]. These works included several conjectures about minimal directed rigidity, i.e., directed rigidity with a minimal number of edges for a fixed number of vertices. In [6], Hendrickx et al. proposed a theoretical framework to analyze these issues, where the name of "persistence" was advanced in preference to "directed rigidity", since the latter notion does not correspond to the immediate transposition of the undirected notion of rigidity to directed graph. The intuitive definition of persistence is the following: An information structure is persistent if, provided that each agent is trying to satisfy all the distance constraints for which it is responsible, every inter-agent distance remains constant and as a result the formation shape is preserved. It is shown in [6] that persistence is actually the conjunction of two distinct notions: rigidity of the underlying undirected graph (undirected graph obtained by ignoring the direction of the edges), and constraint consistence. Intuitively, rigidity means that, provided that all the prescribed distance constraints are satisfied during a continuous displacement, all the inter-agent distances remain constant, as shown in Figure 1. Constraint consistence of an information structure means that, provided that each agent is trying to satisfy all its distances constraints, all the agents actually succeed in doing so. In other words, no agent has an impossible task, as shown in the example in Figure 2. Observe that this last notion depends strongly on the directed structure of the graph, while rigidity only relies on its underlying undirected graph. An example of persistent graph is provided in Figure 3. The persistence of a graph can be checked by checking the rigidity of several subgraphs [6, 14], which for agents evolving in a two-dimensional space can be done in a purely combinatorial way.

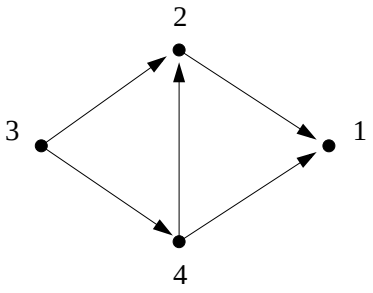

FIGURE 3. Representation of a persistent graph or equivalently a rigid and constraint consistent graph.

In this paper, we focus on minimally persistent graphs, that are persistent graphs having a minimal number of edges (for a given number of vertices), and their connections with minimally rigid graphs. More particularly *we analyze different ways to sequentially build minimally persistent graphs, analogously to the Henneberg sequences for the minimally rigid graphs* [9,12]. It has indeed long been known that every minimally rigid graph can be obtained from the complete graph on two vertices by a sequence of two basic operations, as detailed in Section 2. The natural extension of these operations to directed graphs [4] does *not* allow one to build all minimally persistent graphs, as remarked in [6] and reviewed later in Section 2. We consider here different possible additional operations that would help to achieve this purpose. We also consider the more generic problem of obtaining one persistent graph from another. From an autonomous agent point of view, this corresponds to an on-line reorganization of the agent formation such that the persistence is preserved throughout the process. The subsequent analysis leads us then to the definition of different "distances" between persistent graphs (the distance between two graphs being the smallest number of operations needed to obtain one from the other). Note that although the notion of persistence has been also defined in three or higher dimensions [8,13,14], the present analysis only concerns two-dimensional persistence, i.e., the persistence of graphs representing the information structure of a formation evolving in a two-dimensional space. Extension to the three dimensional case may be difficult; even for undirected graphs, Henneberg sequence theory is effectively incomplete. A fuller version of this work containing proofs (omitted here) and extended explanations of the results announced here is available in preprint form from the authors [7].

In Section 2, we review the main properties of minimally rigid and minimally persistent graphs. We present the two basic undirected operations - vertex addition and edge splitting - involved in the Henneberg sequences, together with their natural extension to directed graphs. We show that although these directed operations preserve minimal persistence, they are not sufficient to build all minimally persistent graphs. This analysis is done by reasoning on reverse construction of persistent graphs using reverse operations. In Section 3, we show how the goal of building all minimally persistent graph can be reached by introducing a third local directed operation - edge reversal. We see that, unlike when building minimally rigid undirected graphs with Henneberg sequences, the required number of operations is not uniquely determined by the size of the graph. We prove via an example in Section 4 that to avoid this drawback (still using operations analog to those involved in the undirected Henneberg sequences), one must use operations involving certain non-confined transformations, i.e., operations reversing the directions of (possibly several) edges that are not affected by the corresponding operation for undirected graphs. We however provide such a set of four operations, and analyze the relations between this set and the set of three operations treated in Section 3. Finally, this paper ends with the concluding remarks of Section 5.

2. Directed and undirected Henneberg sequences

2.1. **Minimally rigid graphs.** All graphs in this section are considered as undirected, but in the rest of this paper, they are always assumed to be directed. And, although the definitions and results of this section are given for undirected graphs, they can also be applied to directed graphs. Note that if G is a directed graph, we call the *underlying undirected graph* of G the undirected graph obtained by ignoring the directions of the edges of G.

The rigidity of a graph has the following intuitive meaning: Suppose that each vertex represents an agent in a formation, and each edge represents an inter-agent distance constraint enforced by an external observer. The graph is rigid if for almost every such structure, the only possible continuous moves are those which preserve every inter-agent distance. This notion also represents the rigidity of a framework where the vertices correspond to joints and the edges to bars. For a more formal definition, the reader is referred to [6, 12]. In $\Re^2$, a combinatorial criterion was provided by Laman to check if a given graph is rigid (Laman's theorem [6,12]). A *minimally rigid* graph is a rigid graph such that no single edge can be removed without losing rigidity, or provably equivalently, a rigid graph with the minimal number of edges for its number of vertices. From Laman's Theorem, it is possible to deduce a criterion to decide if a given graph is minimally rigid:

Proposition 1. *A graph $G = (V, E)$ ($|V| > 1$) is minimally rigid if and only if $|E| = 2|V| - 3$ and for all $E'' \subseteq E, E'' \neq \varnothing$, thereholds $|E''| \leq 2|V(E'')| - 3$.*

We say that a pair of unconnected vertices defines an *implicit edge* in a graph $G = (V, E)$ if their connection would create a subgraph $G' = (V', E')$ with $|E'| > 2|V'| - 3$. Intuitively, this means that the addition of such an edge would not improve the rigidity of the graph, i.e., the constraint that this edge would enforce is a linear combination of already present constraints. One can prove that two unconnected vertices define an implicit edge in a graph if and only if there is a minimally rigid subgraph containing both of them (note that this result cannot be extended to dimensions higher than 2). By extension, we sometimes call an edge of a graph an *explicit edge*. In a (minimally) rigid graph, every pair of vertices is connected by either an explicit or implicit edge. But, if one removes an (explicit) edge in a minimally rigid graph, the corresponding pair of vertices never defines an implicit edge in the graph obtained.

Let j, k be two distinct vertices of a minimally rigid graph $G = (V, E)$. A *vertex addition* operation consists in adding a vertex i, and connecting it to j and k, as shown in Figure 4(a). One can see using Proposition 1 that this operation preserves minimal rigidity. Moreover, if a vertex i has degree 2 in a minimally rigid graph, one can always perform the inverse vertex addition operation by removing it (and its incident edges) and obtain a smaller minimally rigid graph.

Let j, k, l be three vertices of a minimally rigid graph such that there is an edge between j and k. An *edge splitting* operation consists in removing this edge, adding a vertex i and connecting it to j, k and l, as shown in Figure 4(b). This operation provably preserves minimal rigidity [12]. The reverse operation is less straightforward than the reverse vertex addition operation. Given a vertex i connected to j, k and l, the minimal rigidity of the graph is preserved if one removes i and adds one edge among (j, k), (k, l) and (l, j). However, one cannot always freely choose any one of these edges to add. One has indeed to make sure that the added edge does not already belong to the graph, and also that its addition does not create a subgraph $G' = (V', E')$ with $|E'| > 2|V'| - 3$, i.e., that the pair of vertices does not define an implicit edge in the graph obtained after deletion of i. Figure 5 shows an example of such an unfortunate added edge selection. Suppose indeed that the vertex 5 is removed from the minimally rigid graph 5(a). The pair $(1, 4)$ does provably not define an implicit edge, and its addition leads thus to a

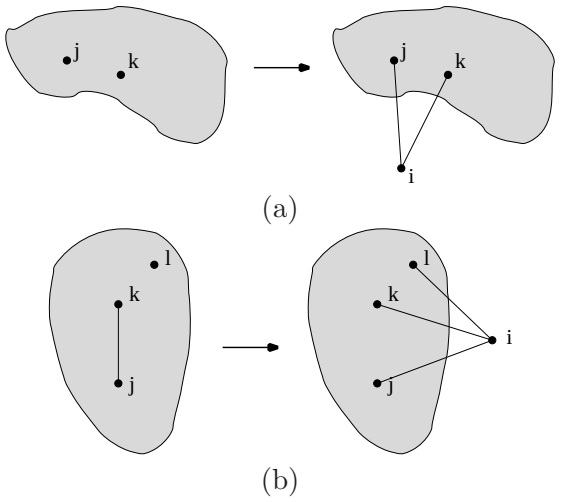

FIGURE 4. Representation of (a) undirected vertex addition operation and (b) edge splitting operation.

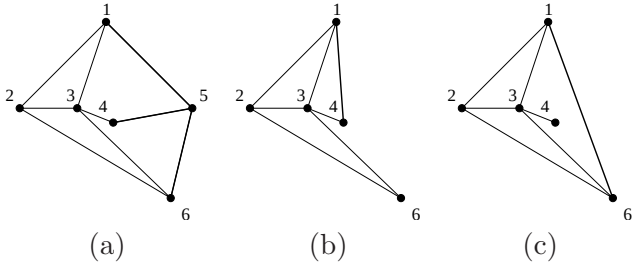

FIGURE 5. Example of unfortunate added edge selection in reverse edge splitting. After the removal of the vertex 5 from the minimally rigid graph (a), minimal rigidity can be preserved by the addition of the edge $(1, 4)$ but not of $(1, 6)$, as shown respectively on (b) and (c). The pair $(1, 6)$ defines an implicit edge in the minimally rigid subgraph induced by 1, 2, 3 and 6.

minimally rigid graph, which is represented in Figure 5(b). However, if $(1, 6)$ is added instead of $(1, 4)$, the graph obtained contains a subgraph $G' = (V', E')$ with $V' = \{1, 2, 3, 6\}$ such that $6 = |E'| > 2\,|V'| - 3 = 5$, as shown in Figure 5(c). The pair $(1, 6)$ defines thus an implicit edge. It is possible to prove that at least one among the three possible pairs of vertices does not define an actual nor an implicit edge [10, 12]. One can thus always perform a reverse edge splitting on any vertex with a degree 3.

A *Henneberg sequence* is a sequence of graphs $G_2, G_3, \ldots, G_{|V|}$ with G_2 being the complete graph on two vertices K_2 and each graph G_i $(i \geq 3)$ can be obtained from G_{i-1} by either a vertex addition operation or an edge splitting operation. Since these operations preserve minimal rigidity and since K_2 is minimally rigid, every graph in such a sequence is minimally rigid.

A simple degree counting argument shows that every minimally rigid graph $G_{|V|} = (V, E)$ with more than 2 vertices contains at least one vertex with degree 2 or 3. One can thus always perform either a reverse vertex addition or a reverse edge splitting operation and obtain a smaller minimally rigid graph $G_{|V|-1}$. Doing this recursively, one eventually obtains a minimally rigid graph on two vertices, which can only be K_2. It is straightforward to see that the sequence

$K_2 = G_2, G_3, \dots G_{|V|}$ is then a Henneberg sequence. We have thus proved the following result [12]:

Theorem 1. *Every minimally rigid graph on more than one vertex can be obtained as the result of a Henneberg sequence.*

2.2. **Minimally persistent graphs.** Consider a group of autonomous agents represented by vertices of a graph. To each of these agents, one assigns a (possibly empty) set of unilateral distance constraints represented by directed edges: the notation (i, j) for a directed edge connotes that the agent i has to maintain its distance to j constant during any continuous move. The persistence of the directed graph means that for almost any set of initial agent positions, provided that each agent is trying to satisfy its constraints, the distance between any pair of connected or non-connected agents is maintained constant during any continuous move, and as a consequence the shape of the formation is preserved. A formal definition of persistence is given in [6].

In a two-dimensional space, an agent having only one distance constraint to satisfy can move on a circle centered on its neighbor, and has thus one degree of freedom. Similarly, an agent having no distance constraint to satisfy can move freely in the plane and has thus two degrees of freedom. We call the *number of degrees of freedom* of a vertex i the (generic) dimension of the set in which the corresponding agent can chose its position (all other agents being fixed). It represents thus in some sense the decision power of this agent. The number of degrees of freedom of a vertex i is given by $\max(0, 2 - d^+(i))$ (where $d^+(i)$ and $d^-(i)$ represent respectively the in- and out-degree of the vertex i).

A graph is *minimally persistent* if it is persistent and if no single edge can be removed without losing persistence, or provably equivalently if it is persistent with the minimal number of edges for its number of vertices. The following result provides a swift criterion to decide minimal persistence :

Proposition 2. *[6] A graph is minimally persistent if and only if it is minimally rigid and no vertex has an out-degree larger than 2.*

As a consequence of Proposition 2, the number of degrees of freedom of a vertex i in a minimally persistent graph is $2 - d^+(i)$. By Proposition 1, it follows after summation on all vertices that the total number of degrees of freedom present in such a graph is always 3. This result is consistent with the intuition; there are indeed three degree of freedom to chose the position and orientation of a rigid body in a 2-dimensional space.

Let j, k be two distinct vertices of a minimally persistent graph $G = (V, E)$. A *directed vertex addition* [4, 5] consists in adding a vertex i and two directed edges (i, j) and (i, k) as shown in Figure 6(a). Since it is a vertex addition operation, it preserves minimal rigidity. Besides, the added vertex has an out-degree 2 and the out-degree of the already existing vertices are unchanged. By Proposition 2, the directed vertex addition thus preserves the minimal persistence. Moreover, if a vertex has an out-degree 2 and an in-degree 0 in a minimally persistent graph, one can always perform a reverse (directed) vertex addition by removing it, and obtain a smaller minimally persistent graph.

Let (j, k) be a directed edge in a minimally persistent graph and l a distinct vertex. A *directed edge splitting* [4, 5] consists in adding a vertex i, an edge (i, l), and replacing the edge (j, k) by (j, i) and (i, k), as shown in Figure 6(b). Again, this operation preserves minimal rigidity since it is an edge splitting operation from an undirected point of view, and since the added vertex has an out-degree 2 and the out-degree of the already existing vertices are unchanged, it also

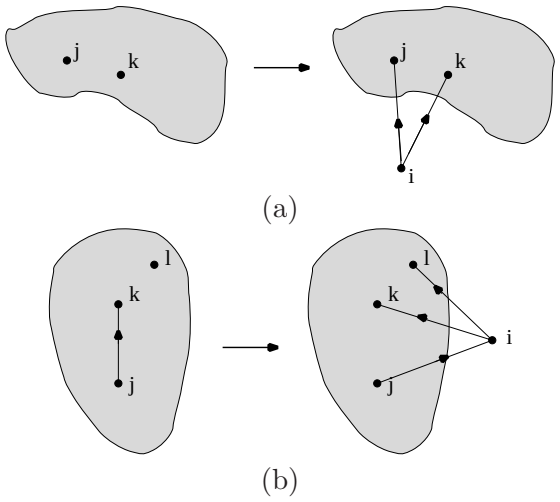

FIGURE 6. Representation of (a) the directed vertex addition and (b) edge splitting.

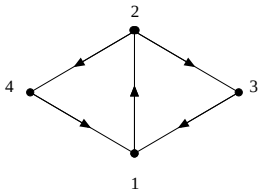

FIGURE 7. Example of situation where a vertex has an out-degree 2 and an in-degree 1 but where a reverse directed edge splitting cannot be performed.

preserves minimal persistence. But, unlike in the case of directed vertex addition, *the reverse operation cannot always be performed*. Suppose indeed that we have a vertex i with out-degree 2 and in-degree 1, and call its neighbors j, k and l. The reverse operation consists in removing i and its incident edges, and adding either (j, k) or (j, l) (note that k and l are interchangeable). Adding any other edge such as (k, l) or (l, k) would indeed prevent the operation from being out-degree preserving, and one could then not guarantee the minimal persistence of the graph obtained (by Theorem 2). But, it can happen that both pairs (j, l) and (j, k) are already connected by explicit or implicit edges. In such a case, minimal rigidity is only preserved by addition of an edge between k and l, which as explained above may not preserve persistence. See Figure 7 for a simple example.

It is shown in [6] and in the example of Figure 7 that there exists some minimally persistent graphs on which none of the reverse version of the above defined directed operations can be performed. They can thus not be obtained by performing a directed vertex addition or edge splitting on a smaller minimally persistent graph. It is possible to prove the existence of infinite classes of such graphs (see [7]). Unlike in the case of minimally rigid graphs, one can therefore not build every minimally persistent graph by performing a sequence of directed vertex additions or edge splitting operations on some seed graph taken in a finite set. However less powerful results are available. Consider indeed a minimally rigid graph G and the sequence of undirected operations allowing one to obtain it starting from K_2. Applying the same sequence of the directed version of these operations on an initial graph of two vertices connected by a directed edge (which is provably minimally persistent) yields to a minimally persistent graph having G as underlying undirected graph. We have thus the following proposition.

Proposition 3. *It is possible to assign directions to the edges of any minimally rigid graph such that the obtained directed graph is minimally persistent and can be obtained by performing a sequence of vertex additions and edge splittings on an initial graph of two vertices connected by one directed edge (called a "leader-follower seed").*

In the following sections, we examine different possibilities of additional operations that allow the construction of all minimally persistent graphs. In order to avoid confusion, we shall sometimes refer to the directed version of vertex addition and edge splitting as *standard vertex addition* and *standard edge splitting*. We denote by $\mathcal{S}$ the set consisting of these two operations and $\mathcal{S}^{-1}$ the one consisting of their inverses (the same convention is used in the sequel for all operations sets). Note that it is always possible to perform an operation of $\mathcal{S}$ on a minimally persistent graph, but we have seen that this is not true for operations of $\mathcal{S}^{-1}$.

3. A purely directed operation

3.1. **Edge, path and cycle reversal.** We introduce a third persistence-preserving operation: the *edge reversal*. Unlike those of $\mathcal{S}$, it does not affect the underlying undirected graph. We then define two macro-operations which allow us to show in Section 3.2 how to obtain any minimally persistent graph from any other one having the same underlying undirected graph by a sequence edge reversal.

Let (i, j) be an edge such that j as at least one degree of freedom, i.e., $d^+(j) = 0$ or $d^+(j) = 1$. The *edge reversal* operation consists in replacing the edge (i, j) by (j, i). As a consequence, one degree of freedom is transferred from j to i. This operation is its auto-inverse and preserves minimal persistence since it does not affect the underlying undirected graph and the only increased out-degree $d^+(j)$ remains no greater than 2. From an autonomous agent point of view j transfers its decision power (or a part of it) to i.

Given a directed path P between a vertex i and a vertex j such that j has a positive number of degrees of freedom, a *path reversal* consists in reversing the direction of every edge of P. As a result, j loses a degree of freedom, i acquires one, and there is a directed path from j to i. Moreover, the number of degrees of freedom of all other vertices remain unchanged. Note that i and j can be the same vertex, in which case the path either has a trivial length 0 or is a cycle. In both of these situations, the number of degrees of freedom is preserved for every vertex. As shown in Figure 8, the path reversal can easily be implemented with a sequence of edge reversals.

Let us now suppose that one wants to transfer a degree of freedom from some vertex j to some vertex i which has at most one degree of freedom (transferring a degree of freedom to a vertex that has already two degrees of freedom would indeed be impossible as there is no edge inwardly incident). The following lemma (which is a particular case of a result available in [8, 14]) guarantees the existence of a directed path from i to j:

Lemma 1. *Let G be a minimally persistent graph, i and j two vertices of G with $d^+(i) \geq 1$ and $d^+(j) \leq 1$. Then, there is a directed path from i to j.*

The transfer can thus be done by reversing this path, which leaves the positions of all other degrees of freedom unchanged. By doing this at most three times, one can reallocate the three degrees of freedom onto any chosen vertices. As a consequence, we have the following result:

Proposition 4. *Let G_A and G_B be two minimally persistent graphs having the same underlying undirected graph. By applying a sequence of at most three path reversals on G_A, it is possible to obtain a minimally persistent graph G'_A in which every vertex has the same number of degrees of freedom as in G_B*

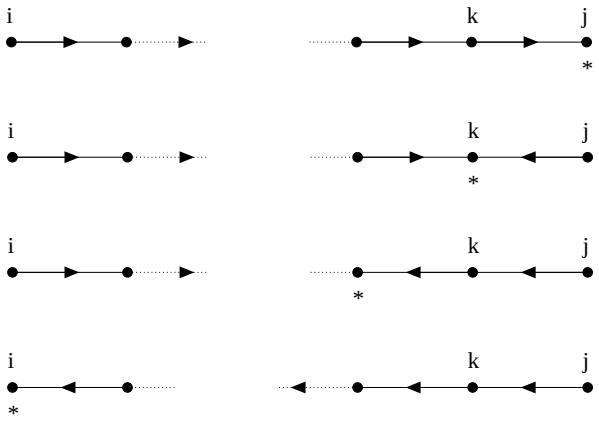

FIGURE 8. Implementation of the path reversal by a sequence of edge reversals. The symbol "*" represents one degree of freedom. The edge reversal can only be performed on an edge inwardly incident to a vertex with at least one degree of freedom.

A *cycle reversal* consists in reversing all the edges of a directed cycle. Note that this operation does not affect the number of degrees of freedom of any vertex nor the underlying undirected graph. It preserves therefore minimal persistence. Moreover, as proved in [7], it can be implemented by a sequence of (persistence-preserving) edge reversals even when no vertex in the cycle has a positive number of degrees of freedom and when no edge of the cycle can then be immediately reversed without losing persistence.

Let us now suppose that G_A and G_B are two minimally persistent graphs having the same underlying undirected graph and degree of freedom allocation (and thus out-degree allocation). If they are not identical, the following lemma guarantees the existence of a cycle of edges that do not have the same direction in G_A as in G_B:

Lemma 2. *Let $G_A = (V, E_A)$ and $G_B = (V, E_B)$ be two graphs having the same underlying undirected graph and such that every vertex has the same out-degree in both graphs. If an edge of G_A has the opposite direction to that in G_B, it belongs to (at least) one cycle of such edges in G_A.*

By reversing this cycle in G_A, one obtains a minimally persistent graph still having having the same underlying undirected graph and degree of freedom allocation as G_B, and having less edges that direction is not the same as in G_B. Doing this recursively, one finally obtains G_B. We have thus argued the following result (formally proved in [7]):

Proposition 5. *Let $G_A = (V, E_A)$ and $G_B = (V, E_B)$ be two minimally persistent graphs having the same underlying undirected graph and such that every vertex has the same number of degrees of freedom in both of them. Then it is possible to obtain G_B from G_A by a sequence of at most $|E_A|/3 = |E_B|/3$ cycle reversals.*

3.2. **Obtaining all minimally persistent graphs using three primitive operations.** Combining Propositions 4 and 5 with the fact that both path reversal and cycle reversal can be implemented by sequences of (persistence preserving) edge reversals, one can prove the following result:

Proposition 6. *By applying a sequence of edge reversals to a given minimally persistent graph, it is possible to obtain any other minimally persistent graph having the same underlying undirected graph. Moreover, every intermediate graph is then minimally persistent.*

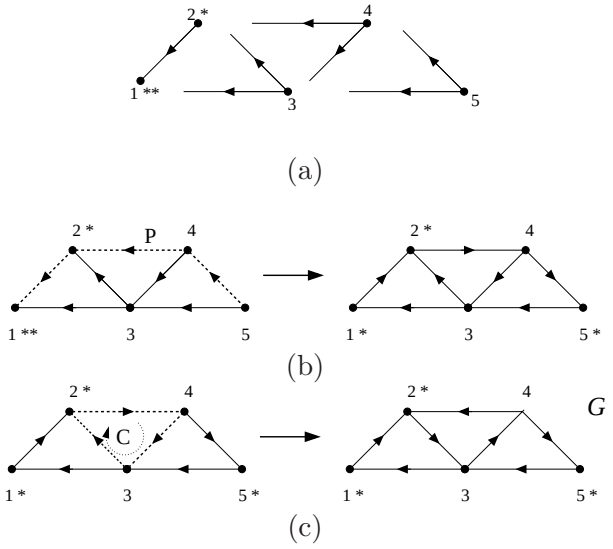

FIGURE 9. Example of obtaining of a minimally persistent graph by applying a sequence of operations of $\mathcal{T}$ on a leader-follower seed. The graph G is obtained from the leader-follower seed by (a) three vertex additions, (b) the reversal of the path P and (c) of the cycle C.

Let $\mathcal{T}$ be the set of operations containing vertex addition, edge splitting, and edge reversal. A *leader-follower seed* is a minimally persistent graph on two vertices. It contains only one edge, leaving a vertex called "the first follower", and arriving at the other one, called "the leader". Consider a minimally persistent (and therefore minimally rigid) graph G. By Proposition 3, there exists a (possibly different) minimally persistent graph having the same underlying undirected graph that can be obtained by performing a sequence of operations of $\mathcal{S} \subset \mathcal{T}$ on an initial leader-follower seed. By Proposition 6, G can then be obtained by applying a sequence of edge reversals on this last graph. And, since every operations of $\mathcal{T}$ preserves minimal persistence, every intermediate graph is minimally persistent. We have thus proved the following theorem:

Theorem 2. *Every minimally persistent graph can be obtained by applying a sequence of operations of $\mathcal{T}$ to an initial leader-follower seed. Moreover, every intermediate graph is minimally persistent.*

As an illustration of Theorem 2, consider the graph G represented in the right hand side of Figure 9(c), which cannot be obtained by applying an operation of $\mathcal{S}$ on a smaller minimally persistent graph, as argued in [7]. By Theorem 2, it can be obtained by applying a sequence of operations of $\mathcal{T}$ on an initial leader-follower seed. Let us take 1 and 2 as respectively leader and first follower of this initial seed. One can begin by adding 3, 4 and 5 using three vertex additions as shown in Figure 9(a). The graph obtained has the same underlying undirected graph as G, but the degrees of freedom are not allocated to the same vertices. By reversing the path P ($\{5, 4, 2, 1\}$) (using a sequence of edge reversals), one can then transfer one degree of freedom from 1 to 5 as shown in Figure 9(b) such that in the obtained graph, all vertices have the same number of degrees of freedom as in G. As stated in Proposition 5, any edge of this graph that does not have the same direction as in G belongs to a cycle of such edges. The only such cycle here is C. By reversing it (using a sequence of edge reversals), one finally obtains the graph G, as shown in Figure 9(c). Note that consistently with Theorem 2, every intermediate graph is minimally persistent.

Theorem 2 also proves that it is always possible to obtain a leader-follower pair from any minimally persistent graph by applying an appropriate sequence of operations of $\mathcal{T}^{-1}$. Starting from a minimally persistent graph, one can thus first use operations of $\mathcal{T}^{-1}$ to obtain a leader-follower pair, and then use operations of $\mathcal{T}$ to obtain any other minimally persistent graph. This method is generally not optimal in terms of the number of operations. However, the argument proves the following corollary.

Corollary 1. *Every minimally persistent graph can be transformed into any other minimally persistent graph using only operations of* $\mathcal{T} \cup \mathcal{T}^{-1}$.

4. An alternative set of four primitive operations

As explained in Section 3, every minimally persistent graph can be obtained by applying a sequence of operations belonging to $\mathcal{T}$ on an initial leader-follower seed, in such a way that all the intermediate graphs are minimally persistent. However, unlike in the case of an undirected Henneberg sequence (see Section 2), the number of vertices in the final graph does not determine uniquely the required number of intermediate graphs, but only an upper bound on it (see Section 3). In this section, we focus on sets of operations equivalent to those of $\mathcal{S}$ from an undirected point of view and that allow one to build all minimally persistent graphs (the number of intermediate graphs being thus uniquely determined by the the number of vertices of the final graph since each operation adds one vertex). It is argued that those sets always contain at least one operation involving the reversal of edges that are not affected by the corresponding operation for undirected graphs (as it is the case for $\mathcal{T}$). We then provide such a set $\mathcal{A}$ of four types of operations and show how it allows one to build any minimally persistent graph $G = (V, E)$ by applying $|V| - 2$ operations to an initial leader-follower seed. Finally we study the relations between the two sets $\mathcal{A}$ and $\mathcal{T}$.

4.1. Necessary involvement of external edges.

A directed version of a vertex addition or edge splitting is said to be *confined* if it only affects edges that are involved in the corresponding undirected operation. For example, every operation of $\mathcal{S}$ is confined, while the edge reversal operation defined in Section 3.1 is not. In the sequel, we adopt the terms *generalized vertex addition* and *generalized edge splitting* for any operation which is equivalent to a vertex addition or an edge splitting from an undirected point of view. Note that from a rigidity point of view, the rules of the corresponding undirected operation apply when using such a generalized operation.

Suppose that one wants to remove a vertex (without losing persistence) from the provably minimally persistent graph represented in Figure 10 using a generalized reverse edge splitting or reverse vertex addition. The only ones that can be removed are those with a label "+", and due to their total degree, this could only be done by a generalized reverse edge splitting operation. Suppose now that one wants to use a confined version of this operation. One would then remove one of the vertices with a label "+" and connect two of its neighbors by a directed edge. Among the three pairs of neighbors of any vertex with a label "+", two are already connected, and the last pair contains two vertices with an out-degree 2. Adding an edge between a pair of neighbors of the removed vertex without reversing the direction of any other edge would thus imply the presence of either a vertex with out-degree 3 (which by Theorem 2 is impossible in a minimally persistent graph) or of a cycle of length 2 (which by Proposition 1 cannot appear in a minimally rigid graph). This removal should therefore be performed by a *non-confined* reverse generalized edge splitting. The following result is thus proved.

Proposition 7. *If a set exists of generalized vertex additions and edge splittings allowing one to build all minimally persistent graphs from an initial leader-follower seed, such a set must always contain a non-confined edge splitting.*

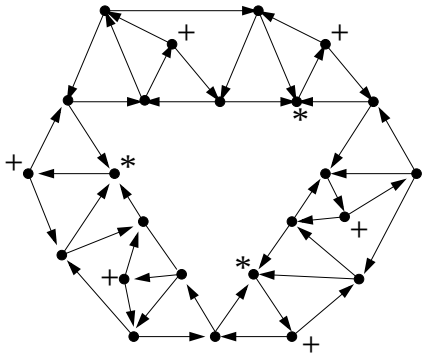

FIGURE 10. A minimally persistent graph no vertex of which can be removed (without losing persistence) by a reverse (generalized) vertex addition or a confined (generalized) reverse edge splitting. The symbol "*" represents one degree of freedom. Vertices that are candidate to be removed by a reverse generalized edge splitting are labelled "+".

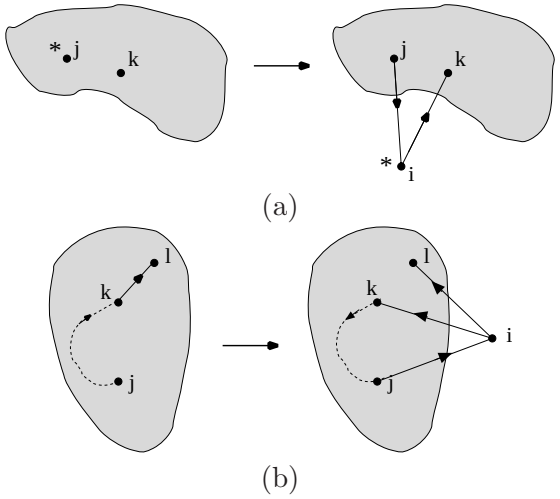

FIGURE 11. Representation of the atypical (a) vertex addition operation and (b) edge splitting operation, both belonging to $\mathcal{A}$. The symbol "*" represents one degree of freedom.

The existence of confined operations that would not be equivalent to vertex addition or edge splitting, but that would however preserve minimal persistence and allow one to build all minimally persistent graphs with $|V|$ vertices in $|V|-2$ operations (starting with a leader-follower seed) remains an open question.

4.2. **Description of a set $\mathcal{A}$ of four primitive operations.** We define here a new set $\mathcal{A}$ of four operations. The first two are the standard vertex additions and edge splitting as described in Section 2.2, which implies that $\mathcal{S} \subset \mathcal{A}$. The two others are atypical versions of these.

Let j, k be two vertices of a minimally persistent graph such that j has at least one degree of freedom. The *atypical vertex addition* operation consists in adding the vertex i, the edges (j, i) and (i, k), as shown in Figure 11(a). As a result, j loses a degree of freedom, and i appears with one. The *reverse atypical vertex addition* operation consists in removing a vertex with in- and out-degree 1.

Let j, k and l be three vertices of a minimally persistent graph such that there is a (simple) directed path from j to k and $(k,l) \in E$. The *atypical edge splitting* operation consists in removing (k,l), adding a vertex i and the edge (j,i), (i,k) and (i,l), and reversing the direction of every edge belonging to the path from j to k, as represented in Figure 11(b). Consider now a vertex i with out-degree 2 and in-degree 1 in a minimally persistent graph, and call its neighbors j, k and l as in Figure 11(b). Suppose that in the graph obtained after deletion of i, there is a path from k (or equivalently l) to j and the pair (k,l) is not connected by an implicit nor an explicit edge. The *reverse atypical edge splitting* consists then in removing i, reversing every edge of the path from k to j to obtain a path from j to k, and adding the edge (k,l).

Proposition 8. *Atypical vertex addition, atypical edge splitting, and their reverse versions preserve minimal persistence.*

The conditions in which the reverse atypical edge splitting can be performed are not always easy to check. However, we have the next result:

Lemma 3. *In a minimally persistent graph, a vertex with in-degree 1 and out-degree 2 can always be removed by either a reverse standard edge splitting or a reverse atypical edge splitting.*

4.3. **Obtaining all minimally persistent graphs using $\mathcal{A}$.** A simple counting argument shows that, in any minimally persistent graph (having more than two vertices), there is at least one vertex with either $(d^-, d^+) = (0,2)$, $(1,1)$ or $(1,2)$. In the first two cases, one can remove this vertex by a reverse standard or atypical vertex addition. In the third case, it follows from Lemma 3 that the vertex can be removed by either reverse standard edge splitting or a reverse atypical edge splitting either a standard. So, given a minimally persistent graph with more than two vertices, it is always possible to obtain a smaller minimally persistent graph by removing a vertex using an operation of $\mathcal{A}^{-1}$. Doing this recursively, one eventually obtains a leader-follower seed (only minimally persistent graph on two vertices) by applying a sequence of operations of $\mathcal{A}^{-1}$. Applying the reverse sequence of operations (of $\mathcal{A}$) on a leader-follower pair results then in the initial graph, which proves the next theorem:

Theorem 3. *Every minimally persistent graph $G = (V,E)$ ($|V| > 1$) can be obtained by performing $|V|-2$ operations of $\mathcal{A}$ on an initial leader-follower seed. Moreover, every intermediate graph is minimally persistent.*

Using the same argument as for Corollary 1, we obtain the following result.

Corollary 2. *Every minimally persistent graph can be transformed into any other minimally persistent graph using only operations of $\mathcal{A} \cup \mathcal{A}^{-1}$.*

Remark 1. *It is possible to show that among the four operations of $\mathcal{A}$ (resp. $\mathcal{A}^{-1}$), none can be removed without being replaced by some alternative new operation if the operation set is to produce all minimally persistent graphs (resp. contain for each minimally persistent graph an operation that can be performed on it). However, the set of generalized vertex additions and edge splittings that we present here is just one among the several sets that we have found allowing one to build all minimally persistent graphs. It offers the advantage that its non-confined operation has a more local character than those contained in the other sets (which are not described here).*

4.4. **Relations between $\mathcal{A}$ and $\mathcal{T}$.** We examine here the relation between the two sets $\mathcal{A}$ and $\mathcal{T}$. Let us first define a (partial) order relation between the group of operations. We say that $\mathcal{X} \leq \mathcal{Y}$ if every operation of $\mathcal{X}$ can be implemented by a sequence of operations of $\mathcal{Y}$. If $\mathcal{X} \leq \mathcal{Y}$ and $\mathcal{X} \geq \mathcal{Y}$, we say that $\mathcal{X} = \mathcal{Y}$. If $\mathcal{X} \leq \mathcal{Y}$ and $\mathcal{X} \neq \mathcal{Y}$, we say that $\mathcal{X} < \mathcal{Y}$. One can see that $\mathcal{X}^{-1} \leq \mathcal{Y}^{-1}$ if and only if $\mathcal{X} \leq \mathcal{Y}$.

Lemma 4. *An atypical vertex addition can be implemented using one standard vertex addition and one edge reversal. An atypical edge splitting can be implemented using one standard edge splitting and one or more edge reversal(s).*

This lemma states that every operation of $\mathcal{A}$ that does not belong to $\mathcal{T}$ can be implemented by a sequence of operations of $\mathcal{T}$. In other words, $\mathcal{A} \leq \mathcal{T}$. Moreover, it is impossible to implement an edge reversal (which belongs to $\mathcal{T}$) by a sequence of operations of $\mathcal{A}$ since they all increase the number of vertices in the graph. We have thus the following relation between the two sets of operations:

Proposition 9. *$\mathcal{A} < \mathcal{T}$, and equivalently $\mathcal{A}^{-1} < \mathcal{T}^{-1}$*

Since the set $\mathcal{T}$ of operations is more powerful than $\mathcal{A}$, Theorem 3 is a stronger result than Theorem 2. However, if we look at the sets containing both normal and inverse operations, the results are different. Suppose indeed that a graph G' is obtained by performing an operation of $\mathcal{T} \cup \mathcal{T}^{-1}$ on a minimally persistent graph G. Since both graphs are minimally persistent, Corollary 2 implies that G' can also be obtained by applying a sequence of operations of $\mathcal{A} \cup \mathcal{A}^{-1}$ on G. Any operation of $\mathcal{T} \cup \mathcal{T}^{-1}$ can thus be implemented by a sequence of operations of $\mathcal{A} \cup \mathcal{A}^{-1}$. Conversely, any operation of $\mathcal{A} \cup \mathcal{A}^{-1}$ can be implemented by a sequence of operations of $\mathcal{T} \cup \mathcal{T}^{-1}$. We have thus shown the following result:

Proposition 10. $\mathcal{A} \cup \mathcal{A}^{-1} = \mathcal{T} \cup \mathcal{T}^{-1}$

5. Conclusions and future work

We have extended the Henneberg sequence concept to directed graphs. From an autonomous agent point of view, this provides a systematic approach to sequentially obtain or reorganize a minimally persistent agent formation in such a way that persistence is maintained throughout the process. We also exposed some natural restrictions to these extensions, the main one being the impossibility of building all minimally persistent graphs using only confined generalized vertex additions or edge splittings.

We proposed two sets of operations, each of which allows one to obtain any minimally persistent graph from a leader-follower seed. The first one ($\mathcal{T}$ in Section 3) contains the two standard vertex additions and edge splittings already introduced in [4] and a purely directed operation (i.e. a neutral operation from an undirected point of view). The second set ($\mathcal{A}$ in Section 4) contains, in addition to the two standard operations, two atypical versions of them, among which one is not a confined operation. It involves indeed the reversal of a path of undetermined length in the graph. However, it is still an open question to know if similar results could be obtained using operations involving a number of reversals fixed or bounded independently of the size of the graph. Note that for the second set, the number of operations required to build a minimally persistent graphs is uniquely determined by the size of the graph.

From an autonomous agent point of view, it would be useful to study how these various operations could be performed in a decentralized way in order to efficiently construct or reorganize a formation. For this purpose, the operations of $\mathcal{T}$ could be preferred for their simplicity and because the successive modifications are only local modifications. This study could also imply the development of an optimal algorithm (using one of the two sets of operations) to reorganize a persistent formation.

As a final remark, note that we have only focused on the transformations of minimally persistent graphs into other minimally persistent graphs. Several practical issues concerning autonomous agent formations arise relating to the merging of such graphs, or their repair after the loss of some vertices or edges. It is thus worthwhile to study these particular problems as well, as already partially done in a paper by some of the authors [1].

References

[1] B.D.O. Anderson, C. Yu, and J.M. Hendrickx. Use of meta-formations for cooperative control. In *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS2006)*, Kyoto (Japan), July 2006.

[2] J. Baillieul and A. Suri. Information patterns and hedging brockett's theorem in controlling vehicle formations. In *Proc. of the 42nd IEEE Conf. on Decision and Control*, volume 1, pages 556–563, Hawaii, December 2003.

[3] A. Das, J. Spletzer, V. Kumar, and C. Taylor. Ad hoc networks for localization and control. In *Proc. of the 41st IEEE Conf. on Decision and Control*, Las Vegas, NV, 2002.

[4] T. Eren, B.D.O. Anderson, A.S. Morse, and P.N. Belhumeur. Information structures to secure control of rigid formations with leader-follower structure. In *Proc. of the American Control Conference*, pages 2966–2971, Portland, Oregon, June 2005.

[5] J.M. Hendrickx, B.D.O. Anderson, and V.D. Blondel. Rigidity and persistence of directed graphs. In *Proceedings of the 44th IEEE Conference on Decision and Control*, Seville, Spain, December 2005.

[6] J.M. Hendrickx, B.D.O. Anderson, J.-C. Delvenne, and V.D. Blondel. Directed graphs for the analysis of rigidity and persistence in autonomous agents systems. *To appear in International Journal of Robust and Nonlinear Control's special issue on Communicating-Agent Systems.*

[7] J.M. Hendrickx, B. Fidan, C. Yu, B.D.O. Anderson, and V.D. Blondel. Primitive operations for the construction and reorganization of minimally persistent formations, preprint. 2005.

[8] J.M. Hendrickx, B. Fidan, C. Yu, B.D.O. Anderson, and V.D. Blondel. Rigidity and persistence of three and higher dimensional formations. In *Proceedings of the First International Workshop on Multi-Agent Robotic Systems (MARS 2005)*, pages 39–46, Barcelona, Spain, September 2005.

[9] L. Henneberg. Die graphische Statik der starren Systeme. Leipzig, 1911.

[10] G. Laman. On graphs and rigidity of plane skeletal structures. *J. Engrg. Math.*, 4:331–340, 1970.

[11] R. Olfati-Saber and R.M Murray. Graph rigidity and distributed formation stabilization of multi-vehicle systems. In *Proceedings of the 41st Conference on Decision and Control*, Las Vegas, NV, December 2002.

[12] T. Tay and W. Whiteley. Generating isostatic frameworks. *Structural Topology*, (11):21–69, 1985.

[13] C. Yu, J.M. Hendrickx, B. Fidan, and B.D.O. Anderson. Structural persistence of three dimensional autonomous formations. In *Proceedings of the First International Workshop on Multi-Agent Robotic Systems (MARS 2005)*, pages 47–55, Barcelona, Spain, September 2005.

[14] C. Yu, J.M. Hendrickx, B. Fidan, B.D.O. Anderson, and V.D. Blondel. Three and higher dimensional autonomous formations: Rigidity, persistence and structural persistence. *To appear in Automatica.*