Solving a problem of Diestel [9] relevant to the theory of cycle spaces of infinite graphs, we show that the Freudenthal compactification of a locally finite graph can have connected subsets that are not path-connected. However we prove that connectedness and path-connectedness do coincide for all but a few sets, which have a complicated structure.

1. Introduction

The Freudenthal compactification $|G|$ of a locally finite graph $G$ is a well-studied space with several applications. For example, Cartwright, Soardi and Woess [8] study it in the context of random walks on infinite graphs and show that it coincides with the Martin compactification whenever $G$ is a tree (see also Woess [21]). Polat [18] investigates its subspace topology on the set of vertices and ends, and relates the existence of certain spanning trees to the metrizability of this space. Bruhn, Diestel, Kühn and Stein [2-6,9,11-14] use it in order to define topological notions of paths, cycles and spanning trees that permit the extension of classical theorems about the cycle space of finite graphs to infinite ones. Finally, Stein [19,20] and Bruhn and Yu [7] have begun to tackle topological variants of extremal-type problems in $|G|$, such as hamiltonicity or forcing highly connected subgraphs, that are standard for finite graphs but would otherwise have no counterpart for infinite graphs.

However, the following fundamental problem has remained unsolved:

**Problem 1 ([9]).** Is every connected subspace of $|G|$ path-connected?

*Mathematics Subject Classification (2000): 05C10*
Apart from being interesting as a basic topological question in its own right, this problem is also important from the graph-theoretical point of view. Indeed, the basic concepts in the context of [2-6,9,11-14] are circles and topological spanning trees, generalising cycles and spanning trees respectively. In order to prove that a certain subspace of $|G|$ is a circle or a topological spanning tree, one has to show that it is path-connected, but it is often much easier to show that it is (topologically) connected. See for example Theorem 8.5.8 in [10], which summarizes the basic properties of the cycle space of a locally finite graph. The reduction of path-connectedness to connectedness simplifies its proof considerably in comparison to the original proof in [13, Theorem 5.2]. Lemma 8.5.13 in [10] is an example of how reducing path-connectedness to connectedness can facilitate proving the existence of a topological spanning tree, which can otherwise be a tedious task as witnessed by the proof of Theorem 5.2 in [14]. Further examples are given in Exercises 65 and 70 of [10], which describe some fundamental properties of circles and topological spanning trees: their proofs become easy when the path-connectedness required is replaced with connectedness, while without this tool they would be arduous and long.

Diestel and Kühn [14] have shown that every closed connected subspace of $|G|$ is path-connected, and expressed a belief that the answer to Problem 1 should be positive also in general. However, we shall construct a counterexample (Section 3):

**Theorem 1.** There exists a locally finite graph $G$ such that $|G|$ has a connected subspace which is not path-connected.

While the construction for the proof of Theorem 1 is our main result, we also prove that ‘most’ connected subsets as above are indeed path-connected (Section 4):

**Theorem 2.** Given any locally finite connected graph $G$, a connected subspace $X$ of $|G|$ is path-connected unless it satisfies the following assertions:

- $X$ has uncountably many path-components each of which consists of one end only;
- $X$ has infinitely many path-components that contain a vertex; and
- every path-component of $X$ contains an end.

Since the existence of a connected but not path-connected subspace was rather unexpected, I consider it as the main result of this paper and present it first. The counterexample can well be read by itself, but it may look somewhat surprising. However, the proof of Theorem 2 will make it less surprising with hindsight: it will show why the example had to be the way it is.