algorithms

MDPI

Article

Constrained Connectivity in Bounded X-Width Multi-Interface Networks

Alessandro Aloisio [1] and Alfredo Navarra [2,*]

[1] Department of Information Engineering, Computer Science and Maths, University of L'Aquila; School of Advanced Studies, Gran Sasso Science Institute, 67100 L'Aquila, Italy; alessandro.aloisio@univaq.it
[2] Department of Mathematics and Computer Science, University of Perugia, 06123 Perugia, Italy
* Correspondence: alfredo.navarra@unipg.it; Tel.: +39-075-585-5046

Received: 10 December 2019; Accepted: 20 January 2020; Published: 26 January 2020



Abstract: As technology advances and the spreading of wireless devices grows, the establishment of interconnection networks is becoming crucial. Main activities that involve most of the people concern retrieving and sharing information from everywhere. In heterogeneous networks, devices can communicate by means of multiple interfaces. The choice of the most suitable interfaces to activate (switch-on) at each device results in the establishment of different connections. A connection is established when at its endpoints the devices activate at least one common interface. Each interface is assumed to consume a specific percentage of energy for its activation. This is referred to as the cost of an interface. Due to energy consumption issues, and the fact that most of the devices are battery powered, special effort must be devoted to suitable solutions that prolong the network lifetime. In this paper, we consider the so-called *p-Coverage* problem where each device can activate at most p of its available interfaces in order to establish all the desired connections of a given network of devices. As the problem has been shown to be NP-hard even for $p = 2$ and unitary costs of the interfaces, algorithmic design activities have focused in particular topologies where the problem is optimally solvable. Following this trend, we first show that the problem is polynomially solvable for graphs (modeling the underlying network) of bounded treewidth by means of the Courcelle's theorem. Then, we provide two optimal polynomial time algorithms to solve the problem in two subclasses of graphs with bounded treewidth that are graphs of bounded pathwidth and graphs of bounded carvingwidth. The two solutions are obtained by means of dynamic programming techniques.

Keywords: multi-interface networks; constrained connectivity; energy optimization; treewidth; pathwidth; carvingwidth

1. Introduction

In the last decade, multi-interface wireless networks have drawn the attention of practitioners and the research community. They are a widely used communication infrastructure which can support a plethora of different, important and popular applications such as file transfers. Multi-interface wireless networks are composed of heterogeneous devices that have different computational capabilities, are battery powered and can be equipped with different communication technologies such as Bluetooth, WiFi, 4G, 5G and GPRS. Mobile phones, laptops, smartwatches, tablets are but a few of the possible heterogeneous devices that can compose a multi-interface wireless network. They can activate one or more interfaces in accordance to the bandwidth of communication that is required, the cost (generally expressed as a percentage of the energy consumption) for maintaining an interface as active, the neighborhood. The activation of an interface has associated an energy cost. This must be carefully considered when establishing the desired connections since devices are battery powered. Suitable solutions should be devised to prolong the network lifetime.

In this paper, we aim at finding the most energy efficient connections by activating a subsets of node interfaces. Actually, a fixed positive integer p is defined which indicates the maximum number of interfaces a single device can activate at most. We describe a multi-interface wireless network by using a graph $G = (V, E)$, where V is the device set and E is the set of connections. This is defined by considering some parameters such as the distance between devices and the interfaces they share. Each $v \in V$ is associated with an interface set $W(v)$. The set $\bigcup_{v \in V} W(v)$ defines all the available network interfaces; k denotes its cardinality. The endpoints of an edge that share at least one active interface can establish a connection. A node u consumes an energy $c(\alpha)$ to keep the interface α active. This provides a maximum communication bandwidth $b(\alpha)$. Figure 1 shows a multi-interface wireless network where various devices such as mobile phones, smartphones, smartwatches, tablets and laptops can perform a point-to point communication by using different interfaces and protocols, i.e., IRdA, Bluetooth, Wi-Fi, GSM, 4G.



Figure 1. Example of multi-interface network composed of heterogeneous devices connected by means of different interfaces.

It is worth noticing that each connection can be established by at least an interface; some devices are not directly connected although they share some interfaces. This can be consequence of different factors such as obstacles, distances and the protocols that are used for communication. A full instance specification could be provided (if necessary) by defining the cost of activating an interface on a specific device together with the interface bandwidth. This can be assumed equal among all devices while the energy spent to activate a specific interface may be different for different devices. We simplify the model by having a cost that refers to the percentage of battery consumed by each device. Effectively, the cost can be considered the same for each device with respect to a specific interface among the whole network. Nevertheless, different assumptions may lead to completely different problems that point out specific peculiarities of the multi-interface networks.

In this paper, we study a variant of the *Coverage* problem [1]. This is used to find the cheapest way to establish all connections that are defined by an input graph G. The interfaces used for establishing a connection and bandwidths requirements are not considered. This problem only aims at ensuring that for each edge of G there is a common active interface at its endpoints. The objective is the minimization of the overall network activation cost. We add to the *Coverage* problem a further constraint [2–5] where a device cannot activate more than p interfaces. This constraint is used to keep under control the energy spent by single devices. In other words, instead of finding the solution that minimizes the overall cost due to the activation of the interfaces along the whole network, or the one that minimizes the maximum cost spent by a single node, the aim becomes to minimize the overall cost subject to the constraint p. This problem has been proved to be NP-hard also for simple instances such as $p = 2$ and unitary costs of the interfaces. For this reason, algorithms design activities have focused their attention to particular topologies where the problem can be solved in an optimal way. We follow

this trend by solving the problem in polynomial time for graphs (modeling the underlying network) of bounded pathwidth and for graphs of bounded carvingwidth, which represent two main graph classes frequently studied for hard problems such as p-Coverage. Actually, we first show that the problem is polynomially solvable for graphs of bounded treewidth. Even though both graphs of bounded pathwidth and graphs of bounded carvingwidth represent two (incomparable) subclasses of graphs of bounded treewidth, the requirement in exploring such classes comes by the fact that specific algorithms allow for better performance.

1.1. Related Work

In the last decade multi-interface wireless networks have drawn the attention of the research community. The benefits of taking advantage of multiple interfaces at each device is usually the main focus. In this context, many standard network optimization problems have been reconsidered [6,7] by focusing on routing issues [8] and network connectivity [9].

Multi-interface wireless network combinatorial problems have been studied in [10] and [1,11] which investigate the *Coverage* problem. The constrained *Coverage* problem referred to as p-Coverage has been introduced in [2] and further investigated in [3].

The authors in [12,13] study the problem of finding the cheapest way to ensure network connectivity. More precisely, they aim at activating for each node a subset of its available interfaces which ensures a path between every pair of nodes in G. These paths should minimize the overall cost of all interfaces that have been activated. This corresponds to a generalization of the Minimum Spanning Tree problem as the set of connections established must form a spanning subgraph of G, not necessarily a tree. In fact, as costs are not on the edges but on the node interfaces, a node can use the same interface to establish several connections thus saving energy. This property of multi-interface networks highlights the advantage as well as the higher complexity of the studied problems.

In [11] the authors still study connectivity but with a different objective function, i.e., the minimization of the maximum cost paid on a single node. This is a widely studied objective function as it is more suitable for distributed settings.

The authors in [13] study the *Cheapest path* problem. The goal is to activate available interfaces in some nodes to guarantee a path between two specified nodes. This path should have a minimum cost in terms of activated interfaces. This problem is the generalization of the Shortest Path problem between two nodes in standard networks. The *cheapest path* is one of the few problems that have been reconsidered in the context of multi-interface networks that maintains its computational complexity. In fact, it can be optimally solved in polynomial time.

The authors in [14] study the *Maximum Matching* problem. As with its classical version, the problem looks for the maximum subset of connections that can be established at the same time without sharing any common node. Each node must appear at most once in the solution since this must have a set of disjoint edges. The *Maximum Matching* problem becomes difficult in the context of the multi-interface networks. When considering a problem instance of an input graph G without the specification of costs and bandwidths, the solution is difficult to be found. In fact, two edges established by means of the same interface included in the solution, cannot be directly connected by another edge. This connection would invalidate the solution since both its endpoints share the same active interface.

The authors [15,16] face bandwidth constraints by investigating flow problems in multi-interface networks. Each interface is associated with one additional parameter that defines the bandwidth the interface can deal with. The *Maximum Flow* problem and the *Minimum Cost Flow* problem aim at ensuring a connection between two given nodes by considering bandwidth constraints on the provided interfaces.

The Maximum Flow problem finds the maximum bandwidth between two selected nodes. Actually, the maximum value achievable can be obtained by standard techniques. More specifically, by considering all the network interfaces as active, then all the allowed connections are established

and the problem coincides with that on standard networks where bandwidth capacities are associated with edges and not to interfaces. However, when one wants to find a solution that guarantees the maximum flow but at minimum cost, then a suitable activation of the available interfaces must be found. Hence, this problem is a generalization of the Maximum Flow problem in standard networks.

The Minimum Cost Flow problem aims at ensuring the following two goals: (i) a communication sub-network between two given nodes which has minimum energy consumption; (ii) a minimum amount B of communication bandwidth. In practice, the problem aims at finding the minimum cost set of interfaces to activate among the input network so that two specified nodes s and t are guaranteed to exchange data with at least B bandwidth. Clearly, the solution might result in a complex graph with source s and tail t composed of nodes with active interfaces. This problem is a generalization of the Minimum Cost Flow problem in standard networks.

The theoretical results of multi-interface wireless networks can be applied in various applications such military applications and the tactical networks that are provided with new software technologies [17,18] which employ resource-constrained mobile devices. The main resource that must be preserved is the battery power of each device thus a suitable usage of the interfaces could be crucial for improving the network lifetime.

1.2. Our Results

In this paper, we are interested in the p-Coverage problem. In particular, each node of the network can activate at most p interfaces. In what follows, the p-Coverage problem will be denoted simply as *CMI(p)*, in accordance to the name provided to the original formalization of the Coverage problem on multi-interface networks where no restrictions on the number of interfaces each device may activate was introduced, see [1]. Actually, we consider what was referred to as the *unbounded case*, in which instead there is no bound on the number k of interfaces available over all the network. Considering the new notation, that problem becomes *CMI(∞)*, i.e., $p = \infty$. Moreover, following the trend of the previous work [2,3] on the same subject (and similarly to the technique used in [19]), we focus on *CMI(2)*. The contribution of this paper is summarized and compared with previous results in Table 1.

Table 1. Complexity of the *CMI(2)* problem. Parameters n, k, Δ and h are the number of nodes, the number of interfaces, the maximum node degree, and the X-width of the input instance of *CMI(2)*, respectively.

Graph Class	Costs	Complexity of *CMI(2)*	Reference
Graphs with $\Delta \geq 4$	unitary	NP-complete (feasibility)	[2]
Treewidth h	arbitrary	feasibility in poly-time regarding h	this paper
Carvingwidth h	arbitrary	optimally solvable in $O(k^{4h}n)$	this paper
Pathwidth h	arbitrary	optimally solvable in $O(k^{2(h+1)}n)$	this paper
Series-Parallel graphs	arbitrary	optimally solvable in $O(k^6n)$	[3]
Complete Bipartite graphs	arbitrary	optimally solvable in $O(k^4n)$	[2]
Complete graphs	arbitrary	optimally solvable in $O(k^3n)$	[2]
Rings	arbitrary	optimally solvable in $O(k^3n)$	[2]
Trees	arbitrary	optimally solvable in $O(\Delta k^2n)$	[2]
Paths	arbitrary	optimally solvable in $O(kn)$	[13]

In particular, we investigate *CMI(2)* on the class of graphs with bounded treewidth, showing that it is indeed polynomially solvable. Then, to obtain specific performance, we consider two well-known (but incomparable) subclasses of graphs with bounded treewidth that are graph admitting a bounded pathwidth or a bounded carvingwidth. While the formal definitions of treewidth, pathwidth and carvingwidth will be provided later, here we give just a first intuition of what kind of graphs we are approaching.

A *tree decomposition* of a graph G is a tree of subsets of vertices of G such that the endpoints of each edge of G appear together in at least one subset and all subsets containing a same vertex of G constitute a connected subtree. The treewidth is one less than the size of the largest subset in a minimal tree decomposition.

A *path decomposition* of a graph G is a sequence (a path) of subsets of vertices of G such that the endpoints of each edge of G appear in at least one subset and each vertex of G appears in a contiguous subsequence of the subsets. The pathwidth is one less than the size of the largest subset in a minimal path decomposition.

A *carving* of a graph G is a tree T whose internal vertices all have degree 3 and whose leaves correspond to the vertices of G. The *width* of a carving T is the maximum size of an edge-cut in G that is induced by an edge of T. The carvingwidth of G is the minimum width of a carving of G.

While graphs admitting bounded pathwidth or bounded carvingwidth also admit bounded treewidth, there is no relation among the two classes of graphs admitting bounded pathwidth and those of bounded carvingwidth. Hence, investigating on the resolution algorithms of *CMI(2)* when restricted to one of the two subclasses is a challenging issue. In fact, since in general the problem has been shown to be NP-hard even for graphs of maximum degree $\Delta \geq 4$ with interfaces of unitary costs, it is worth investigating when the problem becomes affordable. As shown in Table 1, many graph classes have been already investigated, including paths, trees, rings, complete graphs, complete bipartite graphs, and series-parallel graphs. Hence our work is a continuation of this investigation, involving graphs of bounded treewidth, pathwidth or carvingwidth. In particular, while for graphs of bounded treewidth (and hence also for graphs of bounded pathwidth or carvingwidth) we prove that *CMI(2)* is solvable in polynomial time, in the specific cases of graphs with bounded pathwidth and graphs with bounded carvingwidth we also provide specific polynomial time algorithms that solve *CMI(2)*.

1.3. Outline

In the next section we provide all necessary definitions and notation to formalize the *CMI(p)* problem. In Section 3 we provide our first result holding for graphs of bounded treewidth. In Section 4, we focus on *CMI(2)* in the case that the input graph admits a bounded pathwidth. We then provide an optimal resolution algorithm for the specific case. In Section 5, we investigate the optimal resolution of *CMI(2)* when the input graph admits a bounded carvingwidth. The obtained solutions work in polynomial time. Finally, Section 6 provides some useful remarks and discussions for interesting directions of future works.

2. Notation and Definitions

Given a graph $G = (V, E)$, let V be the set of nodes, and E be the set of edge. Moreover, let $n = |V|$ and $m = |E|$. Unless differently specified, G is assumed to be simple (without multiple edges nor self-loops), undirected and connected. For each node $v \in V$, we denote by $\deg(v)$ the degree of v, and let $\Delta = \max_{v \in V} \deg(v)$. A global assignment of the interfaces to the nodes in V is given in terms of an appropriate interface assignment function W, as follows.

Definition 1. *A function* $W\colon V \to 2^{\{1,2,\ldots,k\}}$ *is said to* cover *graph* G *if for each* $\{u, v\} \in E$ *we have* $W(u) \cap W(v) \neq \emptyset$.

For all nodes in V, the activation of a specific interface is assumed to cost the same. The rationale behind it is that each device might be thought to spend the same percentage of its energy. Hence, the costs associated with the interfaces are defined by a function $c\colon \{1, \ldots, k\} \to \mathbb{R}_+$, and the cost of activating interface i is referred to as c_i. The considered *CMI(p)* optimization problem is so formulated:

Algorithm 1. *CMI(p)*: Coverage in Multi-Interface Networks

Input: A graph $G = (V, E)$, an allocation of available interfaces $W\colon V \to 2^{\{1,\ldots,k\}}$ covering graph G, an interface cost function $c\colon \{1, \ldots, k\} \to \mathbb{R}_+$, an integer $p \geq 1$.

Solution: If possible, an allocation of active interfaces $W_A\colon V \to 2^{\{1,\ldots,k\}}$ covering G such that for all $v \in V$, $W_A(v) \subseteq W(v)$ and $|W_A(v)| \leq p$; Otherwise, a negative answer.

Goal: Minimize the total cost of the interfaces that are activated, i.e., $c(W_A) = \sum_{v \in V} \sum_{i \in W_A(v)} c_i$.

In general, the cost function c spans over $\mathbb{R}^+$. When $c(i) = 1$, for any $i = 1, \ldots, k$ then the particular case of *unitary cost* is considered. Clearly $k \geq 2$ is always assumed as the case $k = 1$ admits the obvious and unique solution where all the nodes activate the only available interface.

Please note that *CMI(p)* is a generalization of the original *CMI(∞)* problem (see [1,7]), where each node cannot activate more than p interfaces. Surprisingly, the basic case for $p = 2$ comes out to be more difficult, in general, than *CMI(∞)*. In fact, in [2] the following theorem has been proved:

Theorem 1 ([2]). *Finding a feasible solution for CMI(2) is* NP*-complete for graphs with* $\Delta \geq 4$*, even for the unitary cost case.*

Please note that the feasibility of *CMI(∞)* is easily solvable by the definition of the problem. In fact, by activating all interfaces for each node, a feasible solution for *CMI(∞)* is obtained. However, there are special graph classes that turn out to be much more affordable in *CMI(2)*. Concerning trees and complete graphs, for instance, in [2] the *CMI(∞)* problem has been investigated. For trees *CMI(∞)* comes out to be *APX*-hard whereas for complete graphs it is not approximable within $O(\log k)$. In both topologies, instead, *CMI(2)* is optimally solvable in polynomial time.

In the next section, we show that *CMI(2)* is polynomially solvable in graphs of bounded treewidth, whereas the subsequent sections explore in more details *CMI(2)* in two different and incomparable graph classes, i.e., graphs with bounded pathwidth and graphs with bounded carvingwidth.

3. Graphs with Bounded Treewidth

In this section, we show that *CMI(2)* is polynomially solvable in graphs of bounded treewidth. We start by formally define what the *treewidth* of a graph is. A tree decomposition of a graph G is a way of representing G as a tree-like structure.

Definition 2 ([20]). *A* tree decomposition *of a graph* $G = (V, E)$ *is a pair* $(\{X_i | i \in I\}, T = (I, F))$ *with* $\{X_i | i \in I\}$ *a collection of subsets of* V*, called bags, and* $T = (I, F)$ *a tree, such that*

- (i) for every $v \in V$ there exists $i \in I$ with $v \in X_i$;
- (ii) for every $(u, v) \in E$, there exists $i \in I$ with $v, w \in X_i$;
- (iii) for every $v \in V$,the set $I_v = \{i \in I | v \in X_i\}$ forms a connected subgraph (subtree) of T.

The *width* of a tree decomposition $(\{X_i | i \in I\}, T = (I, F))$ equals $\max_{i \in I} |X_i| - 1$. The treewidth of a graph G is the minimum width of a tree decomposition of G. The treewidth is said to be *bounded* if it is limited by a constant h. In other words, for any fixed constant h, if G has treewidth at most h then G is said to be of *bounded treewidth*.

Before providing the result on the solvability of *CMI(2)* in graphs of bounded treewidth, we need to recall the well-known Courcelle's theorem [21]:

Theorem 2 (Courcelle's [21]). *Given a graph* G *and a property* ϕ *on* G *expressed in the monadic second-order logic, then checking whether* G *satisfies* ϕ *is Fixed Parameter Tractable (FPT) with respect to the treewidth of* G*.*

Courcelle's theorem basically provides a powerful means to understand whether the problem of verifying a property ϕ on a graph G is Fixed Parameter Tractable (FPT) with respect to the treewidth of G. This is actually what we obtain by applying the Courcelle's theorem to *CMI(2)*. Indeed, the next theorem shows that *CMI(2)* can be formulated in the monadic second-order logic.

Theorem 3. *Let I be an instance of CMI(2) such that the input graph G admits a tree decomposition of width h. Then CMI(2) is FPT with respect to h.*

Proof. The proof is based on Courcelle's theorem which states that any graph property that can be defined on monadic second-order logic can be tested in time $f(h) \cdot poly(|I|)$, where f is some computable function and $|I|$ is the dimension of the input. Thus, to complete the proof, we need to express *CMI(2)* in the monadic second-order logic.

Given a graph $G = (V, E)$, a covering function $W\colon V \to 2^{\{1,\ldots,k\}}$ and an interface cost function $c\colon \{1,\ldots,k\} \to \mathbb{R}_+$, we can define *CMI(2)* as follows:

$$\min_{W_A} c(W_A) : \forall x \in V : (W_A(x) \subseteq W(x) \ \wedge \ |W_A(x)| \leq 2) \wedge$$

$$\forall (x, y) \in E : W_A(x) \cap W_A(y) \neq \emptyset.$$

We conclude that within time $f(h) \cdot poly(|I|)$ it is possible to answer whether *CMI(2)* admits a solution or not. □

From Theorem 3, it follows that *CMI(2)* is solvable in polynomial time for any graph with bounded treewidth, since in such a case h is a constant and consequently $f(h)$ is constant as well. However, the performance of the underlying resolution algorithm might be practically heavy, hence requiring further investigation on specific cases. In particular, in the next sections we consider two different and incomparable graph classes that are graphs with bounded pathwidth and graphs with bounded carvingwidth. Please note that such classes are both subclasses of graphs with bounded treewidth but, as we are going to see, they allow different computational times.

4. Graphs with Bounded Pathwidth

In this section, first we formally define what the *pathwidth* of a graph is. Subsequently, we present our new resolution algorithm that optimally solves *CMI(2)* in polynomial time on graphs with bounded pathwidth .

4.1. Definition of Pathwidth

Definition 3 ([20])**.** *A path decomposition of a graph $G = (V, E)$ is a set $\mathcal{P} = (X_1, \ldots, X_r)$ of subsets of V that is $X_i \subseteq V$ for each $i \in \{1, \ldots, r\}$, called bags, such that*

- (i) for every $u \in V$ there exists $i \in \{1, \ldots, r\}$ with $u \in X_i$;
- (ii) for every $(u, v) \in E$, there exists $i \in \{1, \ldots, r\}$ with $u, v \in X_i$;
- (iii) for every three bags X_i, X_j, and X_k, with $i \leq j \leq k$, it holds that $X_i \cap X_k \subseteq X_j$.

The *width* of a path decomposition $\mathcal{P}$ is the maximum number of vertices contained in every bag of $\mathcal{P}$ minus one, i.e., $\max_{i \in \{1,\ldots,r\}} |X_i| - 1$. The *pathwidth* of a graph G, is the minimum width for every possible path decomposition of G. For any fixed constant h, if G has pathwidth at most h then G is said to be of *bounded pathwidth*. In the following, we will call nodes the elements of $\mathcal{P}$, and vertices the elements of V, to avoid confusions.

An important property of path decomposition [20], which we exploit to write our dynamic programming algorithm, is the *pathwidth separator* property. It states that for every three nodes X_i, X_j, and X_k such that X_j is between X_i and X_k, each path in G that connects a vertex in $X_i \setminus X_j$ with a vertex in $X_k \setminus X_j$ contains a vertex in X_j.

This means that node X_j separates the vertices in $X_i \setminus X_j$ from the ones in $X_k \setminus X_j$. Our algorithm works on a particular type of path decomposition called *nice*. A nice path decomposition can be always obtained from a path decomposition in linear time, maintaining the same width.

Definition 4 ([20]). *A path decomposition of a graph $G = (V, E)$ is nice if $|X_1| = |X_r| = 1$, and for every $i \in \{1, 2, \ldots, r-1\}$ there is a vertex $v \in V$, such that either $X_{i+1} = X_i \cup \{v\}$ (introduce node), or $X_{i+1} = X_i \setminus \{v\}$ (forget node).*

It is easy to check that the number of nodes in a nice path decomposition is at most twice the number of vertices in V, because property (iii) in Definition 3 says that every vertex $v \in V$ belongs to a consecutive set of bags. A path decomposition of the graph in Figure 2 is depicted in Figure 3, while a nice path decomposition, of the same graph, can be found in Figure 4. Both decompositions have width equal to 2.

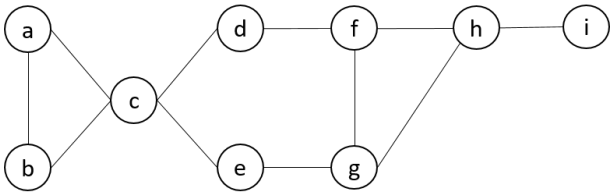


Figure 2. A sample graph admitting pathwidth 2.

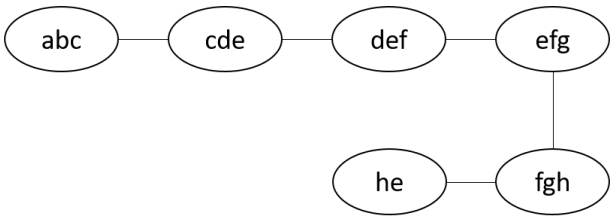


Figure 3. A path decomposition of width 2 for the graph shown in Figure 2.

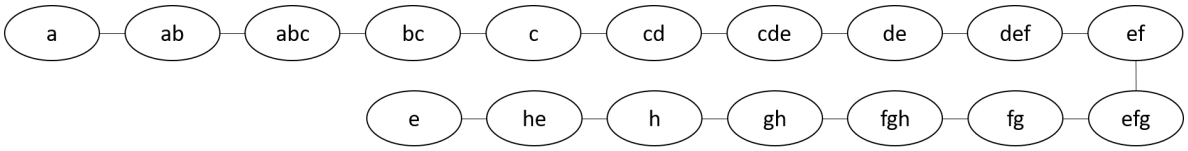


Figure 4. A nice path decomposition of width 2 for the graph shown in Figure 2.

4.2. Solving CMI(2) on Graphs with Bounded Pathwidth

This section describes a polynomial time optimal algorithm for *CMI(2)* on graphs with bounded pathwidth, which uses the dynamic programming technique. The algorithm exploits the nice path decomposition provided above, and in particular the pathwidth separator property.

Given an instance of *CMI(2)*, it is possible in linear time [22,23] to find a pathwidth decomposition for $G(V, E)$ of width h. Then, the algorithm computes a nice path decomposition $\mathcal{P} = (X_1, \ldots, X_r)$ with the same width h, which can be done again in linear time [20]. Denote by $G(X_i)$ the subgraph induced by the vertices in $\bigcup_{j=1}^{i} X_j$. Let $f(X_i, \mathcal{A})$ be the minimum value of *CMI(2)* on $G(X_i)$, where $\mathcal{A}$ is a collection of $|X_i|$ subsets $A(u)$ of the available interfaces $W(u)$, with $u \in X_i$, which satisfies the following constraints.

- The interfaces *active* in the vertex u are those in $A(u)$, with $|A(u)| \leq 2$, for every $u \in X_i$.

By exploiting the pathwidth separator property, the core of the algorithm computes at each node X_i of $\mathcal{P}$ the values $f(X_i, \mathcal{A})$, for every possible collection $\mathcal{A}$. The dynamic programming algorithm starts at X_1, and ends at X_r. Clearly, the constrained version of *CMI(2)* cannot be always solvable.

In this case, we set $f(X_i, \mathcal{A}) = +\infty$. Since in X_1 there is only one vertex u, in $G(X_1)$ there are no edges, and $\mathcal{A} = \{A(u)\}$, the following condition holds:

- $f(X_1, \mathcal{A}) = c(A(u))$ if $|A(u)| \subseteq W(u)$, and $|A(u)| \leq 2$.

Substantially, for X_1 we activate all the possible subsets of at most two interfaces available for u. Actually, since in $G(X_1)$ there is only one vertex, these partial solutions are needed only to build an optimal solution for G, if G contains more than one vertex.

In any *introduce node* $X_{i+1} = X_i \cup \{v\}$, the value $f(X_{i+1}, \mathcal{A})$, for a specific collection $\mathcal{A}$ of active interface sets, is computed by solving the following constrained minimization problem, which uses the values $f(X_i, \mathcal{B})$ already computed in the previous node X_i.

$$\begin{aligned} \min \quad & f(X_i, \mathcal{B}) + c(A(v)) \\ \text{s.t.} \quad & B(u) = A(u) \quad \forall u \in X_i \\ & A(v) \cap B(u) \neq \emptyset \quad \forall u \in X_i \cap N(v) \end{aligned} \tag{1}$$

with $N(v)$ being the set of vertices neighboring v. The second constraint assures that the vertex v can communicate with all its adjacent vertices u in $G(X_{i+1})$ by sharing at least a common active interface. The first constraint simply states that the new solution $\mathcal{A}$ equals $\mathcal{B}$ for all the vertices except the new one v. The objective function sums the already computed optimum value $f(X_i, \mathcal{B})$ with the cost of the interfaces $A(v)$ activated in the new vertex v.

In any *forget node* $X_{i+1} = X_i \setminus \{v\}$, the value $f(X_{i+1}, \mathcal{A})$ for a specific collection of interface subsets $A(u)$, with $u \in X_i$, and $|A(u)| \leq 2$ is computed by solving the following constrained minimization problem.

$$\begin{aligned} \min \quad & f(X_i, \mathcal{A} \cup B(v)) \\ \text{s.t.} \quad & A(u) \cap B(v) \neq \emptyset \quad \forall u \in X_{i+1} \cap N(v) \\ & B(v) \subseteq W(v) \end{aligned} \tag{2}$$

In fact, the value $f(X_{i+1}, \mathcal{A})$ is essentially the minimum value of $f(X_i, \mathcal{A} \cup B(v))$ for every possible subset of active interfaces $B(v)$ compatible with every $A(u)$, which means $A(u) \cap B(v) \neq \emptyset$ for every u adjacent to v.

At the end of the algorithm, the optimum of *CMI(2)* is the minimum value $f(X_r, \mathcal{A})$, for every possible collection of interface subsets $A(u) \subseteq W(u)$, with $|A(u)| \leq 2$, in the unique vertex $u \in X_r$. We conclude this section computing the complexity of the algorithm. At each *introduce node*, we solve at most $(k + \binom{k}{2})^{(h+1)}$ problems (1), one for every possible collection $\mathcal{A}$. In fact, every possible subset of active interfaces $A(u)$ is such that $|A(u)| \leq 2$, and every $|X_{i+1}| \leq h + 1$. Moreover, for every possible collection $\mathcal{A}$, since in each subproblem there are at most $h + 1$ constraints, its resolution requires $O(h) = O(1)$ time.

Analogously, at any *forget node*, we solve at most $(k + \binom{k}{2})^{(h+1)}$ problems (2), given by all the subsets of active interfaces $A(u)$, with $u \in X_{i+1}$, and all the possible subsets of active interfaces $B(v)$ for the forget vertex v. In conclusion, since there are at most n *introduce nodes* and n *forget nodes*, the time complexity of the dynamic programming algorithm is $O(n(k + \binom{k}{2})^{(h+1)})$. We can then state the following:

Theorem 4. *Given an instance of CMI(2) with a graph G, and a pathwidth decomposition $\mathcal{P}$ of width h for G, CMI(2) is solvable $O(n(k + \binom{k}{2})^{(h+1)})$ time.*

5. Graphs with Bounded Carvingwidth

We start by formally define what the *carvingwidth* [24] of a graph is. Subsequently, we present our new resolution algorithm that optimally solves *CMI(2)* on graphs with bounded carvingwidth in polynomial time.

5.1. Definition of Carvingwidth

We briefly recall some basic definitions before describing the algorithm. Given a graph $G = (V, E)$, let $\{V_1, V_2\}$ be a partition of V, and $(V_1, V_2) \subseteq E$ be the subset of edges with one extreme in V_1 and the other in V_2. Clearly, (V_1, V_2) is an edge-cut of G. Let T be a sub-cubic tree where each leaf corresponds to one vertex in V, and all the internal nodes have degree three (two children). For this particular tree, it is possible to define a specific edge-weight in the following way. For every edge $e \in E(T)$, let T_1 and T_2 be the two subtrees obtained removing e from T. Then, let V_1 and V_2 be the sets of vertices in V corresponding to the leaves in T_1 and in T_2. We set the weight $w(e)$ of e at $|(V_1, V_2)|$, i.e., the number of edges between V_1 and V_2. The tree T is called a *carving* of G, and (T, w) is called a *carving decomposition* of G. The width of (T, w) is the maximum weight $w(e)$ overall $e \in E(T)$. The *carvingwidth* of G, denoted by $cw(G)$, is the minimum width over all carving decompositions of G. For any fixed constant h, if G has carvingwidth at most h then G is said to be of *bounded carvingwidth*. We define $cw(G) = 0$ if $|V| = 1$.

An example of a carving decomposition for the graph in Figure 5 is shown in Figure 6, where the red (additional) edges among the leaves correspond to the edges of the decomposed graph. Each (black) edge of the tree is associated with an integer that is the weight given by the decomposition. The width of this particular carving decomposition is 5, which is the maximum of the weights, and corresponds to the cut $(\{1, 2, 5, 6, 7\}, \{3, 4\})$.

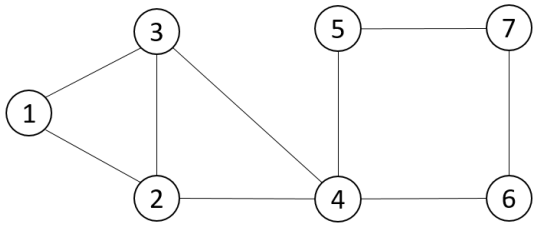


Figure 5. A sample graph admitting the carving decomposition in Figure 6.

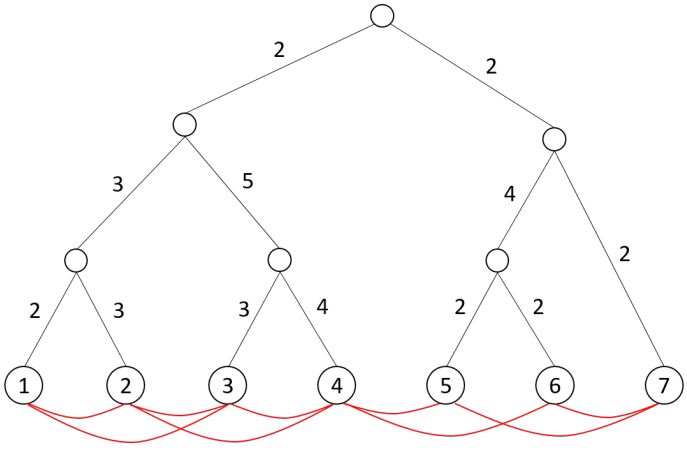


Figure 6. A carving decomposition of width 5 for the graph shown in Figure 5.

5.2. Solving CMI(2) on Graphs with Bounded Carvingwidth

This section describes a polynomial time optimal algorithm for *CMI(2)* on graphs with bounded carvingwidth, which uses the dynamic programming technique.

Given an instance of *CMI(2)*, it is possible in linear time [23,25] to find a carvingwidth decomposition (T, w) for $G(V, E)$ of width h. Denote by *root* the root of T.

We now describe a dynamic programming algorithm to find an optimal solution for *CMI(2)* for G, which exploits the structure and the properties of carving decomposition (T, w). As already done

before for the pathwidth, we will call nodes the vertices of T, to avoid ambiguities between the vertices of the graph and those of T.

Denote by $T(i)$ the subtree induced by the node i of T and all its descendants. Denote also by V_i the leaves of $T(i)$, by V_{i^-} the leaves of T in $V \setminus V_i$, and by $G(i)$ the subgraph of G induced by the vertices in V_i. In the core of the dynamic programming algorithm, we compute the optimal value of a constrained version of *CMI(2)* restricted to the subgraph $G(i)$. The constraint is given by the interfaces active on each node in V_i that is connected with some other nodes in V_{i^-}. In particular, we compute $f(i, \mathcal{A})$, which is the optimum value of *CMI(2)* on $G(i)$, where $\mathcal{A}$ is a collection of interface subsets defined as

$$\mathcal{A} = \bigcup_{u:(u,v)\in(V_i,V_{i^-})} \{A(u)\}, \quad A(u) \subseteq W(u), |A(u)| \leq 2 \tag{3}$$

with the following additional constraints.

- The interfaces active on each vertex u such that $(u, v) \in (V_i, V_{i^-})$ are the ones in $A(u) \in \mathcal{A}$.

Please note that the carvingwidth of G is h, so the maximum number of subsets in $\mathcal{A}$ is exactly h. If i is a leaf corresponding to a vertex $u \in V$, then the following condition holds:

- $f(i, \mathcal{A}) = c(A(u))$.

In fact, $f(i, \mathcal{A})$ is essentially the cost of the active interfaces on the unique vertex u belonging to the subgraph $G(i)$.

If i is an internal node of T with two children j and k, then we compute $f(i, \mathcal{A})$ solving the following minimization problem, where $\mathcal{A}$ has several subsets equal to $|(V_i, V_{-i})|$, as defined in Equation (3). Moreover, there can be edges between $G(j)$ and $G(l)$. If it is the case, the interfaces active on the extremes of these edges must be compatible. The following constrained minimization problem solves $f(i, \mathcal{A})$ for a particular collection $\mathcal{A}$ by using the values $f(j, \mathcal{B})$, and $f(k, \mathcal{C})$ already computed, with $\mathcal{B}$ and $\mathcal{C}$ being the collections defined according to Equation (3).

$$\begin{aligned} \min \quad & f(j, \mathcal{B}) + f(l, \mathcal{C}) \\ \text{s.t.} \quad & A(u) = B(u) \quad \forall u \in V_j, (u, v) \in ((V_j, V_{j^-}) \triangle (V_l, V_{l^-})) \\ & A(u) = C(u) \quad \forall u \in V_l, (u, v) \in ((V_j, V_{j^-}) \triangle (V_l, V_{l^-})) \\ & B(u) \cap C(v) \neq \emptyset \quad \forall (u, v) \in ((V_j, V_{j^-}) \cap (V_l, V_{l^-})) \end{aligned} \tag{4}$$

In fact, by definition of carving decomposition, we have that $V_i = V_j \cup V_l$, $V_j \cap V_l = \emptyset$, and that the edges in (V_i, V_{i^-}) are the ones in $(V_j, V_{j^-}) \cup (V_l, V_{l^-})$ minus the ones in $(V_j, V_{j^-}) \cap (V_l, V_{l^-})$, i.e., $(V_j, V_{j^-}) \triangle (V_l, V_{l^-})$. This means that the collection $\mathcal{A}$ is bipartitioned into two sub-collections, one belonging to $\mathcal{B}$, and the other belonging to $\mathcal{C}$. The former refers the vertices in V_j that are extremes of some edges in (V_i, V_{i^-}), the latter, analogously, refers the vertices in V_l that are extremes of some edges in (V_i, V_{i^-}). This bi-partition leads to the first two constraints in Equation (4), while the third constraint guarantees the compatibility between every couple of subsets $A(u)$, and $A(v)$ in $\mathcal{A}$ such that the edge (u, v) belongs to $G(i)$.

Notice that $\mathcal{A} = \emptyset$ for $f(root, \mathcal{A})$, because V_{root^-} is empty, so there are no edges in (V_{root}, V_{root^-}). This means that the last problem of the algorithm is the following.

$$\begin{aligned} \min \quad & f(j, \mathcal{B}) + f(l, \mathcal{C}) \\ \text{s.t.} \quad & B(u) \cap C(v) \neq \emptyset \quad \forall (u, v) \in ((V_j, V_{j^-}) \cap (V_l, V_{l^-})) \end{aligned} \tag{5}$$

where only the constraint that guarantees the compatibility of the two solutions $\mathcal{B}$ and $\mathcal{C}$ is needed.

Clearly, the optimum of *CMI(2)* in G is the value of $f(root, \mathcal{A})$, which is unique, because $\mathcal{A}$ is empty. The computational time needed to compute $f(i, \mathcal{A})$ for a leaf of T is $k + k^2$, i.e., $O(k^2)$, because $|\mathcal{A}| = 1$, and we essentially try every subset of at most two interfaces for the unique vertex in V_i.

When i is an internal node of T with two children j, and l, we can just combine every collection $\mathcal{B}$ with every collection $\mathcal{C}$ in order to build $\mathcal{A}$, and compute $f(i, \mathcal{A})$. Given two collections $\mathcal{B}$ and $\mathcal{C}$, the check of the feasibility according to Equation (4) costs $O(1)$ since there are at most $O(1)$ edges between $G(j)$ and $G(l)$ that require activation. Moreover, as the cardinality of $\mathcal{B}$ and of $\mathcal{C}$ is at most h, every set $B(u) \in \mathcal{B}$ and every set $C(u) \in \mathcal{C}$ contain at most two interfaces, the time complexity to compute $f(i, \mathcal{A})$ for an internal node is $O(k^{4h})$. Notice that each internal node as exactly two children, and in a perfect binary tree there are $2n - 1$ nodes, then T has at most $2n - 1$ nodes. Concluding, the dynamic programming algorithm to solve *CMI(2)* for a graph with carvingwidth h needs $O((2n - 1)k^{4h})$ time.

We can then state the following:

Theorem 5. *Given an instance of CMI(2) with a graph G, and a carvingwidth decomposition $\mathcal{P}$ of width h for G, CMI(2) is solvable $O(nk^{4h})$ time.*

6. Conclusions

In the context of multi-interface networks, we have investigated a constrained variant of the Coverage problem. Given a graph $G = (V, E)$, the aim is to find the cheapest way to establish all the connections defined by E by activating for each node $v \in V$ a suitable subset of interfaces among those available in v. One further constraint provided as a positive integer p has been considered with respect to the original model. Parameter p specifies how many interfaces a single node can activate at most. The aim of the modification has been to balance the energy consumption among all the nodes of the network, hence prolonging the lifespan of the single nodes. As *CMI(p)* has been proven be much more difficult in general with respect to the basic case of *CMI(∞)*, we keep on investigating whether the problem can be efficiently solved when restricted to specific graph classes. In particular, we have considered graphs with bounded treewidth in general, and graphs with bounded pathwidth or bounded carvingwidth in particular. In these two subcases we could design optimal polynomial time algorithms to solve *CMI(2)*. All the results can be easily extended to *CMI(p)* of any constant $p \geq 1$.

As general main open question on *CMI(p)*, it would be interesting to investigate general graphs in case the input instance is guaranteed to admit a solution, i.e., the complexity of the problem cannot rely on feasibility issues. Moreover, since the completeness proof for the underlying decisional problem holds for graphs with $\Delta \geq 4$, while the problem has been solved for $\Delta \leq 2$ (that is, paths and rings [2]), it remains to show what happens for sub-cubic graphs, i.e., $\Delta \leq 3$. Other directions require investigation into whether *CMI(p)* can be solved on specific graph classes by means of different techniques. Please note that the solutions proposed for the case $p = 2$ can be easily extended to any $p > 2$; however, it is worth investigating whether *CMI(p)* can be solved on specific graph classes in a smarter way. Finally, a generalization of the problem that could cover more realistic cases, is to introduce multi-objective functions, similar to [26], where it is taken into account also global cost constraints. What can be done, for instance, if the underlying graph exhibits specific properties like small-world [27], planarity [28], or a bounded chromatic number [29,30]? How would the problem be affected if the cost function determining the energy spent to activate an interface is related to the bandwidth or to the physical distance among connected devices (see [31])? All such possible investigating directions confirm the general and interdisciplinary nature of the multi-interface networks model.

Author Contributions: Conceptualization, A.A. and A.N.; Methodology, A.A. and A.N.; Software, A.A. and A.N.; Validation, A.A. and A.N.; Formal analysis, A.A. and A.N.; Investigation, A.A. and A.N.; Resources, A.A. and A.N.; Data curation, A.A. and A.N.; Writing—original draft preparation, A.A. and A.N.; Writing—review and editing, A.A. and A.N.; Visualization, A.N.; Supervision, A.N.; Project administration, A.A. and A.N.; Funding acquisition, A.A. and A.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by the European project "Geospatial based Environment for Optimization Systems Addressing Fire Emergencies" (GEO-SAFE), contract no. H2020-691161, by the Italian MIUR PRIN 2017 Project ALGADIMAR "Algorithms, Games, and Digital Markets", and by the Italian National Group for Scientific Computation GNCS-INdAM.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Klasing, R.; Kosowski, A.; Navarra, A. Cost Minimization in Wireless Networks with a Bounded and Unbounded Number of Interfaces. *Networks* **2009**, *53*, 266–275. [CrossRef]
2. Aloisio, A.; Navarra, A. Balancing energy consumption for the establishment of multi-interface networks. In Proceedings of the 41st International Conference on Current Trends in Theory and Practice of Computer Science, (SOFSEM), Pec pod Sněžkou, Czech Republic, 24–29 January 2015; pp. 102–114.
3. Aloisio, A.; Navarra, A.; Mostarda, L. Distributing energy consumption in multi-interface series-parallel networks. In Proceedings of the 5th IEEE AINA International Workshop on Engineering Energy Efficient Internetworked Smart Sensor (E3WSN), Matsue, Japan, 27– 29 March 2019; pp. 734–744.
4. Aloisio, A.; Navarra, A.; Mostarda, L. Energy Consumption Balancing in Multi-Interface Networks. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *2019*, 1–11.
5. Aloisio, A.; Navarra, A. Budgeted Constrained Coverage on Series-Parallel Multi-Interface Networks. In *Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA-2020)*; Springer: Berlin, Germany, 2020; to appear.
6. Bahl, P.; Adya, A.; Padhye, J.; Walman, A. Reconsidering wireless systems with multiple radios. *SIGCOMM Comput. Commun. Rev.* **2004**, *5*, 39–46. [CrossRef]
7. D'Angelo, G.; Di Stefano, G.; Navarra, A. Multi-interface wireless networks: Complexity and algorithms. In *Wireless Sensor Networks: From Theory to Applications*; Ibrahiem, S.R., El Emary, M.M., Ed.; CRC Press: Boca Raton, FL, USA; Taylor & Francis Group: Abingdon, UK, 2013; pp. 119–155.
8. Draves, R.; Padhye, J.; Zill, B. Routing in multi-radio, multi-hop wireless mesh networks. In Proceedings of the of the 10th International Conference on Mobile Computing and Networking (MobiCom), Philadelphia, PA, USA, 26 September–1 October 2004; pp. 114–128.
9. Cavalcanti, D.; Gossain, H.; Agrawal, D. Connectivity in multi-radio, multi-channel heterogeneous ad hoc networks. In Proceedings of the 16th International Symp. on Personal, Indoor and Mobile Radio Communications (PIMRC), Berlin, Germany, 11–14 September 2005; pp. 1322–1326.
10. Caporuscio, M.; Charlet, D.; Issarny, V.; Navarra, A. Energetic Performance of Service-oriented Multi-radio Networks: Issues and Perspectives. In Proceedings of the of the 6th International Workshop on Software and Performance (WOSP), Buenes Aires, Argentina, 5–8 February, 2007; pp. 42–45.
11. D'Angelo, G.; Di Stefano, G.; Navarra, A. Minimize the maximum duty in multi-interface networks. *Algorithmica* **2012**, *63*, 274–295. [CrossRef]
12. Athanassopoulos, S.; Caragiannis, I.; Kaklamanis, C.; Papaioannou, E. Energy-efficient communication in multi-interface wireless networks. *Theory Comput. Syst.* **2013**, *52*, 285–296. [CrossRef]
13. Kosowski, A.; Navarra, A.; Pinotti, C. Exploiting Multi-Interface Networks: Connectivity and Cheapest Paths. *Wirel. Netw.* **2010**, *16*, 1063–1073. [CrossRef]
14. Kosowski, A.; Navarra, A.; Pajak, D.; Pinotti, C. Maximum matching in multi-interface networks. *Theor. Comput. Sci.* **2013**, *507*, 52–60. [CrossRef]
15. Audrito, G.; Bertossi, A.; Navarra, A.; Pinotti, C. Maximizing the overall end-user satisfaction of data broadcast in wireless mesh networks. *J. Discret. Algorithms* **2017**, *45*, 14–25. [CrossRef]
16. D'Angelo, G.; Di Stefano, G.; Navarra, A. Flow problems in multi-interface networks. *IEEE Trans. Comput.* **2014**, *63*, 361–374. [CrossRef]
17. Aloisio, A.; Autili, M.; D'Angelo, A.; Viidanoja, A.; Leguay, J.; Ginzler, T.; Lampe, T.; Spagnolo, L.; Wolthusen, S.; Flizikowski, A.; et al. TACTICS: tactical service oriented architecture. In Proceedings of the 3rd International Conference in Software Engineering for Defence Applications, Rome, Italy, 22–23 September 2014; pp. 1–9.
18. Perucci, A.; Autili, M.; Tivoli, M.; Aloisio, A.; Inverardi, P. Distributed composition of highly-collaborative services and sensors in tactical domains. In Proceedings of the of 6th International Conference in Software Engineering for Defence Applications (SEDA), Rome, Italy, 7–8 June 2018; pp. 232–244.
19. Aloisio, A.; Arbib, C.; Marinelli, F. Cutting stock with no three parts per pattern: Work-in-process and pattern minimization. *Discret. Optim.* **2011**, *8*, 315–332. [CrossRef]
20. Fomin, F.V.; Kratsch, D. *Exact Exponential Algorithms*; Springer: Berlin/Heidelberg, Germany, 2010.

21. Courcelle, B. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Inf. Comput.* **1990**, *85*, 12–75. [CrossRef]
22. Cattell, K.; Dinneen, M.; Fellows M. A Simple Linear-Time Algorithm for Finding Path-Decompositions of Small Width. *Inf. Process. Lett.* **1996**, *57*, 197–203. [CrossRef]
23. Bodlaender, H.L. A linear time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.* **1996**, *25*, 1305–1317. [CrossRef]
24. Belmonte, R.; van't Hof, P.; Kamiński, M.; Paulusma, D.; Thilikos, D.M. Characterizing graphs of small carving-width. *Discret. Appl. Math.* **2013**, *161*, 1888–1893. [CrossRef]
25. Thilikos, D.; Serna, M.; Bodlaender, H. Constructive Linear Time Algorithms for Small Cutwidth and Carving-Width. In Proceedings of the 11th International Conference on Algorithms and Computation (ISAAC), Taipei, Taiwan, 18–20 December 2000; pp. 192–203.
26. Aloisio, A.; Arbib, C.; Marinelli, F. On LP relaxations for the pattern minimization problem. *Networks* **2011**, *57*, 247–253. [CrossRef]
27. Flammini, M.; Moscardelli, L.; Navarra, A.; Pérennes, S. Asymptotically optimal solutions for small world graphs. In Proceedings of the 19th International Conference on Distributed Computing (DISC), Cracow, Poland, 26–29 September 2005; pp. 414–428.
28. Angelini, P.; Eades, P.; Hong, S.; Klein, K.; Kobourov, S.G.; Liotta, G.; Navarra, A.; Tappini, A. Turning cliques into paths to achieve planarity. In Proceedings of the 26th International Symp. on Graph Drawing and Network Visualization GD, Barcelona, Spain, 26–28 September 2018; pp. 67–74.
29. Gavoille, C.; Klasing, R.; Kosowski, A.; Kuszner, L.; Navarra A. On the complexity of distributed graph coloring with local minimality constraints. *Networks* **2009**, *54*, 12–19. [CrossRef]
30. Navarra, A.; Pinotti, C.; Formisano, A. Distributed colorings for collision-free routing in sink-centric sensor networks. *J. Discret. Algorithms* **2012**, *14*, 232–247. [CrossRef]
31. Navarra, A. 3-Dimensional minimum energy broadcasting problem. *Ad Hoc Netw.* **2008**, *6*, 734–743. [CrossRef]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).