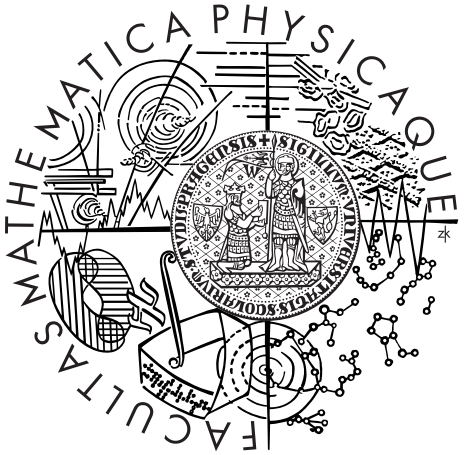
Charles University in Prague
Faculty of Mathematics and Physics

MASTER THESIS



Jitka Novotná

Graph connectivity and resilience

Computer Science Institute of Charles University

Supervisor of the master thesis: Ondřej Pangrác
Study programme: Computer Science
Specialization: Discrete Models and Algorithms

Prague 2015

I want to thank a lot of people who helped me with this thesis. First I want to thank Tomáš Masařík, Jan Kulveit, Tomáš Gavenčiak and Martin Mareš - I'm grateful for their time, support, good advice and also proofreading. Finally I thank my advisor, Ondřej Pangrác, for a lot of patience and effort.

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In ..... date .....

Název práce: Souvislost a resilience grafů

Autor: Jitka Novotná

Ústav: Informatický ústav Univerzity Karlovy

Vedoucí diplomové práce: RNDr. Ondřej Pangrác, Ph.D., Informatický ústav Univerzity Karlovy

Abstrakt: Graf je k-resilientní, jestliže pro každý jeho vrchol existují lokální routovací tabulky takové, že podle nich lze z každého vrcholu dojít do jednoho označeného jako cíl. Existuje domněnka, že k-resilience je ekvivalentní hranové $(k+1)$-souvislosti. Toto dokazujeme pro hranově 3-souvisle grafy a hranově 4-souvislé rovinné triangulace.

Při důkazu používáme nezávislé orientované kostry. Dvě kostry jsou nezávislé, pokud žádnou hranu nepoužijí ve stejné orientaci. Pro $k = 3, 4$ ukazujme, že graf má k nezávislých koster, právě tehdy když je hranově k-souvislý. Kostry hledáme konstruktivně pomocí redukcí částí grafu. Některé redukce mohou být použity i pro obecný k-souvislý případ.

Klíčová slova: graf, podgraf, souvislost, resilience, kostry

Title: Graph connectivity and resilience

Author: Jitka Novotná

Department: Computer Science Institute of Charles University

Supervisor: RNDr. Ondřej Pangrác, Ph.D., Computer Science Institute of Charles University

Abstract: A graph is k-resilient if it is possible to construct local routing tables for each vertex such that we can reach a specified destination vertex from anywhere in the graph. There is a conjecture that k-resilience is equivalent to $(k+1)$-connectivity. We prove this for 3-edge-connected graphs and 4-edge-connected planar triangulations.

In the proof we use independent directed spanning trees. Two spanning trees are independent if they share no common edge with the same direction. For $k = 3, 4$ we show that a graph has k independent spanning trees if and only if it is k-edge-connected. We search for the spanning trees constructively through reductions of parts of the graph. Some of these reductions can also be used in a general k-connected case.

Keywords: graph, subgraph, connectivity, resilience

Contents

Introduction **2**

1 Preliminaries **4**
1.1 Basic definitions 4
1.2 Connected graphs 5
1.3 Routing definitions 5
1.3.1 Independent spanning trees 6

2 2-connected graphs **8**

3 3-connected graphs **10**
3.1 Three independent spanning trees guarantee 2-resilience 10
3.2 Reductions 11
3.2.1 Nontrivial k-cut reduction 11
3.2.2 Useless edge reduction 12
3.2.3 Odd edge reduction 13
3.2.4 Edge near the destination reduction 14
3.3 3-connected graphs are 2-resilient 15

4 4-connected graphs **16**
4.1 4-connected 4-regular graphs have four independent spanning trees 16
4.1.1 Vertex adjacent to destination reduction 16
4.1.2 Proof 17
4.2 4-connected graph has four independent spanning trees 18
4.2.1 Multi-edge reduction. 18
4.2.2 Reduction of vertex with four different neighbors 19
4.2.3 Proof 20
4.3 Grids are 3-resilient 20
4.4 Planar triangulations are 3-resilient 21
4.4.1 Vertex level 22
4.4.2 Types of vertices 23
4.4.3 Graphs without Type 0 vertices 23
4.4.4 Routing tables 24
4.4.5 Routing tables are correct 25

Conclusion **27**

Bibliography **29**

Introduction

The thesis examines resilience of graphs and resilient routing. That is a property closely related to graph connectivity, and it can even lead to an equivalent definition of connectivity.

The main problem studied in this thesis is whether a k-edge-connected graph has local resilient routing tables. This problem is related to the problem of existence of independent spanning trees.

Our goal is to construct tables that are inspired by packet routing. We are given a graph and one vertex marked as the destination. When a packet is sent through an edge to a vertex, routing table for that vertex tells us where to send the packet next. (The formal definitions are given in Section 1.3)

For connected graphs, it is easy to design tables such that each packet reaches the destination. But what happens if some edges of the graph are removed? A k-edge-connected graph remains connected even if $k-1$ edges are removed, so there is a possibility to reach the destination. The hard part is that the tables have to be designed in advance and they are local, i.e., they do not see where are the removed edges.

Our question is whether it is possible to construct routing tables that work even if l edges are removed. If so, we say that graph is l-resilient. There is a conjecture stating that every k-edge-connected graph is $(k-1)$-resilient.

We learned about this problem from Mario Szegedy during a visit of Rutgers University Research for Undergraduates Experience program 2013. He told us about the problem, the conjecture, and an idea that three independent spanning trees are sufficient for 2-resilience.

Previous work

There is a conjecture that every k-edge-connected graph has k independent spanning trees. Khuller and Schieber [8] published a proof of this conjecture which was shown to be incorrect. Gopalan and Ramasubramanian [7] show the correct construction for $k = 3$. We present a simpler proof for $k = 3$ and generalize it for $k = 4$.

It is already know that every k-edge-connected graph has at least $\lfloor \frac{k}{2} \rfloor$ independent spanning trees according to Tutte [1]. There are more known results on independent spanning trees and their generalization [2, 3, 4].

Chiesa and Nikolaevskiy [9] also try to solve the routing problem. Chiesa proves that every k-vertex- connected chordal graph is $(k-1)$-vertex-resilient. Nikolaevskiy proves that modified grids are 3-resilient. By a grid, he means a finite rectangular grid with additional link around the borders. We present his unpublished proof in Section 4.3.

Outline

In the thesis, we present several results. Most of our results hold on both simple graphs and multigraphs. We usually consider k-edge-connectivity and write k-connectivity for short.

In the first chapter, we present an overview of the basic definitions from graph theory, and present our definitions for routing. In the second chapter we construct two independent spanning trees for 2-edge-connected graphs, and using them, we construct 2-resilient routing tables. Third chapter contain analogous result for number 3. Chapter four is dedicated to 4-connected graphs. We construct four independent spanning trees for 4-connected graphs and routing tables for same classes of 3-connected graphs.

1. Preliminaries

1.1 Basic definitions

This section provides basic definitions which we use in the thesis. These definitions are based on B. Bollobás Modern graph theory

A *simple graph* $G = (V, E)$ consists of a set $V(G)$ of n *vertices* and a set $E(G)$ of m 2-element subsets $e = (u, v)$ of $V(G)$ called *edges*. The number of edges containing a vertex v is the *degree* of v. If the degree of each vertex in G is k, then G is *k-regular*. If $k = 3$, then G is *cubic*.

A *multigraph* G consists of a set $V(G)$ of vertices and a multiset $E(G)$ of edges, which are 1 or 2-element subsets of $V(G)$. One element edge is called a *loop*. Edges which are in $E(G)$ multiple times are called *multiple* or *parallel* edges.

A *directed graph* G consists of a set $V(G)$ of vertices and a set $E(G)$ of tuples $\overrightarrow{(u, v)}$, which are directed from vertex u to vertex v ($u \neq v$). Number of edges which lead to v is called the *indegree* and number of edges which lead from v is called the *outdegree*.

The following definitions are written for simple graphs, but they can be easily applied to multigraphs or directed graphs as well.

Let $G = (V, E)$ be a graph. Two vertices u, v are *adjacent* if $(u, v) \in E$. Two edges $(u_1, v_1), (u_2, v_2)$ are adjacent if $|\{u_1, u_2, v_1, v_2\}| \leq 3$. Vertex v is *incident* with edge $e = (u_1, v_1)$ if $v = u_1$ or $v = v_1$.

A *walk* in a graph is a finite or infinite sequence of vertices, in which every two consecutive vertices are adjacent. In *directed walk*, all edges are directed along the sequence from the first vertex towards the last. A *trail* is a walk in which all edges are distinct. A *path* uv is a trail from u to v where all vertices except the first and the last are distinct. A *cycle* is a path which starts and ends with the same vertex.

A graph $H = (V(H), E(H))$ is a *subgraph* of graph $G = (V(G), E(G))$ if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G))$. We will use $G \setminus \{e_1, e_2 \dots e_k\}$ instead of $((V(G), E(G) \setminus \{e_1, e_2 \dots e_k\})$. Subgraph is *induced* if there is no edge $(u, v) \in E(G)$, $u, v \in V(H)$ and $(u, v) \neq E(H)$.

A graph is *connected* if there exists a path between any two vertices. A connected *component* is a maximal connected subgraph of the graph.

In a graph G, *contraction of an edge* e with endpoints u, v is the replacement of u and v with a simple vertex w such that edges incident with w are the edges incident with u or v except the edge e. Contraction can result in multi-edges. The resulting graph $G/\{e\}$ has one less edge than G even in multigraphs. *Contraction of a subgraph* H is contraction of all edges $e \in H$, with resulting graph G/H.

A *plane* graph, or a *planar embedding* of a graph, is a graph drawn on a plane in such a way that its edges are plane curves and intersect only at their endpoints - vertices. A plane graph divides the plane into regions, called *faces*. A graph is *planar* if it has a planar embedding.

A *tree* T is a connected graph which does not contain any cycles. A directed tree is *rooted* with *root* r if all edges are directed towards r. A *spanning tree* T of an undirected graph G is a subgraph that includes all of the vertices of G, and is

	vertex 1		
0	0	2	3
2	0	2	3
3	2	3	0

	vertex 2		
0	0	3	1
1	0	3	1
3	1	0	3

	vertex 3		
0	0	2	1
1	0	2	1
2	0	2	1

Table 1.1: Routing tables for K_4. (0 is the destination)

a tree. A spanning tree can be also rooted. A graph is a *forest* if all its connected component are trees.

1.2 Connected graphs

A graph is *k-edge-connected* if it remains connected after removing any $k-1$ edges. A *k-edge-cut* is a set of k edges whose removal produces a subgraph which is not connected. A 1-cut is called a *bridge*. A k-cut is *minimal* if none of its proper subsets is a cut.

A graph is *k-vertex-connected* if it remains connected even after removing any $k-1$ vertices. A *k-vertex-cut* is a set of k vertices whose removal produces a subgraph which is not connected. A k-cut is *minimal* if none of its proper subsets is a cut. A 1-cut is called *articulation*.

Note Whenever we say that a graph is k-connected, or that it has a k-cut, we mean that it is k-edge-connected, has a k-edge-cut, and so on.

We also use a different definition of k-connectivity based on disjoint paths. A well-known theorem first proved in [5], shows that this definition is equivalent to the first one.

Theorem 1 (Manger's theorem)**.** *A graph is k-connected if there are k disjoint paths between every two vertices.*

Observation 2. *Contraction of k-connected graph is k-connected.*

Proof. Every path from the original graph remains connected after contraction of any edge. So by Manger's theorem contraction of k-connected the graph is k-connected. □

Observation 3. *If C is a minimal k-cut then $G \setminus C$ has exactly two components.*

1.3 Routing definitions

Let G be a graph with one vertex marked as *destination*, or just d.

A *routing table* for vertex v is a table with rows for every incident edge e. In every row are edges adjacent to e.

For edges $e_1, \ldots, e_k$ of a graph G, a *ragged graph* G' is $G \setminus \{e_1, e_2 \ldots e_k\}$.

In the e's row are sorted edges which are *recommend* for moving from v. Let f be the first recommended edge which is in ragged graph G'. Edge f is a *prescribed move* from edge e in G'.

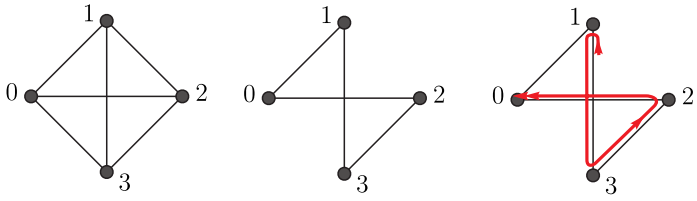


Figure 1.1: K_4; ragged K_4'; prescribed walk from vertex 1.

A *prescribed walk* in G' starting at an arbitrary edge is a sequence of prescribed moves.

We say that graph G is *k-resilient* if there are routing tables such that for every ragged graph obtained from G by deleting k edges, and for every starting edge in both direction, there exists a prescribed walk which reaches the destination.

Note The routing table for vertex v and incoming edge (u, v) recommend edges (v, w_i). It is also possible to say that routing table recommend vertices w_i when prescribed tour came from vertex u. The definition is formally for edges but we can use vertices instead. There is an example of routing tables in 1.1 which are described by vertices.

1.3.1 Independent spanning trees

Independent spanning trees are directed spanning trees which are rooted in the destination and do not use the same edge of G in the same direction. We often assign a color to these trees. An example of three independent spanning trees is in Figure 1.2.

Note Whenever we say that a spanning tree we mean directed spanning tree rooted in the destination.

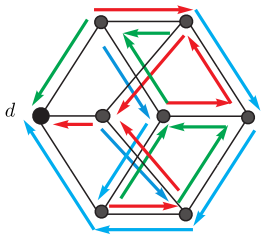


Figure 1.2: Independent spanning trees in a cube.

Lemma 4. *Let G be a k-regular graph with k independent spanning trees, then exactly one spanning tree uses an edge incident with the destination, and exactly two spanning trees use the other edges.*

Proof. The graph G is k-regular with n vertices, so it has $\frac{k \times n}{2}$ edges. k of them incident with destination could be in one tree and the remaining edges could be in two trees So there is $k \times n - k$ possible places for edges of the spanning trees.

Each spanning tree uses $n-1$ edges. All k of them thus use $k \times n - k$ edges, so every possible place is used. $\square$

2. 2-connected graphs

Our proves that 2-connected and 3-connected graphs have 1-routing and 2-routing have the same approach. This chapter we prove the 2-connected case and also prepares ideas for the 3-connected case.

To prove it we show that there are routing tables for which the prescribed walk reaches the destination even if one edge of the graph is removed. We start by the following well-known lemma.

Lemma 5 (Ear decomposition of edge 2-connected graph)**.** *Multigraph G is 2-connected if and only if it can be created iteratively: First, an arbitrary cycle from G is chosen. Then, in each step, we add either all the paths between two vertices, or a cycle touching one vertex.*

Proof. A graph created in this way is 2-connected because a bridge cannot be created.

We show the construction for 2-connected graph G. Let G_0 be an arbitrary cycle of G. Graph G_i is created from G_{i-1} by adding a path or cycle of edges of $G \setminus G_i$. Now suppose for a contradiction that we cannot add any more paths or cycles to G_i, and $G_i \neq G$. Let us focus on an edge $(u, v), u \in V(G_i), v \in V(G \setminus G_i)$. By definition, $G \setminus \{(u, v)\}$ is connected and there is a path P between u and v which does not contain the edge (u, v). We can take the subpath P' that starts at vertex v and ends in vertex w – first vertex of G_i. $P' \cup (u, v)$ is a path which we can be added to G_i. A contradiction. □

Theorem 6. *Every 2-connected multigraph is 1-resilient.*

Proof. We prove this by using two *independent spanning trees* T_1, b, that is, two directed spanning trees which are rooted in the destination and do not use any edge of G in the same direction.

We create routing tables for a vertex v in the following way:

- When something is send to v from an edge which is not in T_1 nor T_2, we recommend an edge of the spanning tree T_1 which leads from v. As a second option, we recommend an edge of T_2 which leads from v.
- When something is send to v from an edge of T_1, we recommend an edge of T_1 which leads from v. As a second option, we recommend an edge of T_2 which leads from v.
- When something is send to v from an edge of T_2, we recommend an edge of T_2 which leads from v. As a second option, we recommend an edge of T_1 which leads from v.

Using these tables, we prescribe a walk which reaches the destination in every ragged graph. The path starts by using one spanning tree and if it encounters a removed edge, switches to the second spanning tree and subsequently reaches the destination.

Graph G constructed by earing lemma starts by cycle which contains the destination. We will prove that there are such spanning trees by induction on the number of added paths.

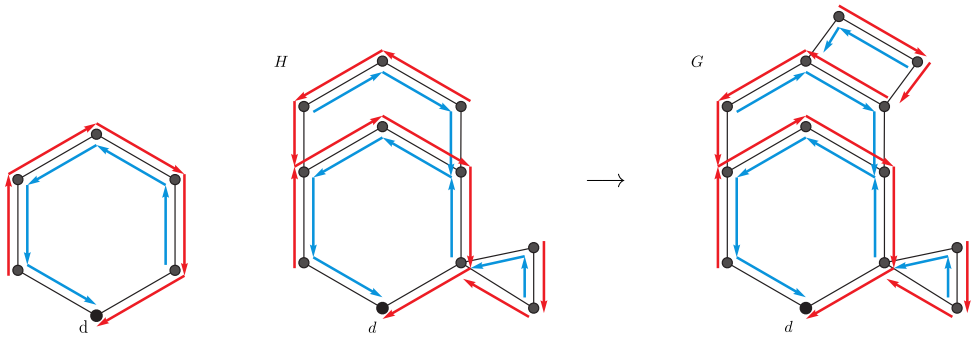


Figure 2.1: Spanning trees in cycle; induction step

If G is just a cycle, then its spanning trees are two maximal paths, as in Figure 2.1. Suppose that G was created from a graph H by adding a path or cycle P. Then the destination is not contained in the path P, because the destination lies in the cycle. The graph H contains, using the induction step, two independent spanning trees. The path P also contains two spanning trees similarly to the case for a cycle. We combine these spanning trees into spanning trees of G as depicted in Figure 2.1. □

3. 3-connected graphs

Every 3-connected graph is 2-resilient. We construct the routing tables using three independent spanning trees

We prove that every 3-connected graph has a three independent spanning trees using the mathematical induction on the number of edges. We found that every 3-connected graph has a certain attribute and it can be reduced to a smaller graph or graphs. The new graphs have three independent spanning trees and they could be extended to the independent spanning trees of the original graph.

The most important reduction is the reduction of nontrivial 3-cut. The 3-cut is trivial when it separates only one vertex. It is possible reduce edge (u, v) if $deg(u) \geq 3$ and $deg(v) > 3$. Considering this reduction is sufficient to find three independent spanning trees for cubic 3-connected graphs. Cubic 3-connected graphs are either small or there is some structure near the destination which could be reduced.

3.1 Three independent spanning trees guarantee 2-resilience

There is a way how to find routing tables for 2-resilience using three *independent spanning trees*.

Theorem 7. *If a multigraph has three independent spanning trees, then it is 2-resilient.*

Proof. Let G be a graph with three independent spanning trees. We show that a prescribed walk reaches the destination even if two arbitrary edges were removed.

We create the routing table for a vertex v in the following way:

- When something is send to v from an edge which is not contained in any independent spanning tree, then we recommend a red edge.
- When something is send to v from an edge which is contained in some of the independent spanning trees, then we recommend an edge of the *same* independent spanning tree.
- If a prescribed edge of the *red* tree was removed, then we recommend a *green* edge.
- If a prescribed edge of the *green* tree was removed, then we recommend a *blue* edge.
- If a prescribed edge of the *blue* tree was removed, then we recommend a *red* edge.

We show that every prescribed walk ends in the destination. Starting edge was red or not contained in any independent spanning tree. If some red edge e on the red path was removed, then the walk continues on a green path. The green

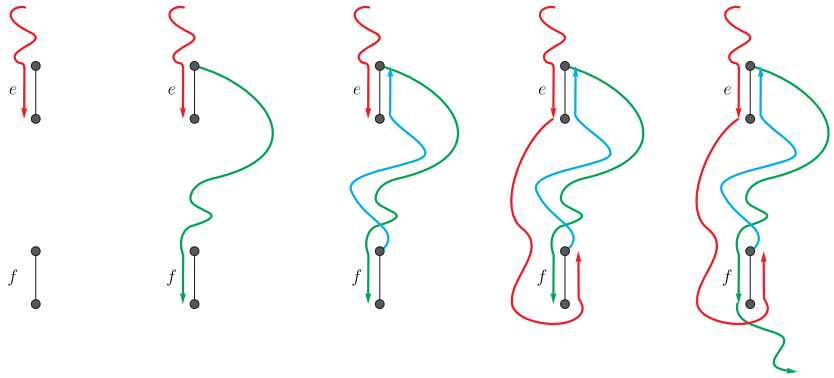


Figure 3.1: A prescribed walk in a graph with three independent spanning trees.

path cannot use the edge e, because one of its direction is already occupied by the red tree and the other direction could create a green cycle. Either the green path reaches the destination or some green edge f was removed. In the latter case, we switch to the blue path. The blue path cannot use an edge f for the same reason as the green path could not use the edge e. The blue path either reaches the destination or it uses an edge e in the opposite direction then the red path did. In this case, we continue on the red path as if the edge e was not removed. The red path either reaches the destination or an edge f. In the second case, we continue on the green path and this time we have to reach the destination. □

3.2 Reductions

We will prove that every 3-connected graph has three independent spanning trees using so-called *reductions*.

Reductions use the following general idea If a graph G has some attribute, then we use it to construct a smaller k-connected graph or graphs. These reduced graphs have k independent spanning trees by mathematical induction on the number of edges. There are only a few possibilities how these trees use some specified subgraph of the graph. For each of these possibilities we construct k independent spanning trees for G.

In the Section 3.3, we prove that for every graph with more than four vertices we can use at least one of the following reduction.

3.2.1 Nontrivial k-cut reduction

A k-cut is *nontrivial* if it separates more than one vertex.

Reduction 1 (k-cut)**.** *Let G be k-connected graph with nontrivial k-cut – set of edges (u_i, v_i) for $i \in (1 \dots k)$. A k-cut divides graph into two components C_1, C_2, where C_1 contains the destination.*

The graph G_{k-red1} is created from G by a contraction of subgraph C_2 into the vertex p. The graph G_{k-red2} is created from G by a contraction of subgraph C_1 into the vertex d and mark this vertex as the destination.

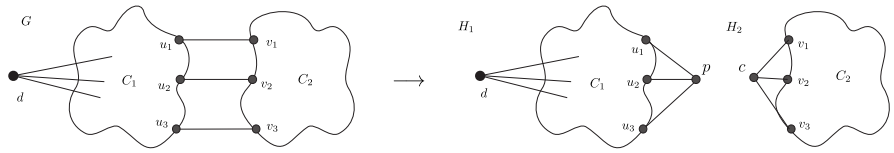


Figure 3.2: 3-cut reduction

Reduction lemma 8 (k-cut)**.** *If a k-connected multigraph G has a nontrivial k-cut then G_{k-red1} and G_{k-red2} are smaller and k-connected. If G is k-regular then G_{k-red1} and G_{k-red2} also k-regular. If G_{k-red1} and G_{k-red2} graphs have k independent spanning trees then G also has k independent spanning trees.*

Proof. G is k-connected so the k-cut has to be minimal and separate graph into exactly two components.

Let $H_1 = G_{k-red1}$ and $H_2 = G_{k-red2}$. The graphs H_1, H_2 are k-connected, because they were created from a k-connected graph by a contraction. The graphs H_1, H_2 have fewer edges then G because the k-cut was nontrivial. If G was k regular then vertices of H_1, H_2 which was in G has same degree and newly created vertices has degree k i.e. edges which was in k-cut so H_1, H_2 are k-regular.

First, we color directed edges of G using independent spanning trees of H_1, H_2. Then we prove that every colored path ends in the destination and every vertex has exactly one outgoing edge of each color. Hence, these colors determine independent spanning trees of G.

Edges in C_1 are colored by the same colors as they have in H_1. Edges $\overrightarrow{T(u_i, v_i)}$ are colored by the same colors as edges $\overrightarrow{(u_i, p)}$ in H_1. We rename colors in H_2 such that the edges $\overrightarrow{(d, v_i)}$ have the same color as the edges $\overrightarrow{(u_i, p)}$. Note that these edges had only one direction colored in H_2. To edges in subgraph C_2 we assign the same color as they have in H_2.

We show that from every node v we can use edges of the same color to reach the destination. For example red color. If $v \in C_2$, then there is a red path from v to d in recolored H_2. This path leads to u_i and then using the red path in H_1 to the destination. If $v \in C_1$, then there was a red path in H_1. The problem is that this path can include the vertex p. In this case, the path uses the edge (u_i, p) which was replaced by edge (u_i, v_i) and then continues via the red path from v_i as described above. Hence, colored edges denote k independent spanning trees of G. □

3.2.2 Useless edge reduction

An edge $e = (u, v)$ of the multigraph G is *useless* if $\deg(u) > k$ and $\deg(v) > k$.

Reduction 2 (Useless edge)**.** *Let G be a multigraph. The graph $G_{ue-red1}$ is $G \setminus e$.*

Reduction lemma 9 (Useless edge)**.** *If a graph G contains useless edge e and has not nontrivial k-cut then $G_{ue-red1}$ is smaller and k-connected. If $G_{ue-red1}$ graphs have k independent spanning trees then G also has k independent spanning trees.*

Proof. First we will show that $H = G \setminus \{e\}$ is k-connected. If H has a $(k-1)$-cut C, then $C \cup \{e\}$ is a k-cut in G. A set $C \cup \{e\}$ is a nontrivial k-cut because

the vertex in a trivial k-cut has degree k. Independent spanning trees of G are exactly the same spanning trees as in $G_{ue-red1}$. □

3.2.3 Odd edge reduction

This reduction is for 3-connected graphs.

We call an edge $e = (u, v)$ *odd* if $\deg(u) = 3$ and $\deg(v) \geq 4$. Neighbourhood of u is $\{v, w_1, w_2\}$.

Reduction 3 (Odd edge)**.** *Let G be a multigraph. The $G_{odd-red}$ is the multigraph which is obtained from $G \setminus \{e\}$ by replacing a vertex u and two incident edges by an edge (w_1, w_2).*

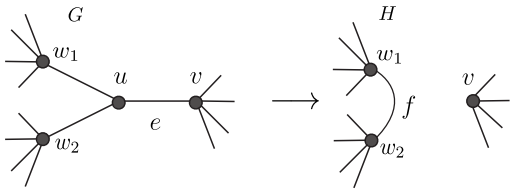


Figure 3.3: Odd edge reduction.

Reduction lemma 10 (Odd edge)**.** *If a multigraph G contains an odd edge and has not nontrivial 3-cut than $G_{odd-red}$ is smaller and 3-connected. If $G_{odd-red}$ has three independent spanning trees, then G also has three independent spanning trees.*

Proof. The graph $H = G_{odd-red}$ is 3-connected for a similar reason as in the previous reduction. If H has a 2-cut $\{f, g\}$, then $\{e, f, g\}$ is a 3-cut in G. If the set $\{e, f, g\}$ is trivial 3-cut either vertex u or v has degree three and it is incident with all edges in the cut. Vertex v has larger degree. Vertex u is not in H and so edges f, g are not incident with u and H is 3-connected.

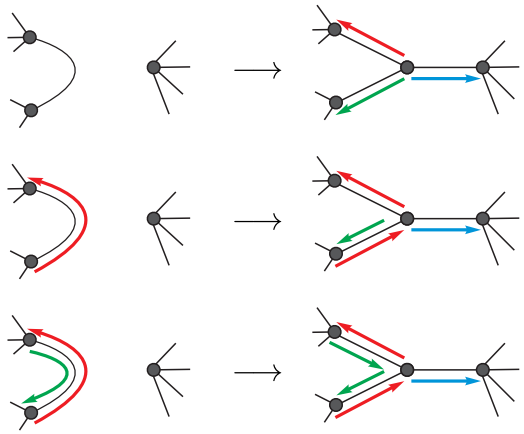


Figure 3.4: Independent spanning trees near an odd edge.

We will color directed edges of G. Each color determines one spanning trees. All edges of $G \cap H$ get the same color as in H. Edges $\overrightarrow{w_1, w_2}, \overrightarrow{w_2, w_1}$ are divided into two parts with same color and direction. Then we add leaf edges from v to all colored subgraphs that do not already contain it. Fortunately, there is enough free edges to add it. See Figure 3.4. Colored spanning trees are independent. □

3.2.4 Edge near the destination reduction

This reduction is for 3-connected graphs.

We call an edge $e = (u, v)$ *near the destination* when u is adjacent with the destination and v is not.

Reduction 4 (Edge near the destination)**.** *Let G be a multigraph. The G_{ed_red} is the multigraph which is obtained from $G \setminus \{e\}$ by replacing vertices u, v and incident edges by edges f, g.*

Reduction lemma 11 (Edge near the destination)**.** *If a cubic 3-connected multigraph G has an edge e near the destination and has not nontrivial 3-cut than G_{ed_red} is smaller, cubic and 3-connected. If G_{ed_red} has three independent spanning trees, then G also has three independent spanning trees.*

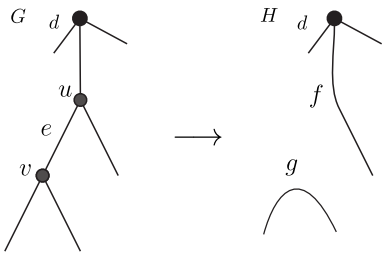


Figure 3.5: Edge near the destination reduction.

Proof. The graph $H = G_{ed\_red}$ is 3-connected for a similar reason as in previous reductions If H has a 2-cut $\{f', g'\}$, then $\{e, f', g'\}$ is a 3-cut in G. If set $\{e, f', g'\}$ is trivial 3-cut either vertex u or v is has degree 3 an it is incident with all edges in cut. Vertices u, v are not in H and so edges f', g' do not incident with u or v. So the graph H is 3-connected.

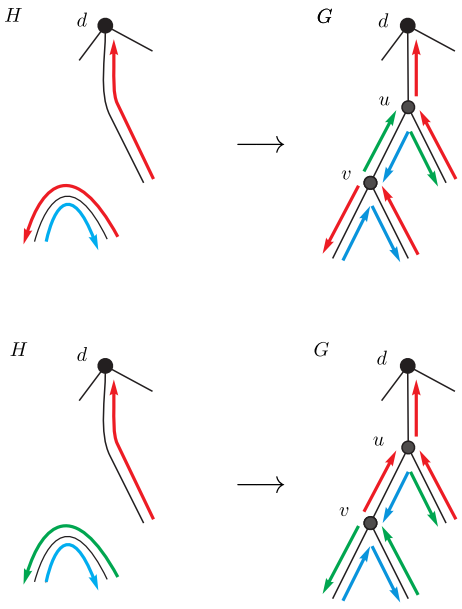


Figure 3.6: Independent spanning trees near the destination.

We will color directed edges of G. Each color determines one spanning trees. All edges of $G \cap H$ get the same color as in H. Edges $\overrightarrow{f}, \overleftarrow{f}, \overrightarrow{g}, \overleftarrow{g}$ are divided into two parts with same color and direction. Then we add leaf edges from u and v to all colored subgraphs that do not already contain it. Fortunately, there is enough free edges to add it. See Figure 3.6. Colored spanning trees are independent. □

3.3 3-connected graphs are 2-resilient

Theorem 12. *Every 3-connected multigraph G is 2-resilient.*

Proof. We prove the statement by induction on the number of edges.

First of all, if the graph has a nontrivial 3-cut, then we use the nontrivial 3-cut reduction. If G contains an edge $e = (u, v)$ such that $\deg(u) \geq 4$ and $\deg(v) \geq 4$ we use the useless edge reduction. If G contains an edge $e = (u, v)$ such that $\deg(u) = 3$ and $\deg(v) \geq 4$ we use the odd edge reduction. The graph G has to be cubic.

If the graph has a multi-edge then there is 2-cut around it.

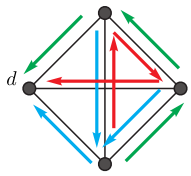


Figure 3.7: Independent spanning trees in K_4.

If $G = K_4$, then it has three independent spanning trees, see Figure 3.7. If $G \neq K_4$ and G is cubic then it has a *edge near the destination* and we can reduce it.

There is a reduction for every graph. So there are three independent spanning trees. These spanning trees determinate routing tables according Theorem 7. □

4. 4-connected graphs

We present several results for 4-connected graphs. First, we show short prove that 4-connected 4-regular graphs have four independent spanning trees. Then we show prove for all 4-connected graph. Than we show that same plane graph are 3-resilient by founding routing tables.

4.1 4-connected 4-regular graphs have four independent spanning trees

The proof is quite similar to the proof that 3-connected graph has three independent spanning trees. We give a following proof because it uses a nice idea of double-count.

To prove this we use k-cut and useless edge reduction showed in sections 3.2.1, 3.2.2 and one more reduction.

4.1.1 Vertex adjacent to destination reduction

Let (u, v) be an edge where u is adjacent with the destination and v is not.

See Figure 4.14 for a notation. Vertex v is incident with four edges which we called $f_1 = (u, v), f_2 = (u, d), e_1 = (u, w_1), e_2 = (u, w_2)$. Destination is also incident with edges g, h different from f_2. In G could be edges which use neither destination nor its neighborhood. If such edges exist we called them i, j.

Reduction 5 (Vertex near the destination)**.** *Let G be a multigraph. Then $G_{ve\_d\_red} = G \setminus \{e_1, e_2, f_1, f_2\} \cup \{e = (d, v), f = (w_1, w_2)\}$. I.e graph where vertex u is replaced by edges f, g.*

Lemma 13. *$G_{ve\_d\_red}$ is 4-connected.*

Proof. Recall that G has not nontrivial 4-cut.

The dispute, assume that $H = G_{ve\_d\_red}$ has 2-cut. Without loss of generality, we can assume that 2-cut uses edges which are shown in Table 4.1. Each of these corresponds to 3-cut or 4-cut in G. In every 4-cut, two edges contain vertex v so 4-cut is nontrivial. Contradiction.

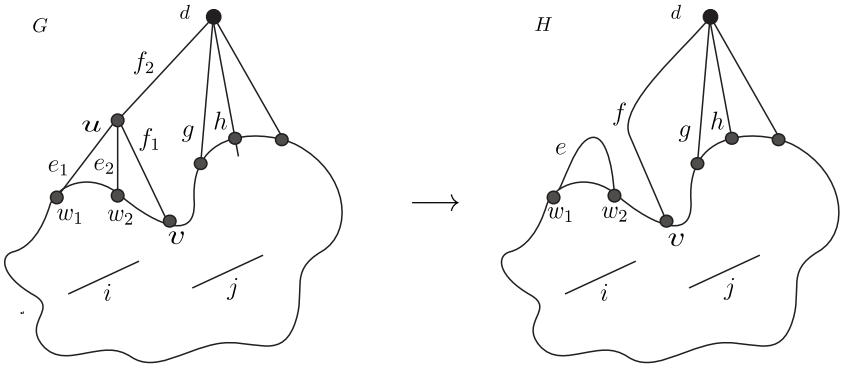


Figure 4.1: Reduction of vertex with degree 4.

2-cut in H	cut in G
e, f	e_1, e_2, f_1
e, g	e_1, e_2, g
f, g	f_1, f_2, g
e, i	e_1, e_2, i
f, i	f_1, f_2, i
g, i	e_1, e_2, g, i
g, h	e_1, e_2, g, h
i, j	e_1, e_2, i, j

Table 4.1: The 2-cuts in G' and corresponding cuts in G.

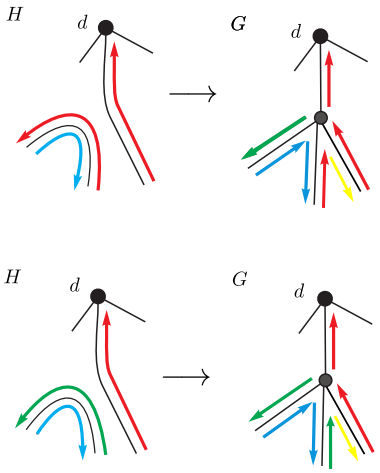


Figure 4.2: Independent spanning trees near destination.

Now suppose that H has 3-cut c_1, c_2, c_3 which is minimal. We will choose an arbitrary component C of $H \setminus \{c_1, c_2, c_3\}$ and count sum of degrees in it. First, we count internal edges and cut — it is $2 \times (\#$ edges in $C) + 3$ which is odd. But G and H is 4-regular and so it is $4 \times (\#$ vertices in $C)$ which is even. Contradiction. □

Reduction lemma 14 (Vertex near the destination)**.** *If a 4-regular 4-connected multigraph G has an edge (u, v) as described above and has not nontrivial 4-cut and $H = G_{ve\_d\_red}$ has four independent spanning trees, then G also has four independent spanning trees.*

Proof. We will color directed edges of G. Each color determines one spanning tree. All edges of $G \cap H$ get the same color as in H. Edges $\overrightarrow{e}, \overleftarrow{e}, \overrightarrow{f}, \overleftarrow{f}$ are divided into two parts with same color and direction. See Figure 4.2. Colored spanning trees are independent. □

4.1.2 Proof

Theorem 15. *Every 4-connected 4-regular multigraph has four independent spanning trees.*

Proof. We prove the statement by induction on the number of edges.

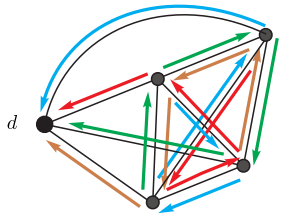


Figure 4.3: Four independent spanning trees in K_5.

First of all, if the graph has a nontrivial 4-cut, then we use the nontrivial 4-cut reduction. K_5 has four independent spanning trees, see Figure 4.7. Every G other then K_5 has an edge (u, v) where u is adjacent with destination and v is not and we use Edge near the destination reduction.

There is a reduction for every graph. So there are four independent spanning trees. □

4.2 4-connected graph has four independent spanning trees

The proof that every 4-edge-connected graph has four independent spanning trees also uses the idea of the reduction. It is possible to reduce nontrivial 4-edge-cut and multiedge. When we do it carefully it is possible reduce vertex with four incident edges into just two edges.

4.2.1 Multi-edge reduction.

Reduction 6 (Multi-edge)**.** *Let G be a 4-connected multigraph with multiple edge $(u, v), \deg(u) = 4$. Then G_{me-red} is created by contradiction (u, v).*

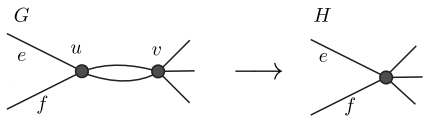


Figure 4.4: Multi-edge reduction.

Reduction lemma 16 (Multi-edge)**.** *If a 4-connected multigraph G contains multi-edge $(u, v), \deg(u) = 4$ then G_{me-red} is smaller 4-connected graph. If G_{me-red} has four independent spanning trees then G also has four independent spanning trees.*

Proof. The graph $H = G_{me-red}$ is 4-connected and smaller because it was created from a 4-connected graph by a contraction.

We will color directed edges of G. Each color determines one spanning trees. All edges of $G \cap H$ get the same color as in H. Few edges will be added. See figure 4.5.

Vertex v had edges e, f other then multi-edge.

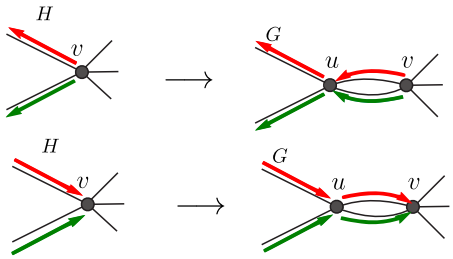


Figure 4.5: Independent spanning trees near a multi-edge.

At most two colors *leave* $u = v$ in H using edges e, f. We add these colors to multi-edge $\overrightarrow{(v,u)}$. At most two spanning trees *enter* $u = v$ in H using edges e, f. We add these colors to multi-edge $\overrightarrow{(u,v)}$ Then we add a leaf edge from u and v to all colored subgraphs that do not already contain it. Colored subgraphs are spanning trees and are independent. □

Note If u or v is the destination, reduction works in the same way. Just the goal cannot be added as a leaf node.

4.2.2 Reduction of vertex with four different neighbors

Reduction 7 (Degree four)**.** *Let G be a 4-connected multigraph without nontrivial 4-cut with vertex v which has four neighbours a, b, c, d.*

$$G_{4-red1} = G \setminus \{v\} \cup \{(a,b),(c,d)\} \tag{4.1}$$
$$G_{4-red2} = G \setminus \{v\} \cup \{(a,c),(b,d)\} \tag{4.2}$$

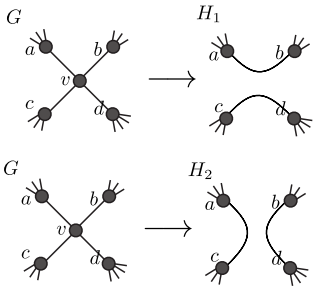


Figure 4.6: Reduction of the vertex with four different neighbors.

Lemma 17. *At least one of graphs G_{4-red1}, G_{4-red2} is 4-connected.*

Proof. Let $H_1 = G_{4-red1}, H_2 = G_{4-red2}$. We start with observation that H_1 and H_2 are at least 3-connected.

If there is a 2-cut $\{f, g\}$ in H_i and not in G, this 2-cut places one or two vertices from $\{a, b, c, d\}$ in separate components. If the 2-cut separates a from b, c, d then $\{f, g, (v, a)\}$ is 3-cut in G. If the 2-cut separates a, b from c, d then $\{f, g, (v, a), (v, b)\}$ is nontrivial 4-cut in G.

For contradiction, suppose that none of H_1, H_2 is 4-connected. They had to have 3-cut which is minimal.

We define auxiliary graph P. 3-cut in H_1 divide vertices into groups V_{1a}, V_{1b} and 3-cut in H_2 divide vertices into groups V_{2a}, V_{2b}. P has four vertices v_1, v_2, v_3, v_4 which are created by contraction:

$$\begin{aligned} v_1 &= V_{1a} \cap V_{2a} \\ v_2 &= V_{1a} \cap V_{2b} \\ v_3 &= V_{1b} \cap V_{2a} \\ v_4 &= V_{1b} \cap V_{2b} \end{aligned}$$

P has at most six edges because all edges in P are included in mentioned cuts. P is not K_4 because K_4 is 4-connected.

Without loss of generality there is not edge between v_1 and v_2. P was created from a 3-connected graph by contraction so it is also 3-connected and there are three edge-disjoint paths between v_1 and v_2. Each of them is exactly two edges long. If three paths use v_3 then v_4 is not connected with the rest of the graph. If two paths use v_3 and one use v_4 then there is 2-cut around v_4. Contradiction. □

Reduction lemma 18 (Degree four)**.** *If a 4-connected multigraph G with none nontrivial 4-cut has vertex v with degree 4 G_{4-red} has four independent spanning trees, then G also has four independent spanning trees.*

Proof. All edges of $G \cap H$ get the same color as in H.

When edge $\overrightarrow{(a,b)}$ in H has red colour then we color edges $\overrightarrow{(a,v)}, \overrightarrow{(v,b)}$ to red and analogously for other color an edges $\overrightarrow{(b,a)}$ $\overrightarrow{(c,d)}$ $\overrightarrow{(d,c)}$.

In such coloring, there could be a color with more than one outgoing edge from v. In that case, we choose the edge from which path continues directly to the destination and remove other from colored G.

Then we add leaf edges from u and v to all colored subgraphs that do not already contain it. Fortunately, there is enough not removed edges to add it.

In such way, we get correct independent spanning trees. □

4.2.3 Proof

Theorem 19. *Each 4-connected multigraph has four independent spanning trees.*

Proof. We prove the statement by induction on the number of edges.

First of all, if the graph has a nontrivial 4-cut, then we use the nontrivial 4-cut reduction. Now we suppose that there is not nontrivial 4-cut.

If there are two or more parallel edges (u, v) in G then they are either useless or one of u, v has degree four and we can use multi-edge reduction, see Section 4.2.1.

If G is K_5 it has four independent spanning trees see Figure 4.7.

Graph G has not useless edge and it is not K_5 so it has the vertex with degree four and we can use vertex reduction. □

4.3 Grids are 3-resilient

This section is a citation from an unpublished text of Ilya Nikolaevskiy [9].

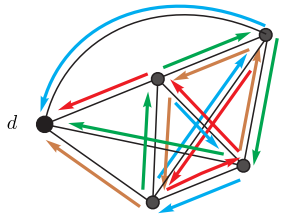


Figure 4.7: Four independent spanning trees for K_5.

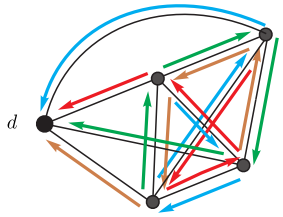


Figure 4.8: Grid decomposition patterns to 2 Hamiltonian cycles for Odd-Odd, Even-Even and Even-Odd grid graphs

In this section we present a 3-resilient routing for a grid network. We consider a rectangular grid with additional links wrapped around borders (see Fig. 4.8). This is a 4-connected graph, thus according to conjecture we expect 3-resilience.

Our routing scheme relies on graph decomposition to 2 edge-independent Hamiltonian cycles. Such decomposition always exists. We provide patterns for different parity of grid dimensions which are extendible by adding two rows or columns for such decomposition. Figure 4.8 shows all possible parity cases. Two cycles are marked with different line types. Repeatable blocks are highlighted with curve brackets.

The routing will be as following. Each cycle is directed in either way. Then packet starts to traverse the first cycle along the direction. If packet encounters a failed link it should start to traverse the same cycle backwards. If the packet encounters a second failed link it should start to traverse the second cycle along the direction. If packet encounters a third failed link it should start to traverse the same cycle backwards. If at any time a packet arrives to the destination it stops to route. Note that each node has exactly 4 links: inbound and outbound links in both cycles. Therefore it is always possible to switch to the second cycle. Because cycles are edge-independent, a packet will never encounter the same failed link twice. Therefore for any 3 failed links any packet will be guided to the destination using proposed routing scheme.

4.4 Planar triangulations are 3-resilient

In this section, we prove that planar multigraphs with some properties are 3-resilient by finding routing tables. We show that every triangulation has that property.

We define a *level* for each vertex in a plane graph. According to this we distinguish four types of vertices. Three of them can appear in a plane triangulation. We prove that the last type cannot appear there. We describe the routing tables for these three types. Finally, we prove that the prescribed walk ends at the destination.

For this chapter, we fix a planar embedding of the graph G.

4.4.1 Vertex level

Definition. *We use an inductive definition of level. The destination has* level *0. Let G_{l-1} be a subgraph induced by vertices with level less or equal to l-1. Every vertex which shares a face with level l-1 vertex gets level l. Subgraph on vertices of level l is called S_l.*

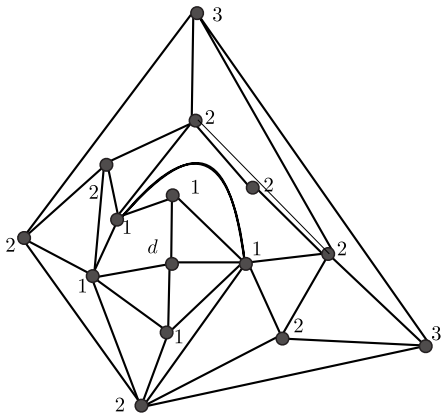


Figure 4.9: Levels in plane graph.

You can see an example of levels in Figure 4.9.

Notice that our levels differ from depths obtained by breadth-first search algorithm. In fact, level is always lower or equal to depth.

Notice that levels of every two adjacent vertices differ by at most one.

Observation 20. *Every vertex of S_l is incident to the same face of S_l.*

Proof. We can obtain S_l by deleting G_{l-1} from G_l. The graph G_{l-1} is connected so it lies in one face of S_l. We call that face *old*.

Every vertex v of S_l shares a face F_v with a level l-1 vertex in G. The face F_v is a part of the old face in S_l. Therefore the old face is incident with every vertex of S_l. □

The *outgoing trail* of S_l is an undirected trail around the old face of each component of S_l. Every vertex of G except the destination has at least two edges in the outgoing trail.

Subgraph S_l is typically a cycle. Then the outgoing trail is an directed cycle. When a component of S_l is a tree, the outgoing trail uses every edge twice. Articulation in S_l has more edges in the outgoing trail. You can see an example of an outgoing trail in Figure 4.10.

Observation 21. *For every outgoing trail, there are at least four edges between the vertices on it and vertices having lower levels.*

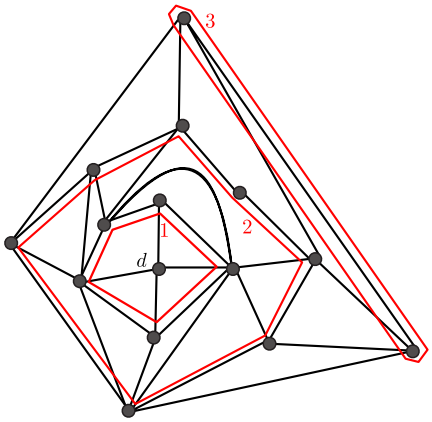


Figure 4.10: Outgoing trails in plane graph.

Proof. Connected component of S_l divides the plane into faces. Vertices having lower level are in one face an vertices having higher level in other faces. Edges between connected component of S_l and G_{l-1} are a cut in the graph. The graph is 4-connected so there are at least four such edges. □

4.4.2 Types of vertices

We distinguish four types of vertices based on how many lower-level neighbours they have. See 4.11.

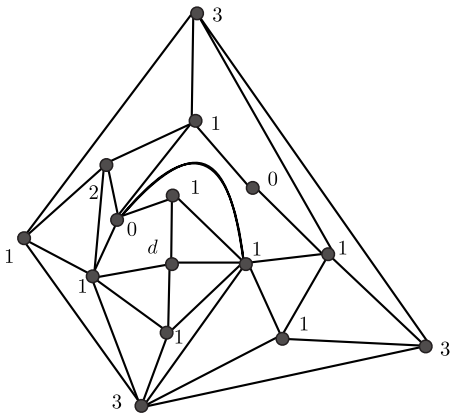


Figure 4.11: Types of vertices in a plane graph.

Type 0 Level l vertex v does not have adjacent level l-1 vertices.

Type 1 Level l vertex v has one adjacent level l-1 vertex.

Type 2 Level l vertex v has two adjacent level l-1 vertices.

Type 3 Level l vertex v has three or more adjacent level l-1 vertices.

4.4.3 Graphs without Type 0 vertices

Lemma 22. *Planar triangulations do not have a vertex of Type 0.*

Proof. Every level l vertex v shares a triangular face with a level l-1 vertex. So v has at least one adjacent vertex of level l-1. □

Lemma 23. *Let G be a graph without a vertex of Type 0. Then the depth obtained by breadth-first search algorithm is equal to the level.*

Proof. We prove it by induction. The destination has both level and depth equal to 0.

In each phase l: Every vertex v from S_l shares a triangular face with a level $l-1$ vertex. The vertex v has at least one adjacent level $l-1$ vertex. So vertex v has both level and depth equal to l. □

4.4.4 Routing tables

In this section we present an algorithm for creating routing tables. When something is sent through the edge u to the vertex v, we recommend edges to continue from v. The recommendation depends on the Type of the vertex v and on the outgoing trail.

We need to denote some adjacent vertices for each type of the vertex v for describing the routing tables and moreover for the proof of 3-resilience.

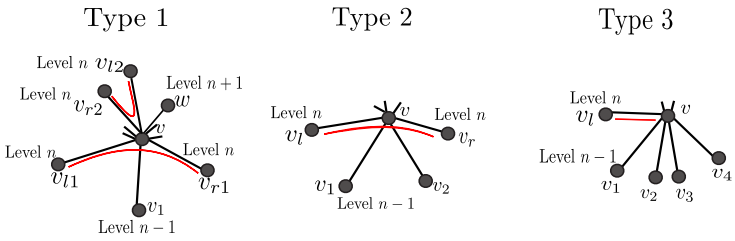


Figure 4.12: Notation for types of vertices.

The vertex v has level l. The outgoing trail uses vertices v_l, v, v_r in clockwise order. When v is an articulation in S_l, the outgoing trail uses vertices $v_{l1}, v, v_{r1}, \ldots, v_{l2}, v, v_{r2}, \ldots$

Type 1 The vertex v has Type 1. It has one adjacent level l-1 vertex v_1. When the prescribed tour come from the level l+1 vertex we denote it w. Otherwise the w is an arbitrary level l+1 vertex.

Type 2 The vertex v has the Type 2. It has two adjacent level l-1 vertices v_1, v_2.

Type 3 The vertex v has the Type 3. It has at least three adjacent level l-1 vertices v_1, v_2, v_3.

Rules for creating the routing tables

We create the routing tables for each vertex using following rules. They are applied in the given order. Every Rule recommends several edges. For 3-resilience routing tables is necessary and sufficient to recommend at least four edges.

Suppose that something was sent from the vertex u to the vertex v.

Rule 1 First we recommend all adjacent level l-1 vertices other then u in an arbitrary order.

Rule 2 When the edge (u, v) is on the outgoing trail, we recommend to use the next edge e on the outgoing trail. When the edge e was removed we recommend the edge (v, u) to send it back.

Rule 3 When something was sent to v by an edge which is not on the outgoing trail, we recommend the edge e on the outgoing trail in the clockwise direction. When the edge e was removed we recommend using the edge on the outgoing trail in the counterclockwise direction.

Rule 4 When the vertex u has the level l+1 we recommend the edge (v, u) otherwise we recommend an arbitrary level l+1 vertex.

4.4.5 Routing tables are correct

Theorem 24. *The prescribed tour using routing tables obtained by Rules 1-4 ends at the destination.*

We prove it by the contradiction. Suppose that the prescribed tour does not end in the destination. It could not end in any other vertex so there is a cycle in it.

In the cycle is vertex which has the minimal level. This is in a contradiction with the following lemma.

Lemma 25 (Type i)**.** *If the prescribed walk enters the vertex v of the Type i and the level l then after finitely many moves it enters the level l-1 vertex.*

We start with some observation which holds for each type of vertices. Then we prove the lemma for Type 3, Type 2 and finally for Type 1. The proofs of observation 27 and for Type 2 use lemma for Type 1.

Observation 26. *Without loss of generality, we can suppose that the prescribed tour starts by the edge which goes from the level l+1 vertex to the level l.*

Proof. We can shorter the prescribed path if there is such edge.

Suppose that there is a cycle C in the prescribed path and all vertices in it have the same level. By the observation 21 there are at least four edges to vertices which has the lower level. At least one of them was not removed and the Rule 1 recommend to use it. Contradiction. □

Observation 27. *If the prescribed tour go from the level l to vertex u the level l+1 vertex v then vertex u has the Type 1.*

Thanks to the Observation, we can use the Lemma Type 1 for the vertex u and we do not have to consider that case for vertex v, tat can have any type.

Proof. Routing tables send something up when Rule 4 is applied. Rule 2 and 3 was not applied so two edges on the outgoing trail was removed. Rule 1 was not applied. The vertex v had only one edge to the lower level and so it has Type 1. □

Proof Type 3. The prescribed tour goes down using v_1, v_2, v_3 unless three edges were removed. In that case it goes on the outgoing trail to a level l vertex which edge to level l-1 vertex cannot have been removed. □

Proof Type 2. The prescribed tour either goes down using v_1 or v_2 unless at least two edges was removed. In that case it goes on the outgoing trail. If an edge on th outgoing trail was removed it continues in reverse direction to the level l vertex v_{r1}. The vertex v_{r1} is either of Type 2 or 3 and there is at least one adjacent not removed edge to the level l-1 vertex or the vertex v_{r1} is of the Type 1 and we apply the Lemma Type 1 on the vertex v_{r1}. □

Proof Type 1. The prescribed tour goes down using v_1 unless at least one edge was removed. Then it goes on the outgoing trail until it reaches a vertex without a removed edge to a level l-1 vertex unless an edge of the outgoing trail was removed. Remark that there are at least four edges from the outgoing trail to lower level vertices according the observation 21.

When these two edges were removed the prescribed tour continues in the opposite direction of the outgoing trail until it reaches a vertex with not removed edge to level l-1 vertex unless the third edge was removed.

The part of the outgoing trail between the two removed edges has only one edge to the level l-1 vertex. When there is no vertex of Type 0 in the graph then the part of the outgoing trail is just one vertex v of Type 1. Then the prescribed tour continues up to the level $l+1$ vertex w according to Rule 4.

The vertex w has either Type 1 or Type 2 or 3. If the vertex w has Type 1, then it continues on outgoing trail to the level $l+1$ vertex w' and continue down to the level l vertex v'. If the vertex w has Type 2 or 3 there is not removed edge to the level l vertex v'.

The outgoing trail is back on the level l. If $v' = v$ then we return to w' and continue on level $l+1$ outgoing trail. After few repeats, we had to reach $v' \neq v$. If not, then the three removed edges are 3-cut in G. When $v' \neq v$, there is not removed edge from v' to level l-1 vertex.

□

The routing tables for the triangular plane graphs are designed in a way that they route to a lower level vertex if possible and stay on the outgoing trail. Such strategy works for all graphs which do not have any vertex of the Type 0.

Conclusion

We studied the principle of k-resilience and proved that 3-edge-connected graphs are 2-resilient and planar triangulated 4-edge-connected graphs are 3-resilient.

We propose further work in three directions.

Four independent spanning trees vs. 3-resilience We prove that every 4-edge-connected graphs have four independent spanning trees. However, this could not be directly used to prove 3-resilience.

In the 2- and 3-connected case, we set a cyclic order on trees; when it is not possible to continue in one tree, we continue in the next one. In the 4-connected case, we cannot use the same strategy. We have established a counterexample on K_5, see Figure 4.13 When color has cyclic order red, green, blue, yellow and the edges $(d, 1), (1, 2), (3, 4)$ were removed, then the prescribed walk starting on the edge $(4, 1)$ creates a cycle on vertices $1, 3, 2, 4, 1, \ldots$ and never reaches the destination.

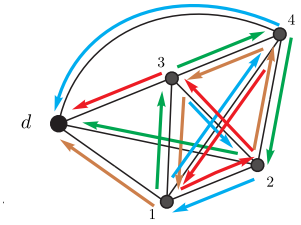


Figure 4.13: Counterexample – four independent spanning trees in K_5.

There is a number of other possible strategies and their combinations such as choosing a tree which has shorter path to destination, or choosing a tree which is most similar to greedy strategy. However, we do not know if they guarantee 3-resilience or not.

Existence of k independent spanning trees in k-edge-connected graphs We prove existence of three and four independent spanning trees for 3- and 4-edge-connected graphs, respectively. We believe that a similar idea could be used for proving that k-edge connected graphs have k independent spanning trees.

There is one reduction which we do not need for our result but could be useful for generalization: Vertex of degree l, where $k + 1 \leq l \leq 2 \cdot k$, could be replaced by the graph K_k.

When we reduce the graph as much as possible, it is sufficient to construct k independent spanning trees for graph whose vertices have degree k or $k + 1$ and with no non- trivial k-cut. Such graphs have only few possible structures near destination.

Planar 4-edge-connected graphs Despite the fact that we did not find the way to use four independent spanning trees for creating 3-resilient routing tables, we have found a way to construct the 3-resilient tables directly for a certain class of planar graphs.

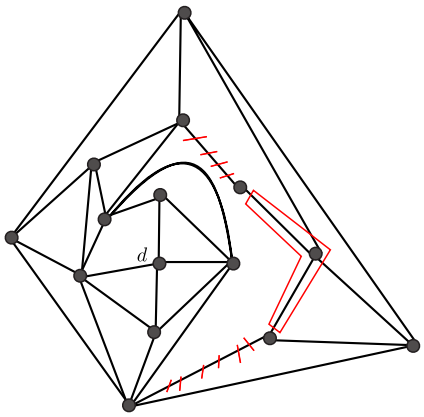


Figure 4.14: Graph with vertices of Type 0 where the prescribed path loops.

Let vertices of a graph have depth defined by the breadth-first search algorithm. We found rules for creating routing tables when every vertex has an edge to a vertex with a lower depth. When a vertex has no such edge, and two vertices like this are on the same outgoing trail, the routing tables created by our rules are no longer 3-resilient: in a situation shown in Figure **??**, the prescribed path will loop between vertices u and v.

We tried to change the rules to avoid this. Every change we tried either did not work, or the resulting rules were too complicated to let us elegantly prove 3-resilience. However, we still believe that simple rules for planar 3-resilient routing tables exist.

Bibliography

[1] Tutte, William Thomas. *On the problem of decomposing a graph into n connected factors.* Journal of the London Mathematical Society,1961.

[2] Frank, An drás. *On disjoint trees and arborescences.* Algebraic Methods in Graph Theory, Colloquia Mathematica Soc. J. Bolyai. Vol. 25. 1978.

[3] Fujishige, Satoru. *A note on disjoint arborescences.* Combinatorica 30.2 (2010): 247-252.

[4] Király, Csaba. *Algorithms for finding a rooted (k, 1)-edge-connected orientation.* Discrete Applied Mathematics 166 (2014): 263-268.

[5] Menger, Karl. *Zur allgemeinen Kurventheorie.* Fund. Math. 10 (1927): 96–115.

[6] Bollobás, Béla. *Modern graph theory.* Springer Science Vol. 184. & Business Media, 1998.

[7] Gopalan, Abishek and Ramasubramanian, Srinivasan. *On the maximum number of linearly independent cycles and paths in a network.* Networking, IEEE/ACM Transactions on Networking, Vol. 22. (2014): 1373-1388.

[8] Khuller, Samir and Schieber, Baruch. *On independent spanning trees.* Information Processing Letters, Elsevier, Vol. 42. (1992): 321–323.

[9] Chiesa, Marco and Nikolaevskiy, Ilya. Private comunication.