Thesis

Sander Beekhuis

October 26, 2016

5 Vertical one-sided dual

We can also adapt Fusy's algorithm to generate a vertically one-sided dual. We then need top generate a regular edge labeling without red faces that have 3 or more edges on both borders.

We will have an additional requirement on top of the requirement that $\bar{G}$ has no separating triangles. We will also require that G has no separating four cycles.

Notational concerns Just as in Section 4 we will use $\mathcal{C}$ to indicate the current sweep line cycle. We will repeatedly only consider the path $\mathcal{C} \setminus \{\mathrm{S}\}$. In that case we will always order it from W to E.

Instead of interior paths we will consider interior walks but we will use similar notation. That is a walk between two distinct vertices of $\mathcal{C}$ of which all vertices except the first and last one are in the interior of $\mathcal{C}$.

We will let $\mathcal{W}$ denote a interior walk. Given such a walk of k vertices we index it's nodes $w_1, \ldots, w_k$ in such a way that w_1 is closer to W then w_k is (and thus that w_k is closer to E then w_1 is).

Then w_1 and w_k indicate the two unique vertices of the walk that are also part of the cycle. We will then let $\mathcal{C}|_{\mathcal{W}}$ denote the part of $\mathcal{C} \setminus \mathrm{S}$ that is between w_1 and w_k (including). $\mathcal{C}_{\mathcal{W}}$ will denote the closed walk formed when we paste $\mathcal{C}|_{W}$ and $\mathcal{W}$.

Since paths are a subclass of walks all of the above notation can also be used for a path $\mathcal{P}$. Note that the closed walk $\mathcal{C}_{\mathcal{P}}$ in this case will actually be a cycle.

5.1 Outline

To describe the algorithm two more definitions are necessary

Definition (Prefence)**.** A prefence $\mathcal{W}$ is a interior walk of $\mathcal{C}$ starting at $v_i \in \mathcal{C}$ and ending at $v_j \in \mathcal{C}$ a both adjacent to S

(P1) For every $v_i \in \mathcal{C} \setminus \{\mathrm{W}, \mathrm{S}, \mathrm{E}\}$ we have that all vertices between v_{i+1} and v_{i-1} in the rotation at v_i are in $\mathcal{W} \setminus \{\mathrm{W}, \mathrm{E}\}$

(P2) For every $w_i \in \mathcal{W} \setminus \{\mathrm{W}, \mathrm{E}\}$ we have that all vertices between w_{i-1} and w_{i+1} in rotation at w_i are in $\mathcal{C} \setminus \{\mathrm{W}, \mathrm{S}, \mathrm{E}\}$

(P3) w_2 and v_{i+1} are consecutive in the rotation at v_i

(P4) v_{j-1} and w_{k-1} are consecutive in the rotation at v_j

We enforce these conditions because they imply (E3) when $\mathcal{W}$ is a path as we will show in Lemma 21.

For a walk however the interior is not clearly defined.

Definition (Fence)**.** A fence is a valid path starting and ending at a vertex adjacent to S

We will show that there is a algorithm if there are no separating 4-cycles in G and no separating 3-cycles in $\bar{G}$.

FiXme: expand on naming/reasons of fence

The algorithm will receive as input a extended graph $\bar{G}$ and will return a regular edge labeling such that all red faces are $(1-\infty)$ using a sweep-cycle approach inspired by Fusy[1].

We will start by creating a prefence W. This may not be a valid path, it doesn't even have to be a path. During the algorithm we will make a number of moves that will turn this prefence into a fence. In each move we shrink C by employing a valid paths and change the prefence.

5.2 Finding a initial prefence

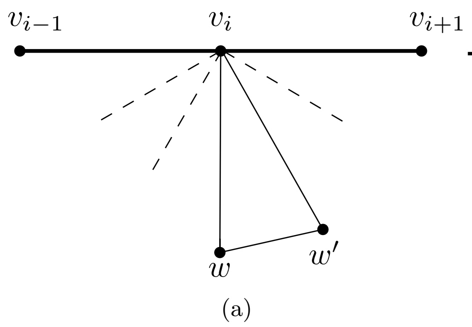
Let v_i denote all the vertices of $\mathcal{C}\backslash\{\mathrm{S}\}$ in the following order $\mathrm{W} = v_1\, v_2\, \ldots v_{n-1}\, v_n = \mathrm{E}$. Some intervals of these vertices will be adjacent to S. However, they can't be all adjacent to S since then the sweepcycle will be non-separating since we can't have separating triangles. We denote by v_i the last vertex of fist interval of vertices adjacent to S and by v_j the first vertex of the second interval. As candidate walk we will start with v_i, we will then take the vertices adjacent to v_{i+1} between v_i and v_{i+2} in the rotation at v_{i+1}, followed the vertices between v_{i+1} and v_{i+3} in the rotation at v_{i+2} and so further until we add the vertices between v_{j-2} and v_j in the rotation around v_{j-1} and finally we finish by adding v_j.

We then remove all subsequent duplicate vertices from $\mathcal{W}$.

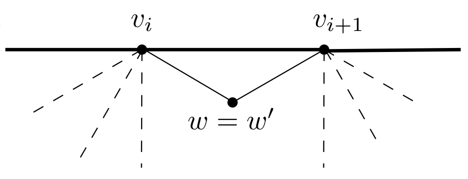
Lemma 20. *The collection W described above is a prefence.*

Proof. We will first show that W is a walk. We will proof that every vertex is adjacent to the next vertex. Let us suppose that w and w' are two subsequent vertices in W, we will show that ww' is an edge if $\{w, w'\} \cap \{v_i, v_j\} = \emptyset$. Afterwards we will consider this edge case. There are then two cases for w, w'. Either (a) w and w' are vertices adjacent to some v_i subsequent in clockwise order or (b) w was the last vertex adjacent to some v_i and thus w' is the first vertex adjacent to v_{i+1}.

The following two situations can also be seen in Figure 5.



(a)



(b)

Figure 5: The two main cases of the proof showing that W is a walk

In case (a) we note that v_iw and v_iw' are edges next to each other in clockwise order around v_i. Since every interior face of $\bar{G}$ is a triangle ww' must be an edge. We thus see that w, w' are adjacent and not duplicates.

In case (b) we note that v_iw and v_iv_{i+1} are edges subsequent in clockwise order, hence wv_{i+1} is also an edge. Hence w is the first vertex adjacent to v_{i+1}

after v_i in clockwise order. Thus $w = w'$. They are duplicates and one of them must have been removed.

Now for the edge cases: Let x be the first vertex adjacent to v_{i+1} and let y be the last vertex adjacent to v_{j-1}. v_i and x are vertices adjacent to v_{i+1} subsequent in clockwise order, and hence connected by Lemma 3. In the same way y and v_j are subsequent vertices in the rotation at v_n and hence connected.

Hence $\mathcal{W}$ is a walk. The above also shows that $v_i v_{i+1} x$ and $v_{j-1} v_j y$ are triangles by Lemma 3 and hence $\mathcal{W}$ satisfies properties (P3) and (P4) of being a prefence.

Moreover this walk satisfies (P1) because $\mathcal{W}$ by construction contains all neighbors of any vertex $v_i \in \mathcal{C}|_{\mathcal{W}} \setminus \{v_i, v_j\}$ between v_{i-1} and v_{i+1} in the rotation of v_i.

Finally to see that $\mathcal{W}$ also satisfies (P2). Consider a vertex $w_j \in \mathcal{W} \setminus \{v_i, v_j\}$ then either it is (a) the neighbor of some vertex v_i and only of this vertex or it is (b) the unique vertex neighboring in the interior of the cycle the $\ell + 1$ vertices $v_i, \ldots, v_{i+\ell}$. This is essentially the same case distinction as above. However now (a) $w_{i-1} w_i v_i$ and $v_i w_i w_{i+1}$ or (b) $w_{i-1} w_i v_i$, $v_i w_i v_{i+1}, \ldots, v_{i+\ell-1} w_i v_{i+\ell}$ and $v_{i+\ell} w_i w_{i+1}$ form a set of triangles spanning the area between w_{i-1} and w_{i+1} in the rotation at w_i. Thus any edge not going to $\mathcal{C}|_{\mathcal{W}} \setminus \{v_i, v_j\}$ in this sector will lead to a separating triangle. We however have assumed G has no separating triangles. Hence (P3) holds.[4] $\square$

We then orient $\mathcal{W}$ from v_i (the vertex closest to W)to v_j (the vertex closest to E) and denote it's vertices by $w_1 \ldots w_k$.

5.3 Irregularities

We will distinguish two kinds of *irregularities* in a prefence.

1. The candidate walk is non-simple in a certain vertex. That is, if we traverse the sequence of vertices in $\mathcal{W}$ we see that $w_i = w_j$ for some $i < j$.
2. The candidate walk has a chord on the right. That is, there is an edge $w_i w_j$ on the right of $\mathcal{W}$ with $i < j$ and i and j not subsequent (i.e. $i < j - 1$).

Note that we can't have a chord can on the left of $\mathcal{W}$ ($\mathcal{W}$ being oriented from W to E), since if it would lie on the left of $\mathcal{W}$ the vertices $w_{i+1}, \ldots, w_{j-1}$ would not have been chosen in the construction of the prefence.

Lemma 21. *If a prefence has no irregularities it is a fence.*

Proof. We will show that all the requirements of being a valid path are met.

Path Let us begin by noting that since there are no non-simple points we have a path and not just a walk.

(E1) It is clear that both w_1 and w_k are not S by the construction of the candidate walk.

[4] FiXme: I believe this is still true when separating triangles are allowed to occur. However the prove will have to be different.

(E2) For $\mathcal{W}$ or $\mathcal{C}|_{\mathcal{W}}$ to have only one edge we need to have that v_iv_j is an edge. However, v_iv_j can not be an edge in $\mathcal{C}$ since v_i and v_j are from different intervals of vertices adjacent to S. It can also not be an edge in $\bar{G} \setminus \mathcal{C}$ since that would be a chord of the cycle and these don't exist by Invariant 15 (I2)

(E3) Every interior edge of $\mathcal{C}_{\mathcal{W}}$ with at least one endpoint on the cycle is of the required type by the conditions (P1) - (P4). We note that these edges in particular have both endpoints on the cycle $\mathcal{C}_{\mathcal{W}}$.

Interior edges with both endpoints not on the cycle can a priori exist. However since a triangulation is a connected graph there must then also be an edge with one endpoint on $\mathcal{C}_{\mathcal{W}}$, and one inside $\mathcal{C}_{\mathcal{W}}$ but this can not be if $\mathcal{W}$ is a prefence. However by the argument above both endpoints must then be on $\mathcal{C}_{\mathcal{W}}$, this is a contradiction.

(E4) The cycle $\mathcal{C}'$ only changes between v_i and v_j. There can be no chord with one vertex from cycle $\mathcal{C} \setminus \mathcal{C}|_{\mathcal{W}}$ and one from $\mathcal{W}$ since such a chord would cross Sv_i or Sv_j. There is no chord with two vertices in $\mathcal{W}$ since that would be a irregularity and there is no chord with two vertices from $\mathcal{C} \setminus \mathcal{C}|_{\mathcal{W}}$ by Invariant 15 (I2).

Hence, if $\mathcal{W}$ has no irregularities it is a valid path.

Furthermore, $\mathcal{W}$ is a path starting and ending at a vertex adjacent to S because it is prefence. And thus it is a fence. □

Definition (Range of a irregularity)**.** For a non-simple point $w_i = w_j$ with $i < j$ has *range* $\{i, \ldots, j\} \subset \mathbb{N}$. A chord w_iw_j with $i < j - 1$ has *range* $\{i, \ldots, j\} \subset \mathbb{N}$.

FiXme: Is it better to call this a non-simple point or a non-simple vertex?

Note that a chord can't have the same range as a non-simple point since then w_iw_j will be a loop and we are considering simple graphs. Furthermore two chords have different ranges because we otherwise have a multiedge. Two nonsimple points with the same range are, in fact, the same. This leads us to the following remark.

Remark 22. *Distinct irregularities have distinct ranges.*

Definition (Maximal irregularity)**.** A irregularity is maximal if it's range is not contained[5] in the range of any other irregularity.

Lemma 23. *Maximal irregularities have ranges whose overlap is at most one integer.*

Proof. We let I and J denote two distinct maximal irregularities with ranges $\{i_1, \ldots i_2\}$ and $\{j_1, \ldots, j2\}$. Let us for the moment suppose that I and J have ranges that overlap more then one integer. Since I and J are both maximal their ranges can not be contained in each other.

Without loss of generality we thus have $i_1 < j_1 < i_2 < j_2$.

Now two chords to the right of $\mathcal{W}$ would cross each other but we have a planar graph so this can't be the case.

[5]Because of Remark 22 being contained is the same as being strictly contained

Now let us without loss of generality suppose that I is a non-simple point. A non-simple point $w_{i_1} = w_{i_2}$ is adjacent to two ranges of vertices in $\mathcal{C} \setminus \{S\}$. $v_a \dots v_b$ and $v_c \dots v_d$ then $\tilde{C} = w_{i_1} v_b \dots v_c$ is a cycle. And because of the rotation at $w_{i_1} = w_{i_2}$ we have that $w_{i_{1+1}}, \dots, w_{i_{2-1}}$ are inside this cycle while $w_1 \dots w_{i_1 - 1}$ and $w_{i_2+1} \dots w_k$ are outside the cycle. See Figure.

FiXme: We could add figure to clarify.

Now if J is a chord we have $\tilde{C}$, which can't be. If J is also a nonsimple point this would imply that the vertex $w_{j_i} = w_{j_2}$ is at the same time inside and outside $\tilde{C}$ which is clearly impossible. □

5.4 Moves

The algorithm will remove these irregularities by recursing on a subgraph for each maximal irregularity. We shrink the cycle $\mathcal{C}$ with every valid path that is found in the recurrence, in the order they are found. Afterwards we update the prefence by removing $w_{i+1}, \dots, w_{j-1}$. In subsection 5.4.3 we will show that the updated prefence is a prefence for the updated cycle $\mathcal{C}$.

We will first show how to remove these maximal irregularities in Subsections 5.4.1 and 5.4.2. That is, we show which subgraph H we recurse upon for both kinds of irregularity. Furthermore we show that these subgraphs suffice the requirements of the algorithm.

Afterwards, in subsection 5.4.3 we will make sure that the subgraphs we recurse upon are edge-disjoint. That is, they only overlap in border vertices.

It is worth noting that other irregularities contained in such a maximal irregularity are solved in the recurrence.

5.4.1 Chords

If we encounter a chord we will extract a subgraph and recurse on this subgraph. A chord $w_i w_j$ has a triangular face on the left and on the right (like every edge). The third vertex in the face to the left will be called x. x is not necessarily distinct from w_{i+1} and/or w_{j-1} but this is also not necessary for the rest of the argument.

The vertex v_a on the cycle is uniquely determined as the vertices adjacent to both w_i and $wi + 1$. In the same way v_b is the unique neighbor of w_{j-1} and w_j.

FiXme: We might also work these out in a Figure.

We will describe a walk $\mathcal{U}$ running from v_a to v_b. This path consists of all vertices adjacent to w_i in clockwise order from v_a (inclusive) to x(inclusive) and subsequently all vertices adjacent to w_j in clockwise order from x (exclusive) to v_b (inclusive). This path is given in bold in Figure 6.

Lemma 24. *$\mathcal{U}$ is a chordfree path*

Proof. We note that $\mathcal{U}$ is a walk by the same reasoning as is given in Lemma 20.

FiXme: Is it nice to refer to a line of reasoning like this?

$\mathcal{U}$ cant have a non-simple point x' since it would have to be connected to at least two vertices. However a vertex x' that is distinct from x and is connected to both w_i and w_j will induce a separating triangle $w_i x' w_j$. $\mathcal{U}$ also can't be nonsimple at x since x is the the third vertex of the triangular face $w_i w_j x$. Hence $\mathcal{U}$ is a path.

$\mathcal{U}$ can't have chords $u_i u_j$ since they would either induce a separating 3- or 4-cycle either $w_i u_i u_j$ or $w_j u_i u_j$ or $w_i u_i u_j w_j$ depending on the vertex adjacent to u_i and u_j. □

FiXme: We use that we have no 4-cycles here

We then consider the interior of the cycle $\mathcal{C}_\mathcal{U}$ and the cycle $\mathcal{C}_\mathcal{U}$ itself as the subgraph H. We then take the tight extension at v_a and v_b. We will then recurse on this graph $\bar{H}_t$. See also Figure 6. Since $\mathcal{C}$ is chordfree by invariant 15 (I2) so is $\mathcal{C}|_\mathcal{U}$. We have also just shown that $\mathcal{U}$ is chordfree. So $\bar{H}_t$ is indeed defined. Furthermore, since H is a induced subgraph of G, $\bar{H}_t$ contains no separating 4-cycles not involving the poles.

We update the prefence by removing $w_{i+1}, \ldots, w_{j-1}$.

5.4.2 Nonsimple points

Removing a non-simple point is done is a similar manner.

The vertex v_a on $\mathcal{C}$ is uniquely determined as the vertices adjacent to both $w_i = w_j$ and $wi + 1$. In the same way v_b is the unique neighbor of w_{j-1} and $w_j = w_i$. Note that it may be that $w_{i+1} = wj - 1$ this does not matter for the rest of the argument.

FiXme: We may show this in a figure.

We will describe a walk $\mathcal{U}$ running from v_a to v_b. This path consists of all vertices in the rotation at $w_i = w_j$ from v_b (inclusive) to v_a(inclusive). This path is given in bold in Figure 7.

Lemma 25. *$\mathcal{U}$ is a chordfree path.*

Proof. If we orient $\mathcal{U}$ from v_a to v_b we see that $\mathcal{U}$ cant have a non-simple point since such a point would have edges to at least two vertices on the right. However every vertex can only be connected to $w_i = w_j$. Hence $\mathcal{U}$ is a path.

$\mathcal{U}$ can't have chords on the right of the path by the way we construct $\mathcal{U}$. Furthermore $\mathcal{U}$ can't have chords $u_i u_j$ on the left since they would either induce a separating 3-cycle $w_i u_i u_j$. □

We then consider the interior of the cycle $\mathcal{C}_\mathcal{U}$ and the cycle $\mathcal{C}_\mathcal{U}$ itself as the subgraph H.

We then take the tight extension of H at v_a and v_b to recurse on. See also Figure 7. Since $\mathcal{C}$ is chordfree by Invariant 15 (I2) so is $\mathcal{C}|_\mathcal{U}$. We have also just shown that $\mathcal{U}$ is chordfree. So $\bar{H}_t$ is indeed defined. Furthermore, since H is a induced subgraph of G, $\bar{H}_t$ contains no separating 4-cycles not involving the poles.

We update the prefence by removing $w_{i+1}, \ldots, w_{j-1}$ and we also recognize that $w_i = w_j$ is now a duplicate subsequent occurrence of the same vertex. So we also remove w_j.

5.4.3 Validity

Lemma 26. *After doing a move the updated prefence W is a prefence for the updated cycle C*

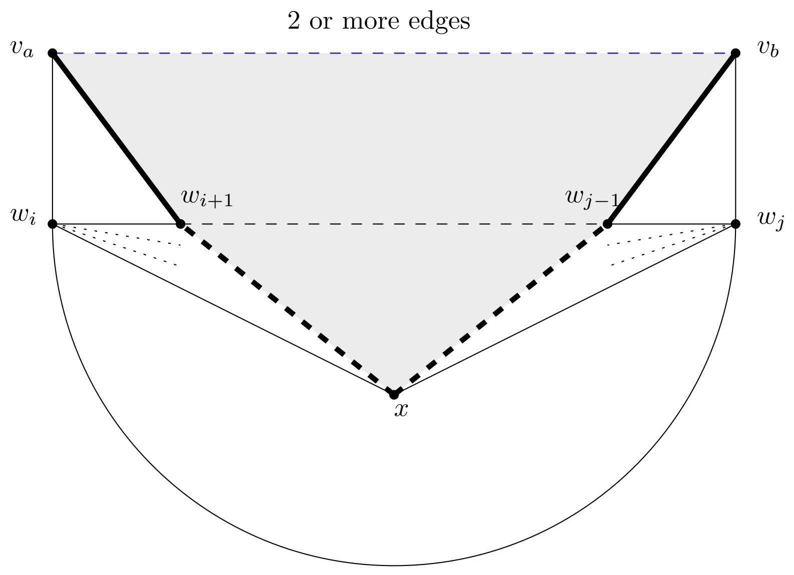
Proof. □

FiXme: TODO

Lemma 27. *Let H_I and H_J be two recursion subgraphs for different maximal irregularities I and J. Then H_I and H_J are edge disjoint.*

Proof. □

FiXme: TODO



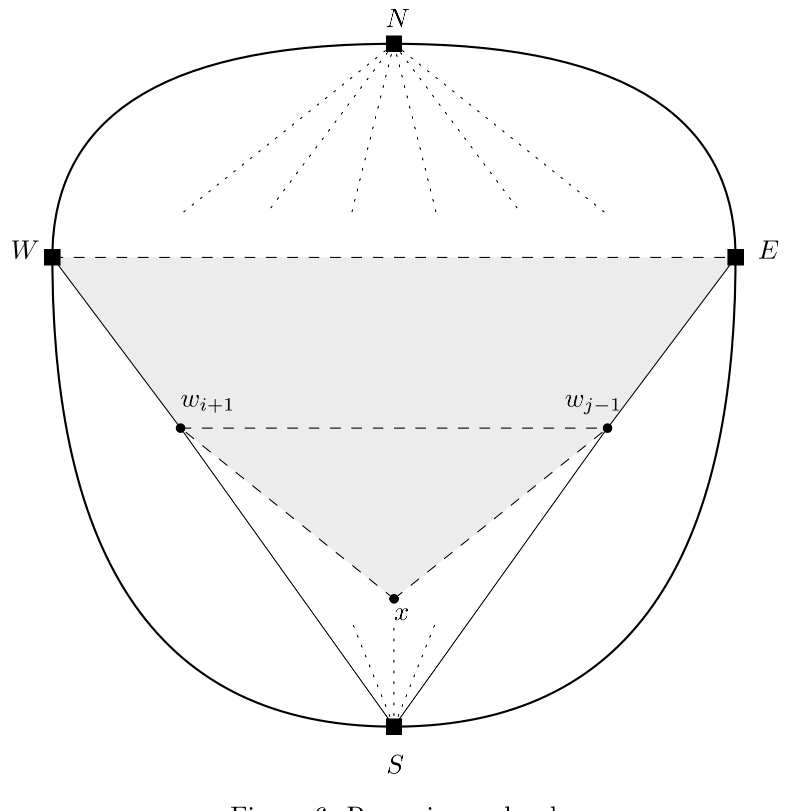
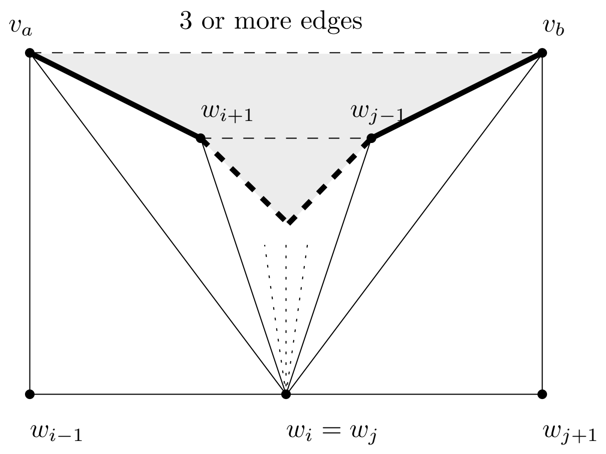


Figure 6: Removing a chord



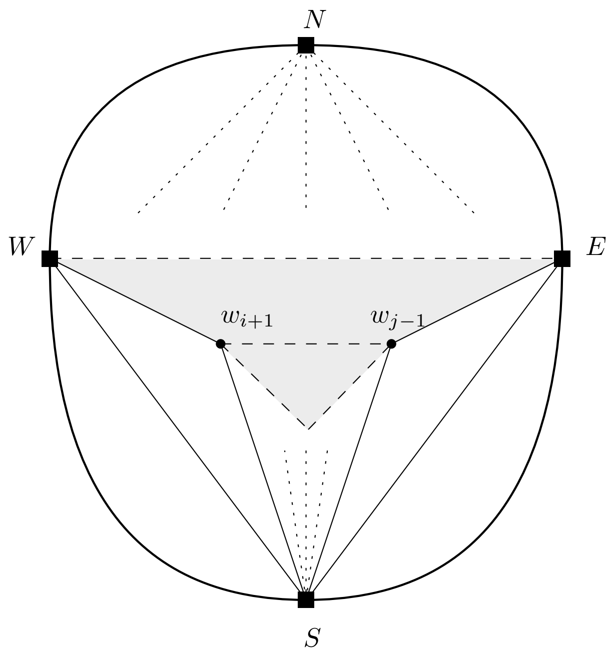


Figure 7: Removing a non-simple point

5.5 Correctness

As long as the interior of $\mathcal{C}$ is nonempty we can find a prefence. And thus we find valid paths. Since we continuously shrink the cycle with valid paths we end up with a regular edge labeling. See core algorithm

The algorithm finishes because it keeps on recursing and shrinking until no graph is left.

5.5.1 The red faces

Let us then argue that the red faces are all $(1 - \infty)$ faces, corresponding to one-sided vertical segments. As is shown in Lemma 10 it is sufficient to show no two vertices subsequent on a blue path are first a merge and then a split or vice versa.

We will show the following

Lemma 28. *A split or merge always happens on a vertex that is adjacent to S for some recursion.*

Proof. Every valid path we shrink the cycle by is found as a fence on some recursion level. In this recursion level both w_1 and w_k are adjacent to S. □

Lemma 29. *A path starting at a certain recursion level will stay at that recursion level. It may share vertices with the north boundary of a lower recursion level but never with the south boundary.*

Proof. A valid path can never leave the subgraph H in which its start- and end-vertex are located. Because it is found as a fence in this subgraph. It can also never run trough a graph H' on a lower recursion level (except for the north boundary path) because in every move the vertices of the prefence in H' are deleted. □

Recall that all our valid paths are oriented from a start vertex to end vertex.

Lemma 30. *A split can't directly be followed by a merge along any valid path during the algorithm.*

Proof. One of paths after the split is no longer on the south boundary of this subgraph H, nor on the south boundary of any other subgraph by Lemma 29. This path hence can't contain a merge.

The other path still potentially follow the south boundary. However merging from the southward side of the path is impossible by Lemma 29 from the northward side is equally impossible since the split and merge have to be neighboring vertices in the rotations of these vertices and thus the path $\mathcal{P}$ that merged must also join again.

But then it is not a valid path. □

However for a blue Z to occur there has to be a valid path that first has a split and then has a merge. Since this can't be all red faces must have only 2 edges on at least one side. Hence the regular edge labeling this algorithm produces corresponds to a vertically one-sided rectangular dual.

FiXme: Show how the algorithm works with some cool examples: For example: The multiple non-simple point $v_i = v_j = v_k$; Example of page $F1$; Example with lots of layered chords

notation We note that the interior of some cycle $\mathrm{Int}(C)$ are all vertices strictly in the interior of this cycle. We will sometimes also take $\mathrm{Int}(C)$ to refer to the induced subgraph of these vertices.

We let $\mathrm{Int}^+(C)$ denote the the vertices of C and C's interior vertices. We will also sometimes let it refer to the subgraph of G induced by these vertices.

coloring In this section we will use lots of figures to demonstrate how to handle each type of 4-cycle.

Usefull lemma's and notions All edges adjacent to the same exterior vertex of $\mathcal{D}$ in the interior of $\mathcal{D}$ have the same color and orientation.

Once we have chosen a direction and color for one such exterior vertex this choice follows for the rest of the exterior vertices.

Hence it is trivial to recurse on a 4-cyles once we haven chosen edge colors.

6 Seperating 4 cycles

Let $\mathcal{D}$ be a maximal separating 4-cycle.

Note that the only problem is given by 4-cycles that are entirely inside the cycle $\mathcal{C}$ maintained by the algorithm. If $\mathcal{C}$ is currently crossing $\mathcal{D}$ then the is not any longer a problem.

We can discern 7 types of adjacency for 4-cycles to a cycle if it's entirely inside some cycle.

(a) $\mathcal{D}$ has 1 edge on the cycle

(b) $\mathcal{D}$ has 2 consecutive edges on the cycle

(c) 2 non-consecutive edges on the cycle

(d) 3 edges on the cycle

(e) Just a vertex on the cycle

(f) Two consecutive vertices on the cycle

(g) Two non-consecutive vertices on the cycle

We will show by case distinction that everything will be okay. Sometimes we will make a move (updating cycle and prefence) to remove a irregularity and sometimes this will not be necessary.

We know that $\mathcal{D} \subseteq \mathrm{Int}^+(\mathcal{C})$. Either $\mathcal{D} \cap \mathcal{C} \neq \emptyset$ or $\mathcal{D} \cap \mathcal{C} = \emptyset$. In the first case we will say that $\mathcal{D}$ is on the cycle. In the second case we have that $\mathcal{D} \subseteq \mathrm{Int}^+(\mathcal{C}_{\mathbb{F}})$ since

6.1 On the cycle

Note that type (c), (d) and (f) can't occur on the cycle $\mathcal{C}$ maintained by the algorithm since they give a chord, offending Invariant 15 (I2).

Type (a) This is a *short chord*, that is a chord with a range of size only 3. Note that we allow a choice of edge flip to be made later.

Type (b) We can do a simple move evading the problem.
We make the move depicted in Figure 8

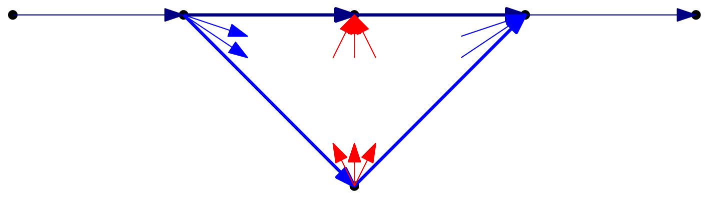


Figure 8: Removing a Type (b) separating 4-cycle

This moves the the cycle $\mathcal{C}$ past the problematic 4-cycle $\mathcal{D}$. Note that we don't allow any freedom in the interior edges in this case.

Type (e) This type of separating 4-cycle does not induce a irregularity in the fence. Hence no operation is necessary. As can be seen in Figure 9, we can just "corner-slice" it. This separating 4 cycle is no problem since it produces no irregularity on the (pre)fence $\mathcal{W}$.

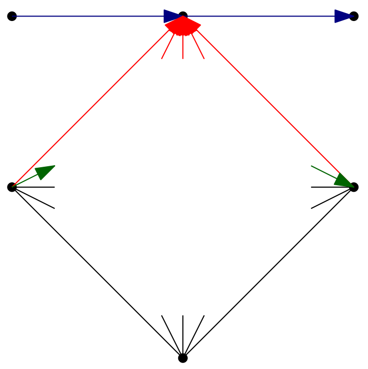


Figure 9: A prefence has no problem with a Type (e) separating 4-cycle

Note that the remainder of this maximal 4-cycle may contain another separating 4-cycle. Even when the two green edges are not a 4-cycle. Such a 4-cycle however is by design non-separating since the fence takes the topmost path.

Type (g) This is just a combination of a ordinary non-simple point and a Type (b) case. If we first recurse on the inner non-simple point we can then solve the rest like the Type (b) case.

6.2 On the fence

Note that Types (a), (b) and (e) do not provide irregularities when they are on the fence. Hence we don't have to do anything to deal with these separating 4-cycles. Instead we can treat them as being "on the cycle" in later iterations of this algorithm.

FiXme: Show that these types are okay with an image

Type (c)

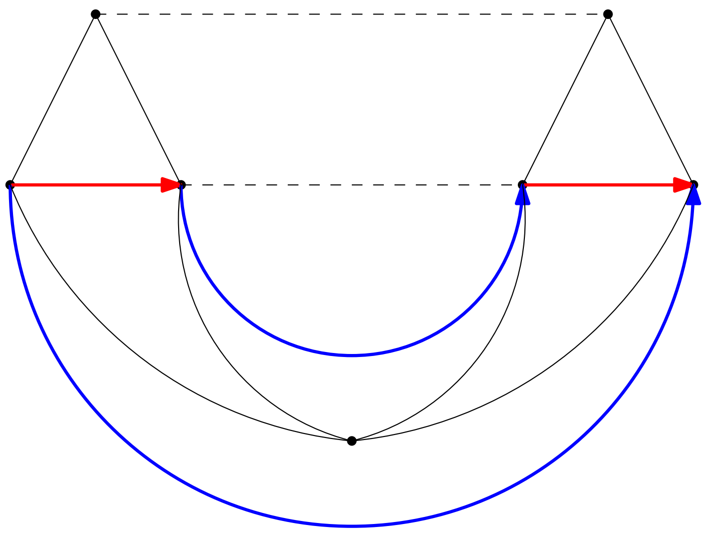


Figure 10: Removing Type (c) on the fence

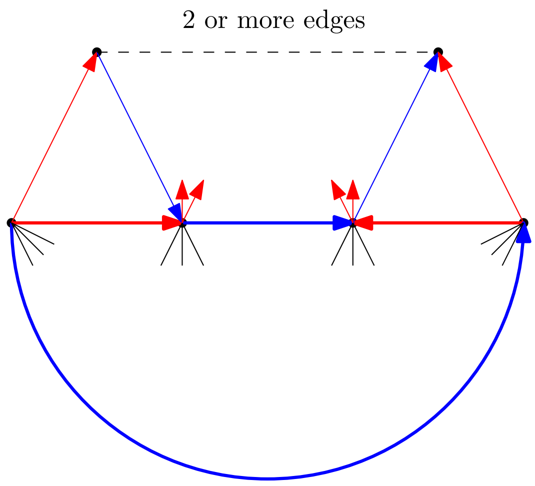


Figure 11: Removing Type (d) on the fence

Type (d)

Type (g)

Type (f) This is the inside of a chord

6.3 Not on the cycle or on the fence

Then $\mathcal{D}$ certainly causes no problems is no problem.

6.4 4-cycles with a complicated interior

We can just recurse

6.5 Adjecent 4-cycles

List of Corrections

We could cite Jordan's paper here. Should I? 2
We could provide a picture illustrating these concepts 2
Is this more a remark then a lemma? 3
We could add a figure to make this more clear 3
It is called irreducible because there is a reduction that works on separating triangles. We might show this reduction 3
We can sharpen this to 4 if we have a irreducible triangulation of the k-gon with a chordfree outer cycle 4
Probably refer to [2] 5
in what sense not unique, provide examples 5
Provide location, Kozminski & Kinnen and ungar, See Siam paper . . 5
This subsection still has to be written 6
This subsection still has to be written 6
This subsection still has to be written 6
TODO 6
right choice of words? 7
We might also want to provide a oriented version of this lemma 7
This lemma implies that any *alternating 4-cycle* is either *left-alternating* or *right-alternating* in the terminology of Fusy 7
Harmonize reference invariants and eligible/valid path requirements. and change E to V (for valids) 10
We need to add a partial inner vertex condition 10
We haven't proven this yet 11
As outlined in last meeting this proof is not complete as is, it has been moved to the appendix. We are stuck on the part where we need to find a path satisfying E4. We might proof this from red algo. . 11
Expand this subsection 12
Define what we mean with *cycle border* and *face border* 12
Revisit notation after writing section on oriented REL 12
expand on naming/reasons of fence 14
I believe this is still true when separating triangles are allowed to occur. However the prove will have to be different. 16
Is it better to call this a non-simple point or a non-simple vertex? . . 17
We could add figure to clarify. 18
We might also work these out in a Figure. 18
Is it nice to refer to a line of reasoning like this? 18
We use that we have no 4-cycles here 19
We may show this in a figure. 19
TODO 19
TODO 19
Show how the algorithm works with some cool examples: For example: The multiple non-simple point $v_i = v_j = v_k$; Example of page $F1$; Example with lots of layered chords 22
Show that these types are okay with an image 24

7 bib

I currently make latex crash if i try to give a bibliography