Generalization of braided streams

Sandro Savino
sandro.savino@dei.unipd.it
Department of Information Engineering, University of Padova, Italy

Introduction

Hydrography is one of the most important themes in a cartographic dataset and in literature many works can be found addressing the generalization of hydrographic networks.
Many of the approaches proposed use selection as the main generalization operator, but also employ data-enrichment techniques to improve the results of selection; in particular data enrichment is used to create a finer classification of the network edges (or a hierarchy) in order to enable the selection operator to choose the relevant parts of the network and delete the less important ones.
Most of these techniques work on the assumption that the topology of a hydrographic network is similar to a tree, mostly without loops, and it is possible then to detect branches that are more or less important.
However, this assumption is not always true: a river with high slope or large sediment load can form what is called a braided river, a type of channel consisting in many small channels branching and merging, separated by islands called braid bars.
In a braided river, the topology of the network is characterized by many loops and this topology makes less effective the generalization approach described above; furthermore, many factors concur to make it difficult to define the edge hierarchy driving the selection process: to due to the small height difference between consecutive edges it's difficult to understand the flow direction of the water and most edges can share the same name and classification.

For these reasons, in some works [Buttenfield, Savino, Touya] different approaches to derive braided streams are proposed; in particular, in this paper we will focus on the method described in [Savino], that has been implemented and tested on two different datasets.

The approach

The approach that we used looks at the network as a planar partition and is based on the duality between faces and edges: instead of working directly with the network edges, the algorithm works on the faces (also called loops), that in a braided stream correspond to the braid bars; the author proposes to generalize braided streams by area amalgamation instead of edge selection, merging smaller islands with their neighbors.

Area amalgamation is a well known generalization technique used by many authors [eg VanOosterom], but working on braided streams requires some extra caution in order to obtain a good and visually coherent generalization.
The first aspect to consider is to preserve a streamlined shape of the network: hard bends or narrow angles would look “odd” to the eye of the user, and should be avoided.

In our approach this is obtained identifying “blocking edges”: a blocking edge is an edge that, if removed, would create a hard bend in the network (see figure 1).

The identification of a blocking edge is performed measuring the angles between the edge and those connected to it (in other terms, if the edge is part of a stroke [see Thompson & Richardson, 1999], it should be kept). The amalgamation of two neighboring areas is allowed only if the edge they have in common is not a blocking edge.

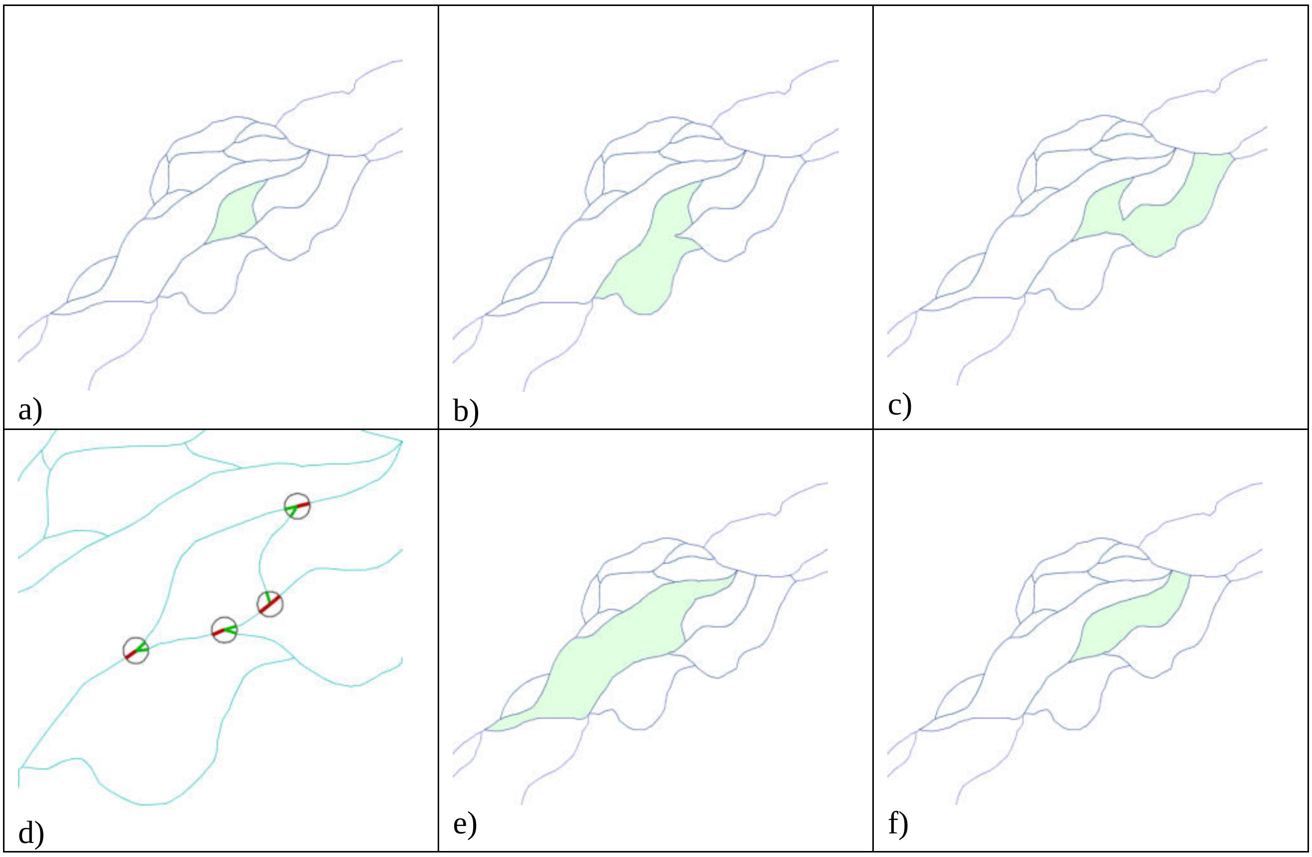


Figure 1: amalgamation, in blue the network edges, in green the faces of the planar partition; a) input data, the green face is the one to generalize, b) and c) examples of bad amalgamation; e) and f) examples of good amalgamation; d) edges and blocking edges for the nodes around the face: red segments indicate edges that, if removed, will produce narrow angles

A second aspect to consider, while amalgamating two islands, is the final shape of the resulting polygon.

Due to the inherent characteristics of braided streams, the network is an over-simplified representation of the actual shape of the river. The channels, represented by the edges, may have shapes –and especially breadth– varying considerably and while this is not represented in the network, it should be taken into account while generalizing.

In particular, using the network as a planar partition, it should be remembered that two islands sitting on adjacent faces in reality might be quite far apart depending on the breadth of the channel between them.

Since merging two islands requires to delete the part of the river polygon between them, amalgamating two islands far away should be avoided, as it could affect negatively the shape of the river by reducing locally its breadth.

In our approach this is taken into account performing the area amalgamation not on the faces of the planar partition, but on the geometry of the islands: these are computed in a pre-processing step subtracting from the faces the area of the river polygons (where not present river polygons are created drawing a user-defined buffer around the edges).

When amalgamating two islands our algorithm estimates the “water loss”, i.e. the size of the river polygon that would be covered by the geometry bridging two islands and tries to minimize it during the selection of the neighbor to use in the amalgamation.

Finally, in order to have a visually more pleasant result, the process tries to avoid creating elongated shapes, as a cluster of long and narrow shapes create more visual clutter than a cluster of broad shapes.
In our approach this is done evaluating the compactness (area/perimeter ratio) of the resulting shape while deciding the neighbor to use in the amalgamation process.

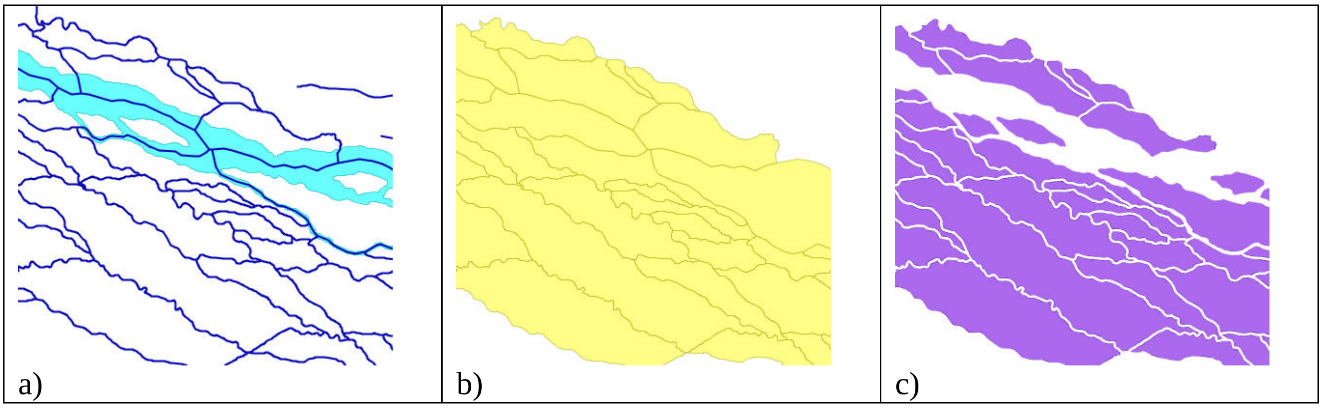


Figura 2: from left to right the input network and the river polygons (a), the faces computed using the network as a planar partition (b), the islands computed as a difference between the faces and the river polygons

.

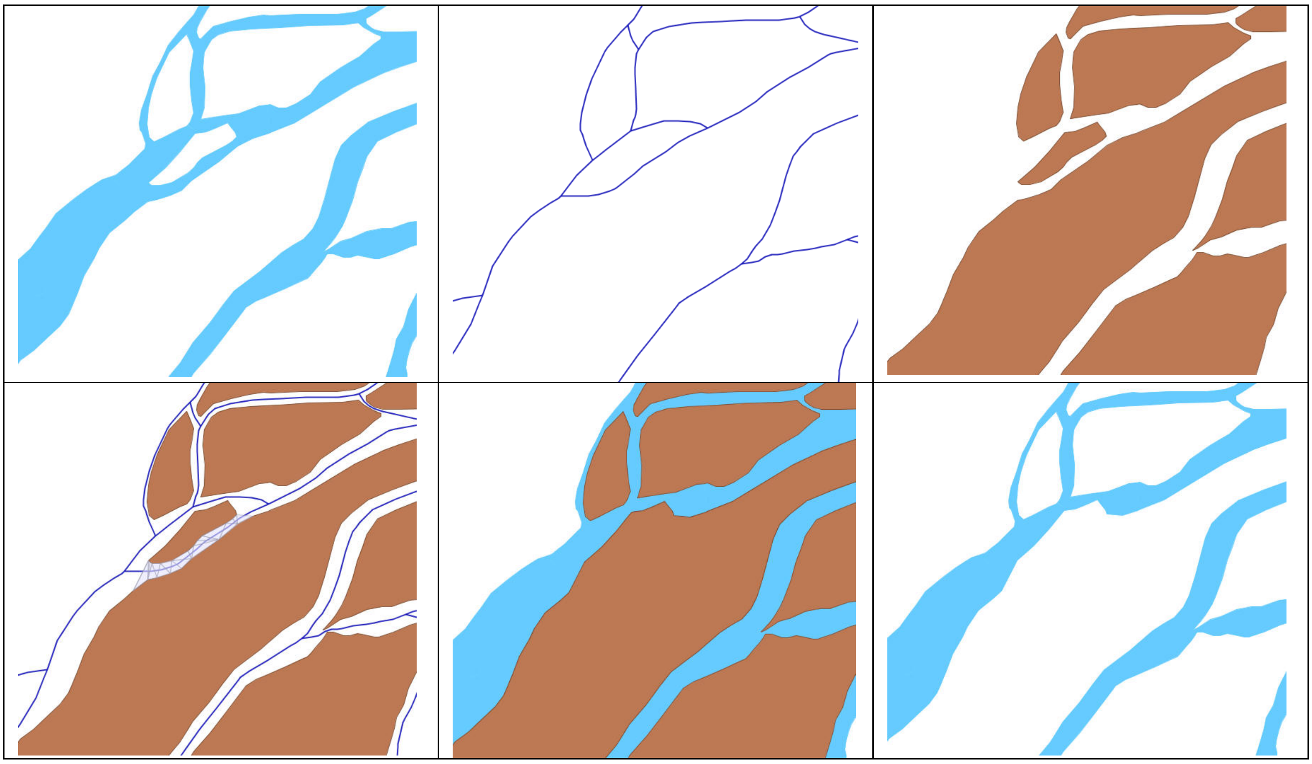
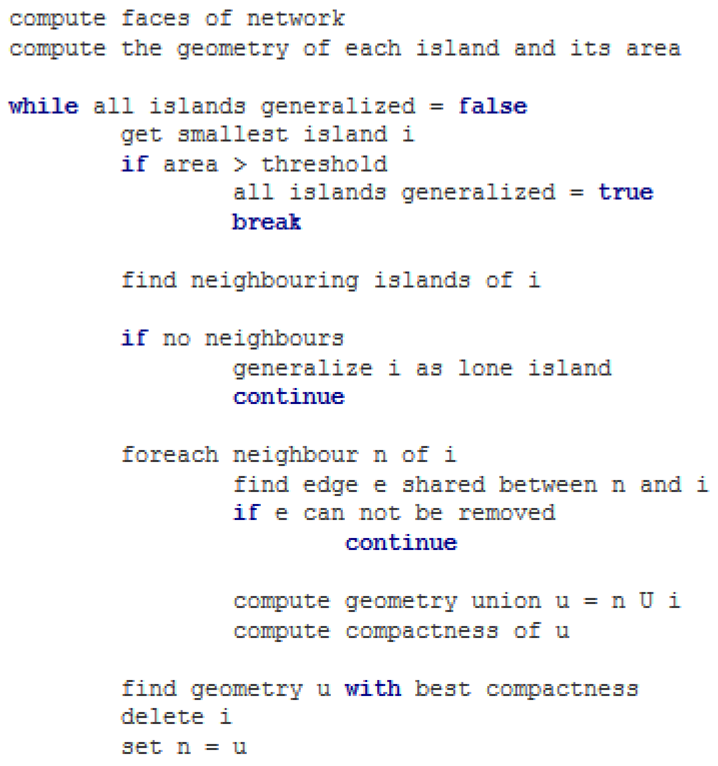


Figura 3: from left to right, top to bottom: the river polygon, the network, the islands, the triangle strip used to merge two islands, the results of the amalgamation

The process

The process pseudocode is illustrated below:



```
compute faces of network
compute the geometry of each island and its area

while all islands generalized = false
    get smallest island i
    if area > threshold
        all islands generalized = true
        break

    find neighbouring islands of i

    if no neighbours
        generalize i as lone island
        continue

    foreach neighbour n of i
        find edge e shared between n and i
        if e can not be removed
            continue

        compute geometry union u = n U i
        compute compactness of u

    find geometry u with best compactness
    delete i
    set n = u
```

The process loops iteratively on every island and stops when all the islands have an area bigger than a user-defined threshold.

Detecting blocking edges

Blocking edge are detected on each node of the network. An edge is called “blocking edge” if after its removal the other edges on the node form a narrow angle. An edge is a “blocking edge” if the sum of the angle between it and the first edge on his left and the first on his right is bigger than a threshold; this angle threshold was set experimentally to 240 degrees.

When merging two adjacent faces, the shared edge between them can be removed only if it is not a blocking edge.

Amalgamation of two islands

The amalgamation is performed on the geometries of the two islands using a triangulation based technique as described in [Jones et al. 1995]. The area of the triangle strip connecting the two islands corresponds to the water loss: its size, together with the compactness of the result geometry are evaluated to decide which neighbor to use in the amalgamation process. After the amalgamation, the edge underneath the triangles is removed from the network.

Generalizing lone islands

An island with no neighbors is called a lone island. It can be generalized in three different ways:

- it is kept, if it is bigger than a threshold

- it is deleted if smaller than a threshold
- it is enlarged if its size is between the two thresholds above

The deletion of a lone island can be performed either collapsing the loop on its midline or deleting some of the edges encircling the island.

The enlargement of an island is performed applying a particular scaling filter to its geometry. This filter was designed to obtain a more compact shape and displaces the vertices of the geometry proportionally to their distance from the centroid.

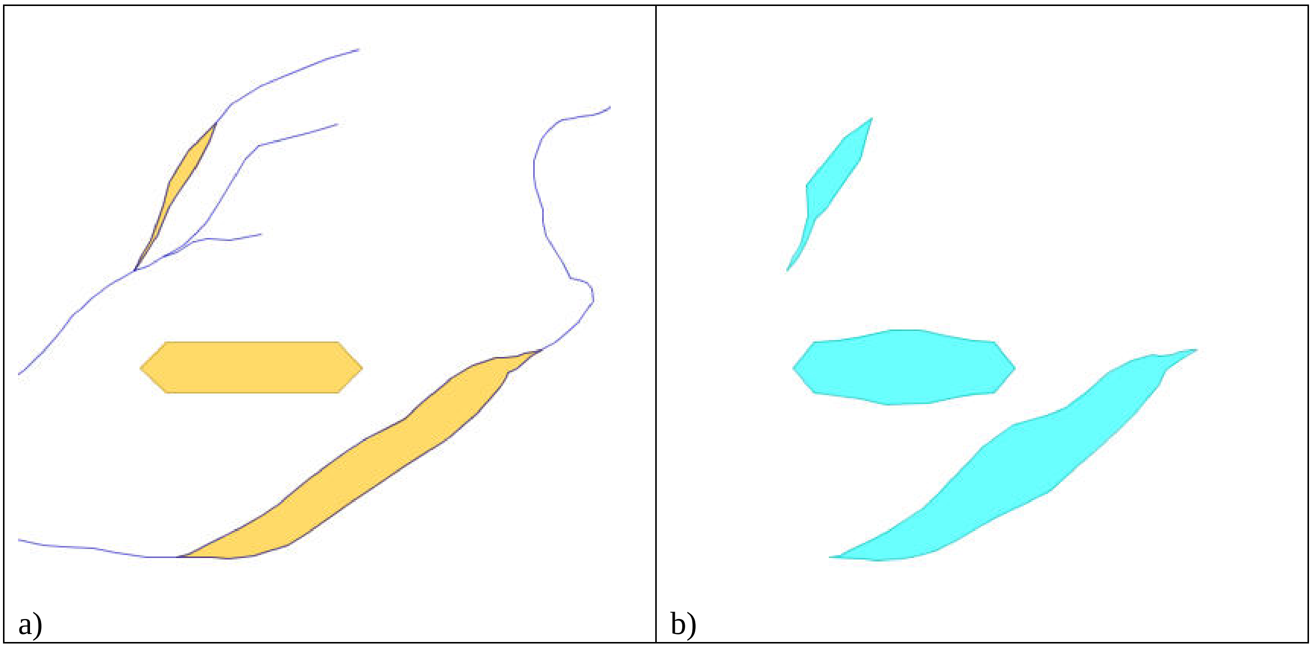


Figura 4: the enlarging of lone islands; in a) the input data, in b) the result; the shape in the middle was added to show the behaviour of the algorithm on a regular shape

Special cases

The approach described was implemented and tested on two datasets consisting each of a water polygons layer and network layer. The first is data in 5K scale of the Regione Veneto, representing the Piave river, the second is USGS data representing the Platter river subbasin.
Generalizing the two datasets required the algorithm to be extended to process some special cases: polygon holes not encircled by network edges and dangling edges inside a loop.

Holes in water polygons

A hole in a water polygon corresponds to an island but, if it is not encircled by network edges, it is not detected by the algorithm. These holes must be identified during the pre-processing and added to the list of the islands to process; they also must be treated differently: as they have no blocking edges, only the compactness and the water loss are evaluated during their amalgamation.

Dangling edges

In braided streams, it is possible that water surfaces from a braid bar, creating channels streaming out of an island; in the network these channels form dangling edges inside a loop.
During generalization these edges should be detected as they have to be generalized accordingly with the loop containing them: if the loop containing them is amalgamated, they should be deleted as well; in the case they also form a loop, this loop should be treated as a lone island with the exception that, if it is not big enough to be kept, it should be deleted (no enlargement allowed) and the edges should be deleted as well.

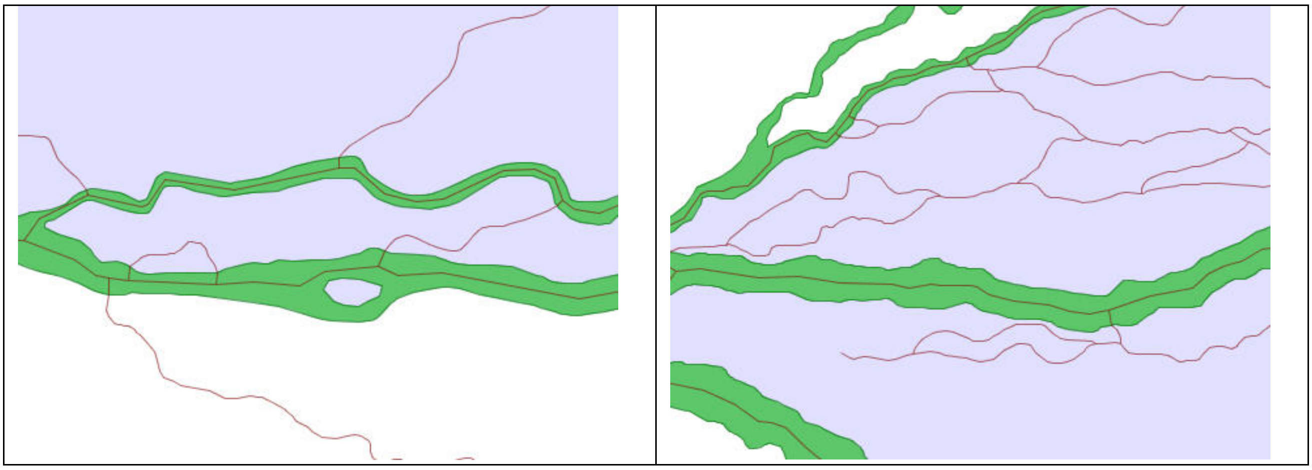


Figure 5: special cases: holes in the water polygons not encircled by the network (left), dangling edges forming a loop inside a bigger loop (right)

Results

The algorithm performed well on both the datasets.
The area thresholds have been experimentally set to 4000 sqm (minimum area) and 1,5 sqkm (maximum area).

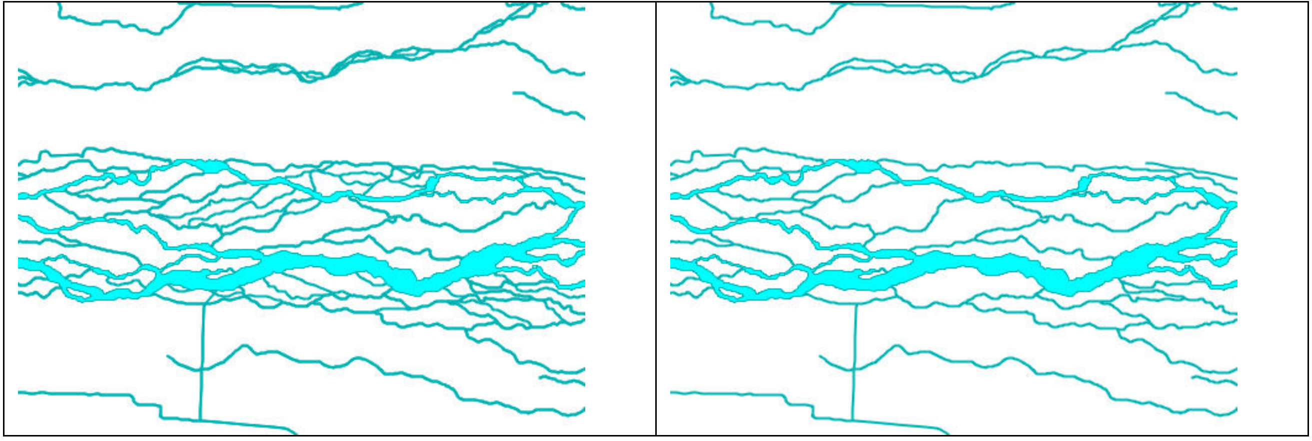


Figure 6: a) input data, b) generalized data

Application to a different context

The approach proposed in this paper can be used also in other contexts where a network has many branches. The algorithm has been tested with a part of a railroad network in proximity of a train station, where the presence of many railroad switches creates a topology similar to a braided river.

The blocking edge constraint prevents to create in the network hard bends that are not compatible with trains turning radius.

As in the case of river generalization, this algorithm does not provide a complete solution, as it can not handle edges that do not form loops (for example spur lines) and should be coupled with a network thinning algorithm able to generalize the single and paired railroad tracks; nevertheless the results are interesting and worth to be noted.

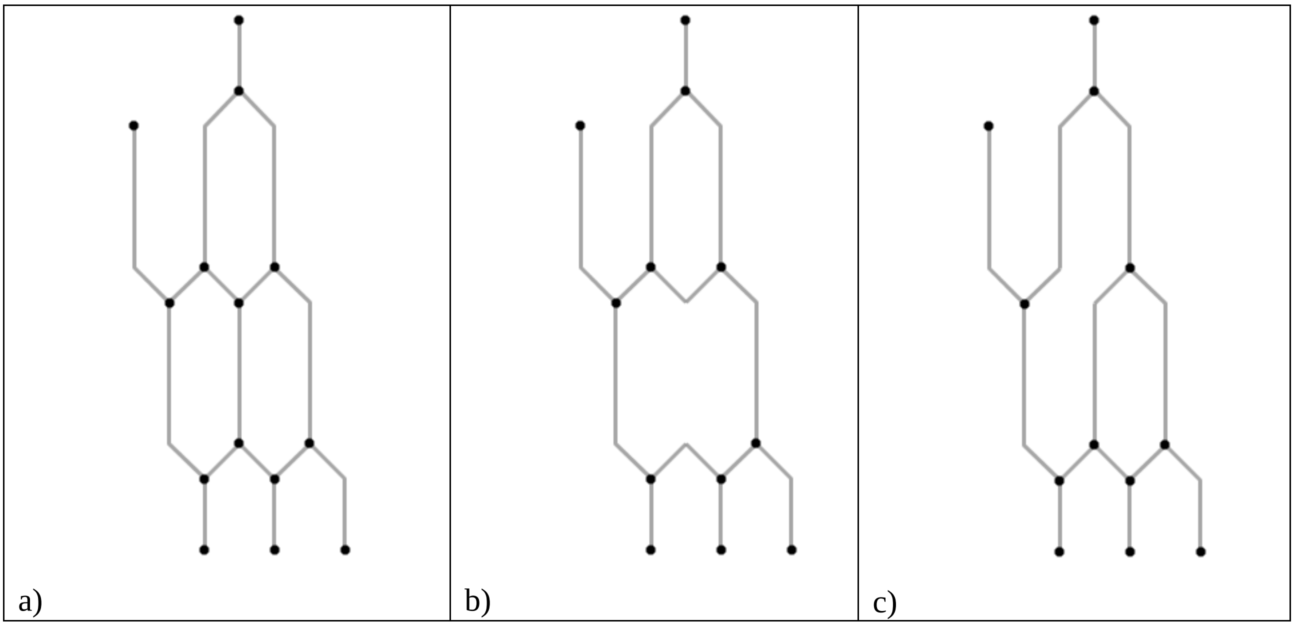


Figura 7: a schematized rail network: input (a), the result of a wrong area amalgamation (b), the result of a correct area amalgamation (c); the concept of blocking edges enables to avoid the situation in b

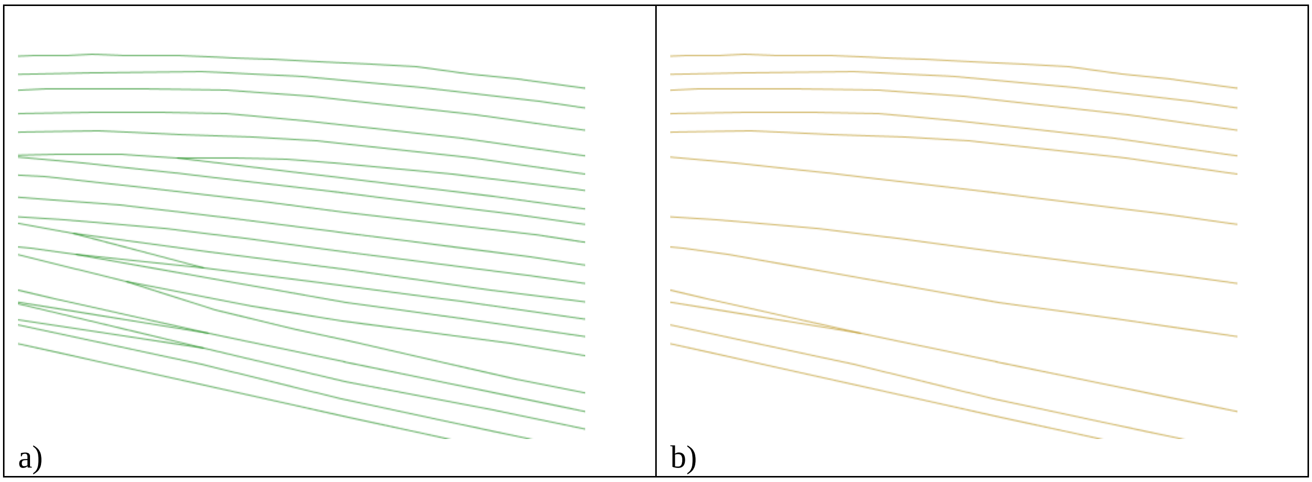


Figura 8: area amalgamation in a railroad network: a) input data, b) data after amalgamation

Conclusion

Braided streams are a challenging feature to generalize.

The area amalgamation technique described allows to overcome the limitations of the normal river network thinning techniques and to produce results that both preserve the characteristic cyclic network topology and enable to control the final shape of the generalized geometries.

The technique described can be useful also in other contexts where networks have many branches as in the case of a railroad network.

The approach however does not solve all the issues on braided streams.
The blocking edge technique relies too tightly on the shape of the network and where the network does not approximate correctly the shape of the river the algorithm to identify “blocking edges” may take wrong decisions.
Also, in some cases amalgamation by displacement could be used to reduce the “water loss”.
Finally, although the algorithm tries to remove only minor channels (i.e. narrow), the algorithm should probably be able to identify primary paths in the braided stream, in order to avoid to delete relevant channels.

Reference

Buttenfield BP, Stanislawski LV, Anderson-Tarver C and Gleason MJ (2013) Alternate Methods for Automatic Selection of Primary Paths Through Braided Hydrographic Networks, in Proceedings of the 16th ICA Generalisation Workshop on Generalisation and Multiple Representation, Dresden, Germany

Haunert JH (2007) Efficient area aggregation by combination of different techniques, in Proceedings of 10th ICA Workshop on Generalisation and Multiple Representation, Moscow, Russia

Jones CB, Bundy GL, Ware MJ (1995) Map generalization with a triangulated data structure. Cartography and Geographic Information Science 22(4):317–331

Horton RE (1945) Erosional Development of Streams and Their Drainage Basins; Hydrophysical Approach to Quantitative Morphology. in Bulletin of the Geological Society of America, 56, 275-370

Savino S, Rumor M, Zanon M (2011) Pattern Recognition and typification of ditches. In A. Ruas (Ed), Advances in Cartography and GISscience. Vol 1, Springer, 425-437 ISBN 9783642191428

Stanislawski L, Savino S (2011) Pruning of Hydrographic Networks: A Comparison of Two Approaches, in Proceedings of the 14th ICA Workshop on Generalisation and Multiple Representation, 2011, Paris, France

Stanislawski LV (2009) Feature Pruning by Upstream Drainage Area to Support Automated Generalization of the United States National Hydrography Dataset. Computers, Environment and Urban Systems, 33: 325-333

Savino S (2014) River network pruning by enrichment and density analysis. In D. Burghardt, C. Duchêne, W. Mackaness (Eds), Abstracting geographic information in a data rich world: methodologies and applications of map generalisation, Springer, 450p ISBN 9783319002026

Touya G (2007) River network selection based on structure and pattern recognition. In: Proceedings 23rd International Cartographic Conference, A Coruña, Spain, July 11-16

Thompson RC, Richardson DE (1999) The 'Good Continuity' Principle of Perceptual Organisation applied to the Generalisation of Road Networks, in Proceeding of the 19th International Cartographic Conference, Ottwawa, 1215-1225

van Oosterom P (2005) Variable-scale topological data structures suitable for progressive data transfer: The GAP-face tree and GAP-edge forest. Cartography and Geographic Information Science 32(4): 331–346