Clustering characteristics of the MCL algorithm

Before considering issues of complexity, scaling, and implementation in the next chapter, a series of examples is given illustrating various characteristics of the MCL algorithm. The performance criteria and the distance between partitions derived in the previous chapter will not be used for the small scale examples in this chapter, instead they are applied to randomly generated test graphs in Chapter 12. The first four sections in this chapter each give a short empirical account of respectively the genesis of attractor systems, the phenomenon of overlap, the effect of adding loops, and the effect of (varying) inflation on cluster granularity. In Section 10.5 the MCL algorithm is applied to various small torus graphs to see whether it is able to recognize a particular characteristic of their structure. In Section 10.6 it is shown that the MCL algorithm is in general not suited for detecting cluster structure if the diameter of the natural clusters is large. The examples used are neighbourhood graphs derived from two dimensional data. A common approach towards detection of clusters in neighbourhood graphs is by finding the borders that separate them. In Section 10.7 it is shown that early stages of the MCL process yield information that can be used to this end.

In order to describe the results of MCL-runs on various test-graphs for various parametrizations, the legend for MCL parametrizations is introduced. The experiments allowed input rows $e(i)$, $r(i)$ which have very simple structure. The row $e(i)$ simply consists of two’s only. The row $r(i)$ is constant on a tail of infinite length, and may assume another constant on a prefix of length $l$, where $l$ can be specified as well. This amounts to three parameters related to the input rows specifying MCL process parameters. The fourth parameter indicates whether loops are added to an input graph $G$. If this parameter assumes a value $c \in \mathbb{R}_{\geq 0}$, the program takes the graph $G + cI$ as actual input. The parameter labels, their meaning, and default setting are found in Table 1. The length $l$ of the initial prefix is indicated by ‘l’, the constant value assumed by $r(i)$ on the initial prefix by ‘r’, the constant value on the infinite postfix by ‘R’, and the loop weight by ‘a’ (stemming from auto-nearness). The main use of introducing a default setting is that in many examples the simplest possible parametrization is chosen, where the initial prefix length is equal to zero. The default setting corresponds with an MCL process for which $e(i) \equiv 2$ and $r(i) \equiv 2$ and no loops added.
### 10.1 Attractors

In practice, for input graphs constructed from real applications for the purpose of clustering (and thus with no strong symmetries present), the equivalence classes $E_1, \ldots, E_d$ (see Definition 8) tend to be singleton sets. In all situations observed so far where this is not the case, see e.g. the limit matrix in Figure 13 on page 53, the elements in an equivalence class of cardinality greater than one are the orbit of a graph automorphism. For the graph $G_3$ in Figure 10 on page 45, the nodes 8 and 10 share exactly the same set of neighbours, and are for that reason indistinguishable with regard to structural properties of the graph. The mapping on the node set of $G_3$ which only interchanges the nodes 8 and 10 is an automorphism of the graph $G_3$. As a second example, consider the graph $G_1$ in Figure 7 on page 43. An MCL run with parametrization $a = 1$ results in 4 clusters, each of which is a triangle in the graph, and where each node is an attractor. Printing attractors in boldface, this is the clustering $\{\{1, 2, 3\}, \{4, 5, 6\}, \{7, 8, 9\}, \{10, 11, 12\}\}$. The cluster $\{1, 2, 3\}$, and likewise the other clusters, is the orbit of either of its elements under rotation of $G_1$ around the symmetry axis orthogonal to the plane spanned by 1, 2, and 3.

Generally, attractors are located in local centres of density, which is best illustrated by large graphs with clear islands of cohesion. If a graph fits the picture, so to speak, of a ‘gradient of density’, the attractors are found in the thickest parts of the graph. This effect is to some extent illustrated by the graph in Figure 19. For this graph, it interferes, however, with another phenomenon that occurs when a graph possesses borders. In that case, the return probabilities of nodes which lie just before those borders, profit immediately and maximally after one expansion step from the ‘dead end’ characteristic of the border. The border of the graph in Figure 19 is the outline of the grid. This explains why all of the attractors in Figure 20 are nodes lying at distance one from the outline of the grid.

### 10.2 Overlap

The phenomenon of overlap in the setting of undirected graphs has only been observed so far for input graphs with specific symmetry properties. For these cases, if the MCL algorithm produces two clusters $C_1, C_2$ with nonempty intersection, there exists an automorphism which transforms $C_1$ into $C_2$ while leaving the intersection invariant. An example is the line graph on 7 nodes the associated matrix of which is found in Figure 14. The automorphism maps $i$ onto $7 - i$, $i = 1, \ldots, 7$, which leaves the intersection $\{4\}$...
invariant. Existence of such an automorphism means that the overlapping part forms a subset of the graph from which the graph looks the same in different directions. If those different directions correspond also with different islands of cohesion, it is rather nice if the overlapping part is not arbitrarily divided among the resulting clusters. Another example of this phenomenon can be found in Figure 20. Overlap occurs at several levels of granularity, and it always corresponds with a symmetry of the graph. For undirected graphs, the amount of possible overlap tends to be proportional to the amount of symmetry present. Naturally, any amount of overlap can be constructed by taking appropriate directed input graphs.

Small perturbations in the input graph generally do not affect the output clustering produced by the MCL algorithm. An exception to this is the case where overlap occurs, as discussed in Section 6.3. If the symmetry corresponding with the overlap is perturbed, the overlap disappears.

### 10.3 The effect of adding loops

For small graphs and graphs with bipartite characteristics such as rectangular grids, adding loops is a beneficial manoeuvre. The reason for this is the same as it was for $k$-path clustering. The possible dependence of the transition probabilities on the parity of the simple path lengths in the graph is removed. More generally, adding loops of weight $c$ to a graph has the effect of adding $c$ to all the eigenvalues in its spectrum, and negative eigenvalues are known to correspond with oscillatory behaviour of the associated matrix. The effect of adding loops on the output clusterings of the MCL algorithm is that connectedness (with respect to the input graph) of the clusters in the output clustering is promoted, and that the granularity of the output clustering is increased. The latter is reflected in the fact that adding loops increases the number of endclasses of the associated DAG of a $d$psd matrix.
10.4 The effect of inflation on cluster granularity

There is a clear correlation between the inflation parameter and the granularity of the resulting output. The higher the parameter \( r \), the more the inflation operator \( \Gamma_r \) demotes flow along long path distances in the input graph. This is illustrated for the graph \( G_5 \) in Figure 19. Figure 20 gives the result of six MCL runs for \( G_5 \) in which the inflation parameter is varied from 1.4 to 2.5, while all other parameters are kept the same (i.e. \( a = 1 \), \( E = 2 \)). Note that the corresponding overlapping clusterings are strongly related to each other. The set of all clusterings excluding the one corresponding with inflation parameter \( R = 1.4 \) is a set of nested overlapping clusterings. This is very satisfactory, as one expects clusters at different levels of granularity to be related to each other. The clusterings at the first three levels \( R = x, x \in \{1.4, 1.5, 1.7\} \), have good visual appeal. It holds for all clusterings that the sizes of the respective clusters are evenly distributed, except perhaps for the clustering with parameter \( R = 2.0 \).

The second example in which the inflation parameter is varied while other parameters are kept the same concerns the graph \( G_4 \) in Figure 18. It is derived from the graph \( G_1 \) in Figure 7 by replacing each of the 12 nodes in \( G_1 \) by a triangle. Note that \( G_4 \) is a simple graph: The length of the edges in the picture do not correspond with edge weights.

Now \( G_4 \) clearly allows two extreme clusterings \( P_1 = \{\text{singletons}(V)\} \) and \( P_4 = \{V\} \), a clustering \( P_2 \) in which each of the newly formed triangles forms a cluster by itself, and a clustering \( P_3 \) with 4 clusters in which each cluster consists of the 9 nodes corresponding to each for the input graph \( \text{TORUS}(10, x) \), \( x = 5, \ldots, 9 \).

\[
\begin{array}{ccc}
\text{Parametrization} & x & 1 \\
\text{Clusterings} & r & R \\
0 - 1.0 - 1.2 & P_4 & 5 \\
0 - 1.3 - 1.4 & P_4 & 6 \\
0 - 1.5 - 3.0 & P_2 & 2 \\
0 - 3.0 - \infty & P_1 & 0 \\
1 - 1.0 - 1.3 & P_3 & 3 \\
1 - 1.4 - 1.7 & P_3 & 3 \\
1 - 1.8 - 5.3 & P_2 & 8 \\
1 - 5.4 - \infty & P_1 & 7 \\
2 - 1.0 - 1.4 & P_4 & 2 \\
2 - 1.5 - 2.4 & P_3 & 3 \\
2 - 2.5 - 6.8 & P_2 & 2 \\
2 - 6.9 - \infty & P_1 & 2 \\
\end{array}
\]

\( a = 1 \) set everywhere

Table 2. MCL runs for the graph \( G_4 \) in Figure 18. The clusterings \( P_1, \ldots, P_4 \) are defined in the text above.

\[
\begin{array}{ccc}
x & 1 & r \\
R & 0.8 & 2.1 - 3.3 \\
2 & 1.0 & 2.1 - 2.8 \\
3 & 1.0 & 2.3 - 3.9 \\
4 & 1.2 & 2.1 - 2.7 \\
5 & 1.2 & 2.1 - 3.2 \\
6 & 1.2 & 2.1 - 4.0 \\
7 & 1.2 & 2.1 - 2.3 \\
8 & 1.2 & 2.1 - 2.9 \\
9 & 1.2 & 2.1 - 2.1 \\
\end{array}
\]

\( a = 1 \) set everywhere

Table 3. Parametrizations for which the \( MCL \) algorithm finds 10 clusters of size \( x \) each for the input graph \( \text{TORUS}(10, x) \), \( x = 5, \ldots, 9 \).
with 3 newly formed triangles. Clustering with parameters $a = 1$ $E = 2$ $R = x$, where $x$ varies, yields the following. Choosing $x \in [1.0, 1.2]$ results in the top extreme clustering $P_4$, choosing $x \in [1.3, 1.4]$ in the clustering $P_3$, choosing $x \in [1.4, 3.0]$ in the clustering $P_2$, and choosing $x \in [3.1, \infty]$ results in the bottom extreme clustering $P_1$. The range of $x$ for which the clustering $P_4$ results is small. This has to do with the fact that the clustering $P_4$ is rather coarse. The dependencies associated with $P_4$ correspond with longer distances in the graph $G_4$ than the dependencies associated with $P_3$. If the inflation parameter increases, the latter dependencies (in the form of random walks) soon profit much more from the inflation step than the former dependencies. By letting expansion continue a while before starting inflation, this can be remedied. Table 2 shows several parameter settings and the resulting clusterings.

The clusterings shown for the torus graphs, the tetraeder-shaped graphs in Figures 7 and 18, and the grid in Figure 19 illustrate that the MCL algorithm ‘recognizes’ structure even if the node degrees in the input graph are homogeneously distributed and the connectivity of the graph is high. The inflation parameter clearly is the main factor influencing the granularity of the output clusterings. The output clustering changes at specific values of the inflation parameter constant (either the prefix or the postfix value), and stays the same for the intervals in between. By prolonging expansion, coarser clusterings can be found.
10.5 Flow on torus graphs

The following examples are rectangular torus-graphs. A $k$-dimensional rectangular torus graph generalizes a ring graph in $k$ dimensions. It is most conveniently defined as a sum of ring graphs, defined on the Cartesian product of the respective node sets.

**Definition 24.** Let $(G_i = (V_i, w_i)), i = 1, \ldots, n$ be an $n$-tuple of simple graphs. The **sum graph** $S$ of $G_1, \ldots, G_n$ is defined on the Cartesian product $V_1 \times \cdots \times V_n$. Two vertices $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ are connected in $S$ if exactly one of the pairs $(x_i, y_i)$ is connected in $G_i$, and $x_i = y_i$ for the remaining $n - 1$ pairs.

**Definition 25.** The 1-dimensional **torus graph** or **ring graph** of cardinality $t$ is the simple graph defined on the integers modulo $t$: $0, \ldots, t - 1$, where there is an edge between $i$ and $j$ iff $i = j + 1 \pmod{t}$ or $j = i + 1 \pmod{t}$.

A graph is called a k-dimensional torus graph if it is the sum graph of $k$ ring graphs. It can be identified with a $k$-tuple $(t_1, \ldots, t_k)$, where $t_i$ is the cardinality of the node set of the $i^{th}$ ring graph. The torus graph corresponding with this $k$-tuple is denoted $\text{TORUS}(t_1, \ldots, t_k)$. □

Here I will use only 2- and 3-dimensional simple torus graphs. A 2-dimensional torus graph $\text{TORUS}(k, l)$ can be thought of as a rectangular grid of width $k$ and depth $l$, where nodes lying opposite on parallel borders are connected. In Section 6.3 it appeared that periodic MCL limits exist which have the same automorphism group as ring graphs. A two dimensional torus graph $G = \text{TORUS}(k, l)$ where $k = l$ has the same homogeneity properties as ring graphs. It is interesting to see what happens if $k > l$. Consider a node pair $(u_1, u_2)$ lying on a ring of length $l$ in $G$ at a (shortest path) distance $t \leq l$ from each other, and a node pair $(v_1, v_2)$ in $G$, also lying at distance $t$ from each other, but not lying on such a ring. The transition probability associated with going in $l$ steps from $u_1$ to $u_2$ is larger than the transition probability associated with going in $l$ steps from $v_1$ to $v_2$, because $u_1$ can reach $u_2$ in two ways along the ring on which they both lie, while this is not true for $v_1$ and $v_2$. Is it possible to find an MCL process in which this effect is boosted such that a clustering of $G$ in $k$ clusters of size $l$ each results? This is indeed the case, and it requires the usage of input rows $r(i)$ which are not constant everywhere. If $l$ is very close to $k$, it is furthermore beneficial to use an initial inflation parameter which is close to or smaller than 1. Without this, the return probability of each node grows too large before paths of length $l$ start to have influence, which is after $\lceil \log_2(l) \rceil$ expansion steps (assuming $e(i) \leq 2$). Table 3 shows parameter settings for which the MCL algorithm output divides the graphs $\text{TORUS}(10, x)$ in 10 clusters of cardinality $x$ each, $x = 5, \ldots, 9$. These are of course not the only parametrizations achieving this, but among the parametrizations found they lead to fast convergence of the MCL process.

The last torus example is the 3-dimensional torus graph $\text{TORUS}(3, 4, 5)$. A priori it is to be expected that the non-extreme clusterings which the MCL algorithm can possibly produce are the clustering $P_2$ corresponding with 20 subgraphs isomorphic to $\text{TORUS}(3)$ and the clustering $P_3$ corresponding with 5 subgraphs isomorphic to $\text{TORUS}(3, 4)$. Denote the top and bottom extreme clusterings by $P_1 = \{\text{singletons}(V)\}$ and $P_4 = \{V\}$ respectively. The table below gives four parameter ranges yielding the four clusterings $P_i$. 

---

[Note: The table and additional text would be included here, but are not shown in this response.]
10.6 Graph clustering and the vector model

In the upper left of Figure 21 a rectangular assembly of points in the plane is shown. A graph is defined on the black spots using the neighbourhood relation depicted to the right of this text (i.e. nodes correspond with black spots and there is an edge between two nodes if they are at most $\sqrt{5}$ units away). The weight of an edge is inversely proportional to the distance between the coordinates of its incident nodes. This is not essential for what follows, nor is the particular neighbourhood structure used. In Figure 21 three clusterings are depicted, corresponding with the simple parametrizations $R = 1.9$, $R = 2.3$, and $R = 2.7$. The MCL process applied to grid-like graphs such as these has the property that columns (equivalently, probability distributions of a node) begin to convergence towards a homogeneous state in the corners and borders first. While this happens, the converging parts of the distributions begin to assume the characteristics of a border themselves, as flow from the border region towards the centre is demoted. This explains the neat division of the graph into blocked patterns.

If the inflation parameter is chosen sufficiently low, the graph will be clustered into a single cluster. This requires considerable time, as the diameter of the graph is ten. Diagonally opposite corners build up distinctly different probability distributions, and it requires several expansion steps at low inflation parameter to let them equalize. Though it is possible (using the parametrization $R = 1.3$) in this simple setting, matters become more complicated if the graph is made part of a larger constellation. Four such constellations are depicted in Figure 22, where the one in the upper left is the same as before. The graphs on the grids are derived using the same neighbourhood structure as before, and Figure 22 shows the result of applying the MCL algorithm with parametrization $R = 1.3$.  

<table>
<thead>
<tr>
<th>Parametrization</th>
<th>$r$</th>
<th>$R$</th>
<th>Clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.2</td>
<td>1.0 (\text{to} ) 2.3</td>
<td>$P_4$</td>
</tr>
<tr>
<td>2</td>
<td>1.2</td>
<td>2.4 (\text{to} ) 3.3</td>
<td>$P_3$</td>
</tr>
<tr>
<td>3</td>
<td>1.2</td>
<td>3.4 (\text{to} ) 6.4</td>
<td>$P_2$</td>
</tr>
<tr>
<td>4</td>
<td>1.2</td>
<td>6.5 (\text{to} ) $\infty$</td>
<td>$P_1$</td>
</tr>
</tbody>
</table>

The torus examples illustrate the strong separating power of the MCL process. This is mainly interesting for a better understanding of the process, and probably not of much help in using the algorithm. The rewarding aspect of the torus examples is that abstract reasoning about the process applied to extreme cases is confirmed by experiments. The further experiments described below will exhibit a characteristic of the MCL algorithm that may be considered a weakness. It concerns the formation of clusters in graphs consisting of weakly connected grids, where certain clusters connect parts of different grids. The phenomenon is rather surprising at first, but it can be be understood in abstract terms. It is indicative for the fact that there are severe problems involved in applying graph cluster methods to neighbourhood graphs derived from vector data.
The upper right clustering is remarkable in that the two small satellite subgraphs form clusters together with regions of the large rectangle. This is explained by the effect that the small subgraphs 'inflate' the probability distributions of the nodes of the rectangle lying on the opposing border (i.e. cause them to be less homogeneous). Random walks departing from these border nodes crossing over to the satellite subgraphs have a relatively small probability initially, but due to the low inflation parameter, and the fact that a satellite subgraph has a great absorbing quality, these probabilities win out easily in the end. This is further illustrated by Figure 23, in which all four constellations are again clustered for parametrization $R = 1.5$. The crossing characteristics of the upper right constellation are now much less dramatic, but still present.

The clustering of the lower left constellation in Figure 22 is remarkable in that the natural bipartition of the large rectangle is rather along the south/north axis than the along the east/west axis. If the rectangle is clustered without satellite systems, then this is the only bipartition which can possibly result, which is explained by the fact that flow is sooner bound along the north/south axis than it is along the east/west axis. This can be
Figure 22. Graph from Figure 21 with different satellite constellations clustered with constant inflation equal to 1.3. The initial edges are defined by the neighbourhood structure depicted on page 117. The upper right clustering exhibits clear border crossing caused by inhomogeneity differentiation. The lower left clustering induces an unnatural bipartition of the large rectangle.

compared to the clustering of a \((k, l)\) torus graph, \(k < l\), for which an MCL process will never result in a clustering into \(k\) rings of size \(l\).

In the lower right constellation of Figure 22 it is noteworthy that the corner nodes are all attracted to the clustering corresponding with the neighbouring south or north satellite rather than the east or west satellite. This is probably caused by the fact that the border between the ‘s’ and ‘v’ clusters is closer by than the border between the ‘a’ and ‘u’ clusters, so that the distribution of the corner node transition probabilities is steeper along the south/north axis than it is along the east/west axis. The situation is different for the constellation in Figure 23, because the convergence of the cluster structure corresponding with the satellite systems took place much sooner than the convergence of the four large clusters in the rectangle.
The behaviour of the MCL process for the grid-like graphs in this section has two reasons. The fact that the natural clusters have sizes differing by orders of magnitudes is an important factor. However, examples exhibiting the same border-crossing behaviour can be constructed with sets of grids of the same size, simply by tiling them such that corners are aligned with borders. The most significant factor is the prolonged expansion at low inflation parameter required in order to equalize the probability distributions of opposing corners and opposing borders. The main characteristics of the subgraph corresponding with the large rectangle are that it is rather large (216 nodes) and has relatively large diameter. The process of equalizing distributions via expansion at low inflation values is costly in terms of space due to the large number of elements, and it is costly in terms of time due to the large diameter. The time requirement causes the process to be sensitive to perturbations in the input graph. This was demonstrated by adding small extra grids; similar phenomena occur if for example some nodes are given greater initial return probabilities than other nodes.
The significance of these observations is that one must be careful in applying the MCL algorithm to neighbourhood graphs derived from vector data, especially if it is known or unknown whether the diameter of the natural clusters is large. If it is known that this quantity is not too large, then the MCL algorithm will work well. This is illustrated by the geometric graph example shown in Chapter 1. The principal cause for the behaviour of the MCL algorithm — large diameter and dimension of clusters — will affect any graph clustering algorithm that is grounded on the principles discussed in Chapter 5. The computation of long distance dependencies, be it via random walks, paths, or shortest paths, will in each case be costly and prone to be sensitive to local fluctuations in density of the vectors inducing the neighbourhood graph.

The detection of clusters in a grid-like setting may be better served by a procedure such as border-detection. It is interesting to try and devise such a procedure using flow formulation and the graph cluster paradigm. The following section describes a small experiment in this direction.

**10.7 Towards border detection using flow simulation**

Clustering in the setting of (graphs derived from) grids has its limitations, as argued in the previous section. It was seen that clusters which correspond with large regions are difficult to detect using the graph clustering paradigm. However, the early stages of flow simulation yield information that can be used to detect the presence of borders between regions that have for example different shades (grey-levels) or colour. This is first illustrated for a simple example in which the mutual attraction between nodes depends on the associated Euclidean distance only. The graph shown on the left of Figure 24 is defined on four neighbouring rectangles, each of size $9 \times 6$, using the neighbourhood relationship on page 117. The borders of the four rectangles are thus weakly connected. The MCL process was applied to this graph with standard parameters, i.e. inflation and expansion both equal to two. The graph shown on the right of the same figure corresponds with the sixth iterand $T$ of the process. The grey level of a node $i$ is inversely proportional to the ratio $T_{ii}/c_i$, where $c_i$ denotes the mass centre of order 2 of the $i^{th}$ column of $T$. Nodes $i$ at the borders of the four rectangles thus have low value $T_{ii}/c_i$, and nodes lying in the centre have a high value. This is explained by the fact that convergence begins first at the borders, with border nodes becoming attracted to nodes which are one step away from the border. The nodes in the centre initially develop rather homogeneous and stable probability distributions.

Next consider an image in the form of a bitmap with different grey-levels. An example is given in Figure 25. This image is a bitmap of dimension $284 \times 380$, where each pixel may assume 256 grey-levels. A graph was created from this bitmap with a node for each pixel. Horizontally, vertically, and diagonally neighbouring nodes (pixels) were connected via an edge with weight inversely proportional to the difference in grey level between the pixels. Loops were added such that the return probability of a node equalled the mass centre of order 2 of the stochastic vector associated with this node.
A variant of the MCL process was applied to this graph which incorporated aggressive pruning. That is, after each matrix multiplication, all nodes were allowed to have at most nine neighbours. The natural choice for this is to pick the nine neighbours with greatest associated probability (see also Chapter 11). After removal of all but the nine largest neighbours of a node, the corresponding pruned column is rescaled to have weight one again. If a pixel is situated in a homogeneous region, then for early stages of the process the neighbours with largest associated transition probability will be just the set of its initial neighbours (including itself), since there is no reason for the symmetry to be broken. Moreover, the return probability will be the largest value in the column, since the symmetry leaves no room for any other direction of attraction. On the other hand, if a pixel is situated near a border or edge in the image, then the distribution of the associated probability vector will be asymmetric with respect to the initial neighbourhood constellation. This will cause the emergence of a direction of attraction, just as in the example in Figure 24. Figure 26 shows the result of interpreting the third iterand of the resulting process using the same principle as in Figure 24 and using a threshold for the indicator value $T_{ii}/c_i$.

The resulting image (Figure 26) indeed shows that the indicator value causes homogeneous regions to become blank and causes clear borders in the image to reappear as such. This is a tentative result, as there is information present in the processed image that hampers further contour detection (i.e. a true symbolic representation of borders), and there is also information lacking that one would like to be present (i.e. the arcs in the original image do not fully reappear in the processed image). However, it must be kept in mind that the chosen approach was extremely simple and naive. This use of the MCL process may well serve as an intermediate processing step in more sophisticated approaches. The value of the MCL process in this application is that it offers a generic modus via which neighbouring and super-neighbouring pixels may influence each other.
Figure 25. San Giorgio Maggiore in Venice.
Figure 26. Result of a bordering process based on the MCL process.