Centrum voor Wiskunde en Informatica

A cluster algorithm for graphs

S. van Dongen

Information Systems (INS)

INS-R0010 May 31, 2000

Report INS-R0010
ISSN 1386-3681

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

A Cluster Algorithm for Graphs

Stijn van Dongen
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

ABSTRACT
A cluster algorithm for graphs called the $Markov\ Cluster\ algorithm$ (MCL algorithm) is introduced. The algorithm provides basically an interface to an algebraic process defined on stochastic matrices, called the MCL process. The graphs may be both weighted (with nonnegative weight) and directed. Let G be such a graph. The MCL algorithm simulates flow in G by first identifying G in a canonical way with a Markov graph G_1. Flow is then alternatingly expanded and contracted, leading to a row of Markov Graphs $G_{(i)}$. Flow expansion corresponds with taking the k^{th} power of a stochastic matrix, where $k \in I\!N$. Flow contraction corresponds with a parametrized operator Γ_r, $r \geq 0$, which maps the set of (column) stochastic matrices onto itself. The image $\Gamma_r M$ is obtained by raising each entry in M to the r^{th} power and rescaling each column to have sum 1 again. The heuristic underlying this approach is the expectation that flow between dense regions which are sparsely connected will evaporate. The invariant limits of the process are easily derived and in practice the process converges very fast to such a limit, the structure of which has a generic interpretation as an overlapping clustering of the graph G. Overlap is limited to cases where the input graph has a symmetric structure inducing it. The contraction and expansion parameters of the MCL process influence the granularity of the output. The algorithm is space and time efficient and lends itself to drastic scaling. This report describes the MCL algorithm and process, convergence towards equilibrium states, interpretation of the states as clusterings, and implementation and scalability. The algorithm is introduced by first considering several related proposals towards graph clustering, of both combinatorial and probabilistic nature.

2000 Mathematics Subject Classification: 05B20, 15A48, 15A51, 62H30, 68R10, 68T10, 90C35.
Keywords and Phrases: Clustering, graph clustering, graph partitioning, random walk, Markov matrix, flow simulation.
Note: Revised version of the report [8]. A more mathematically oriented account on the MCL process is given in [11], establishing that under certain weak conditions the iterands of the MCL process posses structure admitting a cluster interpretation. Various experiments conducted on a wide range of test–graphs are described in [10]. The latter report also describes a generic graph clustering performance measure and a distance defined on the space of partitions. The work was carried out under project INS–3.2, Concept Building from Key–Phrases in Scientific Documents and Bottom Up Classification Methods in Mathematics.

1. Introduction

In this report the Markov Cluster (MCL) algorithm is introduced, a cluster algorithm for graphs which is based on simulation of flow expansion and flow contraction in graphs. The algorithm is specifically suited to sparse graphs, i.e. graphs for which the average node degree is an order of magnitude smaller than the number of nodes in the graph. The algorithm is motivated by considering how the concept of 'cluster' in the setting of sparse graphs can be formalized to some extent. The report is a revised version of [8], and corresponds with chapters 5, 6, and 11 in the PhD thesis [9].

The idea that clustering in the setting of sparse graphs may very well merit from a separate approach, as opposed to viewing this problem as a minor variant of clustering in a more abstract setting, does not seem to be widespread in the cluster analysis and pattern recognition communities. In graph partitioning clustering is sometimes used as a preprocessing step, and in this area of research there exist publications that deal specifically with clustering in the setting of graphs — see [6, 7, 15, 24, 43] and the survey article [3].

The classic setting in cluster analysis is one where entities are represented by vectors of numerical scores (here termed the *vector model*). The (dis)similarity between two elements is in this case defined in terms of a measure on the difference between the vectors associated with the elements. In the setting of graphs, the relationship between two elements is of the kind 'share a property or not' or 'one refers to the other' (here termed the *graph model*). In graph clustering, the goal is to find a clustering of the node set of a graph such that there are few edges in between different clusters, and many edges within each cluster on its own. The fundamental difference between the graph model and the vector model is that in the latter model the (dis)similarities between elements are immediately available, and that the model inspires geometric notions such as *convex hull*, *geometric mean*, *separating hyperplanes*, et cetera. The notion of a cluster is closely related to the *density* of the distribution of the vectors over the vector space. Clusters should induce regions of the vector space where the density is relatively high, and they should generally be separated by regions of the vector space where the density is relatively low. On the other hand, a graph is nothing more than a set of nodes with a notion of connectivity attached to it. Clusters can not be measured in terms of the location of the nodes, they can only be measured in terms of the incidence relation defined on the cartesian product of the node set.

There seem to be few publications linking (graph clustering in the setting of) graph partitioning with cluster analysis or pattern recognition. Several reasons account for this. Cluster analysis can be seen as a unifying framework where exploratory techniques are gathered from different application areas such as biology, chemistry, market research, medicine, and psychology. In this framework the methods are studied in abstracto, separated from application, data, and implementation. The predominant data model in each of these application areas is the vector model described above. The graph model is relatively young in comparison, and does not get much attention in the cluster analysis monographs [4, 12, 13, 16, 22, 23, 25, 29, 34, 38, 42]. Another aspect worth mentioning is that classic methods such as the linkage–based methods (c.q. single link and complete link clustering) are often formulated in terms of *threshold graphs* derived from dissimilarity spaces (see Section 4), and these methods are consequently easy to apply to graphs per se. However, threshold graphs are merely a means of notation, and this approach does not seem particularly suited for finding cluster structure in graphs in general. It fails entirely with respect to the basic challenge of finding cluster structure in simple graphs. A *neighbourhood graph* is another type of graph which is sometimes derived from metric dissimilarity data [22]. This corresponds again with a particular manner of selection and representation, where the transformation step is specifically motivated by the geometric nature of the original data. The clustering methods (c.q. heuristics) applied to neighbourhood graphs depend critically on properties of this transformation step, so they are not suited for graphs in general.

In graph partitioning (i.e. partitioning the node set of a graph into subsets with prescribed sizes such that the total weight of the edges between different subsets is minimal) clustering is sometimes used as an intermediate processing step [3]. Graph partitioning is a well–defined optimization problem due to the fact that the partition sizes are prescribed. The research in this area is characterized by its cohesive nature, a tradition of benchmarking, and demand from industry. There is a set of established (e.g. spectral, move-based, multi-level) techniques, that are continually being refined, extended, and combined in novel ways. The application areas (some of which fall under the common denominator of VLSI design, see also [3]) generate problems with sizes obeying some variant of Moore's law (e.g. doubling every three year or so). The contrast with the exploratory nature of classic applications in cluster analysis is quite clear. The issue is discussed in more detail in [9]. This report deals exclusively with clustering in the setting of (sparse) graphs.

2. Introductory description of the *MCL* algorithm

The basic idea underlying the *MCL* algorithm and process is that dense regions in sparse graphs correspond with regions in which the number of k–length paths is relatively large, for small $k \in \mathbb{N}$. Random walks of length k thus have higher probability for paths with beginning and ending in the same dense region than for other paths. This is especially true if one looks at the subset of all random walks departing from a specific node. If this node is situated in a dense region, random walks departing from it will in general have a tendency to stay in the same region. The crucial element in the *MCL* algorithm is that this effect is deliberately boosted by an iterative procedure. First, an input graph G is mapped in a generic way onto a Markov matrix M_1. Then the set of transition probabilities is iteratively recomputed via expansion and inflation. The expansion step corresponds with normal matrix multiplication (on stochastic matrices), the contraction step corresponds with a parametrized operator Γ_r, called the *inflation operator*, which acts column–wise on (column) stochastic matrices. Henceforth, the term inflation will be used rather than contraction. Via expansion, nodes are able to see new neighbours; via inflation, favoured neighbours are further promoted, and less favoured neighbours are further demoted. For nearly all undirected graphs G, this process triggered by G converges very fast. The structural characteristics of the matrix limit of the process may be very different from the initial Markov matrix M_1. The associated graph of the matrix limit can have a larger number of connected components than the original input graph. This is in fact what makes the algorithm work, since the strongly connected components of the limit, joined with the respective node–sets that reach them, are interpreted as an overlapping clustering of the original graph. As in the usual Markov process, the 'nice' limits are idempotent matrices.

An infinite sequence consisting of repeated alternation of expansion and inflation constitutes a new algebraic process called the Markov Cluster (*MCL*) process. If the inflation operator Γ_r is parametrized such that all inflation steps correspond with the identity operator, a normal Markov process results. Interpretation of the limit then yields a clustering which corresponds with the set of connected components of the original graph. The inflation operator does not distribute over the normal matrix product, as Γ acts on matrices column–wise. For a normal Markov process, the columns of any iterand lie within the convex hull of the columns of any previous iterand. This is not true for the *MCL* process, which is due to Γ again. However, the *MCL* process has remarkable convergence properties. The 'nice' equilibrium states of the process are easily derived, and in practice the algorithm converges nearly always to such a limit. Exceptions to this rule are quite rare. The only ones found so far were made by construction, and agree with heuristic considerations.

For all testcases described in [10], the correlation between input graph, algorithm parameters, and output clustering is in line with heuristic considerations. The examples in [10] show surprising strengths of the algorithm. Most notable among these are separating power and the absence of chaining. All clusterings that are found have the property that the clusters correspond with regions in which there are relatively many k–length paths within. In this sense, it is impossible to find bad clusterings. The number of clusters is influenced by the flow characteristics of the *MCL* process. The cluster granularity can be affected by varying the flow parameters, but the number of clusters need not (and can not) be specified explicitly. The parametrizations of the *MCL* process which are useful for clustering purposes, generally lead to intermediate matrix iterands and equilibrium states which are very sparse in a 'weighted' sense. That is, a column may have many nonzero entries, but most of them are very small compared to the largest entries within the column. This gives the means to scale the algorithm drastically, by applying columnwise pruning.

3. Organization

Notations and definitions are covered in Section 4. Section 5 contains a short account of three related proposals towards graph clustering. They are formulated in terms of *path numbers*, *random walks*, and *shortest paths*. The proposal is made to assemble these notions under the somewhat grandiloquent label *graph clustering paradigm*. In Section 6 proposals towards graph clustering that have a combinatorial nature are discussed. A relaxation of one of them is the subject of Section 7. It is called k-path clustering and uses path numbers to detect cluster structure via single link clustering. This method links the combinatorial cluster notions with the MCL algorithm, as the starting point for the MCL algorithm is a localized version of k-path clustering. In Section 8 probabilistic cluster algorithms based on the ideas in Section 5 are briefly described. Random walks on graphs are introduced, corresponding with a localization of the context in which k-path clustering is applied. The standard way of describing a random walk on a graph associates a particular discrete Markov chain with the graph, and such is also the setup here. An example of (deterministically computed) random walks on an undirected graph possessing (weak) cluster structure is given. The initial characteristics of this stochastic process (c.q. Markov chain) are similar to phenomena observed in applying k-path clustering to the same graph (Section 7) but in the limit of the process all evidence of cluster structure has withered away. A new operator called *inflation* is inserted into the process, and an example run using the same input graph results in a limit which induces a cluster interpretation of the input graph in a generic way. The MCL algorithm and MCL process are formally described at the end of Section 8. The relationship between the MCL process and cluster interpretation of graphs is the subject of Section 9. Section 10 gives mathematical properties of the inflation operator Γ. In Section 11 the theoretically conceivable equilibrium states of the MCL process are categorized. Examples are given for each of the introduced classes. A class of symmetric circulant matrices for which matrix squaring and inflation act as each other's inverse is the subject of Section 12. In Section 13 local convergence properties of the MCL process are studied. The class of nice equilibrium states[1] is subdivided into two categories. It is shown that in the neighbourhood of the equilibrium states in the first subclass the MCL process converges quadratically towards equilibrium. Then it is shown that the equilibrium states S in the second subclass are instable, but that the MCL process converges quadratically at least on a macroscopic scale, once close enough to such an equilibrium state S. That is, it is proven that the structural form of the elements of MCL process converges towards the same block structure as present in S. Roughly speaking, the conclusion is that the phenomenon of cluster overlap is instable in nature, and that otherwise the instability of an equilibrium state, c.q. perturbation followed by convergence towards another equilibrium state, does not change the associated clustering. Section 14 is concerned with complexity and scalability of the algorithm. It is shown that the algorithm can be scaled drastically for large graphs in which the diameters of the natural clusters is relatively small.

In [11] conditions are given under which iterands of the MCL process have real c.q. nonnegative spectrum, and which imply the presence of generalized cluster structure in the iterands. The basic result is that for symmetric input matrix M, all iterands of the MCL process are guaranteed to be diagonally similar to a symmetric matrix. If such an iterand (matrix) has in addition nonnegative spectrum, than determinantal inequalities induce an ordering among the diagonal entries of the matrix which generalizes the mapping from nonnegative idempotent matrices onto overlapping clusterings given here in Definition 8. This ordering is also used in [11] to characterize the working of the inflation operator on its argument matrix.

4. Notation and definitions

This section introduces the terminology needed for graphs, (dis)similarity spaces, and clusterings. Single link and complete link clustering are discussed in some greater detail, because these are methods typically applied to dissimilarity data derived from attribute spaces, and are yet often formulated in graph–theoretical terms.

Graphs

Definition 1 *Let V be a finite collection of elements, enumerated $v_1, \ldots, v_t$.*

i) A **weighted graph** *G on V is a pair (V, w), where w is a function mapping pairs of elements of V to the nonnegative reals: $w : V \times V \to \mathbb{R}_{\geq 0}$.*

a) G is called **undirected** *if w is symmetric, it is called* **directed** *otherwise.*

[1] The states correspond with matrices which are idempotent under both expansion and inflation, the MCL process converges quadratically around these states, and they allow a generic mapping onto overlapping clusterings.

b) G is said to be **irreflexive** *if there are no* loops *in G, that is, $w(v,v) = 0, \forall v \in V$.*

ii) A **dissimilarity space** *$D = (V,d)$ is a pair (V,d), where s is a symmetric function mapping $V \times V$ to $I\!R_{\geq 0}$, satisfying $s(u,v) = 0 \iff u = v$. The function d is called a* **dissimilarity measure** *or* **dissimilarity coefficient**.

iii) A **similarity space** *is a pair (V,s), where s is a symmetric function mapping $V \times V$ to $I\!R_{>0} \cup \{\infty\}$, satisfying $s(u,v) = \infty \iff u = v$. The function s is called a* **similarity measure** *or* **similarity coefficient**.

The elements in V are called the **nodes** *of G. The* **dimension** *of the graph G is defined as the cardinality t of its node set V.*

In this thesis, I shall use similarity coefficients in the exposition of k-path clustering in Section 7.

Let $G = (V,w)$ be a weighted directed graph with $|V| = t$. The **associated matrix** of G lying in ${I\!R_{\geq 0}}^{t \times t}$, denoted $\mathcal{M}_G$, is defined by setting the entry $(\mathcal{M}_G)_{pq}$ equal to $w(v_p, v_q)$. Given a matrix $M \in {I\!R_{\geq 0}}^{N \times N}$, the **associated graph** of M is written $\mathcal{G}_M$, which is the graph (V,w) with $|V| = N$ and $w(v_p, v_q) = M_{pq}$.

An equivalent way of representing a weighted graph G is by identifying G with a triple (V,E,w), where the *edge set* E is a subset of V^2 and where w is a positive weight function defined on E only. A graph represented by such a triple (V,E,w) is in 1–1 correspondence with a graph representation (V,w') (according to Definition 1), by setting $w'(u,v){=}a{>}0$ iff $e{=}(u,v){\in}E$ and $w(e){=}a$, and setting $w'(u,v){=}0$ iff $e{=}(u,v){\notin}E$. The second representation leads to the generalization of graphs called **hypergraph**. A weighted hypergraph is a triple (V,E,w) where the hyperedge set E is a subset of the powerset $\mathcal{P}(V)$, and where w is a weight function on E as before.

Matrices and graphs of dimension N are indexed using indices running from 1 to N. If u,v are nodes for which $w(u,v) > 0$, I say that there is an arc going from v to u with weight $w(u,v)$. Then v is called the **tail node**, and u is called the **head node**. The reason for this ordering lies in the fact that graphs will be transformed later on into stochastic matrices, and that I find it slightly more convenient to work with column stochastic matrices than with row stochastic matrices. The **degree** of a node is the number of arcs originating from it. A graph is called **voidfree** if every node has degree at least one.

A **path** of length p in G is a sequence of nodes $v_{i_1}, \ldots, v_{i_{p+1}}$ such that $w(v_{i_{k+1}}, v_{i_k}) > 0$, $k = 1, \ldots, p$. The path is called a **circuit** if $i_1 = i_{p+1}$, it is called a **simple path** if all indices i_k are distinct, i.e. no circuit is contained in it. A circuit is called a **loop** if it has length 1. If the weight function w is symmetric then the arcs (v_k, v_l) and (v_l, v_k) are not distinguished, and G is said to have an **edge** (v_l, v_k) with weight $w(v_l, v_k)$. The two nodes v_l, v_k are then said to be connected and to be **incident to the edge**. A **simple graph** is an undirected graph in which every nonzero weight equals 1. The simple graph on t nodes in which all node pairs $u,v,u \neq v$, are connected via an edge (yielding $t(t-1)$ edges in all) is denoted by K_t, and is called the **complete** graph on t nodes. A weighted directed graph for which $w(u,v) > 0, \forall u \neq v$, is called a **weighted complete** graph. A weighted directed graph for which $w(u,v) = 0$ for some (or many) pairs (u,v) is called a **weighted structured** graph.

Let $G = (V,w)$ be a directed weighted graph. A **strongly connected component** of G is a maximal subgraph H such that for every ordered pair of nodes x,y in H there is a path from x to y in H. If G is undirected, then the strongly connected components are just called the **connected components**, and G is called **connected** if there is just one connected component (equalling G itself). For G directed, a **weakly connected components** is a maximal subgraph H containing at least one strongly connected component C and all nodes x in G such that there is a path in G going from x to an element of C (and thus to all elements of C). Weakly connected components can thus overlap, but they always contain at least one strongly connected component not contained in any of the other weakly connected components.

Let $G = (V, w)$ be a directed weighted graph $G = (V, w)$. In this thesis the interpretation of the weight function w is that the value $w(u, v)$ gives the *capacity* of the arc (path of length 1) going from v to u. Let G be a simple graph, let $M = \mathcal{M}_G$ be its associated matrix. The capacity interpretation of the weight function w is very natural in view of the fact that the pq entry of the k^{th} power M^k gives exactly the number of paths of length k between v_p and v_q. This can be verified by a straightforward computation. The given interpretation of the entries of M^k extends to the class of weighted directed graphs, by replacing the notion 'number of paths between two nodes' with the notion 'capacity between two nodes'.

The graph which is formed by adding all loops to G is denoted by $G + I$. In general, if Δ is a nonnegative diagonal matrix, then $G + \Delta$ denotes the graph which results from adding to each node v_i in G a loop with weight Δ_{ii}.

Partitions and clusterings

A **partition** or **clustering** of V is a collection of pairwise disjoint sets $\{V_1, \ldots, V_d\}$ such that each set V_i is a nonempty subset of V and the union $\cup_{i=1,\ldots,d} V_i$ is V. A partition $\mathcal{P}$ is called (**top** respectively **bottom**[2]) **extreme** if respectively $\mathcal{P} = \{V\}$ and $\mathcal{P} = \{\text{singletons}(V)\} = \{\{v_1\}, \ldots, \{v_t\}\}$. A **hierarchical clustering** of V is a finite ordered list of partitions $\mathcal{P}_i, i = 1, \ldots, n$ of V, such that for all $1 \leq i < j \leq n$ the partition $\mathcal{P}_j$ can be formed from $\mathcal{P}_i$ by conjoining elements of $\mathcal{P}_i$, where $\mathcal{P}_1 = \{\text{singletons}(V)\} = \{\{v_1\}, \ldots, \{v_t\}\}$ and $\mathcal{P}_n = \{V\}$. An **overlapping clustering** of V is a collection of sets $\{V_1, \ldots, V_d\}$, $d \in N$, such that each set V_i is a nonempty subset of V, the union $\cup_{i=1,\ldots,d} V_i$ is V, and each subset V_i is not contained in the union of the other subsets $V_j, j \neq i$. The latter implies that each subset V_i contains at least one element not contained in any of the other subsets, and this in turn implies the inequality $d \leq t$.

Let s be a similarity coefficient defined on $V = \{v_1, \ldots, v_t\}$. Let $s_1, \ldots, s_n$ be the row of different values that s assumes on $V \times V$, in strictly descending order and with the value 0 added. Remember that $s(u, u) = \infty, u \in V$. Thus, $\infty = s_1 > s_2 > \cdots > s_n = 0$. The **single link clustering** of the pair (V, s) is the nested collection of partitions $\mathcal{P}_i, i = 1, \ldots, n$, where each $\mathcal{P}_i$ is the partition induced by the transitive closure of the relation in which two elements u, v, are related iff $s(u, v) \geq s_i$. According to this definition, subsequent partitions may be equal, $\mathcal{P}_1 = \{\text{singletons}(V)\}$, and $\mathcal{P}_n = \{V\}$. The fact that at each similarity level s_i the single link clustering results from taking the transitive closure implies that the clustering coincides with the connected components of the **threshold graph** of (V, s) at threshold level s_i. This is simply the graph[3] on t nodes where there is an edge between u and v iff $s(u, v) \geq s_i$.

The **complete link clustering** of the pair (V, s), is usually procedurally defined as follows. The bottom partition $\mathcal{P}_1$ is again taken as $\{\text{singletons}(V)\}$. Each clustering $\mathcal{P}_k$, $k > 1$, is subsequently defined in terms of $\mathcal{P}_{k-1}$ by uniting the two clusters $\mathcal{C}_x$ and $\mathcal{C}_y$ of $\mathcal{P}_{k-1}$ for which the threshold level s such that [*the subgraph on* $\mathcal{C}_x \cup \mathcal{C}_y$ *in the threshold graph of* (V, s) *at level* s *is complete*] is maximal. Equivalently, $\mathcal{C}_x$ and $\mathcal{C}_y$ are such that the maximum of the minimal similarity in the restriction of the similarity space (V, s) to $\mathcal{C}_X \cup \mathcal{C}_Y$, is assumed for $X = x$ and $Y = y$. It is not very satisfactory from a mathematical point of view that the clusterings at a given level depend on the previous clusterings. It would be more elegant to define a clustering at a given threshold level as all maximal cliques in the corresponding threshold graph. The drawback is that it will in general result in an overlapping clustering with many clusters. Moreover, different clusters may have large overlap and small symmetric difference. Many variants of this type of complete linkage have been suggested [19, 23, 31], by first forming all maximal cliques at a given threshold level, and subsequently joining clusters (which are cliques) under the transitive closure of some similarity between clusters, e.g. sharing at least k neighbours. The computational requirements of such methods are huge, and they are mostly presented as an exercise in mathematical thought.

[2] The set of all partitions forms a lattice of which these are the top and bottom elements.

[3] Usually threshold graphs are presented in the setting of dissimilarity spaces, using the edge defining inequality $s(u, v) \leq s_i$.

Miscellanea
Numerical experiments are described in this thesis, which means that the realm of finite precision arithmetic is entered. Numerical expressions denote floating point numbers if and only if a dot is part of the expression. Expressions in which single indices or subscripted or superscripted simple expressions are enclosed in parentheses denote the object which results from letting the index run over its natural boundaries. E.g. $e_{(i)}$ denotes a vector or a row (the context should leave no doubt which of the two), $T_{k(i)}$ denotes the k^{th} row of the matrix T, and $(T^{(i)})_{kl}$ denotes the set of kl entries of the powers of T. The fact that each of the entries in a row $e_{(i)}$ equals the same constant c is concisely written as $e_{(i)} \stackrel{c}{=} c$.

5. The graph clustering paradigm

What are natural groups? This is in general a difficult problem, but within the framework of graphs there is a single notion which governs many proposals. This notion can be worded in different ways. Let G be a graph possessing cluster structure, then alternative wordings are the following:

a) The number of higher–length paths in G is large for pairs of vertices lying in the same dense cluster, and small for pairs of vertices belonging to different clusters.

b) A random walk in G that visits a dense cluster will likely not leave the cluster until many of its vertices have been visited.

c) Considering all shortest paths between all pairs of vertices of G, links between different dense clusters are likely to be in many shortest paths.

These three notions are strongly related to each other. The situation can be compared to driving a car in an unfamiliar city in which different districts are connected by only a few roads, with many promising looking turns and roads unreachable due to traffic regulations. Viewing crossings and turns as vertices, and the accessible road segments between them as edges, the notions given above translate to a) There are many different ways of driving (not necessarily taking the shortest route) from A to B if A and B are in the same district, and only few if they are in different districts, under the condition that the number of roads segments visited is equal; b) Driving around randomly, but in line with traffic regulations, will keep you in the same district for a long time; c) If the transportation need of the locals is homogeneously distributed over all departure and destination points, then the roads connecting different districts will be congested.

The idea now is to measure or sample any of these — higher–length paths, random walks, shortest paths — and deduce the cluster structure from the behaviour of the sampled quantities. The cluster structure will manifest itself as a peaked distribution of the quantities, and conversely, a lack of cluster structure will result in a flat distribution. The distribution should be easy to compute, and a peaked distribution should have a straightforward interpretation as a clustering.

I propose to assemble the notions listed above under the denominator of the *graph clustering paradigm*, being well aware of the fact that the paradigm label is somewhat grandiloquent. However, the notions clearly share a common idea that is simple and elegant in that it gives an abstract and implicit description of cluster structure (rather than tying it to a particular optimization criterion); in that it is persistent, as it has surfaced at different times and places[4]; and in that it is powerful, as it can be tied to different graph–theoretical concepts, yielding different clustering methods.

The idea of using random walks to derive cluster structure is mainly found within the graph partitioning community. The various proposals utilizing it are discussed in Section 8. The following section describes

[4]The number of occurrences is not large in itself, but it is significant considering the small number of publications dedicated to graph clustering.

proposals for graph clustering which have a strong combinatorial nature. One of these, the linkage–based k-path clustering method forms the connection between combinatorial and randomized methods. The single linkage paradigm can be seen as the connecting factor. This requires the dismissal of a notion which is seemingly central to single link clustering, namely the global interpretation of the (dis)similarity function. It is argued that this global interpretation hampers the combinatorial clustering methods introduced below; the introduction of random walks naturally requires a localized interpretation of graph connectivity properties.

6. Combinatorial cluster notions

In the clustering and pattern recognition communities, proposals have been made to define clusters in graphs which are more combinatorial in nature. An important contributor in this respect is David Matula, who wrote several articles on the subject. It is noteworthy that Matula's publication record (e.g. [30, 31, 32, 36]) indicates that his primary research interests are in graph theory and discrete mathematics. It seems that his publications in clustering in the setting of (simple) graphs came too early in the sense that at the time of writing there was little interest in the clustering community in simple graphs, except as a means of notation for the description of linkage–based algorithms such as single link and complete link clustering. In fact, Matula presents several graph cluster concepts in [31] as a series of refinements splitting the spectrum between single link and complete link clustering. The presentation of these findings in the setting of general similarity spaces and threshold graphs indicates that the time was not right for clustering in the setting of simple graphs per se. I see several reasons why the combinatorial notions have not caught on, among which the issue of justification in the setting of threshold graphs and the lack of genuine (simple) graph applications and problems. Equally important however are the relative intractability of the proposed notions, and their disposition to produce unbalanced clusterings. Let $G = (V, E)$ be a graph. The following notions each define subgraphs of G.

k-bond	A maximal subgraph S such that each node in S has at least degree k in S.
k-component	A maximal subgraph S such that each pair of nodes in S is joined by k edge–disjoint paths in S.
k-block	A maximal subgraph S such that each pair of nodes in S is joined by k vertex–disjoint (except for endpoints) paths in S.

Each notion defines a corresponding hierarchical cluster method by letting k vary and at each level taking as cluster elements all k-objects and all singletons corresponding with nodes which are not in any k-object, where object may be any of *bond, component*, or *block*. These methods are hierarchical because every $k+1$-object is contained within a k-object. For $k = 1$ all three k-notions boil down to the connected components of G. Moreover, for fixed k, it is true that every k-block of G is a subgraph of some k-component, which is in turn a subgraph of some k-bond of G. This implies that the corresponding cluster methods are successive refinements, going from bond to component to block. In the clustering section of the graph partitioning survey article [3] of Alpert and Kahng one method is mentioned which is a refinement of the k-component method, namely the (K, L)–connectivity method proposed by Garbers et al in [14]. Nodes are (K, L)–connected if there exist K edge disjoint paths of length at most L between them.

Matula finds that k-components and k-blocks provide better resolution into cohesive groupings than k-bonds. The example given here in Figure 1 is taken from the article [31], and it shows a graph with its k-blocks, yielding the most refined clusterings. In this case, the overlapping clustering for $k = 3$ looks reasonably good, although it is a pity that the fifth point in the leftmost 2-block ends up as a singleton in the 3-block clustering.

The lack of balance is even stronger in the graph which is depicted in Figure 2, together with its 3-block clustering. For this graph, the 2-block clustering yields the whole vertex set as a single cluster and the 3-block clustering is very unsatisfactory. This evidence is neither incidental nor contrived. Rather, it is inherent to the k-object methods. They are very sensitive to local variations in node degree. Such sensitivity is unwanted in itself, and in this case leads to unbalanced clusterings. The k-object methods are much too restrictive in

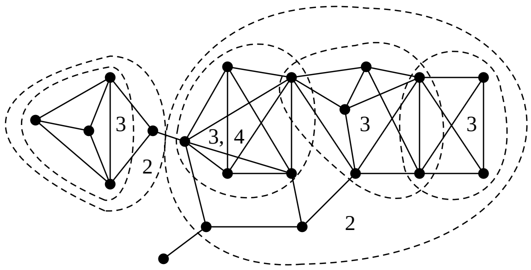

Figure 1: Graph with its k-blocks.

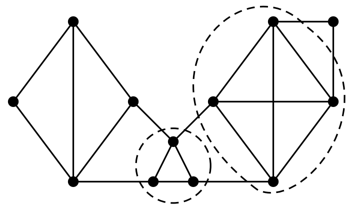

Figure 2: Graph with its 3-blocks.

their definition of cohesive structure, especially taking into account the commonly accepted 'loose' objective of clustering. It is reasonable to demand that a clustering method for simple graphs can recognize disjoint unions of complete (simple) graphs of different sizes, or complete graphs which are sparsely connected. The k-object methods clearly fail to do so, and one reason for this is that local variations in connectivity have severe impact on the retrieved clusters.

Finally, the object methods are highly intractable. Matula [31] and Tomi [40] give time complexities $\mathcal{O}(|V|^{3/2}|E|^2)$ for the retrieval of k-blocks and $\mathcal{O}(\min(|V|^{8/3}|E|, |V||E|^2)$ for the retrieval of k-components. Among others, the algorithms require the solution of the minimum cut network flow problem. Since the number of edges $|E|$ is surely at least $|V|$ for interesting applications, the time complexities are at least cubic in the input size of the graph.

7. k-Path clustering

Of the existing procedural algorithms, single link clustering has the most appealing mathematical properties. This is precisely because it allows *non-procedural* interpretations in terms of Minimal Spanning Trees and in terms of approximating metrics by ultrametrics (trees). See [17] for an extensive treatment of this subject. In this section I shall discuss a variant of single link clustering for graphs which I call k-path clustering. This variant is a further relaxation of the k-block and k-component methods, and its interpretation is related to the interpretation of the MCL algorithm. The basic observation underlying both methods is the fact that two nodes in some dense region will be connected by many more paths of length $k, k > 1$, than two nodes for which there is no such region. This section is mainly an exposition of ideas, and a few examples are studied. The examples are intended to support the heuristic underlying the MCL algorithm, and they provide fruitful insights into the problems and benefits associated with refinements of graph similarities. k-Path clustering is conceptually much simpler than k-block and k-component clustering, but in terms of tractability it is only slightly more viable. It suffers less from a lack of balance in the clusters it produces, but it is still far from satisfactory in this respect. k-Block, k-component, and k-path clustering were also proposed by Tamura [39], who was apparently unaware of the work of Matula.

For $k = 1$, the k-path clustering method coincides with generic single link clustering. For $k > 1$ the method is a straightforward generalization which refines the similarity coefficient associated with 1-path clustering. Let $G = (V, w)$ be a graph, where $V = \{v_1, \ldots, v_t\}$, let $M = \mathcal{M}_G$ be the associated matrix of G. For each integer $k > 0$, a similarity coefficient $Z_{k,G}$ associated with G on the set V is defined by setting $Z_{k,G}(v_i, v_j) = \infty, i = j$, and

$$Z_{k,G}(v_i, v_j) = (M^k)_{ij}, \qquad i \neq j \tag{7.1}$$

Note that the values $(M^i)_{pp}$ are disregarded. The quantity $(M^k)_{pq}$ has a straightforward interpretation as the number of paths of length k between v_p and v_q; this is the exact situation if G is a simple graph. If G has dense regions separated by sparse boundaries, it is reasonable to conjecture that there will be relatively many path connections of length k with both ends in the same region, compared with the number of path connections having both ends in different dense regions. For weighted graphs, the interpretation is in terms of path capacities rather than paths per se, and the formulation is now that the path capacities between different dense regions are small compared with the path capacities within a single dense region. The next example is one in which $Z_{k,G}$ does not yet work as hoped for. It will be seen why and how that can be remedied. For sake of clear exposition, the examples studied are simple graphs.

Odd and even

The graph G_1 in Figure 3 is a tetraeder with flattened tips. It clearly admits one good non-extreme clustering, namely the one in which each of the flattened tips, i.e. the four triangles, forms a cluster. The associated matrix $M = \mathcal{M}_{G_1}$, and the square M^2 are shown in Figure 5.

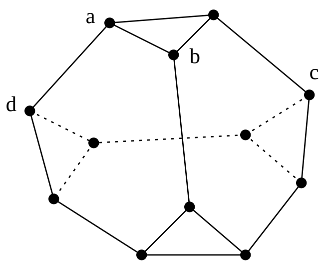

Figure 3: Topped tetraeder G_1.

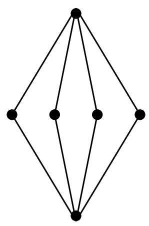

Figure 4: Bipartite graph G_2.

For each of the coefficients Z_{k,G_1}, single link clustering immediately yields the whole vertex set of G_1 as one cluster. How can this be? Somehow, the expectation that there would be relatively more k-length paths within the dense regions, in this case triangles, was unjustified. Now, on the one hand this is a peculiarity of this particular graph and especially of the subgraphs of the triangle type. For even k, spoilers are pairs like (a, c), for odd k, these are pairs like (a, d). This clearly has to do with the specific structure of G_1, where the set of paths of odd length leading e.g. from a to b does not profit from (a, b) being in a triangle, compared with the set of paths leading from a to d. On the other hand the behaviour of any similarity coefficient $Z_{k,G}$ is in general very much influenced by the parity of k. There is a strong effect that odd powers of M obtain their mass from simple paths of odd length and that even powers of M obtain their mass from simple paths of even length. The only exceptions are those paths which include loops of odd length. Note that the only requirement for a loop of even length is the presence of an edge (inducing a loop of length 2).

A countermeasure to parity dependence

The observation in one of the previous paragraphs that paths containing circuits of odd length form an exception brings a solution to the problem of parity dependence. By adding loops to each node in G_1, the parity dependence is removed. Just as every edge induces the minimal loop of even length, every node now induces the minimal loop of odd length. On the algebra side, adding loops corresponds with adding the identity matrix to M. The numbers defining the new coefficients Z_{2,G_1+I} are found in Figure 5, where the largest off-diagonal matrix entries (diagonal entries are disregarded) are printed in boldface. Each coefficient now

$$\begin{pmatrix}0&1&1&0&0&0&0&0&0&0&0&1\\1&0&1&0&0&0&0&1&0&0&0&0\\1&1&0&1&0&0&0&0&0&0&0&0\\0&0&1&0&1&1&0&0&0&0&0&0\\0&0&0&1&0&1&0&0&0&0&1&0\\0&0&0&1&1&0&1&0&0&0&0&0\\0&0&0&0&0&1&0&1&1&0&0&0\\0&1&0&0&0&0&1&0&1&0&0&0\\0&0&0&0&0&0&1&1&0&1&0&0\\0&0&0&0&0&0&0&0&1&0&1&1\\0&0&0&0&1&0&0&0&0&1&0&1\\1&0&0&0&0&0&0&0&0&1&1&0\end{pmatrix}$$
$$M = \mathcal{M}_{G_1}$$

$$\begin{pmatrix}3&1&1&1&0&0&0&1&0&1&1&0\\1&3&1&1&0&0&1&0&1&0&0&1\\1&1&3&0&1&1&0&1&0&0&0&1\\1&1&0&3&1&1&1&0&0&0&1&0\\0&0&1&1&3&1&1&0&0&1&0&1\\0&0&1&1&1&3&0&1&1&0&1&0\\0&1&0&1&1&0&3&1&1&1&0&0\\1&0&1&0&0&1&1&3&1&1&0&0\\0&1&0&0&0&1&1&1&3&0&1&1\\1&0&0&0&1&0&1&1&0&3&1&1\\1&0&0&1&0&1&0&0&1&1&3&1\\0&1&1&0&1&0&0&0&1&1&1&3\end{pmatrix}$$
$$M^2$$

$$\begin{pmatrix}1&1&1&0&0&0&0&0&0&0&0&1\\1&1&1&0&0&0&0&1&0&0&0&0\\1&1&1&1&0&0&0&0&0&0&0&0\\0&0&1&1&1&1&0&0&0&0&0&0\\0&0&0&1&1&1&0&0&0&0&1&0\\0&0&0&1&1&1&1&0&0&0&0&0\\0&0&0&0&0&1&1&1&1&0&0&0\\0&1&0&0&0&0&1&1&1&0&0&0\\0&0&0&0&0&0&1&1&1&1&0&0\\0&0&0&0&0&0&0&0&1&1&1&1\\0&0&0&0&1&0&0&0&0&1&1&1\\1&0&0&0&0&0&0&0&0&1&1&1\end{pmatrix}$$
$$M + I$$

$$\begin{pmatrix}4&\mathbf{3}&\mathbf{3}&1&0&0&0&1&0&1&1&2\\\mathbf{3}&4&\mathbf{3}&1&0&0&1&2&1&0&0&1\\\mathbf{3}&\mathbf{3}&4&2&1&1&0&1&0&0&0&1\\1&1&2&4&\mathbf{3}&\mathbf{3}&1&0&0&0&1&0\\0&0&1&\mathbf{3}&4&\mathbf{3}&1&0&0&1&2&1\\0&0&1&\mathbf{3}&\mathbf{3}&4&2&1&1&0&1&0\\0&1&0&1&1&2&4&\mathbf{3}&\mathbf{3}&1&0&0\\1&2&1&0&0&1&\mathbf{3}&4&\mathbf{3}&1&0&0\\0&1&0&0&0&1&\mathbf{3}&\mathbf{3}&4&2&1&1\\1&0&0&0&1&0&1&1&2&4&\mathbf{3}&\mathbf{3}\\1&0&0&1&2&1&0&0&1&\mathbf{3}&4&\mathbf{3}\\2&1&1&0&1&0&0&0&1&\mathbf{3}&\mathbf{3}&4\end{pmatrix}$$
$$(M + I)^2$$

Figure 5: Several matrices associated with G_1.

yields the best clustering, consisting of the set of four triangles. Adding loops helps in further differentiating the numbers $Z_{k,G_1+I}(s,t)$ for fixed s and varying t.

For a less symmetrical example, consider the simple graph G_3 depicted in Figure 6, also used on page 9. Its associated matrix after adding loops to each node is given next to it in Figure 7. Below are the results of single link clustering at all levels, using the similarity coefficient Z_{2,G_3+I}.

Level	Clustering
$\infty \ldots 6$	$\{\text{singletons}(V)\} = \{\ \{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\},\{9\},\{10\},\{11\},\{12\}\ \}$
5	$\{\ \{9,11\},\ \{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\},\{10\},\{12\}\ \}$
4	$\{\ \{1,10\},\ \{4,8,9,11\},\ \{2\},\{3\},\{5\},\{6\},\{7\},\{12\}\ \}$
3	$\{\{1,6,7,10\}\ \{2,3,5\},\ \{4,8,9,11,12\}\ \}$
$2,1,0$	$\{V\} = \{\{1,2,3,4,5,6,7,8,9,10,11,12\}\}$

The clustering at level 3, which is the first in which no singletons remain, is rather pleasing. This clustering also results if the coefficient is taken to be Z_{3,G_3+I} (not given here). The coefficient Z_{4,G_3+I} starts out accordingly, however, before node 6 gets involved, the groups $\{4,8,9,11,12\}$ and $\{2,3,5\}$ are joined. This is caused by the fact that node 6 is located in the sparsest part of G_3. The weak spot of single link clustering, namely *chaining*, surfaces here in the specific case of k-path clustering.

The last example in this section is a graph G_2 for which single link clustering with coefficient $Z_{k,G_2}, k > 1$, initially groups points together which are not connected. The graph G_2 in Figure 4 is a small bipartite graph. The upper and lower nodes have three simple paths of length 2 connecting them. Even in the presence of loops, the number of k-step paths, $k > 1$, will always be greater for the pair of top and bottom nodes than for any other pair. Bipartite graphs form a class of graphs for which it is natural to cluster each of the two node domains separately[5]. By adding multiple loops to each node of G_2 it can be ensured that the resulting

[5] e.g. Document phrase databases naturally yield bipartite graphs. Clustering the two node domains then yields a document grouping and a phrase grouping.

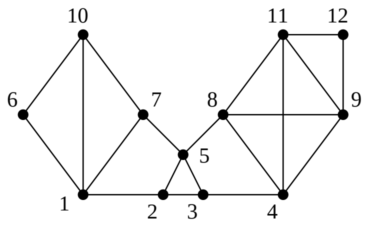

Figure 6: Graph G_3.

$$\begin{pmatrix}
5 & 2 & 1 & 0 & 2 & \mathbf{3} & \mathbf{3} & 0 & 0 & \mathbf{4} & 0 & 0 \\
2 & 4 & \mathbf{3} & 1 & \mathbf{3} & 1 & 2 & 1 & 0 & 1 & 0 & 0 \\
1 & \mathbf{3} & 4 & 2 & \mathbf{3} & 0 & 1 & 2 & 1 & 0 & 1 & 0 \\
0 & 1 & 2 & 5 & 2 & 0 & 0 & \mathbf{4} & \mathbf{4} & 0 & \mathbf{4} & 2 \\
2 & \mathbf{3} & \mathbf{3} & 2 & 5 & 0 & 2 & 2 & 1 & 1 & 1 & 0 \\
\mathbf{3} & 1 & 0 & 0 & 0 & 3 & 2 & 0 & 0 & \mathbf{3} & 0 & 0 \\
\mathbf{3} & 2 & 1 & 0 & 2 & 2 & 4 & 1 & 0 & \mathbf{3} & 0 & 0 \\
0 & 1 & 2 & \mathbf{4} & 2 & 0 & 1 & 5 & \mathbf{4} & 0 & \mathbf{4} & 2 \\
0 & 0 & 1 & \mathbf{4} & 1 & 0 & 0 & \mathbf{4} & 5 & 0 & \mathbf{5} & \mathbf{3} \\
\mathbf{4} & 1 & 0 & 0 & 1 & \mathbf{3} & \mathbf{3} & 0 & 0 & 4 & 0 & 0 \\
0 & 0 & 1 & \mathbf{4} & 1 & 0 & 0 & \mathbf{4} & \mathbf{5} & 0 & 5 & \mathbf{3} \\
0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & \mathbf{3} & 0 & \mathbf{3} & 3
\end{pmatrix}$$

Figure 7: The matrix $(N+I)^2, \quad N = \mathcal{M}_{G_3}$.

clustering corresponds with connected components only (one in this case), but it is difficult to formulate a sufficient condition which guarantees this property for graphs in general. I conjecture that a sufficient condition is for a graph to have nonnegative spectrum. This is a non-trivial conjecture, since spectral properties have to be related to both the ordinal relationship among entries of a matrix power and the 0/1 structure of the matrix itself.

A critical look at k-path clustering

If k-path clustering were to be applied to large graphs, it would be desirable to work with varying k and the corresponding coefficients $Z_{k,G}$. However, for most application graphs in this research, the matrices M^k and $(M+I)^k$ fill very rapidly due to high connectivity of the graphs. The potential number of nonzero elements equals 10^{2N} for graphs of vertex-size $|V| = 10^N$. For $N = 4$ this quantity is already huge and for $N = 5$ it is clearly beyond current possibilities. More importantly, it is quadratically related to N. In large scale applications, this is known to be a bad thing. It is difficult to remedy this situation by a regime of removing smaller elements.

A second minus was mentioned in the discussion of the example graph G_3 in Figure 6. I remarked that under the coefficient Z_{4,G_3+I} groups which had formed already started to unite before the last node left its singleton state. The coefficients $Z_{k,G}$ do account for the local structure around a node. However, a region which is denser than another region with which it connected to a certain extent, will tend to swallow the latter up. This is the effect of chaining in k-path clustering. A third minus is related to the preceding and arises in the case of weighted graphs. Differentiation in the weight function will lead to the same phenomenon of heavy-weight regions swallowing up light-weight regions. It should be noted that this situation is problematic for every cluster method based on single link clustering.

On the credit side I find that at least in a number of examples the idea of considering higher length paths works well. The manoeuvre of adding loops to graphs is clearly beneficial, and the reason for this lies in the fact that parity dependence is removed, leading to a further differentiation of the associated similarity coefficient. The issue of parity dependence has been noted before: Alpert and Kahng criticize the (K, L)–connectivity method of Garbers et al — which is a variant of k-component clustering — for cutting a four–cycle (which is a bipartite graph) into disjoint paths.

8. Random walks and graphs

In this section I briefly discuss probabilistic cluster algorithms proposed in the graph partitioning community and the concept of random walks on graphs. In the graph partitioning community, several randomized cluster algorithms have been proposed. I follow the survey article [3] by Alpert and Kahng which was written in 1995. Karger [24] proposed a heuristic where each vertex starts as a singleton cluster. Edges are iteratively chosen in random fashion, and each time the clusters incident to the currently chosen edge are contracted into a single cluster. A related approach was proposed by Bui et al in [6, 7]. A matching in a graph is a set of edges such that no pair of edges has a common vertex. They propose to find a random maximal matching and merge each pair of vertices into a cluster, resulting in a set of $n/2$ clusters. Both proposals hinge on the fact that there are more edges within clusters than in between different clusters if cluster structure is present. Hagen and Kahng sample random walks for cycles in [15]; the basic setup is that if two nodes co-occur sufficiently often in a cycle, then they are joined within a cluster. Finally, Yeh et al [43] propose a method in which shortest paths between randomly chosen pairs of vertices are computed. Each edge has a cost associated with it, which is adjusted every time the edge is included in a shortest path. In dense clusters, alternative paths are easily found; this not being the case for vertices in different clusters, edges between them will inevitably acquire a higher check.

The basic idea underlying the *MCL* algorithm fits in the same paradigm, but two important distinctions are that random walks are computed *deterministically* and *simultaneously*. The crux of the algorithm is that it incorporates reinforcement of random walks.

Random walks on graphs

The standard way to define a random walk on a simple graph is to let a Young Walker take off on some arbitrary vertex. After that, he successively visits new vertices by selecting arbitrarily one of the outgoing edges.[6] This will be the starting point for the *MCL* algorithm. An excellent survey on random graphs is [26] by Lovász. An important observation quoted from this article is the following:

> A random walk is a finite Markov chain that is time–reversible (see below). In fact, there is not much difference between the theory of random walks on graphs and the theory of finite Markov chains; every Markov chain can be viewed as a random walk on a directed graph, if we allow weighted edges.

The condition that (the chain generated by) a Markov matrix is time–reversible translates to the condition that the matrix is diagonally similar to a symmetric matrix (see below). In order to define random walks on weighted graphs in general, the weight function of a graph has to be changed such that the sum of the weight of all outgoing edges equals one. This is achieved by a generic rescaling step, which amounts to the localization of the weight function alluded to before.

Definition 2 *Let G be a graph on n nodes, let $M = \mathcal{M}_G$ be its associated matrix. The* **Markov matrix** *associated with a graph G is denoted by $\mathcal{T}_G$ and is formally defined by letting its q^{th} column be the q^{th} column of M normalized. To this end, let d denote the diagonal matrix that has diagonal entries the column weights of M, thus $d_{kk} = \sum_i M_{ik}$, and $d_{ij} = 0, i \neq j$. Then $\mathcal{T}_G$ is defined as*

$$\mathcal{T}_G \;=\; \mathcal{M}_G d^{-1} \tag{8.1}$$

The Markov matrix $\mathcal{T}_G$ corresponds with a graph G', which is called the **associated Markov graph** *of G. The directed weight function of G', which is encoded in the matrix $\mathcal{T}_G$, is called the* **localized interpretation** *of the weight function of G.* □

This definition encodes exactly the transformation step used in the theory of random walks on graphs. Given an undirected graph G, the matrix $N = \mathcal{T}_G$ is no longer symmetric, but is diagonally similar to a symmetric matrix. Something can be said about the spectrum of $\mathcal{T}_G$ in terms of the spectrum of $\mathcal{M}_G$ if G is undirected.

[6]Basic notions investigated in the theory of random walks are the *access time* H_{ij}, which is the expected number of steps before node i is visited starting from node j, the *cover time*, which is the expected number of steps to reach every node, and the *mixing rate*, which is a measure of how fast the random walk converges to its limiting distribution.

Lemma 1 *Let G be undirected and void-free*[7]*, let $M = \mathcal{M}_G$ be its associated matrix, let $T = \mathcal{T}_G$ be its associated Markov matrix. Then* the number of positive, negative, and zero eigenvalues are the same for T and M.

Next denote by l and u the minimum respectively maximum column sum, that is, $l = \min_k \sum_i M_{ik}$, and $u = \max_k \sum_i M_{ik}$. Then

$$\frac{\lambda_k(M)}{u} \le \lambda_k(T) \le \frac{\lambda_k(M)}{l} \qquad \lambda_k(T) > 0 \tag{8.2}$$

$$\frac{\lambda_k(M)}{l} \le \lambda_k(T) \le \frac{\lambda_k(M)}{u} \qquad \lambda_k(T) < 0 \tag{8.3}$$

Proof. Let d be the diagonal matrix of column lengths as defined in Definition 2. The matrix $T = Md^{-1}$ is similar to the matrix $d^{-1/2}Md^{-1/2}$, which is congruent to the matrix M. Now the first statement of the lemma follows from Sylvester's law of inertia ([18], page 223). Because of congruence, the inertia of the matrices M and $d^{-1/2}Md^{-1/2}$ are the same, and because of similarity, the spectra of the matrices $d^{-1/2}Md^{-1/2}$ and $T = Md^{-1}$ are the same, which is a stronger property than sharing the same inertia. The fact that the transition matrix $T = d^{-1}$ is diagonally similar to the symmetric matrix $d^{-1/2}Md^{-1/2}$ is in Markov theory phrased as that T is *time–reversible* or that T satisfies the *detailed balance condition.*

The second statement follows from Ostrowski's theorem ([18], page 224), which relates the eigenvalues of a hermitian matrix A to the eigenvalues of the matrix SAS^* in terms of bounding factors $\lambda_1(SS^*)$ and $\lambda_n(SS^*)$. In the lemma, these factors are simply the largest and smallest eigenvalue of the matrix d^{-1}, equalling respectively $1/l$ and $1/u$. It should be noted that this result can be refined by looking at principal submatrices of M. This is useful if there are a few columns of M of small weight compared with the other columns. This refinement is omitted here since it will not be needed. □

A closer look at random walks

Given a graph G and its associated Markov matrix $T = \mathcal{T}_G$, the value T_{pq} now indicates 'how much is the vertex q attracted to the vertex p', and this is meaningful only in the context of the other values found in the q^{th} column. It is still possible to move a node away from *all* its neighbours by increasing the weight of its loop. In Figure 8 the matrix $M = \mathcal{T}_{G_3+I}$ (corresponding with the graph G_3 in Figure 6) is given which results after the rescaling procedure, followed by three successive powers and a matrix labelled M^∞. The matrix M is column stochastic. The fact that for each of its columns all nonzero values are homogeneously distributed can be interpreted as 'each node is equally attracted to all of its neighbours', or 'at each node one moves to each of its neighbours with equal probability'.

All powers of M are column stochastic matrices too. For any Markov matrix N, the powers $N^{(i)}$ have a limit, which is possibly cyclic (i.e. consisting of a sequence of matrices rather than a single matrix). A connected component C of a graph G, which has the property that the greatest common divisor of the set of lengths of all circuits in C is 1, is called *regular.* If for every vertex in C there is a path in C leading to any other vertex in C it is called *ergodic.* If the underlying graph of a Markov matrix N consists of ergodic regular components only, then the limit of the row $N^{(i)}$ is non-cyclic. The graph G_3 in Figure 6 clearly has this property, and the limit is found in Figure 8, denoted as M^∞. The columns of M^∞ each equal the unique eigenvector of M associated with eigenvalue 1. This eigenvector e denotes the equilibrium state of the Markov process associated with M. A good review of Markov theory in the larger setting of nonnegative matrices can be found in [5]. Regrettably, the existing theory on Markov matrices is of little use in this thesis, because an essential ingredient of the *MCL* process is the operator Γ_r which acts on Markov matrices in a non-linear fashion.

[7] All vertices are part of at least one edge.

$$\begin{pmatrix}
0.200 & 0.250 & -- & -- & -- & 0.333 & 0.250 & -- & -- & 0.250 & -- & -- \\
0.200 & 0.250 & 0.250 & -- & 0.200 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.250 & 0.250 & 0.200 & 0.200 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & 0.250 & 0.200 & -- & -- & -- & 0.200 & 0.200 & -- & 0.200 & -- \\
-- & 0.250 & 0.250 & -- & 0.200 & -- & 0.250 & 0.200 & -- & -- & -- & -- \\
0.200 & -- & -- & -- & -- & 0.333 & -- & -- & -- & 0.250 & -- & -- \\
0.200 & -- & -- & -- & 0.200 & -- & 0.250 & -- & -- & 0.250 & -- & -- \\
-- & -- & -- & 0.200 & 0.200 & -- & -- & 0.200 & 0.200 & -- & 0.200 & -- \\
-- & -- & -- & 0.200 & -- & -- & -- & 0.200 & 0.200 & -- & 0.200 & 0.333 \\
0.200 & -- & -- & -- & -- & 0.333 & 0.250 & -- & -- & 0.250 & -- & -- \\
-- & -- & -- & 0.200 & -- & -- & -- & 0.200 & 0.200 & -- & 0.200 & 0.333 \\
-- & -- & -- & -- & -- & -- & -- & -- & 0.200 & -- & 0.200 & 0.333
\end{pmatrix}$$

$$M = \mathcal{T}_{G_3+I}$$

$$\begin{pmatrix}
0.257 & 0.113 & 0.063 & -- & 0.100 & 0.261 & 0.175 & -- & -- & 0.258 & -- & -- \\
0.090 & 0.225 & 0.175 & 0.050 & 0.140 & 0.067 & 0.100 & 0.040 & -- & 0.050 & -- & -- \\
0.050 & 0.175 & 0.225 & 0.090 & 0.140 & -- & 0.050 & 0.080 & 0.040 & -- & 0.040 & -- \\
-- & 0.063 & 0.113 & 0.210 & 0.090 & -- & -- & 0.160 & 0.160 & -- & 0.160 & 0.133 \\
0.100 & 0.175 & 0.175 & 0.090 & 0.230 & -- & 0.113 & 0.080 & 0.040 & 0.063 & 0.040 & -- \\
0.157 & 0.050 & -- & -- & -- & 0.261 & 0.113 & -- & -- & 0.195 & -- & -- \\
0.140 & 0.100 & 0.050 & -- & 0.090 & 0.150 & 0.225 & 0.040 & -- & 0.175 & -- & -- \\
-- & 0.050 & 0.100 & 0.160 & 0.080 & -- & 0.050 & 0.200 & 0.160 & -- & 0.160 & 0.133 \\
-- & -- & 0.050 & 0.160 & 0.040 & -- & -- & 0.160 & 0.227 & -- & 0.227 & 0.244 \\
0.207 & 0.050 & -- & -- & 0.050 & 0.261 & 0.175 & -- & -- & 0.258 & -- & -- \\
-- & -- & 0.050 & 0.160 & 0.040 & -- & -- & 0.160 & 0.227 & -- & 0.227 & 0.244 \\
-- & -- & -- & 0.080 & -- & -- & -- & 0.080 & 0.147 & -- & 0.147 & 0.244
\end{pmatrix}$$

$$M^2$$

$$\begin{pmatrix}
0.213 & 0.133 & 0.069 & 0.013 & 0.090 & 0.259 & 0.198 & 0.020 & -- & 0.238 & -- & -- \\
0.106 & 0.158 & 0.148 & 0.053 & 0.136 & 0.069 & 0.095 & 0.046 & 0.018 & 0.077 & 0.018 & -- \\
0.055 & 0.148 & 0.158 & 0.095 & 0.134 & 0.017 & 0.060 & 0.078 & 0.050 & 0.025 & 0.050 & 0.027 \\
0.013 & 0.066 & 0.119 & 0.161 & 0.085 & -- & 0.023 & 0.156 & 0.165 & -- & 0.165 & 0.151 \\
0.090 & 0.170 & 0.168 & 0.085 & 0.155 & 0.054 & 0.126 & 0.096 & 0.050 & 0.069 & 0.050 & 0.027 \\
0.155 & 0.052 & 0.013 & -- & 0.033 & 0.205 & 0.116 & -- & -- & 0.182 & -- & -- \\
0.158 & 0.095 & 0.060 & 0.018 & 0.101 & 0.155 & 0.158 & 0.026 & 0.008 & 0.173 & 0.008 & -- \\
0.020 & 0.058 & 0.098 & 0.156 & 0.096 & -- & 0.033 & 0.152 & 0.163 & 0.013 & 0.163 & 0.151 \\
-- & 0.023 & 0.063 & 0.165 & 0.050 & -- & 0.010 & 0.163 & 0.204 & -- & 0.204 & 0.233 \\
0.190 & 0.077 & 0.025 & -- & 0.055 & 0.242 & 0.173 & 0.010 & -- & 0.225 & -- & -- \\
-- & 0.023 & 0.063 & 0.165 & 0.050 & -- & 0.010 & 0.163 & 0.204 & -- & 0.204 & 0.233 \\
-- & -- & 0.020 & 0.091 & 0.016 & -- & -- & 0.091 & 0.140 & -- & 0.140 & 0.179
\end{pmatrix}$$

$$M^3$$

$$\begin{pmatrix}
0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 \\
0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 \\
0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 \\
0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 \\
0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 \\
0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 \\
0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 \\
0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 \\
0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 \\
0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 & 0.077 \\
0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 & 0.096 \\
0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058 & 0.058
\end{pmatrix}$$

$$M^\infty$$

Figure 8: Powers of $M = \mathcal{T}_{G_3+I}$, the Markov matrix associated with the graph G_3 in Figure 6, loops added to G_3

Consider Figure 8 again. As is to be expected, the equilibrium state e (each column of M^{∞} equals e) spreads its mass rather homogeneously among the states or vertices of G_3. However, the initial iterands $M^k, k = 2, \ldots,$ exhibit the same behaviour as did the matrices $(N+I)^k$ in Figure 7, inducing the similarity coefficients $Z_{k,G+I}$. Transition values $M^k{}_{pq}$ are relatively high if the vertices p and q are located in the same dense region. There is a correspondence between the numerical distribution of the column $M^k{}_{p(q)}$, and the distribution of the edges of G_3 over dense regions and sparse boundaries.

Boosting the multiplier effect
The obvious interpretation of the new weight function is in terms of flow or random walks rather than in terms of path sets, but the observed behaviour of matrix multiplication is similar. The new interpretation of the weight function more or less suggests a speculative move. Flow is easier within dense regions than across sparse boundaries, however, in the long run this effect disappears. What if the initial effect is deliberately boosted by adjusting the transition probabilities? A logical model is to transform a Markov matrix T by transforming each of its columns. For each vertex, the distribution of its preferences (i.e. transition values) will be changed such that prefered neighbours are further favoured and less popular neighbours are demoted. A natural way to achieve this effect is to raise all the entries in a given column to a certain power greater than one (e.g. squaring), and rescaling the column to have sum 1 again. This has the advantage that vectors for which the nonzero entries are nearly homogeneously distributed are not so much changed, and that different column positions with nearly identical values will still be close to each other after rescaling. This is explained by observing that what effectively happens is that all *ratios* $T_{p_1 q}/T_{p_2 q}$ are raised to the same power. Below four vectors and their image after rescaling with power coefficient 2 are listed. The notation $\Gamma_r v$ is introduced right after these examples.

$$\text{Vector } v: \quad \begin{pmatrix}0\\3\\0\\1\\2\end{pmatrix} \quad \begin{pmatrix}0\\1/2\\0\\1/6\\1/3\end{pmatrix} \quad \begin{pmatrix}1/4\\1/4\\1/4\\1/4\\0\end{pmatrix} \quad \begin{pmatrix}0.151\\0.159\\0.218\\0.225\\0.247\end{pmatrix} \quad \begin{pmatrix}0.086\\0.000\\0.113\\0.801\\0.000\end{pmatrix}$$

$$\text{Image } \Gamma_2 v: \quad \begin{pmatrix}0\\9/14\\0\\1/14\\4/14\end{pmatrix} \quad \begin{pmatrix}0\\9/14\\0\\1/14\\4/14\end{pmatrix} \quad \begin{pmatrix}1/4\\1/4\\1/4\\1/4\\0\end{pmatrix} \quad \begin{pmatrix}0.110\\0.122\\0.229\\0.245\\0.295\end{pmatrix} \quad \begin{pmatrix}0.011\\0.000\\0.019\\0.970\\0.000\end{pmatrix}$$

Definition 3 *Given a matrix $M \in I\!R^{k \times l}$, $M \geq 0$, and a real nonnegative number r, the matrix resulting from rescaling each of the columns of M with power coefficient r is called $\Gamma_r M$, and Γ_r is called the* **inflation** *operator with power coefficient r. Formally, the action of $\Gamma_r : I\!R^{k \times l} \to I\!R^{k \times l}$ is defined by*

$$(\Gamma_r M)_{pq} \quad = \quad (M_{pq})^r \Big/ \sum_{i=1}^{k} (M_{iq})^r$$

If the subscript is omitted, it is understood that the power coefficient equals 2. □

There are no restrictions on the matrix dimensions to fit a square matrix, because this allows Γ_r to act on both matrices and column vectors. There is no restriction that the input matrices be stochastic, since it is not strictly necessary, and the extended applicability is sometimes useful. The parameter r is assumed rather than required to be nonnegative. The reason is that in the setting of the *MCL* process nonnegative values r have a sensible interpretation attached to them. Values of r between 0 and 1 increase the homogeneity of the argument probability vector (matrix), whereas values of r between 1 and ∞ increase the inhomogeneity. In both cases, the ordering of the probabilities is not disturbed. Negative values of r invert the ordering, which does not seem to be of apparent use.

Definition 4 *A nonnegative vector v is called* **homogeneous** *if all its nonzero entries are equal. A nonnegative matrix is called* **column–homogeneous** *if each of its columns is homogeneous.* □

The set of homogeneous probability vectors is precisely the set of vectors which are invariant under $\Gamma_r, r \neq 1$. When applied to vectors, the Γ_r operator has a nice mathematical property in terms of *majorization*. This is discussed in the following section, Section 10.

Iterating expansion and inflation

Figure 9 gives the result of applying Γ_r to the Markov matrix M^2 given in Figure 8. The vital step now is to iterate the process of alternately expanding information flow via normal matrix multiplication and contracting information flow via application of Γ_r. Thus, the matrix $\Gamma_r M^2$ is squared, and the inflation operator is applied to the result. This process is repeated ad libitum. The invariant of the process is that flow in dense regions profits from both the expansion and the inflation step. A priori it is uncertain whether the process converges, or whether convergence will lead to a meaningful limit. However, the heuristic which leads to the formulation of the process suggests that something will happen for graphs possessing sparse boundaries. The transition values corresponding to edges crossing sparse boundaries are given a hard time by the process, and if anything, it is to be expected that they will tend to zero. This is exactly what happens for the example graph. The 5^{rd} iterand, the 9^{th} iterand, and the invariant limit[8] of this process (provisionally denoted by M^{∞}_{mcl}) are given in Figure 9 as well.

The matrix M^{∞}_{mcl} clearly is an idempotent under both matrix multiplication and the inflation operator. It has a straightforward interpretation as a clustering. Four nodes can be said to be an *attractor*, namely those nodes that have positive return probability. The nodes 9 and 11 are as much attracted to each other as they are to themselves. The rest of the vertex set of G_3 can be completely partitioned according to the nodes to which they are attracted. Sweeping attractors and the elements they attract together, the partition $\{4, 8, 9, 11, 12\}$ $\{1, 6, 7, 10\}$ $\{2, 3, 5\}$ results, also found earlier with k-path clustering.

A certain subset of the equilibrium states only admits an interpretation as a clustering with overlap. This is related to the presence of symmetry in the graphs and matrices used. Consider the matrix M depicted in Figure 10, corresponding with a line–graph on 7 nodes, loops added to each node. An *MCL* run with $e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2$ results in the limit T^{∞}_{mcl}. The nodes 2 and 6 are attractors, the node sets $\{1, 3\}$, and $\{5, 7\}$, are respectively attracted to them. The vertex 4 is equally attracted to 2 and 6. The formation of two clusters, or different regions of attraction, is explained by the fact that the nodes at the far ends, i.e. $1, 2, 6, 7$ have higher return probability after the first iterations than the nodes in the middle. Given the symmetry of the graph, it is only natural that node 4 is equally attracted to both regions.

Formal description of the MCL algorithm

The basic design of the *MCL* algorithm is given in Figure 11; it is extremely simple and provides basically an interface to the *MCL* process, introduced below. The main skeleton is formed by the alternation of matrix multiplication and inflation in a for loop. In the k^{th} iteration of this loop two matrices labelled T_{2k} and T_{2k+1} are computed. The matrix T_{2k} is computed as the previous matrix T_{2k-1} taken to the power e_k. The matrix T_{2k+1} is computed as the image of T_{2k} under Γ_{r_k}. The row[9] of expansion powers $e_{(i)}$ and the row of

[8] Idempotent under both Exp_2 and Γ_2.

[9] The notation $e_{(i)}$ is shorthand for $\{e_i\}_{i \in \mathbb{N}}$ and likewise $r_{(i)}$ for $\{r_i\}_{i \in \mathbb{N}}$.

$$\begin{pmatrix}
0.380 & 0.087 & 0.027 & -- & 0.077 & 0.295 & 0.201 & -- & -- & 0.320 & -- & -- \\
0.047 & 0.347 & 0.210 & 0.017 & 0.150 & 0.019 & 0.066 & 0.012 & -- & 0.012 & -- & -- \\
0.014 & 0.210 & 0.347 & 0.056 & 0.150 & -- & 0.016 & 0.046 & 0.009 & -- & 0.009 & -- \\
-- & 0.027 & 0.087 & 0.302 & 0.062 & -- & -- & 0.184 & 0.143 & -- & 0.143 & 0.083 \\
0.058 & 0.210 & 0.210 & 0.056 & 0.406 & -- & 0.083 & 0.046 & 0.009 & 0.019 & 0.009 & -- \\
0.142 & 0.017 & -- & -- & -- & 0.295 & 0.083 & -- & -- & 0.184 & -- & -- \\
0.113 & 0.069 & 0.017 & -- & 0.062 & 0.097 & 0.333 & 0.012 & -- & 0.147 & -- & -- \\
-- & 0.017 & 0.069 & 0.175 & 0.049 & -- & 0.016 & 0.287 & 0.143 & -- & 0.143 & 0.083 \\
-- & -- & 0.017 & 0.175 & 0.012 & -- & -- & 0.184 & 0.288 & -- & 0.288 & 0.278 \\
0.246 & 0.017 & -- & -- & 0.019 & 0.295 & 0.201 & -- & -- & 0.320 & -- & -- \\
-- & -- & 0.017 & 0.175 & 0.012 & -- & -- & 0.184 & 0.288 & -- & 0.288 & 0.278 \\
-- & -- & -- & 0.044 & -- & -- & -- & 0.046 & 0.120 & -- & 0.120 & 0.278
\end{pmatrix}$$

$\Gamma_2 M^2$, M defined in Figure 8

$$\begin{pmatrix}
0.448 & 0.080 & 0.023 & -- & 0.068 & 0.426 & 0.359 & -- & -- & 0.432 & -- & -- \\
0.018 & 0.285 & 0.228 & 0.007 & 0.176 & 0.006 & 0.033 & 0.005 & -- & 0.007 & -- & -- \\
0.005 & 0.223 & 0.290 & 0.022 & 0.173 & -- & 0.010 & 0.017 & 0.003 & 0.001 & 0.003 & 0.001 \\
-- & 0.018 & 0.059 & 0.222 & 0.040 & -- & 0.001 & 0.187 & 0.139 & -- & 0.139 & 0.099 \\
0.027 & 0.312 & 0.314 & 0.028 & 0.439 & 0.005 & 0.054 & 0.022 & 0.003 & 0.010 & 0.003 & 0.001 \\
0.116 & 0.007 & 0.001 & -- & 0.004 & 0.157 & 0.085 & -- & -- & 0.131 & -- & -- \\
0.096 & 0.040 & 0.013 & -- & 0.037 & 0.083 & 0.197 & 0.001 & -- & 0.104 & -- & -- \\
-- & 0.012 & 0.042 & 0.172 & 0.029 & -- & 0.002 & 0.198 & 0.133 & -- & 0.133 & 0.096 \\
-- & 0.001 & 0.015 & 0.256 & 0.009 & -- & -- & 0.266 & 0.326 & -- & 0.326 & 0.346 \\
0.290 & 0.021 & 0.002 & -- & 0.017 & 0.323 & 0.260 & -- & -- & 0.316 & -- & -- \\
-- & 0.001 & 0.015 & 0.256 & 0.009 & -- & -- & 0.266 & 0.326 & -- & 0.326 & 0.346 \\
-- & -- & 0.001 & 0.037 & 0.001 & -- & -- & 0.039 & 0.069 & -- & 0.069 & 0.112
\end{pmatrix}$$

$\Gamma_2(\Gamma_2 M^2 \cdot \Gamma_2 M^2)$

$$\begin{pmatrix}
0.807 & 0.040 & 0.015 & -- & 0.034 & 0.807 & 0.807 & -- & -- & 0.807 & -- & -- \\
-- & 0.090 & 0.092 & -- & 0.088 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.085 & 0.088 & -- & 0.084 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.001 & 0.001 & 0.032 & 0.001 & -- & -- & 0.032 & 0.031 & -- & 0.031 & 0.031 \\
-- & 0.777 & 0.798 & -- & 0.786 & -- & 0.001 & -- & -- & -- & -- & -- \\
0.005 & -- & -- & -- & -- & 0.005 & 0.005 & -- & -- & 0.005 & -- & -- \\
0.003 & 0.001 & -- & -- & 0.001 & 0.003 & 0.003 & -- & -- & 0.003 & -- & -- \\
-- & -- & 0.001 & 0.024 & -- & -- & -- & 0.024 & 0.024 & -- & 0.024 & 0.024 \\
-- & -- & 0.002 & 0.472 & 0.001 & -- & -- & 0.472 & 0.472 & -- & 0.472 & 0.472 \\
0.185 & 0.005 & 0.001 & -- & 0.004 & 0.185 & 0.184 & -- & -- & 0.185 & -- & -- \\
-- & -- & 0.002 & 0.472 & 0.001 & -- & -- & 0.472 & 0.472 & -- & 0.472 & 0.472 \\
-- & -- & -- & 0.001 & -- & -- & -- & 0.001 & 0.001 & -- & 0.001 & --
\end{pmatrix}$$

$(\Gamma_2 \circ Squaring)$ iterated four times on M

$$\begin{pmatrix}
1.000 & -- & -- & -- & -- & 1.000 & 1.000 & -- & -- & 1.000 & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & 1.000 & 1.000 & -- & 1.000 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & --
\end{pmatrix}$$

M^{∞}_{mcl}

Figure 9: Iteration of $(\Gamma_2 \circ Squaring)$ with initial iterand M defined in Figure 8.

Entries marked '−−' are either zero because that is the exact value they assume (this is true for the first two matrices) or because the computed value fell below the machine precision.

$$\begin{pmatrix} 0.5000 & 0.3333 & -- & -- & -- & -- & -- \\ 0.5000 & 0.3333 & 0.3333 & -- & -- & -- & -- \\ -- & 0.3333 & 0.3333 & 0.3333 & -- & -- & -- \\ -- & -- & 0.3333 & 0.3333 & 0.3333 & -- & -- \\ -- & -- & -- & 0.3333 & 0.3333 & 0.3333 & -- \\ -- & -- & -- & -- & 0.3333 & 0.3333 & 0.5000 \\ -- & -- & -- & -- & -- & 0.3333 & 0.5000 \end{pmatrix}$$

Initial iterand $T_1 = M$

$$\begin{pmatrix} 0.3221 & 0.2393 & 0.0493 & 0.0028 & 0.0000 & -- & -- \\ 0.6138 & 0.6120 & 0.2664 & 0.0420 & 0.0021 & 0.0000 & -- \\ 0.0606 & 0.1275 & 0.4259 & 0.2165 & 0.0383 & 0.0010 & 0.0000 \\ 0.0035 & 0.0200 & 0.2159 & 0.4662 & 0.2143 & 0.0200 & 0.0034 \\ 0.0000 & 0.0011 & 0.0403 & 0.2259 & 0.4311 & 0.1282 & 0.0607 \\ -- & 0.0000 & 0.0022 & 0.0436 & 0.2652 & 0.6116 & 0.6137 \\ -- & -- & 0.0000 & 0.0029 & 0.0490 & 0.2392 & 0.3220 \end{pmatrix}$$

Intermediate iterand T_5 (k equals 2)

$$\begin{pmatrix} 0.0284 & 0.0280 & 0.0191 & 0.0015 & 0.0000 & 0.0000 & 0.0000 \\ 0.9647 & 0.9631 & 0.8226 & 0.1205 & 0.0016 & 0.0000 & 0.0000 \\ 0.0066 & 0.0082 & 0.0768 & 0.1362 & 0.0087 & 0.0000 & 0.0000 \\ 0.0003 & 0.0006 & 0.0686 & 0.4309 & 0.0673 & 0.0006 & 0.0003 \\ 0.0000 & 0.0000 & 0.0109 & 0.1677 & 0.0863 & 0.0088 & 0.0069 \\ 0.0000 & 0.0000 & 0.0020 & 0.1414 & 0.8173 & 0.9627 & 0.9644 \\ 0.0000 & 0.0000 & 0.0000 & 0.0018 & 0.0187 & 0.0280 & 0.0284 \end{pmatrix}$$

Intermediate iterand T_9 (k equals 4)

$$\begin{pmatrix} -- & -- & -- & -- & -- & -- & -- \\ 1.0000 & 1.0000 & 1.0000 & 0.5000 & -- & -- & -- \\ -- & -- & -- & -- & -- & -- & -- \\ -- & -- & -- & -- & -- & -- & -- \\ -- & -- & -- & -- & -- & -- & -- \\ -- & -- & -- & 0.5000 & 1.0000 & 1.0000 & 1.0000 \\ -- & -- & -- & -- & -- & -- & -- \end{pmatrix}$$

Limit T_{mcl}^{∞} (idempotent under Exp_2 and Γ_2).

Figure 10: *MCL* run on a line–graph on 7 nodes

```
                                        # G is a voidfree graph.
                                        # e_i ∈ IN, e_i > 1, i = 1, . . . .
MCL (G, Δ, e_(i), r_(i)) {              # r_i ∈ IR, r_i > 0, i = 1, . . . .

        G = G + Δ;                      # Possibly add (weighted) loops.
        T_1 = T_G;                      # Create associated Markov graph
                                        # according to Definition 2.
        for k = 1, . . . , ∞ {
                T_2k = Exp_{e_k}(T_{2k−1});
                T_{2k+1} = Γ_{r_k}(T_2k);
                if (T_{2k+1} is (near–) idempotent)    break;
        }
        Interpret T_{2k+1} as clustering according to Definition 8;
}
```

Figure 11: The basic MCL algorithm. Convergence is discussed in Section 9.

inflation powers $r_{(i)}$ influence the granularity of the resulting partition. The matrices in Figure 9 correspond with an MCL session in which $e_{(i)} \stackrel{c}{=} 2$ and $r_{(i)} \stackrel{c}{=} 2$. If the current iterand is sufficiently close to an idempotent matrix the process stops and the last resultant is interpreted according to Definition 8 and Theorem 1 in the next section. The theorem provides a mapping from the set of nonnegative column allowable idempotent matrices to the set of overlapping clusterings. There are exceptional cases in which the iterands cycle around a periodic limit. These cases, and the issues of convergence and equilibrium states at large, are discussed in Sections 12 and 13. It is useful to speak about the algebraic process which is computed by the MCL algorithm in its own right. To this end, the notion of an MCL process is defined.

Definition 5 *A nonnegative column–homogeneous matrix M which is idempotent under matrix multiplication is called* **doubly idempotent**. □

Definition 6 *A general MCL* **process** *is determined by two rows of exponents $e_{(i)}$, $r_{(i)}$, where $e_i \in I\!N, e_i > 1$, and $r_i \in I\!R, r_i > 0$, and is written*

$$(\,\cdot\,, e_{(i)}, r_{(i)}) \tag{8.4}$$

An MCL process for stochastic matrices of fixed dimension $d \times d$ is written

$$(\,\cdot^{d\times d}\,, e_{(i)}, r_{(i)}) \tag{8.5}$$

An MCL process with input matrix M, where M is a stochastic matrix, is determined by two rows $e_{(i)}$, $r_{(i)}$ as above, and by M. It is written

$$(M, e_{(i)}, r_{(i)}) \tag{8.6}$$

Associated with an MCL process $(M, e_{(i)}, r_{(i)})$ is an infinite row of matrices $T_{(i)}$ where $T_1 = M$, $T_{2i} = \mathrm{Exp}_{e_i}(T_{2i-1})$, and $T_{2i+1} = \Gamma_{r_i}(T_{2i})$, $i = 1, \ldots, \infty$. □

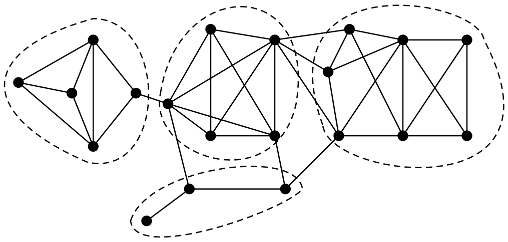

Figure 12: *MCL* Clustering of the graph in Figure 1.

In practice, the algorithm iterands converge nearly always to a doubly idempotent matrix. In Section 13 it is shown that the *MCL* process converges quadratically in the neighbourhood of doubly idempotent matrices. A sufficient property for associating a (possibly overlapping) clustering with a nonnegative column allowable matrix is that the matrix is idempotent under matrix multiplication. In [11] it is shown that the mapping of idempotent matrices onto overlapping clusterings according to Definition 8 can be generalized towards a mapping of time–reversible Markov matrices with nonnegative spectrum onto directed acyclic graphs. This is not a generalization in the strict sense, because stochastic idempotent matrices are in general not time–reversible. However the *MCL* process offers a perspective in which idempotent matrices are the extreme points of the set of time–reversible Markov matrices with nonnegative spectrum [11]. Figure 12 shows the clustering resulting from applying the *MCL* algorithm with standard parameters $e_{(i)} \stackrel{c}{=} 2$ and $r_{(i)} \stackrel{c}{=} 2$ to the example graph in Figure 1 taken from [31], loops added to the graph.

9. Basic *MCL* theory

This section is concerned with basic properties of the *MCL* process. The first section gives a generic mapping from nonnegative idempotent column allowable matrices onto overlapping clusterings. In Section 10 simple properties of the Γ operator are derived. Exceptional cyclic limits for which expansion and inflation act as each other's inverse are the subject of Section 12. The section after that is concerned with convergence towards equilibrium states and the stability of the *MCL* process around these states.

Mapping nonnegative idempotent matrices onto clusterings

The following theorem characterizes the structural properties of nonnegative column allowable idempotent matrices. Using this theorem, Definition 8 establishes a mapping from the class of nonnegative column allowable idempotent matrices to the set of overlapping clusterings. Nonnegative doubly idempotent matrices do not have stronger structural properties than matrices which are idempotent under matrix multiplication only. The theorem can easily be derived from the decomposition of nonnegative idempotent (not necessarily column allowable) matrices given in [5]. However, I choose to give a self-contained proof here, which is inspired more by graph–theoretical considerations. The proof of the theorem is easier to follow by first looking at the large matrix on page 23, and realizing that any nonnegative column allowable idempotent matrix must essentially have a similar 0/1 structure (the matrix is also stochastic and column homogeneous, which is not essential for the theorem below).

Theorem 1 *Let M be a nonnegative column allowable idempotent matrix of dimension N, let G be its associated graph. For s, t, nodes in G, write $s \to t$ if there is an arc in G from s to t. By definition,*

$s \to t \iff M_{ts} \neq 0$. *Let α, β, γ be nodes in G. The following implications hold.*

$$(\alpha \to \beta) \wedge (\beta \to \gamma) \implies \alpha \to \gamma \tag{9.1}$$

$$(\alpha \to \alpha) \wedge (\alpha \to \beta) \implies \beta \to \alpha \tag{9.2}$$

$$\alpha \to \beta \implies \beta \to \beta \tag{9.3}$$

Proof. The first statement follows from the fact that $M_{\gamma\alpha} = (M^2)_{\gamma\alpha} \geq M_{\gamma\beta}M_{\beta\alpha} > 0$. Suppose the second statement does not hold, then there exist α and β with $\alpha \to \alpha$, $\alpha \to \beta$, and $\beta \not\to \alpha$. Denote by V_α the set of nodes which reach α, denote by V_β the set of nodes reachable from β. Then $V_\alpha \neq \emptyset$ because $\alpha \to \alpha$, and $V_\beta \neq \emptyset$ because M is column allowable. It is furthermore true that $V_\alpha \cap V_\beta = \emptyset$ and that there is no arc going from V_β to V_α, for this would imply $\beta \to \alpha$ and $\beta \to \beta$ by 9.1. For $u, w \in V_\alpha, v \in V$, the property $u \to v \to w$ implies $v \in V_\alpha$. For $u, w \in V_\beta, v \in V$, the property $u \to v \to w$ implies $v \in V_\beta$. It follows that for all 2-step paths between node pairs respectively lying in V_α and V_β only indices lying in the same node set V_α, respectively V_β, need be considered. Reorder M and partition the matrix such that its upper left block has the form

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

where the indices of the diagonal block A_{11} correspond with all the elements in V_α, and the indices of the diagonal block A_{22} correspond with all the elements in V_β. It follows from the construction of V_α and V_β that all entries of A_{21} are positive, since for all $u \in V_\alpha, v \in V_\beta$, it is true that $u \to \alpha \to \beta \to v$. Similarly, $A_{12} = 0$. The observation made on 2-step paths with beginning and ending in V_α, respectively V_β, implies that $A_{11} = {A_{11}}^2$ and $A_{22} = {A_{22}}^2$. Furthermore, the inequality $A_{21} \geq A_{21}A_{11} + A_{22}A_{21}$ holds. Multiplying both sides on the left with A_{22} and on the right with A_{11}, the inequality $A_{22}A_{21}A_{11} \geq 2A_{22}A_{21}A_{11}$ results. The fact that A_{21} is positive, and the fact that A_{11} contains one positive row, i.e. the row corresponding with α, imply that $A_{21}A_{11}$ is positive too. Since A_{22} is nonzero, this implies that the product $A_{22}A_{21}A_{11}$ is nonnegative and nonzero, leading to a contradiction. The third statement follows by observing that there must be a path of infinite length going from α to β in G, that is, a path containing a circuit. If this were not the case, there would exist a $k \in I\!N$ such that $(M^k)_{\beta\alpha} = 0$, whereas $M_{\beta\alpha} \neq 0$. The existence of such a circuit implies by 9.2 and 9.3 that $\beta \to \beta$. □

Definition 7 *Let $G = (V, w)$ be the associated graph of a nonnegative voidfree idempotent matrix of dimension N, where $V = \{1, \ldots, N\}$. The node $\alpha \in V$ is called an* **attractor** *if $M_{\alpha\alpha} \neq 0$. If α is an attractor then the set of its neighbours is called an* **attractor system**. □

In the following a formal relationship is established between nonnegative idempotent matrices and overlapping clusterings. In order to sustain insight, it may again be helpful to keep the matrix on page 23 in mind. By Theorem 1, each attractor system in G induces a weighted subgraph in G which is complete. Theorem 1 furthermore provides the means to formally associate an overlapping clustering with each nonnegative column allowable idempotent matrix. Let M be an arbitrary nonnegative idempotent matrix, let $G = (V, w)$ be its associated graph. Denote by V_x the set of attractors of G. Denote the 'arc from $\cdot$ to $\cdot$' relationship in G by $(\cdot \to \cdot)$. The first two statements in Theorem 1 imply that $\to$ is transitive and symmetric on V_x, and $\to$ is reflexive on V_x by definition of V_x. Accordingly, $\to$ induces equivalence classes on V_x. Denote the set of equivalence classes by $\{E_1, \ldots, E_d\}$. The definition below requires the input of a column allowable matrix, in order to be able to distribute the elements of $V \backslash V_x$ over the classes E_i.

Definition 8 *Let M be a nonnegative column allowable idempotent matrix. Let $G = (V, w)$ be its associated graph, let $\to$ be the arc relation associated with G. Let V_x be the set of attractors in G, let $\mathcal{E} = \{E_1, \ldots, E_d\}$*

be the set of equivalence classes of $\rightarrow$ on V_x. Define a relation ν on $\mathcal{E} \times V$ by setting $\nu(E, \alpha) = 1$ if $\exists \beta \in E$ with $\alpha \rightarrow \beta$, and $\nu(E, \alpha) = 0$ otherwise. The overlapping clustering $\mathcal{CL}_M = \{C_1, \ldots, C_d\}$ associated with M, defined on V, has d elements. The i^{th} cluster $C_i, i = 1, \ldots, d$ is defined by Equation (9.4).

$$C_i \quad = \quad \{\, v \in V \mid \nu(E_i, v) = 1\} \tag{9.4}$$

□

Note that the set of clusters is precisely the set of weakly connected components[10] in the directed graph G. The inclusion $E_i \subset C_i$ implies that each cluster has at least one element which is unique for this cluster. All this is in line with the procedures followed while studying the example in the previous section. It should be noted that there is in general a very large number of nonnegative column allowable idempotent matrices which yield the same overlapping clustering according to Definition 8. This is caused by the fact that the number of attractors and the distribution of the attractors over the clusters may both vary without resulting in different clusterings. For example, printing attractors in boldface, the clustering $\{\{1,2\}, \{3,4,5\}\}$ results from all 21 possible combinations of the distributions $\{\mathbf{1}, 2\}$, $\{1, \mathbf{2}\}$, and $\{\mathbf{1}, \mathbf{2}\}$ for the first cluster, and the distributions $\{\mathbf{3}, 4, 5\}$, $\{3, \mathbf{4}, 5\}$, $\{3, 4, \mathbf{5}\}$, $\{\mathbf{3}, \mathbf{4}, 5\}$, $\{\mathbf{3}, 4, \mathbf{5}\}$, $\{3, \mathbf{4}, \mathbf{5}\}$, and $\{\mathbf{3}, \mathbf{4}, \mathbf{5}\}$ for the second cluster. Another example shows the extent to which complicated structure can be present in nonnegative idempotent matrices. The matrix

$$\begin{pmatrix}
1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/6 & 0 & 0 & 1/5 \\
1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/6 & 0 & 0 & 1/5 \\
1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/6 & 0 & 0 & 1/5 \\
0 & 0 & 0 & 1/4 & 1/4 & 1/4 & 1/4 & 0 & 0 & 0 & 0 & 0 & 1/7 & 0 & 0 \\
0 & 0 & 0 & 1/4 & 1/4 & 1/4 & 1/4 & 0 & 0 & 0 & 0 & 0 & 1/7 & 0 & 0 \\
0 & 0 & 0 & 1/4 & 1/4 & 1/4 & 1/4 & 0 & 0 & 0 & 0 & 0 & 1/7 & 0 & 0 \\
0 & 0 & 0 & 1/4 & 1/4 & 1/4 & 1/4 & 0 & 0 & 0 & 0 & 0 & 1/7 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 1/6 & 1/7 & 1/2 & 1/5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 & 1/6 & 1/7 & 1/2 & 1/5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1/6 & 1/7 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

is nonnegative idempotent and gives rise to the set $V_x = \{1,2,3,4,5,6,7,8,9,10\}$, to the equivalence classes $\{1,2,3\}$, $\{4,5,6,7\}$, $\{8,9\}$, $\{10\}$, and to the overlapping clustering $\{1,2,3\ 11,12,15\}$, $\{4,5,6,7,13\}$, $\{8,9,12,13,14,15\}$, $\{10,12,13\}$. This matrix is also doubly idempotent and column stochastic. The *MCL* process converges for nearly all input graphs to a doubly idempotent column stochastic limit[11]. For fixed dimension t, the class of doubly idempotent column stochastic matrices is finite, but extremely large. The fact that it is finite is easy to see: There is only a finite number of values that each matrix entry can assume, namely the set of rationals $\{0, 1, 1/2, \ldots, 1/t\}$.

The results in this section, especially Definition 8, which uses Theorem 1, establish a clear relationship between nonnegative column allowable idempotent matrices and overlapping clusterings. In practice, the equivalence classes $E_1, \ldots, E_d$ (see Definition 8) tend to be singleton sets, and overlap in the setting of undirected graphs has been observed only for graphs having certain symmetries. This is discussed in [10].

[10] For the definition of weakly connected components see page 5.

[11] This is suggested by practical evidence. It is conjectured in [11] that the *MCL* process converges almost always if the input graph is symmetric.

10. Mathematical properties of the inflation operator

The Γ operator establishes a majorization relationship between a probability vector and its image. This is stated in Lemma 2. Concerning just Γ this is a nice property, however, it does not give enough foothold by itself for describing the intricate interaction of the Γ operator with the Exp operator. The Γ operator furthermore distributes over the Kronecker product of matrices, which is stated in Lemma 4. Combined with the distributivity of normal matrix multiplication over the Kronecker product, this yields the result that for each *MCL* process the Kronecker product of the respective iterands corresponding with two input matrices A and B, is equal to the iterands corresponding with the input matrix which is the Kronecker product of A and B. This property is used in Section 11 to show the existence of certain periodic limits of the *MCL* process.

Following [28], if z denotes a real vector of length n, then $z_{[1]} \geq z_{[2]} \geq \cdots \geq z_{[n]}$ denote the entries of z in decreasing order.

Definition 9 *Let x, y be real nonnegative vectors of length n. The vector y is said to* **majorize** *the vector x if (10.1) and (10.2) hold. This is denoted by $x \prec y$.*

$$x_{[1]} + \cdots + x_{[k]} \leq y_{[1]} + \cdots + y_{[k]} \qquad k = 1, \ldots, n-1 \tag{10.1}$$

$$x_{[1]} + \cdots + x_{[n]} = y_{[1]} + \cdots + y_{[n]} \tag{10.2}$$

□

The relationship $\prec$ entails a rather precise mathematical notion of one vector x being more homogeneous than another vector y. It induces a partial order on each set of nonnegative vectors of fixed dimension. It turns out that the inflation operator Γ_r makes probability vectors less homogeneous for values $r > 1$, and makes probability vectors more homogeneous for values $r < 1$, which is stated in Lemma 2. This lemma follows from the fact that the vectors π and $\Gamma_r\pi$ satisfy the stronger condition of *majorization by ratio* (Lemma 3, also found in [28]).

Lemma 2 *Let π be a probability vector, let r be a real number, $r > 0$. The two inequalities (10.3) and (10.4) are implied by the fact that π and $\Gamma_r\pi$ satisfy the conditions of Lemma 3. The two equalities (10.5) and (10.6) are obvious.*

$$\pi \prec \Gamma_r\pi \qquad r > 1 \tag{10.3}$$

$$\pi \succ \Gamma_r\pi \qquad r < 1 \tag{10.4}$$

$$\pi = \Gamma_r\pi \qquad r = 1 \tag{10.5}$$

$$\pi = \Gamma_r\pi \qquad \pi \textit{ is homogeneous} \tag{10.6}$$

Definition 10 *Let x, y be real positive vectors of length n. The vector y is said to* **majorize by ratio** *the vector x, which is written $x \lhd y$, if $\sum x_i = \sum y_i$ and*

$$x_{[1]}/y_{[1]} \leq x_{[2]}/y_{[2]} \leq \cdots \leq x_{[n]}/y_{[n]} \tag{10.7}$$

□

Lemma 3 *([28], page 179) Majorization by ratio implies (normal) majorization.*

Proof. Without loss of generality, assume that $y_{[i]} = y_i$ and $x_{[i]} = x_i$. The claim is that for $k = 1, \ldots, n-1$,

$$\sum_{j=1}^{k} y_j \geq \sum_{j=1}^{k} x_j$$

This follows from

$$\begin{aligned}\sum_{j=1}^{k} y_j \sum_{l=1}^{n} x_l - \sum_{j=1}^{k} x_j \sum_{l=1}^{n} y_l &= \sum_{j=1}^{k} y_j \sum_{l=k+1}^{n} x_l - \sum_{j=1}^{k} x_j \sum_{l=k+1}^{n} y_l \\ &= \sum_{j=1}^{k} \sum_{l=k+1}^{n} y_j y_l \left(\frac{x_l}{y_l} - \frac{x_j}{y_j}\right) \quad \geq \quad 0\end{aligned}$$

□

The behaviour of $\Gamma_r(\pi)$ as r goes to infinity (where π is a stochastic vector of dimension n), is easily described. One has that $\lim_{r\to\infty} \Gamma_r(\pi)=(\sigma_1,\ldots,\sigma_n)$, where $\sigma_i=0$ if $\pi_i < \max_i \pi_i$ and $\sigma_i=1/m$ if $\pi_i=\max_i \pi_i$, where m is the number of indices i such that $\pi_i=\max_j \pi_j$. Also, $\Gamma_0(\pi)=(\tau_1,\ldots,\tau_n)$, where $\tau_i=0$ if $\pi_i=0$ and $\tau_i=1/k$ if $\pi_i \neq 0$, where k is the number of nonzero entries of π. The orbit of $\Gamma_r(\pi)$ under r $(0 \leq r \leq \infty)$, where π is a stochastic vector, has the property that $\Gamma_s(\pi) \triangleleft \Gamma_t(\pi)$ whenever $s < t$, and satisfies the multiplicative property $\Gamma_s\Gamma_t(\pi)=\Gamma_{st}(\pi)$. So the Γ_r operator is fairly well understood, and there are many results concerning the majorization relationship between vectors. One such result is the characterization of so called Schur–convex functions ϕ (which have the property that $x \prec y$ implies $\phi(x) \leq \phi(y)$) in terms of properties of their partial derivatives. In [10] a particular Schur–convex function is one of the main ingredients of a performance criterion for graph clustering. A celebrated result in the theory of majorization is that $x \prec y$ iff there is a doubly stochastic matrix D such that $x = Dy$ [27].

Unfortunately, results from the theory of majorization of vectors do not carry over to matrices in such a straightforward way (i.e. the columns of one matrix majorizing the columns of another matrix). In [27] this issue is discussed at length. However, Lemma 2 clearly shows the inflationary or 'decontracting' effect of Γ_r, $r > 1$, as opposed to the contracting effect of matrix multiplication of nonnegative matrices in terms of the so called *Hilbert distance* between positive vectors (see [11]). Moreover, the inflation operator preserves the majorization by ratio relationship between vectors. For certain perturbations of circulant limits of the *MCL* process introduced in Section 12, matrix multiplication preserves the normal majorization relationship. Both cases (Hilbert distance, majorization) exemplify the phenomenon that the workings of the expansion and inflation operator can be compared and contrasted in special cases. Annoyingly however, inflation does not necessarily preserve normal majorization, and expansion of circulants does not necessarily preserve majorization by ratio. A similar gap exists for the Hilbert distance.

Lemma 4 *Let A, B be nonnegative matrices of respective dimensions $s_1 \times t_1$ and $s_2 \times t_2$, let $r \in I\!R$ be positive. Denote the Kronecker product by $(\cdot \otimes \cdot)$. Equation (10.8) holds.*

$$\Gamma_r(A \otimes B) = \Gamma_r A \otimes \Gamma_r B \tag{10.8}$$

PROOF. Use the following notation for the Kronecker product of matrices. Let $(A \otimes B)_{i,j,k,l}$ denote the entry $(A \otimes B)_{is_2+k,jt_2+l}$, which is by definition equal to $A_{ij}B_{kl}$. Here $i = 1,\ldots,s_1$, $k = 1,\ldots,s_2$, $j = 1,\ldots,t_1$, and $l = 1,\ldots,t_2$. I prove Identity (10.8) by proving that the ratios between two entries in the same column is the same on both sides of Equation (10.8). Let i, j, k, l be as above and let i', k' be additional indices within the same bounds as respectively i and k. The indices j, l, identify the $(jt_1+l)^{th}$ column on both sides of (10.8). The two index pairs (i, k) and (i', k') identify two row entries in this column.

$$
\begin{aligned}
\frac{\Big(\Gamma_r(A\otimes B)\Big)_{i\,j\,k\,l}}{\Big(\Gamma_r(A\otimes B)\Big)_{i'\,j\,k'\,l}} &= \left(\frac{(A\otimes B)_{i\,j\,k\,l}}{(A\otimes B)_{i'\,j\,k'\,l}}\right)^r = \left(\frac{A_{i\,j}\;B_{k\,l}}{A_{i'\,j}\;B_{k'\,l}}\right)^r \\
&= \left(\frac{A_{ij}}{A_{i'j}}\right)^r \left(\frac{B_{kl}}{B_{k'l}}\right)^r = \frac{\Big(\Gamma_r A\Big)_{i\,j}}{\Big(\Gamma_r A\Big)_{i'j}} \frac{\Big(\Gamma_r B\Big)_{kl}}{\Big(\Gamma_r B\Big)_{k'l}} \\
&= \frac{\Big(\Gamma_r A\otimes\Gamma_r B\Big)_{i\,j\,k\,l}}{\Big(\Gamma_r A\otimes\Gamma_r B\Big)_{i'jk'l}}
\end{aligned}
$$

□

Lemma 5 *Let A, B, be square column stochastic matrices with no further restrictions imposed on their respective dimensions. Let $K = A\otimes B$ be their Kronecker product. Suppose all three are input to the same MCL process $(\cdot, e_{(i)}, r_{(i)})$. Denote the respective iterand pairs by (A_{2i}, A_{2i+1}), (B_{2i}, B_{2i+1}), (K_{2i}, K_{2i+1}), $i = 1,\ldots,\infty$. Identity (10.9) holds.*

$$K_j = A_j \otimes B_j \qquad\qquad j = 1,\ldots,\infty \tag{10.9}$$

PROOF. *The lemma follows from the observation that both matrix multiplication and Γ distribute over the Kronecker product.* □

11. Equilibrium states of the *MCL* process

In order to characterize the equilibrium states of the *MCL* process, I make two extra assumptions on the input rows $r_{(i)}$ and $e_{(i)}$. These are

i) $r_i = c$ eventually, $c \in I\!R, c > 1$.

ii) $e_i = 2$ eventually.

The main purpose of these requirements is to study for specific parameters whether matrices exist corresponding with periodic limits. This question will be answered affirmatively below. The first requirement implies that the process differs genuinely from the usual Markov process. It is necessary to require $r_i > 1$ eventually in order to ensure that the limit of the corresponding *MCL* process can in principle have structural properties which are different from the original input graph in terms of the number and distribution of the weakly connected components. Consider a regular ergodic input graph (all example graphs in the figures except graph G_2 in Figure 4 are regular and ergodic). The structural properties of all intermediate iterands (with respect to reachability) are identical, and positive entries can thus only *tend* to zero eventually, they can not become equal to zero eventually. It is true only for the limit of the process that it may differ structurally from the input graph.

An equilibrium state corresponds with an *MCL* process $(M, e_{(i)}, r_{(i)})$ with $e_{(i)} \stackrel{c}{=} 2$, and $r_{(i)} \stackrel{c}{=} c > 1$, for which the associated row of matrix pairs $(T_{(2i)}, T_{(2i+1)})$ is periodic. A periodic row of objects is a row consisting of a finite list of objects repeated infinitely many times. The period of a periodic row is the minimum cardinality of such a finite list, the period of a constant row is 1. An equilibrium state can be associated with the input matrix M, with the infinite row $(T_{(2i)}, T_{(2i+1)})$ generated by M, and with a finite list of matrices constituting a cycle of period p in $(T_{(2i)}, T_{(2i+1)})$. A priori, I distinguish three different types L_i $(i = 1, \ldots, 3)$ of equilibrium states for the MCL process with column stochastic input matrix M, input row $r_{(i)} \stackrel{c}{=} c > 1$, and input row $e_{(i)} \stackrel{c}{=} 2$. A matrix M is said to be of type L_i if its associated output row is of type L_i. In order of decreasing strength of properties, the types L_i are:

- L_1 M is doubly idempotent, implying that all matrices T_{2i} and T_{2i+1} are equal.
- L_2 The row of pairs $(T_{2(i)}, T_{2(i)+1})$ has period 1. Even iterands are $(\mathrm{Exp}_2 \circ \Gamma_c)$–$id$, odd iterands are $(\Gamma_c \circ \mathrm{Exp}_2)$–$id$, and $T_{2i} \neq T_{2i+1}$.
- L_3 The row of pairs $(T_{2(i)}, T_{2(i)+1})$ has period $p > 1$, that is, $T_{2i} = T_{2(i+p)}$ and $T_{2i+1} = T_{2(i+p)+1}$. The even iterands T_{2i} are idempotents under p iterations of the operator $(\mathrm{Exp}_2 \circ \Gamma_c)$, the odd iterands T_{2i+1} are idempotents under p iterations of the operator $(\Gamma_c \circ \mathrm{Exp}_2)$.
- L_{3a} As above, where the matrix T_1 is the Kronecker product of a column homogeneous column stochastic cyclic matrix P with odd period and a matrix A which is of type L_2 or L_1. An example of such P is a permutation matrix containing cycles of odd period only.

Each of the classes L_1, L_2, and L_3 is non-empty. The most important class of equilibrium states is the large class L_1 of doubly idempotent matrices. These matrices are invariant under arbitrary MCL processes. For dimensions $2, 3, 4, 5$ a few matrices of L_2 type for $c = 2$ can be found quickly by algebraic computation. They are depicted on page 29. The general graph templates on n nodes, $n = 2, \ldots, 5$, which were used to derive these examples, are invariant under the automorphism group of the ring-graph of order n. Note that the matrix R_{4b} is the Kronecker product of the matrices $1/2 J_2$ and R_{2a}, where J_2 is the all-one matrix of dimension 2. Higher dimensional versions of the templates in Figure 13 have solutions as well (Lemma 6).

The only clusterings suiting ring graphs are the two extreme clusterings. Slight perturbations of either the MCL process parameters or the input graphs lead the MCL algorithm to converge towards a limit of the L_1 type, corresponding with one of the two extreme clusterings. For example, setting $p = 101/601$ in the 3-dimensional matrix template in Figure 13 leads the algorithm to convergence to the identity matrix, setting $p = 99/601$ leads the algorithm to converge to $1/3\ J$, where J is the all-one matrix. The same behaviour results after respectively setting $c = 201/100$ and $c = 199/100$. For the latter settings, it is in line with heuristic considerations that a slight increase in inflation leads the algorithm to converge towards a matrix corresponding with the bottom extreme partition (i.e. $\{\text{singletons} V\}$), and that a slight decrease in inflation leads the algorithm to converge to a matrix corresponding with the top extreme partition (i.e. $\{V\}$).

The class L_2 consists of equilibrium states which are very instable by nature. The image of the column vectors under either Γ_2 or Exp_2 is very different from the original vector. For this class, expansion and inflation act as each others inverse. A slight perturbation of the MCL process parameters or the equilibrium state leads to one of the two getting the upper hand. This is formally proved for a subclass of the class L_2 in Lemma 6.

So far, all limits resulting from inputting undirected graphs were of the L_1 type. If the condition $e_{(i)} \stackrel{c}{=} 2$ is relaxed to $e_{(i)} \stackrel{c}{=} k$, where $k \in I\!N$ is a constant, examples of the L_{3a} type can be found as well by selecting bipartite graphs, setting $e_{(i)} \stackrel{c}{=} 3$, and refraining from adding loops. This is not surprising, since in bipartite graphs paths of odd length always go from one of the two node sets to the other. As was the case with ring-graphs, the relationship between parameter choice, expected behaviour, and observed behaviour fully agree, so this is an agreeable situation.

The class L_3 is nonempty for rows $e_{(i)} \stackrel{c}{=} 2$ as well. It is easy to construct matrices of the L_{3a} type, by taking the Kronecker product of L_1– or L_2–type matrices and permutation matrices containing odd permutations only, illustrating the use of Lemma 4. Denote by $L_x \backslash L_y$ the class of matrices satisfying the L_x constraints but not satisfying the L_y constraints. It is an open question whether matrices of the type $L_3 \backslash L_{3a}$ exist. If they exist, I expect them in any case to be as sensitive to perturbations of parameter settings and matrix values as are the matrices of the L_2 type. While the L_3 and L_2 classes are of interest for studying the *MCL* process, they do not form a weak spot of the *MCL* algorithm. If a graph constructed from some application such as a thesaurus or a database leads to an *MCL* process which at any stage approaches an L_2 or L_3 type matrix, then the application graph is in all likelihood a curiosity lacking cluster structure anyhow. Moreover, limits of L_3 type have non-real spectrum, and cannot occur if the input graph is symmetric. This follows from the results in [11].

12. Flip–flop equilibrium states

There is a class of matrices which is known not to lead to convergence. In small dimensions, it is easy to find matrices M such that $\Gamma_2 M = M^{1/2}$, representing a flip–flop equilibrium state. Several of these are depicted in Figure 13, each having the form of a symmetric circulant matrix. The three-dimensional specimen is notable for its simple (rational) form. The Kronecker product K of such a matrix with any other stochastic matrix has the property that the *MCL* process $(K, e_{(i)} = 2, r_{(i)} = 2)$ does not converge towards a doubly idempotent matrix. However, such flip–flop equilibrium states are sensitive to perturbations. This can be proven for a subclass of them.

There exists an infinite family of 'basic' (indecomposable in terms of the Kronecker product) flip–flop positive semi-definite equilibrium states of the form $aI_n + (1-a)/nJ_n$. For these states it is relatively easy to prove that they are instable with respect to alternation of Exp_2 and Γ_2.

Lemma 6 *Let $n > 1$. Define α_n by*

$$\alpha_n = \frac{\sqrt[3]{v_n}}{6(n-1)} - \frac{2(3n-4)}{3(n-1)\sqrt[3]{v_n}} - \frac{1}{3(n-1)} \tag{12.1}$$

$$v_n = 108n^2 - 180n + 64 + 12(n-1)\sqrt{3n(27n-32)} \tag{12.2}$$

Then the n-dimensional matrix $A_n = \alpha_n I_n + (1-\alpha_n)/nJ_n$ has the property that $\Gamma_2({A_n}^2) = A_n$. In the class of matrices $\{aI_n + (1-a)/nJ_n | a \in [0,1]\}$, there is no trajectory to the equilibrium (flip–flop) state A_n for the MCL process with parameters e_i and r_i constant equal to 2, thus these states are instable for this process.

Proof. This is derived by computing the square of $A = aI_n + (1-a)/nJ_n$, which equals $B = a^2 I_n + (1 - a^2)/nJ_n$, and subsequently solving for $(B_{11}/B_{12})^2 = A_{11}/A_{12}$. This yields the equation $a(1-a)(a^3(n-1) + a^2 + a - 1) = 0$. The solutions $a = 0$ and $a = 1$ yield the double idempotents I_n and J_n; the term of degree three yields the solution as stated in the lemma. It is straightforward to prove that this term has only one solution in the interval $(0,1)$ (and in fact, only one real solution). It follows that for $a > \alpha_n$ the *MCL* process $(aI_n + (1-a)/nJ_n, e_{(i)} = 2, r_{(i)} = 2)$ converges towards I_n, and that for $a < \alpha_n$ the process converges towards J_n, as is to be expected. The cases where $n = 2, 3, 4, 5$ are depicted in Figure 13. □

In general, one might hope that the analysis of the stability of flip–flop states which correspond with symmetric circulants is easier, even if no explicit representation is known. However, it is difficult to describe expansion and inflation in the same framework. Suppose that a is a positive vector such that the circulant C_a is a flip–flop state, i.e. $\Gamma_2({C_a}^2) = C_a$. Let e be a vector the elements of which sum to zero such that $a + e$ is a nonnegative vector satisfying $a + e \prec a$, let f be likewise a vector such that $a + f \triangleleft a$. Extend the definition of $\prec$ ($\triangleleft$) to circulants by setting $C_x \prec C_y$ iff $x \prec (\triangleleft) y$. Now it is easy to prove that $C_x \prec C_y \implies {C_x}^2 \prec {C_y}^2$,

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix} \quad \begin{pmatrix} 1-2p-2q & p & q & q & p \\ p & 1-2p-2q & p & q & q \\ q & p & 1-2p-2q & p & q \\ q & q & p & 1-2p-2q & p \\ p & q & q & p & 1-2p-2q \end{pmatrix}$$

$$\begin{pmatrix} 1-2p & p & p \\ p & 1-2p & p \\ p & p & 1-2p \end{pmatrix} \quad \begin{pmatrix} 1-2p-q & p & q & p \\ p & 1-2p-q & p & q \\ q & p & 1-2p-q & p \\ p & q & p & 1-2p-q \end{pmatrix}$$

General templates for $(\Gamma_2 \circ \mathrm{Exp}_2)$–$id$ matrices in dimensions $2, 3, 4$, and 5. Explicit solutions for the resulting equations are given below.

$$R_{2a} = \begin{pmatrix} 0.77184 & 0.22816 \\ 0.22816 & 0.77184 \end{pmatrix} \qquad \begin{aligned} p &= \tfrac{2}{3} - \sqrt[3]{v} + \tfrac{1}{18\sqrt[3]{v}}, \\ v &= \tfrac{17}{216} + \tfrac{1}{72}\sqrt{33} \end{aligned}$$

$$R_{3a} = \begin{pmatrix} 2/3 & 1/6 & 1/6 \\ 1/6 & 2/3 & 1/6 \\ 1/6 & 1/6 & 2/3 \end{pmatrix} \qquad p = \tfrac{1}{6}$$

$$R_{4a} = \begin{pmatrix} 0.60205 & 0.13265 & 0.13265 & 0.13265 \\ 0.13265 & 0.60205 & 0.13265 & 0.13265 \\ 0.13265 & 0.13265 & 0.60205 & 0.13265 \\ 0.13265 & 0.13265 & 0.13265 & 0.60205 \end{pmatrix} \qquad \begin{aligned} q &= p, \quad p = \tfrac{5}{18} - \sqrt[3]{v} + \tfrac{1}{162\sqrt[3]{v}} \\ v &= \tfrac{67}{23328} + \tfrac{1}{2592\sqrt{57}} \end{aligned}$$

$$R_{4b} = \begin{pmatrix} 0.38592 & 0.11408 & 0.38592 & 0.11408 \\ 0.11408 & 0.38592 & 0.11408 & 0.38592 \\ 0.38592 & 0.11408 & 0.38592 & 0.11408 \\ 0.11408 & 0.38592 & 0.11408 & 0.38592 \end{pmatrix} \qquad \begin{aligned} q &= \tfrac{1}{2} - p, \quad p = \tfrac{1}{3} - \sqrt[3]{v} + \tfrac{1}{72\sqrt[3]{v}} \\ v &= \tfrac{17}{1728} + \tfrac{1}{576\sqrt{33}} \end{aligned}$$

$$R_{4c} = \begin{pmatrix} 0.59594 & 0.17610 & 0.05205 & 0.17610 \\ 0.17610 & 0.59594 & 0.17610 & 0.05205 \\ 0.05205 & 0.17610 & 0.59594 & 0.17610 \\ 0.17610 & 0.05205 & 0.17610 & 0.59594 \end{pmatrix} \qquad \begin{aligned} q &= p - 4p^2, \quad p = \tfrac{1}{3} - \sqrt[3]{v} + \tfrac{18}{\sqrt[3]{v}} \\ v &= \tfrac{13}{864} + \tfrac{1}{288\sqrt{57}} \end{aligned}$$

$$R_{5a} = \begin{pmatrix} 0.5568 & 0.1108 & 0.1108 & 0.1108 & 0.1108 \\ 0.1108 & 0.5568 & 0.1108 & 0.1108 & 0.1108 \\ 0.1108 & 0.1108 & 0.5568 & 0.1108 & 0.1108 \\ 0.1108 & 0.1108 & 0.1108 & 0.5568 & 0.1108 \\ 0.1108 & 0.1108 & 0.1108 & 0.1108 & 0.5568 \end{pmatrix} \qquad \begin{aligned} q &= p, \quad p = \tfrac{13}{60} - \sqrt[3]{v} + \tfrac{11}{3600\sqrt[3]{v}} \\ v &= \tfrac{233}{216000} + \tfrac{1}{36000\sqrt{1545}} \end{aligned}$$

$$R_{5b} = \begin{pmatrix} 0.5346 & 0.2087 & 0.0239 & 0.0239 & 0.2087 \\ 0.2087 & 0.5346 & 0.2087 & 0.0239 & 0.0239 \\ 0.0239 & 0.2087 & 0.5346 & 0.2087 & 0.0239 \\ 0.0239 & 0.0239 & 0.2087 & 0.5346 & 0.2087 \\ 0.2087 & 0.0239 & 0.0239 & 0.2087 & 0.5346 \end{pmatrix}$$

Values are numerically found roots of a polynomial of degree 8 which is irreducible over the rationals.

Figure 13: $(\Gamma_2 \circ \mathrm{Exp}_2)$–$id$ matrices.

and that $C_x \lhd C_y \implies \Gamma_r(C_x) \lhd \Gamma_r(C_y)$ $(r \geq 1)$. Unfortunately, neither of the corresponding statements of the other pairings $\Gamma_r, \prec$ and $\mathrm{Exp}_2, \lhd$ is in general true, which severely impedes the analysis of the stability of flip–flop states.

One interesting freak flip–flop state exists in dimension 3, which has the form of a nonsymmetric circulant matrix corresponding with the generating vector $(1-b-c, b, c)$. Testing this template for a flip–flop solution in Maple yields an algebraic number α of the form $h(\beta)$, where β is a zero of g, where g is a polynomial of degree 16, and where h is a polynomial of degree 10 divided by a polynomial of degree 9. Numerical computations yield and verify that the matrix below is a genuine flipflop equilibrium state.

$$\begin{pmatrix} 0.795668870 & 0.004344249 & 0.199986881 \\ 0.199986881 & 0.795668870 & 0.004344249 \\ 0.004344249 & 0.199986881 & 0.795668870 \end{pmatrix} \tag{12.3}$$

13. Convergence towards equilibrium states

In this section the stability of the equilibrium states in L_1 is considered. The setting is as follows. Let M be the associated matrix of an equilibrium state in L_1, let ϵ be a perturbation matrix such that $M+\epsilon$ is stochastic. For various types of perturbation ϵ the limit or set of possible limits of the perturbed *MCL* process $(M+\epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)}) \overset{c}{=} 2$ is investigated. The states in L_1 which are stable in every respect correspond with doubly idempotent matrices which have precisely one nonzero entry (equal to 1) in each column. This is stated in Theorem 2. A doubly idempotent matrix M corresponds with an instable equilibrium state if it has columns with more than one nonzero entry. Two cases can be distinguished: the case where all columns with multiple entries correspond with nodes which are attracted to or are part of a single attractor system having more than one attractor (Lemma 8), and the case where p is not an attractor and is attracted to two different attractor systems (Lemma 9). For both cases, it is of interest in which respects the associated clustering of a limit resulting from the perturbed *MCL* process may differ from the associated clustering of M.

In the first case, the equilibrium state is shown to be stable on a macroscopic scale which corresponds with the cluster structure derived from M (Theorem 4). A perturbation ϵ of M may thus lead the *MCL* process $(M+\epsilon, e_{(i)}, r_{(i)})$ to converge towards a different equilibrium state. Theorem 4 guarantees that this new equilibrium state yields a cluster interpretation which is identical to or a refinement of the associated clustering of M. For a restricted class of perturbations ϵ, Theorem 5 guarantees that the new equilibrium state yields a cluster interpretation which is identical to the associated clustering of M. These are perturbations only affecting the principal submatrices $M[\alpha]$, where α is any index set describing an attractor system in M. In words, Theorem 5 states that for such a perturbation an attractor system cannot split into a number of smaller attractor systems.

In the second case, if a perturbation of column p is unevenly spread over the attractor systems towards which p is attracted, then the process $(M, e_{(i)}, r_{(i)})$ will converge towards a state in which p is attracted to just one of those systems. This means that the phenomenon of cluster overlap is instable in nature (Lemma 9). The following theorem identifies the equilibrium states in L_1 for which the associated matrix M is attractor for all input matrices $M+\epsilon$ with regard to the *MCL* process $(M+\epsilon, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$, for ϵ small enough.

Theorem 2 *The MCL process with standard parameters* $(\cdot, e_{(i)} \overset{c}{=} 2, r_{(i)} \overset{c}{=} 2)$*, converges quadratically in the neighbourhood of each nonnegative idempotent column stochastic matrix for which every column has one entry equal to* 1 *and all the other entries equal to* 0*.*

The formulation of this theorem is rather non-technical. What I shall prove is Lemma 7.

Lemma 7 *Let $M \in {I\!R_{\geq 0}}^{n\times n}$ be a nonnegative idempotent column stochastic matrix for which every column has one entry equal to 1 and all other entries equal to 0. Let x_i be the row index such that $M_{x_i i} = 1$. Let $f > 0$ be a real number and let ϵ be a matrix in ${I\!R}^{n\times n}$, the columns of which add to zero, such that $M + \epsilon$ is column stochastic and nonnegative, and such that $[M + \epsilon]_{x_i i} \geq 1 - f$. Define the matrix δ by $\Gamma((M + \epsilon)^2) = M + \delta$.*

For $f \geq 1/4$ the inequality $\max_{i,j} |\delta_{ij}| \leq 8f^2$ holds.

Proof. The structure of nonnegative idempotent matrices as described in Theorem 1 implies the equality $x_{x_i} = x_i$, by the implication $i \to x_i \implies x_i \to x_i$. It furthermore follows from the definition of ϵ that $\max_{i,j} |\epsilon_{ij}| \leq f$. Consider the entry ${[M + \epsilon]^2}_{x_i i}$. The inequalities ${[M + \epsilon]^2}_{x_i i} \geq {[M + \epsilon]^2}_{x_i x_i} {[M + \epsilon]^2}_{x_i i} \geq (1 - f)^2 \geq 1 - 2f$ hold. Now consider the entry $[\Gamma(M + \epsilon)]_{x_i i}$. It is true that $\sum_k {(M + \epsilon)_{ki}}^2 \geq (1 - f)^2$. Furthermore, $\sum_{k \neq x_i} (M + \epsilon)_{ki} \leq f$ and thus $\sum_{k \neq x_i} {(M + \epsilon)_{ki}}^2 \leq f^2$. It follows that $\sum_{k \neq x_i} [\Gamma(M + \epsilon)]_{ki} \leq f^2/(1-f)^2$, and consequently $[\Gamma(M+\epsilon)]_{ki} \geq 1-f^2/(1-f)^2$. For $f < 1/4$) the inequality $1-f^2/(1-f) \geq 1-2f^2$ holds. Combining this inequality and the previous one yields the desired result. □

Theorem 3 *The equilibrium states of the MCL process in L_1 for which the associated doubly idempotent matrices have one or more columns with more than one nonzero entry are instable.*

Two cases are distinguished in proving this theorem, namely the case in which a column with more than one nonzero entry corresponds with an attractor, and the case in which it corresponds with a non-attractor. Both cases are illustrated with simple examples which generalize in a straightforward manner to higher dimensional and more complex cases.

Lemma 8 *Let M, ϵ_f and L be the matrices*

$$M = \begin{pmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{pmatrix} \quad \epsilon_f = \begin{pmatrix} f & f \\ -f & -f \end{pmatrix} \quad L = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$$

For each $f > 0$ the MCL process $(M + \epsilon_f, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$ converges towards L.

Proof. The matrix $M + \epsilon_f$ is idempotent under matrix multiplication for arbitrary f, as it is a rank 1 stochastic matrix. Direct computation shows that $[\Gamma(M + \epsilon_f)]_{11}$ equals $(1/4 + f^2 + f)/1/2 + 2f = 1/2 + 2f/(1 + 4f^2)$. Thus $\Gamma(M + \epsilon_f)$ can be written as $M + \epsilon_{2f/(1+4f^2)}$. For small f, the deviation of $\Gamma(M + \epsilon_f)$ from M is nearly twice as large as the deviation of $M + \epsilon_f$ from M. The lemma follows. □

The proof of the following lemma is nearly identical and is omitted.

Lemma 9 *Let M, ϵ_f and L be the matrices*

$$M = \begin{pmatrix} 1 & 0 & 1/2 \\ 0 & 1 & 1/2 \\ 0 & 0 & 0 \end{pmatrix} \quad \epsilon_f = \begin{pmatrix} 0 & 0 & f \\ 0 & 0 & -f \\ 0 & 0 & 0 \end{pmatrix} \quad L = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

For each $f > 0$ the MCL process $(M + \epsilon_f, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$ converges towards L. □

The previous results do not imply that the *MCL* algorithm is built on quicksand. The instability of the phenomenon of cluster overlap cannot be helped, if only the limit of the *MCL* process is taken into account.

As mentioned before, there is a cure for this by looking at the specific structure which is present in all iterands of the process [11].

The instability of attractor systems consisting of more than one element is not a serious issue if only regarding clustering purposes. Below it is shown that perturbation of doubly idempotent matrices M by a matrix ϵ for which the associated clustering $\mathcal{C}$ does not have overlap, lead the iterands of the MCL process $(M + \epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$ to stay within a class of matrices the block structure of which only admits a clustering which is a refinement of $\mathcal{C}$. These statements are assembled in Theorem 4, which is preceded by two more technical lemmas. This result is extended by Theorem 5, which demonstrates that for a specific class of perturbations the notion 'a refinement of' in Theorem 4 can be strengthened to 'identical to'. The proof of this theorem gives confidence that the result extends to arbitrary perturbations.

If a diagonal block structure can be mapped onto part of a column stochastic matrix M such that the mass of the columns in this part is highly concentrated in the blocks, then the entries outside the diagonal blocks tend to zero quadratically in the MCL process $(M, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$. If it is moreover assumed that the mass of the columns in the remaining part is (for each column separately) concentrated in a set of rows corresponding to at most one diagonal block, then the entries not belonging to these rows tend to zero as well. Conceptually, the proof is very similar to that of Lemma 7. The more complicated setting requires substantial elaboration.

Let M be a column stochastic matrix of dimension n, let $f > 0$ be a real number. Assume that there is a strictly increasing row of indices $k_1, \ldots, k_{l+1}$ with $k_1 = 1$ and $k_{l+1} \leq n+1$ such that the mass of the columns in each principal submatrix $M[k_i, \ldots, k_{i+1}{-}1]$, $i = 1, \ldots, l$ is greater than or equal to $1 - f$. It is convenient to denote the set of indices $\{k_x, \ldots, k_{x+1}{-}1\}$ by α_x, indicating the x^{th} diagonal block.

Lemmas 10 and 11 hold, and are preparatory to Theorem 4. The corresponding statements for matrices which are permutation–similar to a matrix with the required block structure follow from the fact that both matrix multiplication and inflation distribute over simultaneous permutation of rows and columns.

Lemma 10 *Let f, M and $k_0, \ldots, k_l$ be as above. Let T_{2i} and T_{2i+1} be the iterands of the MCL process $(M, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$, where $T_1 = M$. Let α_x be the range of indices $\{k_x, \ldots, k_{x+1}{-}1\}$ and let q be an index in α_x. For f small enough, the entries $(T_i)_{jq}$ tend to zero for all j with $j \notin \alpha_x$ as i goes to infinity.*

Proof. Suppose that $k_{l+1} < n+1$. Thus, the block diagonal structure (the blocks of which have large mass) does not fully cover M, as the last block is indexed by the range $k_l, \ldots, k_{l+1}{-}1$. This is the most general case where nothing is assumed about the remaining columns $k_{l+1}, \ldots, n$. Let α_x and q be as in the lemma, so $q \in \alpha_x$. Let p be any index, $1 \leq p \leq n$.

Consider the p^{th} entry of the q^{th} column of M^2. Consider first the case where $k_{l+1} \leq p \leq n$. The identity ${M^2}_{pq} = \sum_{i=1}^{n} M_{pi} M_{iq}$ holds. Split the latter sum into the parts $\sum_{i \in \alpha_x} M_{pi} M_{iq}$ and $\sum_{i \notin \alpha_x} M_{pi} M_{iq}$. For $i \in \alpha_x$ the inequality $M_{pi} \leq f$ holds. Since $\sum_{i \in \alpha_x} M_{iq} \leq 1$, the first sum is smaller then or equal to f. By similar reasoning it is found that the second sum is smaller than or equal to f^2.

Now consider the case where $p \in \alpha_y, y \neq x$. Write the entry ${M^2}_{pq}$ in three parts: $\sum_{i \in \alpha_x} M_{pi} M_{iq}$, $\sum_{i \in \alpha_y} M_{pi} M_{iq}$, and $\sum_{i \notin \alpha_x \cup \alpha_y} M_{pi} M_{iq}$. For the first part, $M_{pi} \leq f$ and the entries M_{iq} sum to less than one. For the second part, the entries M_{pi} sum to less than $|\alpha_y|$ and $M_{iq} \leq f$. For the third part, $M_{pi} \leq f$ and the entries M_{iq} sum to less than f. Combining these results yields that the full sum is smaller than or equal to $f + |\alpha_y| f + f^2$. So after multiplication, the combined mass of all entries in column q which are not in α_x is bounded from above by $n(n+l)(f + f^2)$, which is of order f.

Estimate the entry $[\Gamma(M)]_{pq}$ as follows. The sum of squares $\sum_{i=1}^{n} {M_{iq}}^2$ is bounded from above by $1/n$. For

$p \notin \alpha_x$ the inequality ${M_{pq}}^2 \leq f^2$ holds and thus $[\Gamma(M)]_{pq} \leq nf^2$. The combined mass of all entries in column q which are not in α_x is thus bounded from above by the (crude) estimate $n^2 f$, which is of order f^2. Combination of this with the result on multiplication yields the following. If f is viewed as the error with which M deviates from the block structure imposed by the index sets α_x (in the index range $1, \ldots, k_{l+1}-1$), then application of $\Gamma \circ \mathrm{Exp}_2$ to M yields a matrix for which the new error is of order f^2. This proves the lemma. □

Lemma 11 *Let f, M and $k_1, \ldots, k_{l+1}$ be as in Lemma 10. Assume moreover that $k_{l+1} < n+1$ and that for each $q \geq k_{l+1}$ there exists a block indexed by $\alpha_x = \{k_x, \ldots, k_{x+1}-1\}$ such that the mass in the submatrix $M[\alpha_x|q]$ (which is part of column q) is bounded from below by $1-f$. Let T_i be the iterands of the MCL process $(M, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$. Then, for f small enough, all entries $(T_i)_{pq}$ tend to zero for $p \notin \alpha_x$ as i goes to infinity.*

Proof. The proof is very similar to that of Lemma 11. Consider the p^{th} entry of the q^{th} column of M^2. First consider the case where $k_{l+1} \leq p \leq n$. The identity ${M^2}_{pq} = \sum_{i=1}^{n} M_{pi} M_{iq}$ holds. Split the latter sum into the parts $\sum_{i \in \alpha_x} M_{pi} M_{iq}$ and $\sum_{i \notin \alpha_x} M_{pi} M_{iq}$. As in the proof of Lemma 11 it is found that the two parts are respectively bounded from above by f and f^2.

Now consider the case where $p \in \alpha_y, y \neq x$. Writing the entry ${M^2}_{pq}$ in three parts: $\sum_{i \in \alpha_x} M_{pi} M_{iq}$, $\sum_{i \in \alpha_y} M_{pi} M_{iq}$, and $\sum_{i \notin \alpha_x \cup \alpha_y} M_{pi} M_{iq}$, it is found that these parts are respectively bounded by f, $|\alpha_y| f$, and f^2. After multiplication, the combined mass of all entries in column q which are not in α_x is bounded from above by $n(n+l)(f+f^2)$, which is of order f.

The entry $[\Gamma(M)]_{pq}$ is estimated as before, yielding $[\Gamma(M)]_{pq} \leq nf^2$, and bounding the combined mass of the entries $[\Gamma(M)]_{pq}$, $q \notin \alpha_x$ by $n^2 f$. Viewing f as the error with which column q deviates from the structure imposed by α_x gives that applying $\Gamma \circ \mathrm{Exp}_2$ to M yields a matrix for which the new error is of order f^2. This proves the lemma. □

Theorem 4 is a general result on perturbation of equilibrium states for which the associated matrix M may have columns with more than one nonzero entry. It states that the associated clustering of any idempotent limit resulting from the perturbed process must be a refinement of the clustering associated with M. The proof of the theorem is a direct consequence of Lemma 10 and 11.

Theorem 4 *Let M be a doubly idempotent matrix in ${\mathbb{R}_{\geq 0}}^{n \times n}$ for which the associated clustering $\mathcal{C}$ is free of overlap. Let $f > 0$ and let ϵ be a matrix in $\mathbb{R}^{n \times n}$, the columns of which sum to zero and for which $\max_{i,j} |\epsilon_{ij}| \leq f$. The iterands T_i of the MCL process $(M+\epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$, for f small enough, have the property that $(T_i)_{pq}$ tends to zero as i goes to infinity, if $q \not\to p$ in the associated graph of M. Consequently, an idempotent limit resulting from the process $(M+\epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$ corresponds with a clustering which is identical to or a refinement of $\mathcal{C}$.* □

The following theorem extends this result for a restricted class of perturbations, namely those that only affect the principal submatrices of the doubly idempotent matrix M which correspond to an attractor system in the associated clustering of M. Theorem 1 implies that such a submatrix has the form $\frac{1}{k} J_k$, where J_k is the all one matrix of dimensions $k \times k$. Theorem 5 is concerned with limits which may possibly result from the *MCL* process $(\frac{1}{k} J_k + \epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$, where ϵ is as before. It appears that for small perturbations ϵ it is guaranteed that the iterands of the process approach arbitrarily close towards the set of rank 1 stochastic matrices, without actually pinpointing a particular limit point. This implies that an idempotent limit of the perturbed process $(M+\epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$, where M is doubly idempotent and ϵ only affects the attractor systems of M, is guaranteed to yield an associated clustering which is the same as that of M, except for the cases where overlap occurs.

Theorem 5 *Let M be a doubly idempotent matrix in $\mathbb{R}_{\geq 0}{}^{n\times n}$ for which the associated clustering $\mathcal{C}$ is free of overlap. Let $f > 0$ and let ϵ be a matrix in $\mathbb{R}^{n\times n}$, the columns of which sum to zero, for which $\max_{i,j} |\epsilon|_{ij} \leq f$, and for which $\epsilon_{kl} \neq 0 \implies k$ and l are attractors in the same attractor system in M. That is, ϵ only affects the diagonal blocks of M corresponding with its attractor systems.*

An idempotent limit resulting from the process $(M + \epsilon, e_{(i)} \stackrel{c}{=} 2, r_{(i)} \stackrel{c}{=} 2)$, has an associated clustering which is identical to $\mathcal{C}$.

This theorem is a consequence of the following lemma. Note that the diagonal blocks of M corresponding with its attractor systems are of the form $\frac{1}{k}J_k$.

Lemma 12 *Let $f > 0$ be a real number, let J be an arbitrary rank 1 column stochastic matrix in $\mathbb{R}_{\geq 0}{}^{n\times n}$, let $\epsilon \in \mathbb{R}^{n\times n}$ be a matrix the columns of which sum to zero and for which $\max_{i,j} |\epsilon|_{ij} \leq f$. For f small enough, the matrix $\Gamma_2[(J+\epsilon)^2]$ can be written as $J' + \delta$, where J' is rank 1 column stochastic, the columns of δ sum to zero and $\max_{i,j} |\delta|_{ij} \leq cf^2$, where $c > 0$ is a constant independent from J, ϵ, and f.*

Proof. Consider $(J+\epsilon)^2$. This product can be written as $J^2 + J\epsilon + \epsilon J + \epsilon^2$. The identities $J^2 = J$ and $J\epsilon = 0$ hold. Furthermore, the sum $J + \epsilon J$ is a rank 1 column stochastic matrix. Thus the product $(J+\epsilon)^2$ can be written as the sum of a rank 1 column stochastic matrix and ϵ^2. It is easy to show that $\max_{i,j} |\epsilon^2|_{ij} \leq nf^2$, which is of order f^2.

Now consider the result of applying Γ_2 to $J+\epsilon$, and compare this with $\Gamma_2 J$. First compute the renormalization weight for the l^{th} column of $\Gamma_2(J+\epsilon)$. This equals $\sum_i (J_{il} + \epsilon_{il})^2$. Split this sum into the parts $\sum_i J_{il}{}^2$, $2\sum_i \epsilon_{il} J_{il}$, and $\sum_i \epsilon_{il}{}^2$. Then $2|\sum_i \epsilon_{il} J_{il}| \leq 2f$, and $\sum_i \epsilon_{il}{}^2 \leq nf^2$. It follows that $\sum_i (J_{il} + \epsilon_{il})^2$ can be written as $\sum_i J_{il}{}^2 + \delta_d$, where $|\delta_d| \leq 2f + nf^2$ (and the d stands for denominator).

Observe that $(J_{kl} + \epsilon_{kl})^2 = J_{kl}{}^2 + 2J_{kl}\epsilon_{kl} + \epsilon_{kl}{}^2$ can be written as $J_{kl}{}^2 + \delta_e$, where $|\delta_e| \leq 2f + f^2$. It follows that $[\Gamma_2(J+\epsilon)]_{kl}$ can be estimated as below.

$$\frac{J_{kl} - \delta_e}{\sum_i J_{il}{}^2 + \delta_d} \quad\leq\quad \frac{(J_{kl} + \epsilon_{kl})^2}{\sum_i (J_{il} + \epsilon_{il})^2} \quad\leq\quad \frac{J_{kl} + \delta_e}{\sum_i J_{il}{}^2 - \delta_d}$$

Now let a/b be a positive fraction less than or equal to one, let x and y be real numbers. Observe that

$$\begin{aligned}
\frac{a-x}{b+y} &= \frac{a}{b} - \frac{x + ay/b}{b+y} \geq \frac{a}{b} - \frac{|x| + |y|}{b+y} \\
\frac{a+x}{b-y} &= \frac{a}{b} + \frac{x + ay/b}{b-y} \leq \frac{a}{b} + \frac{|x| + |y|}{b-y}
\end{aligned}$$

Finally,

$$[\Gamma_2 J]_{kl} - \frac{|\delta_e| + |\delta_d|}{\sum_i J_{il}{}^2 + |\delta_d|} \quad\leq\quad [\Gamma_2(J+\epsilon)]_{kl} \quad\leq\quad [\Gamma_2 J]_{kl} + \frac{|\delta_e| + |\delta_d|}{\sum_i J_{il}{}^2 - |\delta_d|}$$

Since $\sum_i J_{il}{}^2 \geq 1/n$ it follows that the difference $|[\Gamma_2(J+\epsilon)]_{kl} - [\Gamma_2 J]_{kl}|$ can be bounded by cf, where $c > 0$ is a constant depending on n only. This, combined with the result on $(J+\epsilon)^2$ proves the lemma. □

Remark. An alternative proof this lemma is given in [11] using results on the Hilbert distance between positive vectors. In this setting the proof simplifies considerably.

Remark. For the proof of Theorem 5 one needs also consider the behaviour of columns in M, the associated nodes of which are not attractors. It is an easy exercise to show that such columns exhibit the same behaviour as the columns of the attractor systems to which they are attracted. This concludes a series of results on the stability and instability of the equilibrium states in L_1 in both the usual and a macroscopic sense.

The combined results of Theorem 4 and 5 indicate that perturbations of M may only disturb the phenomenon of overlap, which is inherently instable. Intuitively, it is clear that otherwise the clustering associated with an idempotent matrix must be stable under small perturbations. This is because the submatrices corresponding with attractor systems are effectively the only part of the matrix that may affect the associated clustering; the columns of nodes that are attracted to such a system must follow suit (the distribution of such a column c in the powers of M is forced to converge towards the distribution of the corresponding attractor submatrix, no matter how c is perturbed itself). The only thing lacking here is a proof that if the set of columns of M corresponding with an entire attractor system is perturbed, than the same set of columns must have rank 1 in the limit of the powers of the perturbed matrix.

In [10] experimental results are discussed concerning the phenomena of overlap and attractor systems. Current evidence suggests that these phenomena imply the existence of automorphisms of the input graph. Generally, the MCL process converges so fast that idempotency can be recognized long before instability of overlap and attractor systems begin to play a role. This is related to the fact that the examples given here concern small graphs. However, the crucial property is that the natural cluster diameter is small. Thus, large graphs G for which the natural cluster diameter is small may also lead the MCL process $(\mathcal{T}_G, e_{(i)}, r_{(i)})$ to converge towards idempotency before instability starts to play a role. Finally, by using the results in [11] overlap can be detected at early stages. The primary use of the MCL process lies in detecting cluster structure however, and the observed correspondence between graph automorphisms and respectively cluster overlap and attractor systems does not seem particularly useful for detection of the latter two.

14. Scaling the MCL algorithm

The complexity of the MCL algorithm, if nothing special is done, is $\mathcal{O}(N^3)$ where N is the number of nodes of the input graph. The factor N^3 corresponds to the cost of one matrix multiplication on two matrices of dimension N. The inflation step can be done in $\mathcal{O}(N^2)$ time. I will leave the issue aside here of how many steps are required before the algorithm converges to a doubly idempotent matrix. In practice, this number lies typically somewhere between 10 and 100, but only a small number of steps (in a corresponding range of approximately 3 to 10) in the beginning correspond with matrix iterands that are not extremely sparse. The only way to cut down the complexity of the algorithm is to keep the matrices sparse. Fortunately, the MCL process is by its very nature susceptible to such modification. This issue is discussed below, followed by a brief description of the MCL implementation in use at the CWI.

Complexity and scalability

The limits of an MCL process are in general extremely sparse. All current evidence suggests that overlap or attractor systems of cardinality greater than one correspond with certain automorphisms of the input graph [10].

The working of the MCL process with respect to finding cluster structure is mainly based on two phenomena. First, the disappearance of flow on edges between sparsely connected dense regions, in particular the edges in the input graph. Second, the creation of new flow within dense regions, corresponding with edges in the limit graph not existing in the input graph.

Typically, the average number of nonzero elements in a column of a limit matrix is equal to or very close to one, and the intermediate iterands are sparse *in a weighted sense*. The expansion operator causes successive iterands to fill very rapidly, but if natural cluster structure is present and the cluster diameters are not too large (cf. [10]) then the inflation operator ensures that the majority of the matrix entries stays very small, and that for each column the deviation in the size of its entries is large. A small cluster diameter implies that the equalizing of probability distributions is relatively easy as flow need not be transferred over long distances before it eventually stabilizes. This fact is exploited in various proposals for matrix pruning schemes made below.

REMARK. Before introducing these schemes a remark on the justification of pruning is in place. I will not attempt a numerical or perturbation analysis of pruning. Rather I will stick to heuristic reasoning in higher-level terms of cluster structure and random walks when discussing the viability of pruning. This is put to the test by experimenting with randomly generated testgraphs in [10].

PRUNING SCHEMES. If it is assumed that the probabilities of intermediate random walks are indeed distributed inhomogeneously per column, then this leads naturally to the idea that it will do no harm to remove intermediate random walks (i.e. setting matrix entries to zero) which have very small probability. The interpretation of the process then enforces obvious constraints on such pruning:

- The magnitude of a transition probability is only relevant in relationship to the other transition probabilities of the associated tail node. Pruning must be done locally rather than globally, that is, column-wise.
- Pruning should only remove a small part of the overall weight of a column; the corresponding entries should ideally have large (downward) deviation from the column average (for a suitable notion of column average).
- In order to maintain the stochastic interpretation, columns are rescaled after pruning.

Together these form the the key to an efficient implementation of the *MCL* algorithm. Three different pruning schemes have been considered and implemented. Let M be a sparse column stochastic matrix. Suppose a column c of the square M^2 has been computed with full precision. The three schemes are respectively:

- Exact pruning — the k largest entries of the column are computed. Ties are broken arbitrarily or are allowed to increase the bound k. This computation becomes increasingly expensive for larger values of k and increasing deviation between k and the number of nonzero entries of c.
- Threshold pruning — a threshold value f is computed in terms of the mass centre $\mathrm{ctr}(c)$ of order two of c. All values greater than f are kept, the rest is discarded. A typical candidate for such a threshold value is of the form $a\,\mathrm{ctr}(c)(1 - b[\max_i(c_i) - \mathrm{ctr}(c)])$, where $0 < a \leq 1$ and b is chosen in the range $1 \ldots 8$; another one is $a[\mathrm{ctr}(c)]^b$, where $0 < a \leq 1 \leq b$. The motivation for the first depends on the fact that if $\max_i(c_i)$ is close to $\mathrm{ctr}(c)$ then the (large) nonzero entries of the vector c are rather homogeneously distributed.
- A combination of the above, where threshold pruning is applied first in order to lower the cost of exact pruning. It is either allowed or disallowed for threshold pruning to leave a number of nonzero entries smaller than k.

If pruning with pruning constant k is incorporated into the algorithm, the complexity is reduced to $\mathcal{O}(Nk^2)$ for a single matrix multiplication. This follows from the fact that any column of the product of two k-pruned matrices has at most k^2 nonzero entries. It is assumed that pruning can be done in $\mathcal{O}(t)$ time for a vector with t nonzero entries. In the experiments in [10] this was ensured by using threshold pruning.

FACTORS AFFECTING THE VIABILITY OF PRUNING. It is intuitively acceptable that pruning eats away the least probable walks, if they have large downward deviation from the column centre, and if the total number of pruned entries accounts for a relatively small percentage of the column mass, say somewhere in between 5 and 10 percent. If the distribution of a column c is rather homogeneous, with many entries approximately equal to the centre $\mathrm{ctr}(c)$, and if pruning removes a sizeable fraction of the distribution, this will clearly disturb the *MCL* algorithm, rather than perturb. The examples in [10] indicate that the latter will be the case if the diameter of the natural clusters is large and if the subgraphs induced by the clusters are very homogeneous.

CONVERGENCE IN THE PRESENCE OF PRUNING. The convergence properties in the setting sketched above do not change noticeably, and the resulting clusterings are still very satisfactory. Clusterings of graphs with up to a thousand nodes resulting from both normal matrix computation and prune mode with otherwise identical parametrizations were compared. The respective clusterings sometimes differed slightly (e.g. a node moving from one cluster to another) and were often identical. The effect of varying the pruning parameter is investigated quantitatively in [10].

An example of pruning is given in Figure 14. The equilibrium state and several matrix iterands are given for the *MCL* process with input graph G_3, and pruning constant $k = 5$. The clustering resulting from this pruned process is the same as the clustering resulting from the unperturbed process.

MCL implementation

The *MCL* algorithm was implemented at the *CWI* by the author. It is part of a library written in C with extensive support for matrix operations, mapping of matrices onto clusterings, comparison of clusterings, generation of statistics (e.g. for different pruning schemes), and facilities for random generation of partitions and cluster test matrices. Both Jan van der Steen and Annius Groenink have contributed significantly to the matrix section of the library in terms of rigor and elegance. The library will be made available under a public license.

At the heart of the library lies the data structure implementing a matrix. A matrix is represented as an ordered array of vectors, and a vector is represented as an array of index/value pairs. Each index is unique in the array, and the index/value pairs are ordered on increasing index. This generic construction is used to represent a nonnegative vector by its positive entries only. The vector $(4.2, 0.0, 2.7, 3.1, 0.0, 0.0, 5.6)^T$ is thus represented as the array (indexing starts at zero)

$$[0|4.2][2|2.7][3|3.1][6|5.6]$$

There is a choice of representing a matrix via its rows or its columns. A column stochastic matrix M is naturally represented via its columns. Assuming that pruning is applied with pruning constant k, computing the square M^2 requires for each column of M^2 the computation of a weighted sum of at most k columns, resulting in a vector which may have k^2 entries. This vector is pruned down to at most k entries via either of the schemes given above. For large k, say larger than 70, it is pertinent that threshold pruning is applied in order to ease the burden of exact pruning. This may lead to a pruned vector with less than k entries. It is easy to envision a looping process in which several thresholds are tried in order to obtain an optimum threshold value resulting in a vector with a number of entries close or even to k, or even a version of threshold pruning where the pruning regime depends on the weight distribution of the probability vector, so that nodes with a large homogeneous distribution are allowed to have more than k nodes. This was not tried for, but the experiments in [10] indicate that fine-tuning the pruning regime may result in considerably better performance.

REFERENCES

1. ACM/IEEE, editor. *Proceedings of the* 26^{th} *ACM/IEEE Design Automation Conference*. IEEE, June 1993.

2. ACM/SIAM, editor. *Proceedings of the fourth annual ACM–SIAM symposium on discrete algorithms*. ACM, January 1993.

3. Charles J. Alpert and Andrew B. Kahng. Recent directions in netlist partitioning: a survey. *Integration: the VLSI Journal*, 19(1–2):1–81, 1995.

$$\begin{pmatrix}
0.405 & 0.094 & -- & -- & 0.098 & 0.295 & 0.223 & -- & -- & 0.323 & -- & -- \\
-- & 0.376 & 0.228 & -- & 0.192 & 0.019 & -- & -- & -- & -- & -- & -- \\
-- & 0.228 & 0.376 & -- & 0.192 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & 0.094 & 0.365 & -- & -- & -- & 0.219 & 0.146 & -- & 0.146 & 0.083 \\
0.061 & 0.228 & 0.228 & -- & 0.518 & -- & 0.092 & -- & -- & 0.019 & -- & -- \\
0.151 & -- & -- & -- & -- & 0.295 & 0.092 & -- & -- & 0.186 & -- & -- \\
0.120 & 0.074 & -- & -- & -- & 0.097 & 0.369 & -- & -- & 0.148 & -- & -- \\
-- & -- & 0.074 & 0.212 & -- & -- & -- & 0.342 & 0.146 & -- & 0.146 & 0.083 \\
-- & -- & -- & 0.212 & -- & -- & -- & 0.219 & 0.293 & -- & 0.293 & 0.278 \\
0.262 & -- & -- & -- & -- & 0.295 & 0.223 & -- & -- & 0.323 & -- & -- \\
-- & -- & -- & 0.212 & -- & -- & -- & 0.219 & 0.293 & -- & 0.293 & 0.278 \\
-- & -- & -- & -- & -- & -- & -- & -- & 0.123 & -- & 0.123 & 0.278
\end{pmatrix}$$

$\Gamma_2 M^2$ for MCL process with pruning.

$$\begin{pmatrix}
0.464 & 0.066 & -- & -- & 0.050 & 0.431 & 0.381 & -- & -- & 0.437 & -- & -- \\
-- & 0.294 & 0.256 & -- & 0.196 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.242 & 0.312 & -- & 0.196 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & 0.041 & 0.255 & -- & -- & -- & 0.211 & 0.149 & -- & 0.149 & 0.106 \\
0.023 & 0.375 & 0.361 & -- & 0.554 & 0.006 & 0.048 & -- & -- & 0.010 & -- & -- \\
0.119 & -- & -- & -- & -- & 0.155 & 0.090 & -- & -- & 0.130 & -- & -- \\
0.094 & 0.023 & -- & -- & 0.003 & 0.086 & 0.203 & -- & -- & 0.105 & -- & -- \\
-- & -- & 0.030 & 0.196 & -- & -- & -- & 0.228 & 0.142 & -- & 0.142 & 0.103 \\
-- & -- & -- & 0.269 & -- & -- & -- & 0.274 & 0.329 & -- & 0.329 & 0.347 \\
0.300 & -- & -- & -- & -- & 0.323 & 0.278 & -- & -- & 0.318 & -- & -- \\
-- & -- & -- & 0.269 & -- & -- & -- & 0.274 & 0.329 & -- & 0.329 & 0.347 \\
-- & -- & -- & 0.012 & -- & -- & -- & 0.013 & 0.051 & -- & 0.051 & 0.097
\end{pmatrix}$$

$\Gamma_2(\Gamma_2(M^2)\cdot\Gamma_2(M^2))$, for MCL process with pruning.

$$\begin{pmatrix}
0.619 & 0.022 & 0.004 & -- & 0.014 & 0.617 & 0.609 & -- & -- & 0.617 & -- & -- \\
-- & 0.182 & 0.187 & -- & 0.155 & -- & -- & -- & -- & -- & -- & -- \\
-- & 0.178 & 0.193 & -- & 0.155 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & 0.003 & 0.141 & -- & -- & -- & 0.137 & 0.116 & -- & 0.116 & 0.104 \\
0.003 & 0.615 & 0.612 & -- & 0.675 & 0.001 & 0.008 & -- & -- & 0.002 & -- & -- \\
0.048 & -- & -- & -- & -- & 0.050 & 0.045 & -- & -- & 0.049 & -- & -- \\
0.036 & -- & -- & -- & -- & 0.035 & 0.046 & -- & -- & 0.037 & -- & -- \\
-- & -- & -- & 0.119 & -- & -- & -- & 0.119 & 0.102 & -- & 0.102 & 0.092 \\
-- & -- & -- & 0.368 & -- & -- & -- & 0.370 & 0.388 & -- & 0.388 & 0.398 \\
0.294 & 0.003 & -- & -- & 0.001 & 0.297 & 0.292 & -- & -- & 0.296 & -- & -- \\
-- & -- & -- & 0.368 & -- & -- & -- & 0.370 & 0.388 & -- & 0.388 & 0.398 \\
-- & -- & -- & 0.005 & -- & -- & -- & 0.005 & 0.007 & -- & 0.007 & 0.009
\end{pmatrix}$$

$(\Gamma_2 \circ Squaring)$ iterated three times on M for MCL process with pruning

$$\begin{pmatrix}
1.000 & -- & -- & -- & -- & 1.000 & 1.000 & -- & -- & 1.000 & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & 1.000 & 1.000 & -- & 1.000 & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- \\
-- & -- & -- & 0.500 & -- & -- & -- & 0.500 & 0.500 & -- & 0.500 & 0.500 \\
-- & -- & -- & -- & -- & -- & -- & -- & -- & -- & -- & --
\end{pmatrix}$$

M_{mcl}^{∞}

Figure 14: Iteration of $(\Gamma_2 \circ Squaring)$ with initial iterand M defined in Figure 8. Pruning with pruning constant $k = 5$ is applied throughout the process.

4. M.R. Anderberg. *Cluster analysis for Applications.* Academic Press, London, 1973.
5. Abraham Berman and Robert J. Plemmons. *Nonnegative Matrices In The Mathematical Sciences.* Number 9 in Classics in Applied Mathematics. SIAM, 1994. Corrected and extended republication of the 1979 book.
6. T. Bui, S. Chaudhuri, T. Leighton, and M. Sipser. Graph bisection algorithms with good average case behavior. *Combinatorica*, 7(2):171–191, 1987.
7. T. Bui, C.Heigham, C. Jones, and T. Leighton. Improving the performance of the Kernighan–Lin and simulated annealing graph bisection algorithms. In ACM/IEEE [1], pages 775–778.
8. Stijn van Dongen. A new cluster algorithm for graphs. Technical Report INS–R9814, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, December 1998.
9. Stijn van Dongen. *Graph Clustering by Flow Simulation.* PhD thesis, University of Utrecht, May 2000.
10. Stijn van Dongen. Performance criteria for graph clustering and Markov cluster experiments. Technical report, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, 2000. To appear.
11. Stijn van Dongen. A stochastic uncoupling process for graphs. Technical report, National Research Institute for Mathematics and Computer Science in the Netherlands, Amsterdam, 2000. To appear.
12. R.O. Duda and P.E. Hart. *Pattern classification and scene analysis.* Wiley, 1973.
13. Brian S. Everitt. *Cluster Analysis.* Hodder & Stoughton, third edition, 1993.
14. J. Garbers, H.J. Promel, and A. Steger. Finding clusters in VLSI circuits. In IEEE [20], pages 520–523.
15. Lars Hagen and Andrew B. Kahng. A new approach to effective circuit clustering. In IEEE [21], pages 422–427.
16. J.A. Hartigan. *Clustering Algorithms.* Wiley, New York, 1975.
17. Michiel Hazewinkel. Classification in mathematics, discrete metric spaces, and approximation by trees. *Nieuw Archief voor Wiskunde*, 13(3):325–361, 1995.
18. Roger A. Horn and Charles R. Johnson. *Matrix Analysis.* Cambridge University Press, 1990.
19. Lawrence J. Hubert. Some applications of graph theory to clustering. *Psychometrika*, 39(3):283–309, 1974.
20. IEEE, editor. *Proceedings IEEE International Conference on Computer–Aided Design.* IEEE, 1990.
21. IEEE, editor. *Proceedings of the IEEE international Conference on Computer–Aided Design.* IEEE, November 1992.
22. Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data.* Prentice Hall, 1988.
23. Nicholas Jardine and Robin Sibson. *Mathematical Taxonomy.* Wiley Series In Probabilistic And Mathematical Statistics. John Wiley & Sons, 1971.
24. David R. Karger. Global min–cuts in rnc, and other ramifications of a simple min–cut algorithm. In ACM/SIAM [2], pages 21–30.
25. Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data.* John Wiley & Sons, 1983.
26. L. Lovász. Random walks on graphs: A survey. In Miklos et al. [33], pages 353–397.
27. Albert W. Marshall and Ingram Olkin. *Inequalities: Theory of Majorization and Its Applications.* Number 143 in Mathematics In Science And Engineering. Academic Press, 1979.
28. Albert W. Marshall, Ingram Olkin, and Frank Proschan. Monotonicity of ratios of means and other applications of majorization. In Shisha [37], pages 177–197. Wright–Patterson Air Force Base, Ohio, August 19–27.
29. D. Luc Massart and Leonard Kaufman. *The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis.* Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, 1990.
30. David W. Matula. k–Components, clusters and slicings in graphs. *SIAM Journal on Applied Mathematics*, 22:459–480, 1972.

31. David W. Matula. Graph theoretic techniques for cluster analysis algorithms. In Van Ryzin [42], pages 95–129.

32. David W. Matula and Leland L. Beck. Smallest–last ordering and clustering and graph coloring algorithms. Technical Report CSE 8104, Southern Methodist University School of Engineering and Applied Science, July 1981.

33. D. Miklos et al., editors. *Combinatorics, Paul Erdős is eighty*, volume II. Janos Bolyai Mathematical Society, 1996.

34. Boris Mirkin. *Mathematical Classification and Clustering.* Kluwer Academic Publishers, 1996.

35. E. Seneta. *Non–negative matrices and Markov chains.* Springer, second edition, 1981.

36. Farhad Shahrokhi and D.W. Matula. The maximum concurrent flow problem. *Journal of the Association of Computing Machinery*, 37(2):318–334, 1990.

37. Oved Shisha, editor. *Inequalities.* Academic Press, 1965. Wright–Patterson Air Force Base, Ohio, August 19–27.

38. P.H.A. Sneath and R.R. Sokal. *Numerical Taxonomy.* W.H. Freeman, San Fransisco, 1973.

39. Shinichi Tamura. Clustering based on multiple paths. *Pattern Recognition*, 15(6):477–483, 1982.

40. Dolenc Tomi. k–Connectivity and the overlapping stratified clustering algorithm. In Tosic et al. [41], pages 63–75.

41. R. Tosic et al., editors. *Graph Theoy, Proceedings* 8^{th} *Yugoslavian Seminar, Novi Sad, Yugoslavia, 1987.* University of Novi Sad, 1989.

42. J. Van Ryzin, editor. *Classification and Clustering. Proceedings of an Advanced Seminar Conducted by the Mathematics Research Center, The University of Wisconsin at Madison, May 3–5, 1976*, New York, 1977. Academic Press.

43. Ching-Wei Yeh, Chung-Kuan Cheng, and Ting-Ting Y. Lin. Circuit clustering using a stochastic flow injection method. *IEEE Transactions on Computer–Aided Design of Integrated Circuits and Systems*, 14(2):154–162, 1995.